Simultaneous Refinement and Coarsening: Adaptive Meshing with Moving Boundaries [*]

Xiang-Yang Li Shang-Hua Teng Alper Üngör

Abstract. *In the numerical simulation of the combustion process and microstructural evolution, we need to consider the adaptive meshing problem for a domain with a moving boundary, in which, the submesh in the region behind the moving boundary needs to be coarsened while the submesh in the region ahead of the moving boundary needs to be refined. In this paper, we present a unified scheme for simultaneously refining and coarsening a mesh. Our method guarantees that the resulting mesh is well-shaped and is of a size that is within a constant factor of the optimal possible. We also present several practical variations of our provably good algorithm.*

keywords. adaptive meshing, coarsening, refinement, mesh generation, moving boundary, sphere-packing, Delaunay triangulation.

1 Introduction

In the numerical simulation of many problems, we need to handle evolving meshes which change as a function of time or the number of iterations of a numerical procedure. There are two basic scenarios where we need to adaptively and dynamically generate proper evolving meshes:

- **Adaptive refinement based on posterior error analysis**: In the numerical simulation of time-independent problems, we apply an iterative procedure which first generates a mesh based on *a priori* estimates of the local mesh density, solves the numerical system defined on the initial mesh, and then based on the posterior error analysis, adaptively refines the mesh and repeats the steps for the numerical solution and adaptive refinement.
- **Dynamic meshing with a moving boundary**: In the numerical simulation of time-dependent problems such as the combustion process and microstructural evolution, we need to consider the adaptive meshing problem for a domain with a moving boundary, in which, as a function of time, the mesh need to be dynamically changed to be effective for the next step simulation.

In both cases, submeshes in some parts of the domain need to be refined, while submeshes in some other parts need to be coarsened. For example, the moving boundary of a time-dependent problem could divide the domain into two regions: the front region and the back region. See Figure 1. During the simulation, numerical conditions in the front region become stronger, requiring the submesh in the front region to be refined. In contrast, the submesh in the back region needs to be coarsened. Therefore, we need to develop a unified framework for simultaneous mesh refinement and coarsening.

In this paper, we present a sphere-packing based scheme that simultaneously refines and coarsens a mesh M. It constructs the new mesh M' as the following.

1. Based on a dynamic mesh density estimation procedure, compute the new spacing at each mesh point in M;
2. Determine the coarsening factor of each mesh point referred as a C-point whose new spacing is larger than the previous one (such as for mesh points in the back region), and the refining factor of each mesh point referred as an R-point whose new spacing is smaller than the previous one (such as for mesh points in the front region);
3. Properly scale up the spheres of all C-points and scale down the spheres of all R-points, and fill the gaps among the shrunk spheres with new spheres of proper sizes;
4. From the sphere system, construct the point set of the new mesh;
5. Use Delaunay triangulation to generate the new mesh M'.

We will show that our method guarantees that the resulting mesh is well-shaped and is of a size that is within a constant factor of the optimal possible. We also present some practical variations of the algorithms in section 5.

[*] Department of Computer Science, University of Illinois at Urbana-Champaign, Urbana, IL 61801. Supported in part by an NSF CAREER award (CCR-9502540), an Alfred P. Sloan Research Fellowship, and the U.S. Department of Energy through the University of California under Subcontract number B341494 (DOE ASCI).

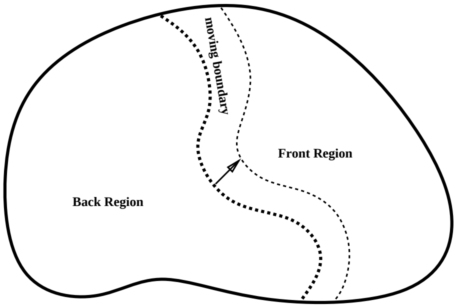

Figure 1: Domain with a moving boundary.

2 The Evolving Mesh Problem

In this section, we define the Evolving Meshing Problem which is more general than the dynamic meshing problem with a moving boundary. We will also introduce notion that will be needed in this paper.

A *mesh* M is a discretization of a domain Ω into a collection of simple elements. We consider unstructured triangular meshes which have varying local topology and spacing, and in which each element is a simplex, i.e., a triangle in 2D or a tetrahedron in 3D. The use of unstructured meshes is necessary for simulating irregular engineering problems, such as the problems that are considered in this proposed project, with fewer mesh elements [2, 9, 11].

Numerical approximation errors depend on the *quality* of the mesh, while the time and the space requirements of numerical algorithms are a function of the number of mesh elements. To properly approximate a continuous function, in addition to the conditions that a mesh must conform to the boundaries of the region and be fine enough, each individual element of the mesh must be *well-shaped.* A common shape criterion for elements is the condition that the angles of each element are not too small, or the aspect ratio of each element is bounded [1, 2, 14]. In this paper, we measure the quality of a triangular mesh by the *radius-edge aspect ratio* defined by Miller, Talmor, Teng, and Walkington [9, 10]. The radius-edge aspect-ratio of a simplex is the ratio of the circum-radius to the length of the shortest edge to of the simplex. A mesh M is *α-well-shaped* for a constant $\alpha > 1$ if the radius-edge aspect-ratio is bounded from above by α. In two dimensions, these definitions are equivalent in the sense that if a triangle is bounded away from being an ill-shaped triangle under one aspect-ratio, it is bounded away under the others as well.

A spacing function specifies how fine a mesh should be at a particular region. Given a well-shaped mesh M over a domain Ω, there are several ways to describe its spacing function:

- **[Edge-length function, el_M]** for each point $x \in \Omega$, $el_M(x)$ is equal to the length of the longest edges of all mesh simplex elements that contain x (note that points on the lower dimensional faces of a simplex are contained in more than one element).
- **[Nearest-neighbor function, nn_M]** Let x be a point in Ω, there are two cases. (1) if x is a mesh point, then $nn_M(x)$ is equal to the distance of x to the nearest mesh point in M. (2) if x is not a mesh point, then $nn_M(x)$ is equal to the distance to the second closest mesh point in M.

Lemma 1 ([9]) *If M is an α-well-shaped, then there exists constants c_1 and c_2 depending only on α such that for all point $x \in \Omega$,*

$$c_1 el_M(x) \leq nn_M(x) \leq c_2 el_M(x).$$

As shown in [9, 12], the spacing function for a well-shaped mesh should be smooth in the sense that it changes slowly as a function of distance. Formally, a function f is *Lipschitz with a constant* α if for any two points x, y in the domain, $|f(x) - f(y)| \leq \alpha||x - y||$.

We now define the Evolving Mesh Problem.

Definition 1 (The Evolving Mesh Problem (EMP)) *The input to the problem has two parts: (1) a well-shaped mesh M and (2) a list of positive reals δ, one for each mesh point, i.e., associated with each mesh point p is a real number $\delta(p)$, such that $l \leq \delta(p) \leq L$ for constants $0 < l < 1$ and $L > 1$.* [1]

We would like to construct a new mesh M' with the following properties:

- *For each mesh point p in M, $nn_{M'}(p) \leq \delta(p) nn_M(p)$;*
- *M' is well-shaped; and*
- *the size of M' is as small as possible.*

For each mesh point $p \in M$, if $\delta(p) > 1$, then it is a C-point (where C stands for coarsening); if $\delta(p) < 1$, then it is a R-point (where R stands for refinement). Our definition of the Evolving Mesh Problem allows some part of the mesh to be coarsened while some other part to be refined.

To model the dynamic meshing problem with a moving boundary by EMP, we can define δ as a function of the moving boundary. For example, we can define the new spacing of a mesh point by applying a proper monotonic function to the distance from it to the closest point on the moving boundary. EMP is more general in the sense that it does not require any correlation in the change of δ among mesh points.

EMP is very closely related with adaptive mesh generation. In the literature, adaptive mesh generation is rather a general term. It has been used as adapting the mesh to the domain geometry or to the error analysis in the mesh generation. Most often, it has been used to refer the problem for adaptive refinement based on new error bound. While we would like to use EMP to emphasize the problem of simultaneously refinement and coarsening of a mesh. The importance of our scheme is being able to handle these both cases that the domain geometry or the dynamic error analysis might bring up.

3 An Adaptive Scheme for Evolving Meshes

The objective of our algorithm for the Evolving Mesh Problem is to use the structure of the current mesh M as much as possible and as efficient as possible. First, for each mesh point p in M, we compute the value of $nn_M(p)$. Because M is well-shaped, it has a linear number of elements and edges in terms of the number of mesh points $|M|$ [9]. Therefore, nn_M can be evaluated in $O(|M|)$ time.

We now extend the spacing-function-based coarsening technique of Miller, Talmor, and Teng [7] to simultaneously refine and coarsen a mesh. The algorithm of [7] does not directly apply to EMP. See the end of Section 4 for a detailed discussion.

For each mesh point p in M, we define a local spacing function $f_p(x)$ as

$$f_p(x) = \delta(p) nn_M(p) + ||x - p||.$$

This spacing function increases with the distance, and has Lipschitz constant 1. The global spacing $f(x)$ is then given as

$$f(x) = \min_{p \in M} f_p(x).$$

In other words, f is the lower envelope of all local spacing functions. It is easy to show that f is 1-Lipschitz [7].

Lemma 2 *For any mesh point p, if $\delta(p) \leq 1$, then $f(p) = \delta(p) nn_M(p)$; if $\delta(p) \geq 1$, then $nn_M(p) \leq f(p) \leq \delta(p) nn_M(p)$.*

Proof. By definition, $f(p) = min(min_{q \neq p} f_q(p), f_p(p))$, and $f_p(p) = \delta(p) nn_M(p)$. It suffices to show $min_{q \neq p} f_q(p) \geq nn_M(p)$. For all $q \neq p$, $f_q(p) = \delta(q) nn_M(q) + ||q - p|| \geq ||q - p|| \geq nn_M(p)$. The last inequality comes from the definition of nn_M. Hence, $min_{q \neq p} f_q(p) \geq nn_M(p)$. If $\delta(p) \leq 1$, then $f(p) = min(min_{q \neq p} f_q(p), f_p(p)) = \delta(p) nn_M(p)$. Otherwise, $f(p) \leq f_p(p) = \delta(p) nn_M(p)$, and $\delta(p) * nn_M(p) \geq nn_M(p)$ implies $f(p) \geq nn_M(p)$. ∎

So for both cases, we have $f(p) \geq l * nn_M(p)$, and hence $nn_M(p) \leq f(p)/l$.

[1] The constant l defines the maximum degree of the refinement. The smaller the value l, the more a certain region of the mesh can be refined. In practice, l is a reasonably large constant, such as $1/4$.

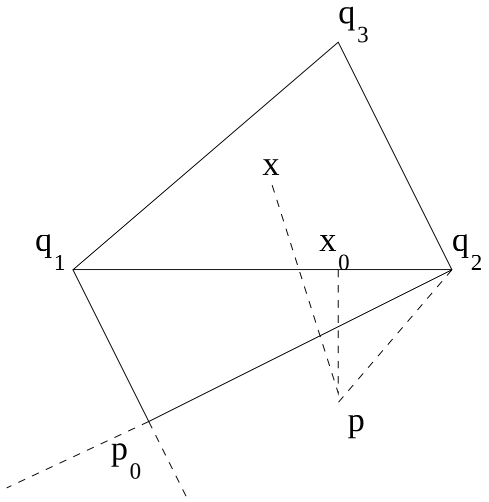

Figure 2: The distance ratio $||p-x||/min_{1\leq i\leq 3}||p-q_i||$ is at least $\tan(\theta)$, where θ is the lower bound on the angle of mesh elements.

Lemma 3 (Local Similarity) *For each edge $(p,q) \in M$, there exists a constant c_3 which depends only on the aspect ratio α of M and the lower bound l of δ, such that*

$$f(p) \leq c_3 * f(q).$$

Proof. $f(p) \leq f(q) + ||p-q|| \leq f(q) + 1/c_1 * nn_M(q) \leq f(q) + f(q)/(l * c_1) = f(q) * (1 + 1/(l * c_1))$. The second inequality follows from Lemma 1 that $||p-q|| \leq el(q) \leq nn_M(q)/c_1$. Hence $f(p) \leq c_3 * f(q)$, where $c_3 = 1 + 1/(l * c_1)$. ■

Similarly, for any point x in a triangle element $q_1q_2q_3$, we have $f(x) \leq c_3 f(q_i)$, for $i = 1,2,3$.

Lemma 4 *Let $q_1q_2q_3$ be a triangle element of M. Let p be a mesh point other than q_1, q_2, q_3. Let x be a point inside the triangle. Then there exists a constant c_4, depending only on the smallest angle θ of M, such that*

$$||p-x||/min_{1\leq i\leq 3}||p-q_i|| \geq c_4.$$

Proof. There are two cases for the nearest point in the triangle to p:

- one of $\{q_1, q_2, q_3\}$; In this case, we have $||p-x||/min_{1\leq i\leq 3}||p-q_i|| \geq 1$.
- a boundary point other than q_1, q_2 or q_3. W.l.o.g., assume q_1q_2 separates p from q_3. Let x_0 be the closest point on the segment q_1q_2 to p. See Figure 2. If p is directly connected to q_1 and q_2 in the mesh, then $||p-x_0||/||p-q_i|| \geq \tan(\theta)$, where $i = 1,2$, and θ is the lower bound on the element angle. Otherwise, assume p_0 is the mesh point other than q_3 directly connected to q_1 and q_2 in the mesh. Either p_0q_1 separates p from q_2 or p_0q_2 separates p from q_1 or both. W.l.o.g., assume p_0q_2 separates p from q_1. Then we have $||p-x_0||/||p-q_2|| \geq \tan(\theta)$, which implies that $||p-x_0||/min_{1\leq i\leq 3}||p-q_i||$ is at least $\tan(\theta)$. The lemma follows from the fact that $||p-x|| \geq ||p-x_0||$.

In both cases, we have

$$||p-x||/||min_{1\leq i\leq 3}||p-q_i|| \geq min(1, \tan(\theta)).$$

■

Lemma 5 *Let x be a point in a triangle element $q_1q_2q_3$ of a well-shaped M. The following is true for the global spacing function: there exists a constant c_5, depending only on the smallest angle θ of M, and the lower bound l on δ, such that*

$$f(x)/min_{1\leq i\leq 3}f(q_i) \geq c_5.$$

Proof. From the definition of f, there exists a mesh point p such that $f(x) = \delta(p) * nn_M(p) + ||x - p||$. If p is one of q_1, q_2, q_3, then

$$f(x) = \delta(p) nn_M(p) + ||x - p|| \geq \delta(p) nn_M(p) \geq min_{1 \leq i \leq 3} f(q_i).$$

Otherwise, let q be the q_i with the minimum distance to p. We have $f(q) \leq \delta(p) * nn_M(p) + ||q - p||$. If $||x - p|| \geq ||q - p||$, then

$$f(x) = \delta(p) * nn_M(p) + ||x - p|| \geq f(q).$$

Otherwise, $f(x)/f(q) \geq (\delta(p) * nn_M(p) + ||x - p||)/(\delta(p) * nn_M(p) + ||q - p||) \geq ||x - p||/||q - p|| \geq c_4$. The last inequality is given in Lemma 4. In both cases, we have $f(x)/f(q) \geq min(1, c_4)$. From Lemma 3, we have $f(q)/min_{1 \leq i \leq 3} f(q_i) \geq 1/c_3$. Hence

$$f(x)/min_{1 \leq i \leq 3} f(q_i) \geq min(1, c_4)/c_3 = min(1, \tan(\theta))/c_3.$$

∎

Let $B(x, r)$ be the sphere of radius r centered at point x. We will use the following notion of sphere packing [7, 15] in our algorithm.

Definition 2 (β-Packing) *Let β a positive real constant. A set S of spheres is a β-packing with centers P of Ω with respect to a spacing function f if*

- *For each point p of P, $B(p, f(p)/2) \in S$;*
- *The interiors of any two spheres s_1 and s_2 in S do not overlap; and*
- *For each point $q \in \Omega$, there is a sphere in S that overlaps with $B(q, \beta * f(q)/2)$.*

To construct the mesh points for M', we first use the following procedures to generate a β-packing of Ω with respect to f by using as many mesh points from M as possible. Here β is a constant to be given later. The mesh M' is the Delaunay triangulation of the centers of the resulting β-packing.

Algorithm `Functional-Refining-Functional-Coarsening`

1. Let $S_1 = \{B(p, f(p)/2) | p \in M\}$;
2. For each triangle element $t = (q_1 q_2 q_3)$ in M, let q be the mesh point q_i with the smallest $f(q_i)$. Let b_t be the smallest box that contains t. We divide b_t into a set of uniform cells with the side length $c_5 * f(q)/(2\sqrt{2})$, where c_5 is a constant given in Lemma 5. See Figure 3. Choose a random point in every cell that intersects t for a nonempty area, and for each such a point x, define a sphere with center x and radius $f(x)/2$. [2] Let S_2 be the set of these spheres;
3. Let $S' = S_1 \cup S_2$;
4. Order the sphere in S' as the following: all spheres whose centers are on the boundary come first, followed by all other spheres in S_1 in the order of increasing radii, followed by all spheres in S_2 in the order of increasing radii;
5. We say two spheres s_1 and s_2 in S' are *conflicting* if their interiors overlap. The conflicting relation defines a *Conflict Graph* (CG) over S'. Let S be the set of spheres which form the *Lexical-First Maximal Independent Set* (MIS) of CG;
6. Let M' be the mesh defined by the constraint Delaunay Triangulation of centers of S.

The lexical-first MIS is defined as the following. The initial MIS is empty. Then we add a sphere with the smallest index that does not conflict with any spheres of the existed MIS until no sphere can be added. The intuition is that we try to conform the boundary, and use as many spheres as possible. In addition the smaller sphere has higher priority to be chosen.

The basic idea of `Functional-Refining-Functional-Coarsening` *FRFC* is to first compute a maximum spacing function that satisfies the new spacing requirement of the Evolving Mesh Problem. We then make use of the point set of M to construct a sphere packing with respect to the spacing function, and hence the point set of the new mesh. Then the new mesh is obtained by using Delaunay triangulation. The maximality of the spacing function f is given in the following lemma.

[2] In practice, we can use $(f(q) + ||x - q||)/2$ to approximate the radius $f(x)/2$. Note that $c_5 f(q) \leq f(x) \leq f(q) + ||x - q|| \leq c_3 f(q) \leq c_3/c_5 f(x)$. In other words, $(f(q) + ||x - q||)/f(x)$ has constant lower and upper bound. Hence, it is reasonable to use $f(q) + ||x - q||$ to approximate $f(x)$.

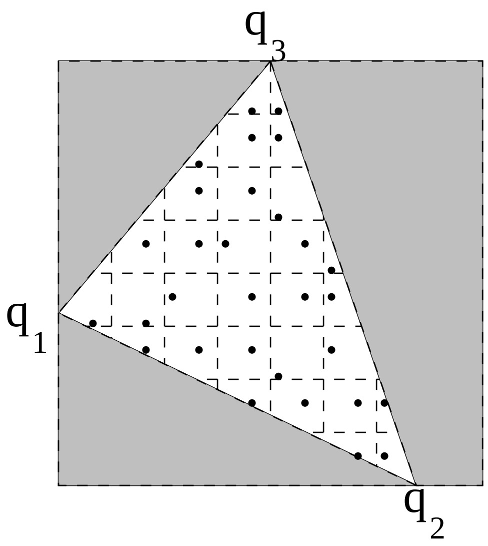

Figure 3: Sampling points in a triangle element.

Lemma 6 (Maximality) *Let g be any spacing function of Lipschitz constant 1 over the domain Ω that satisfies the condition $g(p) \leq \delta(p)nn_M(p)$ for all mesh point p in M. Then for any point q in Ω, not necessarily a mesh point of M, $g(q) \leq f(q)$.*

Proof. Let $g_p(x) = g(p) + ||x-p||, g'(x) = min(g_p(x))$. Then $g'(x) \leq f(x)$, for all x. Now assume P_x is the point that drives x to get the smallest value for g', i.e., $g'(x) = g_{P_x}(x) = g(P_x) + ||P_x - x||$. Note that $g(x) \leq g(P_x) + ||P_x - x||$ because g is 1-Lipschitz function. Then we have $\forall x, g(x) \leq g'(x) \leq f(x)$. ∎

Therefore, let M'' be any mesh that satisfies the condition of the Evolving Mesh Problem. We have for any point q in Ω, $nn_{M''}(q) \leq f(q)$, because $nn_{M''}$ is 1-Lipschitz function, and $nn_{M''}(p) \leq \delta(p)nn_M(p)$.

Lemma 7 (Number of Sample Points) *The number of sampled points in a triangle element $q_1q_2q_3$ is bounded by $2\sqrt{2}/(c_5 * l)$.*

Proof. Note that the number of cells generated in the triangle element is no more than $nn_M(q_i) * 2\sqrt{2}/(c_5 * f(q_i))$. The lemma follows from Lemma 2. ∎

Lemma 8 (Dense Sample) *For any point x in the domain, the sphere $B(x, f(x)/2)$ contains at least one point from $S_1 \cup S_2$.*

Proof. It is sufficient to show a stronger statement that is $B(x, f(x)/2)$ contains at least one cell generated during the sampling procedure. Let $t = q_1q_2q_3$ be the triangle element that contains x. Let q be a mesh point q_i with the smallest $f(q_i)$, $i = 1, 2, 3$. The side length of the cell generated during the sampling procedure is $c_5 * f(q)/(2\sqrt{2})$, where c_5 is given in Lemma 5. If the radius of the sphere is at least the diagonal of the cell, i.e., $f(x)/2 \geq c_5 * f(q)/2$, then the sphere will contain a cell. This is true by Lemma 5 that $f(x) \geq c_5 * f(q)$. ∎

The following lemma is from Miller *et al* [5].

Lemma 9 *Let P be a set of points in domain Ω. Let g be an α-Lipschitz function defined on Ω. Let $S = \{B(p, g(p))|p \in P\}$. If for any point $x \in \Omega$, $B(x, g(x))$ contains at least one point from P, then the maximal-independent-set of the conflict graph of S is $(3 + \alpha)/(1 - \alpha)$-packing.*

Theorem 1 *The S returned by the* FRFC *algorithm is a 7-packing.*

Proof. Note that the spacing function $f/2$ used for spheres is 1/2-Lipschitz. The lemma follows from Lemma 9. ∎

Let Γ be a collection of spheres. The *ply* of a point x, denoted by $ply(x, \Gamma)$, is the number of the spheres in Γ that contains x.

Lemma 10 (Constant Ply) *For $S' = S_1 \cup S_2$, there exists a constant c_6 such that for any point $x \in \Omega$,*

$$ply(x, S') \leq c_6.$$

Proof. Let $C(x)$ be the set of the centers of the spheres in S' that contain x. For any point $c \in C(x)$, we have $2||x - c|| \leq f(c)$. Note that $f(c) \leq f(x) + ||x - c||$ because f is 1-Lipschitz function. Hence $||x - c|| \leq f(x)$, i.e., point c is in $B(x, f(x))$. Let $t = (q_1 q_2 q_3)$ be a triangle element that contains x. Consider the set of triangles, $N(t)$ (stands for neighbors), that are incident to t. The size of $N(t)$ is at most $4\pi/\theta$, where θ is the minimum angle of the mesh. Let $E(t)$ (stands for exteriors) be $\{t' | t' \cap B(x, f(x)) \neq \emptyset$ and $t' \notin N(t) \cup \{t\}\}$. It is sufficient to show that the size of $E(t)$ is bounded by a constant. Let $p_1 p_2 p_3$ be a triangle in $E(t)$ that contains a point $c \in C(x)$. We have $c_3 * f(p_i) \geq f(c) \geq 2 * ||x - c||$. The first inequality is from Lemma 3. Note that $Lnn_M(p_i) \geq \delta(p_i) * nn_M(p_i) \geq f(p_i)$. From Lemma 4, we have $||x - c|| \geq c_4 * min_{q_j}(||c - q_j||)$. Let q be the point with the minimum $nn_M(q_j)$, j=1,2,3. Hence, $||x - c|| \geq c_4 * nn_M(q)$. So $nn_M(p_i) \geq f(p_i)/L \geq f(c)/(L * c_3) \geq (2c_4/(L * c_3)) * nn_M(q)$. By a volumn argument, $f(x) \leq c_3 * f(q) \leq c_3 * L * nn_M(q)$ implies that the number of triangles in $E(t)$ is bounded by a constant. Lemma follows from the fact that the number of points in each triangle is bounded by a constant. ∎

We now analyze the time complexity of the algorithm. The time to compute the global spacing function f is $O(|M| \log(|M|))$. Notice that the mesh point p_j that defines $f(p)$ has the property that for all mesh point $p_k \in M$:

$$\delta(p_j) * nn_M(p_j) + ||p_j - p|| \leq \delta(p_k) * nn_M(p_k) + ||p_k - p||.$$

That is, p_i is contained in the additively weighted Voronoi cell of p_j. Fortune [3] shows how to apply the sweep-line technique to compute the additively weighted Voronoi diagram in $O(|M| \log(|M|))$ time.

The time complexity of step 1 of the algorithm FRFC is $O(|M| \log(|M|))$. During step 2, the number of the points sampled at any triangle element $q_1 q_2 q_3$ is bounded by $nn_M(q_i) * 2\sqrt{2}/(c_5 * f(q_i))$. Lemma 2 implies that the number of sampled points is at most $2\sqrt{2}/(c_5 * l)$. Hence, the time complexity of step 2 is also $O(|M| \log(|M|))$. Note that the time complexity is $O(|M|)$, if we use $(f(q) + ||x - q||)/2$ to approximate $f(x)$. The time to sort all the spheres during step 4 is $O(|M| \log(|M|))$. Because the maximum ply of any point in Ω with respect to $S_1 \cup S_2$ is bounded by a constant (Lemma 10), we can apply the sphere-separator based divide-and-conquer algorithm [6] to construct the conflict graph in $O(|M| \log |M|)$ time. In addition, we know that the conflict graph has at most $O(|M|)$ number of edges. Computing the maximal-independent-set of the conflict graph then takes $\Omega(|M|)$ time. The Delaunay triangulation takes $O(|M'| \log(|M'|))$ time, where $|M'|$ is linear in $|M|$, because the total number of the sampled points is linear in $|M|$. Therefore, the time complexity of FRFC is $O(|M| \log(|M|))$.

4 Size and Quality

We now show that FRFC returns a mesh M' that is well-shaped, and is of a size that is within a constant factor of the optimal possible. We will use the following structure theorem of Miller, Talmor, and Teng [7] which states an equivalent relationship between β-sphere packing and well-shaped meshes.

Theorem 2 (Sphere Packing and Well-Shaped Meshes) 1. *For any positive constant β, there exists a constant α depending only on β such that if f is a spacing function of Lipschitz constant 1 over a domain Ω and S is a β-sphere packing with respect to f, then the Delaunay triangulation M of the centers of S is an α well-shaped mesh; in addition, for each point p in Ω, $nn_M(p) = \Theta(f(p))$, where the constant in Θ depends only on β.*

2. *For any positive constant α, there exists a constant β depending only on α such that if M is an α well-shaped mesh, then the set of spheres*

$$S = \{B(p, nn_M(p)/2) : \text{for all mesh point } p \in M\},$$

is a β-packing with respect to $nn_M/2$.

Therefore, there exists a constant α such that the mesh M returned by the algorithm is α-well-shaped. Note that the point set S returned by the algorithm is 7-packing with respect to $f/2$. The size optimality follows from the fact that f is a maximum spacing function that satisfies the condition of the Evolving Mesh Problem, (see Lemma 6), and the following lemma of [7].

Lemma 11 (Size of a Well-shaped Mesh) *If M is an α-well-shaped mesh of n elements, then*

$$n = \Theta(\int_{\Omega} \frac{dA}{nn_M^2}).$$

First, notice the number of the spheres in S is bounded by

$$\Theta(\int_{\Omega} \frac{dA}{f^2}),$$

by a simple volume argument. Because f is point-wise larger than the nn function of any well-shaped mesh that satisfies the Evolving Mesh Problem. It follows that size M is within a constant factor from the best possible. Therefore,

Theorem 3 (Main) FRFC *constructs a well-shaped mesh that satisfies the spacing condition given by δ. In addition, its size is optimal within a constant factor.*

The key to our algorithms in maintaining the well-shaped condition is to make sure that the shape condition does not deteriorate from M to M'. This is why we add new sampling points to regions near C-points to ensure the constant β in β-packing is maintained. Miller *et al* [7, 8] showed that in their coarsening algorithm that no new point is needed for coarsening. However, to do so, they need to use the original finest mesh directly to generate the coarsening mesh at each level. In other words, if the original mesh is M_0, then mesh point of M_0 are used to build the conflict graph to generate the mesh points for M_i. If they simply use mesh points of M_{i-1}, then mathematically, they can not guarantee the mesh points of M_{i-1} are dense enough for the well-shaped condition through the quality of the packing for M_i. For EMP, because of the mixed refinement, the original mesh does no longer provide fine enough sample points to guarantee the packing condition. Hence, new points has to be added. Our objective here is to add as small number of new points as possible, and meanwhile, by using as simple procedures as possible. In practice, for the moving boundary problem, there is no need to add new sample points to the back region of the moving boundary. We can use the algorithm of Miller *et al* [7, 8] to coarsen the back region.

5 Practical Variations

The δ values decompose the mesh M into a collection of components of maximal submeshes where the δ values of all mesh points in each submesh are either larger than 1 (type C-submeshes), or smaller than 1 (type R-submeshes). In practice the number of such submeshes is bounded by a small constant. For example, this number in most problems with a moving boundary is 2 (one for the front-end of the moving boundary, one for the back-end). As observed in Section 3, we need to insert Steiner points in the submeshes that are required for refinement. From submeshes to be coarsened, we often need to remove some original mesh points. A practical variation of our scheme is to first refine the R-submesh by any adaptive refinement algorithm, such as quad/octree refinement and Delaunay refinement. Then we apply the one-level coarsening algorithm of Miller, Talmor, and Teng [7]. We now present a detailed procedure for the case where Delaunay refinement is used. Recall that the standard Delaunay refinement procedure contains three rules [12, 13]:

1. splitting boundary subsegment whose diametral sphere contains a mesh point other than its end-points in its interior by adding a Steiner point at its midpoint;
2. splitting a boundary subfacet whose equatorial sphere contains a non-coplanar mesh point by adding a Steiner point at its circum-center. However, if the new point would cause any subsegment of the subfacets to split, apply rule one to these subsegments instead.
3. splitting any simplex that does not satisfies the well-shape condition by adding a Steiner point at its circum-center. However, if the addition of this circum-center would cause any subsegment or subfacet to split, then apply rules 1 and/or 2 instead.

We add a fourth rule, which states as: splitting any simplex in which the nn-spacing of any one of its mesh points is more than its delta-value times its *nn*-spacing by adding a Steiner point at its circum-center. However, if the addition of this circum-center would cause any subsegment or subfacet to split, then apply rules 1 and/or 2 instead.

Algorithm `Delaunay-Refining-Functional-Coarsening`
Input: A well-shaped mesh M and a list of positive reals δ.

1. Apply rules 1, 2, 3, 4 until all constraints on the spacing and shape at each mesh point are satisfied. Call the resulting mesh M_I.
2. Apply the one-level coarsening method of Miller, Talmor, and Teng [7] to M_I with coarsening factors given in δ to M_I to construct M'.

The following theorem follows directly from the main theorem of Ruppert [12] for 2D and of Shewchuk [13] for 3D and the coarsening result of Miller *et al* [7].

Theorem 4 `Delaunay-Refining-Function-Coarsening` *(DRFC) constructs a well-shaped mesh that satisfies the spacing condition given by* δ.

One of the shortcomings of `DRFC` is that it may construct a mesh that is larger than necessary. The reason is that in the refinement, we did not remove any original mesh point. In `FRFC`, we may replace some original mesh points in the R-submeshes by new Steiner points, which potentially reduce the mesh size. However, `DRFC` is in general more efficient.

When the lower bound l on δ is very small, the number of points introduced in each triangle could be very large, although it is a constant. Especially for the coarsening regions, (e.g., backend of a moving boundary), this is undesirable. In practice, we have a few alternatives:

- Do not add any points to triangles all of whose mesh points are C-points.
- Add only the barrycenter and/or midpoints of the edges rather than generating random quasi-uniform points based on the local grid.

Talmor [15] showed in her thesis that in practice no new point is needed for the region to be coarsened repeatly. We will conduct more experiments to verify this point in the context of EMP.

6 Conclusion

In this paper, we present a unified approach for coarsening and refining evolving meshes. One application and motivation of our work is for solving time-dependent problems with a moving boundary. In our future work, we will explore the structure of the moving boundary and level sets to speed up the coarsening and refinement procedure. We will also work on incorporate our algorithm into some standard mesh generation software. We will present some experimental results during the conference to show the effectiveness of our algorithm and its practical variations. In addition, all of the lemmas and theorems are applied to three dimensions if the aspect-ratio is used as shape criterion of the well-shaped mesh. However, this does not prohibit the existence of slivers.

In the context of parallel implementation of the Evolving Mesh Problem, the need of mesh evolution could introduce load imbalance among processors, where the load measures the amount of work required by solving the Evolving Mesh Problem itself as well as by numerical calculations thereafter. We need to develop a mesh distribution estimation algorithm to incorporate with the dynamic load balancing scheme developed in [4].

References

[1] I. Babuška and A.K. Aziz. On the angle condition in the finite element method. *SIAM J. Numer. Anal.*, 13(2):214–226, 1976.

[2] M. Bern, D. Eppstein and J. R. Gilbert. Provably good mesh generation. In *31st Annual Symposium on Foundations of Computer Science, IEEE*, 231–241, 1990.

[3] S. Fortune. A sweep line algorithm for Voronoi diagrams. *Algorithmica*, 2:153-174, 1987.

[4] X.-Y. Li and S.-H. Teng. Dynamic load balancing for parallel adaptive mesh refinement. In *5th International Symposium on Solving Irregularly Structured Problems in Parallel*, Berkeley, 144-155, 1998.

[5] G. L. Miller, D. Talmor, S.-H. Teng. Data Generation for Geometric Algorithms on Non-Uniform Distributions. *International Journal of Computational Geometry and Applications*, accepted and to appear, 1998.

[6] G. L. Miller, S.-H. Teng, W. Thurston, and S. A. Vavasis. Automatic mesh partitioning. In A. George, J. Gilbert, and J. Liu, editors, *Sparse Matrix Computations: Graph Theory Issues and Algorithms*, IMA Volumes in Mathematics and its Applications. Springer-Verlag, pp57–84, 1993.

[7] G. L. Miller, D. Talmor, S.-H. Teng. Optimal coarsening of unstructured meshes. In *Journal of Algorithms*, invited and accepted to a special issue for SODA 97.

[8] G. L. Miller, D. Talmor, S.-H. Teng. Optimal Good Aspect Ratio Coarsening for Unstructured Meshes. In *8th Annual ACM-SIAM Symposium on Discrete Algorithms*, pp 538–547, January 1997. ACM-SIAM.

[9] G. L. Miller, D. Talmor, S.-H. Teng, and N. Walkington. A Delaunay based numerical method for three dimensions: generation, formulation, and partition. In *Proc. 27th Annu. ACM Sympos. Theory Comput.*, pages 683–692, 1995.

[10] G. L. Miller, D. Talmor, S.-H. Teng, and N. Walkington. On the radius–edge condition in the control volume method. *SIAM J. on Numerical Analysis*, accepted and to appear, 1998.

[11] S. A. Mitchell and S. A. Vavasis. Quality mesh generation in three dimensions. Proc. ACM Symposium on Computational Geometry, pp 212-221, 1992.

[12] J. Ruppert. A new and simple algorithm for quality 2-dimensional mesh generation. In *Third Annual ACM-SIAM Symposium on Discrete Algorithms*, 83–92, 1992.

[13] J. R. Shewchuk. Tetrahedral mesh generation by Delaunay refinement. In *14th Annual ACM Symposium on Computational Geometry*, 86–95, 1998.

[14] G. Strang and G. J. Fix. *An Analysis of the Finite Element Method*, Prentice-Hall, 1973.

[15] D. Talmor. Well-Spaced Points for Numerical Methods. Ph.D thesis, Carnegie Mellon, 1997.