A CONIC INTERIOR POINT DECOMPOSITION APPROACH
FOR LARGE SCALE SEMIDEFINITE PROGRAMMING

Kartik Krishnan Sivaramakrishnan
Department of Mathematics
North Carolina State University
kksivara@ncsu.edu
http://www4.ncsu.edu/~kksivara

ISMP 2006
Rio de Janeiro, Brazil
July 31, 2006
Contents

• Conic programming
• Motivation for our decomposition approach
• A decomposition framework for conic programming
• Conic Dantzig-Wolfe decomposition
• Algorithm
• Implementational Issues in Algorithm
• Computational results
• Computational results on block angular SDPs
• Conclusions

-1-
Conic programming

\[(P) \quad \min \quad c^T x \]
\[\text{s.t.} \quad Ax = b \]
\[\quad x \in K \]

\[(D) \quad \max \quad b^T y \]
\[\text{s.t.} \quad A^T y + s = c \]
\[\quad s \in K \]

where $A \in \mathbb{R}^{m \times n}$, $b \in \mathbb{R}^m$, $c \in \mathbb{R}^n$, $K = K_1 \times \ldots \times K_r$

- $r = 1$, $K = \mathbb{R}_+^n = \{x \in \mathbb{R}^n : x \geq 0\}$ LP
  Very large LPs ($m, n \leq \text{1,000,000}$) solvable by the simplex method and/or IPMs.

- $K_i = \mathbb{Q}_{+}^{n_i} = \{x \in \mathbb{R}^{n_i} : x_1 \geq ||x_{2:n_i}||\}$ SOCP
  Large SOCPs ($m, n \leq \text{100,000}$) solvable by IPMs.

- $K_i = \mathbb{S}_{+}^{n_i} = \{X \in \mathbb{S}^{n_i} : X \succeq 0\}$ SDP
  Medium sized SDPs ($m, n \leq \text{1000}$) solvable by IPMs.
  (Beyond 10,000 seems impossible today!)
Motivation

(a) Solve large scale structured semidefinite programs (SDP) arising in science and engineering. Typically, these SDPs need not be solved very accurately.

(b) The technique is to iteratively solve an SDP between a mixed conic master problem over linear, second order, and semidefinite cones; and distributed subproblems (smaller SDPs) in a high performance computing environment.

(c) Improve the scalability of interior point methods (IPMs) by applying them instead on the smaller master problem, and subproblems (which are solved in parallel!)

This is our conic interior point decomposition scheme.
Semidefinite programs with a decomposable structure

1. **Preprocessed SDPs after matrix completion:**

2. **Stability analysis of interconnected subsystems:**

3. **Stochastic semidefinite programs:** Mehrotra-Ozëvin (2005)

4. **Polynomial optimization problems with structured sparsity:**

5. **Semidefinite programs arising in lift and project schemes:**
   Rendl-Sotirov (2003) and Burer-Vandenbussche (2006)
Semidefinite programming

(SDP) \[ \begin{align*} \min \quad & C \cdot X \\ \text{s.t.} \quad & \mathcal{A}(X) = b \\ & X \succeq 0 \end{align*} \]

(SDD) \[ \begin{align*} \max \quad & b^T y \\ \text{s.t.} \quad & \mathcal{A}^T y + S = C \\ & S \succeq 0 \end{align*} \]

- Notation
  - \( X, S, C \in S^n, b \in \mathbb{R}^m \)
  - \( A \cdot B = \text{trace}(AB) = \sum_{i,j=1}^{n} A_{ij} B_{ij} \) (Frobenius inner product)
  - The operator \( \mathcal{A} : S^n \to \mathbb{R}^m \) and its adjoint \( \mathcal{A}^T : \mathbb{R}^m \to S^n \) are
    \[ \mathcal{A}(X) = \begin{pmatrix} A_1 \cdot X \\ \vdots \\ A_m \cdot X \end{pmatrix}, \quad \mathcal{A}^T y = \sum_{i=1}^{m} y_i A_i \]
    where \( A_i \in S^n, i = 1, \ldots, m \)

-5-
Assumptions

(1) \(A_1, \ldots, A_m\) are **linearly independent** in \(S^n\).

(2) **Slater condition** for (SDP) and (SDD):

\[
\{ X \in S^n : A(X) = b, X \succ 0 \} \neq \emptyset
\]

\[
\{ (y, S) \in \mathbb{R}^m \times S^n : A^T y + S = C, S \succ 0 \} \neq \emptyset
\]

(3) One of the constraints in (SDP) is \(I \bullet X = 1\) (\(\text{trace}(X) = 1\)).

   (Every SDP with a bounded primal feasible set can be reformulated to satisfy this assumption)
Conic Dantzig-Wolfe decomposition

- Our semidefinite programs are:

\[
\begin{align*}
\text{(SDP)} & \quad \min & C \cdot X \\
\text{s.t.} & & \mathcal{A}(X) = b \\
& & \text{trace}(X) = 1 \\
& & X \succeq 0
\end{align*}
\]

\[
\begin{align*}
\text{(SDD)} & \quad \max & b^T y + z \\
\text{s.t.} & & \mathcal{A}^T y + zI + S = C \\
& & S \succeq 0
\end{align*}
\]

- Consider the set:

\[\mathcal{E} = \{X \in S^n : \text{trace}(X) = 1, X \succeq 0\}\]

- The extreme points of \(\mathcal{E}\) are:

\[\{vv^T : v \in \mathbb{R}^n, v^Tv = 1\}\]  (Infinite number)
Conic Dantzig-Wolfe decomposition:

Any \( X \in \mathcal{E} \) can be written as:

1. \( X = \sum_j \lambda_j d_j d_j^T \) where \( \sum_j \lambda_j = 1, \lambda_j \geq 0 \)

   \( X \succeq 0 \) is replaced by \( \lambda_j \geq 0, \forall j \) (Semi-infinite LP formulation)

2. \( X = \sum_j P_j V_j P_j^T \) where \( \sum_j \text{trace}(V_j) = 1, V_j \in S^2_+ \)

   \( X \succeq 0 \) is replaced by \( V_j \succeq 0, \forall j \) (Semi-infinite SOCP formulation)

   (A \( 2 \times 2 \) SDP constraint can be written as a size 3 SOCP constraint)

3. \( X = \sum_j P_j V_j P_j^T \) where \( \sum_j \text{trace}(V_j) = 1, V_j \in S^{r_j}_+, r_j \geq 3 \)

   \( X \succeq 0 \) is replaced by \( V_j \succeq 0, \forall j \) (Semi-infinite SDP formulation)
A $2 \times 2$ SDP constraint is an SOCP constraint of size 3

\[ X = \begin{pmatrix} X_{11} & X_{12} \\ X_{12} & X_{22} \end{pmatrix} \succeq 0 \]

\[ \iff X_{11} \geq 0, \ X_{22} \geq 0, \ X_{11}X_{22} - X_{12}^2 \geq 0 \]

\[ \iff \left( \begin{array}{cc} X_{11} + X_{22} \\ 2X_{12} \\ X_{11} - X_{22} \end{array} \right) \succeq_Q 0 \]
Conic Dantzig-Wolfe decomposition: Master problem

The decomposed conic problem over LP, SOCP and SDP blocks is:

Primal

\[
\begin{align*}
\min & \quad \sum_{i=1}^{n_l} c_{li} x_{li} + \sum_{j=1}^{n_q} c_{qj}^T x_{qj} + \sum_{k=1}^{n_s} C_{sk} \cdot X_{sk} \\
\text{s.t.} & \quad \sum_{i=1}^{n_l} A_{li} x_{li} + \sum_{j=1}^{n_q} A_{qj} x_{qj} + \sum_{k=1}^{n_s} A_{sk}(X_{sk}) = b \\
& \quad \sum_{i=1}^{n_l} x_{li} + \sum_{j=1}^{n_q} x_{qj} + \sum_{k=1}^{n_s} \text{trace}(X_{sk}) = 1 \\
& \quad x_{li} \geq 0, \quad i = 1, \ldots, n_l \\
& \quad x_{qj} \succeq Q 0, \quad j = 1, \ldots, n_q \\
& \quad X_{sk} \succeq 0, \quad k = 1, \ldots, n_s
\end{align*}
\]

Dual

\[
\begin{align*}
\max & \quad b^T y + z \\
\text{s.t.} & \quad A_{li}^T y + z + s_{li} = c_{li}, \quad i = 1, \ldots, n_l \\
& \quad A_{qj}^T y + z \begin{pmatrix} 1 \\ 0 \\ 0 \end{pmatrix} + s_{qj} = c_{qj}, \quad j = 1, \ldots, n_q \\
& \quad A_{sk}^T y + z I_{r_k} + S_{sk} = C_{sk}, \quad k = 1, \ldots, n_s \\
& \quad s_{li} \geq 0, \quad i = 1, \ldots, n_l \\
& \quad s_{qj} \succeq Q 0, \quad j = 1, \ldots, n_q \\
& \quad S_{sk} \succeq 0, \quad k = 1, \ldots, n_s
\end{align*}
\]
Conic Dantzig-Wolfe decomposition: Separation Oracle

Input: \((y^*, z^*)\) a feasible point for dual master problem.

- If \(\lambda_{\min}(S^*) \geq 0\), report feasibility STOP.
- Else, solve the following subproblem for \(X^*\):

\[
\begin{align*}
\min \ (C - A^T y^* - z^* I) \bullet X \\
\text{s.t.} \quad I \bullet X &= 1 \\
X &\succeq 0
\end{align*}
\]

Factorize \(X^* = DMD^T\) with \(M \succ 0\).
Return the cut \(D^T(C - A^T y - zI)D \succeq 0\) with \(D \in \mathbb{R}^{n \times r}\).

- If \(r = 1\) is an LP cut.
- If \(r = 2\) is an SOCP cut.
- If \(r \geq 3\) is an SDP cut of small size.
Figure 1: Conic Dantzig-Wolfe Algorithm
Choice of the query point: 1

• In the original Dantzig-Wolfe (Kelley) scheme the query point \((y, z)\) is an **optimal solution** to the dual master problem. This scheme has a **poor** rate of convergence.
• Better to solve the master problem approximately initially, and gradually tighten this tolerance as the algorithm proceeds.
  – Initially, the master problem is a poor approximation to the SDP problem.

  Weak tolerance \(\Rightarrow\) Central dual iterates \((y, z)\)
  \(\Rightarrow\) Oracle returns better cuts

  – As the algorithm proceeds, the master problem approximations get tighter.

  Tight tolerance \(\Rightarrow\) More emphasis on the objective function
  \(\Rightarrow\) Convergence to an optimal solution
Choice of the query point: 2

Our adaptive strategy:

- Solve the master problem to a tolerance TOL for \((x^*, y^*, z^*, s^*)\). Compute the following parameters:

\[
\text{GAPTOL}(x^*, s^*) = \max\{1, \frac{1}{2}(c^Tx^* + b^Ty^*)\},
\]

\[
\text{INF}(y^*, z^*) = \lambda_{\min}(C - A^Ty^* - z^*I),
\]

\[
\text{OPT}(x^*, y^*, z^*, s^*) = \max\{\text{GAPTOL}(x^*, s^*), \text{INF}(y^*, z^*)\}
\]

- If \(\text{OPT}(x^*, y^*, z^*, s^*) < TOL\), we lower TOL by a constant factor. More precisely, \(TOL = \mu \times \text{OPT}(x^*, y^*, z^*, s^*)\) with \(0 < \mu < 1\). Else, TOL remains unchanged.

-14-
Figure 2: The complete algorithm

STEP 1: Initialize master problem
Tolerance TOL
Iterate \((x^1, y^1, s^1)\)
\(k=1, \varepsilon=1e-3\)

STEP 2: Call separation oracle at \(y^k\)

STEP 3: Termination check
Is \(\text{OPT} < \varepsilon\)?

STEP 4: Tolerance check
If \(\text{OPT} < \text{TOL}\) tighten TOL
Else TOL is unchanged

STEP 5: Update master problem
(a) Update master problem
(b) Warm start master problem

STEP 6: Solve master problem to tolerance TOL for \((x^{k+1}, y^{k+1}, s^{k+1})\)
(a) Solve master problem
(b) Update bounds LB, UB

STEP 7: (a) Detect unimportant constraints
(b) Aggregate unimportant constraints and update master problem

STEP 8: Construct solution for SDP

Yes

Oracle reports feasibility

Oracle returns cut in Step 2
Figure 3: One iteration of algorithm.
Upper and lower bounds

• The objective value of the primal master problem in every iteration gives an upper bound on the SDP objective value. This is the objective value of the dual master problem plus the duality gap.

• Given a solution \((y^*, z^*)\) to the dual master problem in every iteration, a lower bound is computed as follows:
  
  – Compute \(\lambda^* = \lambda_{\text{min}}(S^*)\), where \(S^* = (C - ATy^* - z^*I)\).
  – Set \(y^{lb} = y^*\) and \(z^{lb} = z^* + \lambda^*\).
  – A lower bound on the SDP objective value is then \(b^ty^{lb} + z^{lb}\)

  **Note:** \((y^{lb}, z^{lb})\) is a feasible point in (SDD).

• We could also terminate if the difference between these bounds is small.
Warm start after adding a matrix: 1

- The solution to the old master problem is \((x^*_l, x^*_s, y^*, s^*_l, s^*_s)\). Consider adding a semidefinite cut \(a_s^Ty \leq d\) in the dual master problem. The new master problem is:

\[
\begin{array}{ll}
\text{min} & c^T_l x_l + c^T_s x_s + d^T \beta \\
\text{s.t.} & A_l x_l + A_s x_s + a_s \beta = b \\
 & x_l \geq 0, \quad \beta \geq 0 \\
 & x_s \geq 0.
\end{array}
\]

\[
\begin{array}{ll}
\text{max} & b^T y \\
\text{s.t.} & s_l = c_l - A_l^T y \geq 0 \\
 & s_s = c_s - A_s^T y \geq 0 \\
 & \gamma = d - a_s^T y \geq 0.
\end{array}
\]

- We have \(\gamma^* \not\geq 0\). We can perturb \(y^*\) (lower bounding) to generate \(y^{lb}\) so that \(\gamma^{lb} \geq 0\).

- The perturbed point \((x^*_l, x^*_s, \beta^*, y^{lb}, s^{lb}_l, s^{lb}_s, \gamma^{lb})\) is feasible in the new master problem with \(\beta^* \geq 0\) and \(\gamma^{lb} \geq 0\), but NOT STRICTLY!.

- We want to increase \(\beta^*\) and \(\gamma^{lb}\), making them strictly feasible, while limiting the variation in the other variables.
Warm start after adding a matrix: 2

- We solve the following problems

\[
\text{(WSP)} \quad \begin{array}{l}
\text{max} \quad \log \det \beta \\
\text{s.t.} \quad A_l \Delta x_l + A_s \Delta x_s + a_s \beta = 0 \\
\quad \sqrt{||D_l^{-1} \Delta x_l||^2 + ||D_s^{-1} \Delta x_s||^2} \leq 1 \\
\beta \geq 0.
\end{array}
\]

\[
\text{(WSD)} \quad \begin{array}{l}
\text{max} \quad \log \det \gamma \\
\text{s.t.} \quad a_s^T \Delta y + \gamma = 0 \\
\quad \sqrt{||D_l A_l^T \Delta y||^2 + ||D_s A_s^T \Delta y||^2} \leq 1 \\
\gamma \geq 0.
\end{array}
\]

for \((\Delta x_l, \Delta x_s, \beta)\) and \((\Delta y, \gamma)\) respectively. Here \(D_l\) and \(D_s\) are appropriate primal-dual scaling matrices for LP and SDP blocks at \((x_l^*, x_s^*, s_{lb}^l, s_{lb}^s)\).

- Compute \((\Delta s_l, \Delta s_s) = (-A_l^T \Delta y, -A_s^T \Delta y)\).
Warm start after adding a matrix: 3

- Compute

\[
\begin{align*}
\Delta x_l &= -D_l^2 A_l^T (A_l D_l^2 A_l^T + A_s D_s^2 A_s^T)^{-1} a_s \beta \\
\Delta x_s &= -D_s^2 A_s^T (A_l D_l^2 A_l^T + A_s D_s^2 A_s^T)^{-1} a_s \beta \\
\Delta y &= -(A_l D_l^2 A_l^T + A_s D_s^2 A_s^T)^{-1} a_s \beta \\
\gamma &= V \beta
\end{align*}
\]

where \( \beta \in S^p \) is the solution to

\[
\min \left\{ \frac{p}{2} \beta^T V \beta - \log \det \beta : \beta \succeq 0 \right\}
\]

where \( V = a_s^T (A_l D_l^2 A_l^T + A_s D_s^2 A_s^T)^{-1} a_s \).

- Finally set

\[
\begin{align*}
(x_{l}^{st}, x_{s}^{st}, \beta^{st}) &= (x_{l}^*, x_{s}^* + \kappa \alpha^p_{\max} \Delta x_l, x_{s}^* + \kappa \alpha^p_{\max} \Delta x_s, \kappa \alpha^p_{\max} \beta) \\
y_{l}^{st} &= y_{l}^{lb} + \kappa \alpha^d_{\max} \Delta y \\
(y_{s}^{st}, \gamma^{st}) &= (y_{s}^{lb} + \kappa \alpha^d_{\max} \Delta s_l, y_{s}^{lb} + \kappa \alpha^d_{\max} \Delta s_s, \kappa \alpha^d_{\max} \gamma)
\end{align*}
\]

where \( \kappa \in (0, 1) \) and \( \alpha^p_{\max}, \alpha^d_{\max} \) are the maximal primal and dual step lengths respectively.
Aggregating unimportant blocks

- Use a primal-dual scaling measure to evaluate blocks:
  - For the $i$th linear block the measure is $\frac{x_i}{s_i}$.
  - For the $j$th semidefinite block the measure is $\frac{\text{trace}(x_j)}{\text{trace}(s_j)}$.

- Blocks with small measure are aggregated in an LP block.
- The resulting master problem is

\[
\begin{align*}
\min \quad & c_{agg} x_{agg} + c_l^T x_l + c_s^T x_s \\
\text{s.t.} \quad & A_{agg} x_{agg} + A_l x_l + A_s x_s = b \\
& x_l \geq 0, \quad x_{agg} \geq 0 \\
& x_s \succeq 0.
\end{align*}
\]

\[
\begin{align*}
\max \quad & b^T y \\
\text{s.t.} \quad & s_l = c_l - A_l^T y \geq 0 \\
& s_s = c_s - A_s^T y \succeq 0 \\
& s_{agg} = c_{agg} - A_{agg}^T y \geq 0
\end{align*}
\]
Computational Results

- Matlab code with the following features:
  
  - Master problem contains linear and semidefinite blocks only.
  
  - SDPT3 is used as conic solver for master problem with H.K.M. scaling
  
  - Oracle is implemented using MATLAB’s Lanczos solver (eigs), which computes the \( r = \lfloor \sqrt{m}/2 \rfloor \) most negative eigenvalues and eigenvectors of \( S \) in each iteration.
  
  - Solve the problem for \( m \) iterations, or until \( \text{TOL} = 1e-3 \), whichever comes earlier.

- All computational results on a 2.4 GHz processor with 1 GB of memory.
### Computational results for Max-Cut

<table>
<thead>
<tr>
<th>Problem</th>
<th>n</th>
<th>Opt value</th>
<th>LP cones</th>
<th>SDP cones</th>
<th>Lower bound</th>
<th>$\epsilon$</th>
<th>Time (h:m:s)</th>
</tr>
</thead>
<tbody>
<tr>
<td>mcp100</td>
<td>100</td>
<td>-226.16</td>
<td>101</td>
<td>152(31)</td>
<td>-226.32</td>
<td>1.6e-3</td>
<td>16</td>
</tr>
<tr>
<td>mcp124-1</td>
<td>124</td>
<td>-141.99</td>
<td>125</td>
<td>187(48)</td>
<td>-141.87</td>
<td>1.5e-3</td>
<td>34</td>
</tr>
<tr>
<td>mcp124-2</td>
<td>124</td>
<td>-269.88</td>
<td>125</td>
<td>173(36)</td>
<td>-270.05</td>
<td>1.4e-3</td>
<td>25</td>
</tr>
<tr>
<td>mcp124-3</td>
<td>124</td>
<td>-467.75</td>
<td>125</td>
<td>174(35)</td>
<td>-468.07</td>
<td>1.5e-3</td>
<td>22</td>
</tr>
<tr>
<td>mcp124-4</td>
<td>124</td>
<td>-864.41</td>
<td>125</td>
<td>189(41)</td>
<td>-864.99</td>
<td>1.5e-3</td>
<td>21</td>
</tr>
<tr>
<td>mcp250-1</td>
<td>250</td>
<td>-317.26</td>
<td>251</td>
<td>339(84)</td>
<td>-317.46</td>
<td>1.3e-3</td>
<td>2:03</td>
</tr>
<tr>
<td>mcp250-2</td>
<td>250</td>
<td>-531.93</td>
<td>251</td>
<td>343(51)</td>
<td>-532.35</td>
<td>1.7e-3</td>
<td>1:05</td>
</tr>
<tr>
<td>mcp250-3</td>
<td>250</td>
<td>-981.17</td>
<td>251</td>
<td>307(44)</td>
<td>-981.86</td>
<td>1.6e-3</td>
<td>49</td>
</tr>
<tr>
<td>mcp250-4</td>
<td>250</td>
<td>-1681.96</td>
<td>251</td>
<td>301(43)</td>
<td>-1683.1</td>
<td>1.6e-3</td>
<td>47</td>
</tr>
<tr>
<td>mcp500-1</td>
<td>500</td>
<td>-598.15</td>
<td>501</td>
<td>735(115)</td>
<td>-598.56</td>
<td>1.6e-3</td>
<td>12:29</td>
</tr>
<tr>
<td>mcp500-2</td>
<td>500</td>
<td>-1070.06</td>
<td>501</td>
<td>634(70)</td>
<td>-1070.70</td>
<td>1.5e-3</td>
<td>6:18</td>
</tr>
<tr>
<td>mcp500-3</td>
<td>500</td>
<td>-1847.97</td>
<td>501</td>
<td>570(57)</td>
<td>-1849.20</td>
<td>1.6e-3</td>
<td>4:33</td>
</tr>
<tr>
<td>mcp500-4</td>
<td>500</td>
<td>-3566.73</td>
<td>501</td>
<td>523(49)</td>
<td>-3569.10</td>
<td>1.6e-3</td>
<td>2:51</td>
</tr>
<tr>
<td>toruspmm3-8-50</td>
<td>512</td>
<td>-527.80</td>
<td>513</td>
<td>681(84)</td>
<td>-528.10</td>
<td>1.4e-3</td>
<td>9:28</td>
</tr>
<tr>
<td>torusg3-8</td>
<td>512</td>
<td>-457.36</td>
<td>513</td>
<td>605(89)</td>
<td>-457.60</td>
<td>1.4e-3</td>
<td>13:11</td>
</tr>
<tr>
<td>maxG11</td>
<td>800</td>
<td>-629.16</td>
<td>801</td>
<td>734(84)</td>
<td>-629.39</td>
<td>1.4e-3</td>
<td>1:49:55</td>
</tr>
<tr>
<td>maxG51</td>
<td>1000</td>
<td>-4003.81</td>
<td>1001</td>
<td>790(54)</td>
<td>-4008.90</td>
<td>1.7e-3</td>
<td>32:24</td>
</tr>
</tbody>
</table>
Figure 4: Variation of bounds for maxG11 problem
Figure 5: Variation of error for maxG11 problem
Figure 6: Distribution of times for maxG11 problem
Computational results on random semidefinite programs

\[
\begin{align*}
\text{max} & \quad b^T y + z \\
\text{s.t.} & \quad S = (C - A^T y - zI) \succeq 0 \\
& \quad l \leq y \leq u.
\end{align*}
\]

<table>
<thead>
<tr>
<th>Prob</th>
<th>n</th>
<th>m</th>
<th>Den</th>
<th>LP cones</th>
<th>SDP cones</th>
<th>Our LB</th>
<th>Our (h:m:s)</th>
<th>SDPT3 LB</th>
<th>SDPT3 (h:m:s)</th>
</tr>
</thead>
<tbody>
<tr>
<td>test-1</td>
<td>100</td>
<td>501</td>
<td>0.1</td>
<td>1100</td>
<td>45(9)</td>
<td>424.68</td>
<td>1:06</td>
<td>424.97</td>
<td>43</td>
</tr>
<tr>
<td>test-2</td>
<td>100</td>
<td>501</td>
<td>0.9</td>
<td>1100</td>
<td>140(28)</td>
<td>181.91</td>
<td>5:21</td>
<td>182.02</td>
<td>1:32</td>
</tr>
<tr>
<td>test-3</td>
<td>500</td>
<td>101</td>
<td>0.1</td>
<td>700</td>
<td>50(25)</td>
<td>33.36</td>
<td>4:33</td>
<td>33.38</td>
<td>4:31</td>
</tr>
<tr>
<td>test-4</td>
<td>500</td>
<td>101</td>
<td>0.9</td>
<td>700</td>
<td>116(29)</td>
<td>-37.99</td>
<td>4:49</td>
<td>-37.96</td>
<td>6:35</td>
</tr>
<tr>
<td>test-5</td>
<td>1000</td>
<td>51</td>
<td>0.1</td>
<td>1100</td>
<td>24(12)</td>
<td>7.99</td>
<td>18:40</td>
<td>7.99</td>
<td>16:03</td>
</tr>
<tr>
<td>test-6</td>
<td>1000</td>
<td>51</td>
<td>0.9</td>
<td>1100</td>
<td>86(19)</td>
<td>-13.33</td>
<td>13:25</td>
<td>-13.32</td>
<td>17.30</td>
</tr>
<tr>
<td>test-7</td>
<td>300</td>
<td>300</td>
<td>0.1</td>
<td>900</td>
<td>105(21)</td>
<td>147.51</td>
<td>2:39</td>
<td>147.56</td>
<td>4:13</td>
</tr>
<tr>
<td>test-8</td>
<td>300</td>
<td>300</td>
<td>0.9</td>
<td>900</td>
<td>142(18)</td>
<td>22.91</td>
<td>4:52</td>
<td>22.92</td>
<td>8:11</td>
</tr>
</tbody>
</table>
Preprocessing sparse SDPs into block-angular SDPs: 1

Consider the SDP

$$\begin{align*}
\text{min} & \quad L \cdot X \\
\text{s.t.} & \quad X_{ii} = 1, \ i = 1, \ldots, n, \\
& \quad X \succeq 0,
\end{align*}$$

where $L$ is the adjacency matrix of the graph

\begin{itemize}
  \item $C_1 = (1, 5, 7)$
  \item $C_2 = (2, 3, 4, 6)$
  \item $C_3 = (4, 6, 7)$
  \item $C_4 = (5, 6, 7)$
\end{itemize}
Preprocessing sparse SDPs into block-angular SDPs: 2

Using matrix completion, one can reformulate the earlier SDP as

$$\min \sum_{k=1}^{4}(L^k \bullet X^k)$$

s.t.

$$X_{23}^1 - X_{13}^4 = 0,$$

$$X_{34}^2 - X_{12}^3 = 0,$$

$$X_{23}^3 - X_{23}^4 = 0,$$

$$X_{ii}^k = 1, \ i = 1, \ldots, |C_k|, \ k = 1, \ldots, 4,$$

$$X^k \succeq 0, \ k = 1, \ldots, 4,$$

which is in block-angular form.

-29-
Preliminary results on block angular semidefinite programs

- Results on the IBM Blade Center Linux Cluster (Henry2) at NC State.
- Each of the 175 nodes is a 2.8-3.2 GHz processor with 4 GB of memory.
- Our code is in C and uses MPI for interprocessor communication.
- CPLEX 9.0 was used to solve the master problem and CSDP was used to solve the subproblems.
- 3 digits of accuracy or an upper limit of 2 hours in computations.

<table>
<thead>
<tr>
<th>Prob</th>
<th>n</th>
<th>(n_p)</th>
<th>m</th>
<th>Our LB</th>
<th>Our LB (h:m:s) (p)</th>
<th>SDPT3 LB</th>
<th>SDPT3 (h:m:s)</th>
</tr>
</thead>
<tbody>
<tr>
<td>mcp124-1</td>
<td>124</td>
<td>60(6)</td>
<td>172</td>
<td>141.92</td>
<td>5(4)</td>
<td>141.99</td>
<td>4</td>
</tr>
<tr>
<td>mcp250-1</td>
<td>250</td>
<td>176(9)</td>
<td>381</td>
<td>317.06</td>
<td>3:33(4)</td>
<td>317.26</td>
<td>9</td>
</tr>
<tr>
<td>mcp500-1</td>
<td>500</td>
<td>324(19)</td>
<td>1222</td>
<td>597.74</td>
<td>34:39(8)</td>
<td>598.15</td>
<td>36</td>
</tr>
<tr>
<td>maxG11</td>
<td>800</td>
<td>216(4)</td>
<td>1208</td>
<td>628.88</td>
<td>8:50(4)</td>
<td>629.16</td>
<td>1:37</td>
</tr>
<tr>
<td>maxG32</td>
<td>2000</td>
<td>1022(2)</td>
<td>2820</td>
<td>1566.20</td>
<td>2:0:0(2)</td>
<td>1567.02</td>
<td>1:32:06</td>
</tr>
</tbody>
</table>

-30-
Figure 7: Variation of bounds for preprocessed maxG11 problem
Ongoing and Future work

- Applying the conic interior point decomposition approach for structured block angular SDPs in a parallel computing environment.

- Incorporating the decomposition scheme in the pricing phase of an SDP based conic branch-cut-price scheme for mixed integer and non-convex problems.
Thank you for your attention!
Questions, Comments, Suggestions?

The slides from this talk are available online at

A technical report appears at
1. **K. Sivaramakrishnan**, *A conic interior point decomposition for large scale structured block angular semidefinite programs*, forthcoming.


-36-