QUADRATIC PROGRAMMING TECHNIQUES FOR GRAPH PARTITIONING *

SOONCHUL PARK[†], TIMOTHY A. DAVIS[‡], WILLIAM W. HAGER[§], AND HONGCHAO ZHANG[¶]

Abstract. In a seminal paper (*An efficient heuristic procedure for partitioning graphs,* Bell System Technical Journal, **49** (1970), pp. 291–307), Kernighan and Lin propose a pair exchange algorithm for approximating the solution to min-cut graph partitioning problems. In their algorithm, a vertex from one set in the current partition is exchanged with a vertex in the other set when the sum of the weights of cut edges is reduced. This algorithm along with the related Fiduccia/Mattheyses scheme are incorporated in state-of-the-art graph partitioning software such as METIS. In this paper we show that a quadratic programming-based block exchange generalization of the Kernighan and Lin algorithm can yield a significant improvement in partition quality.

Key words. graph partitioning, min-cut, quadratic programming

AMS subject classifications. 65K05, 65Y20, 90C20

1. Introduction.

The graph partitioning problem is to partition the vertices of a graph into several disjoint sets satisfying specified size constraints, while minimizing the sum of the weights of (cut) edges connecting vertices in different sets. Graph partitioning problems arise in circuit board and microchip design, in other layout problems (see [21]), and in sparse matrix pivoting strategies. In parallel computing, graph partitioning problems arise when tasks are partitioned among processors in order to minimize the communication between processors and balance the processor load. An application of graph partitioning to parallel molecular dynamics simulations is given in [26].

In [11, 12] we show that the graph partitioning problem can be formulated as a continuous quadratic programming problem denoted QP_1. Since the graph partitioning problem is NP hard, computing a global minimizer of QP_1 is often not easy. When continuous solution algorithms, such as the gradient projection method, are utilized, the iterates typically converge to a local minimizer which is not the global optimum. To escape from this local optimum, we need to make a nonlocal change to obtain a better iterate, which might then be used as a new starting guess for the gradient projection method.

In their seminal paper [20], Kernighan and Lin propose an exchange algorithm, denoted KL, for trying to improve any given partition of the vertices. A pair of vertices in the current partition is exchanged if the weights of the edges connecting the partitioned sets is decreased. Eventually, the algorithm reaches a partition of the

* November 14, 2006 This material is based upon work supported by the U.S. National Science Foundation under Grants 0203270, 0620286, and 0619080 and by the Korean National Science Foundation under grant Brain-Korea21

† scp@knu.ac.kr, School of Electrical Engineering and Computer Science, Kyungpook National University, Daegu, 702-701, Republic of Korea. Phone 82-53-950-7203. Fax 82-53-950-6093.

‡ davis@cise.ufl.edu, http://www.cise.ufl.edu/∼davis, PO Box 116120, Department of Computer and Information Science and Engineering, University of Florida, Gainesville, FL 32611-6120. Phone (352) 392-1481. Fax (352) 392-1220.

§ hager@math.ufl.edu, http://www.math.ufl.edu/∼hager, PO Box 118105, Department of Mathematics, University of Florida, Gainesville, FL 32611-8105. Phone (352) 392-0281. Fax (352) 392-8357.

¶ hozhang@ima.umn.edu, Institute for Mathematics and its Applications (IMA), University of Minnesota, 400 Lind Hall, 207 Church Street S.E., Minneapolis, MN 55455-0436

vertices for which any exchange either increases or leaves unchanged the sum of the weights of the cut edges.

The KL exchange is an example of a nonlocal change; for our quadratic programming formulation of the graph partitioning problem, it amounts to movement of distance $\sqrt{2}$. In this paper, we present a generalization of the KL pairwise exchange in which we allow an arbitrary block of vertices in one set of the partition to be moved to the other set. We show that the optimal exchange is the solution to a new QP, denoted QP_2, which is related to but different from QP_1. The block exchange QP_2 is more robust than KL for escaping from a local minimizer in QP_1 since there is no restriction on the number of vertices being exchanged.

Approaches to the graph partitioning problem in the literature include:

(a) Spectral methods, such as those in [16] and [24], where an eigenvector corresponding to the second smallest eigenvalue (Fiedler vector) of the graph's Laplacian is used to approximate the best partition.
(b) Geometric methods, such as those in [9, 14, 23], where geometric information for the graph is used to find a good partition.
(c) Multilevel algorithms, such as those in [5, 6, 15, 17, 25, 27], that first coarsen the graph, partition the smaller graph, then uncoarsen to obtain a partition for the original graph.
(d) Optimization-based methods, such as those in [1, 2, 3, 7, 28], where approximations to the best partitions are obtained by solving optimization problems.
(e) Methods that employ randomization techniques such as genetic algorithms ([22] or [25]).

State-of-the-art algorithms for graph partitioning which achieve both relatively high quality partitions and fast execution times include p- and h-METIS ([17], [18], [19]). These are multilevel algorithms which use either KL or the related Fiduccia/Mattheyses [8] (FM) schemes to improve the partition at each level. In this paper we show that the final partitions generated by METIS can be further optimized by exploiting QP_1 and QP_2. In a separate paper, we are developing a multilevel implementation of our optimization-based algorithms where the role of the KL or FM are either replaced or assisted by the QP-based optimization algorithms at each level.

The paper is organized as follows. In Section 2 we present QP_1, while Section 3 derives QP_2. In Section 4 we show how to incorporate QP_1 and QP_2 into a general algorithm for graph partitioning. Section 5 analyzes the potential improvement in a partition that can be achieved using the QP-based approach.

2. Graph partitioning. Consider a graph with n vertices

$$\mathcal{V} = \{1, 2, \cdots, n\},$$

and let a_{ij} be a weight associated with the edge (i, j). We assume that $a_{ii} = 0$ and $a_{ij} = a_{ji}$ for each i and j. The sign of the weights is not restricted. Given lower and upper integer bounds l and u respectively, we wish to partition the vertices into two disjoint sets, where one of the sets has between l and u vertices, while minimizing the sum of the weights associated with edges connecting vertices in different sets. An optimal partition is called a min-cut.

Let us consider the following quadratic programming problem which we denote QP_1:

$$\begin{aligned} &\text{minimize} \quad f(\mathbf{x}) := (\mathbf{1} - \mathbf{x})^{\mathsf{T}}(\mathbf{A} + \mathbf{D})\mathbf{x} \\ &\text{subject to} \;\; \mathbf{0} \le \mathbf{x} \le \mathbf{1}, \;\; l \le \mathbf{1}^{\mathsf{T}}\mathbf{x} \le u, \end{aligned} \tag{2.1}$$

where $\mathbf{1}$ is the vector whose entries are all 1, $\mathbf{A}$ is the matrix with elements a_{ij}, and $\mathbf{D}$ is a diagonal matrix. When $\mathbf{x}$ is binary, the cost function $f(\mathbf{x})$ in (2.1) is the sum of those a_{ij} for which $x_i = 0$ and $x_j = 1$. Hence, when $\mathbf{x}$ is binary, $f(\mathbf{x})$ is the sum of the weights of edges connecting the sets $\mathcal{V}_1$ and $\mathcal{V}_2$ defined by

$$\mathcal{V}_1 = \{i : x_i = 1\} \quad \text{and} \quad \mathcal{V}_2 = \{i : x_i = 0\}. \tag{2.2}$$

In [11] we show that for an appropriate choice of the diagonal matrix $\mathbf{D}$, the min-cut is obtained by solving (2.1); that is, (2.1) has a solution $\mathbf{x}$ for which each component is either zero or one, and the two sets $\mathcal{V}_1$ and $\mathcal{V}_2$ in an optimal partition are given by (2.2). The following result [11, Cor. 2.2] shows how to choose $\mathbf{D}$.

THEOREM 2.1. *If* $\mathbf{D}$ *is chosen so that*

$$d_{ii} + d_{jj} \geq 2a_{ij} \tag{2.3}$$

for each i and j, then (2.1) *has a* $0/1$ *solution* $\mathbf{x}$ *and the partition given by* (2.2) *is a min-cut. Moreover, if for each i and j,*

$$d_{ii} + d_{jj} > 2a_{ij},$$

then every local minimizer of (2.1) *is a* $0/1$ *vector.*

The condition (2.3) holds if the diagonal of $\mathbf{D}$ is chosen in the following way:

$$d_{jj} = \max\{a_{ij} : 1 \leq i \leq n\} \tag{2.4}$$

In the quadratic program (2.1), the variable $\mathbf{x}$ is continuous, with components taking values on the interval $[0, 1]$. Theorem 2.1 claims that this continuous quadratic program has a $0/1$ solution which yields a min-cut. As we now show, any feasible point for (2.1) can be transformed to a binary feasible point while not increasing the value of the cost function. Hence, any solution to (2.1) with fractional components can be transformed to a binary solution.

COROLLARY 2.2. *If* $\mathbf{D}$ *satisfies* (2.3)*, then for any* $\mathbf{x}$ *which is feasible in* (2.1)*, there exist a binary* $\mathbf{y}$ *which is feasible in* (2.1) *and* $f(\mathbf{y}) \leq f(\mathbf{x})$*.*

Proof. We first show how to find $\mathbf{y}$ with the property that $\mathbf{y}$ is feasible in (2.1), $\mathbf{1}^\mathsf{T}\mathbf{y}$ is integer, and $f(\mathbf{y}) \leq f(\mathbf{x})$. If $\mathbf{1}^\mathsf{T}\mathbf{x} = u$ or $\mathbf{1}^\mathsf{T}\mathbf{x} = l$, then we are done since l and u are integers; hence, we assume that $l < \mathbf{1}^\mathsf{T}\mathbf{x} < u$. If all components of $\mathbf{x}$ are binary, then we are done, so suppose that there exists a nonbinary component x_i. Since $a_{ii} = 0$, a Taylor expansion of f gives

$$f(\mathbf{x} + \alpha\mathbf{e}_i) = f(\mathbf{x}) + \alpha\nabla_{x_i} f(\mathbf{x}) - \alpha^2 d_{ii},$$

where $\mathbf{e}_i$ is the i-th column of the identity matrix. The quadratic term in the expansion is nonpositive. If the first derivative term is negative, then increase α above 0 until either $x_i + \alpha$ becomes 1 or $\mathbf{1}^\mathsf{T}\mathbf{x} + \alpha$ is an integer. Since the first derivative term is negative and $\alpha > 0$, $f(\mathbf{x} + \alpha\mathbf{e}_i) < f(\mathbf{x})$. If $\mathbf{1}^\mathsf{T}\mathbf{x} + \alpha$ becomes an integer, then we are done. If $x_i + \alpha$ becomes 1, then we reach a point $\mathbf{x}_1$ with one more binary component and with a smaller value for the cost function. If the first derivative term is nonnegative, then decrease α below 0 until either $x_i + \alpha$ becomes 0 or $\mathbf{1}^\mathsf{T}\mathbf{x} + \alpha$ is an integer. Since the first derivative term is nonnegative and $\alpha < 0$, $f(\mathbf{x} + \alpha\mathbf{e}_i) \leq f(\mathbf{x})$. If $\mathbf{1}^\mathsf{T}\mathbf{x} + \alpha$ becomes an integer, then we are done. If $x_i + \alpha$ becomes 0, then we reach a point $\mathbf{x}_1$ with one more binary component and with a smaller value for the cost function. In this latter case, we choose another nonbinary component of $\mathbf{x}_1$ and repeat the process. Hence, there is no loss of generality in assuming that $\mathbf{1}^\mathsf{T}\mathbf{x}$ is an integer.

Suppose that $\mathbf{x}$ is not binary. Since $\mathbf{1}^\mathsf{T}\mathbf{x}$ is an integer, $\mathbf{x}$ must have at least two nonbinary components, say x_i and x_j. Again, expanding f is a Taylor series gives

$$f(\mathbf{x} + \alpha(\mathbf{e}_i - \mathbf{e}_j)) = f(\mathbf{x}) + \alpha(\nabla_{x_i} - \nabla_{x_j})f(\mathbf{x}) + \alpha^2(2a_{ij} - d_{ii} - d_{jj}).$$

By (2.3), the quadratic term is nonpositive for any choice of α. If the first derivative term is negative, then we increase α above 0 until either $x_i + \alpha$ reaches 1 or $x_j - \alpha$ reach 0. Since the first derivative term is negative and $\alpha > 0$, $f(\mathbf{x} + \alpha(\mathbf{e}_i - \mathbf{e}_j)) < f(\mathbf{x})$. If the first derivative term is nonnegative, then we decrease α below 0 until either $x_i + \alpha$ reaches 0 or $x_j - \alpha$ reach 1. Since the first derivative term is nonnegative and $\alpha < 0$, $f(\mathbf{x} + \alpha(\mathbf{e}_i - \mathbf{e}_j)) \le f(\mathbf{x})$. In either case, the value of the cost function does not increase, and we reach a feasible point $\mathbf{x}_1$ with $\mathbf{1}^\mathsf{T}\mathbf{x}_1$ integer and with at least one more binary component. If $\mathbf{x}_1$ is not binary, then $\mathbf{x}_1$ must have at least two nonbinary components; hence, the adjustment process can be continued until all the components of $\mathbf{x}$ are binary. These adjustments to $\mathbf{x}$ do not increase the value of the cost function. ∎

The continuous quadratic programming problem (2.1) is NP hard. Hence, when continuous solution algorithms, such as the gradient projection method, are applied to (2.1), the iterates typically converge to a local minimizer which is not the global optimum. In order to escape from this local optimum, we need to make a nonlocal change in $\mathbf{x}$ to locate a deeper valley than that containing the current best approximation to a solution of (2.1). The KL exchange is an example of such a nonlocal change, the length of the movement is $\sqrt{2}$ since a 0 becomes 1 and a 1 becomes zero in $\mathbf{x}$. However, we have achieved much better success in escaping from local minimizers if we allow many components of $\mathbf{x}$ to change. The next section describes our block exchange QP.

3. Block exchange. Let $\mathbf{x}$ be a 0/1 vector satisfying the constraints of (2.1) and let $\mathcal{V}_1$ and $\mathcal{V}_2$ be the sets defined in (2.2). In a block exchange, the goal is to move some of the vertices of $\mathcal{V}_1$ to $\mathcal{V}_2$ and some of the vertices of $\mathcal{V}_2$ to $\mathcal{V}_1$ while satisfying the constraint that the number of vertices in $\mathcal{V}_1$ should be between l and u. Let $\mathbf{y}$ and $\mathbf{z}$ be subvectors of $\mathbf{x}$ which correspond to the components of $\mathbf{x}$ which are 1 and 0 respectively. In other words, the i-th component of $\mathbf{y}$ corresponds to the i-th vertex in $\mathcal{V}_1$, which we now view as an ordered set. Similarly, the i-th component of $\mathbf{z}$ corresponds to the i-th vertex in $\mathcal{V}_2$. We set $y_i = 1$ if and only if the i-th element of $\mathcal{V}_1$ is moved to $\mathcal{V}_2$. Similarly, let us set $z_j = 1$ if and only if the j-th element of $\mathcal{V}_2$ is moved to $\mathcal{V}_1$. The number of vertices in $\mathcal{V}_1$ is initially $\mathbf{1}^\mathsf{T}\mathbf{x}$. The constraint that the total number of vertices in $\mathcal{V}_1$ lies between l and u after the exchange can be expressed

$$l - \mathbf{1}^\mathsf{T}\mathbf{x} \le \mathbf{1}^\mathsf{T}\mathbf{z} - \mathbf{1}^\mathsf{T}\mathbf{y} \le u - \mathbf{1}^\mathsf{T}\mathbf{x}. \tag{3.1}$$

Let $\mathcal{X}_1$ and $\mathcal{X}_2$ be the support of $\mathbf{y}$ and $\mathbf{z}$ respectively:

$$\mathcal{X}_1 = \{i : y_i = 1\} \quad \text{and} \quad \mathcal{X}_2 = \{j : z_j = 1\}.$$

These sets correspond to the vertices which are exchanged. The indices in $\mathcal{X}_1$ correspond to vertices in $\mathcal{V}_1$ which are moved to $\mathcal{V}_2$; the indices in $\mathcal{X}_2$ correspond to vertices

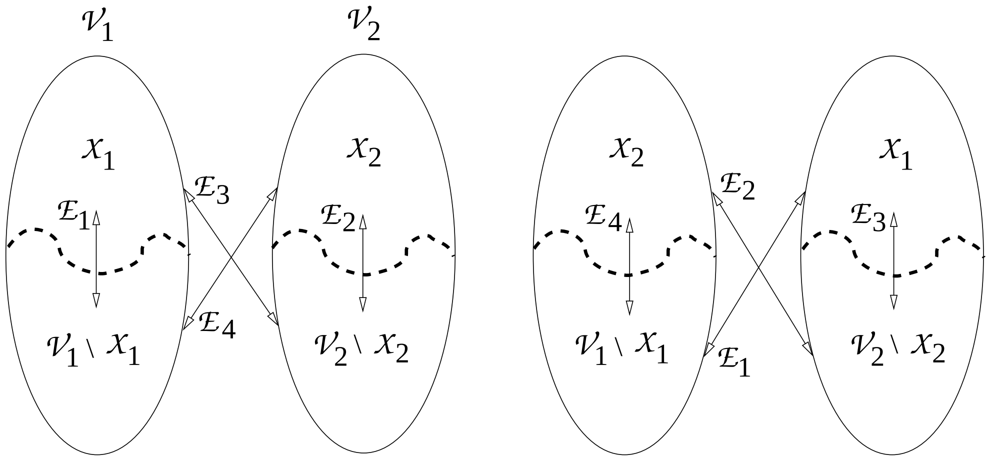

FIG. 3.1. *Exchange vertices $\mathcal{X}_1$ in $\mathcal{V}_1$ with $\mathcal{X}_2$ in $\mathcal{V}_2$*

in $\mathcal{V}_2$ which are moved to $\mathcal{V}_1$. The edges which participate in the exchange are the following (see Figure 3.1):

$$\begin{aligned}
\mathcal{E}_1 &= \text{Edges between } \mathcal{X}_1 \text{ and } \mathcal{V}_1 \setminus \mathcal{X}_1\\
\mathcal{E}_2 &= \text{Edges between } \mathcal{X}_2 \text{ and } \mathcal{V}_2 \setminus \mathcal{X}_2\\
\mathcal{E}_3 &= \text{Edges between } \mathcal{X}_1 \text{ and } \mathcal{V}_2 \setminus \mathcal{X}_2\\
\mathcal{E}_4 &= \text{Edges between } \mathcal{X}_2 \text{ and } \mathcal{V}_1 \setminus \mathcal{X}_1
\end{aligned}$$

Edges connecting $\mathcal{X}_1$ and $\mathcal{X}_2$ and edges connecting $\mathcal{V}_1 \setminus \mathcal{X}_1$ and $\mathcal{V}_2 \setminus \mathcal{X}_2$ are not effected by the exchange so they are ignored.

The change in the number of cut edges due to the exchange of vertices associated with $\mathcal{X}_1$ and $\mathcal{X}_2$ is given by the expression:

$$|\mathcal{E}_1| + |\mathcal{E}_2| - |\mathcal{E}_3| - |\mathcal{E}_4| \tag{3.2}$$

where $|\mathcal{E}_i|$ denotes the number of elements in the set $\mathcal{E}_i$. Before the exchange, the edges $\mathcal{E}_1$ and $\mathcal{E}_2$ are internal edges, while after the exchange, they become external edges that are included in the collection of cut edges. Before the exchange, the edges $\mathcal{E}_3$ and $\mathcal{E}_4$ are external edges, included in the set of cut edges; after the exchange, these edges are internal edges.

Suppose that the rows and columns of $\mathbf{A}$ are symmetrically permuted so that the leading rows and columns correspond to $\mathcal{V}_1$, the support of $\mathbf{x}$, and the trailing rows and columns correspond to $\mathcal{V}_2$. We block partition the resulting $\mathbf{A}$ in the form

$$\mathbf{A} = \begin{pmatrix} \mathbf{A}_{11} & \mathbf{A}_{12} \\ \mathbf{A}_{21} & \mathbf{A}_{22} \end{pmatrix}, \tag{3.3}$$

where $\mathbf{A}_{ii}$ correspond to $\mathcal{V}_i$, $i = 1, 2$. Similar to (3.2), the change in the weight of the cut edges associated with the exchange is given by

$$(\mathbf{1} - \mathbf{y})^{\mathsf{T}} \mathbf{A}_{11} \mathbf{y} + (\mathbf{1} - \mathbf{z})^{\mathsf{T}} \mathbf{A}_{22} \mathbf{z} - (\mathbf{1} - \mathbf{z})^{\mathsf{T}} \mathbf{A}_{21} \mathbf{y} - (\mathbf{1} - \mathbf{y})^{\mathsf{T}} \mathbf{A}_{12} \mathbf{z}. \tag{3.4}$$

The first two terms are the weights of external edges created by the exchange, while the last two terms are the weight of the prior external edges which became internal after the exchange. Observe that the quadratic (3.4) can be written

$$\left(\begin{array}{c}\mathbf{1}-\mathbf{y}\\ \mathbf{1}-\mathbf{z}\end{array}\right)^{\mathsf{T}}\left(\begin{array}{cc}\mathbf{A}_{11} & -\mathbf{A}_{12}\\ -\mathbf{A}_{21} & \mathbf{A}_{22}\end{array}\right)\left(\begin{array}{c}\mathbf{y}\\ \mathbf{z}\end{array}\right). \tag{3.5}$$

Motivated by (3.1) and (3.5), we consider the following quadratic programming problem which we denote QP_2:

$$\left\{\begin{array}{l}\text{minimize } F(\mathbf{y},\mathbf{z}) := \left(\begin{array}{c}\mathbf{1}-\mathbf{y}\\ \mathbf{1}-\mathbf{z}\end{array}\right)^{\mathsf{T}}\left(\begin{array}{cc}\mathbf{A}_{11}+\mathbf{D}_1 & -\mathbf{A}_{12}\\ -\mathbf{A}_{21} & \mathbf{A}_{22}+\mathbf{D}_2\end{array}\right)\left(\begin{array}{c}\mathbf{y}\\ \mathbf{z}\end{array}\right)\\ \text{subject to } \mathbf{0}\le\mathbf{y}\le\mathbf{1},\ \mathbf{0}\le\mathbf{z}\le\mathbf{1},\ l-\mathbf{1}^{\mathsf{T}}\mathbf{x}\le\mathbf{1}^{\mathsf{T}}\mathbf{z}-\mathbf{1}^{\mathsf{T}}\mathbf{y}\le u-\mathbf{1}^{\mathsf{T}}\mathbf{x}.\end{array}\right. \tag{3.6}$$

Here $\mathbf{D}_1$ and $\mathbf{D}_2$ are diagonal matrices. If $\mathbf{y}$ and $\mathbf{z}$ are binary, then the terms

$$(\mathbf{1}-\mathbf{y})^{\mathsf{T}}\mathbf{D}_1\mathbf{y} \quad \text{and} \quad (\mathbf{1}-\mathbf{z})^{\mathsf{T}}\mathbf{D}_2\mathbf{z}$$

have no effect on the cost F since

$$(\mathbf{1}-\mathbf{y})\mathbf{D}_1^{\mathsf{T}}\mathbf{y} = 0 = (\mathbf{1}-\mathbf{z})\mathbf{D}_2^{\mathsf{T}}\mathbf{z}.$$

As with the quadratic formulation (2.1) of the graph partitioning problem, we show that that for a suitable choice of $\mathbf{D}_1$ and $\mathbf{D}_2$, the quadratic formulation (3.6) of the exchange problem has a 0/1 solution. Moreover, the proof reveals how to convert a fractional solution to a 0/1 solution without increasing the cost. In the following theorem, we assume that the original matrix of weights $\mathbf{A}$ has been symmetrically permuted into the form (3.3) so that the leading rows and columns correspond to the support of a 0/1 vector $\mathbf{x}$ feasible in (2.1).

THEOREM 3.1. *If* $\mathbf{x}$ *is a* 0/1 *vector which is feasible for* (2.1), *and the diagonal matrix* $\mathbf{D}$ *satisfies the condition*

$$d_{ii} + d_{jj} - 2a_{ij} \ge 0 \textit{ for all } i \textit{ and } j, \tag{3.7}$$

then (3.6) *has a* 0/1 *solution.*

Again, (3.7) is satisfied for the choice of $\mathbf{D}$ given in (2.4).

Proof. Since l and u are integers and since $\mathbf{x}$ is 0/1, it follows that both $l-\mathbf{1}^{\mathsf{T}}\mathbf{x}$ and $u-\mathbf{1}^{\mathsf{T}}\mathbf{x}$ are integers. Let $\mathbf{y}$ and $\mathbf{z}$ be feasible in (3.6). We first show that there exists a feasible point $(\bar{\mathbf{y}},\bar{\mathbf{z}})$ for (3.6) with $\mathbf{1}^{\mathsf{T}}\bar{\mathbf{z}}-\mathbf{1}^{\mathsf{T}}\bar{\mathbf{y}}$ integer and $F(\bar{\mathbf{y}},\bar{\mathbf{z}}) \le F(\mathbf{y},\mathbf{z})$. If $\mathbf{1}^{\mathsf{T}}\mathbf{z}-\mathbf{1}^{\mathsf{T}}\mathbf{y}$ is not an integer, then at least one component of either $\mathbf{y}$ or $\mathbf{z}$ is not an integer. Suppose that y_i is not an integer and let $\mathbf{e}_i$ denote the i-th column of the identity matrix. Expanding F in a Taylor series gives

$$F(\mathbf{y}+\alpha\mathbf{e}_i,\mathbf{z}) = F(\mathbf{y},\mathbf{z}) + \alpha\nabla_{y_i}F(\mathbf{y},\mathbf{z}) - \alpha^2 d_{ii}$$

since $a_{ii}=0$. The last term $-\alpha^2 d_{ii}$ is nonpositive due to (3.7). If the first derivative $\nabla_{y_i}F(\mathbf{y},\mathbf{z})$ is negative, then increase α above 0 until either $y_i+\alpha=1$ or $\mathbf{1}^{\mathsf{T}}\mathbf{z}-\mathbf{1}^{\mathsf{T}}\mathbf{y}-\alpha$ becomes an integer, whichever occurs first. This leads us to a new point with strictly smaller cost than the original $(\mathbf{y},\mathbf{z})$ since the first derivative term is negative and the cost decreases as α increases. If the increase in α causes $\mathbf{1}^{\mathsf{T}}\mathbf{z}-\mathbf{1}^{\mathsf{T}}\mathbf{y}-\alpha$ to become an integer, then we are done. If $y_i+\alpha$ becomes 1, then we reach a feasible point $\mathbf{y}+\alpha\mathbf{e}_i$ which has one more binary component.

If the first derivative $\nabla_{y_i} F(\mathbf{y}, \mathbf{z})$ is nonnegative, then decrease α below 0 until either $y_i + \alpha = 0$ or $\mathbf{1}^\mathsf{T}\mathbf{z} - \mathbf{1}^\mathsf{T}\mathbf{y} - \alpha$ becomes an integer, whichever occurs first. Since $\nabla_{y_i} F(\mathbf{y}, \mathbf{z})$ is nonnegative, this decrease in α will not increase the value of the cost function. Again, if $\mathbf{1}^\mathsf{T}\mathbf{z} - \mathbf{1}^\mathsf{T}\mathbf{y} - \alpha$ becomes an integer, we are done. Otherwise, $y_i + \alpha$ becomes zero and we reach a point $\mathbf{y} + \alpha\mathbf{e}_i$ which has one more binary component. By inductively applying these adjustments to the fractional components of $\mathbf{y}$ and $\mathbf{z}$, we eventually reach a feasible point $(\bar{\mathbf{y}}, \bar{\mathbf{z}})$ with a better value for the cost function and with $\mathbf{1}^\mathsf{T}\bar{\mathbf{z}} - \mathbf{1}^\mathsf{T}\bar{\mathbf{y}}$ integer. Thus, without loss of generality, we assume that $(\mathbf{y}, \mathbf{z})$ is feasible in (3.6) and $\mathbf{1}^\mathsf{T}\mathbf{z} - \mathbf{1}^\mathsf{T}\mathbf{y}$ is integer.

Suppose that $\mathbf{y}$ has at least two nonbinary components; let y_i and y_j denote nonbinary components of $\mathbf{y}$. Expanding in a Taylor series gives

$$F(\mathbf{y} + \alpha(\mathbf{e}_i - \mathbf{e}_j), \mathbf{z}) = F(\mathbf{y}, \mathbf{z}) + \alpha(\nabla_{y_i} - \nabla_{y_j})F(\mathbf{y}, \mathbf{z}) + \alpha^2(2a_{ij} - d_{ii} - d_{jj}).$$

By (3.7) the α^2 term is nonnegative for any choice of α. If the first derivative term $(\nabla_{y_i} - \nabla_{y_j})F(\mathbf{y}, \mathbf{z})$ is negative, then we increase α above 0 to decrease the cost. We continue to increase α until some component of $\mathbf{y} + \alpha(\mathbf{e}_i - \mathbf{e}_j)$ reaches either 0 or 1. Since $\mathbf{1}^\mathsf{T}(\mathbf{e}_i - \mathbf{e}_j) = 0$, we have

$$\mathbf{1}^\mathsf{T}\mathbf{z} - \mathbf{1}^\mathsf{T}(\mathbf{y} + \alpha(\mathbf{e}_i - \mathbf{e}_j)) = \mathbf{1}^\mathsf{T}\mathbf{z} - \mathbf{1}^\mathsf{T}\mathbf{y}.$$

Hence, this adjustment to components y_i and y_j of $\mathbf{y}$ leads us to a new point with at least one more binary component and with $\mathbf{1}^\mathsf{T}\mathbf{z} - \mathbf{1}^\mathsf{T}\mathbf{y}$ integer. The same adjustment process can be applied to the components of $\mathbf{z}$. Hence, when we are done, $\mathbf{1}^\mathsf{T}\mathbf{z} - \mathbf{1}^\mathsf{T}\mathbf{y}$ is an integer and $\mathbf{y}$ and $\mathbf{z}$ have at most one nonbinary component.

Suppose that $\mathbf{y}$ has one nonbinary component y_i. Since $\mathbf{1}^\mathsf{T}\mathbf{z} - \mathbf{1}^\mathsf{T}\mathbf{y}$ is an integer, $\mathbf{z}$ must have a nonbinary component denoted z_k and $y_i = z_k$. Define $j = k + |\mathcal{V}_1|$. Expanding in a Taylor series gives

$$F(\mathbf{y} + \alpha\mathbf{e}_i, \mathbf{z} + \alpha\mathbf{e}_k) = F(\mathbf{y}, \mathbf{z}) + \alpha(\nabla_{y_i} + \nabla_{z_k})F(\mathbf{y}, \mathbf{z}) + \alpha^2(2a_{ij} - d_{ii} - d_{jj}).$$

By (3.7) the last term is nonpositive for all choices of α. If the first derivative term is negative, then we increase α above 0 until $y_i + \alpha = 1 = z_k + \alpha$. If the first derivative term is nonnegative, then we decrease α below 0 until $y_i + \alpha = 0 = z_k + \alpha$. In either case, after these adjustments in the i-th component of $\mathbf{y}$ and the k-th component of $\mathbf{z}$, the cost value does not increase and the difference $\mathbf{1}^\mathsf{T}\mathbf{z} - \mathbf{1}^\mathsf{T}\mathbf{y}$ does not change; hence, the new point is binary and feasible in QP_2. This completes the proof. □

COROLLARY 3.2. *If* $\mathbf{D}$ *is chosen so that the inequality* (3.7) *is strict, then every local minimizer of* (3.6) *is binary.*

Proof. By the analysis given in the proof of Theorem 3.1, any nonbinary local minimizer can be pushed to the boundary while improving the value of the cost function. If the inequality (3.7) is strict, then when we push to the boundary, the value of the cost function is strictly decreased. Hence, any local minimizer must be binary. □

4. The algorithm. We now explain how to incorporate the theory developed in Sections 2 and 3 in an optimization algorithm for the graph partitioning problem. The overall strategy is to apply an optimization algorithm, such as the gradient projection method, to QP_1 until we reach a local minimizer; next, we apply an optimization algorithm to the exchange quadratic program QP_2 in an effort to escape from the current local minimum. If we are unable to find a better point, then we stop. Otherwise, use the $\mathbf{x}$ obtained from QP_2 as a starting guess in QP_1 and repeat the process.

We use two different optimization algorithms to approximate a solution to QP_1 and QP_2. In the first optimization algorithm, we approximate the feasible set by a sphere and we utilize the algorithm in [10, 13] to efficiently compute the global minimum. Typically, a global minimizer for this sphere constrained problem lies outside the feasible set. Hence, we project a global minimizing point onto the feasible set. Such a projection is easily computed in $O(n)$ time. In the second optimization algorithm, we apply the gradient projection algorithm to either QP_1 or QP_2. We used a version of the gradient projection algorithm based on an Armijo line search along the projection arc (see [4, p. 226]).

In more detail, the steps of the algorithm are as follows:

A1. Define $\mathbf{x}_c = \alpha \mathbf{1}$, $\alpha = (l+u)/(2n)$. Let $\mathbf{x}_1$ be a solution to the following sphere constrained problem

$$\min f(\mathbf{x}) \text{ subject to } \mathbf{1}^{\mathsf{T}}\mathbf{x} = (l+u)/2 \text{ and } \|\mathbf{x} - \mathbf{x}_c\|^2 \le r_1^2.$$

Since QP_1 has a solution with between l and u ones and with the remaining entries zero, we choose r_1 to include all points $\mathbf{x}$ with $(l+u)/2$ ones and with the remaining entries zero. In other words,

$$r_1^2 = (1-\alpha)^2 \left(\frac{l+u}{2}\right) + \alpha^2 \left(n - \frac{l+u}{2}\right).$$

A2. Let $\mathbf{x}_2$ be the projection of $\mathbf{x}_1$ onto the feasible set for QP_1. If $f(\mathbf{x}_2) \ge f(\mathbf{x}_c)$, then reduce r_1 and repeat A1.

A3. Starting from $\mathbf{x}_2$, we apply the gradient projection method to QP_1 until we reach a stationary point denoted $\mathbf{x}_3$.

A4. Using the method developed in Corollary 2.2, we transform $\mathbf{x}_3$ to a binary vector $\mathbf{x}_4$ with a better value for the cost function.

A5. Based on the binary structure of $\mathbf{x}_4$, we partition $\mathbf{A}$ as indicated in (3.5). Let d_y and d_z denote the dimensions of $\mathbf{y}$ and $\mathbf{z}$. With a permutation of $\mathbf{A}$, it can be arranged so that $d_z \le d_y$. We define $\mathbf{z}_c = .5$ and $\mathbf{y}_c = .5d_z/d_y$ (hence, $\mathbf{1}^{\mathsf{T}}\mathbf{z}_c - \mathbf{1}^{\mathsf{T}}\mathbf{y}_c = 0$). Let $\mathbf{x}_5 = (\mathbf{y}, \mathbf{z})$ be any solution of the problem

$$\min F(\mathbf{y}, \mathbf{z}) \text{ subject to } \mathbf{1}^{\mathsf{T}}\mathbf{z} = \mathbf{1}^{\mathsf{T}}\mathbf{y}, \quad \|\mathbf{y} - \mathbf{y}_c\|^2 + \|\mathbf{z} - \mathbf{z}_c\|^2 \le r_5^2, \tag{4.1}$$

where

$$r_5^2 = -.75d_z + d_y + .25d_z^2/d_y.$$

The radius r_5 of the sphere in (4.1) is chosen large enough to ensure that all possible solutions to the problem of minimizing $F(\mathbf{y}, \mathbf{z})$, subject to the constraint that $\mathbf{1}^{\mathsf{T}}\mathbf{z} = \mathbf{1}^{\mathsf{T}}\mathbf{y}$ and $\mathbf{y}$ and $\mathbf{z}$ are binary, are contained in the sphere.

A6. Let $\mathbf{x}_6$ be the projection of $\mathbf{x}_5$ into the feasible set of QP_2. If $F(\mathbf{x}_6) \ge F(\mathbf{y}_c, \mathbf{z}_c)$, then reduce r_5 and repeat A5.

A7. Starting from $\mathbf{x}_6$, we apply the gradient projection method to QP_2 until we reach a stationary point denoted $\mathbf{x}_7$.

A8. Using the method developed in Theorem 3.1, we transform $\mathbf{x}_7$ to a binary vector $\mathbf{x}_8$.

A9. If the exchange associated with $\mathbf{x}_8$ improves the partitioning associated with $\mathbf{x}_4$, then we apply the exchange to $\mathbf{x}_4$ to obtain the new point $\mathbf{x}_9$; set $\mathbf{x}_2 = \mathbf{x}_9$ and branch to A3. If the exchange associated with $\mathbf{x}_8$ does not strictly improve the partitioning associated with $\mathbf{x}_4$, then we are done.

For the numerical experiments reported in this paper, we did not reduce the radius of the spheres, as suggested in A2 and A6, when the solution of the sphere constrained problem yielded a poorer objective function value than the centroid of the sphere. This enhancement will be incorporated in a multilevel version of our algorithms.

5. Numerical results. The optimization-based algorithm developed in Section 4 should require much more CPU time than the multilevel technology of METIS since the optimization algorithms operate on the entire matrix. We are in the process of developing compiled code and multilevel technology where the optimization methodology of Section 4 is applied to the compressed graphs generated in the multilevel approach. As a preliminary assessment of the merits of the optimization-based strategy for graph partitioning, we applied both p- and h-METIS to a series of graph bisection problems. In other words, if n is even, then $l = u = n/2$, and if n is odd, then $l = u = (n+1)/2$. The partitions generated by p- or h-METIS were used as starting points for the optimization algorithm in step A3 to determine whether the METIS generated partitions could be further improved using the optimization algorithms. All the algorithms were implemented in MATLAB, and the test problems were obtained from the UF Sparse Matrix Library maintained by Timothy Davis:

http://www.cise.ufl.edu/research/sparse/matrices/

In our numerical experiments, the diagonal of $\mathbf{A}$ is always zero. The off-diagonal elements are constructed as follows: If $\mathbf{S}$ is a symmetric matrix in the library, then then $a_{ij} = 0$ if $s_{ij} = 0$ and $a_{ij} = 1$ otherwise. If $\mathbf{S}$ is a m by n nonsymmetric matrix with $m \ge n$, then $a_{ij} = 0$ if $(\mathbf{S}^{\mathsf{T}}\mathbf{S})_{ij} = 0$ and $a_{ij} = 1$ otherwise. If $m < n$, then $a_{ij} = 0$ if $(\mathbf{S}\mathbf{S}^{\mathsf{T}})_{ij} = 0$ and $a_{ij} = 1$ otherwise.

Since our codes are in MATLAB, we could not apply them to all the test matrices (without expending a huge amount of CPU time). Altogether, we tried 701 test problems; the mean dimension for $\mathbf{A}$ was 1157 and the mean number of edges in the test problems was 57,057. There were 287 problems with dimension greater than 1,000, and there were 307 problems with more than 10,000 edges in the graph.

To quantitatively evaluate the improvement provided by the optimization routines, we evaluated the quantity:

$$\frac{\text{reduction in number of cut edges due to optimization algorithms}}{\text{the number of cut edges obtained by METIS}} \times 100.$$

This expression gives the percent improvement in the number of cut edges obtained by applying the optimization algorithms to the final partition generated by METIS. In Table 5 we show the percentage of the matrices for which we could improve the partition using the optimization algorithms. A detailed tabulation of our results is posted at the following web site:

http://www.math.ufl.edu/∼hager/papers/GP/

For each of the matrices where the cut edges were improved, we also compute the average percentage of improvement. Overall, we could improve the partitions generated by p-METIS in about 50% of the problems, and the average improvement was about 10%. We could improve the partitions generated by h-METIS in about 31% of the problems, and the average improvement was about 5.7%. For both versions of METIS, the greatest improvement occurred in matrices of the largest dimension. In particular, for matrices of dimensions between 4001 and 5000, the average improvement for p-METIS was 11.8% while the average improvement for h-METIS was 9.2%.

Dimension of problem	Number of problems	Method	Problems with cut edge reduction	Average improvement
1 to 1000	444	p-METIS	193 (44%)	10.02%
		h-METIS	118 (27%)	5.48%
1001 to 2000	156	p-METIS	111 (71%)	11.37%
		h-METIS	50 (32%)	5.52%
2001 to 3000	48	p-METIS	35 (73%)	7.31%
		h-METIS	18 (38%)	4.09%
3001 to 4000	33	p-METIS	18 (55%)	9.64%
		h-METIS	16 (49%)	7.42%
4001 to 5000	20	p-METIS	14 (70%)	11.82%
		h-METIS	12 (60%)	9.21%

Table 5.1
Improvement in p- and h-METIS due to the optimization algorithms

REFERENCES

[1] E. R. Barnes, *An algorithm for partitioning the nodes of a graph*, SIAM J. Alg. Disc. Meth., 3 (1984), pp. 541–550.
[2] E. R. Barnes and A. J. Hoffman, *Partitioning, spectra, and linear programming*, in Progress in Combinatorial Optimization, W. E. Pulleyblank, ed., Academic Press, New York, 1984, pp. 13–25.
[3] E. R. Barnes, A. Vannelli, and J. Q. Walker, *A new heuristic for partitioning the nodes of a graph*, SIAM J. Alg. Disc. Meth., 1 (1988), pp. 299–305.
[4] D. P. Bertsekas, P. A. Hosein, and P. Tseng, *Relaxation methods for network flow problems with convex arc costs*, SIAM J. Control Optim., 25 (1987), pp. 1219–1243.
[5] T. Bui and C. Jones, *A heuristic for reducing fill in sparse matrix factorization*, in Proc. 6th SIAM Conf. Parallel Processing for Scientific Computation, SIAM, 1993, pp. 445–452.
[6] C. K. Cheng and Y. C. Wei, *An improved two-way partioning algorithm with stable performance*, IEEE Trans. Computer-Aided Design, 10 (1991), pp. 1502–1511.
[7] J. Falkner, F. Rendl, and H. Wolkowicz, *A computational study of graph partitioning*, Math. Program., 66 (1994), pp. 211–240.
[8] C. M. Fiduccia and R. M. Mattheyses, *A linear-time heuristic for improving network partition*, in Proc. 19th Design Automation Conf., Las Vegas, NV, 1982, pp. 175–181.
[9] J. R. Gilbert, G. L. Miller, and S. H. Teng, *Geometric mesh partitioning: Implementation and experiments*, SIAM J. Sci. Comput., 19 (1998), pp. 2091–2110.
[10] W. W. Hager, *Minimizing a quadratic over a sphere*, SIAM J. Optim., 12 (2001), pp. 188–208.
[11] W. W. Hager and Y. Krylyuk, *Graph partitioning and continuous quadratic programming*, SIAM J. Alg. Disc. Meth., 12 (1999), pp. 500–523.
[12] ———, *Multiset graph partitioning*, Mathematics of Operations Research, 55 (2002), pp. 1–10.
[13] W. W. Hager and S. C. Park, *Global convergence of SSM for minimizing a quadratic over a sphere*, Math. Comp., 74 (2005), pp. 1413–1423.
[14] M. T. Heath and P. Raghavan, *A Cartesian parallel nested dissection algorithm*, SIAM J. Matrix Anal. Appl., 16 (1995), pp. 235–253.
[15] B. Hendrickson and R. Leland, *A multilevel algorithm for partitioning graphs*, Tech. Rep. SAND93-1301, Sandia National Laboratory, 1993.
[16] ———, *An improved spectral graph partitioning algorithm for mapping parallel computations*, SIAM J. Sci. Comput., 16 (1995), pp. 452–469.
[17] G. Karypis and V. Kumar, *A fast and high quality multilevel scheme for partitioning irregular graphs*, SIAM J. Sci. Comput., 20 (1998), pp. 359–392.
[18] ———, *Multilevel k-way partitioning scheme for irregular graphs*, J. Parallel Distrib. Comput., 48 (1999), pp. 96–129.
[19] ———, *Multilevel k-way hypergraph partitioning*, VLSI Design, 11 (2000), pp. 285–300.
[20] B. W. Kernighan and S. Lin, *An efficient heuristic procedure for partitioning graphs*, Bell System Tech. J., 49 (1970), pp. 291–307.

[21] T. Lengauer, *Combinatorial Algorithms for Integrated Circuit Layout*, John Wiley, Chichester, 1990.

[22] J. G. Martin, *Subproblem optimization by gene correlation with singular value decomposition*, in GECCO'05, Washington, D.C., 2005, ACM.

[23] G. L. Miller, S. H. Teng, W. Thurston, and S. A. Vavasis, *Automatic mesh partitioning*, in Sparse Matrix Computations: Graph Theory Issues and Algorithms, A. George, J. R. Gilbert, and J. W. H. Liu, eds., vol. 56 of IMA Vol. Math. Appl., New York, 1993, Springer-Verlag, pp. 57–84.

[24] A. Pothen, H. D. Simon, and K. Liou, *Partitioning sparse matrices with eigenvectors of graphs*, SIAM J. Matrix Anal. Appl., 11 (1990), pp. 430–452.

[25] A. J. Soper, C. Walshaw, and M. Cross, *A combined multilevel search and multilevel optimization approach to graph-partition*, J. Global Optim., 29 (2004), pp. 225–241.

[26] S.-H. Teng, *Provably good partitioning and load balancing algorithms for parallel adaptive N-body simulation*, SIAM J. Sci. Comput., 19 (1998), pp. 635–656.

[27] C. Walshaw, *Multilevel refinement for combinatorial optimisation ploblems*, Ann. Oper. Res., 131 (2004), pp. 325–372.

[28] H. Wolkowicz and Q. Zhao, *Semidefinite programming relaxations for the graph partitioning problem*, Discrete Applied Math., 96-97 (1999), pp. 461–479.