On the Intrinsic Rent Parameter and Spectra-Based Partitioning Methodologies*

L. Hagen, A. B. Kahng, F. J. Kurdahi† and C. Ramachandran†

UCLA CS Dept., Los Angeles, CA 90024-1596
† UCI ECE Dept., Irvine, CA 92717

Abstract

The complexity of circuit designs has necessitated a top-down approach to layout synthesis. A large body of work shows that a good layout hierarchy, or *partitioning tree*, as measured by the associated Rent parameter, will correspond to an area-efficient layout. We define the *intrinsic Rent parameter* of a netlist to be the minimum possible Rent parameter of any partitioning tree for the netlist. Experimental results show that spectra-based ratio cut partitioning algorithms yield partitioning trees with the lowest observed Rent parameter over all benchmarks and over all algorithms tested. For examples where the intrinsic Rent parameter is known, spectral ratio cut partitioning yields a partitioning tree with Rent parameter essentially identical to this theoretical optimum. These results have deep implications with respect to both the choice of partitioning algorithms for top-down layout, as well as new approaches to layout area estimation. The paper concludes with directions for future research, including several promising techniques for fast estimation of the (intrinsic) Rent parameter.

1 Introduction

As VLSI system complexity and the number of implementation alternatives continue to increase, top-down hierarchical approaches have been widely adopted in layout synthesis. Recursive calls to a bipartitioning algorithm will generate a circuit hierarchy, or *partitioning tree*, that is typically used either to guide the placement/routing phases of layout, or to afford early estimates of layout area and wireability. The bipartitioning algorithm essentially reveals the natural circuit structure in the form of a binary tree of disjoint subcircuits, where connectivity between subcircuits is minimized.

Traditionally, the quality of a partitioning algorithm is measured by the number of nets cut in a 2-way partition of some benchmark circuit. However, such a metric fails to capture the integral role played by the partitioning algorithm in an *overall* layout synthesis or area estimation *process* – i.e., via the recursive top-down application of the algorithm. Since the quality of a partitioning tree has a direct bearing on the quality of the resulting layout, we would like to find the partitioning algorithm that generates the best tree of

*This work was supported in part by NSF MIP-9110696, NSF MIP-8909677, and a SIGDA Graduate Fellowship.

subcircuits. In this work, we propose using the *Rent parameter*, which is a well-established quality measure of a generated partitioning tree, as a quality measure for the underlying partitioning algorithm itself.

The Rent parameter [28] characterizes a power-law relationship between the number of external terminals of a subcircuit in the layout and the number of modules in the subcircuit. This parameter has been studied extensively in the field of area estimation, where it affords accurate predictions of the wiring requirements for a given partitioning hierarchy. In particular, given two partitioning trees for the same design, the one with lower Rent parameter will lead to less wirelength and consequently a denser final layout (see the review of relevant literature given in Section 2.1 below). The Rent parameter is therefore well suited as a quality measure for the complete tree of subcircuits generated by a particular partitioning algorithm.

Our work compares various partitioning algorithms with respect to the Rent parameters of their induced partitioning trees, with the goal of identifying an algorithm that yields trees having Rent parameter closest to optimal. To this end, we introduce the notion of the *intrinsic* Rent parameter of a given circuit: the intrinsic Rent parameter is the minimum possible Rent parameter of *any* partitioning tree for the circuit. Such a lower bound gives a measure of the *required* layout area, independent of layout strategy. An added advantage of our approach is that it allows comparison of the utility of partitioning algorithms, independent of possible differences between the algorithms' individual objective functions.

Extensive experimental results show that so-called spectra-based partitioning algorithms (i.e., methods based on computing eigenvalues and eigenvectors of a particular netlist-derived matrix) are significantly superior to traditional iterative methods such as the Fiduccia-Mattheyses approach. Specifically, we have found that an algorithm which recursively partitions an eigenvector-derived *linear* ordering of the modules to minimize the *ratio cut* objective will yield partitioning trees with better Rent parameter than those produced by any other algorithm. Moreover, with each test case for which the intrinsic Rent parameter is known, the spectra-based ratio cut partitioning tree has Rent parameter essentially *identical* to the theoretical lower bound.

This result has key implications in two areas. First, it affirms previous work of Wei and Cheng [43], who were the first to propose the ratio cut metric as a partitioning objective, and of Hagen and Kahng [15], who proposed using the spectral approach for ratio cut partitioning. Second, our work naturally leads to the development of better predictive layout models, enabling a more efficient search of the space of layout solutions. Indeed, to facilitate this latter application, we center on the spectra-based ratio cut approach and propose several methods for rapidly estimating the (intrinsic) Rent parameter value. These methods include: (i) computing the Rent parameter of incomplete partitioning trees, (ii) constructing a complete partitioning tree using a single eigenvector computation, and (iii) using the second-smallest eigenvalue of the netlist graph *Laplacian* to estimate the intrinsic Rent parameter. Preliminary experimental results show all three of these methods to be promising.

The remainder of this paper is organized as follows. Section 2 reviews Rent's rule and its application to wireability analysis and area estimation. Section 2 also provides a careful definition of the intrinsic Rent parameter of a given circuit. In Section 3, we give a taxonomy of partitioning algorithms, concentrating on those which are most useful for both hierarchical layout and rapid area estimation applications. Section 4 presents experimental results concerning the intrinsic Rent parameter and shows the relative merits of various partitioning algorithms with respect to their induced, "algorithm-dependent" Rent parameters. The paper concludes in Section 5 with a discussion of techniques for rapid estimation of the Rent parameter, as well as other directions for future research.

2 Rent's Rule

Rent's rule is an empirical relation observed in "good" layouts; it reflects a power-law scaling of the number of external terminals of a given subcircuit with the number of modules in the subcircuit. Specifically,

$$T = k \cdot C^p \tag{1}$$

where T is the average number of external terminals (pins) in a subcircuit or partition; k is *Rent's constant*, a scaling constant which has empirically been found to correspond to the average number of pins per module; C is the number of modules in the subcircuit (or partition); and p is the *Rent parameter* or *Rent exponent*, with $0 \leq p \leq 1$.

This relation was first observed by E. F. Rent of IBM in the late 1960s and independently by several others, e.g., Donath [4] derived the same relation from a stochastic model of a hierarchical design process. Landman and Russo [28] performed an extensive study of the relation via partitioning experiments on large "real-life" circuits, and observed Rent parameter values p between 0.47 and 0.75.[1] Following Mandelbrot [30] and Keyes [21], one may view Rent's rule as a dimensionality relationship between pinout of a module and the number of gates in the module. This is in some sense a surface area to volume relationship where, for

[1]Landman and Russo also observed that Rent's rule is actually a two-region relationship: the power law relation (1) above holds in region I, while region II is governed by a more complex relation. In our work, as in [28], we consider scaling laws fitted to the region I domain (cf. the discussion in Section 4.1 below). We also note that in a subsequent paper [34], Russo concluded that, for a given fixed partitioning algorithm, p tends to be high for high performance circuits, and low for low performance circuits. As simple examples, consider that $p \approx 0$ for a shift register in which data is loaded serially, and $p \approx 1$ for a latch circuit where data is loaded in parallel.

example, "intrinsically 2-dimensional" circuits such as memory arrays, PLAs, or meshes will have optimal layouts with $p = 1/2$. Kurdahi and Parker [27] have noted that a Rent parameter value $p > 0.5$ implies that wires *must* grow longer as circuit size increases; in other words, such a circuit cannot be embedded in two dimensions without "dilation", and the *relative* contribution of wiring to layout area will grow with the size of the circuit. For example, a 3-dimensional mesh topology requires a layout having $p \geq 2/3$ [26], and indeed such a topology cannot be embedded in the plane without the introduction of long wires. Note that the tractability of the Rent parameter analysis for meshes will allow us to use 2-D and 3-D meshes as supplementary benchmarks (see Section 4 below) for which the optimum Rent parameter values are known.

2.1 Relationships Between the Rent Parameter and Layout Area

The value of the Rent parameter is closely related to the layout area of a given circuit. Three researchers in particular – Donath [5], Feuer [9] and Sastry [36] [35] – have proposed and experimentally verified relationships between the Rent parameter and the average wire length of a layout. Together, these works show that a lower Rent parameter will result in lower average wire length, which in turn generally implies smaller wiring area and less congestion in the layout.

The early work of Donath [5] assumes a hierarchical partitioning and placement of gates on a square array of slots. Donath found that a circuit with C gates and Rent parameter p will have average wire length $\bar{r}$ given by

$$\bar{r} \sim \begin{cases} C^{p-1/2}, & p > \frac{1}{2} \\ \log C, & p = \frac{1}{2} \\ f(p), & p < \frac{1}{2} \end{cases} \tag{2}$$

with $f(p)$ being independent of the number of gates, C. We make two observations. First, the experimental results in [5] show a large difference between the predicted and actual values of $\bar{r}$; however, $\bar{r}$ was generally found to vary up or down in accordance with the proposed model. Second, Donath's model has obvious ties to the "dimensional" intuition noted above [30] [21] [27]: Donath assumes that circuit layouts are two-dimensional, and the "critical value" of $p = \frac{1}{2}$ in (2) corresponds to the transition between planar and non-planar circuits. In other words, an intrinsically planar netlist ($p \leq \frac{1}{2}$) can be placed such that all connections essentially lie between nearest neighbors, with the average wire length being independent of, or slowly growing in, C. However, this no longer holds if $p > \frac{1}{2}$ and the circuit is non-planar.[2]

[2] The correspondence between dimensionality and Donath's wire length estimate in Equation (2) is supported by the work of Masaki and Yamada [31], who extended Donath's work to model 3-D structures. Their estimate of average wire length identifies a "critical value" of $p = \frac{2}{3}$, which would be expected in this higher dimension.

The work of Feuer [9] establishes a formula for the wire length distribution in terms of the Rent parameter. Using a continuous model, Feuer defines a partition function, $I(R)$, which gives the number of connections born inside and terminating outside a circle of radius R in the Manhattan plane. This partition function is easily reconciled with Rent's rule, and $I(R)$ moreover yields a wire length distribution by integrating an infinitesimal strip around the circle over the whole plane. In this way, Feuer derives the following expression for average interconnection length:

$$\frac{\bar{r}}{\delta} = \sqrt{2} \cdot \frac{2p(3+2p)}{(1+2p)(2+2p)} \frac{C^{p-0.5}}{(1+C^{p-1})} \tag{3}$$

where $\frac{\bar{r}}{\delta}$ is the average wire length expressed in terms of *gate length* units, δ. The predictions of Equation (3) were compared to experimental data on real chips, assuming a "default" constant value of $p = 2/3$, and the results indicate a close agreement. While the model does not accomodate values of $p \leq 1/2$, it should be noted that such low values of p are rarely observed in real layouts [26].

Finally, the Ph.D. thesis of Sastry [35] also assumes a continuous model of logic and uses techniques from reliability theory to derive a relationship between Rent's rule and the wire length distribution. The main result is that if Rent's rule holds, the wire lengths of a layout will follow a Weibull distribution. The average of the Weibull distribution is then used as an estimate of the average wire length:

$$\frac{\bar{r}}{\delta} = \frac{1}{p} \left(\frac{1}{k}\right)^{\frac{1}{p}} \Gamma\left(\frac{1}{p}\right) \tag{4}$$

where k and p are as defined in Rent's rule (Equation (1) above) and Γ is the gamma function. As with the results reported in [5] and [9], estimates from Equation (4) agree closely with actual average wire length values in the (gate array) layouts analyzed.[3]

These results motivate the use of the Rent parameter of a given partitioning tree as a quality measure for the corresponding partitioning algorithm.[4] However, in the next subsection we note that "the" Rent parameter is well-defined only with respect to a fixed methodology for extracting this parameter from a given circuit. Therefore, before we can use the Rent parameter as a quality measure for partitioning algorithms, we find it necessary to extend and solidify the original methodology sketched by Landman and Russo in [28].

[3] Some confirmation of these models was provided by Kurdahi and Parker[26], who compared the accuracy of both Feuer's and Sastry's results using serial multipliers as a testbed. It was found that the two methods of [9] and [35] are of similar quality, e.g., Feuer's estimates of the average wire lengths are within 23% of the actual values, using a "default" value of $p = 0.720$ as the Rent parameter.

[4] Wirelength estimates also lie at the heart of the analytic area estimation methods used in synthesis; see, e.g., El Gamal [12], Heller [8], Kurdahi [27] and Sastry [35]. We note that other authors have proposed wire length distribution models which do not use Rent's rule. These include Sechen[39], Pedram and Preas[33], and Hamada et al.[19], all of whom estimate interconnection length based on local *neighborhood analysis* approaches: each signal net is propagated to random pin locations within a rectangular grid which has the same size as the neighborhood population of the net. These neighborhood analysis methods are also reported to be quite effective, even though they rely solely on local wire length estimates.

2.2 Defining the Intrinsic Rent Parameter

Landman and Russo [28] made the fundamental observation that "the" Rent parameter will depend upon both the structure of the circuit and the partitioning algorithm chosen. This notion of an *algorithm-dependent* Rent parameter motivates our hypothesis that the Rent parameter can be used to characterize the quality of a given partitioning algorithm. Our results below show that for any given circuit, different partitioning algorithms will yield partitioning trees with widely varying Rent parameters, implying that the wirelength and layout area depend considerably on the choice of the partitioning algorithm used in the top-down layout process. Moreover, our experimental results demonstrate that the relative ordering of the algorithm-dependent Rent parameters is very consistent across a variety of benchmark circuits. The Rent parameter thus objectively distinguishes those partitioning algorithms that are most useful within the hierarchical layout approach.

In what follows, we propose the notion of an *intrinsic Rent parameter*, denoted by p^*, which is the *minimum possible* Rent parameter attainable over any partitioning tree of the given circuit. Because the Rent parameter is in some sense a "folklore" concept that has been computed in various ad hoc ways throughout the literature, we devote the remainder of this section to formally defining the concepts of a partitioning tree and the associated Rent parameter computation.

The early work of Landman and Russo [28] computed "the" Rent parameter of a circuit as follows. By using a multi-way partitioning algorithm, Landman and Russo could generate *partitioning instances* P_i, each of which corresponded to a partition of the entire circuit into disjoint subcircuits. To obtain each partitioning instance, ad hoc constraints on the maximum size and the maximum number of pins of any subcircuit were specified, and a heuristic, iterative process was invoked to vary these constraints until they could be satisfied by the output of the multi-way partitioning algorithm. For the i^{th} partitioning instance, the average subcircuit size, $\bar{C}_i$, and the average number of pins per subcircuit, $\bar{T}_i$, were both calculated to yield a single data point. In order to correlate the experimental data to Rent's rule, the original relationship $T = k \cdot C^p$ was re-expressed as $\log T = \log k + p \cdot \log C$, which is a straight-line equation with intercept $\log k$ and slope equal to the Rent parameter p in Rent's rule. Thus, Landman and Russo could compute p by plotting all of the $(\bar{C}_i, \bar{T}_i)$ data points on a *log-log* scale, and then using linear regression to find a straight-line fit. The slope of the fitted line yielded the Rent parameter of the circuit.

Currently, the partitioning algorithms most popular in top-down layout synthesis will recursively divide a circuit into two subcircuits. Thus, we define a partitioning instance within the context of the *partitioning* *tree* obtained by the recursive application of a given bipartitioning algorithm.[5]

Definition: Given a circuit C with n modules, a *partitioning tree* of C is any full binary tree of subcircuits having n leaves. A binary tree is a full binary tree if every internal node (i.e., a subcircuit that has been bipartitioned[6]) has exactly two children (i.e., the resulting two smaller, disjoint subcircuits).

In practice, we only apply the bipartitioning algorithm to subcircuits that are larger than a prescribed threshold, denoted by c_0. This gives rise to the following:

Definition: Given a circuit C and a positive integer c_0, a *restricted partitioning tree of C with threshold c_0* is any full binary tree of subcircuits, such that every internal node corresponds to a subcircuit with at least c_0 modules, and every leaf node corresponds to a subcircuit with fewer than c_0 modules.

Note that the definition of a restricted partitioning tree will subsume the original partitioning tree definition for $c_0 = 1$. Thus, for simplicity we will henceforth use the term "partitioning tree" to refer to a possibly restricted partitioning tree, specifying the value of c_0 as necessary. Generation of a partitioning tree is accomplished using the algorithm of Figure 1. Note that in order to later perform the Rent parameter computation, the size and number of external pins of the two resulting subcircuits is recorded after each execution of the bipartitioning algorithm.

Generate-Partitioning-Tree(C, ALG, c_0)
Input: Circuit netlist C, threshold c_0 and fixed bipartitioning algorithm ALG
Output: Partitioning tree (i.e., binary tree of subcircuits)
if $\|C\| > c_0$ use ALG to bipartition C into subpartitions $C1$ and $C2$ record size and number of pins for $C1$ and $C2$ Generate-Partition-Tree($C1$) Generate-Partition-Tree($C2$)

Figure 1: Recursive application of a bipartitioning algorithm, ALG, to yield a partitioning tree that is restricted with respect to threshold c_0.

Next, we extract *partitioning instances* from the partitioning tree, such that each instance contains a set

[5] This methodology, though very similar in spirit to that of Landman and Russo, has important differences which allow the Rent parameter calculation to become well-defined. We obtain a sequence of partitioning instances by repeatedly applying a fixed bipartitioning algorithm to split a subcircuit into two smaller circuits, while Landman and Russo obtain their sequence of partitioning instances by ad hoc constraints on the number of subcircuits and the maximum number of terminals per subcircuit. Also, it should be noted that our Rent parameter computation will allow more complete use of all partitioning instances, and their associated $(\bar{C}_i, \bar{T}_i)$ data points, than does the method of Landman and Russo. This is because Landman and Russo impose the further requirement that the number of pins per subcircuit decrease monotonically as the number of subcircuits in the partitioning instance increases. This heuristic requirement forces partitioning instances that "buck the trend" to be discarded, resulting in a Rent parameter computation that is artificially "clean".

[6] For convenience, we say that C is a subcircuit of itself, so that the entire circuit C is at the root of the partitioning tree.

of disjoint subcircuits of roughly equal size.[7] Formally, given a complete partitioning tree we define the i^{th} partitioning instance, P_i, to be a set of i disjoint subcircuits containing all modules of the circuit, subject to the constraint that the maximum size of any subcircuit is minimum. Partition instance P_{i+1} with $i+1$ subcircuits is then constructed by replacing the largest subcircuit in P_i by its two children in the partitioning tree. It is easy to see that all partitioning instances $P_2, \ldots, P_m$ can be derived by traversing the partitioning tree as specified in Figure 2 (m will depend on the parameter c_0 used in generating the partitioning tree).

Generate-Partitioning-Instances()
Input: A partitioning tree for circuit C
Output: Sequence of partitioning instances P_i

let C = complete circuit
let $P = C$
while C is an internal node of the partitioning tree
    $P = P - C$
    $P = P \cup$ subcircuits of C in partitioning tree
    Output P
    C = largest element of P

Figure 2: Traversal of the partitioning tree to output partitioning instances $P_2, \ldots, P_m$.

Finally, given that Generate-Partitioning-Instances has output m partitioning instances $P_2, \ldots, P_m$, we apply the methodology described in [28] to divide these instances into two regions: Region I, where Rent's rule applies, and Region II (corresponding to the few top-most levels of the partitioning tree), where the rule breaks down. Formally, Region I is comprised of the partitioning instances $\{P_t, \ldots, P_m\}$, with $2 \leq t < m$, where t is the minimum index such that the straight-line fit to data points on the log-log plot of $(\bar{C}_t, \bar{T}_t), (\bar{C}_{t+1}, \bar{T}_{t+1}), \ldots, (\bar{C}_m, \bar{T}_m)$ has error $\leq 10\%$ for each of the $m-t+1$ data points. Here, we compute each average using geometric averaging, e.g., $\bar{C}_m$ is the geometric average of the sizes of all the subcircuits of P_m.[8] The Rent parameter is the slope of this straight-line fit (see Figure 3).

The Region II partitioning instances can be viewed as stemming from the initial partitioning steps needed to "explode" the original circuit into a state from which the intrinsic behavior of the individual partitioning algorithm will manifest itself. Again, let each partitioning instance P_i correspond to a data point $(\bar{C}_i, \bar{T}_i)$. Assuming $c_0 = 1$ in algorithm Generate-Partitioning-Tree, all partitioning algorithms will yield partitioning

[7] Intuitively, each partitioning instance can be viewed as a "bucket" that contains a subset of the subcircuits generated during the recursive application of the partitioning algorithm. As in the work of Landman and Russo, computing the "average" size and the "average" number of pins in the subcircuits of each partitioning instance will provide the data points that yield the Rent parameter via linear regression on a log-log plot.

[8] We slightly abuse notation in the sense that C_m refers to the average size of all subcircuits in a partitioning instance, rather than to an individual subcircuit; cf. the notation of Figure 1.

Extract-Rent-Parameter()
Input: Partitioning instances $P_2, \ldots, P_m$ corresponding to data points $(\bar{C}_2, \bar{T}_2), \ldots, (\bar{C}_t, \bar{T}_t)$ (using geometric averaging)
Output: Rent parameter p

let $Data$ = set of all data points (log-log scale)
let Not_Yet = True
while Not_Yet
    compute linear regression over $Data$
    if all data points are within 10% of straight-line fit
        Output slope of this straight-line fit
        Not_Yet = False
    else $Data$ = $Data$ - {lowest-index data point}

Figure 3: Extraction of Rent parameter (slope of straight-line fit to data points in "Region I" of Landman and Russo) from partitioning instances $P_2, \ldots, P_m$.

trees that start at the same point $(\bar{C}_1, \bar{T}_1)$ (the complete circuit), and end at the same point $(\bar{C}_n, \bar{T}_n)$ (where $\bar{C}_n = 1$ and $\bar{T}_n = k$ in Rent's rule, i.e., the average number of external pins per module). Therefore, in order for the algorithm-dependent Rent parameters to vary, there must be a stage (Region II) early in the partitioning process where the ratio $\log(\bar{C}_i)/\log(\bar{T}_i)$ has non-linear behavior that is unique to the partitioning algorithm. This initial behavior causes the starting point of Region I to vary with each partitioning algorithm, and consequently the Region I data points of each partitioning algorithm will approach $(\bar{C}_n, \bar{T}_n)$ with distinct slope (i.e., Rent parameter) on the log-log plot. This phenomenon is clearly visible in the plots of Figure 4 (see Section 4 below).

The above procedural specifications imply that the Rent parameter is well-defined for any partitioning tree. Thus, the notion of a minimum Rent parameter over all partitioning trees of a given circuit also becomes well-defined. With this in mind, we now make the following definitions.

Definition: Given a circuit C and a prescribed bipartitioning algorithm, ALG, the *algorithm-dependent Rent parameter* is obtained by (i) executing Generate-Partitioning-Tree using ALG, C, and the threshold parameter $c_0 = 1$; then (ii) executing Generate-Partitioning-Instances; and then (iii) executing Extract-Rent-Parameter.

Definition: Given a circuit C, the *intrinsic Rent parameter* p^* of C is the minimum value obtainable by (i) executing Generate-Partitioning-Instances on any partitioning tree of C, and then (ii) executing Extract-Rent-Parameter.

The remainder of this paper examines various bipartitioning algorithms to see which yield algorithm-dependent Rent parameters that are closest to the intrinsic Rent parameters. Before presenting our experimental results, we briefly review a taxonomy of those partitioning algorithms which are of interest in the dual contexts of hierarchical layout and rapid area estimation.

3 A Partitioning Taxonomy

A general model for VLSI layout associates a hypergraph $H = (V, E')$ with the circuit netlist; vertices in V represent modules and edges in E' represent signal nets. Several standard transformations [29] may be used to convert H into a graph $G = (V, E)$ with vertices and edges of G weighted to reflect module areas and the multiplicity or importance of connections. Two widely used partitioning formulations are:

- **Minimum-Cut Bisection:** Given $G = (V, E)$, find the partition of V into disjoint U and W, with $|U| = |W|$, such that $e(U, W)$, i.e., the number of edges in $\{(u, w) \in E \mid u \in U \text{ and } w \in W\}$, is minimized.

- **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.

Because minimum-cut bisection divides module area evenly, it is a popular objective within the hierarchical layout paradigm. However, the area bisection requirement is unnecessarily restrictive and can preclude finding natural structure within the circuit. Various ad hoc thresholds and penalty functions (e.g., the r-bipartition formulation of Fiduccia and Mattheyses [10]) have not been completely successful in relaxing this constraint. With this in mind, the ratio cut formulation [42] provides a more straightforward tradeoff between nets cut and evenness in the partition: the numerator captures the minimum-cut criterion while the denominator favors near-bisection. It is well known that both minimum-cut bisection and minimum ratio cut are NP-complete [13], so heuristic methods must be used. Previous approaches fall into several classes, as surveyed in [6] [15] [29].[9]

The greedy iterative paradigm is popular either as a stand-alone strategy or as a postprocessing refinement to other methods. Iterative methods are based on local perturbation of the current solution and typically entail variations of the Kernighan-Lin method [20] [37], e.g., the algorithmic speedups of Fiduccia and

[9] The reader will note that our discussion omits several popular approaches, including simulated annealing [22] [38], genetic algorithms [24], and relaxation-based two-dimensional placement/partitioning (e.g., Gordian [23]). These methods are generally too CPU-intensive to be used as the basis of area estimation and thus lie beyond the scope of our discussion.

Mattheyses [10] and Krishnamurthy [25]. Practical implementations will use a number of random starting configurations and return the best result [29] [42] in order to adequately search the solution space and give predictable performance, or "stability". For example, Wei and Cheng [42] propose a ratio cut heuristic that adapts the shifting and group swapping methods of [10] and returns the best of 20 runs. It should be noted that current wireability analysis algorithms rely almost exclusively on Fiduccia-Mattheyses partitioning. This is due to such reasons as: (i) the long history of Kernighan-Lin k-opt methods in physical layout, (ii) the bisection requirement, which is naturally suited to hierarchical decomposition, and (iii) the simplicity of the algorithm implementation. However, while iterative Fiduccia-Mattheyses style optimization is very fast for single runs and has been a very popular approach, certain limitations loom as problem sizes grow large. These include theoretical results on the weakness of local-search methods, as well as instability and the lack of error bounds; all of these factors seem endemic to the *local* strategy.

With respect to the present work, an important class of partitioning algorithms consists of efficient "spectral" methods which use eigenvalues or eigenvectors of matrices derived from the netlist graph. Recall that the circuit netlist may be represented by the simple undirected graph $G = (V, E)$ with $|V| = n$ vertices $v_1, \ldots, v_n$. Often, we use the $n \times n$ *adjacency matrix* $A = A(G)$, where $A_{ij} = 1$ if $\{v_i, v_j\} \in E$ and $A_{ij} = 0$ otherwise. If G has weighted edges, then A_{ij} is equal to the weight of $\{v_i, v_j\} \in E$, and by convention $A_{ii} = 0$ for all $i = 1, \ldots, n$. If we let $d(v_i)$ denote the degree of node v_i (i.e., the sum of the weights of all edges incident to v_i), 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] [6] [7]. More recent eigenvector and eigenvalue methods have dealt with both module placement (Frankle and Karp [11] and Tsay and Kuh [41]) and graph min-cut bisection (Boppana [3] and Blanks [2]). In general, these previous works formulate the partitioning problem as the assignment or placement of nodes into bounded-size clusters or chip locations. The problem is then transformed into a quadratic optimization, and a Lagrangian formulation leads to an eigenvector computation.[10] In [15], Hagen and Kahng established a close relationship between the optimal ratio cut cost and the second smallest eigenvalue of the matrix

[10] A prototypical example is the work of Hall [18], which we now outline. This work is particularly relevant since it uses eigenvectors of the same graph-derived matrix $Q = D - A$ (the same D and A defined above) that we utilize. Donath and Hoffman, Boppana, and others use different matrices derived from the netlist graph, but exploit similar mathematical properties (e.g., symmetry, positive-definiteness) to derive alternate eigenvalue formulations and relationships to partitioning. Hall's result [18] was that the eigenvectors of the matrix $Q = D - A$ solve the *quadratic placement* problem of finding the vector $x = (x_1, x_2, \ldots, x_n)$ which minimizes

$$z = \frac{1}{2} \sum_{i=1}^{n} \sum_{j=1}^{n} (x_i - x_j)^2 A_{ij}$$

$Q = D - A$, where D and A are as defined above:

Theorem 1 (Hagen-Kahng): 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 $c \geq \frac{\lambda}{n}$. □

This result suggests that the eigenvector x corresponding to λ, i.e., the solution of the matrix equation $Qx = \lambda x$, be used to guide the partitioning. For ratio cut partitioning, [15] used x to induce a linear ordering of the modules, and the best "split" in terms of ratio cut cost was returned. To be more specific, the n modules x_i of the eigenvector are sorted to yield an ordering $v = v_1, \ldots, v_n$ of the modules which we call the *spectral ordering*. The splitting index r, $1 \leq r \leq n-1$, is then found which yields the best ratio cut cost when modules with index $> r$ were placed in U and modules with index $\leq r$ were placed in W. This straightforward construction achieved very significant improvements over previous iterative methods, and also exhibited several desirable traits, including speed, provability, and stability. The spectral method is appealing for its use of global rather than local information, and it moreover provides an *inherently spatial* embedding of the circuit graph. Because the spectral approach significantly outperforms iterative Fiduccia-Mattheyses style methods [15], Section 4 uses the spectral ordering as the basis for ratio cut partitioning variants.[11]

4 Experimental Results and Discussion

4.1 Experimental Procedure

Within the preceding taxonomy, the main classes of partitioning algorithms of interest for top-down layout generation and area estimation are greedy iterative methods (in particular, Fiduccia-Mattheyses minimum-

subject to the constraint $|x| = (x^T x)^{1/2} = 1$, with A_{ij} again equal to the strength of the connection between modules i and j. It can be shown that $z = x^T Q x$, so that to minimize z we may form the Lagrangian

$$L = x^T Q x - \lambda(x^T x - 1).$$

Taking the first partial derivative of L with respect to x and setting it equal to zero yields

$$2Qx - 2\lambda x = 0,$$

and this can rewritten as

$$(Q - \lambda I)x = 0$$

where I is the identity matrix. This is readily recognizable as an eigenvalue formulation for λ, and the eigenvectors of Q are the only nontrivial solutions for x. The minimum eigenvalue 0 gives the uninteresting solution $x = (1/\sqrt{n}, 1/\sqrt{n}, \ldots, 1/\sqrt{n})$, and hence the eigenvector corresponding to the second smallest eigenvalue λ is used.

[11] While eigenvalue computations are not cheap, the run-times reported in [15] were actually less than those for the multiple F-M computations needed by, e.g., the RCut1.0 program of Wei and Cheng [43]. Significant algorithmic speedups stem from the need to calculate only a *single* (the second smallest) eigenvalue of a *symmetric* matrix. Moreover, netlist graphs tend to be very sparse due to hierarchical circuit organization and degree bounds imposed by the technology fanout limits; this allows application of sparse numerical techniques such as the block Lanczos algorithm [14].

cut bisection) and methods based on a spectral ordering. Within these classes, we parameterize the algorithms of interest by (i) their objective function: minimum net cut metric or ratio cut metric; and (ii) the balance restrictions in the output bipartition: exact bisection, 1/4 - 3/4 range limited, or no limits on range. The following is a list of the algorithms used in our experiments:

1. SpecRC-Full: spectral ordering; partition the ordering at best splitting point according to ratio cut metric.
2. SpecRC-1/4: spectral ordering; partition the ordering at best splitting point according to ratio cut metric, subject to constraint that maximum partition size is $\leq \frac{3}{4} \cdot |V|$.
3. Spec-Bis: spectral ordering; bisect the ordering. In other words, this is simply "spectral bisection".
4. FM-1/4: Fiduccia-Mattheyses implementation of the Kernighan-Lin method for minimum net cut partitioning; range limitation ("r" in [10]) such that maximum partition size is $\leq \frac{3}{4} \cdot |V|$.
5. FM-Bis: "standard" Fiduccia-Mattheyses method, i.e., for minimum net cut bisection.[12]

To generate a control for these partitioning algorithms (and, separately, to confirm the work of Sutherland and Ostreicher [34] [40], who posit a Rent parameter of 1 for random partitioning methods), we also tested the following three variants of random circuit partitioning:

6. RandMC-1/4: random linear ordering; partition this linear ordering at best splitting point according to minimum net cut metric, subject to constraint that maximum partition size is $\leq \frac{3}{4} \cdot |V|$.
7. RandRC-1/4: random linear ordering; partition this linear ordering at best splitting point according to ratio cut metric, subject to constraint that maximum partition size is $\leq \frac{3}{4} \cdot |V|$.
8. Rand-Bis: random linear ordering; bisect this linear ordering.

All eight partitioning strategies were tested on the four circuits Primary1, Primary2, Biomed and Struct from the MCNC benchmark suite. We also used two highly structured inputs as additional benchmarks: Mesh2D, a 2-dimensional mesh topology, and Mesh3D, a three-dimensional mesh topology. Since Primary1, Primary2 and Biomed are circuits with unknown structure, little can be deduced about their intrinsic Rent parameter. On the other hand, Struct is an array multiplier with a mesh-like topology, meaning that its

[12] The computation time required for spectral-based partitioning is slightly more than that for FM-based partitioning [15]. Thus, for each of FM-1/4 and FM-Bis, we compensate for this CPU difference by reporting the minimum Rent parameter over 10 independently generated partitioning trees.

intrinsic Rent parameter is likely to be 1/2. Additionally, since Mesh2D and Mesh3D have very regular topologies, their respective intrinsic Rent parameters are known to be *exactly* 1/2 and 2/3; we therefore use these latter three benchmarks to assess the near-optimality of the partitioning trees produced by various bipartitioning algorithms. Table 1 summarizes the characteristics of the benchmark circuits.

Benchmark	cells	nets	pins	pads	cell library	technology	p^*
Primary1	752	904	2941	81	sclib	SCell 2-metal	?
Primary2	2907	3029	11226	107	sclib	SCell 2-metal	?
Struct	1888	1920	5471	64	db	MOSIS SCMOS	≈ 0.5
Biomed	6417	5766	22253	97	db	MOSIS SCMOS	?
Mesh2D	2000	4090	8000	180	test	test	0.5
Mesh3D	1000	3300	6000	600	test	test	0.666

Table 1: Characteristics of benchmark examples.

Our experimental procedure was as follows.

- For each of the six input netlists (four industry benchmark circuits and two synthetic mesh topologies), each of the eight partitioning algorithms was used to construct a partitioning tree by applying Generate-Partitioning-Tree with $c_0 = 9$ (i.e., subcircuits of size 9 or less were not partitioned further). After every application of the partitioning algorithm, the size and external pin count of the generated subcircuits was recorded, as specified in the Generate-Partitioning-Tree template.
- To each of the resulting partitioning trees, we applied the Generate-Partitioning-Instances algorithm. For, e.g., a partitioning tree with m subcircuits at the leaves, this yielded $m-1$ partitioning instances $P_2, P_3, \ldots, P_m$ corresponding to i-way circuit partitions for $i = 2, 3, \ldots, m$.
- Finally, we applied the Extract-Rent-Parameter algorithm to each set of partitioning instances. For, e.g., a set of $m-1$ partitioning instances P_i, this entailed plotting the corresponding $(\bar{C}_i, \bar{T}_i)$ data points and then finding the maximum t such that the linear regression to $P_t, P_{t+1}, \ldots, P_m$ had at most 10% error to any of these $m-t+1$ data points. We then returned the algorithm-dependent Rent parameter given by the slope of this straight-line fit.

4.2 Results and Discussion

Our main experimental results are summarized in Table 2, which gives the algorithm-dependent Rent parameters for all input netlists and all bipartitioning algorithms. Since all the Rand variants yielded essentially identical Rent parameters, we report only the Rand-Bis results as representative of this category.

The results indicate that our experimental methodology effectively distinguishes the relative merits of the various bipartitioning algorithms (e.g., see Figure 4, which depicts the Rent parameter fits for the Struct benchmark). In particular, the spectral ratio cut variants SpecRC-1/4 and SpecRC-Full generate partitioning trees with uniformly superior Rent parameters for essentially all test cases. Moreover, for the Struct, Mesh2D and Mesh3D inputs, where we know the value of the intrinsic Rent parameter, p^*, we see that SpecRC-1/4 and SpecRC-Full yield partitioning trees with Rent parameters very close to the optimal values of 1/2 for Struct and Mesh2d, and 2/3 for Mesh3d.[13] Also note that our experimental results are consistent with the conjecture of Sutherland and Ostreicher [40] that random partitioning trees for all circuits will have Rent parameters very close to one.

Benchmark	Partitioning Strategy					
Circuit	Rand-Bis	FM-Bis	FM-1/4	Spec-Bis	SpecRC-1/4	SpecRC-Full
Primary1	.916 (2)	.723 (2)	.670 (2)	.647 (4)	.580 (5)	.488 (7)
Primary2	.951 (2)	.933 (1)	.772 (2)	.746 (6)	.674 (7)	.598 (7)
Struct	.935 (2)	.878 (2)	.726 (3)	.635 (5)	.581 (4)	.572 (3)
Biomed	.920 (4)	.925 (1)	.671 (4)	.779 (0)	.688 (0)	.599 (4)
Mesh2d	.970 (2)	.663 (2)	.563 (1)	.514 (0)	.540 (2)	.524 (1)
Mesh3d	.953 (2)	.711 (1)	.720 (1)	.620 (0)	.654 (0)	.660 (0)

Table 2: Rent parameter results. Numbers in parentheses indicate the number of data points belonging to Region II.

Occasional entries in the table are derived from regressions using slightly fewer data points in Region I than other entries, which would imply a correspondingly lesser statistical significance. However, Table 2 also indicates the number of data points assigned to Region II by the Extract-Rent-Parameter algorithm. The fact that so few data points are discarded indicates that the straight-line fit is very strong. (For example, the total numbers of partitioning instances for Spec-Bis (i.e., numbers of leaves in the corresponding partitioning trees) with $c_0 = 9$ include 1023 instances for Biomed, 255 instances for Struct, etc. Indeed, if we use $c_0 = 1$ the number of partitioning instances becomes equal to the number of modules, while the number of Region II data points remains the same.) Our conclusions are further strengthened when we take into consideration the *quality* of the fit between our Region I data points and the regressed line. Figure 4 shows graphs depicting the Region I data points and their regressed lines for six of the partitioning algorithms, using the Struct benchmark circuit. The goodness of fit was measured using a correlation coefficient [32]. For the three SpecRC algorithms, virtually every example had correlation coefficient higher than .98, indicating a

[13] In some cases (e.g., Mesh3D using Spec-Bis or SpecRC-1/4), we observe that the value of p output by Extract-Rent-Parameter is actually smaller than the theoretical minimum (e.g., 2/3 for Mesh3D). However, this discrepancy is acceptable since the estimation procedure has a built-in maximum error of 10%.

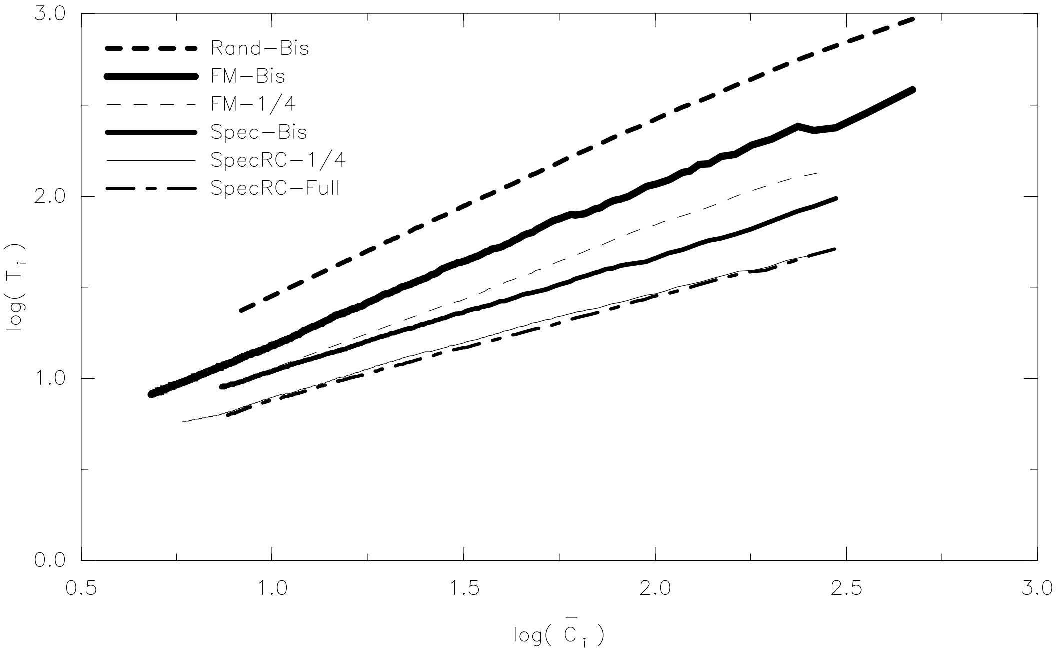

Figure 4: Rent's rule fit for each partitioning tree of the Struct test circuit.

consistently good fit. (The leftmost data points in each regression may vary due to edge effects with respect to application of the threshold c_0, as well as characteristics of the partitioning algorithm. For example, our SpecRC-Full partitioning tree for Primary2 was rather unbalanced, resulting in a large number of leaf subcircuits that were smaller than the threshold $c_0 = 9$.)

Our experimental procedure reflects the recipe, or "black box", given in Section 2.2 for computing the algorithm-dependent Rent parameter of a given circuit C with respect to a given bipartitioning algorithm ALG. We reiterate that not only does our recipe make the (algorithm-dependent) Rent parameter and the intrinsic Rent parameter well-defined, but it also retains the spirit of the original Rent parameter experiment performed by Landman and Russo. Furthermore, while our two algorithms Generate-Partitioning-Instances and Extract-Rent-Parameter might seem to embody several "arbitrary" decisions, we emphasize that Rent's rule refers to a power-law scaling phenomenon which will be "visible" to any of a number of similar methods. Indeed, our research has shown that varying the basic Rent parameter evaluation methodology has such a small effect on our qualitative results that it might be likened to measuring in "inches" rather than "meters". We conclude this section with a brief digression, giving the rationale behind two defining aspects of our Generate-Partitioning-Instances and Extract-Rent-Parameter algorithms.

- First, the reader may note that any number of methods could be used to generate partitioning instances from a given partitioning tree. For example, the standard approach in the literature [28] [34] has been to generate partitioning instances using a breadth-first traversal of the partitioning tree, i.e., to group all the subcircuits at distance one from the root into a partitioning instance, then all of the subcircuits at distance two into another partitioning instance, etc. We make three observations. (1) This standard approach is typically applied to the (perfectly balanced) partitioning trees that are generated by FM-Bis, which is the usual bipartitioning algorithm for top-down layout. In such a context, our methodology in Generate-Partitioning-Instances is consistent with previous work: in fact, we generate a superset of the partitioning instances output by the standard approach. (2) When the partitioning tree is unbalanced, as can occur with the SpecRC-Full, SpecRC-1/4 and FM-1/4 algorithms, the simple breadth-first traversal will output partitioning instances that do not contain all of the modules of the circuit; this contradicts the methodology established by Landman and Russo. On the other hand, Generate-Partitioning-Instances will always output partitioning instances that include all modules of the circuit. (3) When the bipartitioning algorithm can output an uneven partition, the standard breadth-first traversal can output partitioning instances that are highly unbalanced, with a very large size differential between the smallest and largest subcircuits. In contrast, our approach is consistent with Landman and Russo's requirement that the subcircuit sizes be as uniform as possible, e.g., for 1/4 - 3/4 range-limited bipartitioning, the size ratio between the largest and smallest subcircuits in any partitioning instance will be at most four.

- Second, the reader might note that geometric averaging was specifically prescribed for obtaining $\bar{C}_i$ and $\bar{T}_i$ in the Extract-Rent-Parameter algorithm. Here, we make two observations. (1) Specifying the averaging method allowed the Rent parameter computation to be well-defined. (2) The more compelling observation is that in retrospect, the choice of geometric averaging did not qualitatively affect our results, i.e., the choice of averaging method is akin to "inches" versus "meters". Table 3 shows that no matter how $\bar{C}_i$ and $\bar{T}_i$ are respectively averaged (geometric-geometric, arithmetic-geometric, etc.), essentially identical results will be obtained.

- Finally, the reader may notice that there are several instances of "built-in imprecision" in our reported results, namely (i) the use of the threshold $c_0 = 9$ in our application of Generate-Partitioning-Tree, (ii) the averaging of subcircuit sizes and pin counts for each partitioning instance (rather than the use of individual subcircuit data) in Extract-Rent-Parameter, and (iii) the incorporation of a 10% tolerance in the regression fit of Extract-Rent-Parameter. With regard to (i), it should be noted that we employ this threshold in order to avoid plotting (for the larger benchmarks) thousands of data points that are all essentially on top of one another. Due to the effects of averaging and the very large number of subcircuits at this juncture in the generation of partitioning instances, our results are basically identical to those that would be obtained with $c_0 = 1$, i.e., Table 2 indeed reports the true algorithm-dependent Rent parameter. With respect to (ii) and (iii), the fact that the Rent parameter is an *average* scaling relationship for the entire circuit implies that some tolerance (iii) in the straight-line fit is necessary. Also, our work closely follows the original methodology of Landman and Russo in treating subcircuits as they are grouped within whole partitioning instances (rather than outside of the partitioning instance "context"); this avoids undue sensitivity to local variation in the circuit structure.

Averaging Scheme	Partitioning Strategy					
	Rand-Bis	FM-Bis	FM-1/4	Spec-Bis	SpecRC-1/4	SpecRC-Full
g-g	0.920	0.925	0.671	0.779	0.688	0.599
a-a	0.920	0.947	0.658	0.800	0.674	0.608
a-g	0.919	0.929	0.678	0.777	0.674	0.603
g-a	0.925	0.926	0.653	0.803	0.685	0.618

Table 3: Rent parameter results for test circuit Biomed using four different averaging techniques. Here, g = geometric averaging, a = arithmetic averaging and, e.g., g-a implies that the $\bar{C}_i$ are geometrically averaged while the $\bar{T}_i$ are arithmetically averaged. Notice that the results will vary even for bisection because of the varying T_i values in the subcircuits of each partitioning instance.

5 Extensions and Conclusions

The experimental procedure used above relies on an entire partitioning tree to derive the algorithm-dependent Rent parameter. However, constructing an entire partitioning tree is potentially quite time-consuming. Therefore, important extensions of our current work will involve speedup of the Rent parameter computation, and more specifically, the SpecRC-based Rent parameter computation since the SpecRC-based partitioning trees are superior to those produced by other partitioning algorithms. Such speedups will greatly enhance the utility of our approach for such time-critical applications as early wireability analysis or area estimation.[14] With this in mind, we now list three research directions (two of which are specific to the SpecRC class of algorithms) that offer potential speedups to our methodology.

[14] We note that in practice, a single execution of spectral ratio-cut partitioning is significantly faster than 10 executions of the Fiduccia-Mattheyses algorithm, e.g., [16] cites 83 sec of Sun Sparc-1 CPU time for the eigenvector computation in the top-level bipartitioning of Primary1, while 10 FM executions required 204 sec of CPU time. However, the matrix computations used by spectral partitioning algorithms are *potentially* expensive, and we are therefore interested in minimizing the number of such computations.

Incomplete Partitioning Trees. The very strong fit of our data points to the power-law in Rent's rule suggests the following method for estimating the Rent parameter: restrict Generate-Partitioning-Tree to only a small number of applications of the bipartitioning algorithm, then apply the linear regression analysis to the partitioning instances produced from this restricted partitioning tree.[15] We have observed that as successive Region I data points for the regression are obtained by the procedure Generate-Partitioning-Instances, each additional data point results in essentially monotonic convergence toward the Rent parameter of the complete partitioning tree. Moreover, for all input netlists a significant reduction in computation time can indeed be achieved with little corresponding loss of accuracy in the Rent parameter value, simply by restricting the number of partitioning instances (or, equivalently, restricting the size of the partitioning tree). For example, with the Mesh2D benchmark we found that the SpecRC-Bis, SpecRC-1/4, and SpecRC-Full algorithms respectively required 40, 49, and 29 partitioning instances in order to achieve 5% accuracy in the estimate of the Rent parameter. This is in contrast to 2000 partitioning instances if we process the partitioning tree for each algorithm for $c_0 = 1$, or between two and three hundred partitioning instances for these algorithms with $c_0 = 9$. To achieve 10% accuracy, the same three algorithms respectively required only 9, 35 and 16 partitioning instances. While we do not know the Region I - Region II demarcation a priori, in our experience the number of data points in Region II has always been very small (i.e., always less than 10). Thus, we believe that accurate estimates of the (SpecRC) Rent parameter may be obtained after generating only a small portion of the partitioning tree. As an aside, it is also possible that Monte Carlo methods which examine random root-leaf paths in the partitioning tree may similarly be applied to reduce the number of bipartitions computed.

Reducing the Number of Eigenvector Computations. Our second speedup is based on the observation in [15] that the sorted second eigenvector used by the SpecRC partitioning algorithm provides a "uniformly good" ordering of the netlist modules. In other words, after we bipartition a circuit using the sorted second eigenvector of its Laplacian, there is no compelling reason to independently reorder the two resulting subcircuits. Thus, we propose to "recycle" portions of the original linear ordering, and recursively find optimal splitting points in these portions of the ordering in order to construct the partitioning tree. In general, this will not return the same Rent parameter as would be obtained by recomputing eigenvectors for each subcircuit; however, our preliminary experimental results are very encouraging in that this "single-eigenvector" yields partitioning trees with very strong Rent parameter fits. Thus, our current work examines methods which can compensate for the inaccuracy that stems from using only a single eigenvector.

[15] Strictly speaking, Generate-Partitioning-Tree and Generate-Partitioning-Instances are merged under this scenario: the latter algorithm drives the former by selecting the largest existing subcircuit to be the next input to the bipartitioning algorithm.

Direct Use of Netlist Spectra. In view of the superior performance of the SpecRC algorithms, as well as the very good match between the Spec-RC based Rent parameter and the known intrinsic Rent parameter for the Struct, Mesh2D and Mesh3D examples, we conjecture that the spectral ratio cut algorithm is in some sense an "intrinsically good" top-down partitioning strategy. It has been established [15] that the eigenvector-based heuristic ratio cuts $(U|W)$ will in practice have cost very close to the theoretical lower bound of $\frac{\lambda}{n} \leq \frac{e(U,W)}{|U|\cdot|W|}$ provided in Theorem 1 (recall the discussion of Section 3). This suggests that the SpecRC algorithm-dependent Rent parameter is closely tied to the heuristic ratio cut cost, which in turn is closely tied via Theorem 1 to the second-smallest eigenvalue λ of the netlist Laplacian. It is therefore tempting to speculate that a relationship exists between (i) the second-smallest eigenvalues λ encountered as SpecRC is used in Generate-Partition-Tree, and (ii) the intrinsic Rent parameter of the circuit. Preliminary experimental results and heuristic justifications presented in [17] to support such a proposed relationship were at best inconclusive. However, we believe that exploration of a possible "lambda-Rent" relationship is one of the more far-reaching open questions that stem from the present work. Certainly, such a relationship, if established, will offer speedups along the lines of those above, e.g., by estimating p^* from the λ values of only a few eigenvector-based partitioning applications, or even from the single eigenvalue λ obtained in the top-level ratio-cut bipartition.

In conclusion, we have introduced the notion of the *intrinsic Rent parameter* of a circuit, which is the minimum possible Rent parameter of any partitioning tree of the circuit. In making this notion well-defined, we have formulated the Rent parameter computation in terms of three phases: (i) Generate-Partitioning-Tree, (which takes as input a bipartitioning algorithm and a circuit netlist, along with an optional lower bound c_0 on the size of partitioned subcircuits), (ii) Generate-Partitioning-Instances, and (iii) Extract-Rent-Parameter. Our three phases closely follow the spirit of Landman and Russo's original Rent parameter experiments which have guided workers in the field for two decades ago, and we observe that our choices in formulating these phases do not qualitatively change our results or their interpretation.

Motivated by observed relationships between low Rent parameter of a partitioning tree and small total wire length in the corresponding top-down hierarchical layout, we have assessed the utility of existing bipartitioning algorithms by comparing their induced *algorithm-dependent Rent parameters* on various benchmark circuits. Our empirical results indicate that spectral ratio-cut partitioning algorithms generate partitioning trees with lower Rent parameter than those generated by any other algorithms tested. Moreover, these Rent parameter values are essentially optimal for those examples where the intrinsic Rent parameter is known. This result suggests that top-down layout techniques based on spectral ratio cut partitioning will achieve denser layouts than current methods based on Fiduccia-Mattheyses partitioning. Hence, our ongoing work addresses the integration of spectra-based ratio cut partitioning into a top-down layout synthesis package.

Finally, our results imply that tools for layout area estimation and early wireability analysis should also be based on spectral ratio cut partitioning, since this will achieve closer estimates of the *required* resources for a given design. Such applications of the Rent parameter are in general time-critical, and we have offered three extensions which lead to computation of the SpecRC algorithm-dependent Rent parameter using far fewer eigenvector computations than our current methodology.

References

[1] E. R. Barnes. An algorithm for partitioning the nodes of a graph. *SIAM J. Alg. Disc. Meth.*, 3(4):541–550, 1982.

[2] J. Blanks. Partitioning by probability condensation. In *Proc. of the 26th Design Automation Conf.*, pages 758–761, 1989.

[3] R. B. Boppana. Eigenvalues and graph bisection: An average case analysis. In *IEEE Symp. on Foundations of Computer Science*, pages 280–285, 1987.

[4] W. Donath. Hierarchical structure of computers. Technical Report RC 2392, IBM T. J. Watson Res. Cen. Yorktown Heights,N.Y., 1969.

[5] W. E. Donath. Placement and average interconnection lengths of computer logic. *IEEE Transactions on Circuits and Systems*, CAS-26(4):272–277, 1979.

[6] W. E. Donath. Logic partitioning. In B. Preas and M. Lorenzetti, editors, *Physical Design Automation of VLSI Systems*, pages 65–86. Benjamin Cummings, 1988.

[7] W. E. Donath and A. J. Hoffman. Lower bounds for the partitioning of graphs. *IBM J. Res. Dev.*, pages 420–425, 1973.

[8] W. R. Heller et al. Prediction of wiring space requirements for LSI. In *Proceedings of the 14th DA Conference*, pages 20–22. IEEE/ACM, 1977.

[9] M. Feuer. Connectivity of random logic. *IEEE Transactions on computers*, C-31(1):29–33, January 1982.

[10] C. M. Fiduccia and R. M. Mattheyses. A linear-time heuristic for improving network partitions. In *Proc. of the 19th Design Automation Conference*, pages 175–181. IEEE/ACM, 1982.

[11] J. Frankle and R. M. Karp. Circuit placement and cost bounds by eigenvector decomposition. In *Proc. ICCAD-82*, pages 414–417, 1982.

[12] A. A. El Gamal. Two-dimensional stochastic model for interconnections in master slice integrated circuits. *IEEE Transactions on Circuits and Systems*, CAS-28(2):127–138, February 1981.

[13] M. Garey and D. D. Johnson. *Computers and Intractability: A Guide to the Theory of NP-Completeness.* 1979.

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

[15] L. Hagen and A. B. Kahng. Fast spectral methods for ratio cut partitioning and clustering. In *Proc. ICCAD-91*, pages 10–13, 1991.

[16] L. Hagen and A. B. Kahng. New spectral methods for ratio cut partitioning and clustering. *IEEE Trans. CAD*, CAD-11(9):1074–1085, September 1992.

[17] L. Hagen, A. B. Kahng, F. J. Kurdahi, and C. Ramachandran. On the intrinsic rent parameter and spectra-based partitioning methodologies. In *Proc. Euro-DAC 92*, September 1992.

[18] K. M. Hall. An r-dimensional quadratic placement algorithm. *Management Science*, 17:219–229, 1970.

[19] T. Hamada, C. K. Cheng, and P. M. Chau. A new wire length estimation technique utilizing neighborhood density equations. In *Proc. ACM/IEEE 28th Design Automation Conf.*, pages 57–61, June 1992.

[20] B. W. Kernighan and S. Lin. An efficient heuristic for partitioning graphs. *Bell Syst. Tech. Jour.*, 49(2):291–307, 1970.

[21] R. W. Keyes. Fundemental limits in digital information processing. *Proceedings of the IEEE*, 69:267–278, February 1981.

[22] S. Kirkpatrick, C. D. Gelatt Jr., and M. P. Vecchi. Optimization by simulated annealing. *Science*, (220):671–680, 1983.

[23] J. M. Kleinhans, G. Sigl, and F. M. Johannes. Sea-of-gates placement by simultaneous quadratic programming combined with improved partitioning. In *Proc. VLSI89*, 1989.

[24] Ralph M. Kling and Prithviraj Banerjee. Esp: Placement by simulated evolution. *IEEE Trans. Computer-Aided Design*, 8(3):245–256, March 1989.

[25] B. Krishnamurthy. An improved min-cut algorithm for partitioning VLSI networks. *IEEE Trans. on Computers*, C-33(5):438–446, 1984.

[26] F. J. Kurdahi. *Area Estimation of VLSI Circuits*. PhD thesis, Univ. of So. Calif., 1987.

[27] F. J. Kurdahi and A. C. Parker. Techniques for area estimation of VLSI layouts. *IEEE Trans. CAD*, 8(1):81–92, 1989.

[28] B. Landman and R. Russo. On a pin versus block relationship for partition of logic graphs. *IEEE Transactions on Computers*, C-20:1469, 1971.

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

[30] B. B. Mandelbrot. *Fractals: Form, Chance, and Dimension*. W. H. Freeman, San Francisco, 1981.

[31] A. Masaki and M. Yamada. Equations for estimating wire length in various types of 2-d and 3-d system packaging structures. *IEEE Trans. Comp., Hybrids, and Manuf. Tech.*, CHMT-10(2):190–198, 1987.

[32] A. Papoulis. *Probability and Statistics*. Prentice Hall, 1990.

[33] M. Pedram and B. Preas. Interconnection length estimation for optimized standard cell layouts. In *Proc. ICCAD-89*, pages 390–393. IEEE/ACM, 1989.

[34] R. Russo. On the tradeoff between logic performance and circuit-to-pin ratio for lsi. *IEEE Trans. Computers*, C-21(2):147–152, 1972.

[35] S. Sastry. *Wireability Analysis of Integrated Circuits*. PhD thesis, USC, January 1985.

[36] S. Sastry and A.C. Parker. On the relation between wire length distributions and placement of logic on Master Slice ICs. In *Proc. 21st Design Automation Conf.*, 1984.

[37] D. G. Schweikert and B. W. Kernighan. A proper model for the partitioning of electrical circuits. In *Proc. Design Automation Conf.*, 1972.

[38] C. Sechen. *Placement and Global Routing of Integrated Circuits Using Simulated Annealing*. PhD thesis, Dept, of EECS, Univ. of California, Berkeley, 1986.

[39] C. Sechen. Average interconnection length estimation for random and optimized placements. In *Proc. ICCAD-87*, pages 190–193, November 1987.

[40] I. E. Sutherland and D. Oestreicher. How big should a printed circuit board be? *IEEE Transactions on computers*, pages 537–542, May 1973.

[41] R. S. Tsay and E. S. Kuh. A unified approach to partitioning and placement. In *Princeton Conf. on Inf. and Comp.*, 1986.

[42] Y. C. Wei and C. K. Cheng. Towards efficient hierarchical designs by ratio cut partitioning. In *Proc. ICCAD-89*, pages 298–301, 1989.

[43] Y. C. Wei and C. K. Cheng. Ratio cut partitioning for hierarchical designs. *IEEE Trans. on CAD*, 10(7):911–921, July 1991.