A clique graph based merging strategy for decomposable SDPs

Michael Garstka[1] · Mark Cannon[1] · Paul Goulart[1]

[1]University of Oxford, UK

(virtual) Journal Club, Control Group
4th December 2020

Semidefinite programming

- Given matrices $C, A_1, \ldots, A_m \in \mathbb{S}^n$ and $b \in \mathbb{R}^m$, find X:

$$\begin{array}{ll} \text{minimize} & \langle C, X \rangle \\ \text{subject to} & \langle A_i, X \rangle = b_i,\ i = 1, \ldots, m \\ & X \in \mathbb{S}^n_+ \end{array}$$

$$\begin{array}{ll} \text{maximize} & b^\top \nu \\ \text{subject to} & A^\top \nu + Y = C \\ & Y \in \mathbb{S}^n_+. \end{array}$$

Semidefinite programming

- Given matrices $C, A_1, \ldots, A_m \in \mathbb{S}^n$ and $b \in \mathbb{R}^m$, find X:

$$\begin{array}{ll} \text{minimize} & \langle C, X \rangle \\ \text{subject to} & \langle A_i, X \rangle = b_i,\ i = 1, \ldots, m \\ & X \in \mathbb{S}^n_+ \end{array}$$

$\mathcal{O}(n^3)$

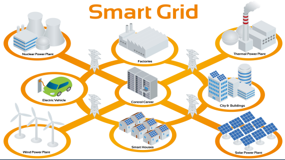
$$\begin{array}{ll} \text{maximize} & b^\top \nu \\ \text{subject to} & A^\top \nu + Y = C \\ & Y \in \mathbb{S}^n_+. \end{array}$$

$n = 10^4 \rightarrow \mathcal{O}(10^{12})$ operations at each step

Semidefinite programming

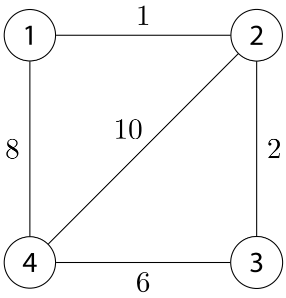
Where do we find positive semidefinite matrices?

- Lyapunov functions
- Linear Matrix Inequalities / S-procedure [Martin S Andersen et al. 2014]
- Kernel matrices [Lanckriet et al. 2004]
- Covariance matrices [Bertsimas and Nino-Mora 1999]
- Graph Laplacian
- Sum-of-Squares [Lasserre 2009]
- Semidefinite relaxation of
  - cardinality constraints (sparse PCA) [d'Aspremont et al. 2004]
  - QCQPs
  - mixed-integer constraints [Goemans and Williamson 1995]



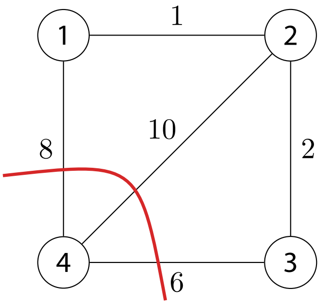
The MAXCUT problem

- Weighted graph $G(V, E)$ with weights $w_{ij} \geq 0$, find $S \subset V$ such that the edge weights between S and $\bar{S} = V \setminus S$ are maximized



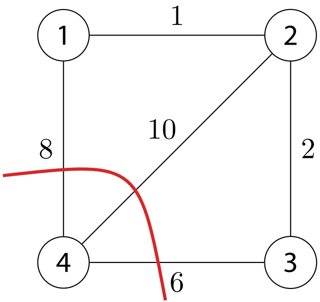
The MAXCUT problem

- Weighted graph $G(V, E)$ with weights $w_{ij} \geq 0$, find $S \subset V$ such that the edge weights between S and $\bar{S} = V \setminus S$ are maximized



The MAXCUT problem

- Weighted graph $G(V, E)$ with weights $w_{ij} \geq 0$, find $S \subset V$ such that the edge weights between S and $\bar{S} = V \setminus S$ are maximized



$$\begin{array}{ll} \text{maximize} & \frac{1}{4} \sum_{i=1}^{n} \sum_{j=1}^{n} w_{ij}(1 - y_i y_j) \\ \text{subject to} & y_i \in \{-1, 1\}, \ \forall i \in V \end{array}$$

- NP-hard, part of Karp's 21 NP-complete problems [Karp 1972]
- best approximation until 1995: $0.5p^*$

The MAXCUT problem: A semidefinite relaxation

- SDP relaxation that guarantees $0.87856\, p^*$ [Goemans and Williamson 1995]

$$\begin{array}{ll} \text{maximize} & \frac{1}{4}\sum_{i=1}^{n}\sum_{j=1}^{n} w_{ij}(1 - y_i y_j) \\ \text{subject to} & y_i \in \{-1, 1\},\ \forall i \in V \end{array}$$

The MAXCUT problem: A semidefinite relaxation

- SDP relaxation that guarantees $0.87856\, p^*$ [Goemans and Williamson 1995]

$$\begin{aligned} \text{maximize} \quad & \frac{1}{4}\sum_{i=1}^{n}\sum_{j=1}^{n} w_{ij}(1 - y_i y_j) \\ \text{subject to} \quad & y_i \in \{-1, 1\},\ \forall i \in V \end{aligned}$$

→ Lifting: $X = yy^\top$

The MAXCUT problem: A semidefinite relaxation

- SDP relaxation that guarantees $0.87856\, p^*$ [Goemans and Williamson 1995]

$$\begin{array}{ll} \text{maximize} & \frac{1}{4}\sum_{i=1}^{n}\sum_{j=1}^{n} w_{ij}(1 - y_i y_j) \\ \text{subject to} & y_i \in \{-1, 1\},\ \forall i \in V \end{array}$$

Lifting: $X = yy^\top$

$$\begin{array}{ll} \text{maximize} & \frac{1}{4}\sum_{i=1}^{n}\sum_{j=1}^{n} w_{ij}(1 - X_{ij}) \\ \text{subject to} & X_{ii} = 1,\ i = 1, \dots, n \\ & X = yy^\top \end{array}$$

The MAXCUT problem: A semidefinite relaxation

- SDP relaxation that guarantees $0.87856\,p^*$ [Goemans and Williamson 1995]

$$\begin{array}{ll} \text{maximize} & \frac{1}{4}\sum_{i=1}^{n}\sum_{j=1}^{n} w_{ij}(1 - y_i y_j) \\ \text{subject to} & y_i \in \{-1, 1\},\ \forall i \in V \end{array}$$

$$\begin{array}{ll} \text{maximize} & \frac{1}{4}\sum_{i=1}^{n}\sum_{j=1}^{n} w_{ij}(1 - X_{ij}) \\ \text{subject to} & X_{ii} = 1,\ i = 1, \dots, n \\ & \sout{X = yy^\top}\ X \succeq 0 \end{array}$$

Lifting: $X = yy^\top$

The MAXCUT problem: A semidefinite relaxation

- SDP relaxation that guarantees $0.87856\, p^*$ [Goemans and Williamson 1995]

maximize $\frac{1}{4}\sum_{i=1}^{n}\sum_{j=1}^{n} w_{ij}(1 - y_i y_j)$

subject to $y_i \in \{-1, 1\}, \; \forall i \in V$

Lifting: $X = yy^\top$

maximize $\frac{1}{4}\sum_{i=1}^{n}\sum_{j=1}^{n} w_{ij}(1 - X_{ij})$

subject to $X_{ii} = 1, \; i = 1, \dots, n$

~~$X = yy^\top$~~ $X \succeq 0$

Primal SDP

maximize $\frac{1}{4}\langle L, X \rangle$

subject to $X_{ii} = 1, \; i = 1, \dots, n$

$X \succeq 0$

The MAXCUT problem: A semidefinite relaxation

- SDP relaxation that guarantees $0.87856\, p^*$ [Goemans and Williamson 1995]

$$\begin{array}{ll} \text{maximize} & \frac{1}{4}\sum_{i=1}^{n}\sum_{j=1}^{n} w_{ij}(1 - y_i y_j) \\ \text{subject to} & y_i \in \{-1, 1\},\ \forall i \in V \end{array}$$

Lifting: $X = yy^\top$

$$\begin{array}{ll} \text{maximize} & \frac{1}{4}\sum_{i=1}^{n}\sum_{j=1}^{n} w_{ij}(1 - X_{ij}) \\ \text{subject to} & X_{ii} = 1,\ i = 1, \dots, n \\ & \sout{X = yy^\top}\ X \succeq 0 \end{array}$$

Primal SDP

$$\begin{array}{ll} \text{maximize} & \frac{1}{4}\langle L, X\rangle \\ \text{subject to} & X_{ii} = 1,\ i = 1, \dots, n \\ & X \succeq 0 \end{array}$$

Dual SDP

$$\begin{array}{ll} \text{minimize} & \sum_i \nu_i \\ \text{subject to} & Y = \text{diag}(\nu) - \frac{1}{4}L \\ & Y \succeq 0 \end{array}$$

The MAXCUT problem: A semidefinite relaxation

- SDP relaxation that guarantees $0.87856\, p^*$ [Goemans and Williamson 1995]

maximize $\frac{1}{4}\sum_{i=1}^{n}\sum_{j=1}^{n} w_{ij}(1 - y_i y_j)$

subject to $y_i \in \{-1, 1\},\ \forall i \in V$

Lifting: $X = yy^\top$

maximize $\frac{1}{4}\sum_{i=1}^{n}\sum_{j=1}^{n} w_{ij}(1 - X_{ij})$

subject to $X_{ii} = 1,\ i = 1, \dots, n$

~~$X = yy^\top$~~ $X \succeq 0$

Primal SDP

maximize $\frac{1}{4}\langle L, X\rangle$

subject to $X_{ii} = 1,\ i = 1, \dots, n$

$X \succeq 0$

Dual SDP

minimize $\sum_i \nu_i$

subject to $Y = \text{diag}(\nu) - \frac{1}{4}L$

$Y \succeq 0$

Overview

Matrix Sparsity and Graphs

Chordal decomposition

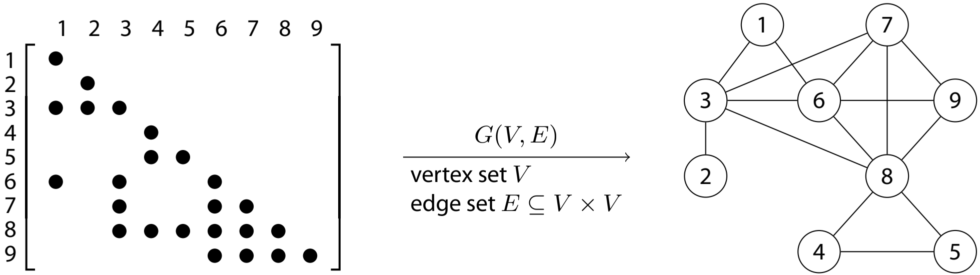
Clique merging

- Clique tree-based merging strategies
- Clique graph-based merging strategy

Benchmarks

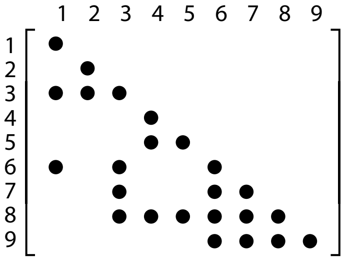
Conclusion

Matrix sparsity and graphs

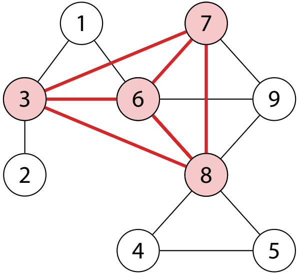


*Survey paper [L. Vandenberghe and M. S. Andersen 2015]

Matrix sparsity and graphs

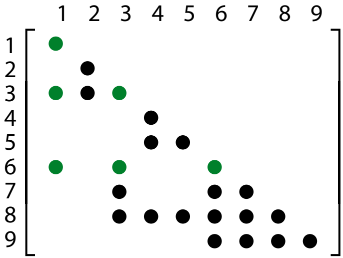


Complete subgraph

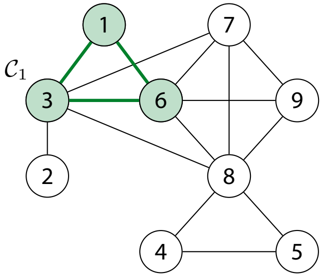


*Survey paper [L. Vandenberghe and M. S. Andersen 2015]

Matrix sparsity and graphs

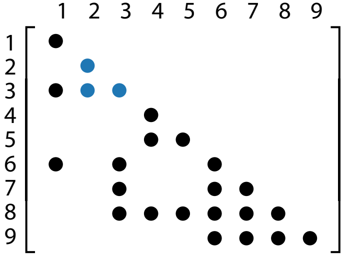


Clique 1

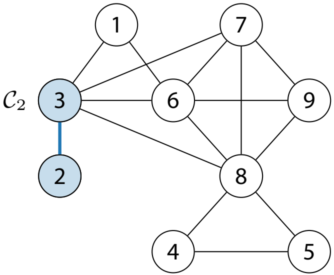


*Survey paper [L. Vandenberghe and M. S. Andersen 2015]

Matrix sparsity and graphs

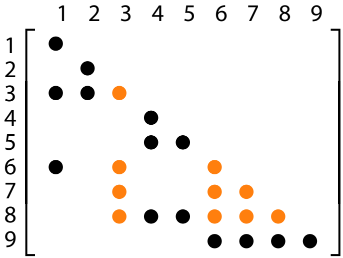


Clique 2

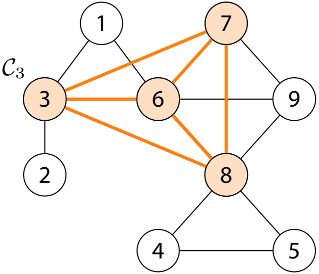


*Survey paper [L. Vandenberghe and M. S. Andersen 2015]

Matrix sparsity and graphs

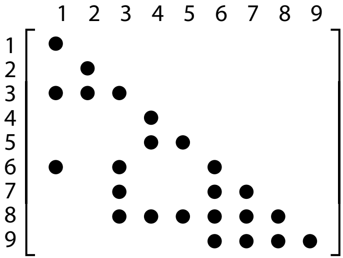


Clique 3

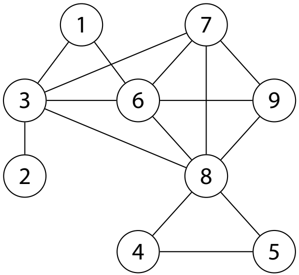


*Survey paper [L. Vandenberghe and M. S. Andersen 2015]

Matrix sparsity and graphs

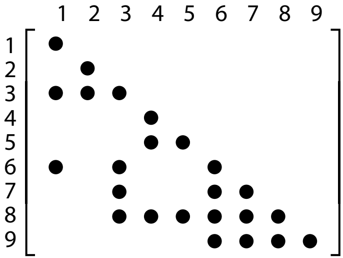


Chordal graph

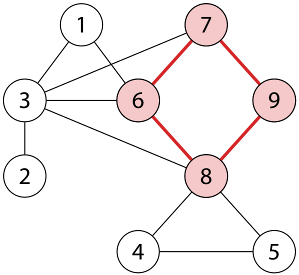


*Survey paper [L. Vandenberghe and M. S. Andersen 2015]

Matrix sparsity and graphs

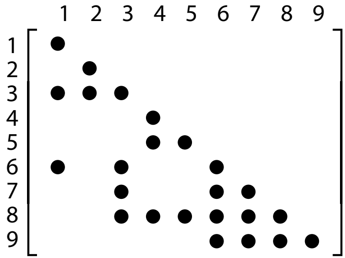


~~Chordal graph~~

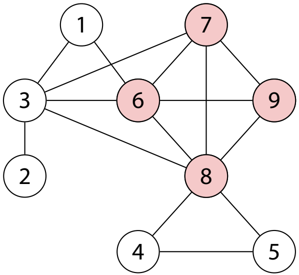


*Survey paper [L. Vandenberghe and M. S. Andersen 2015]

Matrix sparsity and graphs



Chordal graph



* Survey paper [L. Vandenberghe and M. S. Andersen 2015]

Chordal decomposition

Dual SDP*

maximize $b^\top y$

subject to $\sum_{i=1}^{m} A_i y_i + S = C$

$S \in \mathbb{S}_+^n$

Sparse matrix cones

$$\mathbb{S}^n(E,0) := \{S \in \mathbb{S}^n \mid S_{ij} = S_{ji} = 0, \text{ if } i \neq j, (i,j) \notin E\}$$
$$\mathbb{S}_+^n(E,0) := \{S \in \mathbb{S}^n(E,0) \mid S \succeq 0\}$$

*equivalent decomposition result for primal form SDPs exists

Chordal decomposition

Dual SDP*

$$\begin{array}{ll} \text{maximize} & b^\top y \\ \text{subject to} & \sum_{i=1}^{m} A_i y_i + S = C \\ & S \in \mathbb{S}^n_+(E, 0) \end{array}$$

Sparse matrix cones

$$\mathbb{S}^n(E,0) := \{S \in \mathbb{S}^n \mid S_{ij} = S_{ji} = 0, \text{ if } i \neq j,\ (i,j) \notin E\}$$
$$\mathbb{S}^n_+(E,0) := \{S \in \mathbb{S}^n(E,0) \mid S \succeq 0\}$$

*equivalent decomposition result for primal form SDPs exists

Agler's theorem

Agler's theorem*

Let $G(V,E)$ be a chordal graph with a set of maximal cliques $\{\mathcal{C}_1,\dots,\mathcal{C}_p\}$. Then $S \in \mathbb{S}^n_+(E,0)$ if and only if there exist matrices $S_\ell \in \mathbb{S}^{|\mathcal{C}_\ell|}_+$ for $\ell = 1,\dots,p$ such that

$$S = \sum_{\ell=1}^{p} T_\ell^\top S_\ell T_\ell.$$

*[Agler et al. 1988], Grone's theorem for primal form SDPs [Grone et al. 1984]

Example: Chordal Decomposition

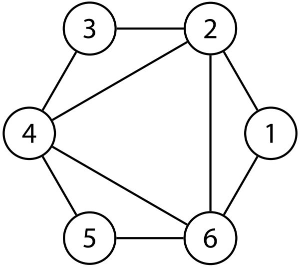
$$\begin{array}{ll} \text{maximize} & b^\top y \\ \text{subject to} & \sum_{i=1}^m A_i y_i + S = C \\ & S \in \mathbb{S}_+^n(E, 0) \end{array}$$

$$\begin{bmatrix} S_{11} & S_{12} & 0 & 0 & 0 & S_{16} \\ S_{21} & S_{22} & S_{23} & S_{24} & 0 & S_{26} \\ 0 & S_{32} & S_{33} & S_{34} & 0 & 0 \\ 0 & S_{42} & S_{43} & S_{44} & S_{45} & S_{46} \\ 0 & 0 & 0 & S_{54} & S_{55} & S_{56} \\ S_{61} & S_{62} & 0 & S_{64} & S_{65} & S_{66} \end{bmatrix}$$

Example: Chordal Decomposition

$$\begin{array}{ll} \text{maximize} & b^\top y \\ \text{subject to} & \sum_{i=1}^m A_i y_i + S = C \\ & S \in \mathbb{S}_+^n(E, 0) \end{array}$$

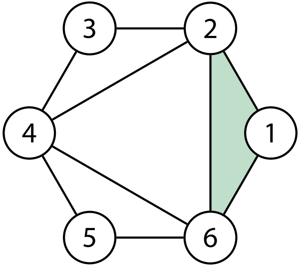
$$\begin{bmatrix} S_{11} & S_{12} & 0 & 0 & 0 & S_{16} \\ S_{21} & S_{22} & S_{23} & S_{24} & 0 & S_{26} \\ 0 & S_{32} & S_{33} & S_{34} & 0 & 0 \\ 0 & S_{42} & S_{43} & S_{44} & S_{45} & S_{46} \\ 0 & 0 & 0 & S_{54} & S_{55} & S_{56} \\ S_{61} & S_{62} & 0 & S_{64} & S_{65} & S_{66} \end{bmatrix}$$



Example: Chordal Decomposition

$$\begin{aligned} \text{maximize} \quad & b^\top y \\ \text{subject to} \quad & \textstyle\sum_{i=1}^m A_i y_i + S = C \\ & S \in \mathbb{S}_+^n(E, 0) \end{aligned}$$

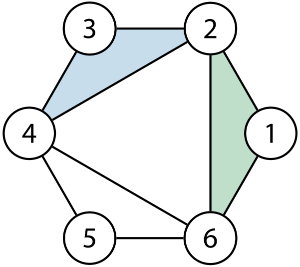
$$\begin{bmatrix} S_{11} & S_{12} & 0 & 0 & 0 & S_{16} \\ S_{21} & S_{22} & S_{23} & S_{24} & 0 & S_{26} \\ 0 & S_{32} & S_{33} & S_{34} & 0 & 0 \\ 0 & S_{42} & S_{43} & S_{44} & S_{45} & S_{46} \\ 0 & 0 & 0 & S_{54} & S_{55} & S_{56} \\ S_{61} & S_{62} & 0 & S_{64} & S_{65} & S_{66} \end{bmatrix}$$



Example: Chordal Decomposition

$$\begin{array}{ll} \text{maximize} & b^\top y \\ \text{subject to} & \sum_{i=1}^m A_i y_i + S = C \\ & S \in \mathbb{S}_+^n(E,0) \end{array}$$

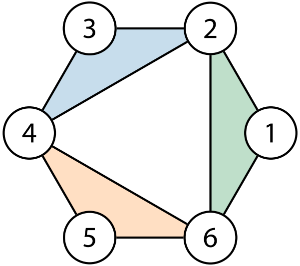
$$\begin{bmatrix} S_{11} & S_{12} & 0 & 0 & 0 & S_{16} \\ S_{21} & S_{22} & S_{23} & S_{24} & 0 & S_{26} \\ 0 & S_{32} & S_{33} & S_{34} & 0 & 0 \\ 0 & S_{42} & S_{43} & S_{44} & S_{45} & S_{46} \\ 0 & 0 & 0 & S_{54} & S_{55} & S_{56} \\ S_{61} & S_{62} & 0 & S_{64} & S_{65} & S_{66} \end{bmatrix}$$



Example: Chordal Decomposition

$$\begin{aligned} \text{maximize} \quad & b^\top y \\ \text{subject to} \quad & \textstyle\sum_{i=1}^m A_i y_i + S = C \\ & S \in \mathbb{S}_+^n(E,0) \end{aligned}$$

$$\begin{bmatrix} S_{11} & S_{12} & 0 & 0 & 0 & S_{16} \\ S_{21} & S_{22} & S_{23} & S_{24} & 0 & S_{26} \\ 0 & S_{32} & S_{33} & S_{34} & 0 & 0 \\ 0 & S_{42} & S_{43} & S_{44} & S_{45} & S_{46} \\ 0 & 0 & 0 & S_{54} & S_{55} & S_{56} \\ S_{61} & S_{62} & 0 & S_{64} & S_{65} & S_{66} \end{bmatrix}$$

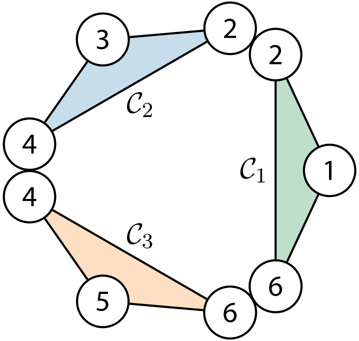


Example: Chordal Decomposition

$$\begin{array}{ll} \text{maximize} & b^\top y \\ \text{subject to} & \sum_{i=1}^m A_i y_i + S = C \\ & S \in \mathbb{S}^n_+(E,0) \end{array}$$

$$\begin{array}{ll} \text{maximize} & b^\top y \\ \text{subject to} & \sum_{i=1}^m A_i y_i + \sum_{\ell=1}^3 T_\ell^\top S_\ell T_\ell = C \\ & S_1 \in \mathbb{S}_+^{|\mathcal{C}_1|},\ S_2 \in \mathbb{S}_+^{|\mathcal{C}_2|},\ S_3 \in \mathbb{S}_+^{|\mathcal{C}_3|} \end{array}$$

$$\begin{bmatrix} S_{11} & S_{12} & 0 & 0 & 0 & S_{16} \\ S_{21} & S_{22} & S_{23} & S_{24} & 0 & S_{26} \\ 0 & S_{32} & S_{33} & S_{34} & 0 & 0 \\ 0 & S_{42} & S_{43} & S_{44} & S_{45} & S_{46} \\ 0 & 0 & 0 & S_{54} & S_{55} & S_{56} \\ S_{61} & S_{62} & 0 & S_{64} & S_{65} & S_{66} \end{bmatrix}$$

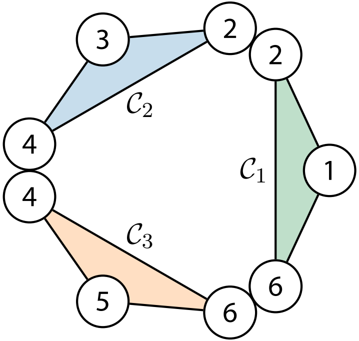


Example: Chordal Decomposition

$$\begin{array}{ll} \text{maximize} & b^\top y \\ \text{subject to} & \sum_{i=1}^m A_i y_i + S = C \\ & S \in \mathbb{S}_+^n(E, 0) \end{array}$$

$$\begin{array}{ll} \text{maximize} & b^\top y \\ \text{subject to} & \sum_{i=1}^m A_i y_i + \sum_{\ell=1}^3 T_\ell^\top S_\ell T_\ell = C \\ & S_1 \in \mathbb{S}_+^{|\mathcal{C}_1|},\ S_2 \in \mathbb{S}_+^{|\mathcal{C}_2|},\ S_3 \in \mathbb{S}_+^{|\mathcal{C}_3|} \end{array}$$

$$\begin{bmatrix} S_{11} & S_{12} & 0 & 0 & 0 & S_{16} \\ S_{21} & S_{22} & S_{23} & S_{24} & 0 & S_{26} \\ 0 & S_{32} & S_{33} & S_{34} & 0 & 0 \\ 0 & S_{42} & S_{43} & S_{44} & S_{45} & S_{46} \\ 0 & 0 & 0 & S_{54} & S_{55} & S_{56} \\ S_{61} & S_{62} & 0 & S_{64} & S_{65} & S_{66} \end{bmatrix}$$

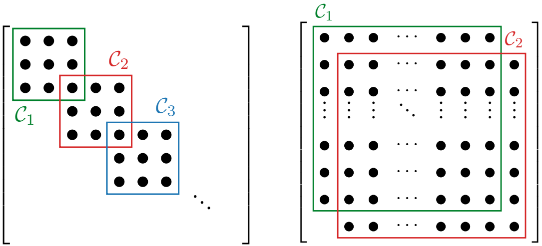


- Interior point method [Fukuda et al. 2001]
- First-order method [Sun, M. S. Andersen, and L. Vandenberghe 2014]
- ADMM-HSDE [Zheng et al. 2019]

Clique merging

- Combine cliques by introducing new edges in the graph
- One merge operation:
  - replaces two PSD constraints by one larger PSD constraint
  - removes equality constraints

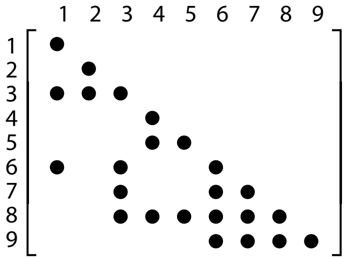
  $\rightarrow$ trade-off depends on the employed solver algorithm
- Obvious cases:

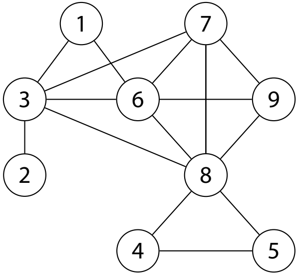


Algorithm: First-order solver

Factor constraint matrix;
while not converged:
[...]
Eigenvalue decomposition of PSD decision variables;

Clique tree-based merging strategies

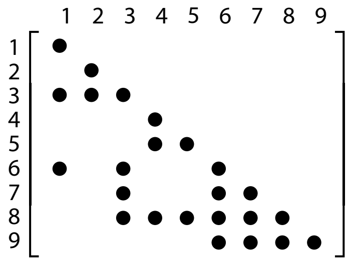


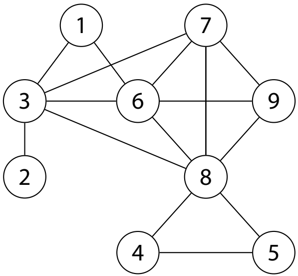


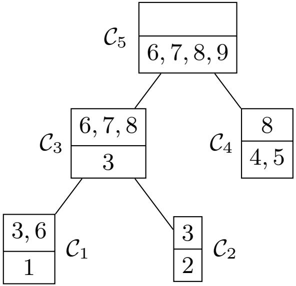
Algorithm: Clique tree-based merging

*Available packages: `SparseCoLO` [Fujisawa et al. 2009], `Chompack` [M. Andersen and Lieven Vandenberghe 2015]

Clique tree-based merging strategies





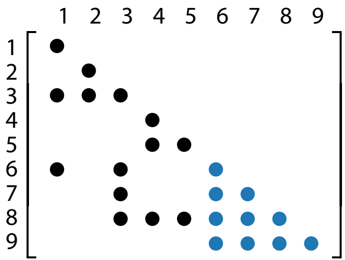


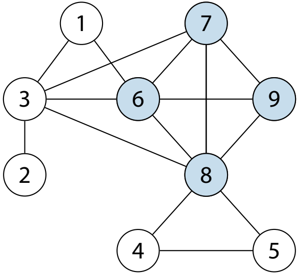
Algorithm: Clique tree-based merging

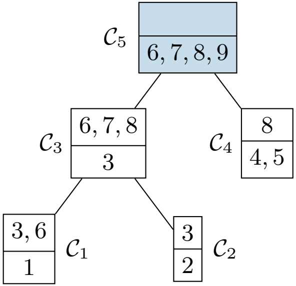
Compute clique tree;

*Available packages: `SparseCoLO` [Fujisawa et al. 2009], `Chompack` [M. Andersen and Lieven Vandenberghe 2015]

Clique tree-based merging strategies





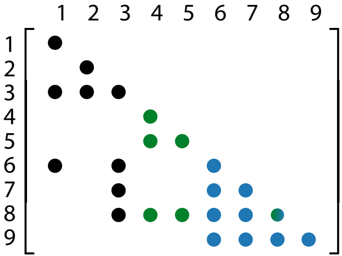


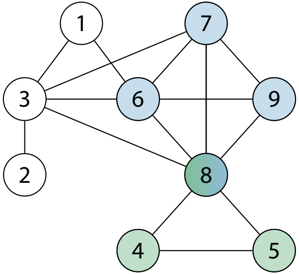
Algorithm: Clique tree-based merging

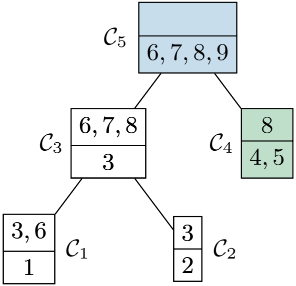
Compute clique tree;
Traverse tree depth-first: $\mathcal{C}_i$:

* Available packages: `SparseCoLO` [Fujisawa et al. 2009], `Chompack` [M. Andersen and Lieven Vandenberghe 2015]

Clique tree-based merging strategies





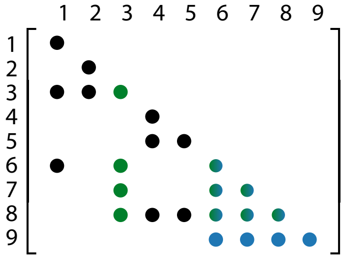


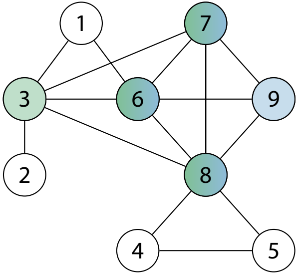
Algorithm: Clique tree-based merging

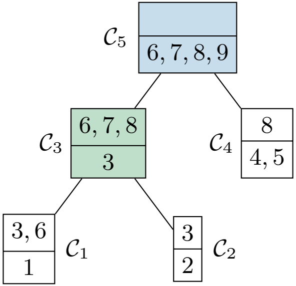
Compute clique tree;
Traverse tree depth-first: $\mathcal{C}_i$:
  Find child node: $\mathcal{C}_j$;
  if heuristic condition $f(\mathcal{C}_i, \mathcal{C}_j) \geq \gamma$ holds:

* Available packages: `SparseCoLO` [Fujisawa et al. 2009], `Chompack` [M. Andersen and Lieven Vandenberghe 2015]

Clique tree-based merging strategies





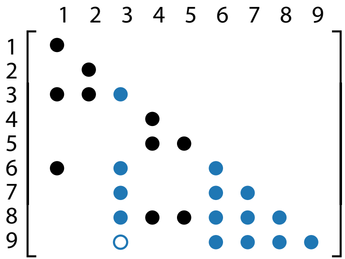


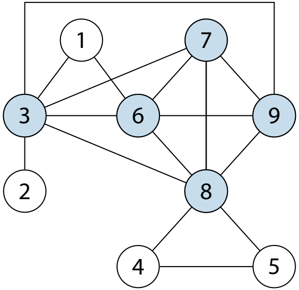
Algorithm: Clique tree-based merging

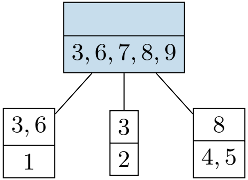
Compute clique tree;
Traverse tree depth-first: $\mathcal{C}_i$:
  Find child node: $\mathcal{C}_j$;
  if heuristic condition $f(\mathcal{C}_i, \mathcal{C}_j) \geq \gamma$ holds:

* Available packages: `SparseCoLO` [Fujisawa et al. 2009], `Chompack` [M. Andersen and Lieven Vandenberghe 2015]

Clique tree-based merging strategies





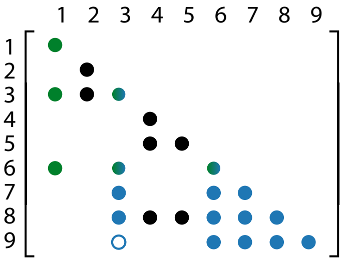


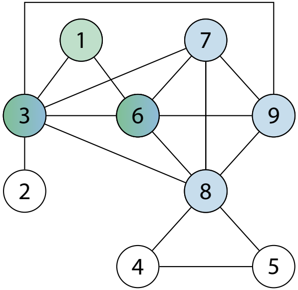
Algorithm: Clique tree-based merging

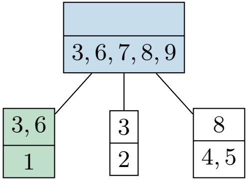
Compute clique tree;
Traverse tree depth-first: $\mathcal{C}_i$:
  Find child node: $\mathcal{C}_j$;
  if heuristic condition $f(\mathcal{C}_i, \mathcal{C}_j) \geq \gamma$ holds:
    $\mathcal{C}_m \leftarrow \mathcal{C}_i \cup \mathcal{C}_j$

*Available packages: `SparseCoLO` [Fujisawa et al. 2009], `Chompack` [M. Andersen and Lieven Vandenberghe 2015]

Clique tree-based merging strategies





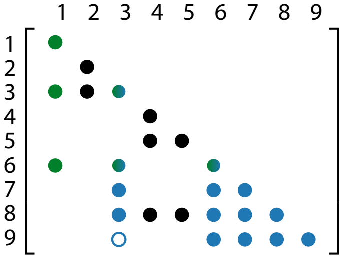


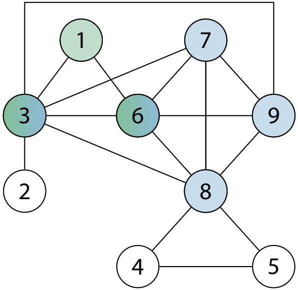
Algorithm: Clique tree-based merging

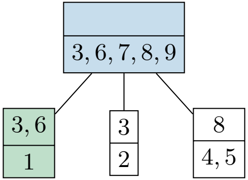
Compute clique tree;
Traverse tree depth-first: $\mathcal{C}_i$:
  Find child node: $\mathcal{C}_j$;
  if heuristic condition $f(\mathcal{C}_i, \mathcal{C}_j) \geq \gamma$ holds:
    $\mathcal{C}_m \leftarrow \mathcal{C}_i \cup \mathcal{C}_j$

* Available packages: `SparseCoLO` [Fujisawa et al. 2009], `Chompack` [M. Andersen and Lieven Vandenberghe 2015]

Clique tree-based merging strategies







- Designed for interior-point solvers

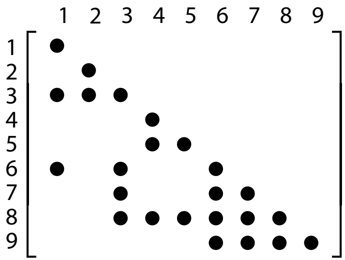
+ Clique tree cheap to compute and evaluate

– Disregards distant merge candidates

– Relies on heuristic parameters

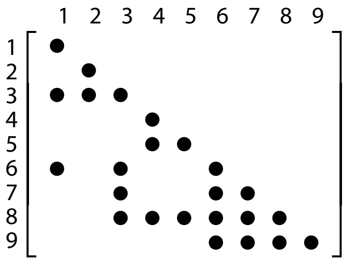
*Available packages: `SparseCoLO` [Fujisawa et al. 2009], `Chompack` [M. Andersen and Lieven Vandenberghe 2015]

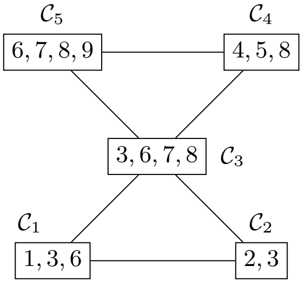
A new clique graph-based merging strategy



Algorithm: Clique graph-based merging

A new clique graph-based merging strategy

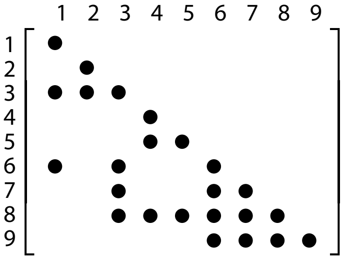


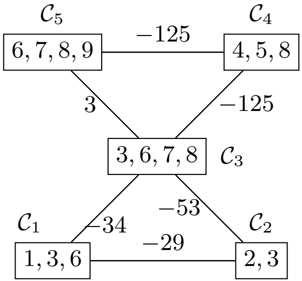


Algorithm: Clique graph-based merging

Compute reduced clique graph;

A new clique graph-based merging strategy



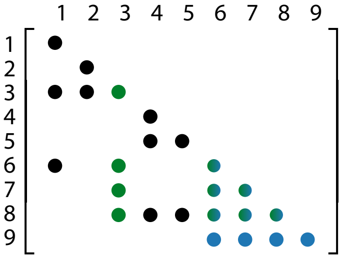


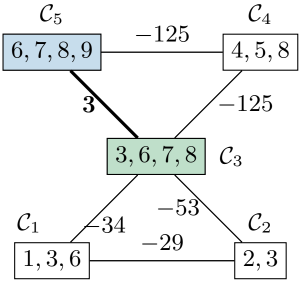
in this example:
$e(\mathcal{C}_i, \mathcal{C}_j) = |\mathcal{C}_i|^3 + |\mathcal{C}_j|^3 - |\mathcal{C}_i \cup \mathcal{C}_j|^3$

Algorithm: Clique graph-based merging

Compute reduced clique graph;
Compute edge weights $w_{ij} = e(\mathcal{C}_i, \mathcal{C}_j)$;

A new clique graph-based merging strategy



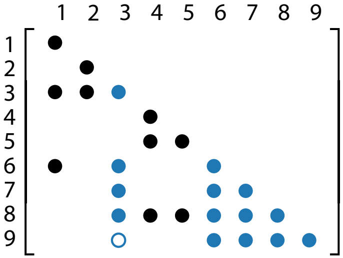


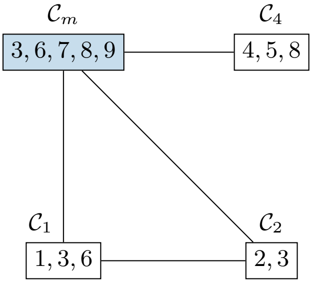
in this example:
$e(\mathcal{C}_i, \mathcal{C}_j) = |\mathcal{C}_i|^3 + |\mathcal{C}_j|^3 - |\mathcal{C}_i \cup \mathcal{C}_j|^3$

Algorithm: Clique graph-based merging

Compute reduced clique graph;
Compute edge weights $w_{ij} = e(\mathcal{C}_i, \mathcal{C}_j)$;
while $w_{ij} > 0$ exists:

A new clique graph-based merging strategy



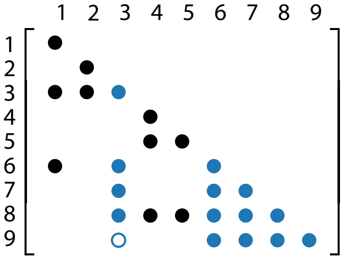


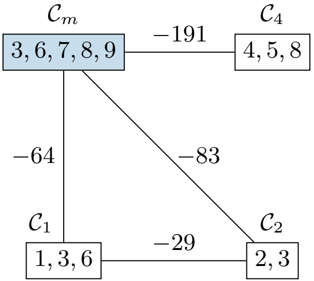
in this example:
$e(\mathcal{C}_i, \mathcal{C}_j) = |\mathcal{C}_i|^3 + |\mathcal{C}_j|^3 - |\mathcal{C}_i \cup \mathcal{C}_j|^3$

Algorithm: Clique graph-based merging

Compute reduced clique graph;
Compute edge weights $w_{ij} = e(\mathcal{C}_i, \mathcal{C}_j)$;
while $w_{ij} > 0$ exists:
  Merge permissible $(\mathcal{C}_i, \mathcal{C}_j)$ with max weight $\to \mathcal{C}_m$;

A new clique graph-based merging strategy





in this example:
$e(\mathcal{C}_i, \mathcal{C}_j) = |\mathcal{C}_i|^3 + |\mathcal{C}_j|^3 - |\mathcal{C}_i \cup \mathcal{C}_j|^3$

Algorithm: Clique graph-based merging

Compute reduced clique graph;
Compute edge weights $w_{ij} = e(\mathcal{C}_i, \mathcal{C}_j)$;
while $w_{ij} > 0$ exists:
  Merge permissible $(\mathcal{C}_i, \mathcal{C}_j)$ with max weight $\rightarrow \mathcal{C}_m$;
  Update edge weights connected to $\mathcal{C}_m$;

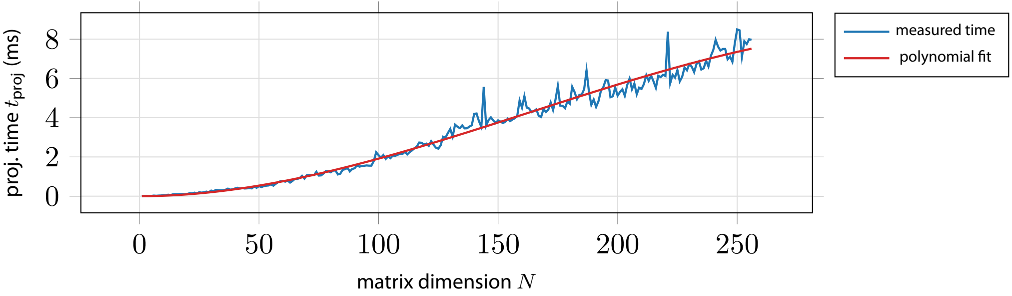
Benchmarks

- **Goal:** Reduce the projection time of our first-order ADMM solver `COSMO`
- **Problem set:** Large sparse SDPs from the SDPLib collection and SuiteSparse Matrix Library
- **Setup:** Compare different merge strategies with our solver

a) No decomposition	b) No merging
c) `SparseCoLO` merging	d) Parent-child merging
e) Clique graph merging (nominal)	f) Clique graph merging (estimated)

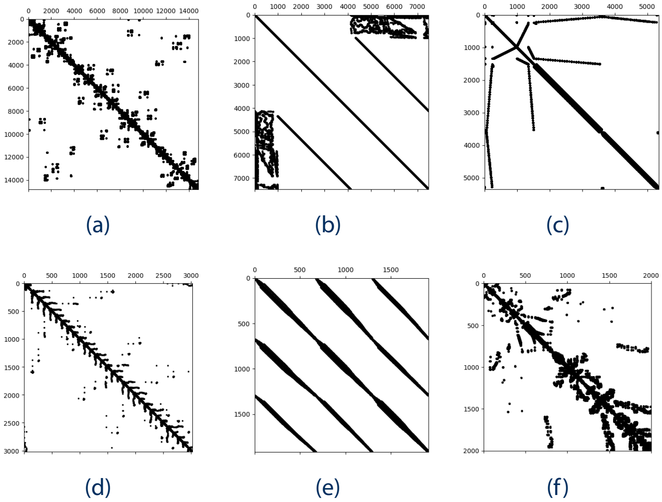
Benchmarks

- **Goal:** Reduce the projection time of our first-order ADMM solver `COSMO`
- **Problem set:** Large sparse SDPs from the SDPLib collection and SuiteSparse Matrix Library
- **Setup:** Compare different merge strategies with our solver
  - a) No decomposition
  - b) No merging
  - c) `SparseCoLO` merging
  - d) Parent-child merging
  - e) Clique graph merging (nominal)
  - f) Clique graph merging (estimated)

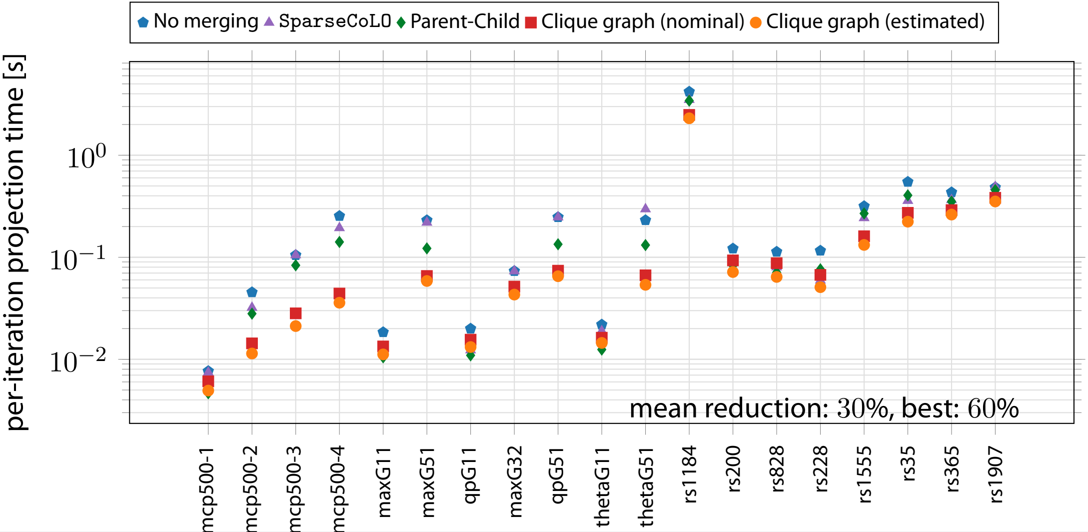


Benchmark sparsity patterns

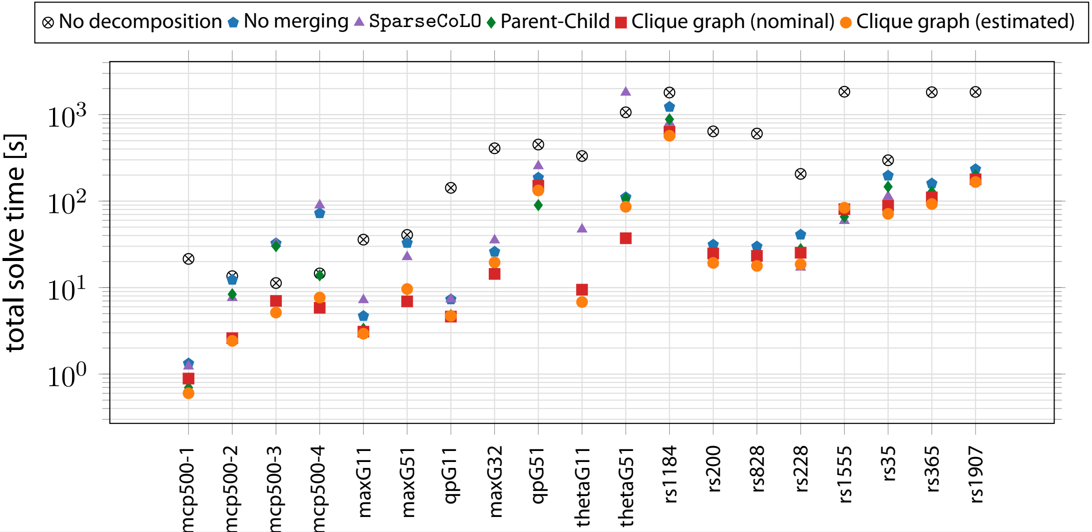
- 500 - 21M nonzeros



Benchmark results - projection time



Benchmark results - solve time



Benchmark results

- Hardware: Oxford ARC-HTC 16 logical Intel Xeon E5-2560 cores, 64GB RAM

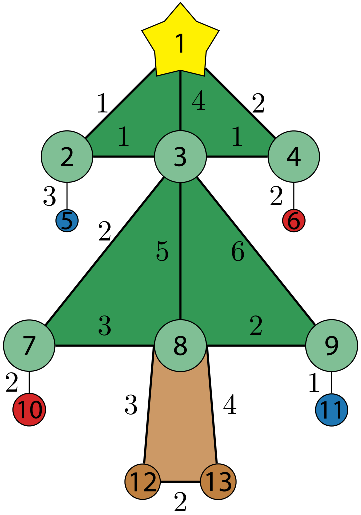
Table: Solve times for different SDP solvers.

problem	COSMO	Mosek	SCS	problem	COSMO	Mosek	SCS
maxG11	**1.47**	4.45	131.8	rs1184	**224.86**	$*\,*\,*^m$	$*\,*\,*^m$
maxG32	**6.25**	50.84	840.79	rs1555	**66.6**	$*\,*\,*^\dagger$	$*\,*\,*^m$
maxG51	**8.09**	9.92	36.56	rs1907	**104.61**	$*\,*\,*^\dagger$	$*\,*\,*^m$
mcp500-1	**0.24**	1.7	29.28	rs200	**12.47**	752.27	$*\,*\,*^\dagger$
mcp500-2	**1.68**	1.75	17.36	rs228	**12.86**	395.24	982.5
mcp500-3	4.41	**1.68**	8.36	rs35	**54.88**	919.19	$*\,*\,*^\dagger$
mcp500-4	8.2	**1.76**	7.4	rs365	**62.65**	$*\,*\,*^\dagger$	$*\,*\,*^\dagger$
qpG11	**2.36**	26.23	734.7	rs828	**10.84**	825.03	$*\,*\,*^\dagger$
qpG51	121.6	**96.42**	527.55	thetaG51	71.21	**50.08**	967.43
thetaG11	**2.32**	8.53	142.53				

^{m} out-of-memory error, † 30min timelimit

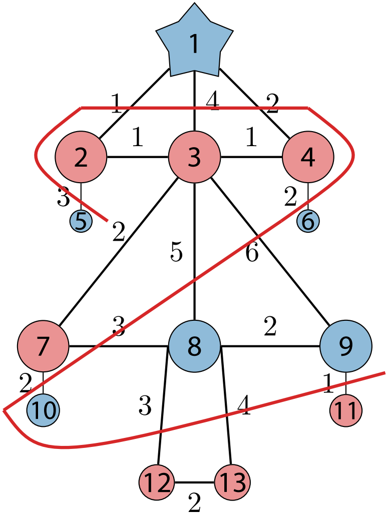
Code example

What is the maximum cut through this Christmas tree?



Code example

What is the maximum cut through this Christmas tree?



Conclusion:

- Novel clique graph based merging strategy
- Considers many pair-wise merge candidates
- Customisable to solver algorithm and hardware used
- Reduced per-iteration time of our solver

Conclusion:

- Novel clique graph based merging strategy
- Considers many pair-wise merge candidates
- Customisable to solver algorithm and hardware used
- Reduced per-iteration time of our solver

Future work:

- Test with interior-point solver
- Tailor strategy for multithreading

Conclusion:

- Novel clique graph based merging strategy
- Considers many pair-wise merge candidates
- Customisable to solver algorithm and hardware used
- Reduced per-iteration time of our solver

Future work:

- Test with interior-point solver
- Tailor strategy for multithreading

Solver package available:

`https://github.com/oxfordcontrol/COSMO.jl`

Conclusion:

- Novel clique graph based merging strategy
- Considers many pair-wise merge candidates
- Customisable to solver algorithm and hardware used
- Reduced per-iteration time of our solver

Future work:

- Test with interior-point solver
- Tailor strategy for multithreading

Solver package available:

`https://github.com/oxfordcontrol/COSMO.jl`

- Michael Garstka, Mark Cannon, and Paul Goulart (2020a). "A clique graph based merging strategy for decomposable SDPs". In: *21st IFAC World Congress*. Berlin, Germany
- Michael Garstka, Mark Cannon, and Paul Goulart (2020b). "COSMO: A conic operator splitting method for convex conic problems". In: *arXiv*. arXiv: 1901.10887 [math.OC]. URL: https://arxiv.org/abs/1901.10887

Conclusion:

- Novel clique graph based merging strategy
- Considers many pair-wise merge candidates
- Customisable to solver algorithm and hardware used
- Reduced per-iteration time of our solver

Future work:

- Test with interior-point solver
- Tailor strategy for multithreading

Solver package available:

`https://github.com/oxfordcontrol/COSMO.jl`

Questions?

- Michael Garstka, Mark Cannon, and Paul Goulart (2020a). "A clique graph based merging strategy for decomposable SDPs". In: *21st IFAC World Congress*. Berlin, Germany
- Michael Garstka, Mark Cannon, and Paul Goulart (2020b). "COSMO: A conic operator splitting method for convex conic problems". In: *arXiv*. arXiv: `1901.10887 [math.OC]`. URL: `https://arxiv.org/abs/1901.10887`