AN ACCURATE HYPERGRAPH MODEL FOR MESH PARTITIONING

CÉDRIC CHEVALIER AND ERIK G. BOMAN

1. Introduction

Numerical simulations of physical phenomena are usually mesh based. In order to compute efficiently in parallel, the data (thus the mesh) has to be carefully distributed. This choice of distribution is achieved by doing a mesh partitioning in p parts when the number of processors used during the computation is p.

A mesh is a discretization of the computational domain, mainly used to solve partial differential equations (PDEs). We focus on unstructured meshes commonly used in finite element and finite volume methods. The mesh is a set of entities (regions) named elements which are geometrically limited by nodes and by the edges between nodes, as shown in Figure 1(a).

The goal of mesh partitioning is to divide the elements into p subsets such that the work per processor is approximately constant and that the communication volume between processors is minimized. Typically, graph partitioning is used even though it is known to have some deficiences [1]. We show that a hypergraph model is more accurate, and enables new partitioning approaches.

2. Models

In the following models, we partition the elements. This is usually preferred because many computations are internal to the elements, so parallel codes require each element to be owned by a single processor.

2.1. **Dual graph models.** In these models, the mesh is modeled as a graph, in order to use a graph partitioning tool. These approaches consist in looking only at the elements of the mesh and their adjacencies. The more common relation of adjacency for two elements is to share a face, and in this case we can talk about face-based dual graph (Figure 1(c)). It is also possible to have a relation of adjacency between two elements based on the fact that they share a node, and in this case we speak about node-based dual graph (Figure 1(d)).

Performing graph partitioning on these dual graphs consists in dividing the elements in p disjunct subsets (or parts) of the same size, while minimizing the cut edges. In the case of a faced-based dual graph, the partitioning will minimize the number of faces shared between elements of different parts.

With a node-based dual graph, we tend to minimize the number of nodes shared between the parts. However, this model is not exact in terms of minimizing the number of nodes, due to the fact that each node correspond to a clique in the graph to partition and therefore the edge cut is not directly a measurement of the number of nodes, even if the two are closely correlated.

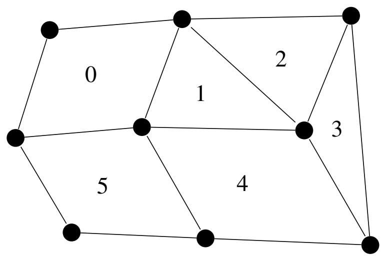
2.2. **Hypergraph Model.** The number of nodes shared is particularly important for the widely used continuous Galerkin finite elements methods because the communications are on the nodes, not on the faces. As the graph-based approach does not model this accurately, we introduce an hypergraph based approach which can resolve this issue.

This model is perhaps easier to describe first with a bipartite graph, which is a graph with two kinds of vertices. One kind will be the elements of the mesh, the other kind being the nodes of the mesh. Edges are only between two vertices of different types and their meaning in this case is that the element needs this node.

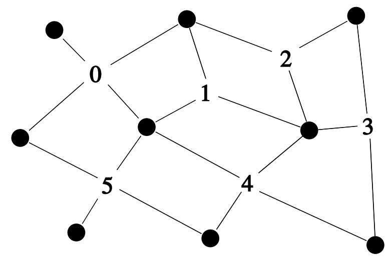
With the hypergraph terminology, this model is defined by the set of vertices which correspond to the elements in the mesh, and a set of hyperedges which correspond to the nodes in the mesh. Hypergraph offers mainly two metrics that are relevant in our case: the number of cut hyperedges, which is the number

---

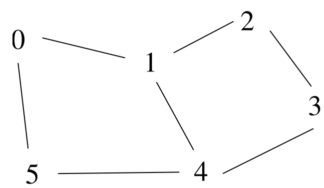
Scalable Algorithms Department, Sandia National Laboratories, Albuquerque, NM. Sandia is a multiprogram laboratory operated by Sandia Corporation, a Lockheed Martin Company, for the United States Department of Energy's National Nuclear Security Administration under Contract DE-AC04-94AL85000.



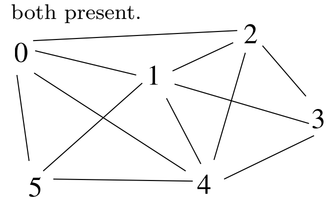
(a) A 2D mesh with 7 elements (3 quadrangles (0,5,4) and 3 triangles(1,2,3)) and 9 nodes.



(b) The hypergraph model corresponding to the previous mesh. Here the elements and the nodes of the mesh are both present.



(c) Dual graph defined by the faces.



(d) Dual graph defined by the nodes.

Figure 1. A mesh and its different associated models.

of shared nodes in the mesh, and a connectivity metric that weights each cut hyperedge with the count the number of participating processors minus one (λ-1 cut metric) which exactly represents the communication volume for a node based mesh method.

3. Observations and Conclusions

There is, in general, no direct correlation between the minimization of the number of shared faces and the number of shared nodes implied by the mesh partitioning. Nevertheless, in our experience, the partitioning model does not impact the results much in most cases. Indeed, when all the faces of all the elements are the same type, as the separators are mostly contiguous, it is almost the same to compute a face separation or a node separation of the mesh.

However, for meshes with mixed types of elements, the hypergraph model will be more accurate to model the number of nodes shared by different parts. For example, for meshes with tetrahedrons, hexahedrons and pyramids, faces can be of different types, like triangle, quadrangle or any type of polygons, and this information does not appear with the dual graph models.

Moreover, the hypergraph model is intrinsically richer than the dual graph models as it keeps information for both elements and nodes. This will allow more complex objectives for partitioning, like minimizing the number of ghosts vertices or elements, or having a better load balancing of the computation and of the memory. Indeed, when a code uses ghosting of elements, the ghost elements are not modeled during the partitioning and thus are not balanced. This is an important phenomenon as the number of parts become larger but the size of locally owned elements stay constant, like on massive parallel computers with limited local memory (e.g., BlueGene).

Another point is that the structure of the hypergraph is exactly the same as the one of the mesh, thus it will be very to update the hypergraph in order to take into account mesh changes to do dynamic load balancing.

Hypergraphs thus offer a more accurate model for mesh partitioning and this model can be more easily extended to handle the new challenges of petascale computing. Parallel software for hypergraph partitioning is currently available (Zoltan).

References

[1] B. Hendrickson and T. G. Kolda. Graph partitioning models for parallel computing. *Parallel Computing*, 26:1519 – 1534, 2000.