ARGONNE NATIONAL LABORATORY
9700 South Cass Avenue
Argonne, Illinois 60439

A Multilevel Approach to Class of Semidefinite Programs

Fu Lin, Zichao Di, and Sven Leyffer

Mathematics and Computer Science Division

Preprint ANL/MCS-P5402-0915

September, 2015

This material is based upon work supported by the U.S. Department of Energy, Office of Science, Office of Advanced Scientific Computing Research, under Contract DE-AC02-06CH11357.

A Multilevel Approach to Class of Semidefinite Programs

Fu Lin, Zichao Di, and Sven Leyffer

***Abstract*— We consider a class of semidefinite programs that arises from combinatorial optimization problems on graphs. We propose a multilevel approach that produces a sequence of progressively coarser problems by coarsening the underlying graphs. We use the solution of each coarse problem to provide an initial approximation to the solution at a finer level. At the coarsest level we employ Newton's method for high-accuracy solutions, and at finer levels we take advantage of the inexpensive coordinate descent updates. We coarsen the graph based on an algebraic distance that can be computed efficiently. Furthermore, our coarsening scheme preserves the properties of the graph Laplacian matrix between fine and coarse levels. Numerical experiments indicate the competitiveness of the hybrid multilevel approach compared with state-of-the-art SDP solvers for both synthetic graphs and real-world power networks.**

Keywords: Algebraic distance, coarsening graphs, coordinate descent, multilevel methods, power networks, semidefinite programs.

I. Introduction

Semidefinite programs (SDPs) are playing an increasingly important role in optimization and control. In many modern applications on networks, the underlying graphs are large but sparse. Multilevel methods have been successful in optimization problems involving these large, sparse graphs [1]–[4].

In this paper, we develop a multilevel approach for a class of SDPs that arise from combinatorial optimization on graphs. We build a sequence of graphs that approximate the original graph. Consequently, we obtain a hierarchy of approximate problems whose solutions are used to initialize iterative methods on the finer-level problems. This multilevel process is repeated until the solution of the original problem is found.

An advantage of the multilevel framework is its degree of freedom in choosing iterative methods at different problem levels. At the coarsest level with small problem sizes, we employ Newton's method to obtain highly accurate solutions. At the finer levels, we take advantage of inexpensive coordinate descent updates to find approximate solutions.

Coordinate descent methods have received renewed interest in recent years, starting from Nesterov's seminal paper [5]. Their simplicity and potential for parallelism make coordinate descent methods well suited to many modern applications with large data sets [6], [7]. The use of coordinate descent methods for SDPs has appeared recently in the literature [8], [9]. However, contrast to the latter approaches [8], [9] that focus on the primal formulation of SDPs, we apply coordinate descent methods to the dual formulation. This allows us to exploit the sparsity structure of the underlying graphs. Furthermore, the graph structure provides insight into the restriction and prolongation schemes that are essential components of the multilevel methods.

A crucial component of a multilevel method is its coarsening scheme. The challenge of designing a coarsening scheme is to strike a balance between the interpolation accuracy and the computation efficiency [1], [3]. We propose a coarsening scheme based on the algebraic distance [2], [3] that can be computed efficiently. Our coarsening scheme also preserves basic properties (e.g., positive semidefiniteness and zero row-sum) of the graph Laplacian matrices.

Our presentation is organized as follows. In Section II, we discuss the advantage of using the dual formulation of the class of SDPs. In Section III, we introduce a restriction/prolongation scheme and show its basic properties. In particular, our restriction/prolongation scheme preserves the graph Laplacian structure between the fine and coarse levels. In Section IV, we propose a multilevel framework in which Newton's method and coordinate descent methods are employed at different levels. In Section V, we compare the performance of our multilevel method with SDPT3 for synthetic graphs and real power networks. In Section VI, we conclude the paper by summarizing our contributions.

II. Problem Statement

Recall the standard form of the semidefinite programs

$$\begin{array}{ll} \text{maximize} & \mathbf{Tr}(LX) \\ \text{subject to} & \mathcal{A}(X) = b \\ & X \succeq 0, \end{array}$$

where $X \in \mathcal{S}^n$ is the decision variable, $L \in \mathcal{S}^n$ and $b \in \mathbb{R}^m$ are the problem data, $\mathcal{S}^n$ denotes the set of n-by-n symmetric matrices, $\mathcal{A} : \mathbb{R}^{n \times n} \to \mathbb{R}^m$ is a

Fu Lin, Zichao Di, and Sven Leyffer are with the Mathematics and Computer Science Division, Argonne National Laboratory, Argonne, IL 60439. E-mail:{fulin, wendydi, leyffer}@mcs.anl.gov.

This material is based upon work supported by the U.S. Department of Energy, Office of Science, Office of Advanced Scientific Computing Research, Applied Mathematics program under contract number DE-AC02-06CH11357.

linear operator, $\mathbf{Tr}(LX) := \sum_{i,j=1}^{n} L_{ij}X_{ij}$, and $X \succeq 0$ denotes positive semidefinite matrices.

In this paper, we focus on a special class of linear operators, namely,

$$\mathcal{A}(X) = \text{diag}(X),$$

which takes the diagonal of X into $\mathbb{R}^n$. Thus, we consider

$$\begin{array}{ll} \text{maximize} & \mathbf{Tr}(LX) \\ \text{subject to} & \text{diag}(X) = b \\ & X \succeq 0. \end{array} \quad (1)$$

Note that b must be elementwise nonnegative for (1) to be feasible. This class of SDPs arises from several applications, including the classic max-cut problem [10], the phase retrieval problem [9], and several combinatorial optimization problems [11].

The data matrix L is typically large but sparse in combinatorial problems on graphs. In this context, the sparsity structure of the graph Laplacian matrix L is determined by the graph structure. To take advantage of sparsity, we follow [11] and consider the dual problem

$$\begin{array}{ll} \text{minimize} & b^T y \\ \text{subject to} & Z = \text{Diag}(y) - L \\ & Z \succeq 0, \end{array} \quad (2)$$

where $Z \in \mathcal{S}^n$ and $y \in \mathbb{R}^n$ are the dual variables and $\text{Diag}(y)$ is a diagonal matrix with y on the main diagonal. Therefore, the dual variable Z has the same sparsity structure as L. This is in contrast to the dense matrix X in the primal formulation (1).

Introducing a log-barrier function for the positive semidefinite cone, we approximate (2) with an unconstrained problem

$$\text{minimize} \;\; f_\mu(y) = b^T y - \mu \log\det(\text{Diag}(y) - L), \quad (3)$$

where the parameter μ belongs to $(0, \infty)$ and a smaller μ implies a more accurate approximation [12]. Therefore, solving a sequence of barrier problems (3) with $\mu \to 0$ provides a sequence of solutions converging to the solution of (2).

III. Restriction and Prolongation

In this section, we introduce a restriction and prolongation scheme that preserves the basic properties of graph Laplacian matrices. We show that the Laplacian matrix of a graph under the restriction or prolongation operation is the Laplacian matrix of another graph. Furthermore, we show that the dual (resp. primal) feasibility is preserved under the restriction (resp. prolongation) operator.

A. Restriction

Given a Laplacian matrix $L \in \mathcal{S}_+^n$, we require that the coarsened matrix $L_c := \mathcal{R}(L)$ be symmetric, positive semidefinite with zero row-sums

$$L_c \in \mathcal{S}_+^r, \quad L_c \mathbf{1}_r = 0, \quad (4)$$

where $\mathbf{1}_r$ denotes the vector of all ones. Consider a restriction matrix with binary elements $R \in \{0,1\}^{n \times r}$ such that

$$R\mathbf{1}_r = \mathbf{1}_n, \quad \text{rank}(R) = r. \quad (5)$$

The dimension after the restriction is $r < n$, and typically $r \approx n/2$.

Lemma 3.1: Let L be a graph Laplacian matrix (i.e., $L \in \mathcal{S}_+^n$ and $L\mathbf{1}_n = 0$). Let the restriction matrix R be defined in (5). Then $L_c = R^T L R$ satisfies (4).

Proof: By construction, $L_c = R^T L R \succeq 0$. To show that L_c has zero row-sums, we compute $L_c \mathbf{1}_r = R^T L R \mathbf{1}_r = R^T L \mathbf{1}_n = 0$. ■

It is instructive to study the sparsity structure of the restriction matrix R. Let the *support* of a vector $v \in \mathbb{R}^n$ be the index set of its nonzero elements

$$\text{supp}(v) := \{i \,|\, v_i \neq 0\}.$$

Then we have the following characterization of R.

Proposition 3.1: Let $R = [\, R_1 \, R_2 \cdots R_r \,] \in \mathbb{R}^{n \times r}$ be a matrix whose elements are either 0 or 1. Then (5) holds if and only if

$$\begin{array}{rll} \text{supp}(R_i) & \neq \emptyset, & i = 1, 2, \ldots, r \quad (6a) \\ \text{supp}(R_i) \cap \text{supp}(R_j) & = \emptyset, & \forall i \neq j \quad (6b) \\ \cup_{i=1}^r \text{supp}(R_i) & = \{1, 2, \ldots, n\}. & \quad (6c) \end{array}$$

Proof: We begin with the (5) ⇒ (6) part. Since R has full-column rank, no column of R is identically zero; hence condition (6a) holds. Since elements of R are binary and $R\mathbf{1}_r = \mathbf{1}_n$, it follows that each row of R has one and only one nonzero element. Therefore, the support of two columns of R is exclusive as in (6b), and the support of all columns covers all indices as in (6c).

For the (5) ⇐ (6) part, we note that (6b) implies $R\mathbf{1}_r \leq \mathbf{1}_n$ and the equality is attained when (6c) holds. The proof is complete by noting that $\text{rank}(R) = r$ follows from (6b) and (6a). ■

For example, a restriction matrix that combines node 1 and node 3 in a 3-node graph is given by

$$R = \begin{bmatrix} 1 & 0 \\ 0 & 1 \\ 1 & 0 \end{bmatrix}, \quad R^T R = \begin{bmatrix} 2 & 0 \\ 0 & 1 \end{bmatrix}.$$

Thus, coarsening a 3-node complete graph yields

$$L = \begin{bmatrix} 2 & -1 & -1 \\ -1 & 2 & -1 \\ -1 & -1 & 2 \end{bmatrix}, \quad R^T L R = \begin{bmatrix} 2 & -2 \\ -2 & 2 \end{bmatrix}.$$

Note that the coarsened 2-node graph has its edge weight equal to the sum of the two edges in the original graph; see Fig. 1.

The columns of R are orthogonal. In fact, columns of R are orthogonal with respect to any diagonal matrix, that is,

$$R^T \mathrm{Diag}(v) R \;=\; \mathrm{Diag}(u), \tag{7}$$

where $v \in \mathbb{R}^n$, $u \in \mathbb{R}^r$, and

$$u_i \;=\; \sum_{k \,\in\, \mathrm{supp}(R_i)} v_k, \quad i = 1, 2, \ldots, r.$$

In particular, setting $v = \mathbf{1}_n$ in (7), we get

$$R^T R \;=\; \mathrm{Diag}(u),$$

where

$$u_i \;=\; \mathbf{1}_n^T R_i, \quad i = 1, 2, \ldots, r.$$

In other words, the ith element of u is equal to the number of nonzero elements in the ith column of R. Therefore,

$$u \;=\; R^T \mathbf{1}_n. \tag{8}$$

B. Prolongation

We next consider the prolongation operator that maps $\mathbb{R}^{r\times r}$ back to $\mathbb{R}^{n\times n}$. Let

$$P \;:=\; RW, \tag{9}$$

where W is a diagonal weight matrix defined as

$$W \;:=\; (R^T R)^{-1} \;=\; (\mathrm{Diag}(u))^{-1}. \tag{10}$$

By definition, we have

$$R^T P \;=\; R^T R W \;=\; I_r \tag{11}$$

and

$$P^T P \;=\; W R^T R W \;=\; W. \tag{12}$$

We next show that the prolongation of a Laplacian matrix is still a graph Laplacian.

Lemma 3.2: Let L_c be a graph Laplacian matrix, that is, $L_c \in \mathcal{S}^r_+$ and $L_c \mathbf{1}_r = 0$. Then $L = P L_c P^T$ is a graph Laplacian matrix.

Proof: By construction, we have $L = P L_c P^T \succeq 0$ and

$$L\mathbf{1}_n \;=\; P L_c W R^T \mathbf{1}_n \;=\; P L_c W u,$$

where we used (9) to get the first equality and (8) to get the second equality. From (10), it follows that $Wu = \mathbf{1}_r$. Therefore, $L\mathbf{1}_n = P L_c \mathbf{1}_r = 0$. ■

For example, a prolongation matrix P from a 2-node graph to a 3-node graph is given by

$$P = \begin{bmatrix} \frac{1}{2} & 0 \\ 0 & 1 \\ \frac{1}{2} & 0 \end{bmatrix}, \; R = \begin{bmatrix} 1 & 0 \\ 0 & 1 \\ 1 & 0 \end{bmatrix}, \; P^T R = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}. \tag{13}$$

Figure 1 illustrates that the prolongation from a 2-node graph yields a 3-node graph with a negative edge:

$$L_c = \begin{bmatrix} 2 & -2 \\ -2 & 2 \end{bmatrix}, \; P L_c P^T = \begin{bmatrix} 1/2 & -1 & 1/2 \\ -1 & 2 & -1 \\ 1/2 & -1 & 1/2 \end{bmatrix}.$$

The restriction operator is the inverse of the prolongation operator, but not vice versa. Let $L = P L_c P^T$ be the prolongation of L_c, and let $\bar{L}_c = R^T L R$ be the restriction of L. From (11), it follows that $\bar{L}_c = R^T P L_c P^T R = L_c$. The reverse is not true; that is, $L \neq P R^T L R P^T$.

C. Primal and Dual Feasibility

We next show that dual feasibility is preserved under restriction. Let $\mathcal{Z}$ and $\mathcal{Z}_c$ be the dual feasibility set at the fine and the coarse level, respectively:

$$\begin{aligned} \mathcal{Z} \;&:=\; \{Z \in \mathcal{S}^n \,|\, Z \;=\; \mathrm{Diag}(y) \;-\; L\}, \\ \mathcal{Z}_c \;&:=\; \{Z_c \in \mathcal{S}^r \,|\, Z_c \;=\; \mathrm{Diag}(y_c) \;-\; L_c\}. \end{aligned}$$

Lemma 3.3: Let $Z_c = R^T Z R$, $L_c = R^T L R$, and $y_c = \mathrm{diag}(R^T\,\mathrm{Diag}(y)\,R)$. If $Z \in \mathcal{Z}$, then $Z_c \in \mathcal{Z}_c$.

Proof: Since R is orthogonal with respect to a diagonal matrix (see (7)), it follows that

$$\mathrm{Diag}(y_c) \;=\; R^T \mathrm{Diag}(y) R.$$

Thus, we have

$$Z_c \;=\; R^T \mathrm{Diag}(y) R \;-\; R^T L R \;=\; \mathrm{Diag}(y_c) \;-\; L_c.$$

■

We conclude this section by showing that primal feasibility is preserved under prolongation. Let $\mathcal{X}$ and $\mathcal{X}_c$ be the feasibility set of the primal formulation (1) at the fine and the coarse level, respectively:

$$\begin{aligned} \mathcal{X} \;&:=\; \{X \in \mathcal{S}^n \,|\, \mathrm{diag}(X) \;=\; b\}, \\ \mathcal{X}_c \;&:=\; \{X_c \in \mathcal{S}^r \,|\, \mathrm{diag}(X_c) \;=\; b_c\}. \end{aligned}$$

Lemma 3.4: Let $X = P X_c P^T$, and let $b = \mathrm{diag}(P\,\mathrm{Diag}(b_c)\,P^T)$. If $X_c \in \mathcal{X}_c$, then $X \in \mathcal{X}$.

Proof: We begin by noting that $\mathrm{diag}(X_c) = b_c$ can be written as $X_c \circ I = \mathrm{Diag}(b_c)$, where $\circ$ denotes the Hadamard product and I denotes the identity matrix. Let J be the matrix of all ones. Then we have

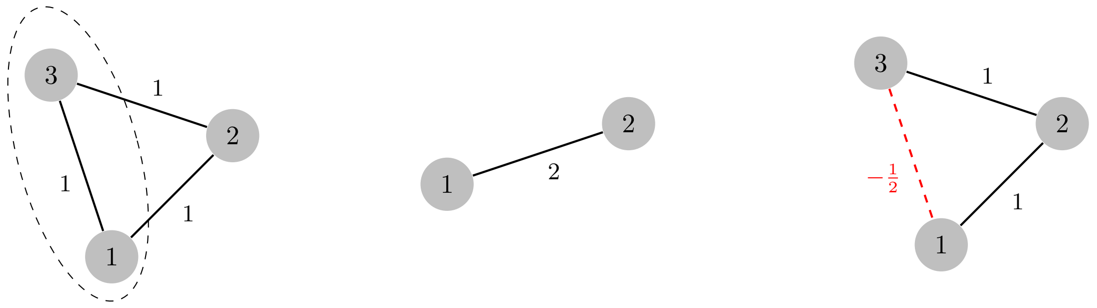
$$X_c \circ I \;=\; X_c \;-\; X_c \circ (J - I).$$

Substituting into $b = \mathrm{diag}(P\,(X_c \circ I)\,P^T)$ yields

$$b \;=\; \mathrm{diag}(X) \;-\; \mathrm{diag}(P(X_c \circ (J - I))P^T). \tag{14}$$

Then it suffices to show that the second term in (14) is zero, that is, the diagonal of $M = P(X_c \circ (J - I))P^T$ is identically zero.

Since each row of R has only one nonzero element (see Proposition 3.1) and since W is a diagonal



$$L = \begin{bmatrix} 2 & -1 & -1 \\ -1 & 2 & -1 \\ -1 & -1 & 2 \end{bmatrix} \qquad L_c = R^T L R = \begin{bmatrix} 2 & -2 \\ -2 & 2 \end{bmatrix} \qquad P L_c P^T = \begin{bmatrix} 1/2 & -1 & 1/2 \\ -1 & 2 & -1 \\ 1/2 & -1 & 1/2 \end{bmatrix}$$

Fig. 1: A 3-node graph (left), the coarsened 2-node graph via aggregation of nodes 1 and 3 (center), and the prolongation back to another 3-node graph (right). Note that a negative edge weight results from prolongation. Here, the restriction matrix R and the prolongation matrix P are given by (13).

matrix in the definition of $P = RW$, it follows that each row of P has only one nonzero element. Let P_i be the ith row of P, and let $s_i = \operatorname{supp}(P_i)$. Then

$$M_{s_i s_i} \;=\; P_i(X_c \circ (J - I))P_i^T \;=\; (X_c \circ (J - I))_{s_i s_i}.$$

Since the main diagonal of $X_c \circ (J - I)$ is identically zero by construction, we conclude that $M_{s_i s_i} = 0$ for $i = 1, 2, \ldots, n$. This completes the proof. ■

IV. A Multilevel Method

Similar to standard multigrid methods [13], our multilevel approach starts with the original problem and builds a sequence of approximate problems with smaller sizes. We coarsen the problem until it can be solved efficiently (e.g., using Newton's method). We then prolongate the coarse level solution back to a finer level and initialize an iterative scheme (e.g., the coordinate descent method) to find an approximate solution at the finer level. We repeat this process until we reach the finest level and solve the original problem of interest.

Our multilevel framework for the log-barrier function minimization problem (3) is given by Algorithm 1. We describe the main components, namely, the coordinate descent method and the coarsening scheme, in Sections IV-A and IV-B, respectively.

Since we employ Newton's method (e.g., see [12, Chapter 9.5]) at the coarsest level, we next provide expressions for the gradient and the Hessian of the log-barrier function $f_\mu(y)$ in (3). Let

$$S(y) \;:=\; (\operatorname{Diag}(y) - L)^{-1}. \tag{15}$$

Then the first-order and the second-order derivatives of $f_\mu(y)$ are given, respectively, by

$$\begin{aligned} \nabla f_\mu(y) &= b - \mu \operatorname{diag}(S(y)), \\ \nabla^2 f_\mu(y) &= \mu\,(S(y) \circ S(y)), \end{aligned}$$

where $\circ$ denotes the Hadamard product of two matrices. Recall that $\mu > 0$ and $b_i \geq 0$ for $i = 1, \ldots, n$.

Algorithm 1 A Multilevel Algorithm for (3).

```
Inputs: Problem data $L \succeq 0$, $b \geq 0$, $\mu > 0$ and an initial guess $y^0 \in \mathbb{R}^n$ such that $\operatorname{Diag}(y^0) - L \succ 0$.
Outputs: An approximate solution $y^\star$ for (3).
if The problem size $n \leq n_{\mathrm{NT}}$ then
    Solve (3) using Newton's method.
else
    Apply coordinate descent (see Algorithm 2)
        $y^{\mathrm{CD}} \leftarrow \mathbf{CD}(y^0)$.
    Coarsen the problem
        $L_c \leftarrow R^T L R$,
        $b_c \leftarrow \operatorname{diag}(R^T \operatorname{Diag}(b)\, R)$,
        $y_c^0 \leftarrow \operatorname{diag}(R^T \operatorname{Diag}(y^{\mathrm{CD}})\, R)$.
    Call Algorithm 1 for the coarsened problem
        $y_c \leftarrow \mathbf{Multilevel}(L_c, b_c, y_c^0)$.
    Prolongate the solution back to a finer level
        $y_p \leftarrow \operatorname{diag}(P \operatorname{Diag}(y_c)\, P^T)$.
    Refine the solution via coordinate descent
        $y^\star \leftarrow \mathbf{CD}(y_p)$.
```

A. Coordinate Descent

We employ the coordinate descent (CD) method because of its simplicity and effectiveness [5] and its ability for parallelism [7]. Since the log-barrier function (3) is convex, the minimization with respect to the

ith coordinate is equivalent to solving $\frac{\partial f_\mu}{\partial y_i} = 0$. We fix y_j for $j \neq i$ and seek a solution to the first-order condition for optimality, which can be written as

$$\left((\mathrm{Diag}(y) + L + \Delta y_i e_i e_i^T)^{-1}\right)_{ii} = b_i/\mu.$$

For the log-barrier problem (3), we next show each CD step can be solved analytically. Furthermore, the solution is guaranteed to satisfy the positive semidefinite constraint; see Lemma 4.1.

Lemma 4.1: Let y be such that[1] $S = (\mathrm{Diag}(y) - L)^{-1} \succ 0$, and let e_i be the ith coordinate vector of $\mathbb{R}^n$. The solution of the nonlinear scalar equation in Δy_i,

$$\left(S^{-1} + \Delta y_i e_i e_i^T\right)^{-1}_{ii} = b_i/\mu,$$

is unique and is given by

$$\Delta y_i = \frac{\mu}{b_i} - \frac{1}{S_{ii}}. \tag{16}$$

Furthermore, the resulting matrix is positive definite:

$$S^{-1} + \Delta y_i e_i e_i^T \succ 0. \tag{17}$$

Proof: Using the Sherman-Morrison formula, we have

$$\begin{aligned} b_i/\mu &= e_i^T \left(S^{-1} + \Delta y_i e_i e_i^T\right)^{-1} e_i \\ &= e_i^T \left(S - \frac{\Delta y_i S e_i e_i^T S}{1 + \Delta y_i e_i^T S e_i}\right) e_i \\ &= S_{ii} - \frac{\Delta y_i S_{ii} S_{ii}}{1 + \Delta y_i S_{ii}} \\ &= \frac{S_{ii}}{1 + \Delta y_i S_{ii}}. \end{aligned}$$

Taking the inverse on both sides yields (16).

Since $S \succ 0$, the positive definiteness condition (17) follows immediately if $\Delta y_i \geq 0$. If $\Delta y_i < 0$, then from the interlacing eigenvalue theorem [14, Theorem 4.3.4], we conclude that $S^{-1} + \Delta y_i e_i e_i^T$ has at most one negative eigenvalue. Suppose that $S^{-1} + \Delta y_i e_i e_i^T$ has one negative eigenvalue. Then its determinant must be negative. From the matrix determinant identity, however, we have

$$\begin{aligned} \det(S^{-1} + \Delta y_i e_i e_i^T) &= (1 + \Delta y_i e_i^T S e_i) \det(S^{-1}) \\ &= (\mu\, S_{ii}/b_i) \det(S^{-1}) \geq 0. \end{aligned}$$

This contradicts the assumption that $S^{-1} + \Delta y_i e_i e_i^T$ has a negative eigenvalue. Thus, (17) holds even when $\Delta y_i < 0$. This completes the proof. ■

We can now describe the coordinate descent method in Algorithm 2.

Algorithm 2 Coordinate Descent Method for (3).

```
Let $y \leftarrow y^0$ such that $S = (\mathrm{Diag}(y) - L)^{-1} \succ 0$.
Compute the residual $\mathbf{r} := b - \mu\,\mathrm{diag}(S)$ and start the counter Iter $= 0$.
while $\|\mathbf{r}\| > \epsilon$ and Iter $\leq$ MaxIter do
    Let $i := \arg\max_k |\mathbf{r}_k|$ for $k = 1, \ldots, n$.
    Let $y \leftarrow y + \Delta y_i e_i$ where $\Delta y_i$ is given by (16).
    Rank-1 update to $S$
        $S \leftarrow S - \left(\frac{\Delta y_i}{1 + \Delta y_i S_{ii}}\right) S e_i e_i^T S.$    (18)
    Iter $\leftarrow$ Iter $+ 1$.
```

Note that the coordinate descent in Algorithm 2 requires only a single matrix inverse, whose computational complexity is $O(n^3)$. On the other hand, the rank-1 update (18) takes $O(n^2)$ operations. If we let the maximum number of CD steps grow linearly with n, then the computational complexity of Algorithm 2 is $O(n^3)$ operations. Similarly, the change of the objective function (3) that resulted from the rank-1 update can be computed efficiently by using the matrix determinant identity

$$f_\mu(y + \Delta y_i e_i) = f_\mu(y) + \Delta y_i b_i - \mu \log(1 + \Delta y_i S_{ii}).$$

B. Coarsening Scheme and the Algebraic Distance

The only component that remains to be explained in our multilevel method is the coarsening scheme, namely, the generation of the restriction matrix R. Several approaches exist in the literature with different emphasis on the interpolation accuracy and the sparsity of the resulting graphs [1]–[3]. Our coarsening scheme is based on the coarsening scheme develop by Ron, Safro, and Brandt in [3], which is proved effective for several combinatorial optimization problems on graphs.

Roughly speaking, the coarsening scheme can be divided into three steps.

1) Choose a set of seeds that will be the nodes in the coarsened graph.
2) Aggregate nonseed nodes to the designated seeds.
3) Assign the edge weights for the coarsened graph.

Since the coarsened graph should be similar to the original graph, it is important to quantify the distance between a pair of nodes in the uncoarsened graph. While a distance based on geometric proximity is intuitively appealing, it may not produce a reasonable coarsened graph for the optimization problem (3) in hand.

Following [3], we consider the *algebraic distance* that is determined by the residual after a number of coordinate descent steps. Starting with a set of K test vectors $y^1, y^2, \ldots, y^K$, we run a fixed number of CD steps to get $y^{(k)} \leftarrow \mathbf{CD}(y^k)$ for $k = 1, \ldots, K$. The algebraic distance between nodes i and j is defined

[1] One choice is $y = 2 \cdot \mathrm{abs}(L)\mathbf{1}$ where $\mathrm{abs}(\cdot)$ takes the absolute value elementwise. This is a valid choice for $\mathrm{Diag}(y) - L \succ 0$ because L is diagonally dominant and positive semidefinite.

as [3]

$$d_{ij} \;:=\; \left(\sum_{k=1}^{K} \left| y_i^{(k)} - y_j^{(k)} \right|^p \right)^{1/p},$$

where $p \geq 2$. A typical choice for the p-norm is the 2-norm or the ∞-norm:

$$\left(\sum_{k=1}^{K} \left| y_i^{(k)} - y_j^{(k)} \right|^2 \right)^{1/2} \quad \text{or} \quad \max_{k=1}^{K} \left| y_i^{(k)} - y_j^{(k)} \right|.$$

When $K = 1$, both norms coincide to be

$$d_{ij} \;=\; \left| y_i^{(1)} - y_j^{(1)} \right|. \tag{19}$$

The *algebraic coupling* between i and j is the reciprocal of the algebraic distance

$$c_{ij} \;:=\; 1/d_{ij}.$$

In choosing the seeds (i.e., the nodes in the coarsened graph), the basic idea is that the seeds should carry significant *mass* of the original graph. Let $v \in \mathbb{R}^n$ be the mass of the nodes in the original graph (e.g., $v = \mathbf{1}$.) The mass of node i weighted by its algebraic coupling with its neighbors is defined as [3]

$$\mathbf{v}_i \;:=\; v_i \,+\, \sum_{(i,j)\in\mathcal{E}} v_j \cdot \frac{c_{ij}}{\sum_{(j,k)\in\mathcal{E}} c_{jk}}, \tag{20}$$

where $\mathcal{E}$ is the set of edges in the uncoarsened graph. Note that for a fixed i, the set $(i,j) \in \mathcal{E}$ denotes the edges that connect to i. Therefore, the summation $\sum_{(i,j)\in\mathcal{E}} v_j \cdot \frac{c_{ij}}{\sum_{(j,k)\in\mathcal{E}} c_{jk}}$ is the weighted sum of the mass of i's neighbors. In particular, the weight $\frac{c_{ij}}{\sum_{(j,k)\in\mathcal{E}} c_{jk}}$ is the ratio of the coupling strength c_{ij} to the total coupling strength of the neighbors of node j.

We choose the nodes with the r-largest weighted masses to be the set of seeds (typically $r \approx n/2$), denoted as C, in the coarsened graph. If two or more nodes have the same mass, then we arbitrarily choose one of them.

Having decided on the seeds, we next aggregate nonseed nodes to the seeds. In principle, a nonseed node, say j, can belong to one or more seeds. Therefore, it may split into several parts with each part aggregating with the designated seed [3]. In this paper, we let each nonseed node to be aggregated with only one seed. The choice of the seed node, say s_j, is determined by

$$s_j \;:=\; \operatorname*{arg\,max}_{i\in C,(i,j)\in\mathcal{E}} \; c_{ij}.$$

In other words, j belongs to the seed i that has the strongest coupling with j among all neighboring seeds. If no neighbor of j is a seed, then we choose any seed that provides the strongest coupling, that is, $s_j = \arg\max \; c_{ij}$ for $i \in C$.

V. Numerical Experiments

We implement our multilevel algorithm in Matlab as a prototype. We note that the implementation has much room for improvement. Nevertheless, we verify the correctness of the multilevel approach and compare the computational time with that of the state-of-the-art solver SDPT3 [15].

We consider two sets of examples: complete graphs and power grid networks. The complete graphs are used to illustrate the scaling trend of the multilevel approach. The power grid examples are used to show the applicability of our multilevel approach to more realistic networks.

We set the log-barrier parameter to be $\mu = 10^{-3}$; the maximum number of CD steps to be MaxIter $= n$; the stopping criterion for CD to be $\|\mathbf{r}\| \leq \epsilon\sqrt{n}$ where $\epsilon = 10^{-3}$; and the stopping criterion for Newton's method to be Newton decrement less than 10^{-6}. Numerical experiments are performed on a workstation with 32 GB memory and two Intel E5430 Xeon 4-core 2.66 GHz CPUs running Matlab R2013a in Ubuntu 12.04.

A. Complete Graphs

A complete graph is a graph in which each pair of vertices is connected by an edge. The Laplacian matrix is $L = nI - J$, where J is the matrix of all ones. We set the threshold size for Newton's method to be $n_{\mathrm{NT}} = 250$ and test on problem size $n = 250$, 500, 1000, 2000, and 4000. Thus, the number of levels for the problem set is $l = 1, 2, 3, 4$, and 5.

Figure 2 shows that the relative objective function value between SDPT3 and our multilevel approach,

$$d_{\mathrm{obj}} \;:=\; |f_{\mathrm{SDPT3}} - f_{\mathrm{ML}}|/f_{\mathrm{SDPT3}}, \tag{21}$$

is less than 10^{-3} for all cases. Furthermore, the relative error is monotonically decreasing as n increases. Similarly, the solution value between SDPT3 and the multilevel approach,

$$d_{\mathrm{sol}} \;:=\; \|y_{\mathrm{SDPT3}} - y_{\mathrm{ML}}\|_2/\|y_{\mathrm{SDPT3}}\|_2, \tag{22}$$

is less than 10^{-3} for all cases and monotonically decreasing as n increases; see Fig. 3. On the other hand, as shown in Fig. 4, the multilevel approach takes approximately 25% of the time required by SDPT3 for small problems ($n = 250$) and 75% of the time required by SDPT3 for large problems ($n = 4000$).

B. IEEE 118-Bus Test Case

We consider the IEEE 118-bus test case taken from MATPOWER [16]. A diagram of the test case is shown in Fig. 5. In this context, the max-cut problem can be interpreted as cutting power lines to achieve the maximum (worst case) loss in the power transmission.

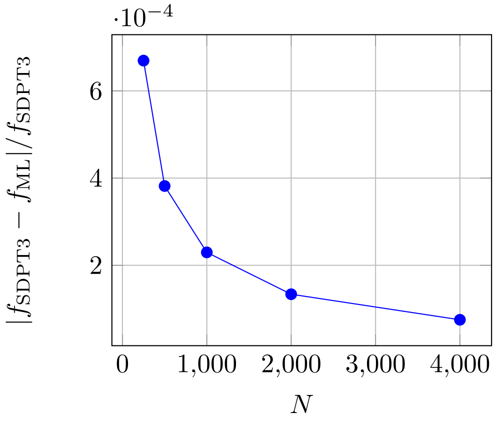


Fig. 2: Relative difference between the objective value from SDPT3 and from the multilevel algorithm for the complete graphs.

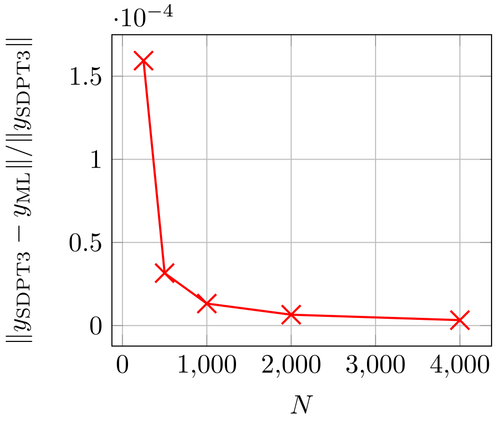


Fig. 3: Relative difference between the solution from SDPT3 and from the multilevel algorithm for the complete graphs.

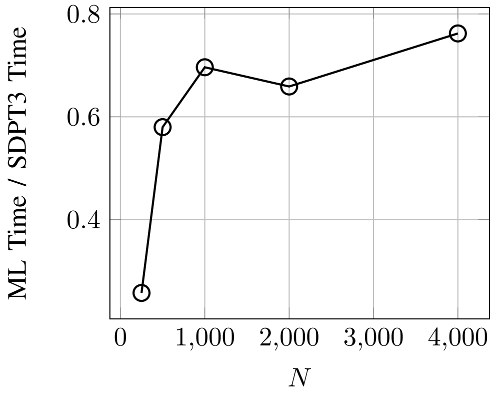


Fig. 4: Computational time of the multilevel algorithm normalized by that of SDPT3 for the complete graphs.

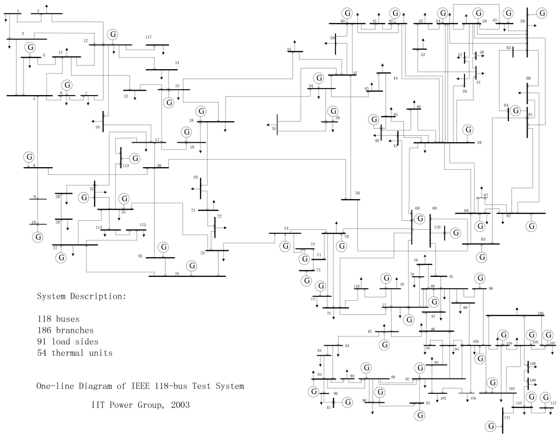


Fig. 5: Diagram of the IEEE 118-bus test case.

Since the network is small, with only $n = 118$ nodes, the multilevel approach simplifies to the single-level Newton's method. We verify the solution with SDPT3. In particular, the relative difference in the objective value (21) is $d_{\mathrm{obj}} = 4.4561 \times 10^{-4}$, and the relative difference in the solution (22) is $d_{\mathrm{sol}} = 7.3132 \times 10^{-5}$. The amount of time that the multilevel algorithm takes is 29.75% of the time that SDPT3 takes.

To illustrate the multilevel approach, however, we next coarsen the graph from $n = 118$ nodes to $n = 60$ nodes. We run coordinate descent on the original problem to get an approximate solution. We compute the algebraic distance (19) and the mass of each node (20).

Figure 6 shows a graphical representation in which the size of the node represents its mass and the color scheme indicates the algebraic distance between the nodes. We pick 60 nodes, whose mass is greater than or equal to the 60th largest mass, to be the seeds in the coarsened graph. We aggregate nonseed nodes to their neighboring seeds with the strongest algebraic coupling. The resulting graph with $n = 60$ nodes is shown in Fig. 7. (For this small example, the multilevel approach is not as competitive as SDPT3, due to the slow convergence rate of coordinate descent and the additional computation of the algebraic distance and the coarsening scheme.)

VI. Conclusions

We propose a multilevel approach for a class of semidefinite programs on graphs. By building a sequence of coarsened graphs, we solve the problem in a hierarchical fashion. At the coarsest level we employ Newton's method for accurate solutions, and at the finer levels we exploit rank-1 updates in coordinate descent for approximate solutions. We propose a coarsening scheme based on the algebraic distance that can be computed efficiently. Numerical experiments indicate that our multilevel approach is promising and competitive with the state-of-the-art SDP solvers for large problems.

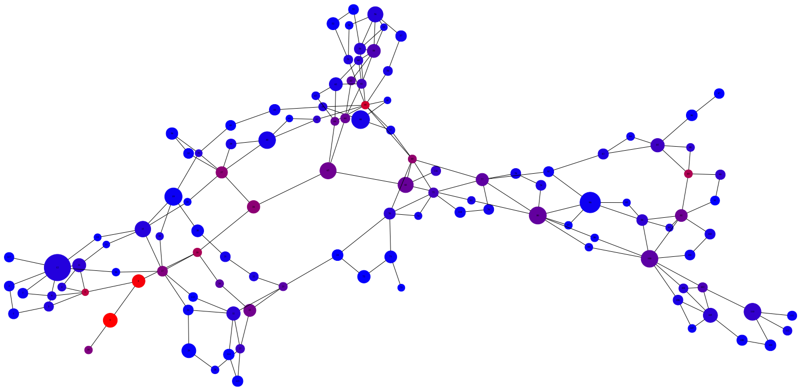


Fig. 6: Graphical representation of the IEEE 118-bus test case. Node size represents node mass (20). The blue-red color scheme indicates the algebraic distance (19) between the nodes.

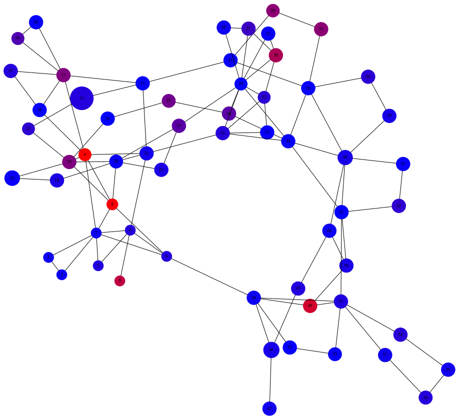


Fig. 7: Coarsened graph with $n = 60$ nodes for the IEEE 118-bus test case.

Acknowledgments

The first author thanks Dr. Todd Munson at Argonne National Laboratory for useful discussions.

References

[1] Y. Koren, L. Carmel, and D. Harel, "ACE: A fast multiscale eigenvectors computation for drawing huge graphs," in *IEEE Symposium on Information Visualization*, 2002, pp. 137–144.

[2] D. Ron, I. Safro, and A. Brandt, "A fast multigrid algorithm for energy minimization under planar density constraints," *Multiscale Modeling & Simulation*, vol. 8, no. 5, pp. 1599–1620, 2010.

[3] ——, "Relaxation-based coarsening and multiscale graph organization," *Multiscale Modeling & Simulation*, vol. 9, no. 1, pp. 407–423, 2011.

[4] O. E. Livne and A. Brandt, "Lean algebraic multigrid (LAMG): Fast graph Laplacian linear solver," *SIAM Journal on Scientific Computing*, vol. 34, no. 4, pp. 499–522, 2012.

[5] Y. Nesterov, "Efficiency of coordinate descent methods on huge-scale optimization problems," *SIAM Journal on Optimization*, vol. 22, no. 2, pp. 341–362, 2012.

[6] J. Nutini, M. Schmidt, I. H. Laradji, M. Friedlander, and H. Koepke, "Coordinate descent converges faster with the Gauss-Southwell rule than random selection," *arXiv preprint arXiv:1506.00552*, 2015.

[7] J. Liu and S. J. Wright, "Asynchronous stochastic coordinate descent: Parallelism and convergence properties," *SIAM Journal on Optimization*, vol. 25, no. 1, pp. 351–376, 2015.

[8] Z. Wen, D. Goldfarb, and K. Scheinberg, "Block coordinate descent methods for semidefinite programming," in *Handbook on Semidefinite, Conic and Polynomial Optimization*. Springer, 2012, pp. 533–564.

[9] I. Waldspurger, A. d'Aspremont, and S. Mallat, "Phase recovery, maxcut and complex semidefinite programming," *Mathematical Programming*, vol. 149, no. 1-2, pp. 47–81, 2015.

[10] C. Helmberg, F. Rendl, R. J. Vanderbei, and H. Wolkowicz, "An interior-point method for semidefinite programming," *SIAM Journal on Optimization*, vol. 6, no. 2, pp. 342–361, 1996.

[11] S. J. Benson, Y. Ye, and X. Zhang, "Solving large-scale sparse semidefinite programs for combinatorial optimization," *SIAM Journal on Optimization*, vol. 10, no. 2, pp. 443–461, 2000.

[12] S. Boyd and L. Vandenberghe, *Convex optimization*. Cambridge University Press, 2004.

[13] U. Trottenberg, C. W. Oosterlee, and A. Schuller, *Multigrid*. Academic Press, 2000.

[14] R. A. Horn and C. R. Johnson, *Matrix analysis*. Cambridge University Press, 1994.

[15] R. H. Tütüncü, K. C. Toh, and M. J. Todd, "Solving semidefinite-quadratic-linear programs using SDPT3," *Mathematical programming*, vol. 95, no. 2, pp. 189–217, 2003.

[16] R. D. Zimmerman, C. E. Murillo-Sánchez, and R. J. Thomas, "Matpower: Steady-state operations, planning, and analysis tools for power systems research and education," *IEEE Transactions on Power Systems*, vol. 26, no. 1, pp. 12–19, 2011.