Discrete Comput Geom 14:411–428 (1995)

Discrete & Computational
Geometry

Linear-Size Nonobtuse Triangulation of Polygons*

M. Bern,[1] S. Mitchell,[2] and J. Ruppert[3]

[1] Xerox Palo Alto Research Center, 3333 Coyote Hill Rd.,
Palo Alto, CA 94304, USA
bern@parc.xerox.com

[2] Applied and Numerical Mathematics Department, Sandia National Laboratories,
Albuquerque, NM 87185, USA
samitch@sandia.gov

[3] IBM Almaden Research Center, 650 Harry Road,
San Jose, CA 95120, USA
ruppert@almaden.ibm.com

Abstract. We give an algorithm for triangulating n-vertex polygonal regions (with holes) so that no angle in the final triangulation measures more than $\pi/2$. The number of triangles in the triangulation is only $O(n)$, improving a previous bound of $O(n^2)$, and the running time is $O(n \log^2 n)$. The basic technique used in the algorithm, recursive subdivision by disks, is new and may have wider application in mesh generation. We also report on an implementation of our algorithm.

1. Introduction

The triangulation of a two-dimensional polygonal region is a fundamental problem arising in computer graphics, physical simulation, and geographical information systems. Most applications demand not just any triangulation, but rather one with triangles satisfying certain shape and size criteria [8]. In order to satisfy these criteria, one typically allows triangles to use new vertices, called *Steiner points*, that are not vertices of the input polygon. The number of Steiner points should not be excessive, however, as this would increase the running time of computations.

* The research of S. Mitchell was supported by the Applied Mathematical Sciences program, U.S. Department of Energy Research and by the U.S. Department of Energy under Contract DE-AC04-76DP00789. J. Ruppert's work was performed while he was at the NASA Ames Research Center as an employee of Computer Sciences Corporation, under NASA Contract NAS 2-12961.

Throughout the application areas named above, it is generally true that large angles (that is, angles close to π) are undesirable. Babuška and Aziz [1] justified this aversion for one important application by proving convergence of the finite element method [28] as triangle sizes diminish, as long as the maximum angle is bounded away from π. They also gave an example in which convergence fails when angles grow arbitrarily flat. (An elementary example in which large angles spoil convergence is Schwarz's paradox [23].)

Any bound smaller than π implies convergence in Babuška and Aziz's model, but a bound of $\pi/2$ on the largest angle has special importance. First, any stricter nonvarying requirement would also bound the smallest angle away from zero; for some inputs (such as a long, skinny rectangle) this forces the triangulation to contain a number of triangles dependent on the geometry—not just on the combinatorial complexity—of the input. Second, a nonobtuse triangulation is necessarily a (constrained) Delaunay triangulation [7]. Third, a nonobtuse triangulation admits a *perpendicular planar dual*, that is, an embedding in which dual edges cross at right angles. Such an embedding is convenient for the "finite volume" method [28]. Finally, a nonobtuse triangulation has better numerical properties [2], [31]. In particular, Vavasis [31] recently proved that, for simulation problems with physical characteristics that vary enormously over the domain, a nonobtuse mesh implies faster convergence of a certain numerical method.

These properties have established nonobtuse triangulation as a desirable goal in mesh generation. Several heuristic methods have been developed to compute nonobtuse triangulations [3], [21]. Baker *et al.* [2] gave the first provably correct algorithm. Their algorithm also bounds the smallest angle away from zero, and hence necessarily uses a number of triangles dependent upon input geometry. Melissaratos and Souvaine [17] gave another algorithm of this type.

From the point of view of theoretical computer science, however, it is important to determine the inherent complexity of nonobtuse triangulation, apart from no-small-angle triangulation. Bern and Eppstein [7] devised a nonobtuse triangulation algorithm using $O(n^2)$ triangles, where n is the number of vertices of the input domain. This result demonstrates a fundamental complexity separation between bounding large angles and bounding small angles. Bern *et al.* [6] later improved this bound to $O(n^{1.85})$ for convex polygons.

In this paper we improve these bounds to linear, using an entirely different—and more widely applicable—technique. Aside from sharpening the theory, our new algorithm boasts other advantages: it parallelizes, thereby placing nonobtuse triangulation in the class $\mathcal{NC}$; and it does not use axis-parallel grids, so the output has no preferred directions. Our algorithm also improves results of Bern *et al.* [6] on no-large-angle triangulation. The superseded results include an algorithm guaranteeing a maximum angle of at most $5\pi/6$ that uses $O(n \log n)$ triangles for simple polygons and $O(n^{3/2})$ triangles for polygons with holes.

2. Overview of the Algorithm

Our algorithm consists of two stages. The first stage (Section 3) packs the domain with nonoverlapping disks, tangent to each other and to sides of the domain. The

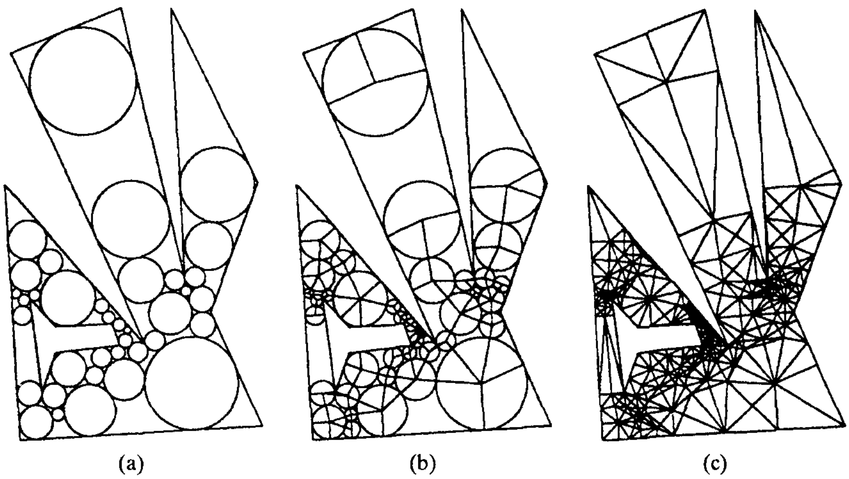

Fig. 1. (a) Disk packing. (b) Induced small polygons. (c) Final triangulation.

disk packing is such that each region not covered has at most four sides (either straight sides or arcs), as shown in Fig. 1(a). The algorithm then adds edges (radii) between centers of disks and points of tangency on their boundaries, thereby dividing the domain into small polygons as shown in Fig. 1(b).

The second stage (Section 4) triangulates the small polygons using Steiner points located only interior to the polygons or on the domain boundary. Restricting the location of Steiner points ensures that triangulated small polygons fit together so that neighboring triangles share entire sides. Certain misshapen small polygons cause technical difficulties; these are neatly solved by packing in more disks. (One of these additional disks is the second from the left along the bottom side of Fig. 1(b).) Figure 1(c) shows the resulting nonobtuse triangulation.

This algorithm is circle-based, rather than grid-based like the previous polynomial-size nonobtuse triangulation algorithm [7]. Analogously, the problem of no-small-angle triangulation has grid-based [9] and circle-based [24] solutions. In retrospect, circle-based algorithms offer a more natural way to bound angles, as well as meshes more intrinsic to the input domain. The nonobtuse meshes of this paper are related to power diagrams and regular triangulations [11]; more precisely, away from the polygon boundary the mesh is the power diagram of the packed disks superimposed with its dual, the regular triangulation. In other recent work, Mitchell uses the "angle buffering" property of circles to give a triangulation, restricted to using only interior Steiner points, with linear size and largest angle nearly as small as possible [19].

3. Disk Packing

In this section we describe the first stage of the algorithm. Let P denote the input: a region of the plane bounded by a set of disjoint simple polygons with a total of n

vertices. An *arc polygon* is a simple polygon with sides that are arcs of circles. The circles may have various radii, including infinity, meaning a straight side.

Throughout the disk-packing stage, we make use of the *generalized Voronoi diagram* (GVD), which is defined by proximity to both edges and vertices. The interior points of polygonal region P are divided into cells according to the nearest vertex of P, or the nearest edge (viewing each edge as an open segment). The resulting partition consists of a set of bisectors, either line segments or parabolic arcs; it is essentially the same as the *medial axis* [22]. The GVD can be similarly defined for arc polygons, or more generally for arbitrary collections of points, segments, and circular arcs. The GVD of a collection of n points, segments, and arcs can be computed in time $O(n \log n)$ using Fortune's sweep-line algorithm [14].

The disk-packing stage consists of three smaller steps. First, one or two disks are placed at each vertex of the polygon. Second, holes in the polygon are connected to the boundary by adding disks tangent to two holes, or to a hole and the outer boundary. Third, disks are added to the as-yet-uncovered regions (called *remainder regions*), recursively reducing their complexity until all have at most four sides.

Disks at Corners

The first step preprocesses P so that we need only consider arc polygons with angle zero at each vertex. At each convex vertex of P, we add a small disk tangent to both edges, as shown in Fig. 2(a). At each concave vertex of P, we add two disks of equal radii, tangent to the edges, and tangent to the angle bisector at the corner, as shown in Fig. 2(b). We can handle a point hole by centering a small disk on the hole. We choose radii small enough so that disks lie within P, and none *overlap* (that is, intersect at interior points). This step isolates a small three- or four-sided remainder region at each corner of P. The large remainder region is an arc polygon of $2n + r = O(n)$ sides, where n is the number of vertices of P and r is the number of concave corners.

The first step can be implemented in time $O(n \log n)$ using the GVD of P. By checking the adjacencies of GVD cells, we can determine the nearest nonincident edge for each vertex v of P; one-eighth this distance gives a safe radius for the disks next to v. (Our implementation actually uses maximal radii in order to reduce output size by a constant factor.)

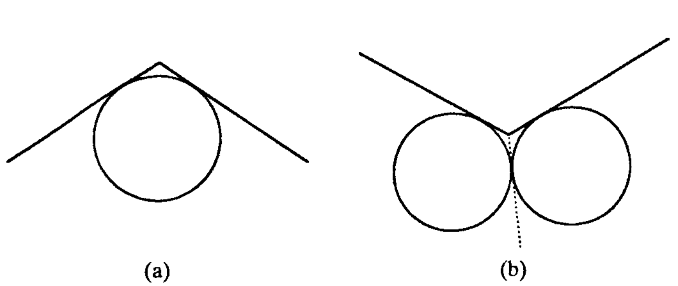

Fig. 2. Adding disks at (a) convex and (b) concave corners of polygonal region P.

Connecting Holes

The second step connects polygonal holes to the outer boundary by repeatedly adding a disk tangent to two or more connected components of the boundary. In this step, previous disks touching a hole boundary are considered to be part of the hole. At the end, the large remainder region is bounded by a simply connected arc polygon with $O(n)$ sides. Each corner of this arc polygon has angle zero, since each results from a tangency.

The second step can be implemented in time $O(n \log^2 n)$. We use a data structure that answers queries of the following form: given a query point p, which data object (straight edge or arc) will be hit first by an expanding circle tangent to a vertical line through p (tangent at p and to the left of the line)? Such a query can be answered using Fortune's $*$-map [14], a sort of warped Voronoi diagram.

The initial set of data objects consists of the edges of the outer boundary of P, along with all disks attached to this outer boundary. The first query point is the leftmost point on any hole. The answer determines a disk D entirely contained within the polygon, touching both the hole and the outer boundary. Disk D is inserted into the query data structure, along with the edges and disks of the hole. Each subsequent query is performed using the leftmost point of all remaining holes. Altogether, the queries yield a set of disks connecting all holes and the exterior of the polygon.

For a static set of data objects, the $*$-map can be built in time $O(n \log n)$ [14], and standard planar subdivision search techniques [22] yield $O(\log n)$ query time. In our case the set of data objects is not fixed, since disks and edges are added following each query. A trick due to Bentley and Saxe [4] allows dynamic insertions to the query structure, with query time $O(\log^2 n)$ and amortized insertion time $O(\log^2 n)$. The trick is to divide the $O(n)$ data objects among $O(\log n)$ $*$-maps of varying sizes. A query searches all data structures in $O(\log^2 n)$ time. An insertion rebuilds all the data structures corresponding to bits that change. The key idea is to divide the n data objects according to the binary representation of n, for example, if $n = 19$, there will be one $*$-map containing sixteen objects, another containing two, and another containing only one object. The next insertion will throw away the two smaller $*$-maps and combine their objects with the newly inserted object in a $*$-map of size four. Because the $*$-map containing 2^k objects is rebuilt only once in every 2^k insertions, the total time required for $O(n)$ insertions is $O(n \log^2 n)$.

The running time of the hole-connecting step determines the overall running time of our algorithm. Subsequent to the first appearance of our paper in the ACM Symposium on Computational Geometry, Eppstein [13] improved the running time of this bottleneck step to $O(n \log n)$. Eppstein's method computes a "minimum spanning tree" of the connected components of P's boundary; in this "tree" the boundary itself has weight zero. The method adds all diameter disks of MST edges and then shrinks these disks one by one in order to remove overlaps. Eppstein uses Sleator and Tarjan's dynamic trees [26] to implement this last step in total time $O(n \log n)$.

A suggestion of Goodrich and Tamassia (personal communication) simplifies this fast algorithm a bit. Perform the hole-connecting step before step one, and rather than shrinking the MST diameter disks, simply let them overlap. It is not hard to prove that the mutual chord of two overlapping MST diameter disks separates their centers. Now perform step one, placing small disks at corners, including corners formed by overlapping disks; this ensures that all remainder regions with more than three sides have zero-degree angles at vertices. These two conditions—center-splitting chords and zero-degree angles on four-sided remainder regions—turn out to be sufficient for our triangulation methods.

Reducing to Three- and Four-Sided Remainder Regions

After the first two steps, there is one simply connected remainder region A with $O(n)$ sides, and $O(n)$ remainder regions in corners with three or four sides. Arc polygon A has the property that, at each vertex, the two arcs form a zero-degree angle. The final step of the disk-packing stage recursively subdivides A by adding disks. The result is a linear number of remainder regions of three and four sides.

To subdivide arc polygon A, we add a disk tangent to three of its sides. Such a disk divides the region enclosed by the arc polygon into four pieces: the disk itself and three smaller regions bounded by arc polygons. We choose a disk tangent to three sides of A, not all of them consecutive, thereby ensuring that each of the three smaller arc polygons has at most $n - 1$ sides. As shown in Fig. 3, a disk tangent to three sides of an arc polygon must be centered at a vertex of the GVD. Since A is simply connected, the edges of its GVD form a tree, a fact that is useful in bounding the running time.

Lemma 1. *It is possible to reduce all remainder regions to at most four sides, by packing $O(n)$ nonoverlapping disks into arc polygon A.*

Proof. Each vertex of the GVD corresponds to a disk tangent to three sides of A. If A has at least five sides, then there is a vertex v of the GVD that is adjacent to two nonleaf vertices of the GVD, and a disk centered at v is tangent to three sides of A that are not all consecutive.

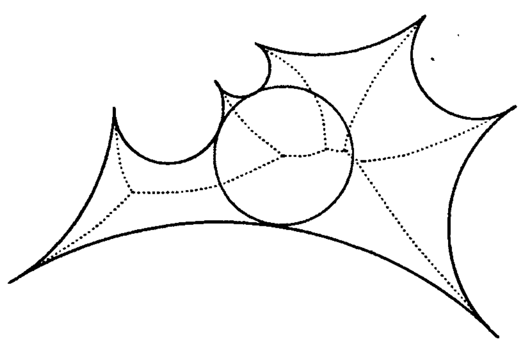

Fig. 3. A disk tangent to three edges of an arc polygon is centered at a vertex of the GVD.

Now let $d(n)$ be the maximum number of disks needed to reduce an n-sided arc polygon to three- and four-sided remainder regions. We prove $d(n) \leq n - 4$ by induction on n. The base cases are $d(3) = 0$ and $d(4) = 0$.

For the inductive step, notice that adding one disk produces three new arc polygons. (We can simply ignore extra tangencies in the degenerate case of four or more tangencies.) Suppose the new arc polygons have k, l, m sides, respectively, with $3 \leq k \leq l \leq m$. Since we are choosing nonconsecutive sides, $m < n$. Counting 1 for the added disk, we have that $d(n) \leq 1 + d(k) + d(l) + d(m)$. Since the disk divides three sides, and is itself divided into three places, we have $k + l + m = n + 6$.

First suppose $k = 3$. Since we are choosing nonconsecutive sides, $l \geq 4$, so

$$\begin{aligned} d(n) &\leq 1 + d(3) + d(l) + d(m) \\ &\leq 1 + 0 + (l - 4) + (m - 4) \\ &= (l + m) - 7 = (n + 3) - 7 = n - 4. \end{aligned}$$

When $k \geq 4$, we have $d(n) \leq 1 + d(k) + d(l) + d(m)$. By induction, $d(n) \leq 1 + (k - 4) + (l - 4) + (m - 4)$, which is equal to $(k + l + m) - 11 = (n + 6) - 11 = n - 5$. □

Finally, we show how to implement this last step of the first stage in time $O(n \log n)$. Any tree contains a vertex, called a *centroid*, whose removal leaves subtrees of size at most one-half the original size. By choosing a disk centered at a centroid of the GVD of A, we split A into arc polygons A_1, A_2, and A_3. We imagine splitting A_1, A_2, and A_3 in parallel, so that altogether there will be at most $\log_2 n$ splitting stages, each involving a set of arc polygons of total complexity $O(n)$. When a disk D is added to an arc polygon A with m sides, we can recompute the GVD of A and split it into the GVDs of A_1, A_2, and A_3 in time $O(m)$, simply by walking from cell to cell in the old GVD. We split each cell into two cells, one for the old arc and one for the arc on D, by adding a parabolic segment equidistant from D and the old arc.

4. Triangulating the Pieces

We now describe the second stage of our algorithm. At this point, polygonal region P has been partitioned into disks and remainder regions with three or four sides, either straight or circular arcs. Each circular arc of a remainder region R is naturally associated with a pie-shaped sector, namely, the convex hull of the arc and the center of the circle containing the arc. We denote the union of R and its associated sectors by R^+. These *augmented* remainder regions define a decomposition of P into simple polygons with disjoint interiors. In an augmented remainder region, we retain vertices at circle tangencies; vertices such as these, at which the angles measure π, are called *subdivision points*.

In this section we show how to triangulate each R^+ region. All Steiner points will lie either on straight sides of R (that is, along P's boundary) or interior to R^+. Thus

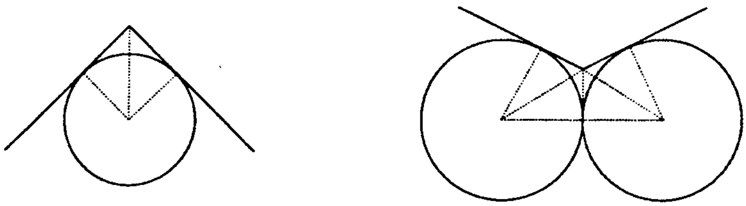

Fig. 4. Remainder regions with vertices of P.

we never place Steiner points on the radii bounding sectors, and triangulated R^+ regions will fit together at the end. Our triangulation method is given in three cases: remainder regions with vertices of P, three-sided remainder regions, and four-sided remainder regions. The first two cases are easy, but the last is quite intricate. In all cases, triangulating a single R^+ region takes $O(1)$ time, so altogether the running time of the second stage is $O(n)$.

Remainder Regions with Vertices of P

Each vertex of P was isolated by one or two disks in the first step of the algorithm. The resulting regions R^+ can be triangulated with at most four right triangles, as shown in Fig. 4, by adding edges from the disk centers to the points of tangency and the vertex of P.

Three-Sided Remainder Regions

A three-sided remainder region R without a vertex of P is bounded by three circular arcs that meet tangentially at the vertices of R. We can consider a straight side to be an arc of an infinitely large circle. We call a Steiner point in an augmented remainder region R^+ *safe* if it lies either interior to R^+ or on the boundary of P.

Lemma 2. *If R is a three-sided remainder region, then R^+ can be triangulated with at most six right triangles, adding only safe Steiner points.*

Proof. First assume that R has a straight side (necessarily at most one), and view R so that this straight side forms a horizontal base. The augmented region R^+ is a trapezoid with two vertical sides, and a subdivision point p along its slanted top side. We cut perpendicularly from p (that is, tangent to both arcs) across R until we hit the base, and there add a safe Steiner point s. We add edges from s to the centers of the arcs' circles to divide R^+ into four right triangles, as shown in Fig. 5(a).

Now assume all the sides of R are arcs of finite radius. Notice that R^+ is a triangle with subdivided sides. Moreover, the subdivision points along the sides of R^+ are exactly the tangency points of the inscribed circle of R^+. (This follows from the fact that the inscribed circle makes each corner of R^+ incident to two edges of

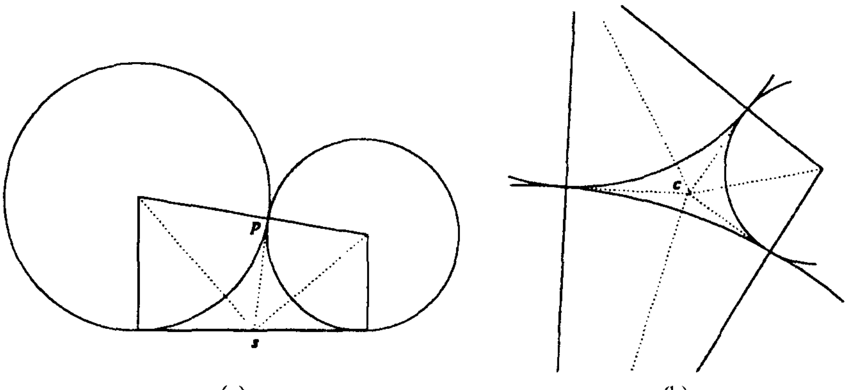

Fig. 5. Three-sided remainder regions: (a) with a straight side, (b) with only finite-radius arcs.

equal length.) So we add the circle's center c and edges from c to all the vertices around R^+, dividing R^+ into six right triangles, as shown in Fig. 5(b). □

Four-sided Remainder Regions

A four-sided remainder region R is bounded by four circular arcs, C_1, C_2, C_3, and C_4 in order around R, that meet tangentially at the vertices of R. A straight side is regarded as an arc of infinite radius. Lemma 3 states two interesting properties of these regions.

Lemma 3. *The arcs of R have total measure 2π. The vertices of R are cocircular.*

Proof. If all arcs have finite radius, then the sum of the measures of the arcs of R is identical to the sum of the measures of the angles at the corners of R^+. For straight sides, we imagine further augmenting R^+ with "infinite sectors" of angle zero.

Next we show that the vertices are cocircular. Let C_1 and C_3 be finite-radius circles containing opposite arcs of R. (Notice that if R has two straight sides, they must be opposite.) Assume the two lines that are externally tangent to both C_1 and C_3 meet at a point x. There is an inversive transformation [10, pp. 77–95] of the projective plane that maps x to infinity and hence the two external tangent lines to parallel lines. The transformed circles C_1' and C_3', corresponding to C_1 and C_3, have equal size, so the vertices of the transformed remainder region R' form an isosceles trapezoid. Any isosceles trapezoid has cocircular vertices. The inverse of the original inversive transformation maps the circle containing the vertices of R' to a circle containing the vertices of R. □

Now if we are lucky, the region R^+ can be triangulated with 16 right triangles, as in the following lemma.

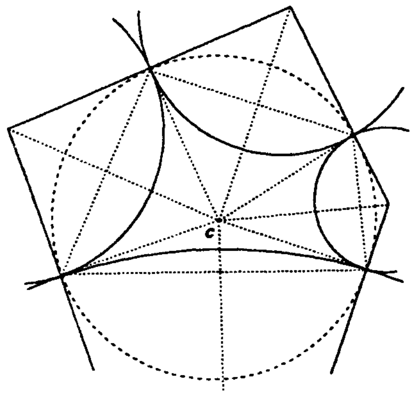

Fig. 6. The good case for four-sided remainder regions.

Lemma 4. *If R is a four-sided remainder region, in which each arc measures at most π and the center of the circle through R's vertices lies in the convex hull of R, then R^+ can be triangulated with* 16 *right triangles, adding only safe Steiner points.*

Proof. We assume that all arcs of R have finite radius. If R has a straight edge, we can apply the triangulation to a region with an infinite sector attached to the straight edge and then simply remove the resulting infinite strips.

The construction is shown in Fig. 6. Here we have added the center c of the circle through R's vertices in order to form four *kites* (quadrilaterals with two adjacent pairs of equal-length sides). □

The triangulation of Fig. 6 can fail in two different ways:

(1) If one of the arcs of R measures more than π (a *reflex* arc), then R^+ has a reflex vertex at which angles will measure more than $\pi/2$.
(2) If center c lies outside the convex hull of R, then it lies on the wrong side of one of the chords and will introduce unwanted intersections.

Each of these difficulties is handled by adding yet another disk.

First assume R has a reflex arc on circle C_3. Add another disk C^*, tangent to C_3 and C_1, such that the center of C^* lies on the line joining the centers of C_1 and C_3. The new disk C^*—unlike any of the disks used up until this point—may overlap C_2 or C_4 and produce a self-intersecting remainder region. In Fig. 7, C^* overlaps C_4. Notice however that C^* cannot overlap "too much": the mutual chord of C^* and C_4 separates their centers. On the other hand, the arc polygon formed by C_1, C_2, C_3, and C^* may still suffer from difficulty (2) above.

Lemma 3 holds without modification for self-intersecting remainder regions. Region R^+, formed as before by adding the associated pie-shaped sectors to R, remains a simple polygon with subdivision points on its sides, specifically a triangle with three subdivisions on one side and one on each of the others. The next lemma shows how to triangulate R^+ with a generalization of the method of Lemma 2.

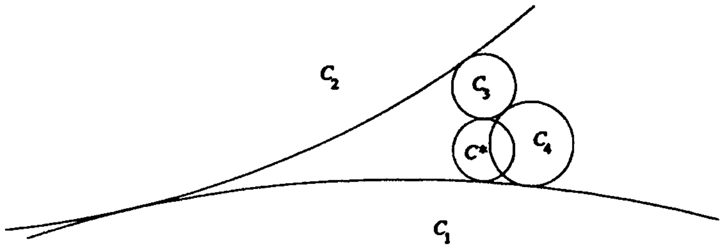

Fig. 7. New circle C^* breaks up a reflex remainder region.

Lemma 5. *Let R be a self-intersecting four-sided remainder region resulting from breaking up a reflex four-sided remainder region by the addition of C^*. Then R^+ can be triangulated with at most* 12 *right triangles, adding only safe Steiner points.*

Proof. Again we may assume that all arcs of R have finite radius, as a solution to this case implies a triangulation for the case of straight sides.

Consider one of the arcs S next to C^*. We claim that the lines tangent to S at its endpoints and the mutual chord of C^* and its opposite arc all meet at a single point p interior to R, as shown in Fig. 8(a). This claims allows the triangulation shown in Fig. 8(b).

Why is the claim true? For each of the three disks—C^*, the opposite disk, and the one with arc S—we define a *power* function. The power function of a circle with center (x_c, y_c) and radius r is $P(x, y) = (x - x_c)^2 + (y - y_c)^2 - r^2$. The power functions of two tangent circles are equal along their mutual tangent line; the power functions of two overlapping circles are equal along a line containing their mutual chord. The point p of the claim is the point at which all three power functions are equal. □

We now consider the second difficulty. Call a (possibly self-intersecting) four-sided remainder region R *centered* if the convex hull of R contains the center of the circle through R's vertices, and *uncentered* otherwise.

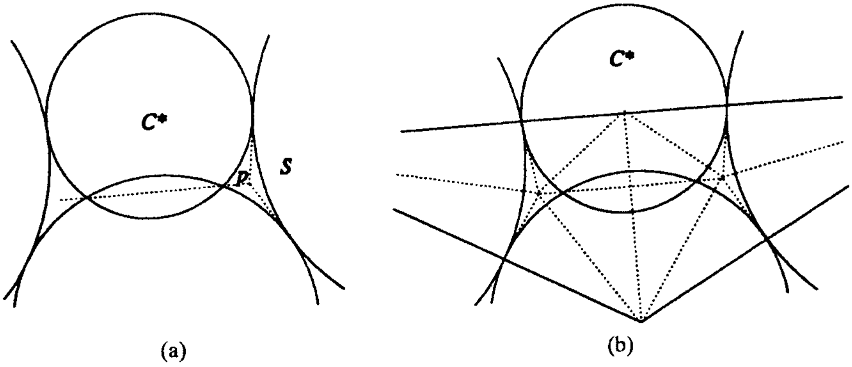

Fig. 8. (a) Mutual tangents and the mutual chord meet at a point. (b) Triangulation.

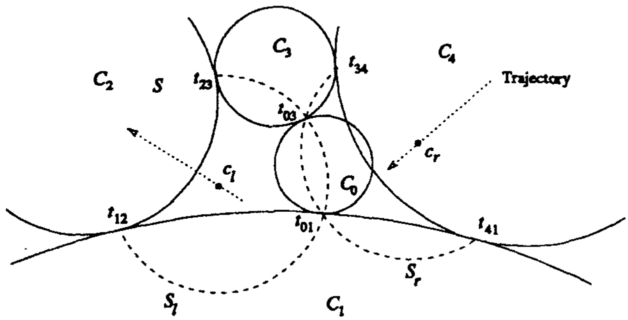

Fig. 9. The trajectories of centers c_l and c_r as C_0 sweeps.

Let R be an uncentered four-sided remainder region, with no boundary arc measuring more than π. Without loss of generality, assume that C_1 contains the arc of R with the longest chord and that the line through the centers of C_1 and C_3 is vertical, as in Fig. 9. Let t_{12} denote the point of tangency of C_1 and C_2, and similarly define t_{23}, t_{34}, and t_{41}.

Lemma 6. *A disk C^* tangent to C_1 and C_3 exists that breaks R into two centered, possibly self-intersecting, four-sided remainder regions. Such a C^* can be computed in time $O(1)$.*

Proof. Let C_0 be any disk tangent to both C_1 and C_3. Let t_{01} and t_{03} be, respectively, the points of tangency of C_0 and C_1 and of C_0 and C_3. Let S_l ($= S_l(C_0)$) be the circular arc, whose existence is guaranteed by Lemma 3, with endpoints t_{12} and t_{23} that passes through t_{03} and t_{01}. Similarly define S_r. Let c_l and c_r be the centers of the circles containing S_l and S_r, respectively.

Imagine sweeping C_0 through a continuum of positions, while keeping it tangent to C_1 and C_3. Consider a generic position of C_0, as shown in Fig. 9. The center c_r of arc S_r lies "outside" the chord $t_{34}t_{41}$ of S_r (that is, on the side away from t_{01} and t_{03}) exactly when S_r has measure less than π. Similarly, c_l lies outside the chord $t_{12}t_{23}$ of S_l exactly when S_l has measure less than π.

We assert that these two bad conditions cannot occur at the same time. It suffices to show that the sum of the measures of S_l and S_r is at least 2π. $\angle t_{23}t_{03}t_{34}$ measures at least $\pi/2$, because the arc of R on C_3 measures at most π. $\angle t_{12}t_{03}t_{41}$ measures more than $\pi/2$, because the center of the circle through the vertices of R lies below $t_{12}t_{41}$. Hence the remaining angles at t_{03}, the two subtended by the endpoints of S_l and S_r, sum to less than π, which implies that the arc measures of S_l and S_r sum to at least 2π.

We start the sweep with the center of C_0 on the line through the centers of C_1 and C_3. At this point, c_l and c_r lie on a horizontal line through the center of C_0, hence exterior to C_1 and C_3. However, c_l may lie outside chord $t_{12}t_{23}$ or c_r may lie outside chord $t_{34}t_{41}$. By the argument above, at most one of these bad conditions occurs. If neither occurs, then $C^* = C_0$ satisfies the conditions of the lemma, and we are done. However, if one of the bad conditions does occur, then we sweep C_0 in the direction that could cure the condition. If c_r lies outside $t_{34}t_{41}$, then we sweep C_0 to the left in Fig. 9; the other case is symmetrical.

During the leftward sweep, c_r moves toward C_1 along the perpendicular bisector of $t_{34}t_{41}$ and c_l moves toward C_2 along the perpendicular bisector of $t_{12}t_{23}$, as shown in Fig. 9. These bisectors never intersect C_3, so c_l and c_r can never lie on the wrong sides of their chords $t_{23}t_{03}$ and $t_{03}t_{34}$ on C_3. The chord of S_l on C_0 is shorter than $t_{12}t_{23}$ throughout the sweep, so c_l can never lie on the wrong side of $t_{01}t_{03}$. (Figure 9 may be a little misleading here; for illustrative purposes, it shows the leftward sweep with C_0 further to the right than its actual starting point.)

By the arc-measure argument above, c_r must hit $t_{34}t_{41}$ and become good before c_l crosses outside $t_{12}t_{23}$ and becomes bad. Thus, we can set C^* equal to the C_0 that places c_r on $t_{34}t_{41}$; at this position $\angle t_{41}t_{01}t_{34}$ is right. □

We apply Lemma 6 to each nonreflex, uncentered four-sided remainder region R. If C^* is centered on the line through the centers of C_1 and C_3, then R reduces to two nonreflex, centered, four-sided remainder regions, that can be triangulated using Lemmas 4 and 5. If this initial choice of C^* does not work (but for some reason it seems to always work!), then C^* creates a new reflex remainder region. The following lemma finesses this difficulty (shall we say circularity?) by triangulating both new augmented regions at once.

Lemma 7. *Let R be a nonreflex, uncentered, four-sided remainder region. Then R^+ can be triangulated into at most* 28 *right triangles, adding only safe Steiner points.*

Proof. Again we may assume that R has only finite-radius arcs, as this case implies a solution for the case of straight sides. We start by adding the "centering" disk C^* guaranteed by Lemma 6. If C^* is centered on the line through the centers of C_1 and C_3, we triangulate R^+ as mentioned above using Lemmas 4 and 5, giving at most 28 right triangles (in seven kites) as shown in Fig. 10(a). Otherwise, C^* places c_r on C_4's chord or c_l on C_2's chord. (Here we are using the notation of the proof of

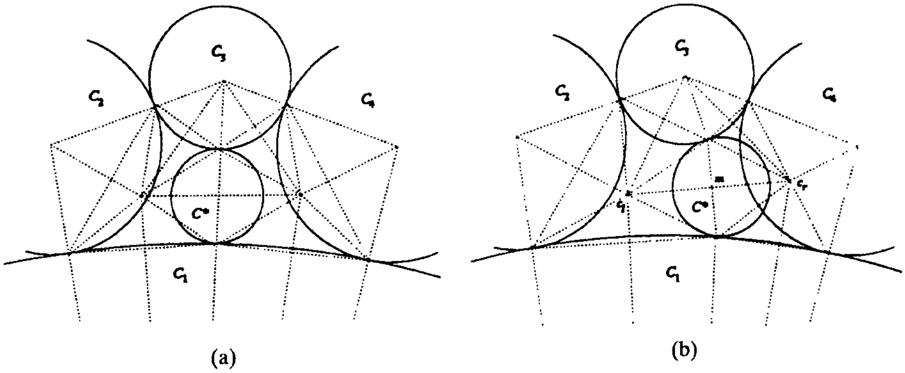

Fig. 10. Triangulations of uncentered four-sided regions, (a) when C^* lies in the center–center line, and (b) when c_r lies on $t_{34}t_{41}$.

Lemma 6.) Assume that c_r, the center of arc S_r, lies on C_4's chord; the other case is symmetrical.

The triangulation adds the following Steiner points: c_l and c_r, the points t_{01} and t_{03} where C^* is tangent to C_1 and C_3, and the midpoint m of segment $t_{01}t_{03}$. See Fig. 10(b).

The triangulation adds the following line segments: all chords around S_l and S_r; segments from c_l to m, to the points on S_l, and to the centers of C_1, C_2, and C_3; and segments from c_r to m, to the points on S_r, and to the centers of C_3, C_4, and C_1. Two more segments connect the center of C_1 with t_{01} and the center of C_3 with t_{*3}.

The resulting triangles form seven kites, but one of the kites—the one with diagonal running from c_r to the center of C_4—has degenerated to two triangles. All triangles are right. Notice that C^* is treated somewhat differently than the other circles, since we do not use its center. Nevertheless, the four triangles around m form a kite, because $t_{01}t_{03}$ is the mutual chord of C^*, S_2, and S_4. □

We have now completed the proof of our main theorem, linear-size nonobtuse triangulation.

Theorem 1. *Any n-vertex polygonal region can be triangulated with $O(n)$ right triangles in time $O(n \log n)$ for simple polygons and $O(n \log^2 n)$ for polygons with holes.* □

5. Implementation

We implemented our algorithm with the Matlab environment [16]. The implementation differs somewhat from the algorithm described in the text. We use several heuristics for disk placement in order to reduce the number of triangles. Also we do not bother to compute GVDs. Rather we use a simple $O(hn)$ method to connect h holes to the boundary, and we choose arbitrary disks touching three nonconsecutive sides, rather than disks centered at GVD centroids. To keep the user entertained during the worst-case $O(n^2)$ running time, we display color-coded disks and triangles as they are added. Finally, although Section 4 describes a construction using only right triangles, the implementation produces some acute triangles, an example being the large downward-pointing triangles in Fig. 1(c).

Experiments with a variety of polygonal regions show that an n-vertex input typically produces about $22n$ triangles, as in Fig. 11. A simple polygon with $n - 3$ reflex corners can produce as many as $25n$ triangles; the maximum for polygons with holes appears to be about $33n$. Since a floating-point representation entails round-off, some of the right angles present in the nonobtuse triangulation become slightly obtuse. The worst test case had an angle of about $\pi/2 + 10^{-11}$ radians (Matlab retains 16 digits), so the implementation is fairly robust, which is somewhat surprising given that our implementation often places very small disks next to very large ones. (Relevant here is a paper by Smith [27] on the precision necessary to represent planar graphs by tangent disks.)

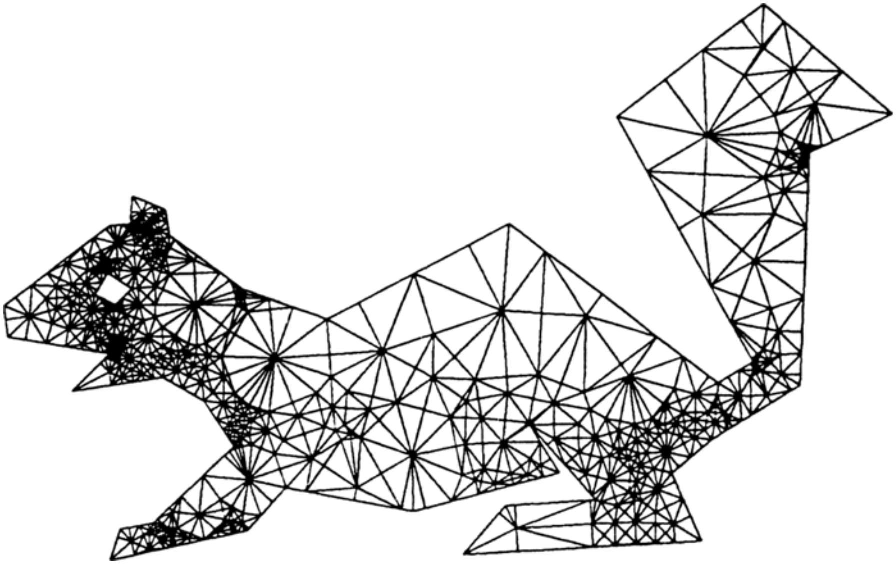

Fig. 11. This 40-vertex polygonal region produced 902 nonobtuse triangles.

6. Parallelizing the Algorithm

We now sketch the first $\mathcal{NC}$ algorithm for nonobtuse triangulation. We give a straightforward though rather inefficient algorithm, with parallel time $O(\log^3 n)$ and processor requirement $O(n^2)$. Both time and processors should be improvable. One bottleneck subproblem is the computation of the GVD of circular arcs; see [15] for the GVD of line segments.

Theorem 2. *An n-vertex polygonal region P (with holes) can be triangulated with $O(n)$ right triangles in $O(\log^3 n)$ time on $O(n^2)$ EREW PRAM processors.*

Proof. Using $O(n^2)$ processors—one for each vertex-edge pair—and time $O(\log n)$, we can compute the nearest nonincident edge for each vertex and hence choose appropriate radii for disks to pack into corners. The second step, connecting holes, is trickier. We first compute a minimum spanning tree (MST) of P's holes; by this we mean the shortest set of line segments S, each segment with both endpoints on the boundary of P, such that the union of S and the exterior of P is a connected subset of the plane. Using $O(n^2)$ processors and time $O(\log n)$, we compute for each vertex the nearest edge lying on a different connected component of P's boundary. We use this information to compute distances between connected components, and add to S the shortest component-joining line segment incident to each component. This reduces the number of components by at least a factor of two, so $O(\log n)$ such merging steps suffices to complete the computation of set S.

Now it is not hard to show that no point of the plane is covered by more than $O(1)$ diameter disks of segments in S. Hence there is a pairwise-disjoint set of diameter disks of cardinality a constant fraction of $|S|$ [30]. It is not hard to find these disks in parallel time $O(\log n)$ using separators. We repeat the process of computing the MST (of the new connected components, holes plus disks) and finding a large independent set of diameter disks. After $O(\log n)$ cycles—for total time of $O(\log^3 n)$—we have reduced to a simply connected arc polygon.

The third step of the disk-packing stage uses the GVD in order to find centroid disks. Using $O(n^2)$ processors and time $O(\log^2 n)$, we can compute the GVD of a set of n circular arcs as follows. We compute the equal-distance curve (bisector) for each pair of arcs. Then, for each arc a, we compute the piecewise-polynomial boundary of a's cell recursively by dividing the set of bisectors into equal halves and then merging the boundaries for each half. Two piecewise-polynomial boundaries of $O(n)$ pieces can be merged in time $O(\log n)$ on n processors. Once, the GVD has been computed, a centroid can be found in time $O(\log n)$ by alternately removing leaves and merging degree-two paths.

Recall that the algorithm requires a "decomposition tree" of centroid disks of height $O(\log n)$, so by simply recomputing the GVD after each centroid, we obtain an overall time for the third disk-packing step of $O(\log^3 n)$. Finally, the triangulation stage consists entirely of local operations, so it is trivially parallelized. □

7. Conclusions

We have presented a new algorithm for nonobtuse triangulation of polygons with holes. The number of triangles produced is linear in the number of vertices of the input, a significant improvement over previous methods. This is of course asymptotically optimal, resolving the question of the theoretical complexity of nonobtuse triangulation of polygons.

Warren Smith (personal communication) has pointed out two other nice features of our disk-packing approach. First, the approach can be generalized in a natural way to the sphere, giving linear-size nonobtuse triangulations of spherical polygons. Second, dynamic programming can be used in the recursive subdivision step of stage one in order to optimize the disk packing (over all such recursive disk packings). A natural optimization is to minimize the final number of triangles.

One direction for further work is extending the algorithm to inputs more general than polygons with holes; these inputs occur in modeling domains made of more than one material. Currently, there is an algorithm for refining a triangulated simple polygon into a nonobtuse triangulation with $O(n^4)$ triangles, and also an $\Omega(n^2)$ lower bound [7]. There is still no algorithm for polynomial-size nonobtuse triangulation of planar straight-line graphs; a solution to this problem would give another solution to "conforming Delaunay triangulation" [12]. There are, however, algorithms that triangulate a planar straight-line graph with angles bounded away from π. Mitchell [18] showed how to achieve maximum angle at most $7\pi/8$, using at most $O(n^2 \log n)$ triangles, and Tan [29] recently improved this result to $11\pi/15$ and $O(n^2)$, matching a lower bound on the number of triangles.

Another direction is exploring whether our ideas can be used for related mesh-generation problems. For instance, disk packing may yield a simpler algorithm for the problem of no-small-angle, nonobtuse triangulation [2], [17]. Perhaps we can use our methods to produce nonobtuse meshes with skinny triangles aligned with the boundary. (See [20] for aligned no-large-angle meshes.) Or perhaps our methods can be allied with a heuristic method called "bubble systems" [25].

Finally, higher dimensions are still a mystery. Do three-dimensional polyhedra admit polynomial-size triangulations without obtuse dihedral angles? Algorithms for point sets are known [5], [9].

Acknowledgments

We would like to thank Paul Chew, Jonathan Shewchuk, Warren Smith, and Shang-Hua Teng for some valuable discussions and Daniel Asimov and Micha Sharir for proofs of Lemma 3.

References

1. I. Babuška and A. Aziz. On the angle condition in the finite element method. *SIAM J. Numer. Anal.* **13** (1976), 215–227.
2. B. S. Baker, E. Grosse, and C. S. Rafferty, Nonobtuse triangulation of polygons. *Discrete Comput. Geom.* **3** (1988), 147–168.
3. R. E. Bank. *PLTMG User's Guide*. Philadelphia, PA, 1990.
4. J. L. Bentley and J. B. Saxe. Decomposable searching problems: 1. Static-to-dynamic transformation. *J. Algorithms* **1** (1980), 301–358.
5. M. Bern, L. P. Chew, D. Eppstein, and J. Ruppert. Dihedral bounds for mesh generation in high dimensions. *Proc. 6th ACM–SIAM Symp. on Discrete Algorithms*, 1995, pp. 189–196.
6. M. Bern, D. Dobkin, and D. Eppstein. Triangulating polygons without large angles. *Proc. 8th Annual ACM Symp. on Computational Geometry*, 1992, pp. 221–231. *Internat. J. Comput. Geom. Appl.*, **5** (1995), 171–192.
7. M. Bern and D. Eppstein. Polynomial-size nonobtuse triangulation of polygons. *Internat. J. Comput. Geom. Appl.* **2** (1992), 241–255.
8. M. Bern and D. Eppstein. Mesh generation and optimal triangulation. Tech. Report CSL-92-1, Xerox PARC, Palo Alto, CA. Also in *Computing in Euclidean Geometry*, World Scientific, Singapore, 1992.
9. M. Bern, D. Eppstein, and J. R. Gilbert. Provably good mesh generation. *Proc. 31st IEEE Symp. on Foundations of Computer Science*, 1990, pp. 231–241. *J. Comput. System Sci.*, **48** (1994), 384–409.
10. H. S. M. Coxeter, *Introduction to Geometry*. Wiley, New York, 1961.
11. H. Edelsbrunner and N. R. Shah. Incremental topological flipping works for regular triangulations. *Proc. 8th Annual ACM Symp. on Computational Geometry*, 1992, pp. 43–52.
12. H. Edelsbrunner and T. S. Tan. An upper bound for conforming Delaunay triangulations. *Discrete Comput. Geom.* **10** (1993), 197–213.
13. D. Eppstein. Faster circle packing with application to nonobtuse triangulation. Tech. Report 94-33, Department of Information and Computer Science, University of California, Irvine, CA, 1994. To appear in *Internat. J. Comput. Geom. Appl.*
14. S. Fortune, A sweepline algorithm for Voronoi diagrams. *Algorithmica* **2** (1987), 153–174.
15. M. T. Goodrich, C. ÓDúnlaing, and C. Yap. Computing the Voronoi diagram of a set of line segments in parallel. *Algorithmica* **9** (1993), 128–141.
16. *MATLAB Reference Guide*, The MathWorks, Inc., Natick, MA, 1992.
17. E. Melissaratos and D. Souvaine. Coping with inconsistencies: a new approach to produce quality triangulations of polygonal domains with holes. *Proc. 8th Annual ACM Symp. on Computational Geometry*, 1992, pp. 202–211.

18. S. A. Mitchell. Refining a triangulation of a planar straight-line graph to eliminate large angles. *Proc. 34th Symp. on Foundations of Computer Science,* 1993, pp. 583–591.
19. S. A. Mitchell. Finding a covering triangulation whose maximum angle is provably small. (Proc. 17th Annual Computer Science Conference.) *Austral. Comput. Sci. Comm.* **16** (1994), 55–64.
20. J.-D. Müller. Proven angular bounds and stretched triangulations with the frontal Delaunay method. *Proc. 11th AIAA Comp. Fluid Dynamics,* Orlando, FL, 1993.
21. S. Müller, K. Kells, and W. Fichtner, Automatic rectangle-based adaptive mesh generation without obtuse angles. *IEEE Trans. Computer-Aided Design* **10** (1992), 855–863.
22. F. Preparata and M. Shamos. *Computational Geometry—an Introduction.* Springer-Verlag, New York, 1985.
23. J. F. Randolph. *Calculus and Analytic Geometry.* Wadsworth, Belmont, CA, 1961, pp. 373–374.
24. J. Ruppert. A new and simple algorithm for quality two-dimensional mesh generation. *Proc. 4th ACM–SIAM Symp. on Discrete Algorithms,* 1993, pp. 83–92.
25. K. Shimada and D. C. Gossard. Computational methods for physically based FE mesh generation. *Proc. IFIP TC5 / WG5.3 8th Internat. Conf. on PROLAMAT,* Tokyo, 1992.
26. D. D. Sleator and R. E. Tarjan. A data structure for dynamic trees. *J. Comput. System Sci.* **24** (1983), 362–381.
27. W. D. Smith. Accurate circle configurations and numerical conformal mapping in polynomial time. Tech. Report 91-091-3-0058-6, NEC Research Center, Princeton, NJ, 1991.
28. G. Strang and G. J. Fix. *An Analysis of the Finite Element Method.* Prentice-Hall, Englewood Cliffs, NJ, 1973.
29. T.-S. Tan. An optimal bound for conforming quality triangulations. *Proc. 10th ACM Symp. on Computational Geometry,* 1994, pp. 240–249.
30. S.-H. Teng. Points, spheres, and separators: a unified geometric approach to graph partitioning. Ph.D. Thesis, CMU-CS-91-184, Carnegie Mellon University, Pittsburgh, PA, 1991.
31. S. A. Vavasis. Stable finite elements for problems with wild coefficients. Tech. Report TR93-1364, Department of Computer Science, Cornell University, Ithaca, NY, 1993.

Received June 1994, and in revised form January 1995.