Fast Spectral Methods for Ratio Cut Partitioning and Clustering

Lars Hagen and Andrew Kahng

UCLA Department of Computer Science
Los Angeles, CA 90024-1596

Abstract

The ratio cut *partitioning objective function successfully embodies both the traditional min-cut and equipartition goals of partitioning. Fiduccia-Mattheyses style ratio cut heuristics have achieved cost savings averaging over 39% for circuit partitioning and over 50% for hardware simulation applications [15]. In this paper, we (i) show a theoretical correspondence between the optimal ratio cut partition cost and the second smallest eigenvalue of a particular netlist-derived matrix, and (ii) present fast Lanczos-based methods for computing heuristic ratio cuts from the eigenvector of this second eigenvalue. Results are better than those of previous methods, e.g., by an average of 17% for the Primary MCNC benchmarks. An efficient* clustering *method, also based on the second eigenvector, is very successful on the "difficult" input classes in the CAD literature. The paper concludes with extensions and directions for future work.*

1 Preliminaries

As system complexity increases, the divide-and-conquer approach is used to keep the circuit design process tractable. The recursive decomposition of the synthesis problem is reflected in the hierarchical organization of boards, multi-chip modules, integrated circuits, macros, etc. Since early decisions will constrain all succeeding decisions, the high-level layout phases are critical to the quality of the final layout. In particular, without a successful partitioning algorithm, good solutions to the placement, global routing and detailed routing problems will be impossible. As noted in, e.g., [5], partitioning comprises the essence of many basic CAD problems, including packaging of designs, clustering analysis, and partition analysis for high-level synthesis.

Because signal delays will typically decrease as we move downward in the design hierarchy (e.g., on-chip communication is faster than inter-chip communication), the traditional metric for the decomposition is the number of signal nets which cross between layout subproblems. Minimizing this number is the essence of partitioning.

1.1 Basic Partitioning Formulations

A standard mathematical model associates a graph $G = (V, E)$ with the circuit netlist; vertices in V represent modules and edges in E represent signal nets. The vertices and edges of G may be weighted to reflect module area and the importance of a connection. Because nets often have more than two pins, the netlist is more generally represented by a *hypergraph* $H = (V, E')$, where hyperedges in E' are the subsets of V contained by each signal [12]. There are several standard transformations from the hypergraph representation of a circuit to a graph representation, as discussed below.

Two common partitioning formulations are: (i) *minimum-cut*, where we find the partition of V into disjoint U and W such that the number of edges $e = \{u, w\}$, $u \in U$ and $w \in W$, is minimized; and (ii) *minimum-width bisection*, which adds the constraint $|U| = |W|$. Because minimum-cut tends to divide modules very unevenly, research has centered on the minimum-width bisection problem. Since this problem is NP-complete, several heuristic approaches have been used: top-down recursive bipartitioning [5]; bottom-up clustering and aggregation [8] [14]; and spectral methods.

The last class of methods uses eigenvalues or eigenvectors of matrices derived from the undirected netlist graph $G = (V, E)$. Often, if $|V| = n$ we use the $n \times n$ *adjacency matrix* $A = A(G)$, where $a_{vw} = 1$ if $(v, w) \in E$ and $a_{vw} = 0$ otherwise. If G has weighted edges, then a_{vw} is equal to the weight of $(v, w) \in E$, and by convention $a_{vv} = 0$ for all $v \in V$. If we let $d(v)$ denote the degree of node v (i.e., the sum of the weights of all edges incident to v), we obtain the $n \times n$ diagonal *degree matrix* D defined by $D_{ii} = d(v_i)$. The eigenvalues and eigenvectors of such matrices are the subject of the relatively recent subfield of graph theory dealing with *graph spectra*. Early theoretical work connecting graph spectra and partitioning is due to Barnes, Donath and Hoffman [1] [5] [6]. More recent eigenvector and eigenvalue methods have dealt with both module placement and graph min-cut bisection, and are surveyed in [12].

1.2 Ratio Cuts

In practice, requiring an exact bisection (rather than, say, a 60% - 40% partition of module area) is unnecessarily restrictive. In the instance of Figure 1, even the optimal bisection will not yield a very sensible partitioning. Penalty function methods have been used to permit not-quite-perfect bisections, but can require rather ad hoc thresholds and penalties. This leads to the very natural *ratio cut* partitioning metric,

CH3026-2/91/0000/0010/$01.00 © 1991 IEEE

recently introduced for CAD applications by Wei and Cheng [15].

Minimum Ratio Cut: Given $G = (V, E)$, find the partition of V into disjoint U and W such that $\frac{e(U,W)}{|U| \cdot |W|}$ is minimized, where $e(U, W)$ is the number of edges $e = \{u, w\}$, $u \in U$ and $w \in W$.

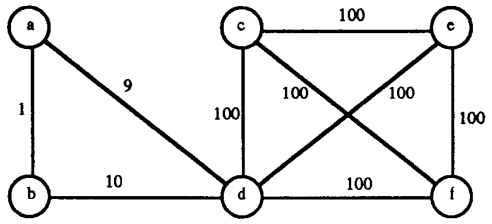

Figure 1: Minimum-cut ($a|bcdef$, cost = 10) and minimum-width bisection ($abd|cef$, cost = 300) are less natural than the optimal ratio cut ($ab|cdef$, cost = 19).

The ratio cut metric gives the "best of both worlds" in that the numerator embodies minimum-cut, while the denominator favors an even partition (see Figure 1). Recent work shows the utility of this metric; [15] reports average cost improvements of 39% over results from the standard Fiduccia-Mattheyses method [7] on industry benchmarks. The ratio cut also has important uses in other areas of CAD: (i) in design for testability, a sparse partition will result in subcircuits that have fewer I/O's, thus requiring fewer test vectors; and (ii) in hardware simulation, runtime is proportional to the number of interconnections at a given level of the hierarchy, so a sparse cut reduces simulation costs. Savings of up to 70% have been achieved for such applications in a number of industry settings [15]. As the minimum ratio cut is easily seen to be NP-complete, multicommodity flow based approximations have been proposed [4], but are prohibitively expensive. Wei and Cheng [15] therefore employed the iterative shifting and group swapping scheme of Fiduccia-Mattheyses. The present work gives new results indicating that spectral heuristics based on matrix eigenvalues can be used to compute even better ratio cuts.

The remainder of this paper is organized as follows. In Section 2, we show a new theoretical connection between graph spectra and the optimal ratio cut. Section 3 presents EIG1, our basic spectral heuristic for minimum ratio cut partitioning and clustering analysis. We derive a good ratio cut partition directly from the eigenvector associated with the second eigenvalue of $Q = D - A$. Section 4 gives performance results and comparisons with previous work, using benchmarks from the MCNC suite as well as classes of "difficult" clustering inputs from the literature. Section 5 concludes with directions for future work.

2 A New Connection: Graph Spectra and Ratio Cuts

Recall that two standard matrices derivable from the circuit netlist are the adjacency matrix A and the diagonal degree matrix D. We use the matrix $Q = D - A$ mentioned above, which we may view as the discrete analog of the Laplace Δ operator. One easily shows several basic properties of Q: (1) Q is symmetric; (2) Q is non-negative definite, so that $xQx = \sum_{i,j} q_{ij} x_i x_j \geq 0$ and all eigenvalues of Q are non-negative; and (3) the smallest eigenvalue of Q is 0 with eigenvector $\mathbf{1} = (1, 1, ..., 1)$. Defining λ to be the second smallest eigenvalue of Q, we also obtain (4) (using the notation $\sum' \equiv \sum_{(i,j) \in E}$ to denote summation over all edges) $xQx = xDx - xAx = \sum'(x_i - x_j)^2$, and (5) by the Rayleigh Principle [9], $\lambda = \min_{x \perp \mathbf{1}, x \neq 0} \frac{xQx}{|x|^2}$.

Properties (4) and (5) lead directly to a relationship between the *optimal ratio cut cost* and λ [10]:

Theorem One: Given a netlist graph $G = (V, E)$ with adjacency matrix A, diagonal degree matrix D, and $|V| = n$, the second smallest eigenvalue λ of $Q = D - A$ yields a lower bound on the cost c of the optimal ratio cut partition, with $c \geq \frac{\lambda}{n}$. □

The $\frac{\lambda}{n}$ lower bound in Theorem One for the optimal partition cost under the ratio cut metric is a tighter result than can be obtained using the early techniques of Donath et al., which essentially rely on the Hoffman-Wielandt inequality [12]. Also note that if we restrict the partition to be an *exact* bisection, Theorem One subsumes the result of Boppana [2]. Given Theorem One, our basic partitioning approach is to compute $\lambda(Q)$ and the corresponding eigenvector v, then use v to construct a heuristic ratio cut.

3 New Ratio Cut Heuristics

Practical implementation of this approach requires closer examination of four main issues: (i) the transformation of the netlist hypergraph into a graph G; (ii) the calculation of the second eigenvector v; (iii) the construction of a heuristic ratio cut partition from v; and (iv) a possible post-processing stage to improve the heuristic ratio cut. We briefly make several observations on these topics.

Hypergraph Models: We have examined two heuristic mappings from hyperedges in the netlist to graph edges in G. The first mapping is via the standard clique model (where a k-pin net is represented by a complete graph on its k modules, with each edge weight equal to $1/(k-1)$). The second mapping is given by using the standard clique model followed by an added sparsifying heuristic, e.g., (i) ignoring less significant (e.g., non-critical or very large) nets, or (ii) thresholding small Q_{ij} to 0 until the matrix has sufficiently few nonzeros.

Numerical Methods: With regard to algorithm implementation, it may at first appear that

eigenvalue computations are too complicated to be practical. However, there are significant algorithmic speedups based on our need to calculate only a *single* (the second-smallest) eigenvalue of a *symmetric* matrix. Furthermore, netlist sparsity due to fanout bounds and design styles allows us to apply the block Lanczos algorithm [9]. We use an adaptation of an existing Lanczos implementation [13] to compute the second-*largest* eigenvalue and the corresponding eigenvector of the matrix $Q' = A - D$ (this is equivalent to computing the negative of the second-smallest eigenvalue of $Q = D - A$, and is preferable because of faster convergence to largest eigenvalues). The numerical code is portable Fortran77; all other code in our system is written in C.

Deriving a Ratio Cut: We have considered a number of heuristics for constructing the ratio cut partition from the second eigenvector v; results below are for our EIG1 method, which sorts the v_i and then determines the splitting rank r, $1 \leq r \leq n-1$, that yields the best ratio cut cost when nodes with rank $> r$ are placed in U and nodes with rank $\leq r$ are placed in W. The cost of evaluating all $n-1$ splitting ranks, i.e., $n-1$ distinct partitions, is asymptotically dominated by the Lanczos computation. With these implementation decisions, our algorithm EIG1 is as follows:

Algorithm EIG1
Input $H = (V, E')$ = netlist hypergraph
Transform each k-pin hyperedge of H into a clique in $G = (V, E)$ with uniform edge weight $\frac{1}{k-1}$;
Compute A, D associated with G;
Compute second-largest eigenvalue of $Q' = A - D$ by Lanczos algorithm ($= -\lambda(Q)$);
Compute eigenvector v associated with $\lambda(Q)$;
Sort components of v;
Find best splitting point for indices (modules) using ratio cut metric.

Figure 2: High-level outline of **Algorithm EIG1**.

As shown in the next section, EIG1 generates initial partitions which are already significantly better than the output of RCut1.0, the iterative Fiduccia-Mattheyses style algorithm of Wei and Cheng [15]. In fact, using the single sorted eigenvector we often find *many* partitions that are better than the Fiduccia-Mattheyses result.

Clustering Is "Free": A bonus from our approach is the observation that clustering is "free" with the spectral computation, since the second eigenvector v contains both partitioning *and clustering* information. In Section 4 below, we demonstrate that the sorted second eigenvector by itself can effectively identify natural clusters in the classes of "difficult" inputs proposed in Bui et al. [3] and Garbers et al. [8]. It is also reasonable to interpret large *gaps* between adjacent components of the sorted eigenvector as delimiters of natural circuit clusters, and "local minimum" (with respect to the ratio cut metric) partitions of the sorted eigenvector may also delineate clusters.

4 Experimental Results

4.1 Ratio Cut Partitioning

In this section, we present computational results using the EIG1 algorithm. We partitioned the MCNC Primary1 and Primary2 standard-cell and gate-array benchmarks, applying Lanczos code to the netlist and then using actual module areas in selecting the best split of the sorted eigenvector. Table 1 compares results with Rcut1.0 output [15] (further benchmark results are reported in [10]). On average, the EIG1 results are 17.6% better. The EIG1 results are completely unrefined: no local improvement has been performed on the eigenvector partition. Also note that the results in [15] are already an average of 39% better than Fiduccia-Mattheyses output in terms of the ratio cut metric (comparing the best of 10 Rcut1.0 runs to the best of 20 F-M runs).

Test problem	W-C (RCut1.0) Areas/Cutsize	H-K (EIG1) Areas/Cutsize	H-K / W-C Ratio
PrimGA1	502:2929 / 11	751:2681 / 15	.989
PrimGA2	2488:5885 / 89	2522:5852 / 78	.855
PrimSC1	1071:1682 / 35	588:2166 / 15	.602
PrimSC2	2332:5374 / 89	2361:5345 / 78	.859

Table 1: Comparison with [20] of ratio cut values on MCNC Primary benchmarks (area sums are rounded to integers).

The CPU times required by our algorithm were very competitive with those cited in [15]. For example, the eigenvector computation for PrimSC2, using our default convergence tolerance of 10^{-4}, required 83 seconds of CPU time on a Sun4/60, versus 204 seconds of CPU for 10 runs of RCut1.0. All of our experiments indicate that the spectral approach is well-suited to partitioning of cell-based designs as well as other large-scale partitioning applications in CAD where the input is an unweighted netlist hypergraph, e.g., partition for testability or hardware simulation [10].

4.2 Clustering

Finally, straightforward interpretation of the second eigenvector yields good clusterings on the two classes of "difficult" inputs in the literature. The first type of input is given by the random graph model $G_{Bui}(2n, d, b)$, developed by Bui et al. [3] in analyzing graph bisection algorithms. Here, the random graphs have $2n$ nodes, are d-regular and have minimum bisection width almost certainly equal to b. We analyzed random graphs with between 100 and 800 nodes and with parameters $(2n, d, b)$ as in Bui's experiments (Table I, p. 188 of [3]). In all cases, the second eigenvector immediately yielded the correct clustering. The second type of input is given by the $G_{Gar}(n, m, p_{int}, p_{ext})$ random model of Garbers et al. [8], which prescribes

n clusters of m nodes each, with all edges inside clusters independently present with probability p_{int} and all edges between clusters independently present with probability p_{ext}. We tested a number of 1000-node examples of such clustered inputs, using the same values (n, m, p_{int}, p_{ext}) as in Table 1 of [8]. In all cases, quite accurate clusterings were immediately evident from the eigenvector, with most clusters completely contiguous in the sorted list, and occasionally pairs of clusters being intermingled. Sample results for both input types are reproduced in [10].

5 Extensions and Conclusions

Many research questions are still under investigation. A promising speedup entails using *condensing* [3] to reduce problem size by finding a random maximal matching on the graph, then using the edges of the matching to condense node pairs into single nodes. After solving the condensed problem, nodes can be re-expanded and an iterative improvement stage may follow. Although there is the drawback of yielding a denser input to the eigenvector computation, the sparsifying heuristics mentioned above may be applied so that a net speedup is still obtained. A second variant weakens *convergence criteria* in the Lanczos implementation, reducing the accuracy of the eigenvector calculation; preliminary experiments indicate that for, e.g., the PrimGA2 benchmark, we can speed up our standard Lanczos computation by a factor of between 1.3 and 1.7 without loss of solution quality. Parallel implementations of the Lanczos algorithm on medium- and large-scale vector processors are also of interest. With any variant, the ratio cuts may be improved with standard iterative techniques; this is also under investigation. Finally, following Wei and Cheng, we may apply EIG1 to other CAD applications such as design for testability and the mapping of logic for hardware simulation. The results in Section 4 suggest that for applications where the Fiduccia-Mattheyses type of ratio cut heuristic has already been successful, our spectral construction will provide further improvements.

In conclusion, we have presented new theoretical analysis showing that a second-eigenvalue construction yields good partitions under the ratio cut metric. Sparse matrix techniques lead to effective, parallelizable algorithms which are competitive with the fastest current methods for circuit partitioning. On standard-cell and gate-array industry benchmarks, our solution quality was significantly improved over that of RCut1.0 [15]. Because the result is derived from a single deterministic execution of the algorithm, multiple runs are not required.

No previous work applies numerical methods to ratio cut partitioning, since the mathematical basis of ratio cuts has only recently been developed. From a historical perspective it is intriguing that spectral methods have not been more popular for other problem formulations such as bisection or k-partition, despite the early results of Barnes, Donath and Hoffman and the availability of standard packages for matrix computations. We speculate that this is for several reasons. First, advances in numerical methods and VLSI CAD have followed disjoint paths; only recently have large-scale numerical computations become reasonable tasks on workstation platforms. Second, early theoretical bounds and empirical performance of spectral methods for bisections were not generally encouraging. Finally, it has only been with growth in problem complexity that possible scaling weaknesses of iterative approaches have been exposed. In any case, we believe that the spectral approach to partitioning, first developed by Barnes, Donath and Hoffman twenty years ago, merits renewed interest in the context of a number of basic CAD applications.

6 Acknowledgements

We are grateful to Professor C. K. Cheng of UCSD, Dr. Arthur Wei of Cadence Design Systems and Mr. Chingwei Yeh of UCSD for providing RCut1.0 code [15]. Dr. Horst Simon of NASA Ames Research Center provided an early version of his Lanczos implementation.

References

[1] E. R. Barnes, "An Algorithm for Partitioning the Nodes of a Graph", *SIAM J. Alg. Disc. Meth.* 3(4) (1982), pp. 541-550.

[2] R.B. Boppana, "Eigenvalues and Graph Bisection: An Average-Case Analysis", *IEEE Symp. on Foundations of Computer Science*, 1987, pp. 280-285.

[3] T. N. Bui, S. Chaudhuri, F. T. Leighton and M. Sipser, "Graph Bisection Algorithms with Good Average Case Behavior", *Combinatorica* 7(2) (1987), pp. 171-191.

[4] C.K. Cheng and T.C. Hu "Maximum Concurrent Flow and Minimum Ratio Cut", Technical Report CS88-141, Univ. of California, San Diego, Dec. 1988.

[5] W.E. Donath, "Logic Partitioning", in *Physical Design Automation of VLSI Systems*, B. Preas and M. Lorenzetti, eds., Benjamin/Cummings, 1988, pp. 65-86.

[6] W.E. Donath and A.J. Hoffman, "Lower Bounds for the Partitioning of Graphs", *IBM J. Res. Dev.* (1973), pp. 420-425.

[7] C.M Fiduccia and R.M. Mattheyses, "A Linear Time Heuristic for Improving Network Partitions", *ACM/IEEE Design Automation Conf.*, 1982, pp. 175-181.

[8] J. Garbers, H. J. Promel and A. Steger, "Finding Clusters in VLSI Circuits" *extended version* of paper in *Proc. IEEE Intl. Conf. on Computer-Aided Design*, 1990, pp. 520-523.

[9] G. Golub and C. Van Loan, *Matrix Computations*, Baltimore, Johns Hopkins University Press, 1983.

[10] L. Hagen and A. B. Kahng, "Fast Spectral Methods for Ratio Cut Partitioning and Clustering", UCLA CS Dept. TR-910012, April 1991.

[11] B.W. Kernighan and S. Lin, "An Efficient Heuristic Procedure for Partitioning of Electrical Circuits", *Bell System Technical J.*, Feb. 1970.

[12] T. Lengauer, *Combinatorial Algorithms for Integrated Circuit Layout*, Wiley-Teubner, 1990.

[13] A. Pothen, H. D. Simon and K. P. Liou, "Partitioning Sparse Matrices with Eigenvectors of Graphs", *SIAM J. Matrix Analysis and its Applications* 11 (1990), pp. 430-452.

[14] G. Vijayan, "Partitioning Logic on Graph Structures to Minimize Routing Cost", *IEEE Trans. on CAD* 9(12) (1990), pp. 1326-1334.

[15] Y.C. Wei and C.K. Cheng, "Towards Efficient Hierarchical Designs by Ratio Cut Partitioning", *IEEE Intl. Conf. on Computer-Aided Design*, 1989, pp. 298-301.