

Entropy, Triangulation, and Point Location in Planar Subdivisions

SÉBASTIEN COLLETTE, Université Libre de Bruxelles
 VIDA DUJMOVIĆ, Carleton University
 JOHN IACONO, Polytechnic Institute of NYU
 STEFAN LANGERMAN, Université Libre de Bruxelles
 PAT MORIN, Carleton University

A data structure is presented for point location in connected planar subdivisions when the distribution of queries is known in advance. The data structure has an expected query time that is within a constant factor of optimal. More specifically, an algorithm is presented that preprocesses a connected planar subdivision G of size n and a query distribution D to produce a point location data structure for G . The expected number of point-line comparisons performed by this data structure, when the queries are distributed according to D , is $\tilde{H} + O(\tilde{H}^{1/2} + 1)$ where $\tilde{H} = \tilde{H}(G, D)$ is a lower bound on the expected number of point-line comparisons performed by any linear decision tree for point location in G under the query distribution D . The preprocessing algorithm runs in $O(n \log n)$ time and produces a data structure of size $O(n)$. These results are obtained by creating a Steiner triangulation of G that has near-minimum entropy.

Categories and Subject Descriptors: I.3.5 [Computational Geometry and Object Modeling]: Algorithms

General Terms: Computational geometry, Point location, Minimum-entropy triangulation

ACM Reference Format:

Collette, S., Dujmović, V., Iacono, J., Langerman, S., and Morin, P. 2011 Entropy, Triangulation, and Point Location in Planar Subdivisions ACM Trans. Algor. V, N, Article A (January YYYY), 18 pages.
 DOI = 10.1145/0000000.0000000 <http://doi.acm.org/10.1145/0000000.0000000>

1. INTRODUCTION

The planar point location problem is the classic search problem in computational geometry. Given a planar subdivision G ,¹ the planar point location problem asks us to construct a data structure so that, for any query point q , we can quickly determine which part of G (vertex, edge, or face) contains q .

The history of the planar point location problem parallels, in many ways, the history of binary search trees. After a few initial attempts [Dobkin and Lipton 1976; Lee and Preparata 1977; Preparata 1981], asymptotically optimal (and quite different) linear-space $O(\log n)$ query time solutions to the planar point location problem were obtained by Kirkpatrick [Kirkpatrick 1983], Sarnak and Tarjan [Sarnak and Tarjan 1986], and Edelsbrunner *et al.*

¹A *planar subdivision* is a partitioning of the plane into points (called *vertices*), open line segments (called *edges*), and maximal connected 2-dimensional regions (called *faces*).

V. Dujmović's research partially supported by NSERC and the Ontario Ministry of Research and Innovation. J. Iacono's research partially supported by NSF grants OISE-0334653 and CCF-0430849 and by an Alfred P. Sloan research fellowship. S. Langerman is Maître de recherches du F.R.S.-FNRS. P. Morin's research partially supported by NSERC, FNRS, the Ontario Ministry of Research and Innovation, Canada Foundation for Innovation, The University of Sydney, and National ICT Australia.

Permission to make digital or hard copies of part or all of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies show this notice on the first page or initial screen of a display along with the full citation. Copyrights for components of this work owned by others than ACM must be honored. Abstracting with credit is permitted. To copy otherwise, to republish, to post on servers, to redistribute to lists, or to use any component of this work in other works requires prior specific permission and/or a fee. Permissions may be requested from Publications Dept., ACM, Inc., 2 Penn Plaza, Suite 701, New York, NY 10121-0701 USA, fax +1 (212) 869-0481, or permissions@acm.org.

© YYYY ACM 1549-6325/YYYY/01-ARTA \$10.00

DOI 10.1145/0000000.0000000 <http://doi.acm.org/10.1145/0000000.0000000>

[Edelsbrunner et al. 1986] in the mid 1980s. These results were based on hierarchical simplification, data structural persistence, and fractional cascading, respectively. All three of these techniques have subsequently found many other applications. An elegant randomized solution, combining aspects of all three previous solutions, was later given by Mulmuley [Mulmuley 1990], and uses randomized incremental construction, a technique that has since become pervasive in computational geometry [de Berg et al. 2008, Section 9.5]. Preparata [Preparata 1990] gives a comprehensive survey of the results of this era.

In the 1990s, several authors became interested in determining the exact constants achievable in the query time. Goodrich *et al.* [Goodrich et al. 1997] gave a linear-size data structure that, for any query, requires at most $2 \log n + o(\log n)$ point-line comparisons and conjectured that this query time was optimal for linear-space data structures.² The following year, Adamy and Seidel [Adamy and Seidel 1998] gave a linear-space data structure that answers queries using $\log n + 2\sqrt{\log n} + O(\log \log n)$ point-line comparisons and showed that this result is optimal up to the third term.

Still not done with the problem, several authors considered the point location problem under various assumptions about the query distribution, D . All these solutions compare the expected query time to the *entropy bound*; in a planar subdivision G with m faces F_1, \dots, F_m , if $\Pr(F_i)$ is the probability that q is contained in F_i , then no algorithm that makes only binary decisions can answer queries using an expected number of decisions that is fewer than

$$H = H(G, D) = \sum_{i=1}^m \Pr(F_i) \log(1/\Pr(F_i)) . \quad (1)$$

In the previous results on planar point location, none of the query times are affected significantly by the structure of G ; they hold for arbitrary planar subdivisions. However, when studying point location under a distribution assumption the problem becomes more complicated and the results become more specific. A *connected subdivision* is a planar subdivision whose underlying (vertex and edge) graph is connected. A *convex subdivision* is a planar subdivision whose faces are all convex polygons, except the outer face, which is the complement of a convex polygon. A *triangulation* is a convex subdivision in which each face has at most 3 edges on its boundary.

Note that, if every face of G has a constant number of sides, then G can be augmented, by the addition of extra edges, so that it is a triangulation without increasing (1) by more than a constant. Similarly, in a convex subdivision G where the query distribution D is uniform within each face of G , the faces of the subdivision can be triangulated without increasing the entropy by more than a constant [Arya et al. 2000]. Thus, in the following we will simply refer to results about triangulations where it is understood that these also imply the same result for planar subdivisions with faces of constant size or convex subdivisions when the query distribution is uniform within each face.

Arya *et al.* [Arya et al. 2000] gave two results for the case where the query point p is chosen from a known distribution where the x and y coordinates of p are chosen independently and G is a convex subdivision. They gave an $O(n)$ space data structure for which the expected number of point-line comparisons is at most $4H + O(1)$ and an $O(n^2)$ space data structure for which the expected number of point-line comparisons is at most $2H + O(1)$. The assumption about the independence of the x and y coordinates of p is crucial to the these results.

For arbitrary distributions that are known in advance, several results exist. Iacono [Iacono 2001; 2004] showed that, for triangulations, a simple variant of Kirkpatrick's original point location structure gives a linear space, $O(H + 1)$ expected query time data structure.

²Here and throughout, logarithms are implicitly base 2 unless otherwise specified.

Simultaneously, and independently, Arya *et al.* [Arya et al. 2001b] showed that a variant of Mulmuley’s randomized data structure also achieves $O(H + 1)$ expected query time. A sequence of papers by Arya *et al.* [Arya et al. 2000; 2001a; Arya et al. 2007] has recently culminated in an $O(n)$ space data structure for point location in triangulations with query time $H + O(H^{1/2} + 1)$ [Arya et al. 2007].

In the current paper, we show that, for any connected planar subdivision, there exists a data structure of size $O(n)$ that can answer point location queries using $\tilde{H} + O(\tilde{H}^{1/2} + 1)$ point/line comparisons. Here, $\tilde{H} = \tilde{H}(G, D)$ is a lower bound on the expected cost of any linear decision tree that solves this problem. Note that \tilde{H} is often greater than the quantity H defined above and this is necessarily so. To see this, consider that the problem of testing whether a query point is contained in a simple polygon P with n vertices is a special case of planar point location in a connected planar subdivision. However, in this special case the subdivision only has 2 faces, so $H \leq 1$. It seems unlikely that, for any simple polygon P and any probability measure D over \mathbb{R}^2 , it is always possible to test in $O(1)$ expected time if a point p drawn from D is contained in P . Indeed, it is not hard to design a convex polygon P and distribution D so that the expected cost of any algebraic decision tree for point location in P , under query distribution D , is $\Omega(\log n)$.

Note that all known algorithms for planar point location that do not place special restrictions on the input subdivision can be described in the linear decision tree model of computation.³ The data structures presented in the current paper are the most general results known about planar point location and imply, to within a lower order term, all of the results discussed in the introduction.

We achieve our results by showing how to compute a Steiner triangulation $\Delta = \Delta(G, D)$ of G that has nearly minimum entropy over all possible triangulations of G and then proving that the entropy of a minimum-entropy Steiner triangulation of G is a lower bound on the cost of any linear decision tree for point location in G . By then applying the recent result of Arya *et al.* to the Steiner triangulation Δ we obtain upper and lower bounds that match to within a lower-order term.

A preliminary version of this paper, which dealt only with convex subdivisions, has appeared in the Proceedings of the 19th ACM-SIAM Symposium on Discrete Algorithms (SODA 2008) [Collette et al. 2008].

The remainder of this paper is organized as follows: Section 2 presents definitions and notations used throughout the paper. Section 3 shows how to compute a near-minimum-entropy triangulation of a simple polygon. Finally, Section 4 applies these tools to obtain our point location structure for connected planar subdivisions.

2. PRELIMINARIES

In this section we give definitions, notation, and background required in subsequent sections.

Interiors and Boundaries. For a set $P \subseteq \mathbb{R}^2$, we denote the boundary of P by ∂P and the interior of P by $\text{int}(P)$. The closure of P is denoted by $\text{clo}(P) = P \cup \partial P$.

Triangles and Convex Polygons. For the purposes of this paper, a *triangle* is the common intersection of at most 3 closed halfplanes. This includes triangles with infinite area and triangles having 0, 1, 2, or 3, vertices. Similarly, a *convex k -gon* is the common intersection of at most k closed halfplanes.

For a closed region $X \subseteq \mathbb{R}^2$, a *triangulation* of X is a set of triangles whose interiors are pairwise disjoint and whose union is X . We use the convention that, unless X is explicitly mentioned, the triangulation in question is a triangulation of \mathbb{R}^2 . This definition of a trian-

³Although significant breakthroughs have recently been made in this area [Chan 2006; Pătraşcu 2006], we deliberately do not survey algorithms that require the vertices of the subdivision to be on integer coordinates.

gulation is often referred to as a *Steiner triangulation* since it allows vertices of the triangles to be anywhere in X , and not at some finite predefined set of locations.

Simple Polygons, Pseudotriangles, and Geodesic Triangles. A (*near-simple*) *polygon* P is a closed subset of \mathbb{R}^2 whose boundary is piecewise linear and such that $\text{int}(P)$ is homeomorphic to an open disk. Note that this definition of a polygon implies that every bounded face of a connected planar subdivision is a polygon. Also, triangles, as defined above, are polygons. Note that near-simple polygons are slightly more general than simple polygons, for which ∂P is a simple closed curve. However, our definition is sufficiently close that algorithms designed for simple polygons continue to work with near-simple polygons.

A reflex chain in a polygon P is a consecutive sequence of vertices p_i, \dots, p_j of P , where the internal angle at p_k is at least π , for all $k \in \{i+1, \dots, j-1\}$. A *pseudotriangle* is a polygon whose boundary consists of 3 reflex chains. An i -convex pseudotriangle ($i \in \{0, 1, 2, 3\}$) is a pseudotriangle in which i of the reflex chains consist of single line segments.

A *shortest path* between points $a, b \in P$, denoted \overline{ab}_P is a curve of minimum length that is contained in P and that has a and b as endpoints. For 3 points, $a, b, c \in P$, a *geodesic triangle* in P , denoted $\Delta_P abc$ is the union of all shortest paths of the form \overline{xc}_P , where $x \in \overline{ab}_P$. Geodesic triangles are closely related to pseudotriangles. In particular, every geodesic triangle t consists of a pseudotriangle \hat{t} and three paths joining the three convex vertices of \hat{t} to a , b , and c [Rote et al. 2006, Section 8.4].

Classification Problems and Classification Trees. A *classification problem* over a domain \mathcal{D} is a function $\mathcal{P} : \mathcal{D} \mapsto \{0, \dots, k-1\}$. A d -ary *classification tree* is a full d -ary tree⁴ in which each internal node v is labelled with a function $P_v : \mathcal{D} \mapsto \{0, \dots, d-1\}$ and for which each leaf ℓ is labelled with a value $d(\ell) \in \{0, \dots, k-1\}$. The *search path* of an input p in a classification tree T starts at the root of T and, at each internal node v , evaluates $i = P_v(p)$ and proceeds to the i th child of v . We denote by $T(p)$ the label of the final (leaf) node in the search path for p . We say that the classification tree T *solves* the classification problem \mathcal{P} over the domain \mathcal{D} if, for every $p \in \mathcal{D}$, $\mathcal{P}(p) = T(p)$.

For each leaf ℓ of a classification tree T there is a maximal subset $r(\ell) \subseteq \mathcal{D}$ such that the search path for any $p \in r(\ell)$ ends at ℓ . Thus, the leaves of T partition \mathcal{P} into subsets.

In this paper, we are especially concerned with *linear decision trees*. These are binary classification trees for a problem \mathcal{P} over the domain \mathbb{R}^2 . Each internal node v of a linear decision tree contains a linear inequality $P_v(x, y) = ax + by \geq c$, and the node evaluates to 1 or 0 depending on whether the query point (x, y) satisfies the inequality or not, respectively. Geometrically, each internal node of T is labelled with a directed line and the decision to go to the left or right child depends on whether p is to the left or right (or on) this line. An immediate consequence of this is that, for each leaf ℓ of T , the closure of $r(\ell)$ is a convex polygon.

Probability. Throughout this paper D is a probability measure over \mathbb{R}^2 that represents the query distribution. The notation $\Pr(X)$ denotes the probability of event X under the probability measure D . The notation $\Pr(Y|X)$ denotes the conditional probability of Y given X , i.e., $\Pr(Y|X) = \Pr(Y \cap X) / \Pr(X)$. For any set S , we use the shorthand $\cup S$ to denote $\bigcup_{s \in S} s$.

For a set S of subsets of \mathbb{R}^2 , we define the *induced entropy of S* , denoted by $H(S)$ as $H(S) = \sum_{s \in S} \Pr(s|\cup S) \log(1/\Pr(s|\cup S))$. For two sets $S_1, S_2 \subseteq \mathbb{R}^2$ with $\cup S_1 = \cup S_2$, the *joint entropy of S_1 and S_2* , is $H(S_1, S_2) = \sum_{s_1 \in S_1} \sum_{s_2 \in S_2} \Pr(s_1 \cap s_2) \log(1/\Pr(s_1 \cap s_2))$. It is well-known that $H(S_1, S_2) \leq H(S_1) + H(S_2)$ (see, for example, Gray [Gray 2008, Lemma 2.3.2]).

⁴A full d -ary tree is a rooted ordered tree in which each non-leaf node has exactly d children.

We will sometimes abuse terminology slightly by referring to a triangulation Δ of X as a *partition* of X into triangles, although strictly speaking this is not true since the triangles in Δ are closed sets that overlap at their boundaries. We will then continue the abuse by computing the induced entropy of Δ . This introduces a technical difficulty in that $\sum_{t \in \Delta} \Pr(t) \geq 1$ and inequality is possible if there exists sets $Y \subset \mathbb{R}^2$ such that the area of Y is 0 and $\Pr(Y) > 0$. To avoid this technical difficulty, we will assume that D is *nice* in the sense that, if the area of Y is 0 then $\Pr(Y) = 0$. This implies that, for every t in Δ $\Pr(t) = \Pr(\text{int}(t))$. This assumption will avoid lengthy technical but uninteresting cases in our analysis. In practice, this problem can be avoided by using a symbolic perturbation of the query point.

The probability measures used in this paper are usually defined over \mathbb{R}^2 . We make no assumptions about how these measures are represented, but we assume that an algorithm can, in constant time, perform each of the following two operations:

- (1) given a triangle t , compute $\Pr(t)$, and
- (2) given a triangle t and a point x at the intersection of two of t 's supporting lines, compute a line ℓ that contains x and that partitions t into two open triangles t_0 and t_1 such that $\Pr(t_0) \leq \Pr(t_1) \leq \Pr(t)/2$.

Requirement 2 is used only for convenience in describing our data structure. It is not strictly necessary, but its use greatly simplifies the exposition of our results. To eliminate requirement 2, one can use the same method described by Collette *et al.* [Collette et al. 2008, Section 5].

For a classification tree T that solves a problem $P : \mathcal{D} \mapsto \{0, \dots, k-1\}$ and a probability measure D over \mathcal{D} , the *expected search time* of T is the expected length of the search path for p when p is drawn at random from \mathcal{D} according to D . Thus, the expected search time of T (under distribution D) can be written as

$$\mu_D(T) = \sum_{\ell \in L(T)} \Pr(r(\ell)) \times \text{depth}(\ell) ,$$

where $L(T)$ denotes the leaves of T and $\text{depth}(\ell)$ denotes the length of the path from the root of T to ℓ . For any tree T we use $V(T)$ to denote the vertices of T .

The following theorem, which is a restatement of (half of) Shannon's Fundamental Theorem for a Noiseless Channel [Shannon 1948, Theorem 9], is what all previous results on distribution-sensitive planar point location use to establish their optimality:

THEOREM 1 (FUNDAMENTAL THEOREM FOR A NOISELESS CHANNEL). *Let $\mathcal{P} : \mathcal{D} \mapsto \{0, \dots, k-1\}$ be a classification problem and let $p \in \mathcal{D}$ be selected from a distribution D such that $\Pr\{\mathcal{P}(p) = i\} = p_i$, for $0 \leq i < k$. Then, any d -ary classification tree T that solves \mathcal{P} has*

$$\mu_D(T) \geq \sum_{i=0}^{k-1} p_i \log_d(1/p_i) . \quad (2)$$

Theorem 1 is typically applied to the point location problem by treating point location as the problem of classifying the query point p based on which face of G contains it. In this way, we obtain the lower bound in (1).

3. MINIMUM ENTROPY TRIANGULATIONS

Let P be a simple polygon with n vertices, denoted p_0, \dots, p_{n-1} as they occur, in counter-clockwise order, on the boundary of P . We will show how to find a triangulation of P that has near-minimum entropy. That is, we will find a triangulation $\Delta = \Delta(P, D)$ such that $H(\Delta)$ is near-minimum over all triangulations of P . In order to shorten the formulas in this

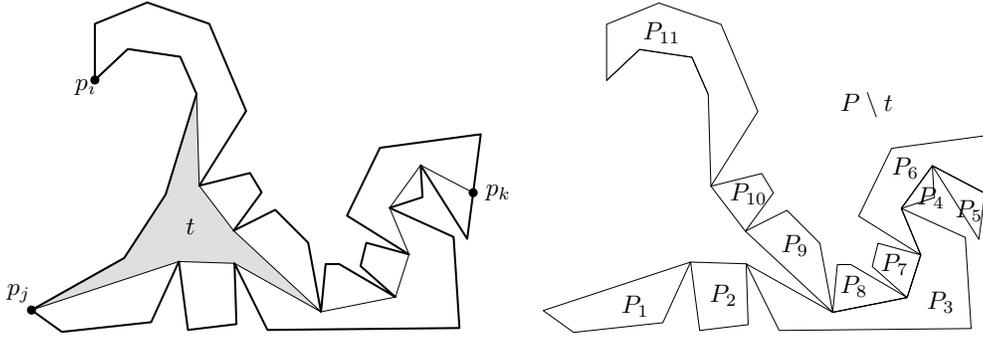


Fig. 1. The geodesic triangle $t = \Delta p_i p_j p_k$ partitions P into several pieces P_1, \dots, P_m .

section, we will implicitly condition the distribution D on P . More precisely, throughout this section the notation $\Pr(X)$ should be treated as shorthand for $\Pr(X|P)$.

Our strategy for computing the minimum-entropy triangulation of P is top-down. It is best compared to the top-down algorithm for constructing a near-optimal binary search tree [Mehlhorn 1975]. This algorithm finds a value x such that $\Pr\{q < x\} \leq 1/2$ and $\Pr\{q > x\} \leq 1/2$. The value x becomes the root of the tree and the algorithm recursively computes the left and right subtrees containing the values less than x and greater than x , respectively.

Our strategy starts with a reflex chain C of P and constructs a geodesic triangle t with C on its boundary (t can be viewed as the root of a tree). The geodesic triangle t has the property that it is *probability reducing*; the probability that the query point q is contained in any particular components of $P \setminus t$ is at most $1/2$. Next, we triangulate the interior of t (a pseudotriangle), also using a top-down strategy. Finally, we recursively triangulate the components of $P \setminus t$.

3.1. The Triangulation $\Delta = \Delta(P, D)$

Our triangulation algorithm is recursive and takes as input a polygon P and a reflex chain p_i, \dots, p_j on the boundary of P . If P is a triangle, then there is nothing to do, so the algorithm outputs P and terminates. Otherwise, the algorithm first selects a point p_k on the boundary of P and adds all the edges of the geodesic triangle $t = \Delta p_i p_j p_k$ to the triangulation Δ . Observe that removing t from P disconnects P into components P_1, \dots, P_m where $\text{clo}(P_i)$ is a polygon that shares a reflex chain C_i with the pseudotriangle t (see Figure 1). The point p_k is selected in such a way that, for all $i \in \{1, \dots, m\}$, $\Pr(P_i) \leq (1/2) \Pr(P)$.⁵ Each of the sub-polygons P_1, \dots, P_m can then be triangulated recursively by applying the algorithm to P_i and the reflex chain C_i .

To complete the triangulation Δ all that remains is to partition the pseudotriangle $\hat{t} = \text{clo}(t \setminus \partial t)$ into triangles. To do this, we first partition \hat{t} into at most one triangle t' and three 2-convex pseudotriangles. Refer to Figure 3.a. Let x_0, x_1 , and x_2 be the three convex vertices of \hat{t} . We shoot a ray r_i from each x_i into the interior of \hat{t} so that it bisects the angle at x_i . These three rays bound a triangle t' in the interior of \hat{t} . By removing the infinite segment of r_i that does not bound t' , we obtain three segments s_0, s_1 , and s_2 that partition the interior of \hat{t} into a triangle t' and three 2-convex pseudotriangles t_0, t_1 , and t_2 .

Let Q_i be the connected component of $(\text{int}(P) \setminus \hat{t}) \cup t_i$ that contains t_i . To complete the triangulation we will partition t_i into triangles, for each $i \in \{0, 1, 2\}$, using a recur-

⁵The existence of such a point p_k is readily established by a standard continuity argument; see Bose *et al.* [Bose et al. 2007] for an example.

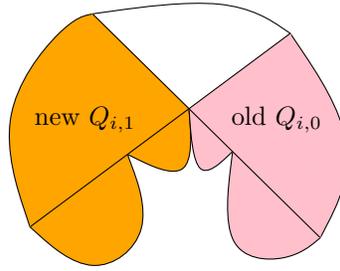


Fig. 2. Showing the existence of the edge e_i .

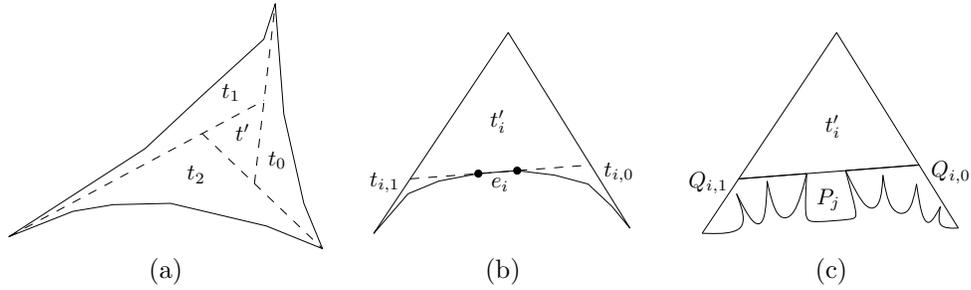


Fig. 3. Partitioning (a) a pseudotriangle \hat{t} into three 2-convex pseudotriangles t_0, t_1, t_2 and one triangle t' (b) a 2-convex pseudotriangle t_i into one triangle t'_i and two 2-convex pseudotriangles $t_{i,0}$ and $t_{i,1}$, and (c) Q_i into 4 pieces.

sive algorithm. This algorithm selects an edge e_i of the reflex chain in t_i and extends e_i in both directions until it reaches the boundary of t_i (see Figure 3.b). The resulting line segment partitions t_i into a triangle t'_i , and two 2-convex pseudotriangles $t_{i,0}$ and $t_{i,1}$ that are triangulated recursively. At the same time, Q_i is partitioned into up to 4 pieces (see Figure 3.c):

- (1) the triangle t'_i , and
- (2) a subpolygon P_j incident to e_i ,
- (3) The two connected components $Q_{i,0}$ and $Q_{i,1}$ of $Q_i \setminus t'_i$ that contain $t_{i,0}$ and $t_{i,1}$, respectively.

The edge e_i is selected so that $\Pr(Q_{i,b}) \leq (1/2) \Pr(Q_i)$ for each $b \in \{0, 1\}$. To see that such an edge exists, imagine trying the edges as they occur, in clockwise order, on the boundary of t'_i . The first edge we try has $\Pr(Q_{i,1}) = 0$ and the last edge we try has $\Pr(Q_{i,0}) = 0$. Furthermore, as we move from one edge to the next $\Pr(Q_{i,1})$ increases, $\Pr(Q_{i,0})$ decreases, and $\Pr(Q_{i,1}) + \Pr(Q_{i,0}) \leq \Pr(Q_i)$. The only possible obstacle to finding the edge e_i we seek could be that, when moving from one edge to the next, we go from a state in which $Q_{i,0} > (1/2) \Pr(Q_i)$ to a state in which $Q_{i,1} > (1/2) \Pr(Q_i)$. However, this can never happen since (the old) $Q_{i,0}$ and (the new) $Q_{i,1}$ are disjoint (see Figure 2), so that

$$\Pr\{\text{old } Q_{i,0}\} + \Pr\{\text{new } Q_{i,1}\} \leq \Pr\{Q_i\} .$$

We conclude that this process therefore finds an edge e_i with the desired properties.

This completes the description of the triangulation Δ . A partially completed triangulation is show in Figure 4.

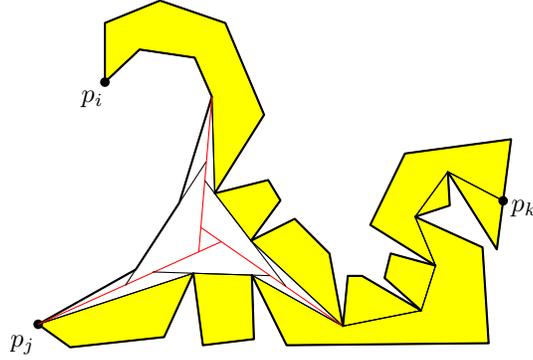


Fig. 4. The triangles obtained during the first level of recursive triangulation. The shaded leaves are triangulated recursively.

3.2. The Δ -Tree $T = T(P, D)$

In order to study the entropy of the triangulation Δ defined above, we will impose a tree structure on the pieces of P induced by the triangles in Δ . The Δ -tree $T = T(P, D)$ for P is a tree whose nodes are subpolygons of P and which has the property that, for any node y that is the child of a node x , $y \subseteq x$.

The tree T has three different kinds of nodes, called P-nodes, T-nodes, and Q-nodes. The root r of T is the polygon P and is a *P-node*. The root of T has the following children (defined in terms of the construction algorithm in the previous section; see Figure 5):

- (1) Each subpolygon P_i whose boundary does not share a segment with \hat{t} is a child of r and is a P-node.
- (2) The subpolygon $Q = \hat{t} \cup Q_0 \cup Q_1 \cup Q_2$ is a child of r and is called a *T-node*.

The subpolygon Q has three children Q_0, Q_1, Q_2 that are called *Q-nodes*. The subtree rooted at Q_i is a ternary tree corresponding to the recursive partitioning of t_i and Q_i done by the algorithm. The leaves of this subtree are P-nodes and the internal nodes of this subtree are Q-nodes. Each internal node has up to 3 children, up to 1 of which may be a P-node corresponding to a subpolygon P_j and up to two of which may be Q-nodes.

Note that the above definition yields a tree whose leaves are P-nodes that correspond to the subpolygons P_1, \dots, P_m obtained by removing t from P . The subtree rooted at each such leaf is obtained recursively from the recursive triangulation of P_i .

Now that we have defined the tree T , we study some of its properties. Our first lemma says that T does a good job of splitting P based on its probabilities.

LEMMA 1. *Let P be a polygon, let D be a probability measure over \mathbb{R}^2 , and let $T = T(P, D)$ be the Δ -tree for (P, D) . Let x be a node of T whose depth is i . Then $\Pr(x) \leq (1/2^{\lfloor i/4 \rfloor}) \Pr(P)$.*

PROOF. Assume $i \geq 4$, otherwise there is nothing to prove. Suppose, for now, that there is an additional P-node (distinct from r) on the path from r to x . Let r' be the first such node. Then, by definition, $\Pr(r') \leq (1/2) \Pr(r)$. If the length of the path from r to r' is at most 4, then we can apply induction on the path from r' to x .

If the length of the path from r to r' is $k > 4$, then this path contains, r , a T-node, and $k-2$ Q-nodes. By construction, for any Q-node y whose parent is a Q-node z , $\Pr(y) \leq (1/2) \Pr(z)$. Therefore, in this case, $\Pr(r') \leq (1/2)^{k-3} \Pr(r) \leq (1/2)^{k/4}$ and we can again apply induction on the path from r' to x .

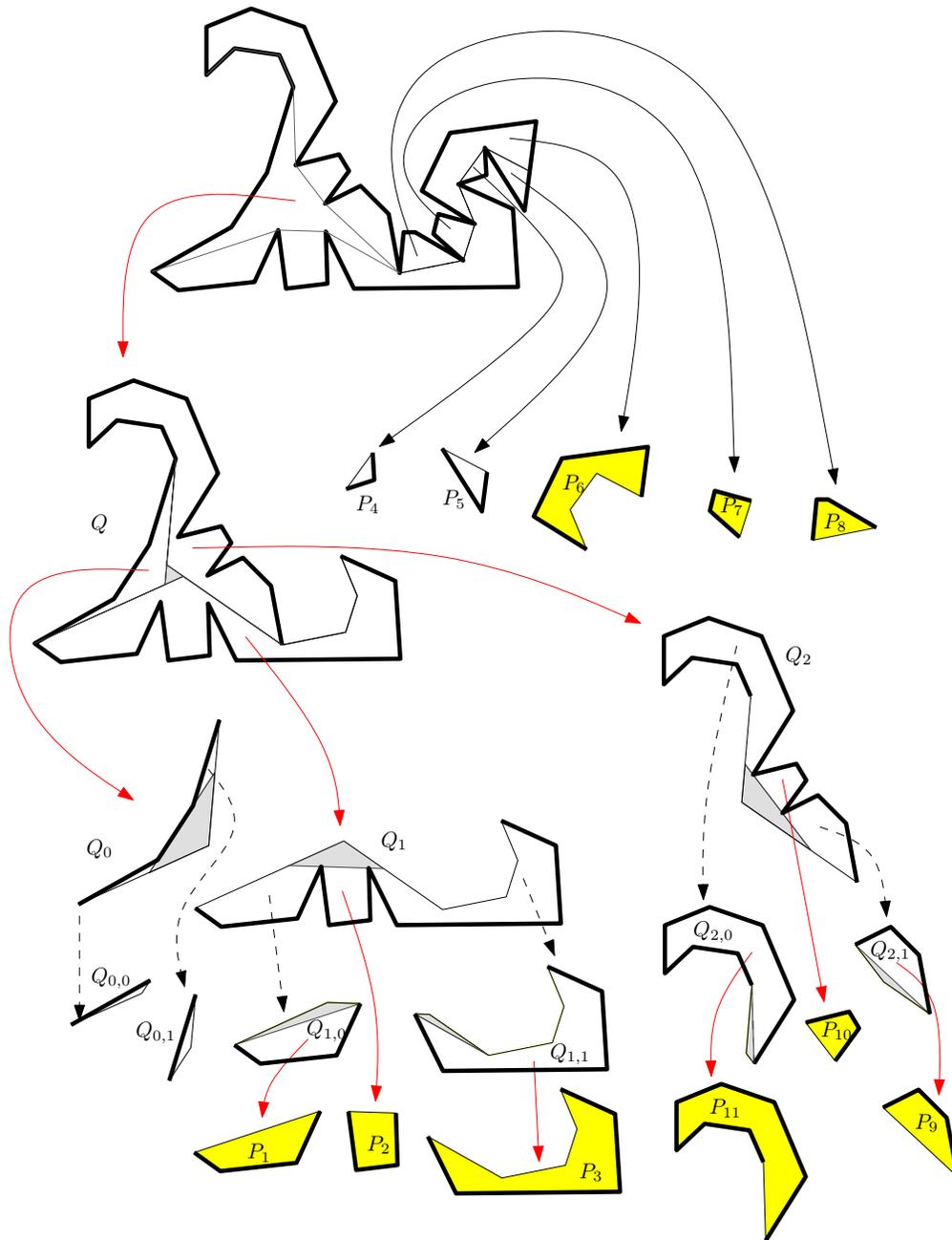


Fig. 5. The Δ -tree T . Shaded leaves in this tree are the root of subtrees obtained recursively. Grey areas show the portions of a node not covered by its children. A black edge from a node x to a node y indicates that $\Pr(y) \leq (1/2)\Pr(x)$. A solid edge leading into a node x indicates that x is separated from the rest of P by a shortest path in P .

Finally, if the path from r to x contains no P-nodes other than r , then it consists of r , a T-node, and $i-2$ Q-nodes. Therefore, $\Pr(x) \leq (1/2)^{i-3} \Pr(r) \leq (1/2)^{i/4} \Pr(r)$, for $i \geq 4$. \square

Our next lemma says that a single line segment does not intersect very many high probability triangles in Δ .

LEMMA 2. *Let P be a polygon, let D be a probability measure over \mathbb{R}^2 , and let $T = T(P, D)$ be the Δ -tree for (P, D) , let $s \subseteq P$ be a line segment, and let $S_i \subseteq V(T)$ be the set of all vertices $x \in V(T)$ that are distance at most i from the root of T and such that $\text{int}(x)$ intersect s . Then $|S_i| \leq 6i^2$*

PROOF. There are 3 types of nodes in T whose interiors intersect s : Type 1 nodes contain one endpoint of s in their interior, Type 2 nodes contain both endpoints of s in their interior, and Type 0 nodes contain no endpoints of s in their interior. Notice that each level of T contains at most 1 Type 2 node or 2 Type 1 nodes, so the total number of Type 1 and Type 2 nodes at distance at most i from the root is at most $2i + 1$. Thus, all that remains is to bound the number of Type 0 nodes whose distance from the root is at most i .

Let w be any P-node such that $\text{int}(w)$ does not contain either endpoint of s . Note that $P \setminus w$ is *half-polygon*, i.e., it is defined by some shortest path between two points on the boundary of P . For any two points in the closure of $P \setminus w$, the shortest path between them remains in the closure of $P \setminus w$. Therefore, if both endpoints of s are in the closure of $P \setminus w$ then so is $\text{int}(s)$. In other words, s does not intersect $\text{int}(w)$. Therefore, all Type 0 nodes are either T-nodes or Q-nodes.

Note that the parent of every T-node is a P-node. Thus, the parent of a Type 0 T-node is a P-node of Type 1 or Type 2. Furthermore, looking more closely at the definition of Q-nodes, we see that two sibling Q-nodes, x and y , whose parent is also a Q-node, are not mutually visible, i.e., there is no line segment $s \subseteq P$ that intersects both $\text{int}(x)$ and $\text{int}(y)$. In particular, no two such Q-nodes can be both of Type 0.

All of this implies that each of the at most $2i + 1$ Type 1 or Type 2 P-nodes is adjacent to at most 1 Type 0 T-node, and this T-node produces at most 3 paths of Type 0 Q-nodes. Each such path is of length at most i . Therefore, the total number of nodes in T that intersect s is at most $2i \cdot 3i = 6i^2$. \square

3.3. Minimum-Entropy Triangulation

Next, we show that the triangulation Δ defined above is nearly-minimum entropy over all possible triangulations of P . We do this by developing a technique for lower-bounding the entropy of one triangulation in terms of the entropy of another triangulation. We then show how to apply this technique to lower bound the entropy of any triangulation Δ^* in terms of the entropy of Δ .

To obtain lower bounds on the entropy of a triangulation Δ^* , consider the following easily proven observation: If each triangle in Δ^* intersects at most c triangles of some triangulation Δ then $H(\Delta^*) \geq H(\Delta) - \log c$.⁶ This observation allows us to use Δ to prove a lower bound on the entropy of a triangulation Δ^* . Unfortunately, the condition that each triangle of Δ^* intersect at most c triangles of Δ is too restrictive for our purposes. Instead, we require the following stronger result:

LEMMA 3. *Let D be a probability measure over \mathbb{R}^2 . Let Δ and Δ^* be triangulations, and let $\{\Delta_1, \dots, \Delta_m\}$ be a partition of the triangles in Δ . Suppose that, for all $i \in \{1, \dots, m\}$*

⁶Proof: Consider the set $X = \{t^* \cap t : t^* \in \Delta^*, t \in \Delta\}$. Each triangle of Δ^* contributes at most c pieces to X , so we have $H(\Delta) \leq H(X) \leq H(\Delta^*) + \log c$.

and for each triangle $t^* \in \Delta^*$, t^* intersects at most c_i triangles in Δ_i . Then

$$H(\Delta) \leq H(\Delta^*) + H(\{\cup\Delta_1, \dots, \cup\Delta_m\}) + \sum_{i=1}^m \Pr(\cup\Delta_i) \log c_i .$$

Intuitively, Lemma 3 can be thought of as follows: If we tell an observer which of the Δ_i a point p drawn according to D occurs in then the amount of information we are giving the observer about the experiment is at most $H(\{\cup\Delta_1, \dots, \cup\Delta_m\})$. However, after giving away this information, we are able to apply the simple observation in the previous paragraph, since each triangle in Δ^* intersects at most c_i elements of each Δ_i . Thus, Lemma 3 is really just m applications of the simple observation. The following proof formalizes this:

PROOF.

$$\begin{aligned} H(\Delta^*) + H(\{\cup\Delta_1, \dots, \cup\Delta_m\}) &\geq H(\Delta^*, \{\cup\Delta_1, \dots, \cup\Delta_m\}) \\ &= \sum_{i=1}^m \sum_{t^* \in \Delta^*} \Pr(t^* \cap \Delta_i) \log(1/\Pr(t^* \cap \Delta_i)) \\ &\geq \sum_{i=1}^m \sum_{t \in \Delta_i} \sum_{t^* \in \Delta^*} \Pr(t^* \cap t) (\log(1/\Pr(t^* \cap t)) - \log c_i) \\ &\geq \sum_{i=1}^m \sum_{t \in \Delta_i} \Pr(t) \log(1/\Pr(t)) - \sum_{i=1}^m \Pr(\cup\Delta_i) \log c_i \\ &= H(\Delta) - \sum_{i=1}^m \Pr(\cup\Delta_i) \log c_i , \end{aligned}$$

and this completes the proof. \square

The remainder of our argument involves partitioning the triangles of Δ into subsets $\Delta_1, \dots, \Delta_m$ and then showing that $H(\cup\Delta_1, \dots, \cup\Delta_m)$ and $\sum_{i=1}^m \Pr(\cup\Delta_i) \log c_i$ are not too big. To help us, we will use the Δ -tree T . For a node x in T with children x_1, \dots, x_k , let $t(x) = x \setminus (\cup_{i=1}^k x_i)$ be the portion of x not covered by x 's children. Note that $t(x)$ is always either the empty set or is a triangle in Δ (see Figure 5). In fact, for every triangle $t \in \Delta$, there is exactly one $x \in V(T)$ such that $t(x) = t$, and for every $x \in V(T)$ such that $t(x)$ is non-empty there is exactly one $t \in \Delta$ such that $t(x) = t$. This implies that⁷

$$H(\Delta) = \sum_{t \in \Delta} \Pr(t) \log(1/\Pr(t)) = \sum_{x \in V(T)} \Pr(t(x)) \log(1/\Pr(t(x))) .$$

For a node $x \in V(T)$, we define $\Pr(x) = \Pr(t(x))$ is the probability that a point drawn from D is contained in $t(x)$.

Next we apply Lemma 3 to obtain a lower bound on the entropy of any triangulation Δ^* .

LEMMA 4. *Let P be a simple polygon, let D be a probability measure over \mathbb{R}^2 , and consider the triangulation $\Delta = \Delta(P, D)$. Then, for any triangulation Δ^* of P ,*

$$H(\Delta) \leq H(\Delta^*) + O(H(\Delta^*)^{1/2} + 1) .$$

PROOF. Let $T = T(P, D)$ be the Δ -tree for (P, D) . Partition the nodes of T into groups G_1, G_2, \dots where

$$G_i = \{x \in V(T) : 1/2^i < \Pr(x) \leq 1/2^{i-1}\} .$$

⁷Here, and throughout the remainder, we slightly abuse notation by using the convention that $0 \cdot \log(1/0) = 0$.

In the following we will fix a value α , $0 < \alpha < 1$, to be defined later. A group G_i is *large* if it contains at least $2^{\alpha i}/2e$ elements, otherwise G_i is *small*. (Euler's constant, e , appears here because of a maximization argument that appears later in this proof.) Let I^+ denote the index set of the large groups, i.e., $I^+ = \{i \in \mathbb{N} : |G_i| \geq 2^{\alpha i}\}$. Let $I^- = \mathbb{N} \setminus I^+$ be the index set of the small groups.

Note that, for any group G_i , Lemma 1 ensures that all elements of G_i have depth at most $4i$ in T . Therefore, Lemma 2 ensures that any triangle of Δ^* intersects at most $3 \times 6 \times (4i)^2 = 288i^2$ triangles of G_i . Therefore, applying Lemma 3 with $c_i = 288i^2$, we obtain:

$$H(\Delta) \leq H(\Delta^*) + H(\{\cup G_i : i \in \mathbb{N}\}) + \sum_{i=1}^{\infty} \Pr(\cup G_i) \log(288i^2) . \quad (3)$$

Thus, all that remains is to bound the contribution of the last two terms on the right hand side of (3). First,

$$\begin{aligned} \sum_{i=1}^{\infty} \Pr(\cup G_i) \log(288i^2) &= \sum_{i=1}^{\infty} \sum_{t \in G_i} \Pr(t) \log(288i^2) \\ &\leq \sum_{i=1}^{\infty} \sum_{t \in G_i} \Pr(t) (O(1) + 2 \log \log(1/\Pr(t))) \\ &= \sum_{t \in \Delta} \Pr(t) (O(1) + 2 \log \log(1/\Pr(t))) \\ &= O(1 + \log H(\Delta)) , \end{aligned}$$

where the last equality follows from Jensen's Inequality. Finally, we show that the contribution of $\bar{H} = H(\{\cup\{G_i\} : i \in \mathbb{N}\})$ is at most $O(H(\Delta)^{1/2})$.

$$\begin{aligned} \bar{H} &= H(\{\cup\{G_i\} : i \in \mathbb{N}\}) \\ &= \sum_{i=1}^{\infty} \Pr(\cup G_i) \log(1/\Pr(\cup G_i)) \\ &= \sum_{i \in I^+} \Pr(\cup G_i) \log(1/\Pr(\cup G_i)) + \sum_{i \in I^-} \Pr(\cup G_i) \log(1/\Pr(\cup G_i)) \\ &\leq \sum_{i \in I^+} \Pr(\cup G_i) \log(1/\Pr(\cup G_i)) + \sum_{i \in I^-} (2^{\alpha i}/2^i e) \log(2^i e/2^{\alpha i}) \end{aligned}$$

This last inequality follows because the expression $x \log(1/x)$ is monotonically increasing for $x \in (0, 1/e]$. Continuing, we have

$$\begin{aligned} \bar{H} &\leq \sum_{i \in I^+} \Pr(\cup G_i) \log(1/\Pr(\cup G_i)) + \sum_{i=1}^{\infty} 2^{(\alpha-1)i} (\log(2^{(1-\alpha)i} + \log e)) \\ &\leq \sum_{i \in I^+} \Pr(\cup G_i) \log(1/\Pr(\cup G_i)) + (1-\alpha) \cdot \sum_{i=1}^{\infty} i 2^{(\alpha-1)i} + (\log e) \sum_{i=1}^{\infty} 2^{(\alpha-1)i} \\ &= \sum_{i \in I^+} \Pr(\cup G_i) \log(1/\Pr(\cup G_i)) + (1-\alpha) \cdot \left(\frac{(1/2)^{1-\alpha}}{(1-(1/2)^{1-\alpha})^2} \right) + \frac{\log e}{1-(1/2)^{1-\alpha}} \\ &\leq \sum_{i \in I^+} \Pr(\cup G_i) \log(1/\Pr(\cup G_i)) + (1-\alpha) \cdot O(1/(1-\alpha)^2) + O(1/(1-\alpha)) , \end{aligned}$$

where the last equality is obtained using the Taylor series expansion for e^x to obtain the inequality $1 - 1/2^x \geq x \ln 2 - (x \ln 2)^2/2$ for x close to 0. Continuing, we get

$$\begin{aligned}
\bar{H} &\leq \sum_{i \in I^+} \Pr(\cup G_i) \log(1/\Pr(\cup G_i)) + O(1/(1-\alpha)) \\
&\leq \sum_{i \in I^+} \Pr(\cup G_i) \log(2^i/|G_i|) + O(1/(1-\alpha)) \\
&\leq \sum_{i \in I^+} \Pr(\cup G_i) \log(2^i/2^{\alpha i}) + O(1/(1-\alpha)) \\
&= \sum_{i \in I^+} \Pr(\cup G_i)(1-\alpha)i + O(1/(1-\alpha)) \\
&= (1-\alpha) \sum_{i \in I^+} \Pr(\cup G_i)i + O(1/(1-\alpha)) \\
&= (1-\alpha) \sum_{i \in I^+} \Pr(\cup G_i) \log(2^i) + O(1/(1-\alpha)) \\
&= (1-\alpha) \sum_{i \in I^+} \sum_{t \in G_i} \Pr(t) \log(2^i) + O(1/(1-\alpha)) \\
&= (1-\alpha) \sum_{i \in I^+} \sum_{t \in G_i} \Pr(t) \log(1/\Pr(t)) + O(1 + 1/(1-\alpha)) \\
&\leq (1-\alpha)H(\Delta) + O(1 + 1/(1-\alpha)) \\
&\leq O(H(\Delta)^{1/2} + 1)
\end{aligned}$$

where the last inequality is obtained by setting $\alpha = 1 - 1/H(\Delta)^{1/2}$. Thus, we have shown that

$$H(\Delta) \leq H(\Delta^*) + O(H(\Delta)^{1/2} + 1) , \quad (4)$$

which implies that $H(\Delta) = O(H(\Delta^*) + 1)$. Applying this to the right hand side of (4) yields $H(\Delta) \leq H(\Delta^*) + O(H(\Delta^*)^{1/2} + 1)$, completing the proof. \square

Lemma 4 shows that the triangulation $\Delta = \Delta(P, D)$ defined previously is nearly minimum-entropy over all triangulations of P . The following theorem gives an algorithmic version of Lemma 4.

THEOREM 2. *Let P be a simple polygon with n vertices, and let D be a probability measure over \mathbb{R}^2 . Then there exists an $O(n \log n)$ time algorithm that computes a triangulation Δ' of P having $O(n)$ triangles and such that, for any triangulation Δ^* of P ,*

$$H(\Delta') \leq H(\Delta^*) + O(H(\Delta^*)^{1/2} + 1) .$$

PROOF. We show how the construction of the triangulation Δ described in Section 3.1 can be modified to run in $O(n \log n)$ time. When constructing Δ the first step is to find the third vertex p_k of the geodesic triangle $t = \Delta p_i p_j p_k$. This can be accomplished in $O(n)$ time by computing the shortest path trees from p_i and p_j to all other vertices of P and using these to find p_k . For an example of a similar computation, see Bose *et al.* [Bose et al. 2007, Section 2.2].

Next, \hat{t} is split into three 2-convex pseudotriangles t_0, t_1, t_2 , which is easily accomplished in $O(n)$ time. The last step, before recursing, is to triangulate each of t_0, t_1, t_2 . This step can be accomplished in $O(n)$ time using a 2-sided exponential searching trick that was used by Mehlhorn [Mehlhorn 1975] in the construction of biased binary search trees (see also, Collette *et al.* [Collette et al. 2008, Theorem 1]).

Finally, the algorithm recurses on each of the pieces P_1, \dots, P_m . In this way, we obtain a divide-and-conquer algorithm for constructing Δ . Unfortunately, this algorithm may have running time $\Omega(n^2)$ since there is no bound significantly smaller than n on the size of an individual subproblem P_i . To overcome this, before recursing on a subproblem P_i we check if it contains more than $n/2$ vertices. If so, then rather than recursing normally on P_i we choose a geodesic triangle t^* , one of whose sides is the reflex chain C_i and such that removing t^* from P_i leaves a set of subpolygons $P_{i,1}, \dots, P_{i,m_i}$ each with at most $n/2$ vertices. This modification then yields an algorithm whose recursion tree has depth $O(\log n)$ and at which the work done at each level is $O(n)$, so the total running time of this algorithm is $O(n \log n)$.

Note that this algorithm yields a triangulation Δ' that is different from Δ . In particular, there may exist one $P_{i,j}$ with $\Pr(P_{i,j}) > \Pr(P_i)/2$. Despite this, all the proofs of Lemmas 1–4 continue to hold almost without modification. The only difference occurs in Lemma 1, which now only guarantees a bound of $1/2^{\lfloor i/8 \rfloor}$, but this has almost no effect on subsequent computations.

Finally, to see that Δ' contains $O(n)$ triangles, we count the different types of edges used in the triangulation Δ' . Some of these edges are edges of P , of which there are at most n . Some of these edges are edges of geodesic triangles, which always connect two vertices of P and do not cross each other, so there are at most $n - 2$ of these. The remaining edges are used to triangulate the interiors of pseudotriangles. A pseudotriangle that has k vertices is triangulated using $3 + 2(k - 3)$ edges. Since the total number of vertices in all pseudotriangles is at most $2n$, this means that there are at most $6n$ edges used to triangulate pseudotriangles. Therefore, the total number of edges used by triangles in Δ' , and hence the number of triangles in Δ' , is $O(n)$. \square

4. POINT LOCATION IN SIMPLE PLANAR SUBDIVISIONS

Next we consider the problem of point location in simple subdivisions. The following theorem of Arya *et al.* [Arya et al. 2007] shows that a low entropy triangulation can be used to make a good point location structure.

THEOREM 3 (ARYA *et al.* 2007). *Let D be a probability measure over \mathbb{R}^2 and let Δ be a triangulation of \mathbb{R}^2 having a total of n triangles. Then there exists a data structure of size $O(n)$ that can be constructed in $O(n \log n)$ time, and for which the expected number of point/line comparisons required to locate the face of G containing a query point p , drawn according to D , is $H(\Delta) + O(H(\Delta)^{1/2} + 1)$.*

The following lemma shows that the entropy of a minimum-entropy triangulation gives a lower bound on the cost of any point location structure.

LEMMA 5. *Let T^* be any linear decision tree for a classification problem \mathcal{P} over \mathbb{R}^2 . Then there exists a linear decision tree T' for \mathcal{P} , such that, for each leaf ℓ of T' , $\text{clo}(r(\ell))$ is a triangle and T' satisfies*

$$\mu_D(T') \leq \mu_D(T^*) + O(\log \mu_D(T^*))$$

for any probability measure D over \mathbb{R}^2 .

PROOF. Each leaf ℓ of T^* has a region $r(\ell)$ that is a convex polygon. If $r(\ell)$ has k sides then the depth of ℓ in T is at least k . To obtain the tree T' replace each such leaf ℓ of T^* by a balanced binary tree of depth $O(\log k)$ by repeatedly splitting the leaf into two children ℓ_1 and ℓ_2 whose regions have $\lceil (k+2)/2 \rceil$ and $\lfloor (k+2)/2 \rfloor$ vertices. For a leaf $\ell \in L(T^*)$, let

$s(\ell)$ denote the set of leaves in T' in the subtree of ℓ . Then

$$\begin{aligned}
\mu_D(T^*) &= \sum_{\ell \in L(T^*)} \Pr(r(\ell)) \cdot \text{depth}(\ell) \\
&= \sum_{\ell \in L(T^*)} \sum_{\ell' \in s(\ell)} \Pr(r(\ell')) \cdot \text{depth}(\ell) \\
&\geq \sum_{\ell \in L(T^*)} \sum_{\ell' \in s(\ell)} \Pr(r(\ell')) \cdot (\text{depth}(\ell') - O(\log(\text{depth}(\ell)))) \\
&= \mu_D(T') - \sum_{\ell \in L(T^*)} \sum_{\ell' \in s(\ell)} \Pr(r(\ell')) \cdot O(\log(\text{depth}(\ell))) \\
&= \mu_D(T') - \sum_{\ell \in L(T^*)} \Pr(r(\ell)) \cdot O(\log(\text{depth}(\ell))) \\
&\geq \mu_D(T') - O(\log(\mu_D(T^*))) ,
\end{aligned}$$

where the last inequality is an application of Jensen's Inequality. \square

Lemma 5 says that for any linear decision tree for point location, there is an underlying triangulation. The entropy of this triangulation gives a lower bound on the cost of the decision tree. Thus, the entropy of a minimum entropy triangulation gives a lower bound on the expected cost of any linear decision tree for point location.

Keeping the above in mind, our point location structure is simple. Let G be a connected planar subdivision whose faces are $F = \{F_1, \dots, F_m\}$ and let D be a probability measure over \mathbb{R}^2 . We assume, without loss of generality that the outer face of G is the complement of a triangle, since otherwise we can add at most 3 vertices and 4 edges to G to make this true. Adding these edges will not increase the entropy of the minimum weight triangulation of G by more than a constant. With this assumption, testing if the query point is in the outer face of G can be done using 3 linear comparisons after which we may safely assume that the query point is contained in an internal face of G .

We triangulate each internal face F_i of G (a near-simple polygon) using Theorem 2 to obtain a triangulation Δ_i . The union of all Δ_i is a triangulation Δ of \mathbb{R}^2 , to which we apply Theorem 3 to obtain a point location structure $R = R(G, D)$ for point location in Δ and hence also in G . The following theorem shows that R is nearly optimal:

THEOREM 4. *Given a connected planar subdivision G with n vertices and a probability measure D over \mathbb{R}^2 , a data structure $R = R(G, D)$ of size $O(n)$ can be constructed in $O(n \log n)$ time that answers point location queries in G . The expected number of point/line comparisons performed by R , for a point p drawn according to D is*

$$\mu_D(R) \leq \mu_D(T^*) + O(\mu_D(T^*)^{1/2} + 1) ,$$

where T^* is any linear classification tree that answers point location queries in G .

PROOF. The space and preprocessing requirements follow from Theorem 2 and Theorem 3. To prove the bound on the expected query time, apply Lemma 5 to the tree T^* and consider the resulting tree T' , each of whose leaves have regions that are triangles and such that

$$\mu_D(T') \leq \mu_D(T^*) + O(\log \mu_D(T^*)) . \quad (5)$$

Observe that each leaf of T' corresponds to a triangle in \mathbb{R}^2 that is completely contained in one of the faces of G . Let Δ' denote this set of triangles and let Δ'_i denote the subset of Δ' contained in F_i . Consider the entropy $H(\Delta')$ of the distribution induced by the leaves

of T' :

$$\begin{aligned}
H(\Delta') &= \sum_{i=1}^m \sum_{t \in \Delta'_i} \Pr(t) \log(1/\Pr(t)) \\
&= \sum_{i=1}^m \Pr(F_i) \sum_{t \in \Delta'_i} \Pr(t|F_i) \log(1/\Pr(t)) \\
&= \sum_{i=1}^m \Pr(F_i) \sum_{t \in \Delta'_i} \Pr(t|F_i) (\log(1/\Pr(t|F_i)) - \log(\Pr(F_i))) \\
&= \sum_{i=1}^m \Pr(F_i) \sum_{t \in \Delta'_i} \Pr(t|F_i) \log(1/\Pr(t|F_i)) + \sum_{i=1}^m \Pr(F_i) \log(1/\Pr(F_i)) \\
&= \sum_{i=1}^m \Pr(F_i) H(\Delta'_i) + H(F) .
\end{aligned} \tag{6}$$

Similarly, the entropy of Δ is given by

$$\begin{aligned}
H(\Delta) &= \sum_{i=1}^m \sum_{t \in \Delta_i} \Pr(t) \log(1/\Pr(t)) \\
&= \sum_{i=1}^m \Pr(F_i) \sum_{t \in \Delta_i} \Pr(t|F_i) \log(1/\Pr(t|F_i)) + H(F) \\
&= \sum_{i=1}^m \Pr(F_i) H(\Delta_i) + H(F) .
\end{aligned}$$

By Theorem 2, the triangles in Δ_i form a nearly-minimum entropy triangulation of F_i . More specifically,

$$H(\Delta_i) \leq H(\Delta'_i) + O(H(\Delta'_i)^{1/2} + 1) . \tag{7}$$

Putting this all together, we have

$$\begin{aligned} H(\Delta) &= \sum_{i=1}^m \Pr(F_i)H(\Delta_i) + H(F) \\ &\leq \sum_{i=1}^m \Pr(F_i)(H(\Delta'_i) + O(H(\Delta'_i)^{1/2} + 1)) + H(F) \end{aligned} \quad (\text{by (7)})$$

$$= H(\Delta') + \sum_{i=1}^m \Pr(F_i)O(H(\Delta'_i)^{1/2} + 1) \quad (\text{by (6)})$$

$$= H(\Delta') + \left(\sum_{i=1}^m \Pr(F_i)O(H(\Delta'_i)) \right)^{1/2} + O(1)$$

(by Jensen's Inequality)

$$= H(\Delta') + \left(O(1) \cdot \sum_{i=1}^m \Pr(F_i) \sum_{t' \in \Delta'_i} \Pr(t'|F_i) \log(1/\Pr(t'|F_i)) \right)^{1/2} + O(1)$$

$$= H(\Delta') + \left(O(1) \cdot \sum_{i=1}^m \sum_{t' \in \Delta'_i} \Pr(t') \log(\Pr(F_i)/\Pr(t')) \right)^{1/2} + O(1)$$

$$\leq H(\Delta') + \left(O(1) \cdot \sum_{i=1}^m \sum_{t' \in \Delta'_i} \Pr(t') \log(1/\Pr(t')) \right)^{1/2} + O(1)$$

$$= H(\Delta') + O(H(\Delta')^{1/2} + 1)$$

$$\leq \mu_D(T') + O(\mu_D(T')^{1/2} + 1)$$

(by Theorem 1)

$$\leq \mu_D(T^*) + O(\mu_D(T^*)^{1/2} + 1) \quad (\text{by (5)})$$

Finally, since we preprocess Δ using Theorem 3, the expected number of comparisons required to answer a query is

$$\begin{aligned} \mu_D(R) &= H(\Delta) + O(H(\Delta)^{1/2} + 1) \\ &\leq \mu_D(T^*) + O(\mu_D(T^*)^{1/2} + 1) \end{aligned}$$

and this completes the proof, and the paper. \square

Acknowledgements

The authors are grateful to anonymous referee who showed us that a small modification of our original calculations could be used to reduce the lower-order term in our results from $O(\tilde{H}^{2/3})$ to $O(\tilde{H}^{1/2})$.

This research began while the last author was a visiting researcher at the Université Libre de Bruxelles and was completed while the author was a visiting researcher at National ICT Australia and the University of Sydney.

REFERENCES

ADAMY, U. AND SEIDEL, R. 1998. On the exact worst case query complexity of planar point location. In *Proceedings of the Ninth Annual ACM-SIAM Symposium on Discrete Algorithms*. 609–618.

- ARYA, S., CHENG, S. W., MOUNT, D. M., AND RAMESH, H. 2000. Efficient expected-case algorithms for planar point location. In *Proceedings of the seventh Scandinavian Workshop on Algorithm Theory*. 353–366.
- ARYA, S., MALAMATOS, T., AND MOUNT, D. M. 2000. Nearly optimal expected-case planar point location. In *Proceedings of the 41st annual Symposium on Foundations of Computer Science*. 208–218.
- ARYA, S., MALAMATOS, T., AND MOUNT, D. M. 2001a. Entropy-preserving cuttings and space-efficient planar point location. In *Proceedings of the Twelfth Annual ACM-SIAM Symposium on Discrete Algorithms*. 256–261.
- ARYA, S., MALAMATOS, T., AND MOUNT, D. M. 2001b. A simple entropy-based algorithm for planar point location. In *Proceedings of the Twelfth Annual ACM-SIAM Symposium on Discrete Algorithms*. 262–268.
- ARYA, S., MALAMATOS, T., MOUNT, D. M., AND WONG, K. C. 2007. Optimal expected-case planar point location. *SIAM J. Comput.* 37, 2, 584–610.
- BOSE, P., DEMAINE, E. D., HURTADO, F., LANGERMAN, S., IACONO, J., AND MORIN, P. 2007. Geodesic ham-sandwich cuts. *Discrete & Computational Geometry* 37, 3, 325–330. Preliminary version appears in *Proceedings of the Twentieth ACM Symposium on Computational Geometry (SoCG 2004)*, pages 1–9. ACM Press, 2004.
- CHAN, T. M. 2006. Point location in $o(\log n)$ time, Voronoi diagrams in $o(n \log n)$ time, and other transdichotomous results in computational geometry. In *Proceedings of the 47th annual Symposium on Foundations of Computer Science*. 333–342.
- COLLETTE, S., DUJMOVIĆ, V., IACONO, J., LANGERMAN, S., AND MORIN, P. 2008. Distribution-sensitive point location in convex subdivisions. In *Proceedings of the 19th ACM-SIAM Symposium on Discrete Algorithms (SODA 2008)*. 912–921.
- DE BERG, M., CHEONG, O., VAN KREVELD, M., AND OVERMARS, M. 2008. *Computational Geometry: Algorithms and Applications* 3rd Ed. Springer-Verlag.
- DOBKIN, D. AND LIPTON, R. 1976. Multidimensional searching problems. *SIAM J. Comput.* 5, 181–186.
- EDELSBRUNNER, H., GUIBAS, L. J., AND STOLFI, J. 1986. Optimal point location in a monotone subdivision. *SIAM J. Comput.* 15, 2, 317–340.
- GOODRICH, M., ORLETSKY, M., AND RAMAIYER, K. 1997. Methods for achieving fast query times in point location data structures. In *Proceedings of the Eighth Annual ACM-SIAM Symposium on Discrete Algorithms*. 757–766.
- GRAY, R. M. 2008. *Entropy and Information Theory*. Free book available online at <http://www-ee.stanford.edu/~gray/it.html>.
- IACONO, J. 2001. Optimal planar point location. In *Proceedings of the Twelfth Annual ACM-SIAM Symposium on Discrete Algorithms*. 240–241.
- IACONO, J. 2004. Expected asymptotically optimal planar point location. *Computational Geometry Theory and Applications* 29, 1, 19–22.
- KIRKPATRICK, D. 1983. Optimal search in planar subdivisions. *SIAM J. Comput.* 12, 1, 28–35.
- LEE, D. T. AND PREPARATA, F. P. 1977. Location of a point in a planar subdivision and its applications. *SIAM J. Comput.* 6, 594–606.
- MEHLHORN, K. 1975. Nearly optimal binary search trees. *Acta Informatica* 5, 287–295.
- MULMULEY, K. 1990. A fast planar partition algorithm. *Journal of Symbolic Computation* 10, 253–280.
- PREPARATA, F. P. 1981. A new approach to planar point location. *SIAM J. Comput.* 10, 473–483.
- PREPARATA, F. P. 1990. Planar point location revisited: A guided tour of a decade of research. *International Journal of Foundations of Computer Science* 1, 1, 71–86.
- PĂTRAȘCU, M. 2006. Planar point location in sublogarithmic time. In *Proceedings of the 47th annual Symposium on Foundations of Computer Science*. 325–332.
- ROTE, G., SANTOS, F., AND STEINU, I. 2006. Pseudo-triangulations — a survey. In *Surveys on Discrete and Computational Geometry—Twenty Years Later*, J. E. Goodman, J. Pach, and R. Pollack, Eds. Contemporary Mathematics Series, vol. 453. American Mathematical Society, 343–410.
- SARNAK, N. AND TARJAN, R. E. 1986. Planar point location using persistent search trees. *Commun. ACM* 29, 7, 669–679.
- SHANNON, C. E. 1948. A mathematical theory of communication. *Bell Systems Technical Journal*, 379–423 and 623–656.