Mesh Partitioning Techniques and New Observations for 3-regular Graphs

Fabio Guerinoni

INRIA

INSTITUT NATIONAL DE RECHERCHE EN INFORMATIQUE ET EN AUTOMATIQUE

Mesh Partitioning Techniques and New Observations for 3-regular Graphs

Fabio Guerinoni

N° 2623

Juillet 1995

PROGRAMME 6

ISSN 0249-6399

Mesh Partitioning Techniques and New Observations for 3-regular Graphs

Fabio Guerinoni*

Programme 6 — Calcul scientifique, modélisation et logiciel numérique
Projet Aladin

Rapport de recherche n°2623 — Juillet 1995 — 19 pages

Abstract: We describe in detail some algorithms currently in use for unstructured mesh partitioning, with some emphasis on spectral methods, that is, those methods which involve eigenvector computations. When applied to 3-regular graphs, previous methods can be theoretically improved, should a stated conjecture proves true.

Key-words: unstructured meshes, graphs, partitioning, bisections

(Résumé : tsvp)

*Fabio.Guerinoni@irisa.fr

Unité de recherche INRIA Rennes
IRISA, Campus universitaire de Beaulieu, 35042 RENNES Cedex (France)
Téléphone : (33) 99 84 71 00 – Télécopie : (33) 99 84 71 71

Partitionnement des maillages et nouvelles observations pour les "graphs 3-réguliers"

Résumé : Nous décrivons queques algorithmes courants pour le partitionnement des maillages non structurés. L'accent est mis sur les méthodes spectrales, demandant le calcul des vecteurs propres. Appliquées aux "graphes 3-réguliers," une conjecture est proposée permettant d'améliorer en théorie les algorithmes précédents.

Mots-clé : maillages non structurés, graphes, partitionnement, bisection

1 Introduction

The present article is intended to give an up to the minute survey of the most important methods for the mesh partitioning problem: given an unstructured mesh and a distributed memory system with P processors, find P disjoint subsets of vertices of the mesh of about the same size (a *balanced partition*) such that the communication between the processors is minimal.

In addition, we have made a study of the so-called *spectral* methods when applied to $3-$*regular* graphs. These studies lead to interesting conjectures and at the same time opens some areas of research in numerical linear algebra, combinatorial optimization, and applications to unstructured mesh generation and other fields..

The problem is a well known problem in graph theory [7] which is intrinsically difficult: there are not known algorithms to solve it *exactly* in polynomial time. Moreover, if one could find such an algorithm, a number of such problems would also be solved in polynomial time.

Especially from the graph-theoretical point of view, the literature is vast and complex. Most of the issues discussed here have been taken from papers from the application-oriented point of view, mainly for parallel computations on unstructured meshes. The papers [9], [2] provide a fairly balanced list of references.

As is often the case, discrete problems are usually harder than their continuous counterparts. For the problem of *mesh* partitioning (as opposed to *graph* partitioning) good results can be obtained by the introduction of metrics. However, a class of purely graph-theoretical methods called *spectral bisection* has gained popularity in the last years. We are particularly interested in them, so they will receive more attention. As a matter of fact, in this survey we classify these methods as "spectral" and "non-spectral."

All of the algorithms surveyed here use, or can be adapted to, *recursive bisection*. That is to find a partition with P subsets, one applies bisection (balanced partition in two subsets) $\log_2 P$ times. In Algorithm 1, this recursive bisection procedure is called *recbisect(G,P)* and the bisection algorithm is *bisect(G,*V_1, V_2*)*. The V_1, V_2 represent the two vertex subsets (output). The cut size (measuring comunication between processors) will be denoted K throughout this article (see Appendix for that and other notational conventions).

ALGORITHM 1: Recursive Bisection

recbisect(G, P)
(* G graph, P number of subsets *)
  if $P = 1$ **then**
    return V
  else
    call bisect (G, V_1, V_2)
    $P_1 = \lfloor P/2 \rfloor,\ P_2 = \lceil P/2 \rceil$
    call recbisect (V_1, P_1)
    call recbisect (V_2, P_2)
  endif
return

There are some inconveniences with the use of recursive bisection. First, it may fail significantly when P is not a power of two: if $P = 3$ the resulting partition will not be equally balanced. Second, the graph partitioning problem cannot be solved by a divide and conquer strategy: an optimal 4-partition is not the result of twice bisecting a graph, even if each bisection is optimal.

In spite of this observation, mentioned before the algorithms considered here are of this type, that is, they are all *bisection* based. In other words, we will describe with some detail the diverse forms of the routine *bisect*(G, V_1, V_2).

To simplify the description of some of the bisection algorithms, we will use the concept of *separator field* definition introduced by Williams [15], used by all procedures. Each algorithm gives a scalar quantity σ_i for each vertex v_i, and from there the bisection proceeds as in Algorithm 2. The procedure to compute the *median* Ω (that is the $\lfloor p/2 \rfloor$ largest σ_i) takes $O(p \log p)$ and only half the time as a full good sort. However, "quasi" linear algorithms exist, see for example [6].

ALGORITHM 2: Divide by median

divmed(G, σ, V_1, V_2)
(* G graph, σ separator field *)
find the median Ω of $\{\sigma_i\}$ **then**
$V_1 = \emptyset$; $V_2 = \emptyset$
for $i = 1, p$ **do**
if $\sigma_i \leq \Omega$ **then**
$V_1 = V_1 \cup \{v_i\}$
else
$V_2 = V_2 \cup \{v_i\}$
endif
enddo
return

2 Non-spectral algorithms

The first algorithm, the Kernighan-Lin algorithm, as described here does not compute a bisection (although it can be easily modified to do so) but rather *improves* a given bisection. As will be discussed later, it is used often after a bisection has been computed by other means.

The second algorithm for bisection uses metrics or imbeddings for partitioning. In this respect, it is different from all the others algorithms considered here.

2.1 The Kernighan-Lin Algorithm

The Kernighan-Lin (KL) algorithm is a procedure to *locally* improve the size of the cut K. It was developed in the early 70's but it has become a fundamental "textbook" graph algorithm [8].

Suppose we have a bisection (V_1, V_2) with cut size K. For any vertex a let $D(a)$ be the 'out-degree' (edges going outside of its subset) minus the 'in-degree' (edges within its subset). Moving a to the other partition results in a new cutsize.

$$K' = K - D(a)$$

and thus exchanging two vertices a and b, in V_1 and V_2 respectively, results in

$$K' = K - (D(a) + D(b) - 2d_{ab})$$

where d_{ab} is the number of edges between a and b. (the algorithm is valid for multigraphs also). Thus if we define the *gain*

$$g(a, b) = D(a) + D(b) - 2d_{ab},$$

it is in theory posible to compute $g(a, b)$ for each pair of vertices in opposite subsets in $O(n^2)$ and select the 'swap' that maximizes g. However, it might be the case that there is no positive g. The KL algorithm suggests a way to escape such local minima, by repeatedly considering the least worth option.

The algorithm is based in the following observation. Suppose that a and b have been exchanged. A new D is computed as follows

$$\begin{aligned} D'(x) &= D(x) + 2d_{xa} - 2d_{xb} \quad x \in A - \{a\} \\ D'(y) &= D(y) + 2d_{yb} - 2d_{ya} \quad y \in B - \{b\} \end{aligned}$$

To see this, if x is connected to leaving node a, its out-degree will increase and its in-degree will decrease by the same amount, resulting in a total increase of D by $2d_{xa}$. A similar situation occurs for a entering node b.

The KL algorithm calculates a sequence of swaps $a_1, b_1, a_2, b_2 \ldots a_n, b_n$ and corresponding gains $g_1, g_2 \ldots g_n$, finds the index k of the *total* maximum gain

$$\max_{1 \le i \le n} G_i = \max_i \sum_1^n g_i$$

If this gain is positive, perform the k swaps which have resulted in the maximum total gain. The procedure is repeated several times until no swaps are made.

ALGORITHM 3: Kernighan-Lin

kerlin(G, V_1, V_2)
(* Given a bisection (V_1, B) each subset of size n and cut size K, *)

(* the algorith constructs another bisection with cutzize $K' \leq K$ *)
$A = V_1$; $B = V_2$
find $a \in A, b \in B$ such that $g(a,b)$ is max.
$g_1 = g$; $G_1 = g$
$a_1 = a$; $b_1 = b$;
$A = A - \{a\}$; $B = B - \{b\}$;
for $j := 2$ **to** n **do**
for $x \in A, y \in B$ **do**
$D(x) = D(x) + 2d_{xa} - 2d_{xb}$;
$D(y) = D(y) + 2d_{yb} - 2d_{ya}$;
enddo
find $a \in A, b \in B$ such that $g(a,b)$ is max.
$g_j = g$; $G_j = G_{j-1} + g$;
$a_j = a$; $b_j = b$;
$A = A - \{a\}$; $B = B - \{b\}$;
enddo
(* total gains have been calculated *)
find k such that G_i is maximum
if $G_k > 0$ **then**
for $i := 1$ **to** k **do**
$V_1 = V_1 - \{a_i\}$; $V_2 = V_2 - \{b_i\}$;
$V_1 = V_1 \cup \{b_i\}$; $V_2 = V_2 \cup \{a_i\}$;
enddo
(* The new cutsize is $K' = K - G_k$ *)
endif
return

2.2 Principal Inertial Axis Algorithms

Due to its physical nature, to our knowledge this algorithm is only applied to meshes. Indeed, its application to graph-theoretical problems, depends on the *embedding* of the graph into some manifold. However, it is possible to use the algorithm, by embedding the graph in a one dimensional discrete space according to the graphical distance to some reference node. The resulting algorithm (after the recursive construction) has been called *recursive graph bisection* RGB [12]. We will mention it briefly in the last two sections.

The concept comes from elementary solid mechanics; see for example [1], and works very well for two dimensional, uniform meshes. We consider a mesh to be a solid object S with density $\rho(\vec{x})$, positioned with respect to an orthonormal reference system. If we represent an axis H through the origin as a unitary vector h, the *moment of inertia* I of the solid S can be written as a quadratic form

$$I(S,h) = h^T I(S) h$$

The matrix $I(S)$ is called an inertia matrix; it is symmetric and definite positive. The diagonal entries are the momentums of inertia with respect to the reference axis

$$\begin{aligned} I_{11}(S) &= \int_S (y^2 + z^2)\ \rho(\vec{x})\ d\vec{x} \\ I_{22}(S) &= \int_S (z^2 + x^2)\ \rho(\vec{x})\ d\vec{x} \\ I_{33}(S) &= \int_S (x^2 + y^2)\ \rho(\vec{x})\ d\vec{x} \end{aligned}$$

The non diagonal terms are the products of inertia

$$\begin{aligned} I_{12}(S) &= -\int_S (yz)\ \rho(\vec{x})\ d\vec{x} \\ I_{13}(S) &= -\int_S (zx)\ \rho(\vec{x})\ d\vec{x} \\ I_{23}(S) &= -\int_S (xy)\ \rho(\vec{x})\ d\vec{x} \end{aligned}$$

The scalar $I(S, h)$ measures the inertia (opposition to change in angular momentum) from the object S when torsion is applied along the axis H. From the properties of the matrix I(S), we know that there are j_1, j_2, j_3 orthonormal vectors with $I(S)\ j_i = \alpha_i j_i$, and α_i can be assume to be in descending order. Furthermore, in view of the Courant-Fischer minimax principle , we have

$$\alpha_3 = \min_{||h||=1} h^T I(S) h$$

and the minimum is obtained at $h = j_3$. It is easy to prove that this optimal h is invariant under a translation of the reference system. However, the value itself is not: it is minimum only if we take a system with origin at the center of mass $\vec{c}$

$$\begin{aligned} c_x &= (1/M) \int_S x\ \rho(\vec{x})\ d\vec{x} \\ c_y &= (1/M) \int_S y\ \rho(\vec{x})\ d\vec{x} \\ c_z &= (1/M) \int_S z\ \rho(\vec{x})\ d\vec{x} \end{aligned}$$

with M being defined as the integral of the density.

Be as it may, the eigenvector j_3 provides what will be the "longest" axis for a uniform object. It can be computed by any elementary procedure, such as power or Rayleigh iteration. Then the algorithm proceeds as expected: project into the axis and bisect the find the bisection by sorting. The procedure is described in Algorithm 4.

ALGORITHM 4: Principal Inertia

pi(G, V_1, V_2)
(* each vertex v_i must have a 3-dimensional embedding $\vec{x}_i = (x_i, y_i, z_i)$ *)
$M = p$; $c_x = 0$; $c_y = 0$; $c_z = 0$
(* center of mass *)
for $i := 1$ **to** p **do**
$c_x = c_x + x_i$; $c_y = c_y + y_i$; $c_z = c_z + z_i$;
enddo
$\vec{c} = \vec{c}/M$;
(* initialize inertia matrix *)
$I = 0$;
(* momenta and products of inertia *)
for $i := 1$ **to** p **do**

$$I_{xx} = I_{xx} + (y_i - c_y)^2 + (z_i - c_z)^2 \; ; \quad I_{xy} = I_{xy} - (y_i - c_y)(z_i - c_z)$$
$$I_{yy} = I_{yy} + (z_i - c_z)^2 + (x_i - c_x)^2 \; ; \quad I_{yz} = I_{yz} - (z_i - c_z)(x_i - c_x)$$
$$I_{zz} = I_{zz} + (x_i - c_x)^2 + (y_i - c_y)^2 \; ; \quad I_{zx} = I_{zx} - (x_i - c_x)(y_i - c_y)$$

enddo
$I_{yx} = I_{xy}$; $I_{zy} = I_{yz}$; $I_{xz} = I_{zx}$;
find eigenvector j of least eigenvalue
for $i := 1$ **to** p **do**
$\sigma_i = j \; j^T \vec{x}_i$
enddo
(* the projections σ are the separator field *)
call divmed(G, σ, V_1, V_2)
return

3 Spectral Algorithms

3.1 Spectral Bisection

The use of eigenvectors for graph-theroretical computation can be traced back to more than twenty years, but its revival as a useful tool for partitioning unstructured meshes is due to Pothen *et al* [9] and Simon [12].

The core of the algorithm is the use of the *Laplacian* L of the graph to compute a characteristic valuation, as described in the Appendix. Subsets of corresponding vertices of such characteristic valuations (used as a separator field) preserve the original connectivity of the graph.

In general such schemes and variations are more expensive than others, but they seem to produce better results [5], [9]. More on this subject on Section 5.

The key issue in the spectral algorithm, is the eigenvector computation. Lanczos method has been favored, nonetheless the convergence is slowed because the smallest eigenvalue is zero. Shifting and deflation techniques are then used.

ALGORITHM 5: Spectral Bisection
sb(G, V_1, V_2)
(* A is the adjacency matrix of a connected graph G *)
(* D diagonal matrix, with entries the sum of edge-weights *)
$L = D - A$
find second eigenvector x_F using (modified) Lanczos method
(* the components x_F are the separator field *)
$\sigma = x_F$
call divmed(G, σ, V_1, V_2)
return

3.2 Applications to 3-regular graphs

This subsection goes a little beyond the expository aspects of the paper and presents new observations on the issue on restricted class of graphs.

Three-regular graphs have been well studied in combinatorial optimization. For example, their number (up to certain order) has been determined [11]. Thus, it is computationally feasible to analyze carefully the performance of the spectral bisection.

We introduce some minimal notation necessary to describe the results. The Appendices contains additional or complementary information.

A **bisection** is a member of the set

$$B = \{\mathbf{1}^T\mathbf{p} = 0, p_1 = -1, p_i = \pm 1\}$$

notice that B is a subset of the $(n-1)$-subspace $\mathbf{1}^\perp$. The orthogonality relation in the definition is fundamental, and explains why not a binary $\{0,1\}$ notation is employed. The **cut size** induced by $\mathbf{p}$ is denoted by E_p and if E^* is the "optimal" cut size we have

$$E^* = \min_{p \in B} \frac{\mathbf{p}^T L \mathbf{p}}{4} = \frac{\mathbf{p}^{*T} L \mathbf{p}^*}{4}$$

Thus, the problem of graph bisection can be stated as that of finding $p^* \in B^*$, where $B^* \subset B$ is the set of optimal bisections.

Eigenvectors of L for a regular graph are those of the adjacency matrix, and thus Fiedler's Theorem applies (see Appendix 1) and hence we can say something about the connectivity, but nothing about bisection. On the other hand Pothen [9] suggested the sorting procedure to approximate a bisection.

The difference between Pothen's and Fiedler's criteria is that the first one requires the computation of the median, which as we said before can be done faster than $O(n \log n)$. Fiedler's can be thought of requiring $O(1)$.

Additionally, we have proposed and algorithm, which represents and improvement over both of the above if Conjecture 1 is true. So far, we have not found a counterexample, but we have identified a possible candidate.

We define a *separation by median* function: if Ω is the median (the $\lfloor n/2 \rfloor$ largest x_i)

$$S(x_i) = \begin{cases} 1, & \text{if } x_i \geq \Omega \\ -1, & x < \Omega \end{cases} \qquad S(x_i) \in B.$$

where x_i are the components of an (eigen)vector.

After computing a characteristic valuation, x_a, the Fiedler's connectivitiy criteria approximates the space B^* with

$$\bar{x}_a = \text{sign}(x_a)$$

while Pothen's with

$$\bar{x}_a = S(x_a)$$

Neither of the above is well defined. In the first, case our approximation $\bar{x}_F$ may not even be in B due to the presence of 0's (this situation has been analyzed in [10]) and in the second case the separation by median is not unique.

Now we are ready for the conjecture:

Conjecture 1 : For any even order,connected, 3-regular graph there exists a characteristic valuation x_a such that

$$\exists p^* \in B^*, \quad p^* = S_k(x_a)$$

where S_k is a separation by median function.

Using an implementation an in Mathematica to generate regular graphs[13], Conjecture 1 has been verified for all connected, 3-regular graphs of even order $n \leq 10$, except *one*.

The graph shown prove to be very special, not only for possibly "failing" Conjecture 1 , but due to the exceptional difficulty in generating it : It took about 45 minutes on a SUN 10 workstation. After that the program has encountered difficulties with imaginary values arising during the QR decomposition. Other interesting facts are its reduced Laplacian spectrum consisting of $0, 2, 5$, and the unusual number of characteristic valuations.

Of course, if we can avoid any median computation we are much better off. Our experiments show that this might work in some particular cases, as in the following conjecture verified for $n \leq 10$.

Conjecture 2 : For an even order, connected, 3-regular graph with $|B^*| = 1$ then

$$p^* = \text{sign}(x_a)$$

Perspectives on Proposed Algorithm

Table 1 shows all connected 3-regular graphs including the "black sheep" (Graph 20). Graph 1 is not connected but it is included nonetheless. The approximation methods are graded as follows: if $\bar{x}$ is the approximation, we compute

$$\text{grade}(\bar{x}) = \min_{p \in P*} ||p^* - \bar{x}||_1$$

This quantity has the desired property that

$$\text{grade}(\bar{x}) = 0 \Longleftrightarrow \bar{x} \in P^*$$

and since in most cases (except Fiedler's) $\bar{x} = \pm 1$

$$\text{grade}(\bar{x}) = 2(\text{number of misses})$$

The improved algorithm searches for each x_a, in a straightforward way, for other suitable S_k, guaranteed by Conjecture 1.For these particular cases we have succeeded quickly, as shown on the last line of the table.

Unfortunately, the algorithm is not polynomial. If x denotes the values taken by a vertex under a characteristic valuation, we call *the clustering* $cl(x)$, the number of times it occurs. The clustering property, proved in the Appendix, provides some intuition about its meaning. Ideally one would like to have S_k explicitly in terms of spectral information about L, but at this stage of the research, we will be satisfied in finding or stating sufficient conditions.

Graph	1*	2	3	4	5	6	7	8	9	10
Opt. Cuts	6	1	2	2	2	2	1	1	1	12
Size	3	1	3	3	3	3	3	3	3	5
Alg. Conn	0	0.22	0.43	0.56	0.58	0.58	0.85	0.88	0.92	1
Multp.	1	1	1	1	1	1	1	1	1	1
Fiedler's	6	0	2	2	2	2	0	0	0	4
Pothen's	0	0	0	0	0	0	0	0	0	0
Improved	0	0	0	0	0	0	0	0	0	0

Graph	11	12	13	14	15	16	17	18	19	20
Opt. Cuts	1	13	12	15	11	11	10	9	8	6
Size	3	5	5	5	5	5	5	5	5	5
Alg. Conn	1	1.06	1.10	1.12	1.12	1.38	1.38	1.38	1.44	2
Multp.	1	1	1	2	1	2	2	1	1	5
Fiedler's	0	0	4	3	3	2	2	2	2	6
Pothen's	0	0	4	0	4	0	0	0	0	4
Improved	0	0	0	0	0	0	0	0	0	**?**

Table 1: 3-regular connected graphs of order 10 (* not connected)

A sufficient condition for computational feasibility of improved algorithm is the following. Let k a constant independent of n such that

$$cl(\Omega) \leq k \qquad \forall x_a$$

then the improved algorithm is feasible. Clearly, then any S function may miss at most $O(K)$ times. To find p^* one must check a combinatorial function of $K/2$ times to find an optimal solution.

3.3 Multilevel Spectral Bisection

Very recently, Barnard and Simon [2] have devised a variation of spectral bisection which provides roughly and order of magnitude improvement in speed to compute a bisection. The idea is based on a faster computation of the characteristic valuation which they call a *Fiedler vector*.

For that purpose, the original mesh (or graph) is contracted according to Algorithm 6, which involves the computation of a maximal independent set W. For a general graph it can be shown that the contraction preserves connectivity. This algorithm is an exception in the context of this paper: it does not compute a bisection, but another graph.

Suppose we have the characteristic valuation x_F for a contracted graph. This value is extrapolated to the original graph by 'averaging' for no overlapping vertices. Values for overlapping vertices (values in the original maximal independent set W) are exactly the same as their corrresponding x_F.

The obtained value is taken as the initial value of a Rayleigh quotient iteration to converge to an eigenpair of the Laplacian L of original mesh. Since the extrapolated value was an approximation to it, the procedure converges to a characteristic valuation of the original mesh, from which the partition is obtained using divmed(G, σ, V_1, V_2).

ALGORITHM 6: Graph Contraction

contr(G, G')
(* Original graph is $G = (V, E)$ *)
(* Contracted graph is $G' = (W, F)$ *)
find a max. ind. set W
$F = \emptyset$; $S = \emptyset$;
for $v \in W$ **do**
$N(v) = \{v\}$
$S = S \cup \{v\}$
enddo
while $S \neq V$ **do**
for $x \in W$ **do**
(* $n(x)$: adjacent vertices to x *)
$N(x) = N(x) \bigcup_{y \in N(x)} n(y)$;
$S = S \cup \{v\}$
enddo
for $(i, j) \in W \times W - F$ **do**
if $N(i) \cap N(j) \neq \emptyset$ **then**
$F = F \cup \{(i, j)\}$
endif
enddo
enddo
return

3.4 Variations for dynamic Meshes

How to compute partition for meshes changing in time as when using adaptive refinement? A solution has been proposed by Van Driessche and Roose [14]. Instead of the Laplacian eigenvalue problem of the standard spectral bisection they propose solving

$$Lx = \lambda Dx,$$

or equivalently

$$D^{-1/2}LD^{-1/2}z = \lambda z \qquad z = D^{1/2}x$$

which has provably eigenvalues λ in the interval $[0, 2]$. According to the authors, bisections using this modified eigenproblem are superior to those computed by the characteristic valuation.

For the simple example that they propose, the modified approach yields an optimal bisection while the original algorithm slightly fails. This might be a problem-dependent observation, and may not guarantee a generalization. More interesting is the absence of the "clustering effect"(Appendix 2), which can be observed in the original problem. Such effect makes impossible to separate the characteristic valuation components as the graphs disconnects. It seems that the new algorithm prevents this effect. These and related issues are currently under investigation.

In any case, the "preconditioned" eigenvalue problem is simple enough to implement from the original spectral bisection so as to readily allow further scrutiny.

Besides being more general than the original spectral bisection algorithm as originally formulated (it uses arbitrary edge-weights), the method leading to the modified Laplacian, uses a *statistical* technique. Such technique is, perhaps, a better instrument for understanding spectral bisection, intermediate between heuristics and formal approaches.

Furthermore, the technique allows for the concept of *hypergraphs* in which special edges may join three or more vertices. Hypergraphs are used for the problem of changing meshes. A discussion of hypergraphs as well as the statistical formulation and their properties is beyond the scope of this report.

4 Software Packages

We describe one of them, which contains most algorithms describe here (the one in section 3.3 being the exception). The package is completely written in $C++$.

4.1 TOP/DOMDEC

The NASA developed software package TOP/DOMDEC [3] presents a synthesis of mesh partition algorithms. Besides the well known Reverse Cuthill-McKee for bandwidth minimization and its variations, the greedy technique, it incorporates the following mesh partitions algorithms

* **Principal Inertia Algorithms:** this algorithms have proven fast, simple and effective. The idea is to partition the mesh based on the principal axis of inertia as described in Section 2.2. Also included as special cases RGB and the possibility to define the projection axis arbitrarily (coordinate bisection).

* **Recursive Spectral Bisection:** including multilevel strategy.

* **Others:** greedy strategies for partitioning, the reverse Cuthill-Mckee (for bandwidth minimization rather than partition) and topological algorithms which guarantee certain neighborhood restrictions on the subsets.

Three optimization algorithms are included for post-processing of the partitioned mesh, which allows the user to fine tune the output. In addition, the user can control a number of parameters of the partition, a very useful feature in view of the conclusion of this survey in the next section.

5 Observations and Needs

Which is a good algorithm for *mesh* partitioning? The research issue has been active for five years (at least) and there are perhaps about twenty papers on the subject, but still there is not a clear answer. For the same reason, the authors of TOP/DOMDEC have included so many options within their package. In this section we will present what we have identified as the main problems.

The first problem is the different *types* that a mesh may assume. This type bears no relation to the geometry of the problem, to be specific we will refer to a 2-dimensional airfoil with a few components. Unfortunately this issue is not clearly emphasized in any of the papers which show timings for the algorithms.

To illustrate the point consider the two following tables. They represent results for computing a 16-partition with the basic spectral algorithms, perhaps the only published results with timings (although omitted in this discussion). The first set of data comes from Hu and Blake [5] and the second from Barnard and Simon [2].

$\|V\|$	K	$K/\|V\|$	$K/\|E\|$
788	108	0.137	0.091 *
3564	261	0.073	0.048 *

$\|V\|$	K	$K/\|V\|$	$K/\|E\|$
4720	758	0.160	0.055
15606	1330	0.085	0.028
36519	3091	0.084	0.021

First notice that the size of the problems considered by the two authors is complementary. Hu has tested his problems on small meshes and the other authors on large meshes. The notation is consistent with that of the Appendix 1.

The problems are of the same 'airfoil' type. Observe the cut vertex ratio of the last mesh on the left and the first one on the right. It seems that in the latter size of the cut is proportionaly too large (0.160) with respect to the value of mesh not much different in size

(0.073). The explanation lies precisely on the 'type' of mesh on which the algorithms were tested.

There are basically three mesh type: cell-centered (also called dual), vertex-centered and true-comunication. They correspond respectively to finite volume, finite element and hybrid numerical solvers in fluid dynamics. While they all may represent the same physical mesh, their graph is entirely different. Some papers in the literature do not specify clearly the type and thus make drawing conclusions difficult.

This explains the previous observation. The first set of data refers to a cell-centered mesh, the second to a vertex-centered scheme. In two dimensions there is at list twice as many vertices per edge in the first case. By using edge numbers (estimating when data is missing) one obtains more consistent values as the size of the problem increases, These are provided in the fourth column of the tables. When edge numbers data were estimated to compute the value, it is marked with an asterisk.

The second problem is the difficulty of evaluating a partition. Most papers evaluate a partition by the cut size. By this measure, observations lean toward spectral bisection and variations (sections 3.1-3.2), and leave out other aspects such as load balance. In [5], a combination of RGB (see section 2.2) and Kernighan-Lin (section 2.1) is reported as very competitive as well, but only cut size is the evaluation parameter. Fortunately, the importance of load balance is beginning to be considered within the framework of dynamic meshes [14] and software implementations [3].

The third problem is the evaluation of the algorithm itself. There is no need to comment further on this issue but to pose the eternal question in numerical analysis: What is better: an algorithm that computes an approximation of the solution fast or an algorithm which computes a good solution but slow?

Unfortunately, there is not much information yet on the multilevel techniques, but they may be promising. As discussed in Section 3, in this paper an important issue seems to be the contraction of the mesh, which in " in some sense preserves the global structure of the mesh," reduces the size and thus the cost of the eigenvector computation. But it is not clear how the proposed algorithm differs from from a faster eigenvalue computation achieved by a multigrid effect.

To summarize, in order to decide on a partioning algorithm one must have

1. A good *mesh classification* system, broader than the concept of type proposed here, which should include issues like, geometry, scale, connectivity etc.

2. A *definition* of what constitutes a good partition. This may be application dependent. In some cases balance may be more important than communication in other cases, the other way around.

3. *Quantitative* information about the algorithms, intervals of reliability and accuracy. Especially with respect to spectral algorithms and its relation with matrix theory, much can be done in this respect in the upcoming years.

Ackowledgements

I am grateful to Jocelyne Erhel and Rémi Choquet for providing me with material and resources to complete this survey.

Appendix 1: A quick account of Fiedler's theory

Let $G = (V, E)$ be a directed graph of order $p = |V|$ (number of vertices) and size $q = |E|$ number of edges. If we define the $q \times p$ *incidence matrix*

$$c_{ij} = \begin{cases} 1, & \text{vertex } j \text{ is head edge } i \\ -1, & \text{vertex } j \text{ is tail edge } i \\ 0, & \text{otherwise.} \end{cases}$$

and the weight matrix

$$\operatorname*{diag}_{1 \le k \le q} w_k \quad 0 \le w_i$$

The Laplacian L of G is defined

$$L = C^T W C$$

Equivalently, we may write $L = D - A$ where D is diagonal, and A is the weighted adjacency matrix. The following properties can be easily obtained from the definition:

* L is independent of the direction of the edges. In consequence the directed incidence matrix can be replaced by an undirected incident matrix.
* L is an M_0 matrix, that is nonnegative definite with nonnegative diagonal, and with nonpositive off-diagonal entries. $D + A = L + 2A$ is nonnegative definite.

Not so immediate but interesting nonetheless is the following

* (Matrix-Tree Theorem) If $W = A$ and G is connected then all cofactors of L are equal and their common value is the number of spanning trees

Let $K = (V_1, V_2)$ a partition of the set of vertices V, with cut size $|K|$ and p a vector such that $p_i = 1,\ i \in V_1,\ p_i = -1,\ i \in V_2$, the following is also an easy consequence of the definition of Laplacian.

$$|K| = \frac{p^T\ L\ p}{4}$$

The rank of the weighted incidence matrix $W^{1/2}C$ is related to the structure of G; if G has k components and no edge-weight is 0, we have

$$\begin{aligned} \text{rank } L &= \text{rank } (W^{1/2}C)^T (W^{1/2}C) \\ &= \text{rank } (W^{1/2}C) \\ &= p - k \end{aligned}$$

We define λ_F as the value of the second largest eigenvalue (out of p of them, counting repetitions) and x_a a corresponding eigenvector (which in general is not unique) x_a is called a *characteristic valuation* although some authors have recently favored the name Fiedler's vector.

When G is connected, the minimum eigenvalue is always 0 and $\lambda_F > 0$. The basis for spectral bisection is the following theorem.

Theorem (Fiedler [4]) Let L be the Laplacian of a connected graph. and $y = x_a$ a characteristic valuation. Then, for any $r \geq 0$ each one of the following sets is connected

$$\mathcal{P} = \{i \in V \ / \ y_i \geq -r\} \quad \mathcal{O} = \{i \in V \ / \ -r \geq y_i \leq r\} \quad \mathcal{N} = \{i \in V \ / \ y_i \leq r\}$$

Appendix 2: The clustering property

Let G be a connected graph of order $2p$ with vertex set V. Construct a new graph of order $p+2$ as follows

1. Bisect the graph in V_1, V_2, each vertex set of order p.
2. Add two vertices v_{p+1}, v_{p+2}
3. Connect every node of V_i with vi_{p+i} with an edge of weight $z > 0$

The resulting graph G_z has order p+2 and size $q + 2n$. Its Laplacian will be called L_z and $\lambda_F^{(z)}$ the second largest eigenvalue of L_z

Proposition Suppose that

$$\lim_{z->a} L_z = \hat{L}$$

then

$$\lambda_F^{(z)} \rightarrow \lambda_F$$

Proof The mapping $z \rightarrow \det(xI - L_z)$ is a continuous function of R^+ onto the space of monic polynomials $\mathcal{P}_{2p+2}(x)$.

Clearly, if exist, the limit matrix must preserve the M_0-property. Moreover, $\hat{L}$ must be also a Laplacian of a weighted graph.

Proposition (Clustering Property) Suppose that each l_z has a unique unit characteristic valuation $x_a^{(z)}$. If such $x_a^{(z)}$ is expressed with respect to the base $(V_1, v_{p+1}, V_2, v_{p+2})$ then

i) (Clustering at infinity)

$$\lim_{z->\infty} x_a^{(z)} = [\alpha \mathbf{1}_{n+1},\ \beta \mathbf{1}_{n+1}]^T$$

ii) (Clustering at zero)

$$\lim_{z->0} x_a^{(z)} = [\gamma \mathbf{1}_n,\ \alpha,\ \gamma \mathbf{1}_n,\ \beta]^T$$

in either case $\alpha^2 + \beta^2 \neq 0$

Proof We proof first clustering at infinity. By construction,

$$L_z = \left|\begin{array}{ccc|cc} d_{11}+z & & & -\mathbf{1}_n z & \\ & \cdots & & & -\mathbf{1}_n z \\ & & d_{2n,2n}+z & & \\ \hline -\mathbf{1}_n^T z & & & nz & \\ & -\mathbf{1}_n^T z & & & nz \end{array}\right|$$

It is clear that L_z and L_z/z have the same eigenvector and

$$\lim_{z->\infty} L_z/z = \left|\begin{array}{cc|cc} I_n & & -\mathbf{1}_n & \\ & I_n & & -\mathbf{1}_n \\ \hline -\mathbf{1}_n^T & & n & \\ & -\mathbf{1}_n^T & & n \end{array}\right|$$

This is a has rank $2p$ and the eigenvectors corresponding to the multiplicity 2 zero of the characteristic polynomial is are

$$x = \alpha[\mathbf{1}^T,\ 0, 1, 0] + \beta[0,\ ,\mathbf{1},\ 0,\ 1]^T$$

thus $x_a^{(z)}$ must to converge to a vector of this form. The limit of the eigenvector sequence must preserve the norm, thus $\alpha^2 + \beta^2 \neq 0$.

Clustering at zero follows the same technique.

References

[1] P. Agati, Y. Bremont, and G. Delville. – *Mécanique du solide.* – Dunod, Paris, 1986.

[2] S.T Barnard and H.D. Simon. – Fast multilevel implementation of recursive spectral bisection. – *Concurrency: Practice and Experience*, 6(2):101–117, 1994.

[3] C. Farhat and H.D. Simon. – Top/domdec— a software tool for mesh partitioning and parallel processing. – Technical Report RNR-93-011, NASA, June 1993.

[4] M. Fiedler. – A property of eigenvectors of nonnegative symmetric matrices and its application to graph theory. – *Czechoslovak Math. Jour.*, 25(100):619–633, 1975.

[5] Y.F. Hu and R.J. Blake. – Numerical experience with partitioning of unstructured meshes. – *Parallel Computing*, (20):815–829, 1994.

[6] D. Knuth. – *Sorting and Searching*, volume 2 of *The Art of Computer Programming.* – Addisson–Wesley, 1973.

[7] Garey M.R. and D.S. Johnson. – *Computers and Intractability.* – Freeman, San Francisco, 1973.

[8] C.H. Papadimitriou and K. Steiglitz. – *Combinatorial Optimization.* – Prentice-Hall, 1982.

[9] A. Pothen, H. Simon, and K.-P Liou. – Partitioning sparse matrices with eigenvalues of graphs. – *SIAM J. Matrix Anal. Appl.*, 11(3):430–452, 1990.

[10] D. Powers. – Graph partitioning by eigenvectors. – *Jour. Lin. Alg. Appl.*, (101):121–133, 1988.

[11] R. Read. – Some applications of computers in graph theory. – In L.W. Beineke and R. Wilson, editors, *Selected Topics in Graph Theory*, chapter 15, pages 415–444. Academic Press, 1978.

[12] H. Simon. – Partitioning of unstructured problems for parallel processing. – *Computing Systems in Engineering*, 2(2/3):135–148, 1991.

[13] D. Skiena. – *Implementing Discrete Mathematics.* – The Advanced Book Program. Addisson–Wesley, 1990.

[14] R. Van Driessche and D. Roose. – An improved spectral bisection algorithm and its application to dynamic load balancing. – *Parallel Computing*, (21):29–48, January 1995.

[15] R.D. Williams. – Performance of dyanamic load balance for unstructured mesh calculations. – *Concurrency: Practice and Experience*, 3(5):457–481, October 1991.

Unité de recherche INRIA Lorraine, Technopôle de Nancy-Brabois, Campus scientifique,
615 rue du Jardin Botanique, BP 101, 54600 VILLERS LÈS NANCY
Unité de recherche INRIA Rennes, Irisa, Campus universitaire de Beaulieu, 35042 RENNES Cedex
Unité de recherche INRIA Rhône-Alpes, 46 avenue Félix Viallet, 38031 GRENOBLE Cedex 1
Unité de recherche INRIA Rocquencourt, Domaine de Voluceau, Rocquencourt, BP 105, 78153 LE CHESNAY Cedex
Unité de recherche INRIA Sophia-Antipolis, 2004 route des Lucioles, BP 93, 06902 SOPHIA-ANTIPOLIS Cedex

Éditeur
INRIA, Domaine de Voluceau, Rocquencourt, BP 105, 78153 LE CHESNAY Cedex (France)
ISSN 0249-6399