A PARALLEL CONIC INTERIOR POINT DECOMPOSITION APPROACH FOR BLOCK-ANGULAR SEMIDEFINITE PROGRAMS

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

Graduate Seminar
Department of Mechanical and Industrial Engineering
University of Minnesota
March 28, 2007

Contents

Conic programming

$$\text{(P)} \quad \max \; c^T x \quad \text{s.t.} \; Ax = b, \; x \in \mathcal{K}$$

$$\text{(D)} \quad \min \; b^T y \quad \text{s.t.} \; A^T y - s = c, \; s \in \mathcal{K}$$

where $A \in \mathbb{R}^{m \times n}$, $b \in \mathbb{R}^m$, $c \in \mathbb{R}^n$, $\mathcal{K} = \mathcal{K}_1 \times \ldots \times \mathcal{K}_r$

- $r = 1, \mathcal{K} = \mathbb{R}^n_+ = \{x \in \mathbb{R}^n : x \geq 0\}$ **LP**
  Very large LPs ($m, n \leq$ **1,000,000**) solvable by the simplex method and/or IPMs.
- $\mathcal{K}_i = \mathbb{Q}^{n_i}_+ = \{x \in \mathbb{R}^{n_i} : x_1 \geq ||x_{2:n_i}||\}$ **SOCP**
  Large SOCPs ($m, n \leq$ **100,000**) solvable by IPMs.
- $\mathcal{K}_i = \mathcal{S}^{n_i}_+ = \{X \in \mathcal{S}^{n_i} : X \succeq 0\}$ **SDP**
  Medium sized SDPs ($m, n \leq$ **1000**) solvable by IPMs.
  (Beyond 10,000 seems impossible today!)

Every convex program is a conic problem

1. Consider the following convex program with $B \in \mathbb{R}^{k \times n}$

$$
\begin{array}{lrcl}
\text{min} & c^T x & & \\
\text{s.t.} & x^T B^T B x + a^T x + b & \leq & 0 \\
 & x & \geq & 0
\end{array}
$$

2. Observe

$$
x^T B^T B x + a^T x + b \leq 0 \;\Leftrightarrow\; u \succeq_Q 0, \;\text{ with}
$$

$$
u_1 = \tfrac{1 - a^T x - b}{2} \;\text{ and }\; u_{2:k+2} = \begin{pmatrix} Bx \\ \frac{1 + a^T x + b}{2} \end{pmatrix}
$$

3. This gives the conic problem

$$
\begin{array}{ll}
\text{min} & c^T x + 0^T u \\
\text{s.t.} & \begin{pmatrix} B & 0 & -I & 0 \\ -a^T & 0 & 0 & 1 \\ a^T & 1 & 0 & 0 \end{pmatrix} \begin{pmatrix} x \\ u_1 \\ u_{2:k+1} \\ u_{k+2} \end{pmatrix} = \begin{pmatrix} 0 \\ \frac{1+b}{2} \\ \frac{1-b}{2} \end{pmatrix} \\
 & x \geq 0, \;\; u \succeq_Q 0
\end{array}
$$

Semidefinite programming

$$\text{(SDP)} \quad \max \; C \bullet X \quad \text{s.t.} \; \mathcal{A}(X) = b, \; X \succeq 0$$

$$\text{(SDD)} \quad \min \; b^T y \quad \text{s.t.} \; \mathcal{A}^T y - S = C, \; S \succeq 0$$

- **Notation**
  - $X, S, C \in \mathcal{S}^n$, $b \in \mathbb{R}^m$
  - $A \bullet B = \text{trace}(AB) = \sum_{i,j=1}^n A_{ij} B_{ij}$ (Frobenius inner product)
  - The operator $\mathcal{A} : \mathcal{S}^n \to \mathbb{R}^m$ and its adjoint $\mathcal{A}^T : \mathbb{R}^m \to \mathcal{S}^n$ are

$$\mathcal{A}(X) = \begin{pmatrix} A_1 \bullet X \\ \vdots \\ A_m \bullet X \end{pmatrix}, \quad \mathcal{A}^\top y = \sum_{i=1}^m y_i A_i$$

where $A_i \in \mathcal{S}^n$, $i = 1, \ldots, m$

Motivation

(a) Exploit the **sparsity** and/or **symmetry** in the underlying SDP and preprocess it into an equivalent SDP having a **block angular structure**.

(b) Solve block-angular SDP **iteratively** between a **conic master problem** over linear and smaller dimensional semidefinite cones; and **decomposed and distributed subproblems** (smaller SDPs) in a parallel 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 **parallel conic interior point decomposition** scheme.

Example 1: Combinatorial Optimization

1. Consider the following integer quadratic program

$$\begin{array}{ll} \max & x^T L x \\ \text{s.t.} & x_i \in \{-1,1\}^n, \;\; i=1,\ldots,n \end{array}$$

2. Setting $X = xx^T$, gives an **equivalent** formulation

$$\begin{array}{lrcl} \max & L \bullet X & & \\ \text{s.t.} & X_{ii} & = & 1, \;\; i=1,\ldots,n \\ & X & \succeq & 0 \\ & \text{rank}(X) & = & 1 \end{array}$$

3. Dropping the rank constraint gives an SDP **relaxation**

$$\begin{array}{lrcl} \max & L \bullet X & & \\ \text{s.t.} & X_{ii} & = & 1, \;\; i=1,\ldots,n \\ & X & \succeq & 0 \end{array}$$

4. Goemans and Williamson developed an 0.878 approximation algorithm for the maxcut problem which uses this SDP relaxation.

Example 2: Polynomial Optimization: 1

1. Consider the problem of minimizing Rosenbrock's function

$$\min \quad (1-x_1)^2 + 100(x_2 - x_1^2)^2 = 100x_1^4 - 200x_1^2x_2 + 100x_2^2 + x_1^2 - 2x_1 + 1$$

2. Consider all monomials in two variables x_1 and x_2 with degree less than or equal to 4.

3. We associate each monomial in the function with a new variable, i.e., $x_1 = y_{10}, \;\; x_2 = y_{01}, \;\; x_1^2 = y_{20}, \;\; x_1x_2 = y_{11}, \;\; \ldots, \;\; x_2^4 = y_{04}$ etc.

4. y_{ij} satisfy non-convex relations such as $y_{10}y_{01} = y_{11}$, $y_{10}y_{10} = y_{20}$ etc.

5. We relax these relations with the following constraint

$$\begin{pmatrix} 1 & y_{10} & y_{01} & y_{20} & y_{11} & y_{02} \\ y_{10} & y_{20} & y_{11} & y_{30} & y_{21} & y_{12} \\ y_{01} & y_{11} & y_{02} & y_{21} & y_{12} & y_{03} \\ y_{20} & y_{30} & y_{21} & y_{40} & y_{31} & y_{22} \\ y_{11} & y_{21} & y_{12} & y_{31} & y_{22} & y_{13} \\ y_{02} & y_{12} & y_{03} & y_{22} & y_{13} & y_{04} \end{pmatrix} \succeq 0$$

Example 2: Polynomial Optimization: 2

1. Consider the following relaxation

$$\begin{array}{ll} \min & 100y_{40} - 200y_{21} + 100y_{02} + y_{20} - 2y_{10} + 1 \\ \text{s.t.} & \begin{pmatrix} 1 & y_{10} & y_{01} & y_{20} & y_{11} & y_{02} \\ y_{10} & y_{20} & y_{11} & y_{30} & y_{21} & y_{12} \\ y_{01} & y_{11} & y_{02} & y_{21} & y_{12} & y_{03} \\ y_{20} & y_{30} & y_{21} & y_{40} & y_{31} & y_{22} \\ y_{11} & y_{21} & y_{12} & y_{31} & y_{22} & y_{13} \\ y_{02} & y_{12} & y_{03} & y_{22} & y_{13} & y_{04} \end{pmatrix} \succeq 0 \end{array}$$

2. This is an SDP with $m = 14$ and $n = 6$. Solution gives the global minimum $x^* = (1,1)$ with objective value 0.

3. Develop a hierarchy of relaxations as follows: The next relaxation contains all monomials in x_1 and x_2 of degree upto 6, i.e., $y_{10}, \ldots, y_{06}$.

4. The objective values of the SDPs in the hierarchy converge to the global minimum value of the polynomial optimization problem. There is also a solution extraction scheme (Lasserre, Henrion & Lasserre).

Semidefinite programs with a decomposable structure

1. **Preprocessed SDPs after matrix completion:**
   Fukuda-Kojima-Nakata-Murota (2000)
2. **Control and stability analysis of interconnected subsystems:**
   Langbort-D'Andrea-Xiao-Boyd (2003)
3. **Stochastic semidefinite programs:** Mehrotra-Ozëvin (2005)
4. **Polynomial optimization problems with symmetry/sparsity:**
   Parrilo-Gatermann (2004), Waki-Kim-Kojima-Muramatsu (2005)
5. **Exploiting group symmetry in SDPs:**
   Kanno et al. (2001), Parrilo-Gatermann (2004), De Klerk et al. (2005)

Semidefinite programming with block angular structure

$$
\begin{array}{lrcl}
\max & \sum_{i=1}^{r} C_i \bullet X_i & & \\
\text{s.t.} & \sum_{i=1}^{r} \mathcal{A}_i(X_i) & = & b \\
 & X_i & \in & \mathcal{C}_i, \quad i = 1, \ldots, r
\end{array}
$$

- **Notes**
  - $X_i, C_i \in \mathcal{S}^{n_i}$, $b \in \mathbb{R}^m$.
  - $\mathcal{C}_i = \{X_i : \mathcal{B}_i(X_i) = d_i, X_i \succeq 0\}$ - compact semidefinite feasibility sets.
  - The objective function and coupling constraints are **block separable**.
  - In the absence of coupling constraints, solve r independent problems

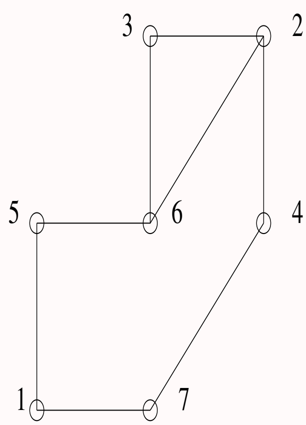
$$\max_{X_i \in \mathcal{C}_i} \; C_i \bullet X_i$$

Exploiting sparsity to get block-angular SDPs: 1

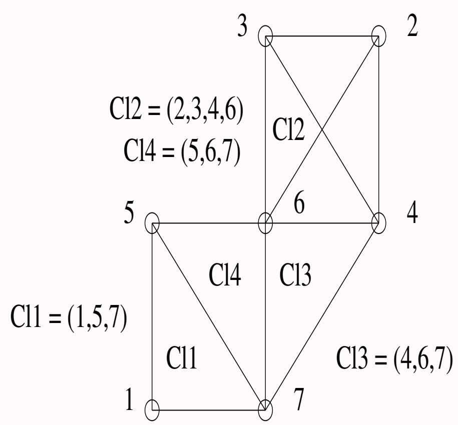
Consider the SDP

$$
\begin{array}{lrcl}
\max & L \bullet X & & \\
\text{s.t.} & X_{ii} & = & 1, \;\; i = 1, \ldots, n, \\
 & X & \succeq & 0,
\end{array}
$$

where L is the adjacency matrix of the graph



GRAPH



CHORDAL EXTENSION OF GRAPH

Exploiting sparsity: 2

Using **matrix completion**, one can reformulate the earlier SDP as

$$
\begin{array}{lrcl}
\max & \displaystyle\sum_{k=1}^{4} (L^k \bullet X^k) & & \\
\text{s.t.} & X^1_{23} - X^4_{13} & = & 0, \\
& X^2_{34} - X^3_{12} & = & 0, \\
& X^3_{23} - X^4_{23} & = & 0, \\
& X^k_{ii} & = & 1, \ \ i = 1, \ldots, |C_k|, \ \ k = 1, \ldots, 4, \\
& X^k & \succeq & 0, \ \ k = 1, \ldots, 4,
\end{array}
$$

which is in **block-angular** form.

Exploiting sparsity: 3

1. Construct the aggregate sparsity graph $G = (V, E)$ from data matrices $A_0 = C$ and A_i, $i = 1, \ldots, m$ of SDP. We have $V = \{1, \ldots, n\}$ and
$$E \;=\; \{(i,j) \in V \times V : \exists k \in \{0, 1, \ldots, m\} \text{ s.t. } (A_k)_{ij} \neq 0\}$$
2. Construct a minimal **chordal** extension $G' = (V, E')$ of $G = (V, E)$.
3. Find maximal cliques Cl_i, $i = 1, \ldots, k$ in $G' = (V, E')$. We have
$$X \succeq 0 \;\Leftrightarrow\; X_{Cl_i, Cl_i} \succeq 0, \;\; i = 1, \ldots, k$$
4. Each block in the block-diagonal SDP corresponds to a maximal clique.
5. Introduce additional equality constraints for common nodes and edges in the cliques ; for instance if cliques Cl_k and Cl_l share a common edge $\{i, j\}$ then $X^k_{ij} = X^l_{ij}$ etc.
6. In some special cases, the resulting block-diagonal SDP has a block-angular form.

Exploiting symmetry to get block-angular SDPs: 1

1. We are given a finite group G, elements $g \in G$ and its representation $\rho(g) : G \rightarrow GL(\mathcal{R}^n)$ (the set of real invertible matrices of size n). Also, assume $\rho(g)$ to be an orthogonal matrix.

2. We say that a SDP has the symmetry of group G if

   (a) X feasible in the SDP implies $\rho(g)^T X \rho(g)$ is feasible too.

   (b) $C\rho(g) = \rho(g)C, \ \forall g \in G$.

3. In this case, one can add the constraints

$$\rho(g)X \ = \ X\rho(g), \ \forall g \in G$$

   to the SDP without changing its objective value. So, C and X commute with every representation $\sigma(g)$ of G.

4. Schur's theorem suggests that one can compute a symmetry adapted basis T (an orthogonal matrix) such that $T^T C T$ and $T^T X T$ are block-diagonal.

Exploiting symmetry: 2

1. This gives an equivalent SDP

$$\begin{array}{lrcl} \max & \sum_{i=1}^{r} n_i(C_i \bullet X_i) & & \\ \text{s.t.} & \sum_{i=1}^{r} \mathcal{A}_i(X_i) & = & b \\ & X_i & \succeq & 0 \quad i = 1, \ldots, r \end{array}$$

   where r is the number of irreducible representations of G and n_i is the dimension of the ith irreducible representation. Also, $X_i \in \mathcal{S}^{m_i}$ where m_i is the multiplicity of the ith irreducible representation in $\sigma(g)$.

2. In most cases, the above block-diagonal SDP is actually in block-angular form.

3. In some cases, $m_i = 1$, $\forall i$ and the SDP reduces to an LP.

Exploiting symmetry: 3

1. Consider the SDP

$$
\begin{array}{lrcl}
\max & -X_{11} + X_{22} + X_{33} & & \\
\text{s.t.} & X_{12} & = & X_{13} \\
& \text{trace}(X) & = & 1 \\
& X & \succeq & 0.
\end{array}
$$

2. SDP unchanged under a simultaneous permutation of the last two rows and columns of X, i.e., has the symmetry of the permutation group S_2.

3. S_2 has two 1 dimensional irreducible representations 1 and -1 with multiplicities 2 and 1 in $\sigma(g)$ respectively

4. Apply a symmetry adapted transformation to get

$$
\begin{array}{lrcl}
\max & -X^1_{11} + X^1_{22} + X^2 & & \\
\text{s.t.} & \text{trace}(X^1) + X^2 & = & 1 \\
& X^1 & \succeq & 0 \\
& X^2 & \geq & 0
\end{array}
$$

an SDP with a semidefinite cone of size 2 and a linear cone of size 1.

The Lagrangian dual problem

- The Lagrangian dual problem is

$$\min_y \ \theta(y) = b^T y + \textstyle\sum_{i=1}^r \theta_i(y)$$

where

$$\theta_i(y) \ = \ \max_{X_i \in \mathcal{C}_i} \ (C_i - \mathcal{A}_i^T y) \bullet X_i$$

- Dual is an unconstrained **convex** but **nonsmooth** problem.
- Given y^k, we have $\theta(y^k) = b^T y^k + \sum_{i=1}^r (C_i - \mathcal{A}_i^T y^k) \bullet X_i^k$

  and a subgradient $g(y^k) = (b - \sum_{i=1}^r \mathcal{A}_i(X_i^k))$ where

$$X_i^k \ = \ \operatorname*{argmax}_{X_i \in \mathcal{C}_i} \ (C - \mathcal{A}_i^T y^k) \bullet X_i$$

  (these can be computed in parallel!)

Solving the Lagrangian dual

1. Construct a model $\theta^k(y)$ an **underestimate** for $\theta(y)$

$$\theta^k(y) \;=\; b^T y + \sum_{i=1}^r \max_{j=1,\ldots,J^k(i)} (C_i - \mathcal{A}_i^T y) \bullet X_i^j$$

   from the function values and subgradient information.

2. The **regularized** master problem then is

$$\min_y \;\; \theta^k(y) + \frac{u^k}{2}||y - x^k||^2$$

   where $u^k \geq 0$ and x^k is our current center (best iterate so far!)

3. The dual to this quadratic program (much easier to solve!) is

$$\begin{aligned} \max \;\; & -\frac{1}{2u^k}||\sum_{i=1}^r \sum_{j=1}^{J^k(i)} \mathcal{A}_i(X_i^j)\lambda_i^j - b||^2 + \sum_{i=1}^r \sum_{j=1}^{J^k(i)} ((C_i - \mathcal{A}_i^T x^k) \bullet X_i^j)\lambda_i^j \\ \text{s.t.} \;\; & \sum_{j=1}^{J^k(i)} \lambda_i^j \;=\; 1, \;\; i = 1, \ldots, r \\ & \lambda_i^j \;\geq\; 0, \;\; i = 1, \ldots, r, \;\; j = 1, \ldots, J^k(i). \end{aligned}$$

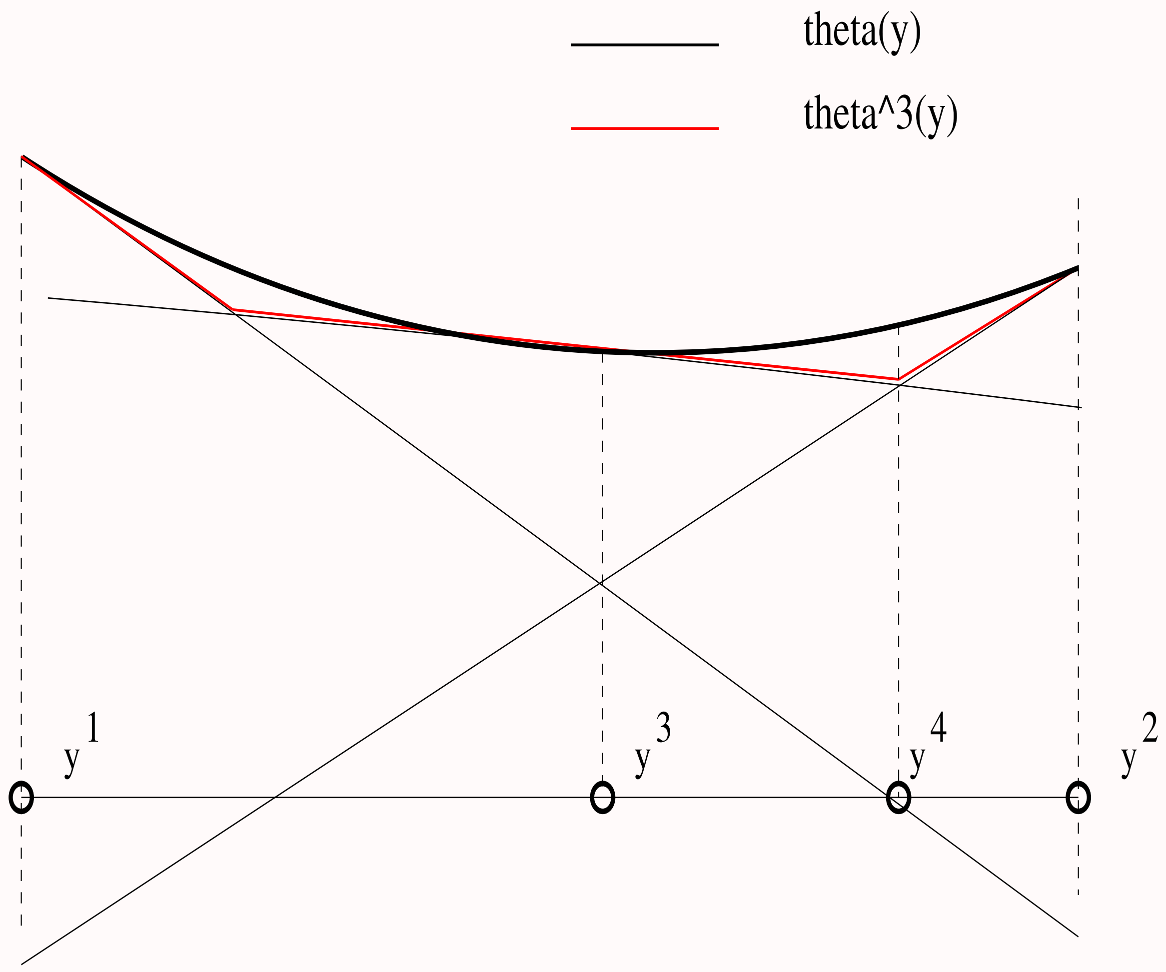


Figure 1: Solving the Lagrangian dual problem

The complete algorithm

1. **Initialize:** Set $k = 1$, $J^1(i) = \phi$, $z_i^1 = -\infty$, $i = 1, \ldots, r$. Choose $u^1 > 0$, $y^1 \in \mathbb{R}^m$; and set $x^1 = y^1$, and $\theta^1(y^1) = -\infty$.

2. **Solve subproblems in parallel:** For $i = 1, \ldots, r$, solve ith subproblem with $y = y^k$ for optimal objective value $\theta_i(y^k)$.

3. **Update model function:** If $\theta_i(y^k) > z_i^k$, update the ith model function $\theta_i^{k+1}(y)$ and set $J^k(i) = J^{k-1}(i) \cup \{k\}$. Else, $J^k(i) = J^{k-1}(i)$.

4. **Update the center x^k:** If $k = 1$ or if

$$\theta(y^k) \;\leq\; (1-\gamma)\theta(x^{k-1}) + \gamma\theta^k(y^k)$$

   then set $x^k = y^k$ (serious step); otherwise set $x^k = x^{k-1}$ (null step).

5. **Solve master problem:** Solve dual master QP for λ_i^j and $z^{k+1} = \left(z_1^{k+1}, \ldots, z_r^{k+1}\right)$ (dual variables). Compute y^{k+1} using

$$y^{k+1} \;=\; x^k + \tfrac{1}{u^k}\left(\sum_{i=1}^r \sum_{j=1}^{J^k(i)} \mathcal{A}_i(X_i^j)\lambda_i^j - b\right)$$

   and let $\theta^{k+1}(y^{k+1}) = b^T y^{k+1} + \sum_{i=1}^r z_i^{k+1}$. If $\theta^{k+1}(y^{k+1}) = \theta(x^k)$, we are done! Else, set $k = k+1$ and return to Step 2.

General case: Conic master problem

1. For some blocks, $\mathcal{C}_i = \{X_i : \text{trace}(X_i) = 1, \;\; X_i \succeq 0\}$. We will assume that the first s of the r blocks are of this type.

2. The function $\theta(y)$ is

$$\theta(y) = b^T y + \sum_{i=1}^{s} \lambda_{\max}(C_i - \mathcal{A}_i^T y) + \sum_{i=s+1}^{r} \max_{X_i \in \mathcal{C}_i} \; (C_i - \mathcal{A}_i^T y) \bullet X_i$$

3. For the first s blocks, the subproblem is **Lanczos** solver that computes the maximum eigenvalue of $(C_i - \mathcal{A}_i^T y)$ and its associated eigenspace (fast and very easy to implement!).

4. Our model function $\theta^k(y)$ is then

$$\begin{aligned} \theta^k(y) \;\; = \;\; & b^T y + \sum_{i=1}^{s} \max_{j=1,\ldots,J^k(i)} \lambda_{\max}(P_i^j (C_i - \mathcal{A}_i^T y) P_i^j) + \\ & \sum_{i=s+1}^{r} \max_{j=1,\ldots,J^k(i)} (C_i - \mathcal{A}_i^T y) \bullet X_i^j \end{aligned}$$

   where P_i^j contains the eigenspace associated with $\lambda_{\max}(C_i - \mathcal{A}_i^T y^j)$.

5. This gives rise to a master problem which is a quadratic conic problem over linear, and smaller dimensional semidefinite cones.

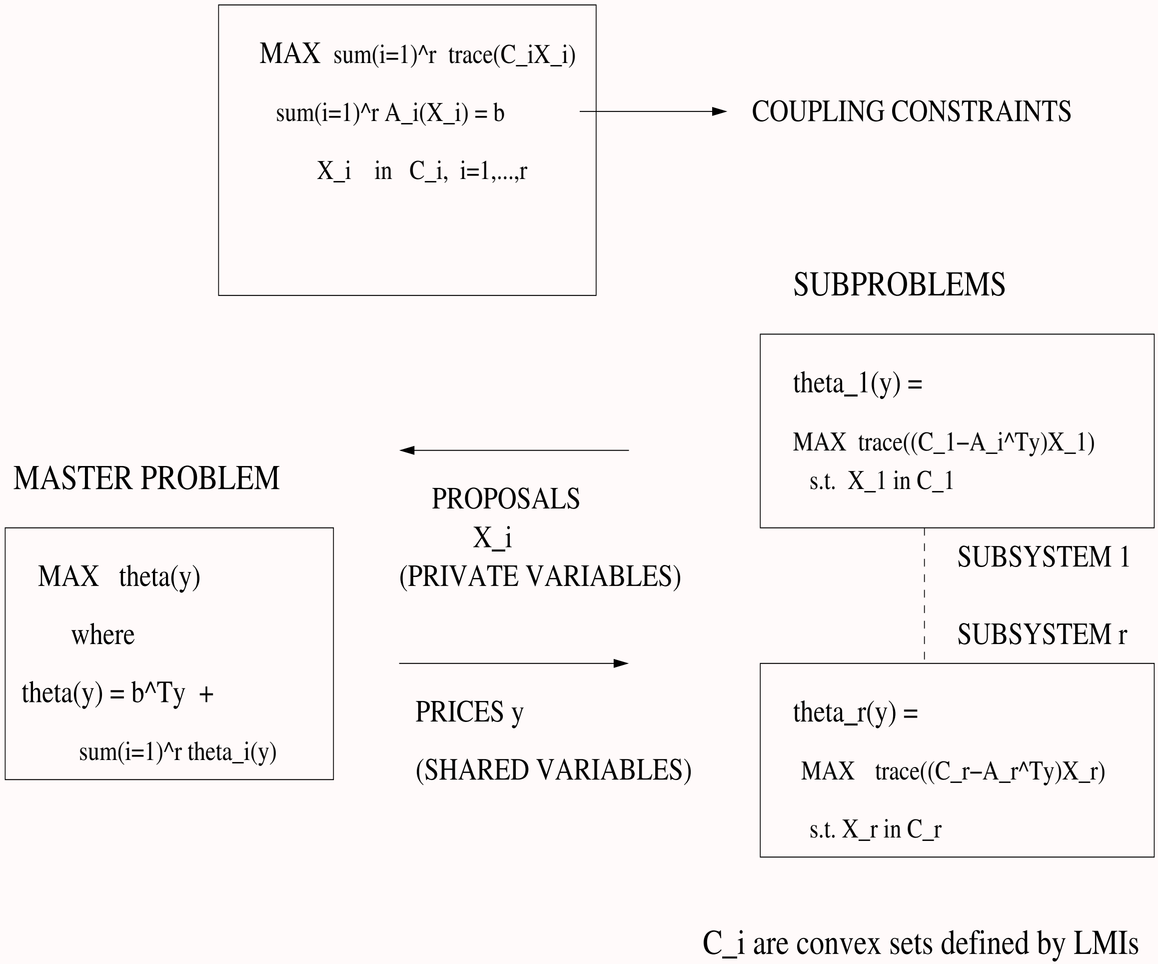


Figure 2: Decomposition by prices

Computational Results: 1

- 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 used to solve the master problem and CSDP 5.0 (Borchers) used to solve the subproblems.
- 3 digits of accuracy or an upper limit of 2 hours in computations.
- Used updating scheme due to Kiwiel to adjust the weight u.
- Employed aggregation scheme of Kiwiel to limit size of master problem.
- Warm-start master problem and subproblems.
- Tested on SDPs from SDPLIB and DIMACS repositories.
- Details can be found in Sivaramakrishnan (2006).

Computational Results: 2

Prob	n	Opt value	n_p	m_p	p	Our UB	Time (h:m:s)
mcp124-1	124	141.99	60(6)	172(43)	6	142.05	2
mcp250-1	250	317.26	176(9)	381(124)	9	317.36	34
mcp500-1	500	598.15	204(63)	1584(951)	10	598.71	38
maxG11	800	629.16	216(4)	1208(360)	4	629.51	47
maxG11	800	629.16	176(38)	2324(1228)	4	629.41	2:26
maxG51	1000	4003.81	971(24)	1677(517)	10	4008.95	38:5
maxG32	2000	1567.64	1022(3)	2830(786)	3	1568.78	8:46
maxG32	2000	1567.64	526(8)	5310(3143)	4	1568.57	15:46
qpG11	1600	2448.66	432(4)	1256(408)	4	2450.32	2:48
qpG11	1600	2448.66	254(83)	4515(3068)	8	2450.21	2:2
qpG32	4000	6226.55	1054(14)	5566(3370)	10	6229.84	31:28
qpG32	4000	6226.55	432(97)	16402(13318)	10	6229.88	47:44

Timing results

Prob	n_p	m_p	p	Serious steps	Null steps	Master (h:m:s)	Subproblem (h:m:s)
maxG11	216(4)	1208(360)	4	7	29	1	46
maxG11	176(38)	2324(1228)	4	13	65	21	2:05
maxG32	1022(3)	2830(786)	3	5	6	1	8:45
maxG32	526(8)	5310(3143)	4	11	50	30	15:15
qpG11	432(4)	1256(408)	4	7	16	1	2:47
qpG11	254(83)	4515(3068)	4	13	32	19	2:56
qpG32	1054(14)	5566(3370)	10	9	67	25	31.03
qpG32	432(97)	16402(13318)	10	15	160	30:04	17.40

Scalability of algorithm on test problems on up to 8 processors

Prob	n_p	m_p	1	2	4	8
maxG11	216(4)	1208(360)	2:47	1:28	47	-
maxG11	176(38)	2324(1228)	4:44	4:1	2:26	1:47
maxG32	1022(3)	2830(786)	14:47	8:46	-	-
maxG32	526(8)	5310(3143)	56:23	29:18	15:46	-
qpG11	432(4)	1256(408)	7:33	4:28	2:28	-
qpG11	254(83)	4515(3068)	6:24	4:8	2:21	2:2

Thank you for your attention!.

Questions, Comments, Suggestions ?

The slides from this talk are available online at

http://www4.ncsu.edu/∼kksivara/publications/kartik-umn.pdf

A technical report appears at

http://www4.ncsu.edu/∼kksivara/publications/parallel-conic-blockangular.pdf

Bibliography: 1

1. K. Sivaramakrishnan, *A parallel conic interior point decomposition approach for block angular semidefinite programs*, submitted. Available at
`http://www4.ncsu.edu/~kksivara/publications/parallel-conic-blockangular.pdf`

2. K. Sivaramakrishnan, *A parallel conic interior point decomposition approach for polynomial optimization problems*, under preparation.

3. K. Sivaramakrishnan, G. Plaza, and T. Terlaky, *A conic interior point decomposition approach for large scale semidefinite programming*, submitted.
`http://www4.ncsu.edu/~kksivara/publications/conic-ipm-decomposition.pdf`

4. J.B. Lasserre, *Global optimization with polynomials and the problem of moments*, SIAM Journal on Optimization, 11(2001), pp. 796-817.

5. M. Fukuda, M. Kojima, K. Murota, and K. Nakata, *Exploiting sparsity in semidefinite programming via matrix completion I: General framework*, SIAM Journal on Optimization, 11(2000), pp. 647-674.

6. K. Gatermann and P.A. Parrilo, *Symmetry groups, semidefinite programs, and sums of squares*, Journal of Pure and Applied Algebra 192(2004), pp. 95-128.

7. H. Waki, S. Kim, M. Kojima, and M. Muramatsu, *Sums of squares and semidefinite programming relaxations for polynomial problems with structured sparsity*, SIAM Journal on Optimization, 17(2006), pp. 218-242.

8. S. Mehrotra and M.G. Özevin, *On the implementation of interior point decomposition algorithms for two-stage stochastic programs*. Available at
`http://www.iems.northwestern.edu/images/PDF/SDP_computational.pdf`.

Bibliography: 2

9. C. Langbort, R. D'Andrea, L. Xiao, and S. Boyd, *A decomposition approach to distributed analysis of networked systems*, Proceedings of the 43rd IEEE Conference on Decision and Control, pp. 3980-3985.

10. D. Henrion and J.B. Lasserre, *GLOPTIPOLY: Global optimization over polynomials with MATLAB and SeDuMi*, `http://www.laas.fr/~henrion/papers/gloptipoly.pdf`

11. E. de Klerk, *Exploiting algebraic symmetry in semidefinite programming: a review*, Talk at Delft, the Netherlands, October, 2005. `http://www4.ncsu.edu/~kksivara/etienne-symmetry.pdf`

12. A. Fässler and E. Stiefel, *Group Theoretical Methods and Their Applications*, Birkhäuser, Boston, 1992.

13. M. Goemans and D.P. Williamson, *Improved approximation algorithms for max cut and satisfiability problems using semidefinite programming*, J. A.C.M 42 (1995), pp. 1115-1145.

14. K.C. Kiwiel, *Proximity control in bundle methods for convex nondifferentiable optimization*, Mathematical Programming, 46(1990), pp. 105-122.

15. B. Borchers, *CSDP, a C library for semidefinite programming*, Optimization Methods and Software, 11 & 12(1999), pp. 613-623. `http://infohost.nmt.edu/~borchers/csdp.html`.

16. B. Borchers, *SDPLIB 1.2, A library of semidefinite programming test problems*, Optimization Methods and Software, 11, 1999, pp. 683-690. `http://infohost.nmt.edu/~sdplib/`

17. DIMACS 7th challenge. `http://dimacs.rutgers.edu/Challenges/Seventh/Instances/`