

Approximation Algorithms for a Triangle Enclosure Problem

Karim Douïeb*

Matthew Eastman*

Anil Maheshwari*

Michiel Smid*

Abstract

Given a set S of n points in the plane, we want to find a triangle, with vertices in S , such that the number of points enclosed by it is maximum. An exact solution can be found by considering all $\binom{n}{3}$ triples of points in S . We consider the problem of approximating this triangle. We show that, by considering only triangles with at least 1, 2, or 3 vertices on the convex hull of S , we obtain various approximation algorithms that run in $o(n^3)$ time.

1 Introduction

Let S be a set of n points in the plane. A triangle Δpqr , with vertices $p, q, r \in S$, is defined to be *optimal* if the number of points of S enclosed by it is maximum. Eppstein *et al.* [1] have shown that this optimal triangle can be computed in $O(n^3)$ time: They present an algorithm that preprocesses the set S in $O(n^2)$ time so that, for any triple (p, q, r) of points in S , the number of points enclosed by Δpqr can be computed in $O(1)$ time. By considering all $\binom{n}{3}$ triples, we find an optimal triangle in $O(n^3)$ time.

Since it is not known if an optimal triangle can be computed in $o(n^3)$ time, we consider the problem of approximating it. That is, we will present several sub-cubic algorithms that compute triangles with vertices in S that enclose at least $1/c$ times as many points as an optimal triangle with vertices in S , for some approximation ratio c .

Our main approach is based on the simple fact that if a triangle Δ can be covered by c triangles, then one of them is a c -approximation of Δ .

We show that, by considering only triangles that contain at least 1, 2, or 3 vertices on the convex hull of S , we obtain approximation algorithms, for various values of c , that run in $o(n^3)$ time. Let h denote the number of vertices on the convex hull of S . A summary of our results is given in Table 1.

2 Preliminaries

We will assume that no three points in S are collinear and that no two points have the same y -coordinate.

*School of Computer Science, Carleton University, Ottawa, Canada. This work was supported by the Natural Sciences and Engineering Research Council of Canada.

vertices on the convex hull	approximation ratio	runtime
≥ 1	2	$O(n^2)$
≥ 2	3	$O(nh^2 \log n)$
≥ 2	4	$O(n \log^2 n)$
3	4	$O(nh^2 \log h)$
3	8	$O(n \log^2 h)$
3	$3 \log h$	$O(n \log h)$

Table 1: Summary of results.

The number of points *enclosed* by a triangle Δpqr is the number of points contained in the interior of Δpqr . We say that Δpqr , with $p, q, r \in S$, is *optimal* if the number of points of S enclosed by it is maximum.

A triangle Δ is a c -approximation of a triangle Δpqr if Δ encloses at least $1/c$ times as many points as Δpqr .

Observation 1 *If a triangle Δpqr can be covered by a set of c triangles then at least one of these triangles is a c -approximation of Δpqr .*

In order to show that an algorithm gives a c -approximation of a triangle Δpqr it is enough to show that the algorithm counts the number of points enclosed by each of the c triangles that cover Δpqr .

Let $l(p, q)$ denote the directed line through points p and q , and let \overline{pq} denote the line segment between p and q . Define the *wedge* of a vertex p in a triangle Δpqr as the area bounded by the lines $l(q, p)$ and $l(r, p)$ opposite the interior angle $\angle rpq$.

Lemma 1 *The three wedges of an optimal triangle with vertices in S cannot contain any points of S .*

Proof. Let Δpqr be an optimal triangle with vertices in S . Assume that the wedge of p contains a point p' as in Figure 1. Then the triangle $\Delta p'qr$ encloses more points than Δpqr , as it encloses all of the points enclosed by Δpqr in addition to the point p , giving a contradiction. \square

We refer to the three wedges of an optimal triangle as the *empty regions* of the optimal triangle.

3 Counting points in triangles with two fixed vertices on the convex hull

In order to approximate an optimal triangle in $o(n^3)$ time we need to be able to count the number of points

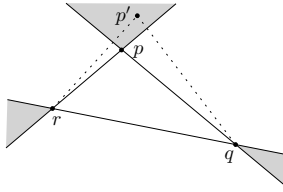


Figure 1: The wedge of p cannot contain any points. The shaded regions denote the empty regions of Δpqr .

in a set of triangles in $o(n^3)$ time. Fixing two vertices of every triangle on the convex hull of S allows us to count the number of points enclosed by these triangles in $O(n \log n)$ time, or $O(n \log h)$ time if we only consider triangles with the third vertex on the convex hull.

Lemma 2 *Given two points t_i and t_j on the convex hull of S we can count the number of points enclosed by every triangle $\Delta t_i t_j s$, $s \in S$, in $O(n \log n)$ time.*

Proof. Without loss of generality assume that t_i is below t_j . Let S_L be the set of points of S lying to the left of $l(t_i, t_j)$ and let S_R be the set of points of S lying to the right of $l(t_i, t_j)$.

The following algorithm counts the number of points enclosed by every triangle $\Delta t_i t_j s$, $s \in S_L$. Counting the number of points enclosed by every triangle $\Delta t_i t_j s$, $s \in S_R$, is symmetric.

For each point $s \in S_L$, let s' be the intersection between the horizontal line through s and $l(t_i, t_j)$. Let S_L^- be the set of points in S_L lying below the horizontal line through t_i and let S_L^+ be the set of points lying above the horizontal line through t_i .

Let T be an initially empty balanced binary search tree such that every node in T stores the size of its subtree. Rotate a line anchored at t_i clockwise over the set S_L^- . When this line intersects a point $s \in S_L^-$ insert s into T using its y -coordinate as the key. The number of points enclosed by $\Delta t_i s s'$ is the number of successors of s in T immediately after inserting s .

To see why this is true let u be a successor of s in T found immediately after inserting s into T . Since u was inserted before, s the angle $\angle ut_i s'$ is less than $\angle st_i s'$. Since u is a successor of s in T , u is higher than s . Therefore u is enclosed by $\Delta t_i s s'$ (see Figure 2).

The number of points enclosed by every triangle $\Delta t_i s s'$, $s \in S_L^+$, is found using the same technique, except that the line is rotated counter-clockwise over S_L^+ and the number of points in each $\Delta t_i s s'$, $s \in S_L^+$, is the number of predecessors of s in T immediately after inserting s .

Counting the number of points enclosed by every triangle $\Delta t_j s s'$, $s \in S_L$, is symmetric.

For each point $s \in S_L$ let $a_{i,s}$ be the number of points enclosed by $\Delta t_i s s'$ and let $a_{j,s}$ be the number of points enclosed by $\Delta t_j s s'$. Then the number of points enclosed

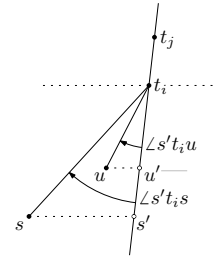


Figure 2: Point u is enclosed by $\Delta t_i s s'$.

by $\Delta t_i t_j s$ is either (1) $-a_{i,s} + a_{j,s}$ if s is below t_i , (2) $a_{i,s} - a_{j,s}$ if s is above t_j , or (3) $a_{i,s} + a_{j,s}$ otherwise. These cases are shown in Figure 3.

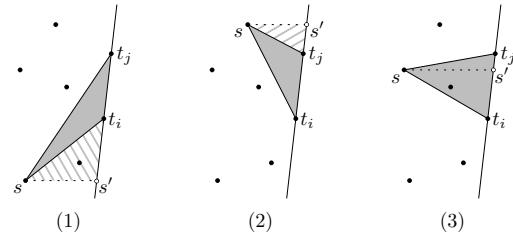


Figure 3: The three cases encountered when calculating the number of points enclosed by $\Delta t_i t_j s$.

It takes $O(n \log n)$ time to sort the points by angle about t_i and t_j . Inserting each point into the binary search tree takes $O(\log n)$ time. Since the binary search tree keeps track of the size of each subtree we can calculate the number of predecessors or successors of a point in the tree in $O(\log n)$ time. The total runtime is $O(n \log n)$. \square

If we fix two vertices on the convex hull of S we can count the number of points enclosed by every triangle containing these two vertices, with the third vertex on the convex hull, without sorting the entire set S . This lets us count the number of points enclosed by every such triangle in $O(n \log h)$ time.

Lemma 3 *Given two points t_i and t_j on the convex hull of S we can count the number of points enclosed by every triangle $\Delta t_i t_j t_k$ where t_k , $1 \leq k \leq h$, is a point on the convex hull of S , in $O(n \log h)$ time.*

Proof. Without loss of generality assume that t_i is below t_j . Let S_L be the set of points of S lying to the left of $l(t_i, t_j)$ and let S_R be the set of points of S lying to the right of $l(t_i, t_j)$.

The following algorithm counts the number of points enclosed by every triangle $\Delta t_i t_j t_k$, where $t_k \in S_L$ is a point on the convex hull between t_i and t_j . Counting the number of points enclosed by every triangle $\Delta t_i t_j t_k$, where $t_k \in S_R$ is a point on the convex hull, is symmetric.

The number of points enclosed by $\triangle t_i t_j t_k$, with $t_k \in S_L$, is found by subtracting the number of points in S_L lying to the left of $l(t_i, t_k)$, or to the right of $l(t_j, t_k)$, from the number of points in S_L .

A point $s \in S_L$ lies to the left of $l(t_i, t_k)$ if the line $l(t_i, s)$ intersects the convex hull between t_i and t_k . Similarly, s lies to the right of $l(t_j, t_k)$ if $l(t_j, s)$ intersects the convex hull between t_k and t_j (see Figure 4).

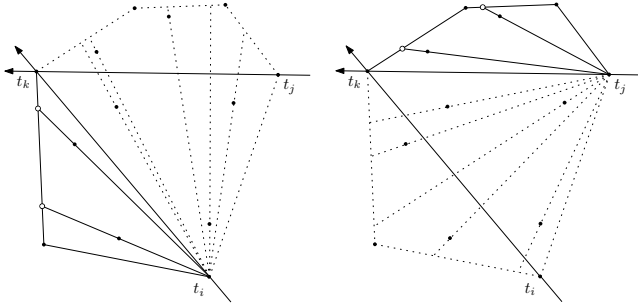


Figure 4: Lines through t_i and the points lying to the left of $l(t_i, t_k)$ intersect the convex hull between t_i and t_k . Lines through t_j and points lying to the right of $l(t_j, t_k)$ intersect the convex hull between t_k and t_j .

Let $a_{i,k}$ be the number of lines $l(t_i, s)$, $s \in S_L$, that intersect the edge $\overline{t_k t_{k+1}}$ of the convex hull and let $a_{j,k}$ be the number of lines $l(t_j, s)$, $s \in S_L$, that intersect the edge $\overline{t_k t_{k+1}}$ of the convex hull.

Let $b_{i,k}$ be the total number of lines $l(t_i, s)$, $s \in S_L$, that intersect the convex hull between points t_i and t_k and let $b_{j,k}$ be the total number of lines $l(t_j, s)$, $s \in S_L$, that intersect the convex hull between t_k and t_j .

The number of points enclosed by triangle $\triangle t_i t_j t_k$ is $|S_L| - (b_{i,k} + b_{j,k} - 1)$.

The sets S_L and S_R are found in $O(n)$ time. The convex hull can be found in $O(n \log h)$ time and the intersection of a line and the convex hull can be found in $O(\log h)$ time by performing a binary search on the edges of the convex hull. Then the a -variables are computed in $O(n \log h)$ time and the b -variables are computed in $O(h)$ time. The total runtime is $O(n \log h)$. \square

4 Triangles with one fixed vertex on the convex hull

Lemma 4 *Let z be the lowest point in S . Let x and y be points in S such that $\triangle xyz$ encloses the maximum number of points of S . Then $\triangle xyz$ is a 2-approximation of an optimal triangle with vertices in S .*

Proof. Let $\triangle pqr$ be an optimal triangle with vertices in S . Draw a line from z to each vertex of $\triangle pqr$. By Lemma 1 the point z cannot lie in any of the empty regions of $\triangle pqr$. Then one of the lines from z must cross an edge of $\triangle pqr$.

Without loss of generality assume that \overline{zp} crosses the edge \overline{qr} . Then the two triangles $\triangle pqz$ and $\triangle rpz$ cover

$\triangle pqr$ (see Figure 5). By Observation 1 one of these triangles is a 2-approximation of $\triangle pqr$. \square

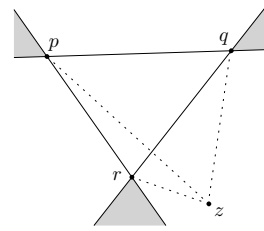


Figure 5: Triangles $\triangle pqz$ and $\triangle rpz$ cover $\triangle pqr$.

Theorem 5 *A 2-approximation of an optimal triangle with vertices in S can be found in $O(n^2)$ time.*

Proof. Let z be the lowest point in S . Count the number of points enclosed by every triangle containing vertex z and return the triangle found that encloses the most points.

There are $\binom{n}{2}$ triangles containing vertex z so this takes $O(n^2)$ time using the data structure from [1]. The approximation ratio follows from Lemma 4. \square

5 Triangles with at least two vertices on the convex hull

In this section we consider triangles with at least two vertices on the convex hull of S .

Lemma 6 *Let \triangle be a triangle, with vertices in S , such that at least two of its vertices are on the convex hull of S , that encloses the maximum number of points of S . Then \triangle is a 3-approximation of an optimal triangle with vertices in S .*

Proof. Let $\triangle pqr$ be an optimal triangle with vertices in S . Assume that none of the vertices of $\triangle pqr$ lie on the convex hull of S . Then there exist edges $\overline{t_i t_{i+1}}$, $\overline{t_j t_{j+1}}$ and $\overline{t_k t_{k+1}}$ of the convex hull that cross the empty regions of $\triangle pqr$. Figure 6 shows how we can use the end points of two of these edges, and one vertex of $\triangle pqr$, to cover $\triangle pqr$ with three triangles. By Observation 1 one of these triangles is a 3-approximation of $\triangle pqr$. \square

This approximation factor is tight. Figure 7 shows an example of a set of points where $\triangle pqr$ encloses three times as many points as any triangle \triangle , with vertices in S , with at least two vertices on the convex hull of S . There is no such triangle \triangle that covers more than one of the shaded regions in Figure 7. If we put m points in each of these regions then $\triangle pqr$ will enclose $3m$ points while any triangle with at least two vertices on the convex hull of S can enclose at most $m+1$ points.

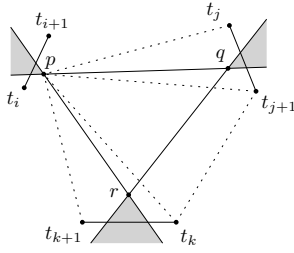


Figure 6: Three triangles that cover Δpqr .

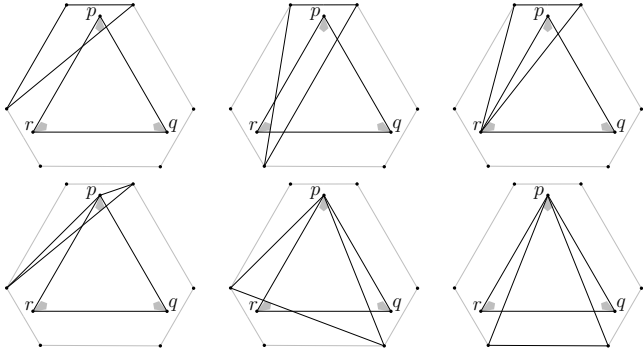


Figure 7: A set S , with an optimal triangle Δpqr , such that there are no triangles with at least two vertices on the convex hull of S that enclose more than $1/3$ times as many points as Δpqr . Symmetric cases are not shown.

Theorem 7 A 3-approximation of an optimal triangle with vertices in S can be found in $O(\min(n^2 + nh^2, nh^2 \log n))$ time.

Proof. Count the number of points enclosed by every triangle with at least two vertices on the convex hull of S and return the triangle found that encloses the most points. There are $(n - h) \binom{h}{2}$ such triangles so this takes $O(n^2 + nh^2)$ time using the data structure from [1] or $O(h^2 n \log n)$ time using the algorithm presented in Lemma 2. The approximation ratio follows from Lemma 6. \square

Theorem 8 A 4-approximation of an optimal triangle with vertices in S can be found in $O(n \log^2 n)$ time.

Proof. Consider the following algorithm: Sort the points of S clockwise by angle about the lowest point z in S . Let s_m be the median of S by angle and let $\overline{t_i t_{i+1}}$ be the edge of the convex hull that intersects $l(z, s_m)$. Count the number of points enclosed by every triangle $\Delta zt_i s$ and $\Delta zt_{i+1} s$, $s \in S$, using the algorithm in Lemma 2. Let S_L be the set of points lying to the left of $l(z, s_m)$ and let S_R be the set of points lying to the right of $l(z, s_m)$. Recursively run the algorithm on the sets S_L and S_R and return the triangle found that encloses the most points.

To prove the approximation ratio, let Δpqr be an optimal triangle with vertices in S . Let x and y be points

in S such that Δxyz encloses the maximum number of points of S . From Lemma 4 Δxyz is a 2-approximation of Δpqr .

Consider the recursive call where x and y lie on opposite sides of the line $l(z, s_m)$. At least one of t_i and t_{i+1} must lie above $l(x, y)$, otherwise $\overline{t_i t_{i+1}}$ wouldn't be an edge of the convex hull. If t_i lies above $l(x, y)$ then Δxyz is covered by triangles Δzxt_i and Δyzt_i (as in Figure 8). Otherwise if t_{i+1} lies above $l(x, y)$ then Δxyz is covered by triangles Δzxt_{i+1} and Δyzt_{i+1} . By Lemma 1 one of these triangles is a 2-approximation of Δxyz and, therefore, a 4-approximation of Δpqr .

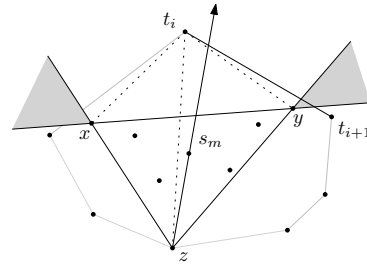


Figure 8: Triangles Δzxt_i and Δyzt_i cover Δxyz .

Sorting the points by angle takes $O(n \log n)$ time. Finding the edge of the convex hull that intersects the line through z and the median takes $O(\log h)$ time if we perform a binary search on the precomputed edges of the convex hull. Counting the number of points enclosed by every triangle $\Delta zt_i s$ and $\Delta zt_{i+1} s$, $s \in S$, takes $O(n \log n)$ time using the algorithm presented in Lemma 2. The total amount of work done at each step is $O(n \log n)$. S_L and S_R each contain half of the points of S so the complexity of this algorithm satisfies the equation $T(n) = 2T(n/2) + O(n \log n)$ which solves to $O(n \log^2 n)$. \square

6 Triangles with three vertices on the convex hull

In this section we consider triangles with three vertices on the convex hull of S .

Lemma 9 Let Δ be a triangle, whose vertices are on the convex hull of S , that encloses the maximum number of points of S . Then Δ is a 4-approximation of an optimal triangle with vertices in S .

Proof. Let Δpqr be an optimal triangle with vertices in S . Assume that none of the vertices of Δpqr lie on the convex hull of S . Then there exist edges $\overline{t_i t_{i+1}}$, $\overline{t_j t_{j+1}}$ and $\overline{t_k t_{k+1}}$ of the convex hull that cross the empty regions of Δpqr . Figure 9 shows how we can use the end points of these edges to find a set of at most four triangles that cover Δpqr . By Lemma 1 one of these triangles is a 4-approximation of Δpqr . \square

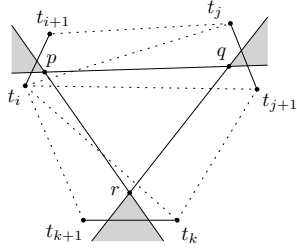


Figure 9: Four triangles that cover Δpqr .

This approximation factor is tight. Figure 10 shows an example where Δpqr encloses four times as many points of S as any triangle Δ , whose vertices are on the convex hull of S . There is no such triangle Δ that covers more than one of the four shaded regions in Figure 10. If we put m points in each of these regions then Δpqr will enclose $4m$ points while any triangle whose vertices are on the convex hull of S can enclose at most $m + 1$ points.

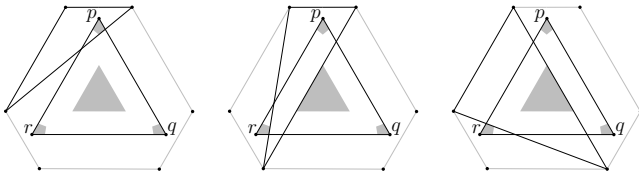


Figure 10: A set S , with an optimal triangle Δpqr , such that there are no triangles, whose vertices are on the convex hull of S , that cover more than $1/4$ times as many points as Δpqr . Symmetric cases are not shown.

Theorem 10 *A 4-approximation of an optimal triangle with vertices in S can be found in $O(\min(n^2 + h^3, h^2 n \log h))$ time.*

Proof. Count the number of points enclosed by every triangle with three vertices on the convex hull of S and return the triangle found that encloses the most points. There are $\binom{h}{3}$ such triangles so this takes $O(n^2 + h^3)$ time using the data structure in [1] or $O(h^2 n \log h)$ time using the algorithm presented in Lemma 3. The approximation ratio follows from Lemma 9. \square

Theorem 11 *An 8-approximation of an optimal triangle with vertices in S can be found in $O(n \log^2 h)$ time.*

Proof. Consider the following algorithm: Let $t_1 \dots t_h$ be the vertices of the convex hull of S given in clockwise order starting at the lowest point $z = t_1$ and let t_m be the median of the convex hull. Count the number of points enclosed by every triangle containing vertices z and t_m , with the third vertex on the convex hull of S , using the algorithm described in Lemma 3. Let S_L be the set of points of S lying on or to the left of $l(z, t_m)$ and

let S_R be the set of points of S lying on or to the right of $l(z, t_m)$. Recursively run the algorithm on the sets S_L and S_R and return the triangle found that encloses the most points.

To prove the approximation ratio, let Δpqr be an optimal triangle with vertices in S . Let x and y be points in S such that Δxyz encloses the maximum number of points of S . From Lemma 4 Δxyz is a 2-approximation of Δpqr .

Assume that x and y are not on the convex hull. Then there exist edges $t_i t_{i+1}$ and $t_k t_{k+1}$ that cross the empty regions of Δxyz . Let t_j be any point on the convex hull between t_{i+1} and t_k . Figure 11 shows how we can use the points $z, t_i, t_{i+1}, t_j, t_k$ and t_{k+1} to construct four triangles that cover Δxyz . By Lemma 1 one of these triangles is a 4-approximation of Δxyz and, therefore, an 8-approximation of Δpqr .

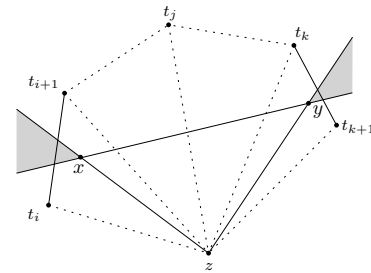


Figure 11: Four triangles that cover Δxyz .

Consider the recursive call where x and y lie on opposite sides of $l(z, t_m)$. When this occurs t_m is on the convex hull between t_{i+1} and t_k . Thus, in the previous argument, we can take $t_j = t_m$. Then in this call we count the number of points in triangles $\Delta z t_j t_{i+1}$ and $\Delta z t_j t_k$.

In another recursive call either t_i or t_{i+1} is the median of the convex hull and we count the number of points enclosed by the triangle $\Delta z t_i t_{i+1}$. Similarly there is a recursive call where either t_k or t_{k+1} is the median and we count the number of points enclosed by $\Delta z t_k t_{k+1}$.

The convex hull of S can be found in $O(n \log h)$ time [2] and does not need to be computed at each step. Each step requires $O(n)$ time to find S_L and S_R and $O(n \log h)$ time to count the number of points enclosed by every triangle with vertices z and t_m , with the third vertex on the convex hull of S , by Lemma 3. When we recursively call the algorithm on the sets S_L and S_R the size of the convex hulls of S_L and S_R are half the size of the convex hull of S and the total number of points in S_L and S_R is the number of points in S . The complexity of this algorithm satisfies the equation $T(h, n) = T(h/2, n_1) + T(h/2, n - n_1) + O(n \log h)$ for some $1 \leq n_1 < n$. The solution to this equation is $O(n \log^2 h)$. \square

We can obtain an $O(\log h)$ -approximation of the optimal triangle with vertices in S in $O(n \log h)$ time by triangulating the convex hull of S and choosing the triangle in this triangulation that encloses the maximum number of points of S .

Let $T = \emptyset$ be an initially empty set of triangles. Initialize $R = r_1, r_2, \dots, r_h$ to the points of the convex hull of S given in clockwise order. For each point $r_i \in R$ such that i is odd add the triangle $\Delta r_i r_{i+1} r_{i+2}$ to T and remove r_{i+1} from R . Renumber the elements of R as r_1, r_2, \dots and repeat the previous steps until R has less than 3 points. This gives a triangulation T of the convex hull of S (see Figure 12). At each iteration we remove half of the points in R , so T is constructed in $O(h)$ time after constructing the convex hull of S in $O(n \log h)$ time [2].

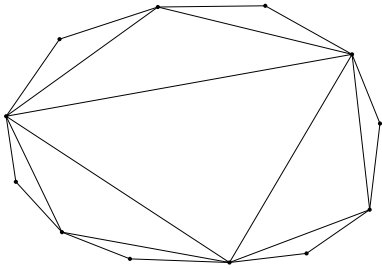


Figure 12: Triangulation of the convex hull of a set of points.

Lemma 12 Any line crosses at most $2 \log h$ triangles of T .

Proof. Let R_i denote the sequence of points in R at the i th iteration of the triangulation algorithm.

Observe that R_i and R_{i+1} are convex polygons and that any triangle added to T in the i th iteration has edges in R_i and R_{i+1} only (see Figure 13). Then any line can intersect at most two of the triangles of T added during the i th iteration of the triangulation algorithm.

There are $\log h$ iterations of the algorithm, so any line crosses at most $2 \log h$ triangles in T . \square

Lemma 13 Any triangle Δ , with vertices in S , can be covered by at most $3 \log h$ triangles in T .

Proof. Observe that any triangle in T that partially covers Δ must cross at least two edges of Δ , since every triangle in T has vertices on the convex hull of S . By Lemma 12 each edge of Δ can cross at most $2 \log h$ triangles in T . Then the edges of Δ can cross at most $6/2 \log h$ different triangles in T . Therefore Δ can be covered by at most $3 \log h$ triangles in T . \square

Theorem 14 A $3 \log h$ -approximation of an optimal triangle with vertices in S can be found in $O(n \log h)$ time.

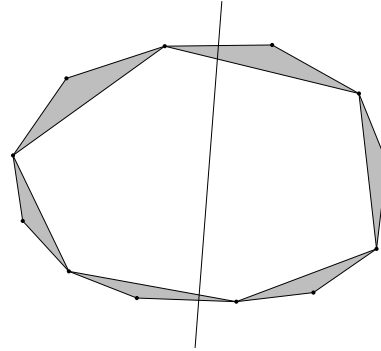


Figure 13: Any line can cross at most two of the triangles added during the i th iteration of the algorithm. The shaded regions denote triangles added to T during the i th iteration.

Proof. For each point $s \in S$ we can find the triangle in T enclosing s in $O(\log h)$ time: Start with the innermost triangle $\Delta t_i t_j t_k$. If s is in this triangle we are done. Otherwise s lies to the left of one of the lines $l(t_i, t_j)$, $l(t_j, t_k)$ or $l(t_k, t_i)$. Without loss of generality let s lie to the left of $l(t_i, t_j)$. Repeat the previous steps with the triangle immediately to the left of the line $l(t_i, t_j)$. At each step we remove $2/3$ of the triangles. Since there are $O(h)$ triangles it takes $O(\log h)$ time to find the triangle of T that encloses s . Therefore it takes $O(n \log h)$ time to find the triangle in T that encloses the maximum number of points of S . The approximation ratio follows from Observation 1 and Lemma 13. \square

7 Conclusion

It is not known whether the $O(n^3)$ time algorithm used to find the triangle enclosing the most points is optimal. Similarly it is unclear if the runtimes of our approximations are optimal.

Eppstein *et al.* [1] studied the more general problem of finding a convex k -gon that is optimal for some weight function, for example the minimum or maximum number of points, or the minimum perimeter. Their algorithm runs in $O(kn^3)$ time. It would be interesting to see if any of our results can be applied to these problems.

References

- [1] D. Eppstein, M. Overmars, G. Rote, and G. Woeginger. Finding minimum area k -gons. *Discrete Comput. Geom.*, 7(1):45–58, 1992.
- [2] D.G. Kirkpatrick and R. Seidel. The ultimate planar convex hull algorithm. *SIAM Journal on Computing*, 15(1):287–299, 1986.