Parallel Guaranteed Quality Planar Delaunay Mesh Generation by Concurrent Point Insertion*

Extended Abstract

Andrey N. Chernikov and Nikos P. Chrisochoides
Computer Science Department
College of William and Mary
Williamsburg, VA 23185

Abstract *We develop a theoretical framework for constructing parallel guaranteed quality Delaunay planar meshes using commercial off-the shelf software (COTS). We call two points Delaunay-independent if they can be inserted concurrently without destroying the conformity and Delaunay properties of the mesh. First, we present a sufficient condition of Delaunay-independence. It is based on the distance between points, can be verified very efficiently and used in practice. Second, we show that a simple block mesh decomposition can be utilized in order to guarantee a-priori Delaunay-independence of points in certain regions. Third, we derive an expression which relates three mesh quality and size parameters that allow to conduct the preprocessing step of our approach using a sequential Delaunay refinement algorithm. We conclude with our current work in progress that includes extending the presented approach to generate nonuniform graded meshes.*

Introduction

Nave, Chrisochoides, and Chew [7] presented a practical provably-good parallel mesh refinement algorithm for polyhedral domains. The approach in [7], due to absence of sequential code reuse, as well as intensive unpredictable communication and setbacks, is labor intensive. In this paper, we develop an approach which allows to use COTS, requires only structured bulk communication, and eliminates setbacks by ensuring that the inserted points are Delaunay-independent.

Linardakis and Chrisochoides [6] described a Parallel Domain Decoupling Delaunay method for 2-dimensional domains, which is capable of leveraging the serial meshing codes. However, it is based on the Medial Axis which is very expensive and difficult to construct for 3-dimensional geometries. The approach developed in the present work is domain decomposition independent, i.e. it does not require an explicit construction of internal boundaries.

Blelloch, Hardwick, Miller, and Talmor [2] describe a divide-and-conquer projection-based algorithm for constructing Delaunay triangulations of pre-defined point sets in parallel. Our goal, though, is to refine an existing mesh by inserting triangle circumcenters, i.e., the set of points in the final mesh is not known in advance.

Kadow in [5] extended [2] for parallel mesh generation. The principal difference between [7] and [5] is that in [5] the need to construct an initial mesh sequentially is eliminated.

Edelsbrunner and Guoy [4] define the points x and y as independent if the closures of their *prestars* (or *cavities* [7]) are disjoint. We start with proving a similar condition of point independence [3]. Our formulation is less restrictive: it allows the cavities to share a point. However, computing the cavities and their intersections for all candidate points is very expensive. That is why we do not use coloring methods that are based on the cavity graphs and we prove a theorem, which allows to use only the distance between the points for checking their Delaunay-independence. The minimum separation distance argument in [4] is used to derive the upper bound on the number of inserted vertices and prove termination, but not to ensure point independence.

Spielman, Teng, and Üngör [9] presented the first theoretical analysis of the complexity of parallel Delaunay refinement algorithms. However, the assumption is that the global mesh is completely retriangulated each time a set of independent points is inserted [11]. In [10] the authors developed a more practical algorithm which takes $\mathcal{O}(\log m)$ time (i.e. number of parallel iterations) using m processors, where m is the size of the output. In contrast, our approach [3] uses only four parallel refinement iterations with a fixed number of processors, where each iteration on a single processor is performed by a sequential mesher [8]. The present work is an extension of the work we presented in [3].

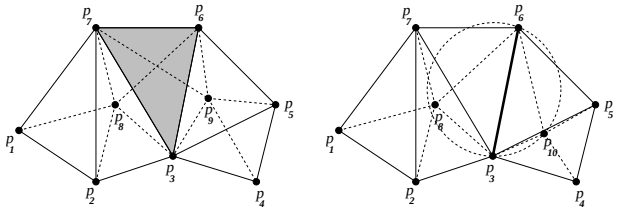
The Theoretical Framework

Sequential Delaunay refinement algorithms are based on inserting circumcenters of triangles which violate the required bounds, e.g. the upper bound $\bar{\rho}$ on circumradius-to-shortest edge ratio, and the upper bound $\bar{\Delta}$ on triangle area. Let the *cavity $\mathcal{C}_{\mathcal{M}}(p)$ of point p with respect to mesh $\mathcal{M}$* be the set of triangles in $\mathcal{M}$, whose open circumdisks include p. We expect our parallel Delaunay refinement algorithm to insert multiple circumcenters concurrently in such a way that at every iteration the mesh will be both conformal and Delaunay. Figure 1 illustrates how the concurrently inserted points can violate one of these conditions.

Theorem 1 *Let $\bar{r}$ be the upper bound on triangle circumradius in the mesh and $p_i, p_j \in \Omega \subset \mathbb{R}^2$. Then if $\|p_i - p_j\| \geq 4\bar{r}$, then independent insertion of p_i and p_j will result in a mesh which is both conformal and Delaunay.*

To show that Theorem 1 is applicable throughout the run of the algorithm, we prove that the execution of the

*This work was supported by NSF grants: CCR-0049086, ACI-0085969, EIA-9972853, EIA-0203974, and ACI-0312980



(a) **(b)**

Figure 1: **(a)** If $\triangle p_3p_6p_7 \in \mathcal{C}(p_8) \cap \mathcal{C}(p_9)$, then concurrent insertion of p_8 and p_9 yields a non-conformal mesh. Solid lines represent edges of the initial triangulation, and dashed lines — edges created by the insertion of p_8 and p_9. Note that the intersection of edges p_8p_6 and p_9p_7 creates a non-conformity. **(b)** If edge p_3p_6 is shared by $\mathcal{C}(p_8) = \{\triangle p_1p_2p_7, \triangle p_2p_3p_7, \triangle p_3p_6p_7\}$ and $\mathcal{C}(p_{10}) = \{\triangle p_3p_5p_6, \triangle p_3p_4p_5\}$, the new triangle $\triangle p_3p_{10}p_6$ can have point p_8 inside its circumdisk, thus, violating the Delaunay property.

# of processors	Pipe with holes		Unit square	
	Time, sec.	# of elmnts, $\times 10^6$	Time, sec.	# of elmnts, $\times 10^6$
4	179.1	14.6	293.7	23.8
64	273.6	233.3	300.1	470.7
121	212.7	441.1	293.7	873.5

Table 1: Scaled workload, the area bound is inversely proportional to the number of processors.

Bowyer/Watson kernel [7], either sequentially or in parallel, does not violate the condition that $\bar{r}$ is the upper bound on triangle circumradius in the entire mesh.

Theorem 2 *The condition that $\bar{r}$ is the upper bound on triangle circumradius in the entire mesh holds both before and after the insertion of a point.*

In order not to check the independence condition for every pair of candidate points, we utilize a coarse-grained domain decomposition scheme. A coarse uniform lattice is overlapped over the triangulation domain in such a way that any pair of points in non-adjacent cells are guaranteed to be no less that $4\bar{r}$ apart. To enforce the $\bar{r}$ circumradius bound in the mesh we derive the following relation which allows to use the standard sequential Delaunay refinement algorithms for preprocessing:

Theorem 3 *If $\bar{\rho}$ and $\bar{\Delta}$ are upper bounds on triangle circumradius-to-shortest edge ratio and area, respectively, then $\bar{r} = 2(\bar{\rho})^{3/2}\sqrt{\bar{\Delta}}$ is an upper bound on triangle circumradius.*

Some results for shared and distributed memory implementations[1] are shown in Tables 1 and 2. Table 1 also indicates that there is potential for improvement by using the Load Balancing Library [1].

[1]This work was performed using computational facilities at the College of William and Mary which were enabled by grants from Sun Microsystems, the National Science Foundation, and Virginia's Commonwealth Technology Research Fund.

# of processors	Time, sec. MPI	Time, sec. OpenMP	# of elmnts, $\times 10^6$
4	220.3	214.1	14.6

Table 2: Pipe cross-section, distributed (MPI) and shared (OpenMP) memory implementations.

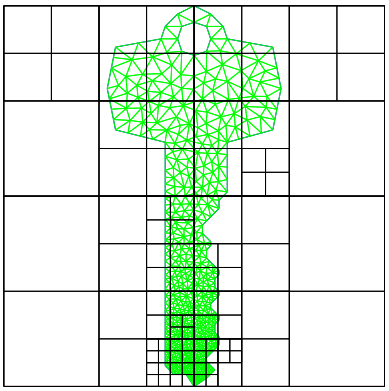


Figure 2: Graded mesh of Jonathan Shewchuk's key. The parallel refinement is guided by a quadtree.

Conclusions and Work In Progress

The approach we developed allows the use of sequential COTS for guaranteed quality parallel meshing.

Currently, we are working on extending our results to graded meshes like the one shown in Fig. 2 by using a quadtree instead of a uniform lattice, and to 3 dimensions.

References

[1] K. Barker, A. Chernikov, N. Chrisochoides, and K. Pingali. A load balancing framework for adaptive and asynchronous applications. *IEEE TPDS*, 15(2):183–192, Feb. 2004.

[2] G. E. Blelloch, J. Hardwick, G. L. Miller, and D. Talmor. Design and implementation of a practical parallel Delaunay algorithm. *Algorithmica*, 24:243–269, 1999.

[3] A. N. Chernikov and N. P. Chrisochoides. Practical and efficient point insertion scheduling method for parallel guaranteed quality Delaunay refinement. *Proc. 18th Intl. Conf. on Supercomputing*, 48–57. ACM Press, 2004.

[4] H. Edelsbrunner and D. Guoy. Sink-insertion for mesh improvement. *Proc. 17th Simp. on Comp. Geom.*, 115–123. ACM Press, 2001.

[5] C. Kadow. Adaptive dynamic projection-based partitioning for parallel Delaunay mesh generation algorithms. *SIAM Workshop on Combinatorial Sci. Comp.*, Feb. 2004.

[6] L. Linardakis and N. Chrisochoides. Parallel domain decoupling Delaunay method. *SIAM J. Sci. Comp.*, under revision, 2004.

[7] D. Nave, N. Chrisochoides, and L. P. Chew. Guaranteed–quality parallel Delaunay refinement for restricted polyhedral domains. *Comp. Geom.: Theory and Applications*, 28:191–215, 2004.

[8] J. R. Shewchuk. Triangle: Engineering a 2D quality mesh generator and Delaunay triangulator. *Proc. 1st Workshop on Appl. Comp. Geom.*, 123–133, 1996.

[9] D. A. Spielman, S.-H. Teng, and A. Üngör. Parallel Delaunay refinement: Algorithms and analysis. *Proc. 11th Intl. Meshing Roundtable*, 205–217, 2001.

[10] D. A. Spielman, S.-H. Teng, and A. Üngör. Time complexity of practical parallel Steiner point insertion algorithms. *Proc. 16th Simp. on Parallelism in alg. and arch.*, 267–268. ACM Press, 2004.

[11] S.-H. Teng. Personal Communication. *SIAM PP'04*, Feb. 2004.