Computer Science Department Technical Report
University of California
Los Angeles, CA 90024-1596

A NEW APPROACH TO EFFECTIVE CIRCUIT CLUSTERING

L. Hagen
A. B. Kahng

September 1992
CSD-920041

A New Approach to Effective Circuit Clustering*

Lars Hagen and Andrew B. Kahng

UCLA CS Dept., Los Angeles, CA 90024-1596

Abstract

The complexity of next-generation VLSI systems exceeds the algorithmic capabilities of top-down layout synthesis tools, particularly in netlist partitioning and module placement. Thus, a bottom-up clustering phase is needed to "condense" the netlist so that the problem size becomes tractable to existing optimization methods. Previous ad hoc clustering methods fall into the classes of *global* and *local* methods. The former are too expensive to allow application to large problems, while the latter are too "blind" to yield effective results.

In this paper, we use the notion of *intrinsic circuit structure* [18] to establish the *DS quality measure*, which provides the first general metric for objective evaluation of clustering algorithms. The DS metric is motivated by a theoretical relationship between the difficulty of cutting a natural cluster and the number of distinct paths between two nodes in that cluster. This derivation in turn motivates our RW-ST algorithm, which is a new *self-tuning* clustering method based on random walks in the circuit netlist. The RW-ST method is highly efficient, and can be shown to quickly capture an implicitly global circuit clustering.

We evaluate the RW-ST algorithm in two ways: (i) by the quality of clusters as measured by the DS metric, and (ii) by its "real-world" effect in enhancing the solution quality of the Fiduccia-Mattheyses top-down partitioning approach. Extensive experimental results were conducted using both MCNC benchmark circuits and the class of random inputs defined by [13]. Using either criterion, our results are markedly better than those of all previous methods, including those of Bui et al. [2] [3], Cong et al. [6], and Garbers et al. [13]. Specifically, we have achieved the following results: (i) our methodology finds natural circuit clusters (e.g., optimal solutions to the difficult examples of [13]) without requiring a priori specification of target cluster sizes, number of clusters, or any other parameters; (ii) the quality of our results improves smoothly with the resources available to the random walk, and moreover the clustering operation is *self-tuning* with respect to the length of the walk; (iii) average cluster quality as measured by the DS metric is very high (neither the method of [6] nor the method of [2] [3] give results comparable to the RW-ST results); and (iv) when incorporated within a two-phase iterative Fiduccia-Mattheyses partitioning implementation as suggested by [3], the RW-ST clustering improves bisection width by an average of 17% and is significantly better than the matching-based clusters of [2] [3].

1 Introduction

Top-down approaches are widely used to cope with increasing problem complexity in layout synthesis. Recursive calls to a partitioning algorithm generate a circuit hierarchy which subsequently guides the placement/routing phases of layout. The partitioning process essentially tries to uncover *natural circuit structure*, i.e., a hierarchy of subcircuits which minimizes the connectivity between subcircuits. As surveyed

*This work was supported in part by NSF MIP-9110696, ARO DAAK-70-92-K-0001, and ARO DAAL-03-92-G-0050. A. B. Kahng is also supported by an NSF Young Investigator Award and California MICRO grants from Zycad Corporation and Cadence Design Systems.

by Donath [10] and Lengauer [25], typical partitioning objectives such as minimum-width bisection [29] and minimum ratio cut [35] are NP-complete and require such heuristics as simulated annealing [30], greedy k-opt interchange [23], or quadratic optimization (via relaxation [5] [34] or spectral [16] methods). Our work addresses the impending failure, when faced with million-gate designs, of the partitioning algorithms used in top-down layout approaches. This failure will occur for several reasons:

1. **Scaling and instability.** The solution quality of iterative algorithms (particularly Kernighan-Lin variants and simulated annealing) becomes less predictable and does not scale with problem size [25]; this reflects the so-called "error catastrophe" of local search methods, i.e., that most local optima tend to be of only average quality [22]. To compensate for this instability, practical implementations must use multiple trials with random starting points, e.g., the recent work of Wei and Cheng [35] achieves stability by returning the best of 10 or 20 random runs, and Johnson et al. [20] suggest that 500 is an appropriate number of trials for Kernighan-Lin k-opt bisection of graphs with vertex cardinality $n \approx 10^5$.

2. **Complexity.** Actual infeasibility of current methodologies for large instances becomes a fundamental obstacle. Already, problem instances of several millions of modules (gates) are typical with system-level partitioning onto multiple FPGAs or other ASICs. For this level of complexity, both simulated annealing and relaxation methods for quadratic optimization have infeasible time/space requirements.[1] We note that large-scale system partitioning applications occur in all phases of CAD (partitioning for testability, mapping to hardware emulation and simulation engines, etc.), so that both the infeasibility of current methods and the utility of our proposed approach below will apply in many areas other than physical layout.

3. **Unnatural formulations.** Standard approaches compute a heuristic S-way partitioning for a prescribed value of S. As observed by Wei and Cheng [36], use of an inappropriate value of S will prevent identification of "natural" divisions which might be more readily discovered by bottom-up analysis. However, the proper value of S cannot be known a priori, since this in effect requires prior knowledge of the circuit structure.

Given these difficulties, a bottom-up clustering approach is used to *enable* successful top-down partitioning by condensing the circuit netlist into clusters and reducing problem size. The clustering approach is attractive because it is "safe", i.e., it avoids making the far-reaching decisions that are inherent in a

[1] Studies of typical simulated annealing implementations (e.g., recent Timberwolf releases) show that at least $\Omega(n^{4/3})$ moves must be generated at each temperature in order to achieve the equilibrium condition [30] [33]; this equilibration must occur for hundreds of temperatures. With problem sizes now approaching $n = 10^6$ modules, this algorithmic complexity is impractical. Similar analysis can be performed for the time/space requirements of, e.g., the spectral computations of [16] or such relaxation-based quadratic optimization methods as [5] [32]; these methods must store and manipulate the adjacency matrix of the netlist and are clearly infeasible for very large n even when sparse-matrix techniques are used.

a cluster). While density-based clustering is cited in [13] as a folklore method, it entails checking all module subsets of cardinality c, which is impractical. Hence, the closely related concept of (k,l)-connectivity was recently proposed by Garbers et al. [13] for use in circuit clustering.

A Global Approach: (k,l)-Connectivity. If there are k edge-disjoint paths of length l between modules u and v, then u and v are said to be (k,l)-connected; [13] showed that for certain highly structured classes of random inputs, the transitive closure of the (k,l)-connectedness relation gives an equivalent clustering to that induced by the edge density criterion. Indeed, on some instances of a class of random inputs and for a highly structured standard-cell benchmark, the (k,l)-connectivity criterion yields reasonable solutions. However, the method suffers from several main weaknesses. First, (k,l)-connectivity may yield nonintuitive results: modules v_i and v_j can belong to a cluster even when no module on any path between v_i and v_j belongs to the cluster (e.g., a cycle of length four through modules A, B, C and D will be broken into an (A,C) cluster and a (B,D) cluster by the $(2,2)$-connectivity criterion; this solution has twice the cutsize of the more natural $(A,B)(C,D)$ clustering). Second, the values of k and l which allow extraction of the "correct" clustering must be determined experimentally for each circuit netlist, and this determination is not easy, as shown by the results obtained in [13]. Third, although determining (k,l)-connectivity appears to be a more global and algorithmically tractable criterion than edge density, it actually entails solving an NP-complete problem for $l \geq 5$, with NP-completeness of the case $l = 4$ still an open question [19]. For small values of $l \leq 3$, the global nature of the (k,l)-connectivity algorithm becomes less clear. Again, this is reflected by the results of [13], which we reproduce for comparison in Section 4: the (k,l)-connectivity computation was feasible only for $l = 2$ and could not easily discern the strongly clustered structure of the inputs.

Other Clustering Methods. Three other methods should be noted. The epitaxial growth or "direct" method [10] iteratively adds the most closely connected unclustered module to the current cluster. This method is highly local, and depends on heuristic choices of cluster seeds, the number of clusters, the tie-breaking rules, etc.[4] The global "top-down clustering" method of [36] is essentially equivalent to top-down recursive application of the ratio cut partitioning approach given in [35]. We do not consider it to be a bona-fide clustering algorithm because it assumes heuristic partitioning can be performed on the flat netlist, and our premise is that if such is possible, clustering is not needed. Finally, [6] gave a global method which, like the present work, is based on a random walk in the netlist.

[4] As a top-down partitioning method, seeded epitaxial growth has been discarded in favor of min-cut or other approaches [25]; it is not clear that the method will somehow perform better for large S than small S.

top-down approach.[2] Moreover, while clustering will restrict the partitioning solution space, some work indicates that this actually improves the results of iterative partitioning methods via a two-phase application of Kernighan-Lin optimization [2] [3] [25] (we examine this issue more closely in Section 4). Nevertheless, in practice clustering is avoided because of inherent weaknesses in current bottom-up algorithms, namely, that grouping decisions are based only on local criteria such as the number of connections to modules in an existing cluster. While this locality is needed to maintain reasonable algorithm complexity, it may lead to unfortunate grouping decisions. Thus, top-down partitioning, *while it remains tractable*, remains the preferred method of decomposing a given layout problem. The goal of clustering is then to reduce problem size while deferring far-reaching decisions until *well-considered* top-down optimizations become feasible.

1.1 Previous Work

Previous work in circuit clustering ranges from highly *local* to highly *global* approaches. Generally speaking, local approaches are more efficient but can result in unnatural groupings of modules. On the other hand, global approaches give potentially more useful and "natural" results, but may require prohibitive amounts of computation. For our discussion, two particularly relevant approaches are respectively due to Bui et al. [2] [3] and to Garbers et al. [13].

A Local Approach: Matching-Based Compaction. In [2] [3], Bui et al. proposed a two-phase matching based compaction strategy. With this approach, the modules pairs of a maximal random matching in the netlist graph are used to induce a compacted partitioning instance on $n/2$ vertices which correspond to the matching edges. A heuristic Kernighan-Lin partitioning of this compacted netlist is found and then re-expanded into an initial "flat" starting configuration for a second Kernighan-Lin phase. The approach may be iterated, with matching performed recursively on the compacted netlist until the problem size becomes manageable [3]. The heuristic justification for this approach [2] [3] is that the Kernighan-Lin k-opt method yields significantly better results when the graph topology is sufficiently dense, i.e., has large average degree.[3]

The approach of [2] [3] in effect performs clustering by finding cliques of size 2, i.e., the matching edges. We may generalize compaction into a more global approach by finding c-cliques for $c > 2$. Even more generally, we could find netlist subgraphs that have size c and a prescribed *density* (e.g., if more than $\epsilon \cdot C(c, 2)$ edges are present among c modules in the netlist, then the c modules would be considered to form

[2]Top-down partitioning makes early, *permanent* decisions of form, "keep modules $M_1, \ldots, M_{n/2}$ forever on the opposite half of the chip from modules $M_{n/2+1}, \ldots, M_n$". As problem sizes become large, such decisions must be made in increasingly ad hoc ways, with higher likelihood of a harmful decision. While the bottom-up clustering approach also makes permanent decisions, they are of form "always keep modules $M_1, \ldots, M_k$ positioned close together", and are less far-reaching than those of top-down partitioning, since $k << n$.

[3]Bui et al. claim that compacting until average degree in the netlist is ≥ 3 suffices for K-L to become essentially optimal. Lengauer [25] and the authors of [2] conjecture that this is because there are fewer local minima in the k-interchange neighborhood structure when the netlist graph has higher average degree.

1.2 Organization of Paper

The remainder of this paper is organized as follows. In Section 2, we describe the notion of a *natural* circuit decomposition and its intuitive motivation of the DS quality measure, which we propose as a general tool for objective evaluation of clustering heuristics. The analysis which leads to the DS quality measure also leads directly to the new clustering approach given in Section 3: we infer graph structure from the sequence of modules generated as we iteratively move to a random adjacent module in the netlist. Theoretical results bound the required length of the random walk. Section 3 also develops a practical clustering algorithm for netlist hypergraphs, and Section 4 gives experimental results for both the random input classes of Garbers et al. [13] as well as a number of MCNC benchmark netlists. We measure the utility of our method by DS quality of the clusters, as well as the improvement to partitioning algorithms that is afforded by the "condensed", clustered netlist. Our algorithm outperforms all the previous work of [2] [6] [13], and moreover enjoys such attractive features as stability and perfect parallelizability. The paper concludes in Section 5 with several directions for future research.

2 A Proper Clustering Metric

Our primary goal is to find an efficient clustering algorithm which is effective in the sense of losing as little structural information as possible. To this end, we first develop a robust measure of netlist structure, which we call the *DS quality* measure, and which gives an objective metric for distinguishing good clustering decompositions and clustering algorithms. We begin the discussion by establishing a criterion for *natural*, or *intrinsic*, hierarchical decompositions of a circuit netlist. We then list several theoretical implications of this criterion which provided the heuristic motivation for our development of the DS quality measure.

2.1 On Natural Structure and Separation in the Netlist

Bottom-up clustering, just as with top-down partitioning, entails *decomposition* of the netlist modules into S disjoint subsets. The quality of a decomposition algorithm is traditionally measured by the number of nets cut by a 2-way partition of some benchmark circuit [25]. However, such a measurement does not capture the integral role played by the partitioning algorithm in the *complete* layout synthesis process via its recursive (top-down) application. It is the overall hierarchical decomposition, rather than just a single partition at any given level, which has direct bearing on final layout quality. Thus, we wish to find a decomposition algorithm that generates the best *hierarchy* of subcircuits.[5] To this end, we follow recent

[5] While a bottom-up picture of circuit hierarchy is not usual, it is appropriate because in practice we must apply clustering hierarchically to achieve desired reductions in problem size. For example, spectral approaches in [17] break down at $n = 25000$, and application of such algorithms would require average cluster size of 40 if we start with a million-node netlist. With, e.g., the iterative matching-based compaction of Bui et al. [3], this would imply at least five or six levels in the clustering hierarchy.

work [18] and use the Rent parameter, a well-established quality measure for layout *hierarchies*, as a quality measure for the top-down or bottom-up decomposition *algorithms* that yield these hierarchies.

The so-called 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, $T = b \cdot C^p$, where T is the average number of external terminals (pins) in a subcircuit or partition (note that this is exactly what the net cut metric in S-way partitioning will measure!); b is a scaling constant which empirically corresponds to the average number of terminals per module; C is the number of modules in the subcircuit (or partition); and p, with $0 \leq p \leq 1$, is the *Rent parameter* of the decomposition.[6]

The Rent parameter has been studied extensively in the field of area estimation, where it affords accurate predictions of the layout wiring requirements for a given partitioning hierarchy. Donath [9] and Feuer [11] showed 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 authors of [18] proposed the notion of an *intrinsic Rent parameter*, denoted by p^*, which is the *minimum* Rent parameter attained over all hierarchical decompositions of a given circuit. By the results of Donath, Feuer and others, this lower bound gives a measure of the *required* layout area, independent of layout strategy; moreover, it affords a new methodology for comparing the utility of decomposition algorithms, independent of possible differences between the algorithms' individual objective functions. Using this approach, [18] showed that spectral partitioning algorithms which optimize the ratio cut metric [35] based on eigenvectors of the discrete Laplacian of the netlist graph[7] yield circuit hierarchies with much lower Rent parameters than traditional iterative methods such as the Fiduccia-Mattheyses k-opt approach [12]. Moreover, with each test case for which the intrinsic Rent parameter is known (e.g., the 2-D mesh with $p \geq 1/2$), the spectra-based ratio cut decomposition hierarchy had Rent parameter essentially *identical* to this theoretical lower bound. In other words, the spectral ratio cut approach was found to be in some sense an intrinsically good partitioning strategy. This correspondence may be formalized:

Fact 1: [17] The second smallest eigenvalue λ of the discrete Laplacian gives a tight lower bound for

[6] This relation was first observed by E. F. Rent of IBM in the late 1960s and independently by several others, e.g., Donath [8] derived the same relation from a stochastic model of a hierarchical design process. Following Mandelbrot [26], 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 example, "intrinsically 2-dimensional" circuits such as memory arrays, PLAs, or meshes will have optimal layouts with $p = 1/2$.

[7] 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*; in particular, the discrete Laplacian of the graph, $Q = D - A$, has been well studied [27] [31]. The spectral method in [16] computes the eigenvector corresponding to the second-smallest eigenvalue of the Laplacian $Q = D - A$ (also known as the *spectral gap*, since the smallest eigenvalue of Q is zero), and successfully uses it to induce a heuristic partition.

minimum ratio cut cost: $\frac{e(U,W)}{|U|\cdot|W|} \geq \frac{\lambda}{n}$, where n is the number of modules in the netlist and $e(U,W)$ is the net cut of the (U,W) module partition,

from which the authors of [18] derive a strong relationship between λ and the intrinsic Rent parameter p^*:

Fact 2: $\frac{\lambda}{n} = k\left[x^*(n-x^*)\right]^{\frac{p^*}{2}-1}$

where x and $n-x$ are the respective sizes of the optimal ratio partition. For our purposes, we need the following corollary of these results:

Fact 3: A larger λ value implies that the underlying graph is not easily separated, i.e., its optimal ratio cut cost is high.

Fact 3 gives a criterion for *natural decomposability* of a given (sub)circuit: any decomposition of a graph that has large λ value will cut a number of nets; similarly, when the graph has small λ value the subcircuit naturally admits further decomposition.

As noted in [18], these observations may be used to establish a relationship between the optimum hierarchy of subcircuits and the λ values of the discrete Laplacians of the subcircuits in this hierarchy. Our work heuristically exploits this relationship to achieve a bottom-up (hierarchical) clustering. We note that the theoretical correspondence between λ and cluster quality does *not* afford an efficient clustering algorithm: examining all subsets of c nodes in a netlist, computing the second eigenvalue of the induced subgraph for each node set, selecting the best cluster, etc. is impractical. Also, while Hagen et al. [18] used the theory of intrinsic Rent parameters to show the utility of top-down spectral ratio cut partitioning, our premise is that such an algorithm cannot be applied in a top-down fashion due to problem size. Thus, the theory of intrinsically good decompositions [18] yields neither an efficient clustering algorithm nor even an efficient means of assessing the quality of a cluster.

The key contribution of the present work lies in bringing together disparate results from graph theory and the theory of Markov processes in order to motivate a more useful measure of cluster quality, which we call the *DS quality measure*. The DS measure moreover directly suggests the possibility of an efficient random-walk based algorithm that is capable of discerning natural circuit clusters.

2.2 The DS Quality Measure

The DS metric is motivated by the following question: given a graph $G = (V, E)$, how easy is it to separate two nodes $s,\ t \in V$? Observe that (i) if s and t are hard to separate, then there must be more s-t paths and it is more likely that s and t belong to the same natural cluster; (ii) conversely, if s and t are easy to separate, then there must be fewer s-t paths and s and t probably do not belong to the same natural

cluster.[8]

We have found that the weighted average of the cluster *degree / separation* (DS) is a robust quality measure: (i) cluster *degree* is the average number of nets incident to each module of the cluster and having at least two pins in the cluster; and (ii) cluster *separation* is the average length of a shortest path between two nodes in the cluster, with separation ∞ if two nodes in the cluster are disconnected. Preliminary experiments indicate that the DS quality measure is highly robust; moreover, it is asymptotically easier to evaluate than, e.g., such metrics as k-l connectivity.

We calculate the DS quality of a clustering as the weighted average of the DS quality of each cluster, with a cluster containing a single node having DS quality equal to zero. The DS qualities of several different clusterings for the same eight-node graph are shown in Figure 1.[9] The intuition behind maximizing the DS quality is that we wish to find a decomposition of the graph such that nodes will *on average* have the highest possible degree and the shortest possible separation from the other nodes in their respective clusters. [10]

The DS quality suggests that the goal of a clustering algorithm should entail finding the neighborhood structure of a node v and comparing it with the neighborhood structure of other nodes to determine which nodes should be clustered with v. This notion of recognizing a node's neighborhood structure motivates our random walk based clustering algorithm.

3 Random Walks Yield Circuit Clusters

We now present our new RW-ST methodology, which computes a circuit clustering based on a random walk in the netlist graph. A *random walk* is a discrete-time stochastic process which iteratively moves from the current module (vertex) to a random adjacent module, with all adjacencies equiprobable.[11]

It is instructive to consider the progress of a random walk on the "barbell" example of Figure 2, which consists of two cliques joined by a chain. With some thought, a number of standard probabilistic results (cf. [4] [21]) are clear: (i) if we start at x, then all nodes in cluster A will be visited, i.e., "covered", within

[8] Relationships between the multiplicity of paths between nodes of a graph ("combinatorial entropy") and the second eigenvalue λ of the discrete Laplacian Q have been established by, e.g., Sinclair and Jerrum [31] in the study of Markov processes that arise in simulated annealing. Also, recall that the spectral gap, i.e., the size of λ, is inversely related to the decomposability of the cluster. Leighton and Rao [24] give an analysis of "flux cuts" (which are the same as ratio cuts) and multicommodity flows, showing that the connectedness of a cluster is, not surprisingly, closely related to the average multiplicity of paths between nodes of the cluster.

[9] For example, each cluster in the 2-clustering has average node degree = 10/4, and average separation = 14/12.

[10] We give clusters containing a single node a DS measure of zero to ensure that a clustering where each cluster contains only one node does not have DS measure higher than the DS measure of the original circuit. Defining the DS measure of a single-node cluster to be zero may actually be somewhat harsh to a "conservative" method such as our RW-ST algorithm below, which hesitates to commit modules to clusters. However, our results indicate that even with this handicap the random walk clusterings have considerably better DS measures than the DS measures of the complete netlists.

[11] Previous work in [6] uses a complicated weighting scheme for transition probabilities in the random walk. Our investigations have led to the somewhat surprising conclusion that an unweighted walk actually leads to better results; a side benefit is that the unweighted walk is also more tractable to existing analytic techniques.

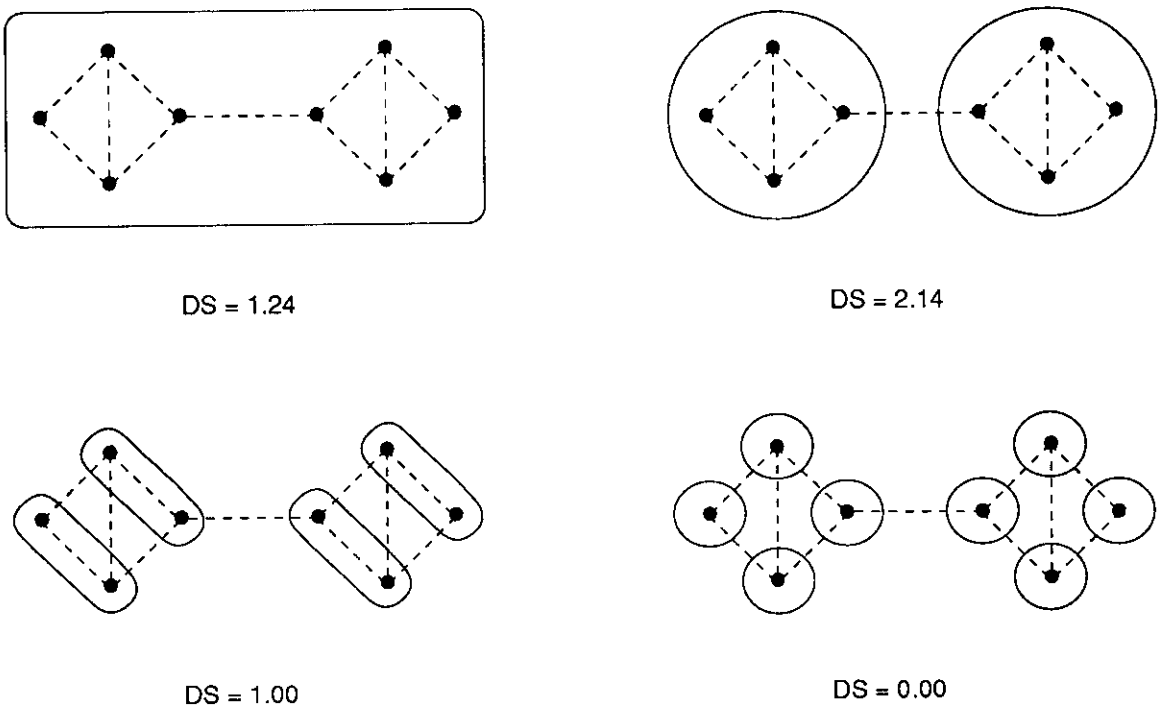

Figure 1: DS quality of various clusterings of the same graph.

$O(n\ log\ n)$ steps; (ii) the walk is not expected to escape to the middle chain until $\Omega(n^2)$ steps have been taken. If we remove the left clique A from the picture, then (iii) starting from y it will require $\Omega(n^2)$ steps before z is visited, but (iv) if we start at z, then $\Omega(n^3)$ steps will elapse before y is visited (since every time we return to z, the walk will wander around cluster B before again escaping to the chain).

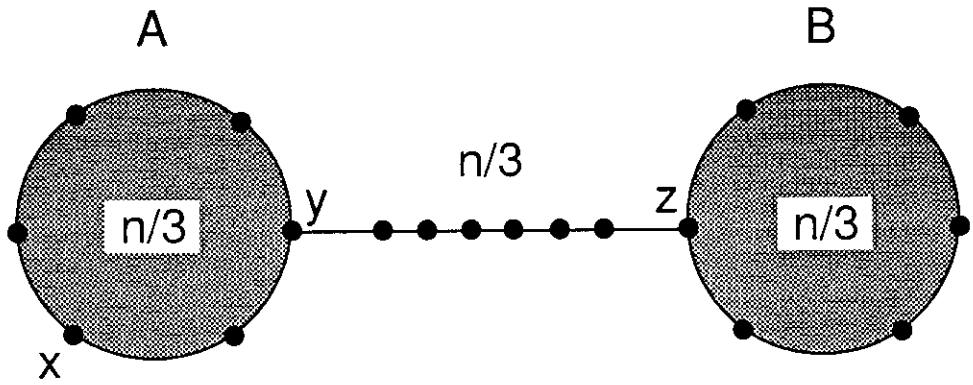

Figure 2: "Barbell" of two $n/3$-cliques joined by a chain of size $n/3$.

Define the *cover time* of G as the maximum, over all possible starting vertices, of the expected length of a random walk that visits all vertices in G. The following result shows that a random walk will with high probability manage to explore the netlist structure in a small number of steps.[12]

[12]By the discussion of the previous section, we know that the second eigenvalue of the discrete Laplacian, λ, captures the intrinsic graph structure. It turns out that λ also captures the cover time of random walks in the graph; cf. Gobel and Jagers [15] and the elegant resistive network analysis of Chandra et al. [4]. Thus, the intuitive motivation behind our methodology is

Fact 4: There is an $O(n^2)$ upper bound on the cover time of a random walk in a d-regular or nearly d-regular graph of n nodes; there is also an $\Omega(n\ log\ n)$ lower bound on the cover time of this class of graphs; and there exist examples which show that both bounds are tight. [21] □

Other authors [7] have shown that the $O(n^2)$ upper bound also holds for cover times of d-bounded graphs. We note that these results apply immediately to netlists of cell-based designs, which are essentially d-regular and certainly d-bounded. Therefore, we may compute a single random walk of length $\Theta(n^2)$ in the netlist graph, and expect to sample the entire netlist graph.[13]

We propose a method for extracting clusters from the random walk via the following concept of a *cycle*. Consider the sequence of nodes encountered during the random walk. A *cycle* is a contiguous subsequence $\{v_p, v_{p+1}, \ldots, v_q\}$ in the walk with $v_p = v_q$ and all v_i distinct, $i = p, p+1, \ldots, q-1$. The set of modules in each cycle should correspond to (part of) a natural cluster because if there is a more tightly coupled node subset of the cycle, then the random walk will recur (i.e., complete a smaller cycle) within that subset and we would not have found the original cycle. This is shown intuitively in Figure 3, where the y-y portion of the walk does not delimit a natural cluster since it contains a denser x-x cycle; the x-x cycle does not contain any denser portion, so we say that it is a bona fide cluster.

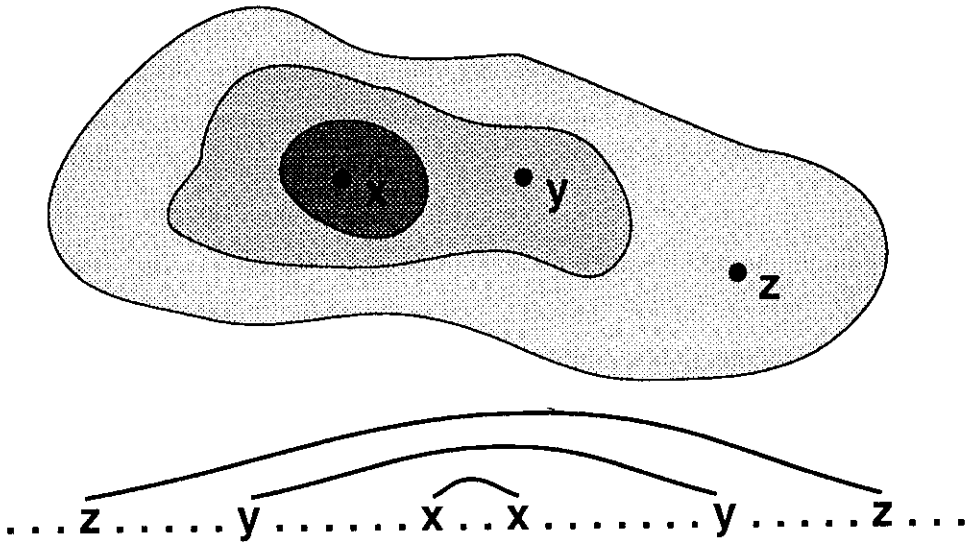

Figure 3: Progress of a random walk through areas with different edge density.

We have designed a *linear-time* algorithm for identifying all of the cycles in a random walk. This algorithm is given in Figure 4.

that the covering and "recurrence" properties of random walks might correspondingly capture the intrinsic netlist structure.

[13] It is interesting to consider related theoretical results of Gerl [14], of Mohar [28], and of Broder and Karlin [1], who show $O(n\ log\ n)$ upper bounds on the cover time of the isoperimetric and d-regular expander graph classes. Such graph classes have some similarities to real, hierarchical circuit netlists, and we do not know whether our $\Theta(n^2)$ walk length can be shortened to $\Theta(n\ log\ n)$.

```
Find-Cycles(RW)
Input: A sequence of nodes RW
for each node i
    visited[i] := FALSE
first := 1
last := 1
visited[last] := TRUE
while last < |RW|
    increment last
    if visited[RW[last]] = TRUE
        while RW[first] ≠ RW[last]
            visited[RW[first]] := FALSE
            increment first
        increment first
    visited[RW[last]] := TRUE
```

Figure 4: Finding cycles in linear time.

In [6], a random walk was computed in the netlist, *maximal* cycles $C(v_j)$ were determined for all modules v_j, and then the transitive closure of the relation $\bowtie$, defined by $v_a \bowtie v_b$ if $v_a \in C(v_b)$ and $v_b \in C(v_a)$, was used to induce a heuristic clustering. However, the experimental results of [6] fail to reflect the intuitively "correct" circuit organization. Our present work offers a different approach, the RW-ST algorithm, which extracts a good heuristic clustering from the cycle information. It should be emphasized that the RW-ST is a heuristic and that we do not yet have strong theoretical justification for its observed success.

The RW-ST algorithm clusters node pairs based on their *sameness*. The sameness of nodes u and v reflects the commonality of the sets of nodes that are visited in cycles originating at u and at v. To calculate the sameness, for each node v we must keep track of how often a node u occurs in some cycle originating at v. This number is saved in the array CC (CycleCount).

```
Sameness(u, v)
Input: A pair of nodes u and v
Output: The sameness value S of u, v
if (CC[u][v] = 0) or (CC[v][u] = 0)
    S := 0
else
    S := 2 · (CC[u][v] + CC[v][u])
    for each node w in the circuit
        if (w ≠ u) and (w ≠ v)
            if CC[u][w] > CC[v][w]
                S := S + 4 · CC[v][w] - CC[u][w]
            else
                S := S + 4 · CC[u][w] - CC[v][w]
```

Figure 5: Computing sameness of two nodes.

Using the CC array, the sameness value for nodes u and v is calculated as shown in Figure 5. If both $CC[u][v]$ and $CC[v][u]$ are greater than zero, i.e., each node occurs at least once in the other's cycles, sameness is initialized to $2 \cdot (CC[u][v] + CC[v][u])$. For each node w, the u-v sameness is increased if the values $CC[u][w]$ and $CC[u][v]$ are approximately equal; the sameness is decreased if these quantities vary by a significant amount. To be specific, for each node w in the circuit other than nodes u and v, we add $4 \cdot min - max$ to the sameness value, where min and max are respectively the smaller and larger of the two values $CC[u][w]$ and $CC[v][w]$.

Note that the term $4 \cdot min - max$ measures the commonality of nodes u and v with respect to w. If min and max are equal, sameness is increased by $3 \cdot min$; if min is zero or if max is considerably greater than min, sameness is decreased by max. Intuitively, this bias toward increasing the sameness affords some leeway in how close min and max must be in order to still have a positive impact on the sameness value; this is because the random walk cannot guarantee to visit u and v equally often even if they look identical to the rest of the circuit.

```
RW-ST(G)
Input: A graph G
Output: A set of clusters C
Construct a random walk RW on G
Find-Cycles(RW)
for each node u in G
    C(u) := u
for each pair of nodes u and v in G
    S := Sameness(u, v)
    if S > 0
        C(u) := C(u) ∪ C(v)
```

Figure 6: High-level description of RW-ST.

As shown in Figure 6, algorithm RW-ST first finds and processes all cycles in the random walk, then computes sameness for all node pairs, and finally clusters those node pairs with sameness greater than zero. In some sense, the sameness computation within the random walk implicitly compares the neighborhood structures of a given node pair. The time complexity of RW-ST is a function of the time required to process the random walk and the time required to calculate sameness for all node pairs. As mentioned above, we use a random walk of length $O(n^2)$ and find all cycles in the random walk in $O(n^2)$ time. Processing a cycle of length l_c requires $O(l_c)$ operations, yielding worst-case time complexity of $O(n^3)$ to process the random walk. However, in practice the average l_c value seems to grow sublinearly in n. Calculating the sameness of a node pair requires $O(n)$ operations, resulting in $O(n^3)$ time to calculate sameness values for all $O(n^2)$ node pairs. Since processing the random walk and calculating sameness values both have complexity $O(n^3)$, the overall worst-case complexity of RW-ST is $O(n^3)$. RW-ST is observed to be much faster since most

node pairs have no cycles in common, thus eliminating the need to calculate their sameness. The space requirements of our heuristic are $O(n^2)$ because the CC array records the cycle count for each node pair. Sparse matrix techniques can be used to reduce the required space at the expense of added time complexity.

4 Experimental Results

We tested the RW-ST method on two very distinct classes of inputs: (i) the random clustered inputs $G_{Gar}(m, n, p_{int}, p_{ext})$ studied by Garbers et al. [13], and (ii) the Primary and Test circuit netlists from the MCNC benchmark suite. Three different experiments were performed: (1) discovery of known clusters in G_{Gar} graphs; (2) DS measures of MCNC benchmark clusterings generated by RW-ST and the matching based compaction (MBC) scheme of Bui et al. [3]; (3) two-phase Fiduccia-Mattheyses (FM) style partitioning using RW-ST and MBC clusterings.

m	n	p_int	p_ext	proper (by DS)	Garbers (k,l) Big/Small	RW-ST $(nm)^2$ Big/Small	RW-ST $10(nm)^2$ Big/Small
100	10	0.1	0.0001	10	(2,2) 9/3	10/57	10/20
100	10	0.1	0.0002	10	(2,2) 3/3	10/62	10/20
100	10	0.1	0.0003	10	(2,2) 3/0	10/90	10/24
100	10	0.1	0.0004	10	(2,2) 1/0	10/88	10/27
100	10	0.1	0.001	10	(3,2) 9/49	10/264	10/61
100	10	0.1	0.002	10	(3,2) 1/45	6/881	10/242
100	10	0.1	0.003	10	(3,2) 2/40	0/1000	10/427
100	10	0.1	0.004	10/1	(3,2) 1/40	0/1000	10/527

Table 1: Comparison of random walk based clustering with (k,l)-connectivity based clustering. Randoms walks of lengths $(nm)^2$ and $10(nm)^2$ were examined. The results give the numbers "Big" and "Small" for each clustering: following the presentation of Garbers et al., "Big" is defined as the number of clusters containing more than $\frac{1}{10}n$ nodes, while "Small" is the number of nodes that do not belong to any "Big" cluster.

Garbers et al. in [13] presented a class of random graphs defined as $G_{Gar}(m, n, p_{int}, p_{ext})$, where m is the number of clusters, n is the size of a cluster, and an edge (u, v) is independently present with probability p_{int} if u and v are in the same cluster and probability p_{ext} otherwise. We used this class of random examples in our first set of experiments, to determine whether RW-ST could find the "correct" graph clustering. Our results are compared against the published statistics in [13]. The RW-ST algorithm was run on walks of length $(nm)^2$ and $10(nm)^2$ in order to see how walk length affected solution quality. The results of Table 1 show that RW-ST gives much more consistent results than (k,l)-connectivity. For walks of length $10(nm)^2$, RW-ST found 10 distinct clusters for each benchmark tested. In contrast, (k,l)-connectivity found "correct" clusterings for only two of the benchmarks, even when we allow the best results over a range of k values. This gives experimental confirmation of the self-tuning property inherent in RW-ST.

The "proper" field in the table indicates which of the 10-clustering or the 1-clustering (i.e., the complete circuit) has higher DS quality. Note that for $G_{Gar}(10, 100, 0.1, 0.004)$ these two values are nearly identical, i.e. this circuit no longer has an obvious clustering structure by our criterion.

Matching-Based-Compaction(H, k)

Input: A hypergraph H
Desired number of clusters k

Output: A set C of clusters

```
convert H into a simple graph G using clique model
for each node u in G
    C(u) := u
while |C| > k
    mark each cluster free
    construct G' the subgraph of G induced by C
    while (|G'| > 0) and (|C| > k)
        (u, v) := random edge in G'
        G' := G' - (u, v)
        if C(u) and C(v) are free
            C(u) := C(u) ∪ C(v)
            mark C(u) not free
```

Figure 7: Algorithm to generate a k-way partitioning using random maximal matchings, as suggested by Bui et al.

The second set of experiments compared the RW-ST method with the matching based compaction (MBC) method of Bui et al. [3] by examining the DS quality of their respective clusterings on MCNC benchmarks. To ensure a "fair" comparison, we required the MBC clustering to have the same number of clusters as the RW-ST clustering. The original MBC results in [3] were based on constructing a clustering by finding a random maximal matching of the nodes. However, the number of clusters in an RW-ST clustering will normally be much less than half the original size of the circuit. We therefore modified the original MBC code to iteratively compute maximal random matchings, with each new matching performed on the graph induced from the previous clustering, until the desired reduction in problem size was obtained. Figure 7 gives the pseudo-code for our implementation of MBC.[14]

Table 2 shows the DS quality of the RW-ST and MBC clusterings. The RW-ST clusterings uniformly dominate the MBC clusterings in terms of DS quality. In addition, the improvement in DS quality is greater for larger circuits, possibly indicating that the random matching method breaks down as the problem size increases. For the two large examples Test04 and Test05, we observe improvements in DS quality of over 30%. Finally, note that the work of [6] only analyzed the Primary1 and bm1 benchmarks, obtaining DS qualities of 0.922 and 0.852, respectively.

[14] A note regarding the MBC code concerns the matter of hyperedges. In the original work of Bui et al. [2] only simple graphs were analyzed. We transformed netlist hyperedges to cliques, thus preserving the connectivity of the original circuit and allowing a random maximal matching to be constructed in a straightforward fashion.

Benchmark	Size	MBC			RW-ST		
		DS	Areas	Net cut	DS	Areas	Net cut
19ks	2844	1.166	5619:5383	456	1.578	5501:5501	153
bm1	882	1.189	1812:1668	94	1.221	2197:1283	39
PrimGA1	833	1.258	1719:1712	82	1.325	2180:1251	37
PrimSC1	833	1.258	1377:1376	91	1.325	1701:1052	40
PrimGA2	3014	1.238	4187:4186	303	1.566	4464:3909	154
PrimSC2	3014	1.238	3877:3829	266	1.566	4079:3627	145
Test02	1663	1.231	38141:18909	75	1.593	37132:19918	42
Test03	1607	1.185	14748:7481	132	1.566	12629:9600	74
Test04	1515	1.297	21105:20935	61	1.879	21055:20985	45
Test05	2595	1.275	62437:10161	51	1.689	39067:33531	10
Test06	1752	1.331	8485:8483	381	1.367	9444:7524	89

Table 2: DS qualities and Fiduccia-Mattheyses partitioning results of RW-ST and MBC clusterings.

To further confirm the greater utility of the RW-ST clusterings over MBC clusterings, we ran Fiduccia-Mattheyses (FM) partitioning on the resulting clustered graphs. These results are also summarized in Table 2, and we readily observe that the MBC clusterings produce very poor partitionings. This is somewhat surprising, since random matching based clustering was reported to be an efficient way of obtaining good initial starting points for the Kernighan-Lin approach [2] [3].

Cluster-FM(G, C)
Input: A graph G and a clustering C of G
construct G' the subgraph of G induced by C $P :=$ FM-partition of G' expand P to be a partition of G run FM-partition on G with P as the initial partition

Figure 8: Two-phase FM partitioning algorithm.

Our final experiments tested the original conjecture in [2], namely, that a good clustering will improve the solution quality of FM partitioning. For each heuristic clustering, we applied a two-phase FM algorithm outlined in Figure 8 which in the first phase partitioned the graph induced by the clustering, and then in the second phase used the expanded partition from the first phase as the starting point for FM partitioning on the "flat" circuit.

The results of this experiment are summarized in Table 3. Note that the results presented in Tables 2 and 3 are the best of 20 trials. We compared the results from running the two-phase FM partitioning algorithm on RW-ST and MBC clusterings against the results from running FM partitioning on the original circuit. Also in conformance with [3] we verified that the average degrees of the MBC clustering graphs were all greater than three (in fact, they ranged from 8 to 15, which more than meets the criterion given by Bui et al. [3] for the two-phase strategy to return "near-optimal" Kernighan-Lin results). In both cases there

Benchmark	Size	Standard FM		MBC		RW-ST	
		Areas	Net cut	Areas	Net cut	Areas	Net cut
19ks	2844	5501:5501	151 (1.000)	5501:5501	156 (1.033)	5501:5501	146 (0.967)
bm1	882	1740:1740	65 (1.000)	1740:1740	54 (0.831)	1740:1740	58 (0.892)
PrimGA1	833	1716:1715	66 (1.000)	1718:1713	48 (0.727)	1716:1715	47 (0.712)
PrimSC1	833	1377:1376	59 (1.000)	1377:1376	61 (1.034)	1377:1376	58 (0.983)
PrimGA2	3014	4187:4186	242 (1.000)	4187:4186	187 (0.773)	4187:4186	165 (0.682)
PrimSC2	3014	3853:3853	235 (1.000)	3858:3848	175 (0.745)	3853:3853	159 (0.677)
Test02	1663	37132:19918	42 (1.000)	37132:19918	42 (1.000)	37132:19918	42 (1.000)
Test03	1607	11115:11114	84 (1.000)	13729:8500	59 (0.702)	13188:9041	71 (0.845)
Test04	1515	40732:1308	12 (1.000)	40938:1102	20 (1.667)	40932:1108	14 (1.167)
Test05	2595	38753:33845	24 (1.000)	62586:10012	4 (0.167)	39089:33509	5 (0.208)
Test06	1752	8484:8484	87 (1.000)	8484:8484	83 (0.954)	8484:8484	82 (0.943)

Table 3: Comparison of two-phase Fiduccia-Mattheyses partitioning of random walk clusterings and random matching based clusterings. Standard Fiduccia-Mattheyses partitioning results are included as a control. RW-ST clusterings lead to a 17% improvement in net cut over standard FM.

was a significant improvement over the standard FM solution quality, with a 12% improvement obtained using MBC clusterings and a 17% improvement obtained using RW-ST clusterings. These results in some sense confirm the conclusions of [3].

An interesting observation is that the huge discrepancy in FM partition quality between the RW-ST and MBC clusterings, as shown in Table 2, are not reflected in the two-phase FM partitioning results, i.e., a large improvement in the quality of the starting partition does not translate into a correspondingly large increase in the quality of the final partition.

5 Extensions

There are many promising directions for future work. First, we are currently pursuing a parallel implementation of the random walk methodology. In other words, we partition the random walk computation evenly among p available processors; the cycle-finding within the random walks is also performed on each separate processor. This is appropriate for two reasons: (i) the hierarchical organization and sparsity of real netlist graphs permits only very short self-avoiding walks, and thus little information is lost by breaking the random walk up among processors (this is indeed confirmed by our results), and (ii) results of Coppersmith et al. [7] show that the separate walks together will reproduce a single long walk.[15] This parallel approach achieves perfect speedup over our current uniprocessor implementation.

Second, we hope to use the DS quality measure as the basis of other "implicitly global" clustering methods. Certainly, standard combinatorial methods and direct epitaxial-growth approaches can both be

[15] To be specific, [7] shows that two random walks will "collide" within a very short time when the graph is of low maximum degree and small diameter, as is the case with real netlist graphs having fanout limitations and large (e.g., clock) signal nets.

modified to incorporate the DS criterion within the clustering objective. We suspect that these sorts of algorithm variants can be shown to achieve globally good clustering decompositions in a probabilistic sense. (Also note that the DS clustering is enabling for Kernighan-Lin on small problems, in addition to the motivating million-node instances!)

Finally, the concept of a "natural clustering", independent of both the predefined cluster cardinality S and size limitations on the natural clusters, gives rise to a host of very interesting layout problems. In particular, the placement phase of layout becomes one of placing *malleable*, variable-size clusters, and is certainly of independent research interest. Following the basic premise of our work, the natural clustering will also enable use of more sophisticated optimizations such as the spectral and relaxation methods in the context of "fast placement" for the next generation standard cell and sea of gates designs.

5.1 Acknowledgements

Fiduccia-Mattheyses code was provided by J. Cong and M. Smith. We are also grateful to A. Steger for providing access to the preliminary results of [13].

References

[1] A. Z. Broder and A. R. Karlin, "Bounds on the Cover Time", *Journal of Theoretical Probability*, 2(1), 1989, pp. 101-119.

[2] 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.

[3] T. N. Bui, "Improving the Performance of the Kernighan-Lin and Simulated Annealing Graph Bisection Algorithms", *Proc. ACM/IEEE Design Automation Conf.*, 1989, pp. 775-778.

[4] A. Chandra, P. Raghavan, W. L. Ruzzo, R. Smolensky and P. Tiwari, "The Electrical Resistance of a Graph Captures its Commute and Cover Times", *Proc. ACM Symp. on Theory of Computing*, May 1989, pp. 574-586.

[5] C. K. Cheng and E. S. Kuh, "Module Placement Based on Resistive Network Optimization", *IEEE Trans. on CAD* 3(1984), pp. 218-225.

[6] J. Cong, L. Hagen and A. B. Kahng, "Random Walks for Circuit Clustering", *Proc. IEEE Intl. Conf. on ASIC*, June 1991, pp. 14.2.1 - 14.2.4.

[7] D. Coppersmith, P. Tetali and P. Winkler, "Collisions Among Random Walks on a Graph", to appear in *SIAM J. Discrete Math.*.

[8] W. Donath, "Hierarchical structure of computers", Technical Report RC 2392, IBM T. J. Watson Research Center, Yorktown Heights,N.Y., 1969.

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

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

[11] M. Feuer, "Connectivity of random logic", *IEEE Transactions on Computers* C-31(1) (1982), pp. 29-33.

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

[13] J. Garbers, H. J. Promel and A. Steger, "Finding Clusters in VLSI Circuits", (*preliminary version* of paper in) *Proc. IEEE Intl. Conf. on Computer-Aided Design*, 1990, pp. 520-523. Also *personal communication*, A. Steger, April 1992.

[14] P. Gerl, "Random Walks on Graphs with a Strong Isoperimetric Property", *Journal of Theoretical Probability*, 1(2), 1988, pp. 171- 187.

[15] F. Gobel and F. F. Jagers, "Random Walks on Graphs", *Stochastic Processes and their Applications*, 2 (1974), pp. 311-336.

[16] L. Hagen and A. B. Kahng, "Fast Spectral Methods for Ratio Cut Partitioning and Clustering", *Proc. IEEE Intl. Conf. on Computer-Aided Design*, 1991, pp. 10-13.

[17] L. Hagen and A. B. Kahng, "New Spectral Methods for Ratio Cut Partitioning and Clustering", to appear in *IEEE Trans. on CAD*, Oct. 1992.

[18] L. Hagen, A. B. Kahng, F. Kurdahi and C. Ramachandran, "On the Intrinsic Rent Parameter and New Spectra-Based Methods for Wireability Estimation", to appear in *Proc. European Design Automation Conference*, October 1992.

[19] A. Itai, Y. Perl and Y. Shiloach, "The Complexity of Finding Maximum Disjoint Paths with Length Constraints", *Networks* 12 (1982), pp. 277-286.

[20] D. S. Johnson, C. R. Aragon, L. A. McGeoch and C. Schevon, "Optimization by Simulated Annealing: An Experimental Evaluation, Part I, Graph Partitioning", *Operations Research* 37 (1989), pp. 865-892.

[21] J. D. Kahn, N. Linial, N. Nisan and M. E. Saks, "On the Cover Time of Random Walks on Graphs", *J. of Theoretical Probability* 2(1) (1989), pp. 121-128.

[22] S. Kauffman and S. Levin, "Toward a General Theory of Adaptive Walks on Rugged Landscapes", *J. Theoretical Biology* 128 (1987), pp. 11-45.

[23] B. W. Kernighan and S. Lin, "An efficient heuristic for partitioning graphs", *Bell Syst. Tech. J.* 49(2) (1970), pp.291-307.

[24] T. Leighton and S. Rao, "An Approximate Max-Flow Min-Cut Theorem for Uniform Multicommodity Flow Problems with Applications to Approximation Algorithms", *IEEE Annual Symp. on Foundations of Computer Science*, 1988, pp. 422-431.

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

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

[27] M. Mihail, "Conductance and Convergence of Markov Chains – A Combinatorial Treatment of Expanders –", in *Proc. IEEE Symp. on Foundations of Computer Science*, 1989, pp. 526-531.

[28] B. Mohar, "Eigenvalues, Diameter, and Mean Distance in Graphs", *Graphs and Combinatorics*, 7 (1991), pp. 53-64.

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

[30] C. Sechen and K. W. Lee, "An Improved Simulated Annealing Algorithm for Row-Based Placement", *Proc. IEEE Intl. Conf. on Computer-Aided Design*, 1987, pp. 478-481.

[31] A. Sinclair and M. Jerrum, "Approximate Counting, Uniform Generation and Rapidly Mixing Markov Chains", *Information and Computation*, 82 (1989), pp. 93-133.

[32] A. Srinivasan, K. Chaudhury and E. S. Kuh, "RITUAL: Performance Driven Placement Algorithm for Small Cell ICs", *Proc. IEEE Intl. Conf. on Computer-Aided Design*, 1991, pp. 48-51.

[33] R. Swartz and C. Sechen, "New Algorithms for the Placement and Routing of Macro Cells", *Proc. IEEE Intl. Conf. on Computer-Aided Design*, 1990, pp. 336-339.

[34] R. S. Tsay and E. S. Kuh, "A unified approach to partitioning and placement" in *Proc. Princeton Conf. on Inf. and Comp.*, 1986.

[35] Y. C. Wei and C. K. Cheng, "Towards efficient hierarchical designs by ratio cut partitioning", in *Proc. IEEE Intl. Conf. on Computer-Aided Design*, 1989, pp. 298-301.

[36] Y.C. Wei and C.K. Cheng, "A Two-Level Two-Way Partitioning Algorithm", *Proc. IEEE Intl. Conf. on Computer-Aided Design*, 1990, pp. 516-519.