SC '95, San Diego, CA

Analysis of Multilevel Graph Partitioning *

GEORGE KARYPIS AND VIPIN KUMAR

University of Minnesota,
Department of Computer Science
Minneapolis, MN 55455

{karypis, kumar}@cs.umn.edu

Abstract

Recently, a number of researchers have investigated a class of algorithms that are based on multilevel graph partitioning that have moderate computational complexity, and provide excellent graph partitions. However, there exists little theoretical analysis that could explain the ability of multilevel algorithms to produce good partitions. In this paper we present such an analysis. We show under certain reasonable assumptions that even if no refinement is used in the uncoarsening phase, a good bisection of the coarser graph is worse than a good bisection of the finer graph by at most a small factor. We also show that for planar graphs, the size of a good vertex-separator of the coarse graph projected to the finer graph (without performing refinement in the uncoarsening phase) is higher than the size of a good vertex-separator of the finer graph by at most a small factor.

*This work was supported by IST/BMDO through Army Research Office contract DA/DAAH04-93-G-0080, and by Army High Performance Computing Research Center under the auspices of the Department of the Army, Army Research Laboratory cooperative agreement number DAAH04-95-2-0003/contract number DAAH04-95-C-0008, the content of which does not necessarily reflect the position or the policy of the government, and no official endorsement should be inferred. Access to computing facilities were provided by Minnesota Supercomputer Institute, Cray Research Inc, and by the Pittsburgh Supercomputing Center. Related papers are available via WWW at URL: **http://www.cs.umn.edu/users/kumar/papers.html**

1 Introduction

Graph partitioning is an important problem that has extensive applications in many areas, including scientific computing, VLSI design, and task scheduling. The problem is to partition the vertices of a graph in p roughly equal parts, such that the number of edges connecting vertices in different parts is minimized. For example, the solution of a sparse system of linear equations $Ax = b$ via iterative methods on a parallel computer gives rise to a graph partitioning problem. A key step in each iteration of these methods is the multiplication of a sparse matrix and a (dense) vector. Partitioning the graph that corresponds to matrix A is used to significantly reduce the amount of communication [11]. If parallel direct methods are used to solve a sparse system of equations, then a graph partitioning algorithm can be used to compute a fill reducing ordering that lead to high degree of concurrency in the factorization phase [11, 4].

The graph partitioning problem is NP-complete. However, many algorithms have been developed that find reasonably good partitions. Spectral methods [17, 7] have been shown to be quite effective for partitioning unstructured problems in a variety of applications, but have very high computational complexity. The MSB algorithm produces partitions that are as good as those produced by the original spectral bisection, but it is one to two orders of magnitude faster as it computes the Fiedler vector of the graph using a multilevel approach [1]. Geometric partition methods [5, 15, 16] are quite fast but they often provide worse partitions than those of more expensive methods such as spectral. Furthermore, geometric methods are applicable only if coordinate information for the graph is available.

Recently, a number of researches have investigated a class of algorithms that have moderate computational complexity, and provide excellent (even better than spectral) graph partitions [3, 7, 9]. The basic idea behind these algorithms is very simple. The graph G is first coarsened down to a few hundred vertices, a bisection of this much smaller graph is computed, and then this partition is projected back towards the original graph (finer graph) by periodically refining the partition. Since the finer graph has more degrees of freedom, such refinements usually decrease the edge-cut. These are called multilevel graph partitioning schemes. In particular, in [9, 10] we have developed a multilevel graph partitioning scheme that produces high quality partitions in small amount of time. Our algorithm produces partitions that are 10% to 60% better than those produced by spectral partitioning algorithms [17, 1], and 5% to 30% than those produced by other multilevel algorithms [7]. Furthermore, our algorithm is 10 to 40 times faster than multilevel spectral bisection (as implemented in the Chaco graph partitioning package [6]), and 2 to

6 times faster than the multilevel algorithm of [7]. We also used our multilevel graph partitioning scheme to compute fill reducing orderings for sparse matrices [9]. Surprisingly, our scheme substantially outperforms the multiple minimum degree algorithm [13], which is the most commonly used method for computing fill reducing orderings of a sparse matrix.

From the experiments presented in [9, 3, 7], it is clear that multilevel graph partitioning algorithms are able to find high quality partitions for a variety of unstructured graphs. However, there exists little theoretical analysis that could explain the ability of multilevel algorithms to produce good partitions. In this paper we present such an analysis. We show under certain reasonable assumptions that even if no refinement is used in the uncoarsening phase, a good bisection of the coarser graph is worse than a good bisection of the finer graph by at most a small factor. We also show that the size of a good vertex-separator of the coarse graph projected to the finer graph (without performing refinement in the uncoarsening phase) is higher than the size of a good vertex-separator of the finer graph by at most a small factor.

2 Multilevel Graph Bisection

In this section we briefly describe the various phases of the multilevel algorithm. The reader should refer to [9] for further details.

Coarsening Phase During the coarsening phase, a sequence of smaller graphs $G_i = (V_i, E_i)$, is constructed from the original graph $G_0 = (V_0, E_0)$ such that $|V_i| > |V_{i+1}|$. Graph G_{i+1} is constructed from G_i by finding a maximal matching $M_i \subseteq E_i$ of G_i and collapsing together the vertices that are incident on each edge of the matching. In this process no more than two vertices are collapsed together because a matching of a graph is a set of edges, no two of which are incident on the same vertex. Vertices that are not incident on any edge of the matching, are simply copied over to G_{i+1}.

When vertices $v, u \in V_i$ are collapsed to form vertex $w \in V_{i+1}$, the weight of vertex w is set to be equal to the sum of the weights of vertices v and u, while the edges incident on w is set equal to the union of the edges incident on v and u minus the edge (v, u). If there is an edge that is incident to both on v and u, then the weight of this edge is set equal to the sum of the weights of these edges. Thus, during successive coarsening levels, the weight of both vertices and edges increases.

Maximal matchings can be computed in different ways [9]. The method used to compute the matching greatly affects both the quality of the bisection, and the time required during the uncoarsening phase. Here we briefly describe two such matching schemes.

The first scheme, which we called **random matching** (RM), computes the maximal matching by using a randomized algorithm [3, 7]. The RM scheme works as follows. The vertices of the graph are visited in random order. If a vertex u has not been matched yet, then an unmatched adjacent vertex v is randomly selected and the edge (u, v) is included in the matching. The second scheme, which we call **heavy-edge matching** (HEM), computes a matching M_i, such that the weight of the edges in M_i is high. The HEM matching is computed using a randomized algorithm similar to the one used for RM. The vertices are again visited in random order. However, instead of randomly matching a vertex with one of its adjacent unmatched vertices, HEM matches it with the unmatched vertex that is connected with the heavier edge. As a result, the HEM scheme reduces the sum of the weights of the edges in the coarser graph by a larger amount than RM. In [9], we experimentally evaluated both the RM and HEM matching schemes, and we found that the HEM scheme produces consistently better results than RM, and the amount of time spent in refinement is less than that of RM.

Initial Partitioning Phase The second phase of a multilevel algorithm is to compute a balanced bisection of the coarsest graph $G_k = (V_k, E_k)$. An evaluation of different algorithms for partitioning the coarser graph can be found in [9].

Uncoarsening Phase During the uncoarsening phase, the partition of the coarsest graph G_k is projected back to the original graph by going through the graphs $G_{k-1}, G_{k-2}, \ldots, G_1$. Furthermore, even if the partition of G_i is at a local minima, the partition of G_{i-1} obtained by this projection may not be at a local minima. Hence, it may still be possible to improve the the partition of G_{i-1} obtained by the projection by using local refinement heuristics. For this reason, a partition refinement algorithm is used. A number of refinement algorithms are investigated in [9], that significantly improve the quality of the projected partition.

Matrix Name	No. of Vertices	No. of Edges	Description
3ELT	4720	13722	2D Finite element mesh
4ELT	15606	45878	2D Finite element mesh
BCSSTK31	35588	572914	3D Stiffness matrix
BRACK2	62631	366559	3D Finite element mesh
WAVE	156317	1059331	3D Finite element mesh
WHITAKER3	9800	28989	2D Finite element mesh

Table 1: Various matrices used in evaluating the multilevel graph partitioning and sparse matrix ordering algorithm.

3 Analysis

The central theme of the analysis is as follows. If the graph is coarsen "perfectly", then the coarsest graph will be an exact replica of the finer (original) graph except that it will be smaller. Hence, an optimal bisection of this coarsest (smaller) graph will also be an optimal bisection of the finer graph. If the graph has been coarsened (using a perfect coarsening scheme) enough times, then it becomes small enough that we can find a "near-optimal" bisection in a reasonable amount of time. This near-optimal bisection of the coarsest graph will also be a good bisection of the original graph. If the coarsening scheme is not good, then it is entirely possible that a near-optimal bisection of the coarsest graph is an arbitrarily bad bisection of the finer (original) graph. The matching strategies (RM and HEM) used in our multilevel algorithm do not lead to optimal coarsening. Hence, a near-optimal bisection of the coarsest graph obtained using RM or HEM, is not guaranteed to be a near-optimal bisection of the finer graph. The analysis in Section 3.2 shows that under some reasonable assumptions, the edge-cut of a near-optimal bisection of the coarser graph is larger than the edge-cut of a near-optimal bisection of the finer graph only by a small constant factor. The analysis also motivates why this "penalty" factor is even smaller when the HEM scheme is used (instead of RM).

Similarly, if the graph is coarsened perfectly, then a near-optimal vertex separator of the coarser graph can be projected to the finer graph to construct a near-optimal separator of the finer graph. As for the case of the edge-cut, if the coarsening scheme is not good, then the projected separator of the coarser graph can lead to an arbitrarily bad separator of the finer graph. The vertex-separator analysis in Section 3.3 shows that under reasonable assumptions, the projection of a near-optimal separator of the coarser graph leads to a separator for the finer graph that is worse than a near-optimal separator (for the finer graph) only by a small constant factor.

Both of these analyses show that even if no refinement is performed during the uncoarsening phase, the bisection of the coarsest graph is also a good bisection of the original graph, especially if HEM is used for coarsening. These observations are supported by experimental results in both cases.

3.1 Definitions and Assumptions

Analyzing the quality of the bisections produced by multilevel graph partitioning algorithms is particularly hard, and can only be done if certain assumptions are made about the original graphs and the coarsening process. In the rest of this section we present these assumptions and some notation that will be used throughout the analy-

sis.

Let $G_0 = (V_0, E_0)$ be the original graph, and $G_i = (V_i, E_i)$ be the ith level coarser graph. For each graph G_i, let $W(M_i)$ be the sum of the weights of the edges in the matching used to obtain G_{i+1}, $W(E_i)$ be the sum of the weights of the edges, ω_i be the average edge-weight, d_i be the average degree of a vertex, and C_i be the size of the edge-cut (*i.e.*, the weight of the edges crossing parts).

To simplify the presentation of the analysis we assume that at each successive level of coarsening, the number of vertices reduce by a factor of two[1], *i.e.*, $|V_i| = 2|V_{i+1}|$. Consequently, the matching at level i contains $|V_i|/2$ edges; hence, the weight of the matching is equal to $W(M_i) = \delta\omega_i|V_i|/2$, where δ is a constant that captures the properties of RM and HEM. In particular, $\delta = 1$ for RM (because the matched edges in RM are chosen randomly), and $\delta \geq 1$ for HEM (because the HEM prefers edges with higher weight). Also, to simplify the analysis we assume that the average degree of the successive coarser graphs changes at a constant rate, *i.e.*, $d_i/d_{i+1} = \beta$. Note that if $\beta < 1$, the average degree increases, if $\beta > 1$ the average degree decreases, and if $\beta = 1$ the average degree remains the same.

In the rest of the analysis we assume that the following are true.

Assumption 1 (Small Separators) *There are constants α and γ such that each graph G_i has a balanced separator of size less than or equal to $\alpha|V_i|^\gamma$.*

Assumption 2 (Size of the Edge-Cut) *The edge-cut of G_i is proportional to $\alpha d_i\omega_i|V_i|^\gamma$. That is, C_i is proportional to the size of the balanced separator, the average degree of the graph, and the average weight of each edge.*

In the case of graphs arising in finite element applications, the small separator assumption is true. In particular, for planar graphs $\gamma = 0.5$ [12], and for the graphs that correspond to 3D finite element meshes, $\gamma = 2/3$ [15]. Assumption 2 follows directly from Assumption 1 if the successive coarser graphs have constant bounded degree[2]. The edge-cut in Assumption 2 corresponds to the bisection in which the separator is the boundary of one of the two parts.

[1]The analysis will go through if we assume that the number of vertices reduces by a constant factor. This assumption appears to be valid for well-shaped finite element meshes, as from our experiments we know that for most coarsening levels somewhere between 85% to 95% of the vertices get matched. RM tends to match more vertices than HEM, but the difference between them is small (less than 5%) for most coarsening levels.

[2]Graphs that correspond to 2D and 3D finite element meshes have bounded degree. In the coarsening process due to randomization, the degrees remain bounded even for the coarser graphs. Due to the bounded degree, the edge-cut of these graphs cannot be smaller asymptotically, because it will lead to an asymptotically smaller vertex separator.

3.2 Edge-Cut Analysis

The multilevel algorithm can be viewed as an approximation algorithm since every successively coarse graph becomes a successively rougher approximation of the original graph. Since more and more features of the original graph are eliminated in this process, the coarser graphs have bisections with larger edge-cut than the original graph. It is natural to ask how bad can the edge-cut of the coarser graph get with respect to the edge-cut of the original graph. In particular the following theorem is true [8].

Theorem 1 *The size of the edge-cut for graph* G_i *for* $i \geq 1$ *is*

$$C_i \propto 2^{1-\gamma} \left(1 - \frac{\delta \beta^{i-1}}{d_0}\right) C_{i-1}. \quad (1)$$

Equation 1 reveals significant information about the quality of edge-cut when it is computed at a coarse graph. In particular, for a given graph, the increase in the edge-cut between successive coarse graphs decreases as either δ or β increases. That is, if the weight of the edges in the independent set used to coarsen the graph is much higher than the average weight of the edges ($\delta > 1$), then the penalty paid for finding a partition at the coarse graph is smaller. Similarly, as the average degree of the coarse graph decreases ($\beta > 1$), again the increase of the edge-cut at the coarse graph is smaller. In the next two sections we see how Equation 1 applies to graphs that correspond to 2D and 3D finite element meshes with triangular and tetrahedron elements, respectively.

3.2.1 2D Finite Element Meshes

The 2D finite elements meshes correspond to planar graphs. Furthermore, when the elements of the mesh are triangles (*e.g.*, when the finite element mesh is generated using Delaunay triangulation), then the graph that corresponds to the interior of the mesh is maximally planar. For the rest of this section we use this correspondence and we only concentrate on maximally planar graphs.

Planar graphs have been extensively studied and a great deal of properties are known about them. In particular, for maximally planar graphs $\gamma = 0.5$ [12], and $d_0 \approx 6$. Also, in [12] it was shown that edge contraction also preserves maximal planarity; thus, $\beta = 1$.

From Equation 1, and for RM (*i.e.*, $\delta = 1$), we have that

$$C_i^{2D-RM} \propto 1.18 C_{i-1}^{2D-RM} = 1.18^i C_0. \quad (2)$$

Thus, the edge-cut increases only by 18% at each successive coarsening level. For instance, the edge-cut after 10 coarsening levels is only 5.2 times worse than the edge-cut of the original graph.

However, the size of G_{10} is smaller than G_0 by a factor of 1024, so it is much quicker to find a good partition of G_{10} than of G_0. As discussed in Section 3.2, the increase in the edge-cut at successive coarsening levels is smaller when HEM is used, because in this case $\delta > 1$.

3.2.2 3D Finite Element Meshes

The graphs that correspond to 3D finite element meshes do not correspond to any extensively studied class of graphs as the 2D finite element graphs did. Nevertheless, for these type of graphs it is known that $\gamma = 2/3$ [15] and that for most finite element applications d_0 ranges between 12 and 16 [2]. However, in order to apply Equation 1, we need to know the value of β. Unlike maximally planar graphs, for which the average degree of successive coarser graphs remains the same, the average degree of 3D finite element graphs does not always remain the same. In particular, if $d_0 > 12$, and RM is used to coarsen the graph, d_i increases in successive coarser graphs. However, if HEM is used to coarsen the graph, then the average degree of the graph actually decreases in successive coarsening levels as discussed later. In particular, for 3D finite element meshes, the following theorem holds for the RM matching scheme [8].

Theorem 2 (Average Degree of 3D Graphs) *The average degree of the interior vertices of the ith level coarser graph G_i, of a graph G_0 that corresponds to a 3D finite element mesh with tetrahedron elements is*

$$d_i = 2\left(d_{i-1} - 7 + \frac{12}{d_{i-1}}\right). \tag{3}$$

From Equation 3 we have that when $d_0 = 12$, $d_i = 12$ for all coarse graphs. However, if $d_0 < 12$, d_i decreases whereas if $d_0 > 12$, d_i increases at each successive coarse graph.

Thus, when $d_0 = 12$, $\beta = 1$, and from Equation 1 we have that the edge-cut for 3D finite element graphs when RM is used to coarsen the graph (*i.e.*, $\delta = 1$) is

$$C_i^{3D-RM} \propto 1.15 C_{i-1}^{3D-RM} = 1.15^i C_0^{3D-RM}. \tag{4}$$

Comparing this equation with Equation 2, we see that the increase in the edge-cut at successive coarsening levels is smaller for 3D graphs than it is for 2D graphs. As it was the case with 2D graphs, the increase in the edge-cut is small, compared to the reduction in the graph size. If HEM is used to coarsen the graph, then $\delta > 1$, and the decrease in quality is even smaller.

The increase in the average degree of the graph can be controlled if coarsening is done in a certain way. In proving Equation 3, we assumed that each edge in the matching is such that the number of triangular phases incident on it is equal to the average. This

assumption is valid when RM is used to coarsen the graph which randomly selects unmatched edges. However, the HEM matching scheme tends to select edges that have a large number of triangular phases incident on them. As a result, the average degree d_i of the coarse graphs produced by HEM will be smaller [8]. In fact, our experiments show the average degrees of the coarse graphs produced by HEM not only do not increase but they actually decrease.

3.2.3 Experimental Results

To verify the analysis presented in Sections 3.2.1 and 3.2.2, we instrumented our multilevel algorithm to report various statistics during coarsening. In the rest of this section we present results for the following four matrices: 4ELT is a 2D finite element mesh; BRACK2 and WAVE are 3D finite element meshes. This set of matrices is a representative sample of the matrices shown in Table 1.

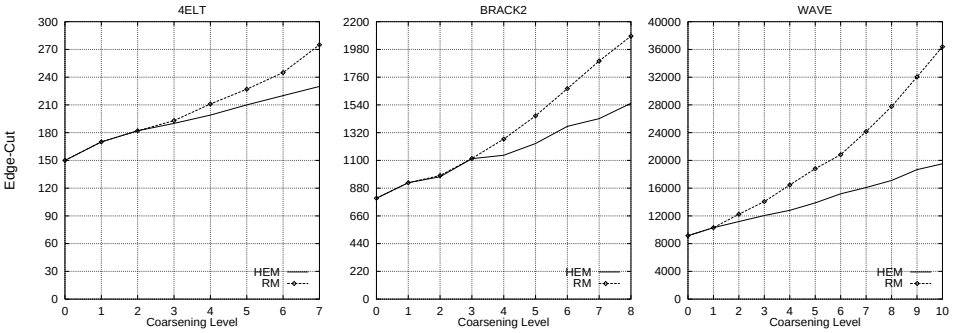

Figure 1: The increase in the edge-cut at successive coarsening levels.

Figure 1 shows the edge-cut C_i for successive coarsening levels for both the RM and HEM coarsening schemes. The edge-cut at the ith level was obtained by performing i coarsening levels to obtain G_i, and then using our multilevel algorithm to find a good partition of this coarser graph. The plotted value of C_i is the minimum of the edge-cuts produced by the multilevel algorithm and SB. From this graph we see that as indicated by our analysis, for all four matrices, the edge-cut C_i increases slowly at successive coarsening levels. When RM is used to coarsen the graphs, the edge-cut at the last coarsening level is only 1.8, 2.4, 2.9, and 4.2 times worse than C_0 for 4ELT, BRACK2, and WAVE respectively. This increase in the edge-cut is actually lower than the one predicted by the analysis. This should not be surprising since Equation 1 is only an upper bound. Also, from Figure 1 we see that when HEM is used to coarsen the graph, the edge-cuts at the coarser graphs and their rate of increase at successive coarsening levels are smaller than those of the RM coarsening scheme. This is also predicted by our analysis since for HEM, $\delta > 1$. Furthermore, for 3D finite element meshes, HEM tends to decrease the average degree of the graph with successive coarsening levels as shown in

Figure 2.

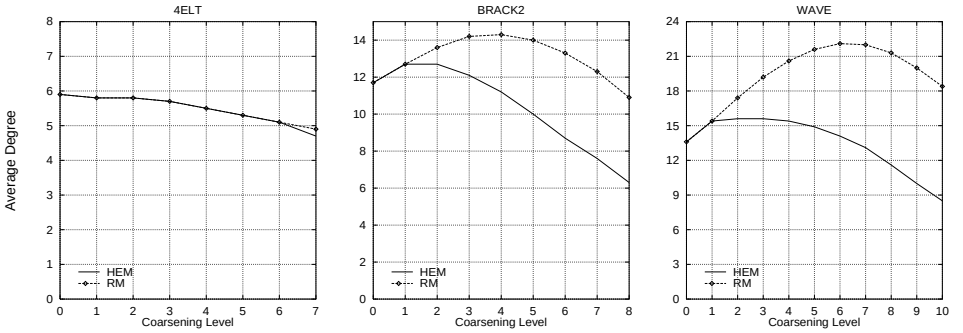

Figure 2: The average degrees at successive coarsening levels.

Figure 2 shows the average degree of the graphs at successive coarsening levels for both RM and HEM. For 4ELT, we see that for both coarsening schemes, the average degrees are similar and they slowly decrease. This decrease is due to the boundary elements of the mesh. However, for the 3D finite element meshes (BRACK2 and WAVE), the average degree of the graphs when RM is used to coarsen them, increases at successive coarsening levels. This is consistent with the analysis presented in Section 3.2.2, which indicated that the average degree of 3D finite element meshes increases when RM is used. Also, comparing the degrees of BRACK2 and WAVE, we see that the degrees increase at a higher rate if d_0 is high. This again follows directly from Equation 3. However, the increase in the average degree of the graph tappers off after a number of coarsening levels because of the boundary elements (like it was for the 2D case).

Our analysis in Section 3.2.2 also predicted that if HEM is used to coarsen the graph, then the average degree of 3D finite element meshes will increase at a slower rate than RM. This is clearly illustrated in Figure 2. The average degree of the graph increases for the 1st level coarser graph (d_1) because HEM and RM are equivalent for the first coarsening level (initially all edges have weight of one). The average degree of subsequent coarser graphs, not only do not increase but actually they decrease quite substantially. The advantages of the decreasing average degree can be seen by comparing the edge-cut of RM and HEM at the coarsest graph for BRACK2 and WAVE. For instance, at the last coarsening level, the HEM edge-cut for WAVE is only 1.8 times higher while the RM edge-cut is 2.9 times higher.

Finally, Figure 3 shows the average weight of the adjacency list of a vertex for successive coarsening levels. Recall that the average weight of the adjacency list at the ith level is equal to $d_i\omega_i$ and from Assumption 2 it is directly related to the edge-cut of the graph. From Figure 3 we see that for all four matrices, the weight of the adjacency

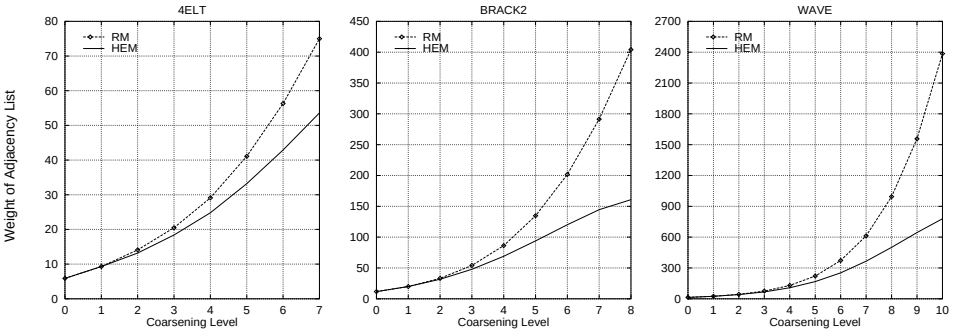

Figure 3: The average weight of the adjacency lists at successive coarsening levels. The weight of the adjacency list at level i is $d_i \omega_i$.

lists when HEM is used to coarsen the graph is smaller than that for RM.

3.3 Vertex Separator Analysis

In the previous section we showed that the edge-cut of the coarser graph is higher than the edge-cut of the original graph G_0 by a small factor. Consequently, from Assumption 2, this implies that the vertex separator induced by the bisection of the coarser graph is larger than the separator of the original graph by a small factor. In this section we analyze the coarsening process by looking at the vertex separators. In particular, we will show that the vertex separator obtained in the coarse graph is also a small vertex separator in the original graph. This analysis is focused on maximal planar graphs that satisfy the assumptions in Section 3. Furthermore, it is assumed that the separator forms a simple path or cycle [14]. This analysis can be extended to the graphs that correspond to 3D finite element meshes, when the separators are simple surfaces.

Consider the kth level coarse graph $G_k = (V_k, E_k)$. From the small separator assumption, we know that G_k has a separator S_k that contains no more than $\alpha\sqrt{|V_k|}$ vertices (recall that $\gamma = 0.5$ for planar graphs [12]). Let S'_0 be the union of the vertices of S_k projected to G_0. S'_0 forms a balanced separator of G_0 and its size is

$$|S'_0| = 2^k |S_k| \leq \alpha 2^k \sqrt{|V_k|} = \alpha 2^k \sqrt{|V_0|/2^k} = \alpha(\sqrt{2})^k \sqrt{|V_0|}.$$

However, applying the small separator assumption to the original graph, the size of the separator of G_0 is $|S_0| \leq \alpha\sqrt{|V_0|}$. In the limiting case, when $k = O(\log |V_0|)$, we have that $(\sqrt{2})^k = \sqrt{|V_0|}$, in which case $|S'_0| \leq O(|V_0|)$. Thus, S'_0 contains $O(\sqrt{|V_0|})$ more vertices than a small separator. However, not all the vertices in S'_0 are required to form a separator for G_0, and a smaller separator can be constructed from S'_0 by **dropping** some vertices.

Figure 4 illustrates the basic idea behind the processes of vertex dropping. In this figure, the 1st level coarser graph is constructed

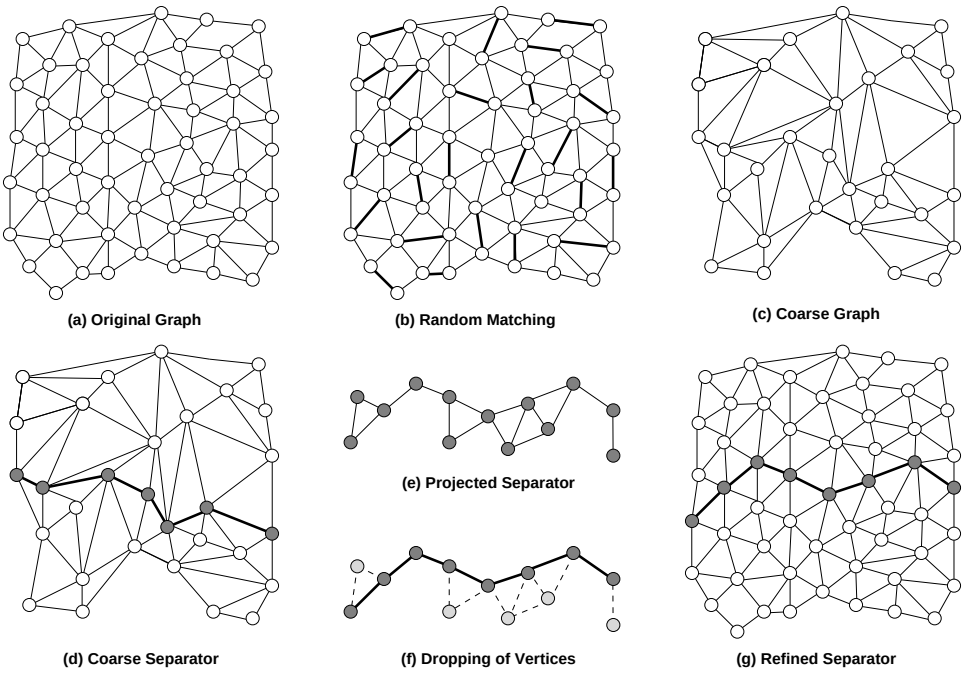

Figure 4: The sequence of one level coarsening, finding a separator for the coarse graph, projecting the separator to the original graph, and refining the separator by dropping vertices.

using RM, and a separator of this coarse graph is computed (Figure 4(d)). Figure 4(e) shows the projected separator S_0' that corresponds to S_1. Note that not all the vertices of S_0' are necessary to form a separator for G_0. As Figure 4(f) illustrates certain vertices can be dropped. In the rest of this section we compute the average number of vertices being dropped in successive uncoarsening levels.

Consider the graph G_1, and let P_1 be a simple path or cycle of G_1. Let F be the subgraph of G_0 induced by the vertices of P_1 projected onto graph G_0. The subgraph F contains $2|P_1|$ vertices, and on the average these vertices do not form a simple path or cycle.

Lemma 1 (Path Projection for Planar Graphs) *Let G_1 be a one level coarse graph obtained from G_0 using random perfect coarsening, and let P_1 be a simple path of G_1, between vertices u_1 and v_1. Let F be the subgraph of G_0 that has P_1 as its minor, and let u_0 and v_0 be the vertices of F that are mapped onto u_1 and v_1 respectively. On the average, the shortest path between u_0 and v_0 in F contains less than $1.5|P_1|$ vertices [8].*

The path projection lemma is very powerful and can be used to compute the size of the projected separator as a function of the separator of the coarser graph. Furthermore, as the next lemma shows, it can also be used to show that the sub-optimality of a separator at

a coarser graph, decreases as this separator is projected to successively finer graphs.

Lemma 2 (Simple Separator for Planar Graphs) *Let G_k be the k level coarse graph obtained from G_0 using random matching. There is a separator for G_0 whose size is bounded by $\phi 0.75^k|V_0|$ for some constant ϕ [8].*

The simple separator lemma is interesting when we consider the case in which $k = O(\log n)$. In this case, the separator for G_0 is bounded by

$$|S_0| = \phi 0.75^k|V_0| = \phi 0.75^{\log|V_0|}|V_0| = \phi|V_0|^{\log 0.75}|V_0| \leq \phi|V_0|^{0.59}.$$

Thus, even though the separator of G_k contained $O(|V_k|)$ vertices, this same separator when it is projected onto the original graph contains only $O(|V_0|^{0.59})$ vertices. Hence, if k is sufficiently large, a suboptimal separator of G_k does not significantly affect the size of the separator of the graph G_0.

3.3.1 The Nature of a Good Separator

The key element in the proof of the path projection lemma is that the edge-weight of the path in question was average. This is certainly true for any simple path but is it true for a separator path as well?

The answer to this question depends on the algorithm used to compute the vertex separator. In the multilevel algorithm, the vertex separator is computed as being the path along the boundary of the bisection. Since, the bisection is computed so that the number of edges crossing the two parts is minimized, it is not unreasonable to assume that an equal or larger amount of edge-weight does not cross the boundary. Because of this, the separator path obtained from the partition boundary should have on the average at least as much weight as any other path.

Our experimental results verify this observation. In fact, for all coarsening schemes, if we look at the number of vertices as being projected from a coarse graph to the next level finer graph, the increase in the separator size is almost always bounded by 1.5. Hence, assuming that the edge-weight of the separator path is no less than that of any other path, the following lemma is true.

Lemma 3 *The separator of G_0 obtained by projecting the separator of G_k is bounded by $\alpha(1.06)^k\sqrt{|V_0|}$ [8].*

3.3.2 Experimental Results

To verify the correctness of Lemmas 1, and 3 we performed experiments with three different planar graphs that have triangular faces. As we did throughout the analysis in this section, each vertex $v_i \in V_i$

is treated as a single vertex, irrespective of its weight. Furthermore, the separators were computed as the vertices of the first part that lay along the boundary of the bisection. That is, if V_0 is bisected into A and B, then the set $A' \subset A$ of vertices that are connected to some vertices of B is taken as the separator.

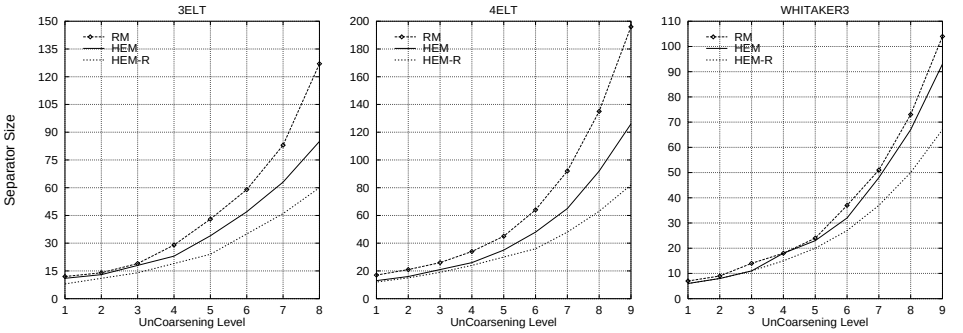

Figure 5: The increase in the number of nodes in the separator at successive uncoarsening levels.

Figure 5 shows how the size of the separator increases at successive coarsening levels. For each matrix, three curves are shown. The two of them, labeled RM and HEM, correspond to random matching and heavy-edge matching with no refinement during the uncoarsening phase, while the one labeled HEM-R, corresponds to heavy-edge with boundary greedy refinement [9]. From this graph, we see that at successive uncoarsening levels, the size of the separator increases by a factor smaller than 2. For example, when RM is used for 4ELT, going from the 7th to the 8th uncoarsening level, the separator increases from 92 to 135 vertices—an increase by a factor of 1.47. Furthermore, comparing RM with HEM, we have that HEM consistently produces smaller separators, which is not surprising, since HEM finds bisections with smaller edge-cuts (Section 3.2). Also, when boundary refinement is used (HEM-R), the size of the final separator is much smaller, and tends to increase at a lower rate.

Note that for all the graphs in Figure 5, the size of the separator increases much slower than 1.5 for the first few coarsening levels. This is because the size of the graphs during these last levels does not decrease very fast. For this reason we constructed the graphs shown in Figure 6. In this figure, for each graph and matching scheme we plotted the relative increase of the size of the separator for successive uncoarsening levels, over the size of the initial separator. Also, for each graph and matching scheme we computed the ideal relative increase so that the $\alpha\sqrt{|V_i|}$ bound is maintained [8]. Since, RM and HEM lead to coarser graphs that have slightly different number of vertices (*i.e.*, the maximal matching computed by RM and HEM are not of the same size), each matching scheme has a different ideal

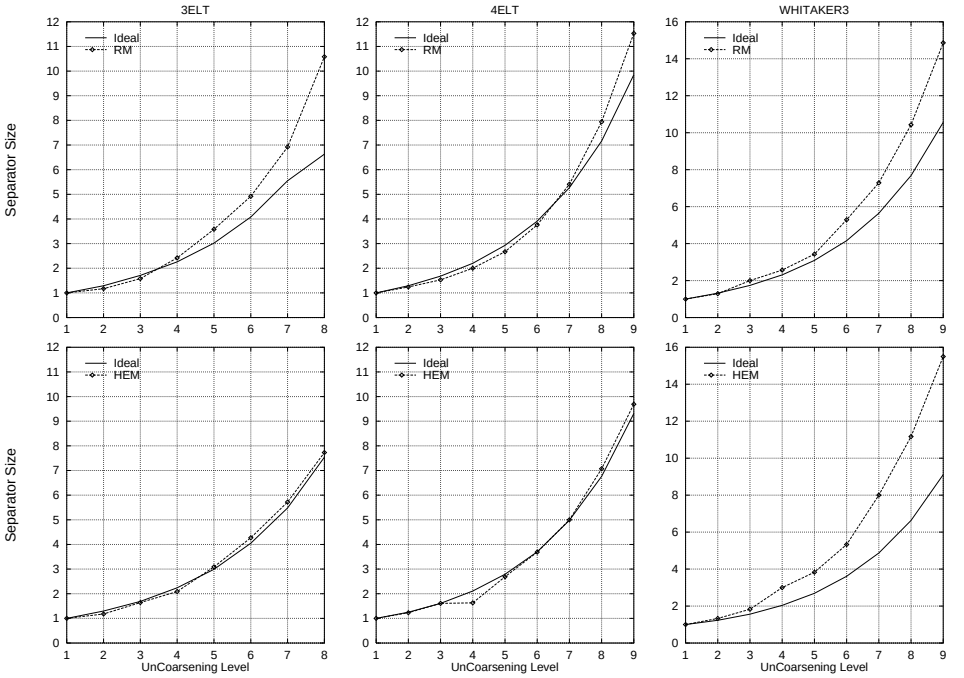

Figure 6: The rate of increase of the separator compared to the ideal increase.

curve. From these graphs we see that, the overall rate of increase in the size of the separator is worse than the ideal increase. However, the difference is usually very small. The graph for WHITAKER3 is particularly interesting, because both RM and HEM lead to a relatively high increase (a factor of two) in the size of the separator over the ideal increase. The reason for that, is that the initial separator of WHITAKER3 is actually quite small compared to the size of the graph. In particular, the coarsest graph for RM has 88 vertices while the separator has 7, and the coarsest graph for HEM has 118 vertices while the separator has only 6. Consequently, vertices cannot be dropped at the rate dictated by the ideal curve, since it will lead to separators that are contradictory small.

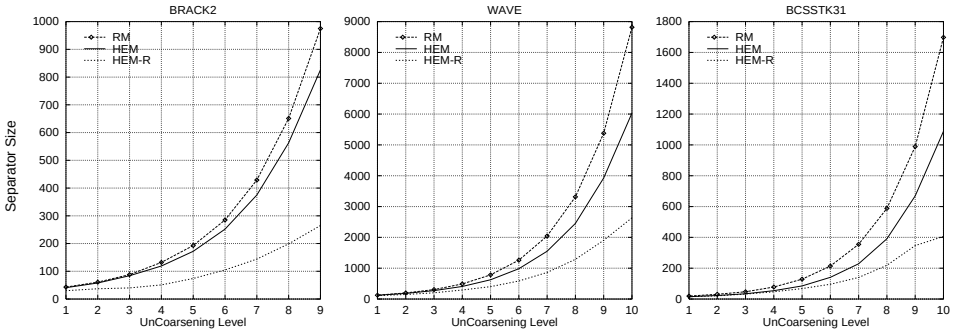

Figure 7: The rate of increase of the separator for 3D finite element meshes.

Finally, Figure 7 shows how the size of the separator increases

at successive uncoarsening levels for graphs that correspond to 3D finite element meshes. As it was the case for planar graphs, the size of the separator decreases by a factor smaller than two at each successive uncoarsening level. Also, HEM finds smaller separators at the coarsest graph, and the size of the separator increases at a slower rate than RM. Also, in the case of HEM-R, the size of the separator increases very slowly. For 3D graphs, the ideal increase of the separator size should be $2^{0.75} \approx 1.68$ at each successive uncoarsening level. From these graphs, we that the rate of increase is usually higher than that by a small factor. For instance, in the case of BCSSTK31 and RM, going from the 9th to the 10th uncoarsening level, the separator increased from 989 vertices to 1698 vertices, an increase by a factor of 1.72.

References

[1] Stephen T. Barnard and Horst D. Simon. A fast multilevel implementation of recursive spectral bisection for partitioning unstructured problems. In *Proceedings of the sixth SIAM conference on Parallel Processing for Scientific Computing*, pages 711–718, 1993.

[2] Timothy J. Barth. Aspects of unstructured grids and finite-volume solvers for the euler and navier-strokes equations. In *AGARD Report 787 on Unstructured Grid Methods for Advection Dominated Flows*, pages 6.1–6.60, 1992.

[3] T. Bui and C. Jones. A heuristic for reducing fill in sparse matrix factorization. In *6th SIAM Conf. Parallel Processing for Scientific Computing*, pages 445–452, 1993.

[4] A. George and J. W.-H. Liu. *Computer Solution of Large Sparse Positive Definite Systems*. Prentice-Hall, Englewood Cliffs, NJ, 1981.

[5] M. T. Heath and P. Raghavan. A Cartesian nested dissection algorithm. Technical Report UIUCDCS-R-92-1772, Department of Computer Science, University of Illinois, Urbana, IL 61801, 1992. To appear in *SIAM Journal on Matrix Analysis and Applications*, 1994.

[6] Bruce Hendrickson and Rober Leland. The chaco user's guide, version 1.0. Technical Report SAND93-2339, Sandia National Laboratories, 1993.

[7] Bruce Hendrickson and Rober Leland. A multilevel algorithm for partitioning graphs. Technical Report SAND93-1301, Sandia National Laboratories, 1993.

[8] G. Karypis and V. Kumar. Analysis of multilevel graph partitioning. Technical Report TR 95-037, Department of Computer Science, University of Minnesota, 1995. Also available on WWW at URL http://www.cs.umn.edu/users/kumar/papers/mlevel_analysis.ps.

[9] G. Karypis and V. Kumar. A fast and high quality multilevel scheme for partitioning irregular graphs. Technical Report TR 95-035, Department of Computer Science, University of Minnesota, 1995. Also available on WWW at URL http://www.cs.umn.edu/users/kumar/papers/mlevel_serial.ps. A short version appears in Intl. Conf. on Parallel Processing 1995.

[10] G. Karypis and V. Kumar. METIS: Unstructured graph partitioning and sparse matrix ordering system. Technical report, Department of Computer Science, University of Minnesota, 1995. Available on the WWW at URL *http://www.cs.umn.edu/users/kumar/metis/metis.html*.

[11] Vipin Kumar, Ananth Grama, Anshul Gupta, and George Karypis. *Introduction to Parallel Computing: Design and Analysis of Algorithms*. Benjamin/Cummings Publishing Company, Redwood City, CA, 1994.

[12] R. J. Lipton and R. E. Tarjan. A separator theorem for planar graphs. *SIAM Journal on Applied Mathematics*, 36:177–189, 1979.

[13] J. W.-H. Liu. Modification of the minimum degree algorithm by multiple elimination. *ACM Transactions on Mathematical Software*, 11:141–153, 1985.

[14] G. L. Miller. Finding small simple cycle separators for 2-connected planar graphs. *Journal of Computer and system Sciences*, 32(3):265–279, June 1986.

[15] Gary L. Miller, Shang-Hua Teng, W. Thurston, and Stephen A. Vavasis. Automatic mesh partitioning. In A. George, John R. Gilbert, and J. W.-H. Liu, editors, *Sparse Matrix Computations: Graph Theory Issues and Algorithms*. (An IMA Workshop Volume). Springer-Verlag, New York, NY, 1993.

[16] B. Nour-Omid, A. Raefsky, and G. Lyzenga. Solving finite element equations on concurrent computers. In A. K. Noor, editor, *American Soc. Mech. Eng*, pages 291–307, 1986.

[17] Alex Pothen, Horst D. Simon, and Kang-Pu Liou. Partitioning sparse matrices with eigenvectors of graphs. *SIAM Journal of Matrix Analysis and Applications*, 11(3):430–452, 1990.

Bibliographical Sketch of George Karypis

George Karypis received the B.S. degree in Computer Science from University of Minnesota, Minneapolis, in 1992. College Park, in 1982. He is currently pursuing a Ph.D. degree in Computer Science at the University of Minnesota. His current research interests include parallel algorithms for scientific computing, communication libraries, and scalable parallel libraries. He is author of over 10 research articles, and is a coauthor of a text book “Introduction to Parallel Computing" (Publ. Benjamin Cummings/Addison Wesley, 1994). Recent publication can be found at **http://www.cs.umn.edu/~karypis**.

Bibliographical Sketch of Vipin Kumar

Vipin Kumar received the B.E. degree in electronics & communication engineering from University of Roorkee, India, in 1977; the M.E. degree in electronics engineering from Philips International Institute, Eindhoven, Netherlands, in 1979; and the Ph.D. degree in computer science from University of Maryland, College Park, in 1982. He is currently a Professor of Computer Science at the University of Minnesota. His current research interests include scalable parallel computing and artificial intelligence. He is author of over 80 research articles on the above topics, and is a coauthor of a text book “Introduction to Parallel Computing" (Publ. Benjamin Cummings/Addison Wesley, 1994). Kumar has chaired many workshops and served on the program committee of many conferences in parallel processing. He is Vice Chair for Applications Area for the 9th International Parallel Processing Symposium. Kumar serves on the editorial boards of IEEE Parallel and Distributed Technology, IEEE Transactions of Data and Knowledge Engineering, and the Journal of Parallel and Distributed Computing. **http://www.cs.umn.edu/~kumar**.