GRADED DELAUNAY DECOUPLING METHOD FOR PARALLEL GUARANTEED QUALITY PLANAR MESH GENERATION

LEONIDAS LINARDAKIS AND NIKOS CHRISOCHOIDES

Abstract. We present a method for creating large 2D graded, guaranteed quality, Delaunay meshes on parallel machines. The method decouples the mesh generation procedure for each subdomain, and thus eliminates the communication between processors, while the final mesh is of guaranteed quality. This work extents our previous result on the uniform Delaunay Decoupling method [19], by allowing the element size to be governed by a sizing function, or a background mesh, and thus to produce large graded meshes. The graded Delaunay decoupling method demonstrates high efficiency, ten billion elements can be created in five minutes, with very small over-refinement (about 2%), and also results high speed-ups. This parallelization approach is effective, using off-the-self, well tested and fine tuned, sequential mesh generation libraries without modification.

Key words. mesh generation, domain decomposition, Delaunay triangulation, parallel algorithms, distributed memory computers

AMS subject classifications. 65M50, 65N50, 65L50

1. Introduction.

In [19] we described a decoupling procedure for uniform parallel guaranteed quality Delaunay mesh generation. In this paper we extend the decoupling method for generating large graded Delaunay meshes in parallel. A sizing function, or a background mesh, is used to control the mesh element size, and thus the gradation of the mesh. The sizing function is considered to be a real (hence isotropic) continuous positive function defined over the whole domain. On the other hand, the background mesh consists of a set of nodes in the domain, that store the desired element size. The decoupling procedure eliminates the communication during the mesh generation procedure, by applying a sequential mesh generator independently on each subdomain. This approach is proved to be proved to be effective, efficient and stable.

The Parallel Delaunay decoupling method consists of two major steps: the domain decomposition step, and the decoupling procedure step. The domain decomposition is governed by the same sizing function, or the background mesh, as the mesh generation procedure. The gradation of the mesh should to be small to result a good quality mesh, and large meshes will be locally almost uniform. We propose a domain decomposition procedure that identifies regions of the domain where the mesh has bounded gradation. This is achieved by imposing gradation bounds that the sizing function should satisfy in the interior of every subdomain. The mesh gradation along neighboring subdomains is also bounded, which in turn will allow the decoupling of the mesh generation procedure. Moreover, the decomposition takes into account geometric quality criteria, like the angles formed by the separators, as well as work-load balancing issues. This domain decomposition procedure is described in Section 4.

The second step is the decoupling procedure. The separators created by the domain decomposition are refined during a preprocessing step before the parallel mesh generation procedure. The refining procedure results a decoupling property that allows the subdomains to be meshed in parallel and independently, thus with no communication, and at the same time guarantees the conformity and quality of the global mesh. The decoupling step is described in Section 5.

After decoupling the subdomains, a state-of-art mesh generator (Triangle [43]) is

†Department of Computer Science, College of William and Mary, P.O. Box 8795, Williamsburg, VA 23187-8795 (lxlina@wm.edu, nikos@cs.wm.edu).

used without modifications, as a sequential mesh generation library. Other off-the-self sequential Delaunay mesh generators can be used as well. The mesh generator is applied independently and in parallel on all the subdomains. In Section 6 we describe the parallel implementation and our experimental results.

In the rest of the paper we define the domain Ω to be the closure of an open connected bounded set in $\mathbb{R}^2$, and the boundary $\partial\Omega$ to be a planar straight line graph (PSLG), which is formed by a set of line segments, intersecting only at their end-points.

2. Background.

In order to generate a mesh on a multicomputer environment it is necessary to decompose the mesh generation problem. This can be achieved in two ways: by a mesh data-decomposition approach, or by a geometric domain decomposition approach. Mesh data-decomposition approaches decompose the mesh data structure, without inserting physical separators into the geometry. On the other hand, geometric domain decompositions partition the domain by inserting separators into the geometry, and these separators will be a permanent part of the geometry. Methods that follow a mixed approach have also been proposed [11].

Mesh data-decomposition methods are attractive and have been studied extensively, because they do not have have to face the difficult geometric domain decomposition problem. A data-decomposition approach is used by Löhner and Cebral [23], who employ an octree decomposition of the domain to partition the current front in an advancing front mesh method. For parallel Delaunay mesh generation an octree decomposition is used by Chernikov and Chrisochoides [5] to identify parts of a Delaunay mesh that can be refined independently. Another common data-decomposition approach is to create an initial mesh, and then decompose it using a graph partitioner. The refining procedure can applied on each part of the mesh, with some communication to maintain the conformity; this approach has been followed by Chrisochoides and Nave [9]. Finally, Kadow and Walkington [15] employ a projective method [1] and alternate cuts to create in parallel, and decompose, an initial Delaunay triangulation; the triangulation is further refined in parallel, and the communication is controlled via an encroachment zone along the cuts.

Geometric domain decomposition approaches insert separators into the domain, and these are treated as a constrained part of the geometry. The separators will be a permanent part the geometry, they should satisfy certain quality conditions, like the angles they form. These conditions impose additional difficulty to parallel mesh generation methods that use geometric decompositions. On the other hand these methods have the advantage of low cost of communication during the parallel run. Such a method is the parallel constrained Delaunay Triangulation, proposed by Chew et al. [8]. The core Delaunay mesh refinement procedure is fast (although memory intensive), and the increasing processing power of the CPUs reveal the network as the bottleneck for parallel processing; therefore it is natural to attempt to eliminate the communication. A projective method that eliminates the communication for parallel Delaunay triangulation was proposed by Blelloch et al. [1]. A parallel mesh generation procedure with no communication for distributed memory machines is described by Said et al. [36]. The procedure though does not preserve the Delaunay properties globally and does not provide quality guarantees along the separators. A similar, more elaborated, approach is described by Ivanov et al. [14]. The procedure gives super-linear speedup, but presents the same disadvantages as in [36]. In [13] J. Galtier and P. L. George present a parallel projective Delaunay meshing method which guarantees the quality of the elements and eliminates communication, but may suffer setbacks in the form of regenerating part of the mesh.

In order to guarantee the stability (in terms of the quality and size of the final global mesh), and the termination of the parallel procedure, it is necessary to produce quality geometric domain decompositions and also to prove that the decoupling procedure guarantees the conformity of the final mesh, without compromising its quality. In [19] the authors describe such a decoupling procedure for uniform Delaunay meshes. In this paper we extend the decoupling method for creating graded Delaunay meshes in parallel with no communication.

2.1. Graded mesh generation. Delaunay mesh generation procedures, as the ones proposed by Chew [6, 7], Ruppert [34, 35], and further developed by Shewchuk [39, 41], create boundary conforming triangular meshes of good quality. The area of the elements in Delaunay meshes can grow fast, as we move away from the boundary, resulting meshes of optimal size (up to a constant factor)[35, 26]. The gradation reflects the geometric properties of the domain, but it does not reflect the computational characteristics of the model. Regions of the domain where is harder to approximate the solution should be meshed more intensively.

The way to control the element size of a mesh is to employ a sizing function that determines the element size. In the anisotropic case this function can be viewed as a tensor field over the domain [2], while in the isotropic case as a real function. In this paper we consider only the isotropic case. The sizing function can be defined over the whole domain, or over a background mesh on the domain (alternatively the sizing function can be defined over a control space). The objectives of the sizing function are two-fold: to capture the complexity of the geometry, and to optimize the quality of the mesh with regard to a specific model.

The complexity of the geometry lies on the properties of the boundary, which in turn can be used to specify a sizing function. Some of these geometric properties used to determine a sizing function are the angle variation between boundary faces [21, 22], the curvature of the boundary [10, 28, 12, 29, 17, 44], and the proximity between different boundary entities [44, 32, 33].

The behavior of the model can be assessed based on previous experience and error estimations. Sources of activity can be translated to geometrical entities, which in turn give sizing functions [21, 22, 44] usually in terms of the distance from the source. Another way is to utilize an initial, relatively coarse, mesh to obtain error estimations. This mesh can be used as a background mesh for generating a new mesh, with element sizes governed by the error estimations. The element size at each point of the domain can be determined through an interpolation procedure [28, 29]. Alternatively, Cartesian [12], and octree based background grids have been proposed to control the element sizes.

Bounding the gradation improves the quality of the mesh, and several methods have been proposed for smoothing sizing function. For the discrete cases the use of interpolation smoothing methods is common [3, 28, 18], while for the continuous case gradient limiting methods can be applied [31].

3. The Geometric Domain Decomposition Problem. Creating good geometric domain decompositions for graded mesh generation is a challenging problem because of the several constraints the decomposition has to satisfy. Some mesh generation algorithms, like Ruppert's [35], require the boundary angles to be above a bound, in order to guarantee the termination. This problem is solved in later versions of the algorithm (see [40, 4, 30]), but inevitably small boundary angles will result into meshes with poor element quality, and with more elements than necessary. Geometric decompositions create permanent separators, and the created angles will also be permanent, so they should be of good quality. We formalize this condition in the following.

Condition 1. *If Φ is any angle created between two separators, or between a separator and the boundary $\partial\Omega$, then*

$$\Phi_o \leq \Phi,$$

where $\Phi_o < \pi/2$ is a predefined constant.

A second problem is the size of the separators. Permanent separators create an artificial internal boundary on the geometry. In the cases that the parallel mesh generator is based on some communication, this communication occurs along the internal boundary [8], and it is analogous to the size of the separators. On the other hand, decoupling methods eliminate the communication, but will result some over-refinement along the separators. In both cases, a small length of separators is desirable.

Another concern is the creation of small artifacts, i.e. the insertion of separators with internal points too close to the boundary. These artifacts will cause the creation of unnecessary small elements, and thus unnecessary large meshes. What constitutes a small artifact depends on the geometry, but also on the size of the mesh. We observe that separators that form good angles (greater than a bound) tend to "stay away" from each other and from the boundary, and thus do not create small artifacts. The requirements described above are satisfied by the Medial Axis domain decomposition method, which is summarized in Section 4.1.

The mesh gradation is commonly controlled by a sizing function, or by a background mesh. The gradation produced by the sizing function should be bounded, and, especially in the case of large meshes, we expect the mesh to be locally near uniform. Our goal during the domain decomposition is to identify bounded gradation regions of the domain. This can be achieved by imposing a constant upper bound to the gradation of the sizing function inside each subdomain. Moreover, neighboring parts of the mesh should not present large size difference, and so the gradation among neighboring subdomains should also be bounded. Decompositions with the above properties can be used to decouple the mesh generation procedure.

We formulate the above two conditions as follows. For any subdomain D_i, let $m(D_i)$ denote the minimum element area and $M(D_i)$ the maximum element area inside D_i, as these are defined by the sizing function or the background mesh.

Condition 2. *For a predefined constant $R_1 > 1$ we should have*

$$M(D_i) \leq R_1 m(D_i),$$

for all subdomains D_i.

Condition 3. *For a predefined constant $R_2 > 1$ we should have*

$$m(D_i) \leq R_2 m(D_j),$$

for all neighboring (sharing a common internal boundary) subdomains D_i and D_j.

We expect the gradation inside a subdomain to be at most as large as among neighboring subdomains, so we require $R_1 \leq R_2$. An interesting theoretical problem is to find an optimal domain decomposition, in terms of the number of subdomains, that satisfies the above two conditions.

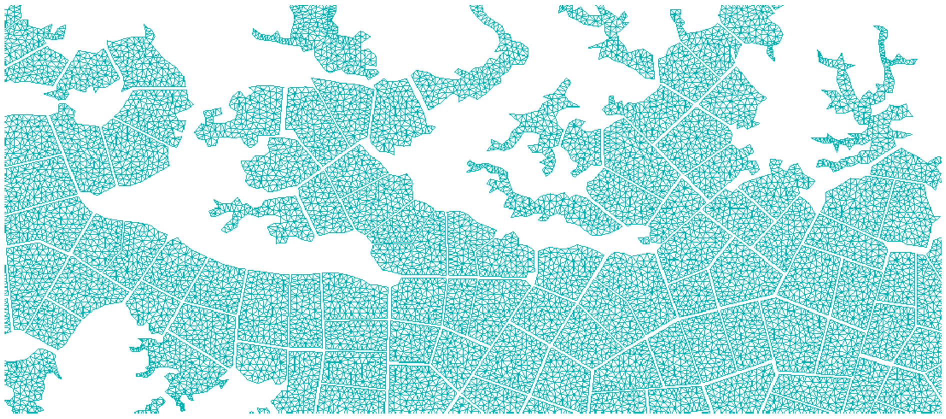

Fig. 4.1. *Part of the* Chesapeake bay *decomposed uniformly by the MADD method, and meshed using the decoupling procedure.*

4. Graded Medial Axis Domain Decomposition. In this section we describe a geometric domain decomposition procedure that satisfies the conditions that where formulated in the previous section. The procedure is based on the Medial Axis domain decomposition [20], which is applied iteratively until the conditions 2 and 3 are met. We examine both the cases of a sizing function f and of a background mesh G as control mechanisms for the maximum size of the elements. The sizing function f is considered to be positive and continuous over the whole domain Ω, while the background mesh G is an unstructured mesh over the domain.

4.1. The Medial Axis Domain Decomposition. The Medial Axis domain decomposition (MADD) was first introduced in [19], and an improved method is presented in [20]. The MADD method uses an approximation of the Medial Axis as an auxiliary structure to determine separators that form good angles. A refined Delaunay triangulation of the domain is used to approximate the medial axis. The triangulation is mapped to a graph, which is contracted and decomposed; in turn it results a geometric decomposition of the domain (see Fig. 4.1).

The core algorithm decomposes a domain into two subdomains, so that the angles angles created by the separators are no less than a tolerance Φ_o, the subdomains have approximately the same area, and the separators are relatively small. N-way decompositions are produced by iteratively applying the MADD on the subdomains. The criteria for selecting which subdomains will be decomposed determine the gradation of the final decomposition. A sizing function or a background mesh is employed to determine which subdomains will be decomposed, and so to produce graded decompositions that reflect the element size and gradation values of the mesh.

The MADD satisfies the requirements described in the beginning Section 3, and especially Condition 1. In the following sections we describe the algorithm that produces decompositions that satisfy all the conditions of Section 3.

4.2. Domain Decomposition controlled by a sizing function. We will apply the MADD procedure iteratively, so that the final decomposition satisfies the conditions 2 and 3. Given a decomposition $\mathcal{D}_n$, we identify the set $\mathcal{B}_n$ of subdomains that do not satisfy either condition 2 or condition 3. Namely, if for some subdomain $D_i \in \mathcal{D}_n$ we have $M(D_i) > R_1 m(D_i)$, then $D_i \in \mathcal{B}_n$, and if for two neighboring subdomains $D_i, D_j \in \mathcal{D}_n$ we have that $m(D_i) > R_2 m(D_j)$, then $D_i, D_j \in \mathcal{B}_n$. The largest subdomain of $\mathcal{B}_n$ is decomposed using the MADD procedure, giving a new decomposition $\mathcal{D}_{n+1}$. The procedure is repeated until all subdomains satisfy these two conditions. Algorithm 1 outlines this iterative procedure.

Algorithm 1.
```
 1.  input initial decomposition $\mathcal{D}_1 = \{\Omega\}$
 2.  identify the set $\mathcal{B}_1 \subseteq \mathcal{D}_1$ of non-acceptable subdomains
 3.  $i = 1$
 4.  while $\mathcal{B}_i \neq \emptyset$ do
 5.     let $B \in \mathcal{B}_i$ be the largest subdomain
 6.     apply MADD to $B$
 7.     $i = i + 1$
 8.     let $\mathcal{D}_i$ be the new decomposition
 9.     identify the set $\mathcal{B}_i \subseteq \mathcal{D}_i$ of non-acceptable subdomains
10.  endwhile
```

The termination of the algorithm will guarantee that the produced decomposition satisfies both the conditions 2 and 3. In order to prove the termination we will use the observation that the MADD produces decomposition topologies equivalent to the Euclidian topology, i.e., the maximum diameter of the subdomains tends to zero, when we apply iteratively the MADD on the largest subdomain. This notion is formally expressed as follows: Let $\mathcal{D}_n$ be a sequence of decompositions, each produced from the previous by applying the MADD to the largest subdomain. Then, $\max_{D\in\mathcal{D}_n} \delta(D) \to 0$, where $\delta(D) = \max \|y - x\|, x, y \in D$ is the diameter of the subdomain D.

Commonly the objectives of graph partitioners are two-fold. The first objective is to create balanced decompositions, a quality that can be described as follows: There is a constant $b_1 < 1$, so that after we decompose any subdomain D into the subdomains D_i, D_j, we have $\max\{|D_i|, |D_j|\} \leq b_1|D|$. The second objective is the creation of small separators, which is usually formulated as minimizing the ratio $\frac{|\partial D_i|}{|D_i|}$. These objectives allow us to prove that the MADD produces decomposition topologies equivalent to the Euclidean.

Lemma 4.1. *Let $\mathcal{D}_n$ be a sequence of decompositions, each produced from the previous by applying the MADD to the largest subdomain, for which the following two conditions hold: There is a constant $b_1 \in \mathbb{R}$, $b_1 < 1$, such that $\max\{|D_i|, |D_j|\} \leq b_1|D|$, for any subdomains D_i, D_j obtained by decomposing a subdomain D. There is a constant $b_2 \in \mathbb{R}$ such that $\frac{|\partial D_i|}{|D_i|} \leq b_2$ for any subdomain D_i. Then we have $\max_{D\in\mathcal{D}_n} \delta(D) \to 0$, where $\delta(D) = \max \|y - x\|, x, y \in D$ is the diameter of the subdomain D.*

Proof. The proof is performed in three steps:

Step 1. Let $A_n = \max_{D\in\mathcal{D}_n} |D|$. We will show that $A_n \to 0$. The sequence $\{A_n\}$ is clearly decreasing, so we only need to find a subsequence $\{A'_n\} \subseteq \{A_n\}$, such that $A'_n \to 0$.

Let $A'_n = \max_{D\in\mathcal{D}_{2^n}} |D|$. We will prove by induction that $A'_n \leq b_1^n|\Omega|$. For $n = 0$, and $\mathcal{D}_1 = \{\Omega\}$, the relation is obviously true. Suppose the claim is true for some $n = m$, we will prove it is true for $n = m + 1$.

For any subdomain $D \in \mathcal{D}_{2^m}$ decomposed into two subdomains D_i, D_j we have

$$\max\{|D_i|, |D_j|\} \leq b_1|D| \leq b_1 b_1^m|\Omega| = b_1^{m+1}|\Omega|.$$

Next we will show that any subdomain $D \in \mathcal{D}_{2^m}$, with $|D| > b_1^{m+1}|\Omega|$, will be decomposed. Observe that the decomposition $\mathcal{D}_{2^m}$ contains $|\mathcal{D}_{2^m}| = 2^m$ subdomains, and any new subdomain will have area less or equal to $b_1^{m+1}|\Omega|$. The decomposition $\mathcal{D}_{2^{m+1}}$ is obtained from $\mathcal{D}_{2^m}$ after decomposing $2^m = |\mathcal{D}_{2^m}|$ subdomains. So, all the subdomains $D \in \mathcal{D}_{2^m}$, with $|D| > b_1^{m+1}|\Omega|$, will be decomposed.

From the above we conclude that for any subdomain $D \in \mathcal{D}_{2^{m+1}}$ we have $|D| \leq b_1^{m+1}|\Omega|$, and thus $A'_n \leq b_1^{m+1}|\Omega|$.

From the induction we have $A'_n \to 0$, and consequently $A_n \to 0$.

Step 2. It is easy to see that $\max_{D\in\mathcal{D}_n} |\partial D| \to 0$. For any subdomain D we have $|\partial D| \leq b_2|D|$, and so

$$\max_{D\in\mathcal{D}_n} |D| \to 0 \Rightarrow \max_{D\in\mathcal{D}_n} |\partial D| \to 0.$$

Step 3. The subdomains are connected, so we have $\delta(D) \leq |\partial D|$. Consequently

$$\max_{D\in\mathcal{D}_n} |\partial D| \to 0 \Rightarrow \max_{D\in\mathcal{D}_n} \delta(D) \to 0,$$

and the lemma is proved. □

There are no strict mathematical proofs, in general, that a graph partitioner will achieve the two objectives mentioned above. In practice though, we have observed that state-of-art partitioners, like Metis [25], will give decompositions that meet these two objectives for all the geometries we have tested, and for very large scale decompositions. We proceed to give a proof of termination of the algorithm under the conditions of the previous lemma.

LEMMA 4.2. *If for a sequence of decompositions $\mathcal{D}_n$ we have $\max_{D\in\mathcal{D}_n} \delta(D)) \to 0$, then there is a decomposition $\mathcal{D}_k$ that satisfies the conditions 2 and 3.*

Proof. We have that f is continuous over a compact domain, and thus is uniformly continuous. Moreover f is bounded below by a constant positive number. Then for any $\epsilon > 0$ there is a $\delta > 0$, such that if $\|x - y\| \leq \delta$, we have $\frac{f(x)}{f(y)} \leq 1 + \epsilon$. Let δ be such that the inequality is satisfied for $1 + \epsilon = \min(R_1, R_2)$.

If $\mathcal{D} = \{D_i\}$ is a decomposition such that $\max\{\delta(D_i)\} \leq \delta$, then $\mathcal{D}$ obviously satisfies the conditions 2 and 3. Let $\mathcal{D}_k$ such that $\max_{D\in\mathcal{D}_k} \delta(D) < \delta$. Such decomposition exists, because $\max_{D\in\mathcal{D}_n} \delta(D) \to 0$, and satisfies the conditions 2 and 3. □

THEOREM 4.3. *Under the conditions of Lemma 4.1, Algorithm 1 terminates, giving a decomposition that satisfies the conditions 2 and 3.*

Proof. If Algorithm 1 terminates, then by the construction it will produce a decomposition that satisfies the conditions 2 and 3. We will prove the termination by contradiction.

We observe that if $B' \in \mathcal{B}_{n+1}$, then $B' \subseteq B$ for some $B \in \mathcal{B}_n$. We have from Lemma 4.1 that $\max_{B\in\mathcal{B}_n} \delta(B) \to 0$. Suppose that the algorithm does not terminate, then for some k we will have $\max_{B\in\mathcal{B}_k} \delta(B) < \delta$, where δ is defined in Lemma 4.2. Then, from the same lemma, $\mathcal{B}_k$ satisfies the conditions 2 and 3, which contradicts the definition of $\mathcal{B}_k$. □

4.3. Domain Decomposition controlled by a background mesh. Another way for controlling the size of the elements is to use a background mesh. This approach is common when error estimations on an existing mesh are used to govern the creation of a new mesh. We use an unstructured background mesh $G = \{g_i\}$, where each of

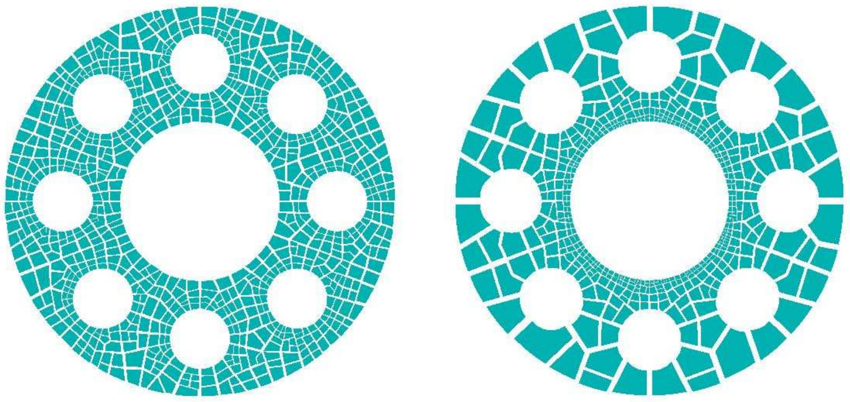

FIG. 4.2. **Left:** *The Pipe domain decomposed into 804 subdomains using a sizing function that corresponds to sources at the centers of the holes.* **Right:** *The Pipe domain decomposed into 487 subdomains using a background mesh, the element size being smaller near the center.*

its nodes g_i is assigned a sizing value $f(g_i)$. This value determines the element size at the neighborhood of the node. In the cases where the sizing value is assigned to the elements, instead of the nodes, of the background mesh, we can use an interpolation procedure to obtain sizing values on the nodes.

As in the case of a function, we assign to each subdomain D two values, $m(D) = \min\{f(g) |\ g \in D \cap G\}$ and $M(D) = \max\{f(g) |\ g \in D \cap G\}$. The decomposition should satisfy the conditions 2 and 3, stated in Section 3. The procedure described by Algorithm 1 will produce such a decomposition for a background mesh. There are though two questions we should answer, in order to show that this algorithm can be used for a background mesh: 1. What the values $m(D)$ and $M(D)$ should be, when no mesh nodes of G are in D? 2. The termination of the algorithm in the case of a continuous function f is based on the continuity of f; can it be guaranteed in the case of the background mesh?

Both questions are addressed by employing an interpolation scheme, which is used when no node of G is contained in a subdomain D.

4.4. Subdomain interpolation procedure. When no background node is contained in a subdomain D, then the minimum and the maximum element size in D will be computed using interpolation. Let D such a subdomain, with $D \cap G = \emptyset$. We compute the desirable area of the elements in D by geometric interpolation, using the values of its neighboring subdomains. Let $m_1 = \min m(D')$ and $m_2 = \max m(D')$, where the minimum and maximum values are taken over all the neighboring (sharing common boundary) subdomains D' of D. Then we assign $m(D) = M(D) = \sqrt{m_1 m_2}$. We choose the geometric mean to compute the new values because it best complies with the nature of conditions 2 and 3. Specifically, the value $\max \frac{m(D_1)}{m(D_2)}$, where $D_1, D_2 \in \{D, D'\}$, is minimized when $m(D)$ is obtained by the geometric mean. Moreover, the geometric interpolation induces a continuous function in the following sense: as the size of the decomposition grows, the values $\frac{m(D_i)}{m(D_j)}$ and $\frac{M(D_i)}{M(D_j)}$ for neighboring subdomains tend to 1. In other words, the discrete sizing values given by the geometric interpolation procedure approximate a continuous function, and following the arguments in Section 4.2, Algorithm 1 terminates.

5. Graded Delaunay Decoupling Procedure. The Delaunay decoupling procedure is aiming to generate Delaunay meshes in parallel and independently for each subdomain, thus eliminating the communication cost. This procedure was first introduced in [19] for uniform meshes, and is based on the notions of the decoupling zone and the decoupling path.

Let Ω be the domain, and $\mathcal{D} = \{D_i\}$ a geometric decomposition by a set of piecewise linear separators $\mathcal{P}$. Let $\mathcal{M}$ be a Delaunay mesh generation procedure.

DEFINITION 5.1. *The set of the open diametral circles of all the segments that form $\mathcal{P}$ is be called the* decoupling zone of $\mathcal{P}$ *with respect to* $\mathcal{M}$*, if after applying* $\mathcal{M}$ *independently on the subdomains* D_i*, this set is empty (see Fig. 5.1). We denote the decoupling zone of* $\mathcal{P}$ *by* $\mathcal{Z}_{\mathcal{P}}$*, and we call* $\mathcal{P}$ *a* decoupling path *with respect to* $\mathcal{M}$*.*

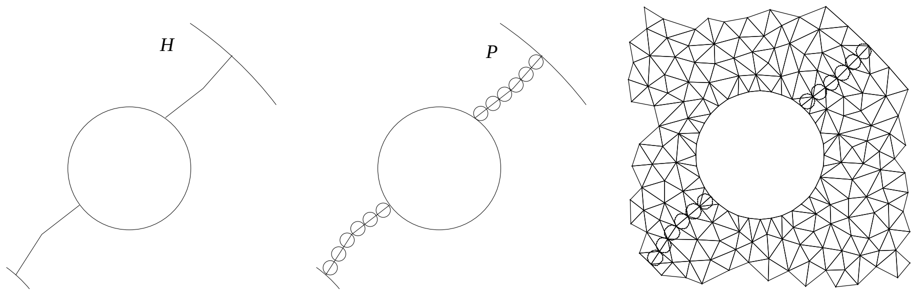

FIG. 5.1. **Left:** *A separator $\mathcal{H}$ inserted by MADD.* **Middle:** *Refining $\mathcal{H}$ gives a decoupling path $\mathcal{P}$; the decoupling zone $\mathcal{Z}_{\mathcal{P}}$ is depicted.* **Right:** *Ruppert's algorithm was applied on the subdomains independently, with a uniform element area restriction. $\mathcal{Z}_{\mathcal{P}}$ is empty and $\mathcal{P}$ is invariant, the final mesh is Delaunay conforming.*

The decoupling path presents the following property [19].

PROPOSITION 5.2. *Let M_i be the mesh produced by $\mathcal{M}$ on the subdomain D_i. If $\mathcal{P}$ is a decoupling path with respect to $\mathcal{M}$, then the union $\cup M_i$ is a conforming Delaunay triangulation.*

Proposition 5.2 proves that, after the construction of a decoupling path, the subdomains can be meshed independently, and the final mesh will be Delaunay conforming. In the rest of the section we prove the existence and the construction of the decoupling path for graded Delaunay mesh generation.

5.1. The Graded Delaunay Decoupling Path. Let $\mathcal{H}$ be the set of the piecewise linear separators produced by the domain decomposition procedure. The decoupling path is constructed by refining the initial separators $\mathcal{H}$, so that they form a decoupling path $\mathcal{P}$. The termination conditions of the Delaunay mesh generation allow us to compute a length size that should be used for refining $\mathcal{H}$ into a decoupling path $\mathcal{P}$. The two most popular Delaunay mesh generation procedures are Ruppert's algorithm [35] and Chew's algorithm [6], along with their variations. We will describe the construction of a decoupling path for Ruppert's algorithm. For uniform meshes it is known the following ([19]).

THEOREM 5.3. *Let $k = \min\{lfs_{\min}(\Omega_{\mathcal{H}}), \frac{1}{2}\sqrt{\frac{A}{\sqrt{2}}}\}$, where $lfs_{\min}(\Omega_{\mathcal{H}})$ is the minimum local feature size of $\Omega \cup \mathcal{H}$ and A is a constant bounding below the maximum triangle area. If for all the edges $E \in \mathcal{P}$ of the refined separators we have $\frac{2}{\sqrt{3}}k \leq |E| < 2k$, $|E|$ being the length of E, then $\mathcal{P}$ is a decoupling path with respect to Ruppert's algorithm, under the constrains of maximum circumradius to shortest edge ratio less or* *equal to $\sqrt{2}$ and maximum triangle area bound greater or equal to A.*

The sizing function can be constructed so that it captures the local feature size [44, 32, 33]. Moreover, for large enough meshes, we have $\frac{1}{2}\sqrt{\frac{A}{\sqrt{2}}} \leq \mathrm{lfs}_{\min}(\Omega_{\mathcal{H}})$, and consequently $k = \frac{1}{2}\sqrt{\frac{A}{\sqrt{2}}}$. Parallel mesh generation targets very large meshes, so in these cases we only have to consider the sizes of the triangles as these are determined by the sizing function.

Theorem 5.3 assumes that the triangle area bound A is constant, and thus cannot be applied as is in the case of graded meshes. It can still be used though in the graded case, for the construction of the decoupling path in the following way. Let $\mathcal{D} = \{D_i\}$ a decomposition of Ω and $m(D_i)$ as defined in Sections 4.2 and 4.3. We can apply Theorem 5.3 on each individual subdomain D_i, obtaining the following result.

PROPOSITION 5.4. *Let $k_i = \frac{1}{2}\sqrt{\frac{m(D_i)}{\sqrt{2}}}$. If for all the edges $E \in \mathcal{P} \cap D_i$, that belong to the internal boundary of D_i, we have $\frac{2}{\sqrt{3}}k_i \leq |E| < 2k_i$, then the edges of $\mathcal{P} \cap D_i$ will remain invariant after applying Ruppert's algorithm on D_i, with the constrains of maximum circumradius-to-shortest-edge ratio equal to $\sqrt{2}$ and maximum triangle area bound greater or equal to $m(D_i)$.*

Each separator E of $\mathcal{P}$ is shared by two subdomains, and in order to prove that the whole set of separators $\mathcal{P}$ forms a decoupling path, we have to examine if E remains invariant after applying the mesh generator independently to both subdomains. By applying Proposition 5.4 to each of the neighboring subdomains we obtain the following result.

PROPOSITION 5.5. *Let $E \in \mathcal{P}$ be any edge of the separators, with $E \in D_i \cap D_j$ and length $|E| = l$. If both relations $\frac{2}{\sqrt{3}}k_i \leq l < 2k_i$ and $\frac{2}{\sqrt{3}}k_j \leq l < 2k_j$ hold (for k_i, k_j as defined in Proposition 5.4), then $\mathcal{P}$ is a decoupling path.* Figure 5.2 depicts a graded

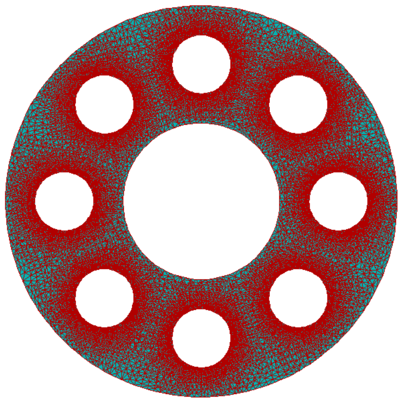
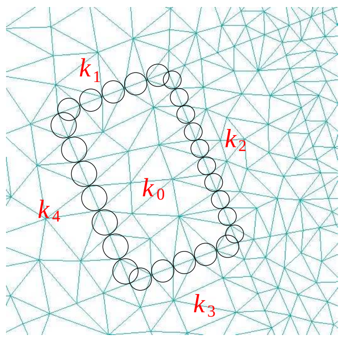

FIG. 5.2. **Left:** *Graded Delaunay mesh based on decoupling the subdomains. The sizing function reflects sources at the centers of the holes.* **Right:** *Detail of the mesh; the decoupling zone $\mathcal{Z}_{\mathcal{P}}$ for one subdomain is depicted by the circles.*

Delaunay mesh created by decoupling the subdomains, and also the decoupling zone for one subdomain.

We proceed to examine the prerequisites under which the hypothesis of the above proposition is true. Let D_i, D_j be two neighboring subdomains; without loss of generality we assume $k_i \leq k_j$. Then there exists l that satisfies both conditions of Proposition 5.5, if and only if, $\frac{k_j}{k_i} < \sqrt{3}$. If $\frac{k_j}{k_i} \geq \sqrt{3}$, it is obvious that no such l

exists. On the other hand, if $\frac{k_j}{k_i} < \sqrt{3}$, then there is such l that satisfies both conditions (for example we can choose $l = k_j$). More general, for any $l = \sqrt{3}k_i - \epsilon$, with $0 < \epsilon \leq \sqrt{3}k_i - k_j$, both conditions are true. From the definition of k_i, k_j we observe that $\frac{k_j}{k_i} < \sqrt{3} \Leftrightarrow \frac{m(D_j)}{m(D_i)} < 3$. Thus, by taking $R_2 < 3$ in Condition 3, Section 3, the relation $\frac{k_j}{k_i} < \sqrt{3}$ holds, and thus the decoupling path $\mathcal{P}$ exists.

5.2. Construction of the Graded Delaunay Decoupling Path. The condition $R_2 < 3$ allows the theoretical existence of a decoupling path, but we have to take into account that the decoupling path $\mathcal{P}$ will be constructed by refining the existing separators $\mathcal{H}$, which were created by the domain decomposition procedure. Let $E' \in \mathcal{H}$ be an edge of the separator shared by D_i, D_j, which must be refined, so that the resulting subsegments satisfy the conditions of Proposition 5.5. The refining procedure will break E' into, say, ν subsegments. Then the conditions $\frac{2}{\sqrt{3}}k_j \leq \frac{|E'|}{\nu} \Leftrightarrow \nu \leq \frac{\sqrt{3}|E'|}{2k_j}$ and $\frac{|E'|}{\nu} < 2k_i \Leftrightarrow \nu > \frac{|E'|}{2k_i}$ must hold, where $k_j \geq k_i$. In other words, an integer value should exist between the values $\frac{|E'|}{2k_i}$ and $\frac{\sqrt{3}|E'|}{2k_j}$. A sufficient condition for the above relation to be true is $\frac{\sqrt{3}|E'|}{2k_j} - \frac{|E'|}{2k_i} \geq 1$. In result, we have for the length $|E'|$ the condition

$$|E'| \geq \frac{2k_jk_i}{\sqrt{3}k_i - k_j} = \frac{2k_j}{\sqrt{3} - \frac{k_j}{k_i}} \tag{5.1}$$

in order for the created separators to satisfy the above relation, we have to keep the denominator of the right side fraction bounded below. This can be done by defining the R_2 constant to be small enough. In our experiments we use the value $R_2 = 1.5$, so that the denominator is always greater than 0.5. Then, the relation 5.1 is satisfied if

$$|E'| \geq 4k_j.$$

The decomposition is controlled by the gradation of the sizing function, and not by the sizing values, so it is invariant when we decrease the sizing function by a constant factor. While the values $|E'|$ are kept invariant, the values k_j decrease, and thus, for large meshes, the relation 5.1 holds.

We sum our results for constructing the decoupling path in the following theorem.

THEOREM 5.6. *Let the relation 5.1 hold for all separators $E' \in \mathcal{H}$, which were created by the domain decomposition procedure. Then the refined set of separators $\mathcal{P}$ is a decoupling path for Ruppert's algorithm, with the constrains of maximum circumradius to shortest edge ratio $\sqrt{2}$, and maximum triangle area bounded by the sizing function f.*

Proof. The above discussion shows that relation 5.1 guarantees that the refinement of the separators will satisfy the hypothesis of Proposition 5.5. The conclusion is driven by Proposition 5.5. □

The above theorem allows the creation of graded meshes in parallel and with no communication, since the subdomains can be meshed independently after they have been decoupled. The final mesh will be globally Delaunay, satisfying the area constrains defined by the sizing function, as well as the quality constrain of having maximum circumradius to shortest edge ratio less or equal to $\sqrt{2}$.

6. Implementation and Experimental Results.

The meDDec program [24] implements the parallel graded Delaunay decoupling procedure. It is written in c99 standard C using the LAM/MPI library. The Triangle library [38, 43] was used for the creation of the Delaunay triangulation during the MADD procedure. Triangle was also used as the off-the-shelf sequential mesh generator on each subdomain for the parallel decoupled Delaunay mesh generation. The Metis library [16, 25] was used for the graph partitioning step in the MADD procedure.

We ran three sets of experiments. A sequential set of experiments was performed to assess the stability of the decoupling method, and specifically the resulting over-refinement. A set of parallel experiments was performed on a homogenous environment in order to assess the efficiency of the method, and in particular the parallel speedup. Another set of parallel experiments was performed on a heterogenous environment in order to examine the efficiency of the method on an environment consisting of machines with different processing power and memory.

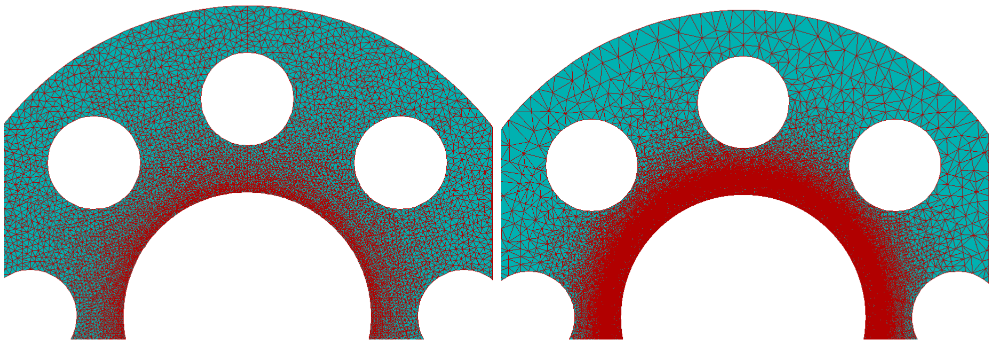

FIG. 6.1. *Part of the Pipe domain meshed by the decoupling procedure according to two sizing functions.* **Left:** *The element size is given by the function d, which is analogous to the distance from the inner hole.* **Right:** *The element size is governed by d^4, which is analogous to the fourth power of the distance from the inner hole.*

6.1. Experimental Set-up.

The domain used for our experiments is the *Pipe* model (Figs. 5.2, 6.1), which is an approximation of cross section of a rocket geometry. We tested the performance for four sizing functions. The function f_s reflects sources at the centers of the holes and is analogous to fourth power of the distance of the centers (see Fig. 5.2 left). The functions d, d^2 and d^4 are analogous to the distance from the inner hole, raised to the power of one (Fig. 6.1 left), two and four (Fig. 6.1 right), respectively. The gradation constant R_2 was set to 1.5, while R_1 was set to 1.425.

Our experiments were performed on the SciClone cluster [37]. For the homogenous environment experiments we used the tempest subcluster, consisting of 32 dual cpus at 2.4 GHz, 4 GB memory. The heterogenous environment is composed by the subclusters whirlwind (64 single cpus, 650 MHz, 1 GB memory), twister (31 dual cpus, 900 MHz, 2 GB memory) and vortex (4 quad cpus, 1.28 GHz, 8 GB memory), giving a total of 142 cpus.

6.2. Sequential Experiments.

We have ran a set of sequential experiments to observe the number of additional elements created by the decoupling procedure for different sizing functions. The results are described in Table 6.1. The over-refinement is analogous to the length of the separators, which in turn is analogous to the number of the subdomains. The gradation of the sizing function controls the decomposition, and the number of the created subdomains increases, as the local gradation gets larger. The over-refinement is relatively small, even for sizing functions that show large gradation. For the function d^4, the global gradation is 1/707281, while the additional elements after decoupling are 2.28% of the non-decoupled sequentially generated mesh size.

Size Function	Sub-domains	Triangle elements	Decouple elements	% Add. elements	Global Size gradation
d	1310	69221990	69625612	0.58	1/29
d^2	4965	70787036	71685252	1.27	1/841
d^4	19448	69614458	71198934	2.28	1/707281
f_s	10214	70761174	72032140	1.80	1/77

Table 6.1

The number of additional elements created by the decoupling procedure, as compared to the elements created by the sequential, non-decoupled, procedure.

6.3. Parallel Experiments. For the parallel mesh generation step we follow a master/worker scheme. The decomposition is performed sequentially, and the decomposed geometry is read by the master processor. The master processor controls the parallel mesh generation procedure. It maintains a sorted list of the non-processed subdomains, in terms of the expected mesh size for each subdomain. It sends the lager subdomain to the first avaliable processor, and the procedure is repeated until all the subdomains are meshed.

An important parameter that affects the parallel performance is the good balance of the work-loads among the processors. Over-decomposition of the domain, i.e. creating much more subdomains than the number of processors, has proved to be an effective approach [19]. This approach allows work-load differences for processing each subdomain to be absorbed, by assigning a set of subdomains to each processor. Over-decomposition though is less effective when the work-loads for some of the subdomains are much larger than the average work-load of all the subdomains. Moreover, the created meshes for each subdomain should fit into the avaliable memory[1]. We can bound the mesh size generated for each subdomain, and consequently the work-load, in the following way.

Condition 4. *For a predefined constant T, that designates the maximum number of elements per subdomain, we should have*

$$|D| \leq m(D_i)T,$$

where D_i is any subdomain, and $|D_i|$ denotes the area of D_i.

The above condition can be met by further decomposing the subdomains that do not satisfy it. Following the arguments of Section 4.2, the decomposition procedure will terminate. The constant T depends on the machines to be used.

The parallel mesh generation control represents a greedy load balancing strategy. This approach is effective, even for heterogenous environments, provided we have a large enough over-decomposition. Figure 6.2 depicts the load balance for two different decompositions. The load balance on the left is for 2,064 subdomains, and although

[1]The maximum mesh size we were able to create using Triangle [43] in 500MB memory, without disk swapping, is about 6M elements.

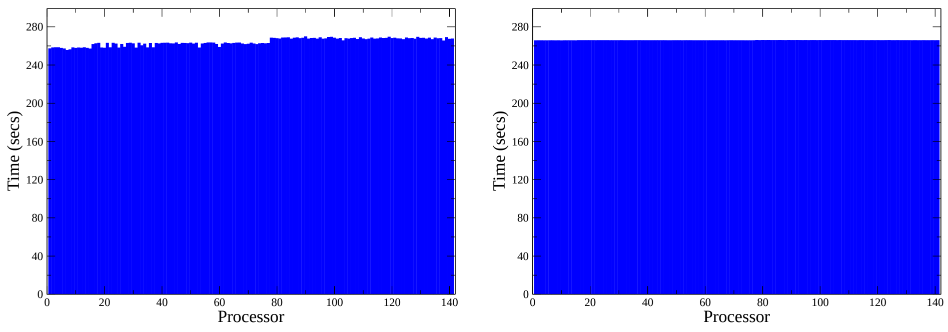

FIG. 6.2. *Load balance on heterogenous environment of 141 cpus.* **Left:** *The load balance for the d function. The decomposition is 2,064 subdomains and the created mesh is 5 billion elements.* **Right:** *The load balance for the d^4 function. The decomposition is 19,847 subdomains and the created mesh is 5 billion elements.*

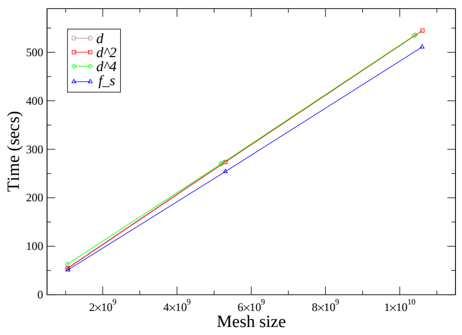

FIG. 6.3. *The time performance for the heterogenous environment (142 cpus).*

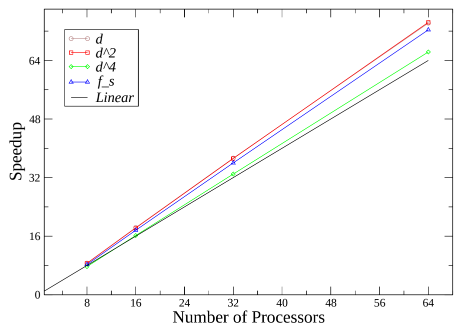

FIG. 6.4. *The speedup for the homogenous environment.*

is good, it is not perfect. The load balance on the right figure is for a much larger decomposition, 19,847 subdomains where used, and it is almost perfect. Of course, higher over-decomposition implies a higher overhead cost, and also higher communication cost. A study of optimal load balancing strategies, while keeping the overhead and communication cost small, is part of our future work.

The time performance of the decoupled mesh generation procedure in the heterogenous environment is depicted in Figure 6.3. The times are independent of the sizing function, and appear to be linear in terms of the created mesh size.

Sizing function	d	d^2	d^4	f_s
Mesh Size	10.4 B	10.6 B	10.4 B	10.6 B
Subdomains	2,526	5,086	19,839	10,404
Decomposition Time	1.23	2.55	14.4	5.92025
Meshing Time	277.57	278.76	271.58	288.801

TABLE 6.2
Performance results for the homogenous environment (64 cpus). The times are in seconds and the mesh size is in billions of elements. The meshing time includes the decomposition read and distribute time.

The performance for the homogenous environment is presented in Table 6.2. The results show that we can create 2 billion elements in less than one minute. The speedup is depicted in Figure 6.4. We have created about 81 million elements per processor, and calculated the speedup against the sequential run of Triangle for a mesh of 30 million elements (with no disk swapping). The parallel times include the decomposition cost. The decoupling procedure gives super-linear speedup, a result commonly observed for decoupling approaches. This is due to the slightly non-linear time of the mesh generation procedure, and probably because of the larger accumulative cache size. Moreover, we observe better speedup as we increase the number of processors. This is explained by the fact that we always define one processor to be the master, and dedicate it to control the mesh generation procedure.

7. Conclusions and Future Work. We presented a decoupling procedure method for creating large 2D graded, guaranteed quality, Delaunay meshes on parallel machines. The element size is controlled through a sizing function, or a background mesh. A geometric domain decomposition is described that produces graded decompositions, suitable for decoupling the mesh generation procedure. The decoupling method allows a sequential, off-the self and state-of-art, mesh generator to be applied independently on each subdomain, and thus eliminates the communication during the parallel mesh generation procedure. We have proved that the final global mesh will by conforming and of of the same quality as the sequential. Our experimental results on a cluster of workstations demonstrate the efficiency, scalability and stability of the decoupling procedure, for both homogenous and heterogenous environments.

The proof of the decoupling property of the refined separators has the same precondition as Ruppert's algorithm. For the sequential mesh generation procedure the input angles should be greater than 60°, and the same holds for the decoupling procedure. Improvements have been proposed to enable the sequential Delaunay mesh generation algorithm to terminate when the input angles are less than 60° [40, 30]. We have observed in out experiments that the decoupling procedure gives conforming meshes, even in the case of small input angles. A theoretical result that will guarantee the decoupling property in the presence of small input angles is part of our future work

Acknowledgments. The Triangle library [43, 38], created by Jonathan Shewchuk, was used as an off-the-self sequential Delaunay mesh generator. The Metis library [25, 16], by George Karypis et al, was used for the graph partitioning step in the MADD procedure. Showme [42] and mview [27] were used to create the figures. 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.

The first author gratefully acknowledges the financial support from the Virginia Institute of Marine Science under the grants ONR N00014-05-1-0831 and NOAA NA04NOS4730254.

REFERENCES

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

[2] H. Borouchaki, P. L. George, F. Hecht, P. Laug, and E. Saltel, *Delaunay mesh generation governed by metric specifications, Part I. Algorithms and Part II. Applications*, Finite Elements in Analysis and Design, 25 (1997), pp. 61–83 and 85–109.

[3] H. Borouchaki, F. Hecht, and P. J. Frey, *Mesh gradation control*, in 6th International Meshing Roundtable, Sandia National Laboratories, Oct. 1997, pp. 131–141.

[4] S. W. Cheng, T. K. Dey, E. A. Ramos, and T. Ray, *Quality meshing for polyhedra with small angles*, in Proc. 20th Annu. Sympos. Computational Geometry, 2004, pp. 290–299.

[5] A. Chernikov and N. Chrisochoides, *Practical and efficient point insertion scheduling method for parallel guaranteed quality Delaunay refinement*, in Proceedings of the 18th annual international conference on Supercomputing, Malo, France, 2004, pp. 48–57.

[6] L. P. Chew, *Guaranteed quality triangular meshes*, Tech. Report TR-89-983, Department of Computer Science, Cornell University, 1989.

[7] ———, *Guaranteed-quality mesh generation for curved surfaces*, in 9th Annual Symposium on Computational Geometry, San Diego, California, 1993, ACM, pp. 274–280.

[8] L. P. Chew, N. Chrisochoides, and F. Sukup, *Parallel constrained Delaunay triangulation*, in ASME/ASCE/SES Special Symposium on Trends in Unstructured Mesh Generation, Evanston, IL, 1997, pp. 89–96.

[9] N. Chrisochoides and D. Nave, *Parallel Delaunay mesh generation kernel*, International Journal for Numerical Methods in Engineering, 58 (2003), pp. 161–176.

[10] H. de Cougny and M. Shephard, *Surface meshing using vertex insertion*, in Proceedings of the 5th International Meshing Roundtable, 1996, pp. 243–256.

[11] H. de Cougny and M. Shephard, *Parallel volume meshing using face removals and hierarchical repartitioning*, Computer Methods in Applied Mechanics and Engineering, 174 (1999), pp. 275–298.

[12] F. Deister, U. Tremel, O. Hasan, and N. P. Weatherill, *Fully automatic and fast mesh size specification for unstructured mesh generation*, Engineering with Computers, 20 (2004), pp. 237–248.

[13] J. Galtier and P.-L. George, *Prepartitioning as a way to mesh subdomains in parallel*, in 5th International Meshing Roundtable, Pittsburgh, Pennsylvania, 1996, pp. 107–122.

[14] E. G. Ivanov, H. Andrä, and A. N. Kudryavtsev, *Domain Decomposition approach for automatic parallel generation of tetrahedral grids*, Computational Methods in Applied Mathematics, 6 (2006), pp. 178–193.

[15] C. Kadow and N. Walkington, *Design of a projection-based parallel Delaunay mesh generation and refinement algorithm*, in 4th Symposium on Trends in Unstructured Mesh Generation, 2003.

[16] G. Karypis and V. Kumar, *MeTis: Unstructured Graph Partitioning and Sparse Matrix Ordering System, Version 2.0*, 1995.

[17] C. K. Lee, *On curvuture element-size control in metric surface mesh generation*, International Journal for Numerical Methods in Engineering, 50 (2001), pp. 787–807.

[18] X. Li, J.-F. Remacle, N. Chevaugeon, and M. S. Shephard, *Anisotropic mesh gradation control*, in Proceedings, 13th International Meshing Roundtable, Williamsburg, VA, 2004, Sandia National Laboratories, pp. 401–412.

[19] L. Linardakis and N. Chrisochoides, *Delaunay Decoupling Method for parallel guaranteed quality planar mesh refinement*, SIAM Journal on Scientific Computing, 27 (2006), pp. 1394–1423.

[20] ———, *A static geometric Medial Axis Domain Decomposition in 2D Euclidean space*, ACM Transactions on Mathematical Software, to appear (2006).

[21] R. Löhner, *Extensions and improvements of the advancing front grid generation technique*, Communications in Numerical Methods in Engineering, 12 (1996), pp. 683–702.

[22] R. Löhner, *Automatic unstructued grid generators*, Finite Elements in Analysis and Design, 25 (1997), pp. 111–135.

[23] R. Löhner and J. Cebral, *Parallel advancing front grid generation*, in International Meshing Roundtable, Sandia National Labs, 1999.

[24] meDDec. `http://www.cs.wm.edu/~leonl01/meddec/meddec.html`.

[25] Metis. `http://www-users.cs.umn.edu/~karypis/metis/index.html`.

[26] S. Mitchell, *Cardinality bounds for triangulations with bounded minimum angle*, in Sixth Canadian Conference on Computational Geometry, 1994, pp. 326–331.

[27] mview. `http://mview.sourceforge.net/`.

[28] S. J. Owen and S. Saigal, *Neighborhood-based element sizing control for finite element surface meshing*, Park City, UT, 1997.

[29] ———, *Surface mesh sizing control*, International Journal for Numerical Methods in Engineering, 47 (2000), pp. 497–511.

[30] S. Pav and N. Walkington, *Robust three dimensional delaunay refinement*, in 13th International Meshing Roundtable, Williamsburg, VA, September. 2004.

[31] P.-O. Persson, *PDE-based gradient limiting for mesh size functions*, in Proc. of the 13th Int.

Meshing Roundtable, August 2004, pp. 377–387.

[32] ———, *Mesh size functions for implicit geometries and pde-based gradient limiting*, Engineering with Computers, (To appear).

[33] W. R. Quadros, S. J. Owen, M. Brewer, and K. Shimada, *Finite element mesh sizing for surfaces using skeleton*, in 13th International Meshing Roundtable, Williamsburg, VA,, 2004, Sandia National Laboratories, pp. 389–400.

[34] J. Ruppert, *A new and simple algorithm for quality 2-dimensional mesh generation*, in 4th ACM-SIAM Symp. on Discrete Algorithms, 1993, pp. 83–92.

[35] ———, *A Delaunay refinement algorithm for quality 2-dimensional mesh generation*, J. Algorithms, 18 (1995), pp. 548–585.

[36] R. Said, N. Weatherill, K. Morgan, and N. Verhoeven, *Distributed parallel Delaunay mesh generation*, Comp. Methods Appl. Mech. Engrg., 177 (1999), pp. 109–125.

[37] SciClone. `http://www.compsci.wm.edu/SciClone/index.html`.

[38] J. R. Shewchuk, *Triangle: Engineering a 2D quality mesh generator and Delaunay triangulator*, in Applied Computational Geometry: Towards Geometric Engineering, M. C. Lin and D. Manocha, eds., vol. 1148 of Lecture Notes in Computer Science, Springer-Verlag, May 1996, pp. 203–222. From the First ACM Workshop on Applied Computational Geometry.

[39] ———, *Delaunay Refinement Mesh Generation*, PhD thesis, School of Computer Science, Carnegie Mellon University, Pittsburgh, Pennsylvania, May 1997. Available as Technical Report CMU-CS-97-137.

[40] ———, *Mesh generation for domains with small angles*, in Proceedings of the sixteenth annual symposium on Computational geometry, 2000, pp. 1–10.

[41] ———, *Delaunay refinement algorithms for triangular mesh generation*, Computational Geometry: Theory and Applications, 22(1-3) (2002), pp. 21–74.

[42] Showme. `http://www.cs.cmu.edu/~quake/showme.html`.

[43] Triangle. `http://www.cs.cmu.edu/~quake/triangle.html`.

[44] J. Zhu, T. Blacker, and R. Smith, *Background overlay grid size functions*, in 11th International Meshing Roundtable, Sandia National Laboratories, September 2002, pp. 65–74.