Adaptive Dynamic Projection-Based Partitioning for Parallel Delaunay Mesh Generation Algorithms

Clemens Kadow*
Carnegie Mellon University
Pittsburgh, PA 15213-3890, USA

Extended Abstract

Meshes of high quality are an important ingredient for many applications in scientific computing. An accurate discretization of the problem geometry with elements of good aspect ratio is required by many numerical methods. In the finite element method, for example, interpolation error is related to the largest element angle in the mesh [1]. There is a critical need for algorithms that can generate meshes of provably high quality. For large-scale problems that require frequent remeshing (such as problems with evolving geometry), these algorithms must run in parallel on distributed memory machines.

Whereas in recent years great strides have been made in parallel solvers, automatic parallel mesh generation for arbitrary domains remains an unsolved problem.

Delaunay Refinement has proven useful for generating meshes of good aspect ratio. Provably good working algorithms that generate meshes for arbitrary domains exist in two dimensions. Efficient sequential implementations are available [2,3]. In three dimensions the problem is more challenging. Recent theoretical results [4] suggest algorithms to solve the general three dimensional meshing problem, but all sequential implementations available today can only cope with input that respects large angle bounds.

Delaunay Refinement algorithms generate a Delaunay triangulation of the input vertices. New vertices are then added to the mesh while maintaining the Delaunay property such that the final triangulation respects the input boundary and all elements in the mesh satisfy certain quality constraints. The difficulty in Delaunay Refinement lies in selecting the appropriate vertices to add. In the end a correct refinement algorithm will generate points such that the vertex density at any point p in the domain is related to the *local feature size* at p. The *local feature size* is a quantity determined purely by the input. Delaunay Mesh Generation and Refinement is a much more challenging problem than triangulation. The complexity of the problem has only recently been established [5].

The meshing problem is tightly coupled and sequential in nature. The only efficient method known to determine the local feature size is to actually run an incremental Delaunay Refinement algorithm and look at the vertex density. This makes parallel extensions to the existing sequential algorithm hard. A single vertex insertion only changes local parts of the mesh, but in an unstructured fashion.

Rather than extend existing sequential algorithms, in this paper we will present a new fully parallel algorithm to efficiently generate and refine two dimensional conforming Delaunay meshes on coarse-grain distributed memory parallel computers.

Our algorithm is based on the optimal parallel triangulation algorithm of Blelloch, Miller, et al. [6]. They use a projection idea to partition the input among the processors. In their team-parallel approach, initially all input is distributed arbitrarily over all processors. Then the set of processors is divided recursively. Once a team consists of a single processor, a fast serial version of the algorithm is executed on each processor.

The key step is the partitioning step, in which a set of edges in the final mesh is determined by a projection method prior to generating any elements. The input is then split and redistributed among the processors in each team. Since the partition between processors contains edges of the final mesh, the submeshes on each processor will fit together without need for a synchronizing operation.

While in the triangulation problem all mesh vertices are known at the beginning, a mesh generation algorithm has to allow for dynamic vertex insertions at runtime. Earlier approaches to parallel mesh generation were based

---

[*] kadow@cmu.edu, this research is supported by the National Science Fundation under the Information Technology Research program, grant ACI-0121667

on some strategy that would partition the problem once in the beginning. Then the partitions were kept fixed, and the parts were meshed. Problems with this approach arise when new vertices need to be inserted on or close to partition boundaries.

There has been successful work by Chrisochoides et al. [7] on refining existing two- and three-dimensional meshes in parallel. They present several strategies to deal with dynamically changing partitions.

Our approach is somewhat similar. But in contrast solves the full two-dimensional meshing problem. We do not need to create an initial mesh sequentially. Thus, our algorithm can cope with problems so large that the input does not fit on a single processor. All steps are fully parallel. Also, in our approach we can very precisely define how the final mesh is distributed over the machine. This way we might already supply a partitioned mesh to some numerical simulation to follow.

The central idea of our algorithm is an adaptive extension to the projection-based partitioning by Blelloch, Miller, et al. The initial partition evolves as new vertices are added to the mesh. Our algorithm works in three phases. First, we partition the input points as in the algorithm by Blelloch, Miller et al. We also partition the boundary segments among the processors as we work on the points. Then, we generate a conforming mesh in parallel. Depending on the problem some new vertices will need to be inserted to make the mesh conform to the boundary. In a third phase we refine the distributed mesh in parallel by adding circumcenters of badly shaped triangles. As required by a correct Delaunay Refinement algorithm a circumcenter can only be added to the mesh if it is not close to a segment. Our parallel algorithm needs to be able to resolve this even across partition boundaries.

Whenever a new vertex is inserted into the mesh, the partitions potentially change. Each processor has simple geometric tests at hand to decide if a vertex insertion is local. Most vertex insertions are local and no modifications to the partitions are needed. Only vertices that change a partition will get communicated to the neighboring processes. The updates to the local meshes can be done asynchronously without blocking operations by uniqueness of the Delaunay triangulation.

Our approach readily extends to higher dimensions. However, there are two issues that need to be solved. First, to extend the projection-based partitioning one would need a work-optimal parallel convex hull algorithm for dimensions higher than two, which is not yet available. Second, as mentioned before, the meshing problem for general three dimensional problems with arbitrary (small) angles is unsolved even sequentially. For input with large angle bounds, our parallel algorithm has a straightforward extension to three dimensions.

This is joint work with my advisors Noel Walkington, Gary Miller, Guy Blelloch, and Omar Ghattas at Carnegie Mellon University.

References

[1] I. Babuska and A. K. Aziz, "On the Angle Condition in the Finite Element Method", *SIAM Journal on Numerical Analysis*, 13(2):214–226, 1976.

[2] J. R. Shewchuk, "Triangle: Engineering a 2D Quality Mesh Generator and Delaunay Triangulator", *WACG: 1st Workshop on Applied Computational Geometry: Towards Geometric Engineering*, 1996.

[3] S. Pav, "Delaunay Refinement Algorithms", PhD Thesis, Carnegie Mellon University, 2003.

[4] D. Cohen-Steiner, E. Colin de Verdiere, M. Yvinec, "Conforming Delaunay Triangulations in 3D", *18th ACM Symposium on Comp. Geometry, Barcelona, Spain*, p. 199-208, 2002.

[5] G. Miller, "A Time Efficient Delaunay Refinement Algorithm", accepted paper at *ACM-SIAM Symposium on Discrete Algorithms*, 2004

[6] G. Blelloch, J. Hardwick, G. Miller, and D. Talmor, "Design and Implementation of a Practical Parallel Delaunay Algorithm", *Algorithmica*, 24(3/4), 1999.

[7] D. Nave, N. Chrisochoides, P. Chew, "Guaranteed–Quality Parallel Delaunay Refinement for Restricted Polyhedral Domains", *18th ACM Symposium on Comp. Geometry, Barcelona, Spain*, p. 135-144, 2002.