A Computational Study of Graph Partitioning

Julie Falkner
Department of Mathematics
Massey University
Palmerston North, New Zealand
Franz Rendl
Technische Universität Graz
Institut für Mathematik
Kopernikusgasse 24, A-8010 Graz, Austria
and
Henry Wolkowicz
University of Waterloo
Department of Combinatorics and Optimization
Waterloo, Ontario, N2L 3G1, Canada

Abstract

Let $G = (N, E)$ be an edge-weighted undirected graph. The graph partitioning problem is the problem of partitioning the node set N into k disjoint subsets of specified sizes so as to minimize the total weight of the edges connecting nodes in distinct subsets of the partition. We present a numerical study on the use of eigenvalue-based techniques to find upper and lower bounds for this problem. Results for bisecting graphs with hundreds of nodes are given. We show that the techniques are very robust and consistently produce upper and lower bounds having a relative gap of typically a few percentage points.

1 Introduction

Let $G = (N, E)$ be an edge-weighted undirected graph with node set $N = \{1, \dots, n\}$, edge set E and weights w_{ij}, $ij \in E$. We consider the problem of partitioning the node set N into k disjoint subsets $S_1, \dots, S_k$ of specified sizes $m_1 \geq m_2 \geq \dots \geq m_k$, $\sum_{j=1}^{k} m_j = n$, so as to minimize the total weight of the edges connecting nodes in distinct subsets of the partition. Partitioning problems are important, for example, in the context of layout problems and VLSI design. Several researchers have developed methods for finding good partitions, and one of the most successful heuristics was proposed by Kernighan and Lin [5] in 1970. A recent survey by Johnson et al. [4] compares several heuristics for the graph bisection problem, which is the problem of partitioning the nodes into just two sets of equal size.

Less attention seems to have been given to estimating the quality of a partition in terms of upper and lower bounds on the optimal solution values. In the early 70s Donath and Hoffman [3] provided an eigenvalue-based upper bound on the weight of the edges not cut by any partition. They also proposed a parametric improvement strategy for their bound and provided numerical results on sparse random graphs with up to 100 nodes. Their results indicate that the gap between lower and upper bounds is fairly large as the number of nodes increases. Recently Boppana [2] has proposed a bounding technique for the special case of graph bisection. He does not give any numerical results but shows that on a certain class of random graphs, his bound is asymptotically tight.

In [6] several new eigenvalue-based bounds for the graph partitioning problem are presented; the purpose of the present paper is to study the performance of these bounds on various classes of graphs.

2 Basic notation and problem statement

Let $G = (N, E)$ denote an undirected graph with edge weights w. We denote by $A = (a_{ij})$ the weighted adjacency matrix of G, i.e.

$$a_{ij} = \begin{cases} w_{ij} & ij \in E \\ 0 & \text{otherwise.} \end{cases}$$

Since G is undirected, A is symmetric. The j−largest eigenvalue of a symmetric matrix M will be denoted by $\lambda_j(M)$. The operator *diag(.)* is used in two ways. If v is a vector, diag(v) is the diagonal matrix formed from v. If M is a (square) matrix, diag(M) is the vector containing the main diagonal of M. The trace of M is denoted by tr(M). The column vector consisting of all ones is denoted by u (or u_l to indicate its size). The (column) vector of row sums of a matrix M is denoted by $r(M)$, thus $r(M) = Mu$. Similarly, $s(M) := u^t Mu$ denotes the sum of all elements of M. We will also make extensive use of the following $l \times (l-1)$-matrix V_l, representing $u^{\perp}$:

$$V_l = \begin{bmatrix} y & \cdots & y \\ 1+x & & x \\ & \cdots & \\ x & & 1+x \end{bmatrix}, \tag{2.1}$$

where $x = \frac{-1}{l+\sqrt{l}}, y = \frac{-1}{\sqrt{l}}$ and $l \geq 2$. It can easily be verified that $V_l{}^t u_l = 0$ and $V_l{}^t V_l = I_{l-1}$.

An instance of a graph partitioning problem is described by a symmetric matrix A of size n and an integer vector $m = (m_1, \ldots, m_k)$ such that $m^t u = n$, defining the specified sizes for the subsets of the partition. We assume without loss of generality that

$$m_1 \geq \ldots \geq m_k \geq 1 \quad \text{and that} \quad k < n.$$

Finally we denote by $w(E)$ the sum of all edge weights of G, i.e. $w(E) = s(A)/2$, and by $w(E_{cut})$ the total weight of the edges cut by an *optimal* partition. Moreover let $w(E_{uncut}) := w(E) - w(E_{cut})$.

The following nonlinear optimization problem solves the graph partitioning problem, see e.g. [6].

$$\text{(GP)} \quad w(E_{uncut}) = \max \frac{1}{2}\text{tr}X^tAX$$

such that

$$X^tX = \text{diag}(m) \tag{2.2}$$

$$Xu_k = u_n; \quad X^tu_n = m \tag{2.3}$$

$$X \geq 0. \tag{2.4}$$

The constraints guarantee that all entries of the $n \times k$ matrix X are either 0 or 1. The nonzero entries of column j of X represent the nodes contained in S_j.

3 Eigenvalue-Based Bounds

We can use the model (GP) to obtain tractable relaxations for graph partitioning and hence upper bounds on $w(E_{uncut})$. Dropping the constraints (2.3) and (2.4) leads to one of the first relaxations for graph partitioning. It was proposed by Donath and Hoffman in the 1970s [3].

$$w(E_{uncut}) \leq \max\{\frac{1}{2}\text{tr}X^tAX : X \text{ satisfies (2.2)}\} = \frac{1}{2}\sum_{j=1}^{k} m_j\lambda_j(A). \tag{3.1}$$

The Donath-Hoffman bound can be further strengthened by dropping only the nonnegativity conditions from (GP), see [6]. In the case where the m_j are all equal (to n/k), the linear term in the bound becomes a constant. From now on we will focus on this special case. **Assumption:** $m_1 = \ldots = m_k = n/k$.

$$w(E_{uncut}) \leq \max\{\frac{1}{2}\text{tr}X^tAX : X \text{ satisfies (2.2),(2.3)}\} = \frac{n}{2k}\sum_{j=1}^{k-1} \lambda_j(V_n^tAV_n) + \frac{1}{2k}s(A). \tag{3.2}$$

This upper bound is attained for

$$X = \frac{1}{k}u_nu_k^t + \sqrt{\frac{n}{k}}V_nZV_k^t, \tag{3.3}$$

where Z contains a set of $k-1$ orthonormal eigenvectors corresponding to the largest $\lambda_j(V_n^tAV_n)$. By construction, this X satisfies the orthogonality constraint (2.2) and the row and column sum constraint (2.3), but it need not be integer, since nonnegativity is dropped.

A further improvement can be achieved along the following lines [3,6]. Let $d \in \Re^n$ and X be an arbitrary feasible partition, i.e. X satisfies (2.2),(2.3),(2.4). Then it can readily be seen that

$$\text{tr } X^t(\text{diag}(d) - \frac{d^tu}{n}I)X = 0.$$

Therefore, see [6], we conclude that

$$\begin{aligned} w(E_{uncut}) &= \max\{\frac{1}{2}\text{tr } X^t(A + \text{diag}(d) - \frac{d^t u}{n}I)X : X \ldots \text{ feasible partition } \} \\ &\leq \max\{\frac{1}{2}\text{tr } X^t(A + \text{diag}(d) - \frac{d^t u}{n}I)X : X \text{ satisfies } (2.2), (2.3)\} \\ &= \frac{1}{2k}s(A) + \frac{n}{2k}\sum_{j=1}^{k-1}\lambda_j(V_n^t(A + \text{diag}(d) - \frac{d^t u}{n}I)V_n) \\ &=: \frac{1}{2k}s(A) + \frac{n}{2k}f(d) \end{aligned}$$

thereby defining $f(d)$.

In [6] it was observed that $\inf\{f(d) : d \in \Re^n\}$ is attained and therefore the best choice for d to produce an upper bound on $w(E_{uncut})$ is to find

$$\min\{f(d) : d \in \Re^n\}.$$

This leads to the following upper bound for partitioning the nodes into subsets of equal size:

$$w(E_{uncut}) \leq \frac{s(A)}{2k} + \min\{\frac{n}{2k}f(d) : d \in \Re^n\}. \tag{3.4}$$

X can also be used to obtain lower bounds on $w(E_{uncut})$, see [6,1]. The problem

$$\max\{\text{tr } X^t AY : Y \text{ satisfies } (2.3)(2.4)\} \tag{3.5}$$

amounts to a linearization of the graph partitioning cost function at X. The optimal Y from this problem is a best partition corresponding to this linearized model. This problem can be solved efficiently as a transportation problem. Note that for linear objective functions the constraint (2.2) can be dropped because the set characterized by (2.3), (2.4) has only integer extreme points.

We will take a closer look at the transportation problem in the case of the bisection problem. We use

$$V_2^t = \frac{1}{\sqrt{2}}(-1 \quad 1), \quad Y = (y_1, u_n - y_1)$$

where y_1 is a zero-one column having $n/2$ entries equal to one. Thus

$$\text{tr } X^t AY = \frac{1}{2}u_n^t Au_n + \frac{\sqrt{n}}{2}(AV_nZ)^t u_n - \sqrt{n}(AV_nZ)^t y_1.$$

Note that the first two summands on the right hand side are constant and therefore maximizing tr$X^t AY$ with respect to Y is equivalent to minimizing

$$(AV_nZ)^t y_1, \text{where } y_1 \in \{0,1\}^n, y_1^t u_n = \frac{n}{2}.$$

It is easy to see that the optimal solution to this problem is obtained simply by setting the elements of y_1 corresponding to the $n/2$ largest elements of AV_nZ to zero and setting the remaining terms to one. Therefore the problem simply requires finding the median of a real vector of n components to get a bisection.

4 Numerical Results

4.1 Introduction

In this section we discuss implementation details and present computational experience for the bound (3.4) for the bisection problem. Since we need to minimize a convex, but possibly nonsmooth function, it seems natural to apply an iterative procedure. Several software packages designed for this type of problem exist, and we have chosen to use the Bundle Trust (BT) method proposed in [7]. In each iteration we have to calculate

$$\lambda_1(V_n^t(A + diag(d))V_n).$$

We use a block Lanczos Algorithm from [8] for this purpose.

At the end of the sequence of BT-iterations, the algebraic multiplicity of the largest eigenvalue is normally greater than one. Thus we have an infinite variety of eigenvectors available for use in the generation of lower bounds. The eigenvectors found by the Lanczos algorithm form an orthonormal basis of the eigenspace. We approximate the search of the entire eigenspace by considering linear combinations of pairs of eigenvectors from the basis. Then we apply the Kernighan-Lin [5] heuristic to the best solution found. Given any feasible bisection, the Kernighan-Lin heuristic attempts to improve it by performing a series of interchanges between the two sets of the partition.

The numerical results concentrate in particular on the relative gap between the upper and lower bounds, which gives an estimate of the distance of the feasible solution from optimality. We define the gap as

$$\text{gap} := \frac{\text{upper bound - lower bound}}{\text{lower bound}}.$$

4.2 Random graphs with uniform edge probabilities

Firstly 27 random graphs with edge weights in the range 1 to 10 were generated. Each edge was generated independent of other edges according to the edge probability p controlling the density of the graph. The graphs had between 50 and 500 nodes and densities ranging from 10% to 100%. Table 4.1 contains the results for 21 of these problems. The mean percentage difference between lower and upper bounds was 3.8, with standard deviation 1.45.

Secondly, 22 further random graphs were generated; these were as above except that the edge weights were now in the range 1 to 100. Similar results were obtained for these problems: the mean percentage difference was 3.6, with standard deviation 1.42. Thirdly, 19 unweighted random graphs were generated. These also had between 50 and 500 nodes, and the densities ranged from 10% to 75%. The mean percentage difference was 3.8, with standard deviation 1.69.

These results are encouraging. The techniques outlined in this paper can obtain, for random graphs, feasible solutions which are within a few percentage points of optimality.

Experiments were also performed with random graphs provided by David Johnson; Johnson et al. [4] use these graphs to test their simulated annealing algorithm,

n	$\|E\|$	Density (%)	Upper bd	Lower bd	% gap
50	136	10	542	516	5.0
50	301	25	1057	1000	5.7
50	630	50	1902	1857	2.4
50	925	75	2486	2418	2.8
50	1225	100	3265	3219	1.4
100	511	10	1881	1792	5.0
100	1249	25	3741	3582	4.4
100	2465	50	6882	6659	3.4
100	3691	75	9716	9481	2.5
100	4950	100	12503	12260	2.0
150	1105	10	3703	3517	5.3
150	2775	25	8215	7930	3.6
150	5603	50	15130	14754	2.6
150	8426	75	21698	21238	2.2
150	11175	100	27784	27389	1.4
200	2015	10	6626	6229	6.4
200	4975	25	14327	13692	4.6
200	9942	50	26455	25698	3.0
300	4489	10	14049	13257	6.0
300	11126	25	31807	30274	5.1
300	22672	50	58876	57306	2.7

Table 4.1: Weighted Random Graphs (Weights in range 0 to 10)

n	$\|E\|$	Density (%)	Upper bd	Lower bd	% gap	Johnson
124	149	2	141	136	3.7	136
124	318	4	271	254	6.7	255
124	620	8	467	442	5.6	442
124	1271	17	853	822	3.8	822
250	331	1	316	301	5.0	302
250	612	2	531	495	7.3	498
250	1283	4	981	925	6.1	926
250	2421	8	1675	1588	5.5	1593
500	625	0.5	600	573	4.7	573
500	1223	1	1071	1001	7.0	1004
500	2355	2	1844	1713	7.6	1727
500	5120	4	3564	3358	6.1	3376
1000	1272	0.25	1228	1172	4.8	1170
1000	2496	0.5	2193	2030	8.0	2045
1000	5064	1	3958	3676	7.7	3697

Table 4.2: Johnson et al. Random Graphs

n	$\|E\|$	Upper bd	Lower bd	% gap	Johnson
500	1282	1281	1280	0.1	1278
500	2355	2347	2329	0.8	2329
500	4549	4493	4370	2.8	4371
500	8793	8629	8381	3.0	8381
1000	2394	2394	2393	0.0	2391

Table 4.3: Johnson et al. Geometric Graphs

which provides good partitions but not upper bounds. The results of our experiments are presented in Table 4.2, together with the cost of the best partitions provided in [4]. For these graphs the mean percentage gap was 6.0 with standard deviation 1.40.

It can be seen that, with one exception, the Johnson partitions are at least as good as the best partitions obtained by our method: on average they are 0.6% better. It is however important to note that Johnson et al. report the best solutions ever found after performing, for each graph, 20 runs of simulated annealing, 2000 runs of Kernighan-Lin, and 2000 runs of a local optimization algorithm. In contrast, we performed just one run of our algorithm on each graph, and each run required only a few calls to the Kernighan-Lin algorithm.

4.3 Geometric Graphs

Geometric graphs come closer to those arising in real-world applications of graph partitioning. These graphs are generated as follows. We use a square grid of given size and select each gridpoint with a predefined probability to represent a vertex of the graph. Then we introduce edges (of weight 1) between selected gridpoints whenever their (Euclidean) distance is below a predefined threshold value.

Thirty geometric graphs were considered, and all of these were either connected, or had at most one isolated node. For these graphs, which had up to 1936 nodes, the mean percentage difference was only 1.3, with standard deviation 0.44. It is encouraging to see that such good feasible solutions can be obtained for graphs which resemble real-world graphs.

We were also able to obtain data for five of the eight geometric graphs which Johnson et al. experimented with. They again performed 20 runs of annealing and thousands of runs of Kernighan-Lin and local optimization, and also found it necessary to develop a special hybrid algorithm to take the geometry of the graphs into account. Our results on these problems are presented in Table 4.3. It can be seen that the percentage gaps are small and that, with one exception, the feasible solutions obtained are as good as or better than the Johnson partitions.

4.4 Summary

The upper bound can be computed efficiently. The bounding technique relies essentially on the ability to calculate the largest eigenvalue of the symmetric matrix

$V_n^t A V_n$. We use a representation of V_n that takes full advantage of the possible sparsity of the adjacency matrix A, without ever multiplying out the triple matrix product.

The upper bound leads to good partitions. An additional advantage of our approach is that we are able to generate, at low computational cost, partitions that are often only a few percentage points from optimality.

The approach is robust and efficient. We tested our bounding procedure on a variety of graphs and consistently obtained lower and upper bounds having a relative gap of less than 10%. It is particularly interesting to see the good performance on graphs having some underlying geometric structure.

Acknowledgements

We thank H. Schramm and J. Zowe for permission to use their BT-Code. J. Falkner was supported by a Research Fellowship from NSERC of Canada, and also thanks the Christian Doppler Labor "Diskrete Optimierung" for financial support of a visit to Graz. F. Rendl and H. Wolkowicz thank the Austrian Ministry of Science and NSERC of Canada for continuing financial support.

References

[1] E.R. Barnes, A. Vannelli, and J.Q. Walker, *A new heuristic for partitioning the nodes of a graph,* SIAM J. Discrete Mathematics, 1 (1988), pp 299–305.

[2] R.B. Boppana, *Eigenvalues and graph bisection: An average case analysis,* In: Proceedings of the 28th Annual Symposium on Computer Science, IEEE (1987), pp 280–285.

[3] W.E. Donath and A.J. Hoffman, *Lower bounds for the partitioning of graphs,* IBM J. of Research and Development, 17 (1973) pp 420–425.

[4] D.S. Johnson, C.R. Aragon, L.A. McGeoch and C. Schevon, *Optimization by simulated annealing: an experimental evaluation; part 1, graph partitioning,* Operations Research, 37 (1989), pp 865–892.

[5] B.W. Kernighan and S. Lin, *An efficient heuristic procedure for partitioning graphs,* Bell Syst. Tech. J., 49 (1970), pp 291–307.

[6] F. Rendl and H. Wolkowicz, *A projection technique for partitioning the nodes of a graph,* Technical Report CORR 90-20, University of Waterloo, Waterloo, Canada (1990).

[7] H. Schramm and J. Zowe, *A version of the bundle idea for minimizing a nonsmooth function: Conceptual idea, convergence analysis, numerical results,* SIAM J. Optimization, 2 (1992), pp 121–152.

[8] D.S. Scott, *Block lanczos software for symmetric eigenvalue problems.* Technical Report 84-08, Oak Ridge National laboratory (1979).