Separating point sets in polygonal environments

Erik D. Demaine* Jeff Erickson† Ferran Hurtado‡ John Iacono§
Stefan Langerman¶ Henk Meijer‖ Mark Overmars** Sue Whitesides††

Abstract

We consider the separability of two point sets inside a polygon by means of chords or geodesic lines. Specifically, given a set of red points and a set of blue points in the interior of a polygon, we provide necessary and sufficient conditions for the existence of a chord and for the existence of a geodesic path which separate the two sets; when they exist we also derive efficient algorithms for their obtention. We study as well the separation of the two sets using a minimum number of pairwise non-crossing chords.

*MIT Laboratory for Computer Science, 200 Technology Square, Cambridge MA 02139, USA. edemaine@mit.edu

†Department of Computer Science, University of Illinois at Urbana-Champaign, 1304 W. Springfield Avenue, Urbana, IL 61801, USA. jeffe@cs.uiuc.edu

‡Departement de Matemàtica Aplicada II, Universitat Politècnica de Catalunya Pau Gargallo 5, 08028 Barcelona, Spain. Ferran.Hurtado@upc.es

§Polytechnic University, 5 MetroTech Center, Brooklyn, NY, 11201, USA. jiacono@poly.edu

¶Chargé de recherches du FNRS, Université Libre de Bruxelles, ULB CP212, Av. F. D. Roosevelt 50, 1050 Bruxelles, Belgium. Stefan.Langerman@ulb.ac.be

‖Dept. of Computing and Information Science, Queen's University, Kingston, Ontario, K7L 3N6, Canada. henk@cs.queensu.ca

**Institute of Information and Computing Sciences, Utrecht University, P.O.Box 80.089, 3508 TB Utrecht, the Netherlands. markov@cs.uu.nl

††School of Computer Science, McGill University, 3480 University St. room 318, Montreal, Quebec H3A 2A7, Canada. sue@cs.mcgill.ca

1 Introduction

Let us consider the following basic question: given two point sets R and B in the interior of a polygon (the *red points* and the *blue points*, respectively), is there a chord separating R from B? This is the starting problem we study in this paper, where we also consider several related problems.

Problems on separability of point sets and other geometric objects have generated a significant body of research in computational geometry, and many kind of separators have been considered, among others lines [11], circles [3, 4], convex polygons [6] or wedges and strips [9]. A thorough study is given in [14]. The main motivations underlying these different works arise in disciplines like spatial data organization or statistical analysis or, more in general, wherever clustering or classification methods are useful.

In the plane (in fact in every dimension, but we focus here on the bidimensional case) an ideal paradigm of separability is by means of a single line, whenever possible. This partitions the plane into two clean regions, and gives an according easy classification rule for any query point. Nevertheless, if we constrain our working space to the interior of a polygon, it is easy to see that points belonging to the same population may lye in many different cells (Figure 1, left). On the opposite sense, it can also be the case that the two point sets are separable simply by one chord, and that no simple separation is available in the underlying plane if the polygon boundary disappears (Figure 1, right).

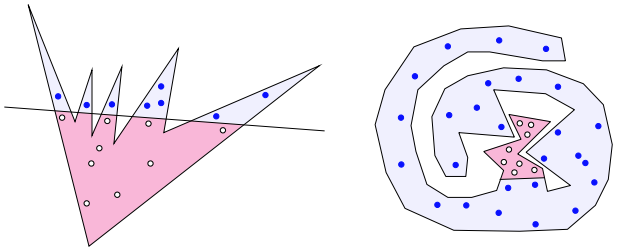

Figure 1: Left: red and blue points that are linearly separable in the plane but generate many regions in the polygon. Right: a chord that separates point sets which cannot be separated with a line in the plane.

On the other hand, the study of basic geometric structures when only the interior of a polygon is taken in consideration leads to deal with its geodesic properties, a topic that has also been attracting a lot of attention; some examples are the geodesic diameter [15], the relative convex hull [17], the 1-center problem [13, 16] and the geodesic Voronoi diagrams [1, 2, 12].

In Section 2 we study, both from the structural and computational viewpoint, the two more natural ways to separate point sets in a polygon: by means of one chord, and by means of a single geodesic line, i.e., a shortest path between two boundary points. In fact, we prove that the necessary and sufficient conditions for both kinds of separability are closely related.

In Section 3 we study the problem of separating the two point sets using as few non-crossing chords as possible. We show that the problem is polynomially solvable when P is very simple and that it becomes NP-complete when P may have holes. In between, there is what we consider to be an intriguing open problem.

Throughout the paper R and B are two given finite sets of points inside a polygon P (the *red points* and the *blue points*, respectively), and their cardinalities are denoted by $r = |R|$ and $b = |B|$. The points in $R \cup B$ are occasionally called *sites*. The total number of sites and polygonal vertices is denoted by n, and we use k for the number of reflex vertices of P.

Let us finally mention that a related problem, the existence of ham-sandwich separators for two points sets in the interior of a polygon, is studied in [5].

2 Linear separability

Let C be a Jordan curve connecting two points on the boundary of a polygon P. C decomposes P into two closed subsets $\overline{C}^+$ and $\overline{C}^-$, with $\overline{C}^+ \cup \overline{C}^- = P$ and $\overline{C}^+ \cap \overline{C}^- = C$. We also write $C^+ = \overline{C}^+ - C$ and $C^- = \overline{C}^- - C$. We say that C *separates* two sets R and B if $R \subseteq C^\alpha$ and $B \subseteq C^\beta$, where $\alpha = +, \beta = -$ or $\alpha = -, \beta = +$. We say that C *weakly separates* two sets R and B if $R \subseteq \overline{C}^\alpha$ and $B \subseteq \overline{C}^\beta$.

When the curve C is a geodesic, the sets $\overline{C}^+$ and $\overline{C}^-$ are called *half-polygons*. The *geodesic convex hull* $GH(S)$ for a set S of points inside a polygon P is the intersection of all half-polygons that contain S.

Theorem 1 *Two sets of points in a polygon P are separable by a chord if and only if their geodesic convex hulls are disjoint.*

Proof: Let C be a chord with endpoints p and q separating sets R and B in P. A chord is a geodesic line, so by the definition of geodesic convex hulls, $GH(R) \subseteq \overline{C}^\alpha$ and $GH(B) \subseteq \overline{C}^\beta$, and so, $GH(R) \cap GH(B) \subseteq C$. Moreover, since C does not contain any points of R or B, $GH(R)$ (resp. $GH(B)$) cannot contain p or q unless that point is a reflex vertex in C^α (resp. C^β), and $GH(R)$ (resp. $GH(B)$) cannot contain an interior point of C unless it contains both p and q. Note that p can only be a reflex vertex for at most one of C^α and C^β. This implies that $GH(R)$ and $GH(B)$ can not intersect.

Now suppose $GH(R)$ and $GH(B)$ are disjoint. Let D be the shortest geodesic with endpoints $u \in GH(R)$ and $v \in GH(B)$, let s be some line segment from D, let ℓ be the bisector of s, and let $m = s \cap \ell$. Define the chord $C = (p, q)$ where p and q are the intersections of ℓ and the boundary of P closest to m in both directions. We claim that the chord C separates $GH(R)$ and $GH(B)$. Suppose that on the contrary, the boundary of $GH(R)$ intersects the segment mp, and let p'

be the intersection closest to m. Let Q be the closed Jordan curve composed of the portion of D from m to u, the boundary of $GH(R)$ from u to p', and the segment from p' to m. Note that the boundary of P does not intersect the interior of the region surrounded by Q, and so the geodesics from m to u and from u to p' (which only intersect in u) are both concave. Consider the ray r from m, orthogonal to the line up', and intersecting that line in u', and let u'' be the first intersection of r and the geodesic from u to p'. Since the geodesic from u to p' is concave, the geodesic distance from m to u, $d(m,u) \geq d(m,u') > d(m,u'')$, and $d(v,u) = d(v,m)+d(m,u) > d(v,m)+d(m,u'') > d(v,u'')$. This implies that D was not the shortest geodesic from R to B, a contradiction. □

Lemma 1 *If $GH(R)$ and $GH(B)$ do not intersect, the two points $u \in GH(R)$ and $v \in GH(B)$ that minimize the length of the geodesic path between u and v can be found in $O(n)$ time.*

Proof: Consider a point $v \in GH(B)$, and a point $u \in GH(R)$. If the geodesic path from v to u (excluding u) intersects $GH(R)$ then u cannot be the the closest point from v in $GH(R)$. The same is true for $GH(B)$ and v. Let $\hat{d}(v,u)$ be the length of that geodesic path if it does not intersect $GH(R)$ and $GH(B)$ and ∞ otherwise. Note that the set of points u on the boundary of $GH(R)$ for which $\hat{d}(v,u) < \infty$ forms a convex chain. Thus the matrix $\hat{d}(v,u)$ for all vertices v of $GH(B)$ and u of $GH(R)$ has every row, and every column unimodal, and so we can find the minimum in the array after querying $O(\log n)$ entries using a two-dimensional Fibonacci search. Querying $\hat{d}(v,u)$ can be done in $O(\log n)$ time after $O(n)$ time preprocessing using the data structure of Guibas and Hershberger [8]. Once the minimum found, we can report the shortest path between the two remaining edges in linear time using the same data structure. □

Corollary 1 *There is a $O(n \log n)$ algorithm which given sets R of red points and B of blue points in a simple polygon P, either finds a chord that separates R and B or reports that no such chord exists.*

Proof: Computing $GH(R)$ and $GH(B)$ can be done in $O(n \log n)$ time and verifying that they don't intersect can be done within the same time bounds. By the previous lemma, we can find the shortest geodesic connecting $GH(A)$ and $GH(B)$ in $O(n)$ time. The separating chord chord can then be found in $O(n)$ time. □

Theorem 2 *Two sets of points in a polygon P are weakly separable by a geodesic line if and only if the interior of their geodesic convex hulls are disjoint.*

Proof: By the definition of a geodesic convex hull, if a geodesic line C weakly separates R and B in P, then $GH(R) \subseteq \overline{C}^{\alpha}$ and $GH(B) \subseteq \overline{C}^{\beta}$, and so $GH(R)$ and $GH(B)$ have disjoint interiors.

On the other hand, if $GH(R)$ and $GH(B)$ have disjoint interiors, then $I = GH(R) \cap GH(B)$, if not empty, is a curve. Furthermore, it is a geodesic between its two endpoints u and v. The boundaries of $GH(R)$ and $GH(B)$ are intersecting on one side of u, and start with two disjoint line segments s_α and s_β on the other side. Draw a line segment from u along a ray bisecting the angle between s_α and s_β, until the first intersection with P, and do the same for v. The resulting curve I' is a geodesic line, and we claim it weakly separates R and B. Indeed, suppose that the ray from u is intersected by $GH(R)$, and let u' be the closest intersection to u. The segment uu' is not intersected by the boundary of P, so that segment must be contained in $GH(R)$, But then uu' must also be included into $GH(B)$ since uu' bisects the angle between s_α and s_β, and therefore $uu' \subseteq I$ which is a contradiction. □

Corollary 2 *There is a $O(n \log n)$ algorithm which given sets R of red points and B of blue points in a simple polygon P, either finds a geodesic that weakly separates R and B or reports that no such geodesic exists.*

Proof: Computing $GH(R)$ and $GH(B)$ can be done in $O(n \log n)$ time and verifying that their interiors don't intersect can be done within the same time bounds. If there is a separating

chord, we can find it in $O(n \log n)$ time using the algorithm from Corollary 1. Otherwise, find $I = GH(R) \cap GH(B)$ in $O(n \log n)$ time using a linesweep algorithm, and extend I as explained in Theorem 2. □

Theorem 3 *Given sets R of red points and B of blue points in a simple polygon P, deciding whether any geodesic (or any chord) in P separates r from B requires $\Omega(n \log n)$ time in the algebraic computation tree model.*

Proof: We prove the lower bound by describing a linear-time reduction from the integer set intersection problem: Given two sets X and Y of integers, determine whether any integer lies in both sets.[1] Yao [18] proved that solving this problem requires $\Omega(n \log n)$ time in the algebraic computation tree model; the lower bound applies even if one of the sets is given in sorted order. Let X be a set of n integers, and let Y be a sorted sequence of n integers. We construct a simple polygon P with $O(n)$ edges as follows. The polygon is a rectangle centered along the x-axis, with a thin crack of width $1/8$, mostly along the x-axis. For every integer $y \in Y$, the crack has a square bump of width $1/2$ and height 1 centered at the point $(y, 1/2)$. Next, we transform Y into a set of n blue points $\{(y, 1/3) \mid y \in Y\}$. Finally, we transform X into a set of $n + 4$ red points; a point at $(x, 2/3)$ for each $x \in X$, plus two additional red points near the bottom corners of the large rectangle. The reduction can be performed in linear time within in the algebraic computation tree model. If X and Y are disjoint, then all the non-corner red points are above the crack. In this case, the red and blue points can be separated by a geodesic. In fact, by making a few small adjustments to the ends of the crack, we can guarantee that there actually is a separating chord; see Figure 2.

On the other hand, if X and Y are not disjoint, then one of the bumps in the crack has both a red point r and a blue point b immediately below it. Any geodesic that separates these two points must pass below r and above b; however, every separating geodesic is above both of the bottom corner red points. It follows that the red and blue points cannot be separated by any geodesic in P. □

[1] We can avoid the restriction to integer sets by replacing the small fractions in our construction with formal infinitesimals; however, this change would limit our lower bound to algebraic *decision* trees.

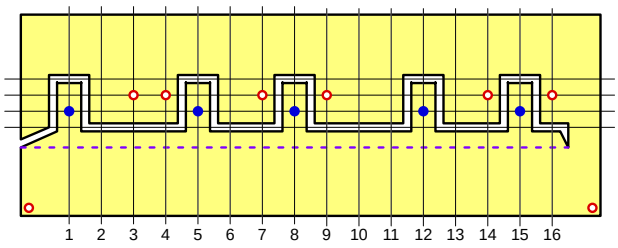

Figure 2: The result of our reduction from $X = \{3, 4, 7, 9, 14, 16\}$ and $Y = \langle 1, 5, 8, 12, 15 \rangle$.

3 Separability by non-crossing chords

We consider next a natural generalization of one of the problems studied in Section 2: to separate the two point sets using as few non-crossing chords as possible. If crossings were allowed and the points were placed closely together, the solution would consist of a minimum set of *lines* separating the sets in the plane, and finding such a set is known to be NP-hard [7].

We show that the problem is polynomially solvable when P is very simple, namely a pair of parallel lines or a triangle, and becomes NP-complete when P may have holes. In between, there is an intriguing open problem on which we comment at the end of the paper.

3.1 Separating points inside a strip

Let R and B be sets of red and blue points in a vertical strip. Theorem 1 implies that if R and B are separable by a chord, then they have disjoint convex hulls. In this case, R and B can be weakly separated by a chord that passes through one red point and one blue point, and this canonical separating chord can be found in $O(n)$ time using linear programming. In the more general case where more than one chord is required to

separate the red and blue points, we define a canonical set of separating chords as follows. Say that a chord is *pinned* if it passes through a point in $R \cup B$ and *trapped* if it passes through a point in R and a point in B. A *fan* is a set of chords with a common endpoint, called its *apex*. A *canonical fan* is a fan where every chord is pinned and at least one chord is trapped. Finally, a set of chords that weakly separate $R \cup B$ is *canonical* if it consists of a sequence of canonical fans whose apexes lie on alternate sides of the strip. See Figure 3.

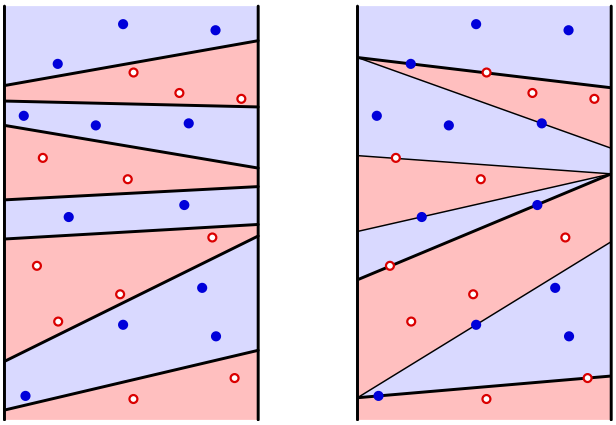

Figure 3: Left: red and blue points in a strip, separated by non-crossing chords. Right: a canonical weak separation into the same red and blue subsets; thicker chords are trapped.

Lemma 2 *Let R and B be sets of red and blue points in a strip. For any set of non-crossing chords that weakly separate R and B, there is a canonical set of non-crossing chords that weakly separates R and B into the same subsets.*

Proof: We describe an algorithm to canonicalize any weakly-separating set C of non-crossing chords. The algorithm proceeds in two phases. In the first phase, we move each chord in turn, from lowest to highest. Each chord is moved downward as far as possible until it touches either a point in $R \cup B$ or an endpoint of the next lower chord. In the latter case, we rotate the chord around the common endpoint until it touches a point in $R \cup B$. At the end of this phase, every chord is pinned; we call the point in $R \cup B$ on any chord the *pivot*. In the second phase, the algorithm maintains an *active fan* of chords with a common endpoint on one side of the strip. The chords below the active fan (if any) belong to an alternating sequence of canonical fans; the apex of the active fan (if any) lies on the opposite side of the strip from the highest canonical fan. Initially, the active fan consists of just the lowest chord; we can choose either endpoint as the apex. We lift the apex of the active fan, maintaining contact between each chord in the fan and its pivot point, until one of the following events occurs:

1. The top chord in the active fan touches an endpoint of the next higher chord. In this case, we add the next higher chord to the active fan and continue.

2. The bottom chord in the active fan touches the apex of the next lower canonical fan. In this case, we *freeze* the lowest chord, removing it from the active fan and adding it to the next lower fan. If the active fan is now empty, we use the next higher chord as the new active fan.

3. A chord in the active fan touches a second point in $R \cup B$ with the same color as its pivot. In this case, we consider the new point to be the pivot of that chord and continue.

4. A chord in the active fan touches a second point in $R \cup B$ with the opposite color from its pivot. (This includes the case where two chords in the fan coincide.) In this case, that chord is now trapped. We freeze the active fan, and the next higher chord (if any) becomes the new active fan, with its apex on the opposite side of the strip from the old active fan's apex.

The process ends when the topmost chord is frozen, at which point the entire set of chords is canonical. □

Theorem 4 *A minimal set of non-crossing chords that weakly separate a set of red points*

from a set of blue points in an infinite strip can be computed in $O(n^5)$ time.

Proof: The previous lemma implies that it is sufficient to search for a minimal *canonical* set of non-crossing chords. We compute such a set by considering all possible sequences of non-crossing *trapped* chords, using a straightforward dynamic programming algorithm. As we show below, for each such sequence, the minimum number of additional non-trapped chords that must be added to weakly separate the red and blue points can be computed in linear time, after a global preprocessing stage. For any two non-crossing trapped chords t^- and t^+, where t^- is above t^+, let $T(t^-, t^+, \mathsf{left})$ denote the minimum number of non-crossing chords that weakly separate the red and blue points between t^- and t^+, where (1) every chord shares either the left endpoint of t^- or the non-left endpoint of t^+; (2) any points between t^- and the next higher chord are the same color as the left point on t^-; and (3) any points between t^+ and the next lower chord are the same color as the left point on t^+. We define $T(t^-, t^+, \mathsf{right})$ analogously. See Figure 4. We can easily compute $T(t^-, t^+, \mathsf{left})$ by drawing a chord through every point in the trapezoid, either from the bottom left corner or from the top non-left corner—only one of these two chords lies entirely within the strip—and then discarding any chord that passes through the same color point as the chord above it, or the *left* point of the top chord. With no preprocessing, this computation requires $O(n \log n)$ time to sort points angularly around the opposite corners of the trapezoid, plus $O(n)$ time to scan through the sorted list. We can speed this up by computing the arrangement of lines dual to $R \cup B$ in a $O(n^2)$-time preprocessing phase. The angular order of $R \cup B$ around any point p is identical to the order in which the lines dual to $R \cup B$ intersect the line p^* dual to p; the Zone Theorem implies that we can compute this order in $O(n)$ time by simply walking around the boundary of the zone of p^*. A similar algorithm computes $T(t^-, t^+, \mathsf{right})$ in linear time. Now for any trapped chord t, let $C(t, \mathsf{left})$ denote the size of the minimum canonical set of chords that weakly separate the red and blue points on or above t, where apex of the lowest fan is the left endpoint of t, and the points between t and the next higher chord have the same color as the left point on t. We define $C(t, \mathsf{right})$ analogously. Clearly, $C(t, \mathsf{left}) = 0$ if every point above t has the same color as the left point on t (in particular, if there are no points above t). Otherwise, we have the recurrence

$$C(t, \mathsf{left}) = 1 + \min_{t'} \big(T(t, t', \mathsf{left}) + C(t', \mathsf{right})\big)$$

where t' ranges over all trapped chords that lie entirely above t. For each trapped chord t, the function $C(t, \mathsf{left})$ depends on $O(n^2)$ other trapped chords t', and each $T(t, t', \mathsf{left})$ can be evaluated in time $O(n)$. Thus, not counting recursion, we can compute $C(t, \mathsf{left})$ in $O(n^3)$ time. Since there are $O(n^2)$ trapped chords t, we can compute $C(t, \mathsf{left})$—and analogously, $C(t, \mathsf{right})$—for *all* t in $O(n^5)$ time by straightforward dynamic programming. Finally, the minimum number of non-crossing chords that separate R from B is the smaller of $C(-\infty, \mathsf{left})$ and $C(-\infty, \mathsf{right})$, where $-\infty$ denotes a symbolic chord infinitely far below all of the points. □

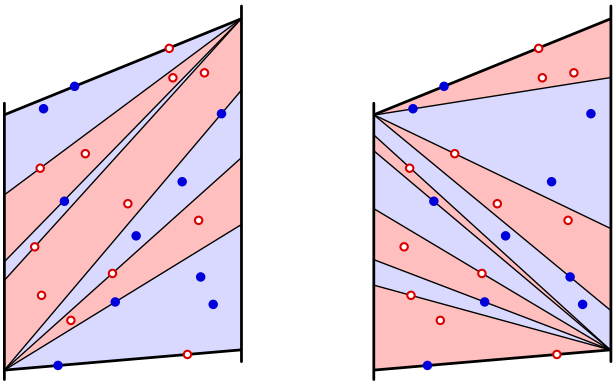

Figure 4: $T(t^-, t^+, \mathsf{left}) = 6$ and $T(t^-, t^+, \mathsf{right}) = 8$.

Our algorithm requires one slight modification if we desire a minimal set of chords that *strictly* separate the red and blue points, where no point in $R \cup B$ lies on a chord. Instead of using the points themselves to define canonical chord sets, we replace each point p with two perturbed

points $p^\flat = p - (0, \varepsilon)$ and $p^\sharp = p + (0, \varepsilon)$, where ε is a symbolic infinitesimal. Now a pinned chord passes through at least one perturbed point $p^\sharp$ and a trapped chord passes through two perturbed points $p^\flat$ and $q^\sharp$ of different colors. The remainder of the algorithm is unchanged.

3.2 Separating points in a triangle

Now suppose the points lie inside a triangle. If the optimal set of separating chords has a simple linear structure, then a straightforward generalization of our strip algorithm can find it in $O(n^5)$ time—we simply treat two edges of the triangle as one side of the "strip", with the third edge forming the other side. However, the optimal separating set could have a tree-like structure instead, with a single central region bounded by three chords and three (possibly empty) subsets of triangle edges. In this case, more effort is required, in part because we cannot assume that any of these three chords passes through a point of each color. Figure 5 shows a set of red and blue points separated by three non-crossing chords; if we require some chord to pass through both a red point and a blue point, then at least four chords are required. To find an optimal solution of this

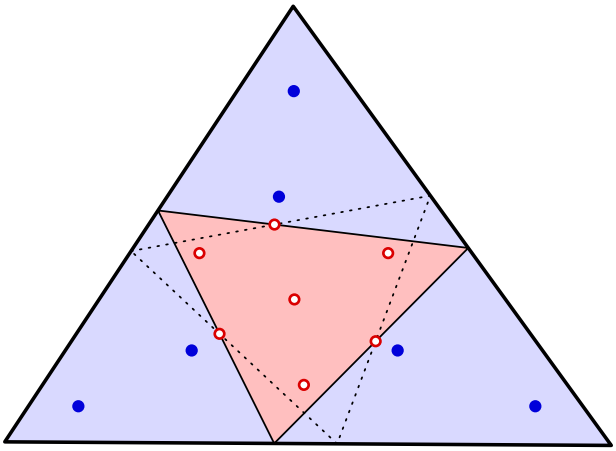

Figure 5: Separating points in a triangle. There is no separating set of three chords where one chord hits points of both colors.

form, we must modify our definition of "canonical" separating sets. We still require that the chords comprise three sequences of alternating fans, where each fan contains either a trapped chord or bounding chord of the central region. The central region is bounded by three chords, which can be either trapped or merely pinned. However, any pinned chord must share an endpoint with an adjacent chord, and two pinned chords can only share an endpoint if all three central chords are pinned and form a triangle, as in Figure 5.

Theorem 5 *A minimal set of non-crossing chords that weakly separate a set of red points from a set of blue points in a triangle can be computed in $O(n^6)$ time.*

Proof: The algorithm begins by computing the optimal strip-like solution in $O(n^5)$ time, and only then considers tree-like solutions. There are $O(n^3)$ pinned triangles. We can compute the optimal decomposition outside any pinned triangle in $O(n^3)$ time, by determining the trapped chord closest to each pinned triangle edge; the best decomposition beyond that trapped chord was already computed during the strip-like phase of the algorithm. To handle the case where the central region has a trapped bounding chord, we introduce a pair of *ghost* chords. These ghost chords form a triangle with the trapped chord, and exactly one of the ghost chords passes through an input point. There are $O(n^3)$ ghost chords, and we can compute the optimal decomposition outside each ghost chord in $O(n^3)$ time, exactly as we did for trapped triangle edges. (The ghost chords do not actually contribute to the cost of the solution.) Finally, for any trapped chord, we can find the best pair of ghost chords in $O(n)$ time. □

For any constant t, a similar dynamic programming algorithm can be used to separate red and blue points in any simple t-gon, or any polygon with holes with a total of t edges, in time $n^{O(t)}$. As t increases, the algorithm considers chords determined by larger subsets of input points. Since the algorithm is inefficient even for very small value of t, we omit further details.

3.3 Polygons with holes

Theorem 6 *Finding the minimal number of non-intersecting chords that separate blue from red points in a polygon with holes is NP-hard.*

Proof: Let $Exp(x)$ be a boolean expression in conjunctive normal form with n variables and m clauses such that each clause has three literals. Let G_{Exp} be the graph (V, E) where V consists of the variables and clauses of $Exp(x)$, and $(x_i, c_j) \in E$ if and only if variable x_i occurs in clause c_j. If G_{Exp} is planar, then deciding whether there is a assignment of true and false values for x such that $Exp(x)$ is true is known as the *planar 3SAT problem*. If we also require that each clause has exactly one true literal, the problem is known as *planar 1-in-3SAT*. Laroche [10] proved that planar 1-in-3SAT is NP-complete. We prove our theorem by describing a polynomial-time reduction from this problem.

We first show how to create a polygon P and a set of blue and red points from a planar embedding of the graph G_{Exp}. Figure 6 shows the encoding of a variable. We imagine that the boundary of the variable gadget is oriented clockwise. The *inside* of the variable is the connected portion of the boundary that bounds a hole; the remainder of the boundary is the *outside*. There are exactly two minimal sets of chords that separate the blue and red points within each variable gadget; these correspond to assigning the values true and false to the variable. The true setting consists of chords that from the inside of the variable gadget go in clockwise direction across to the outside; in the false setting, the chords are oriented counterclockwise. Figure 7 shows two close ups of part of a variable, one set to true and one set to false. We assume that the true and false settings each consist of k chords. Notice that any other set of chords that separate the red and blue points in a variable gadget requires more than k chords.

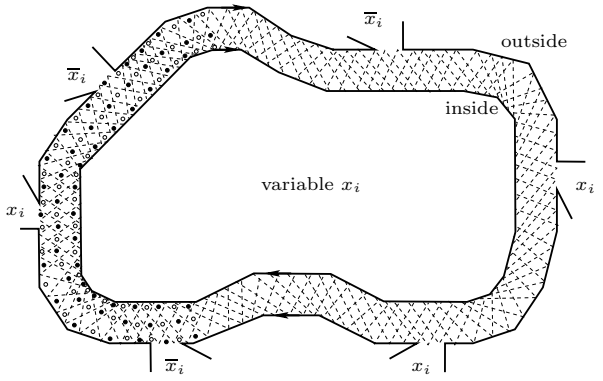

Figure 6: A variable contained in six clauses; black and white dots represent blue and red points respectively.

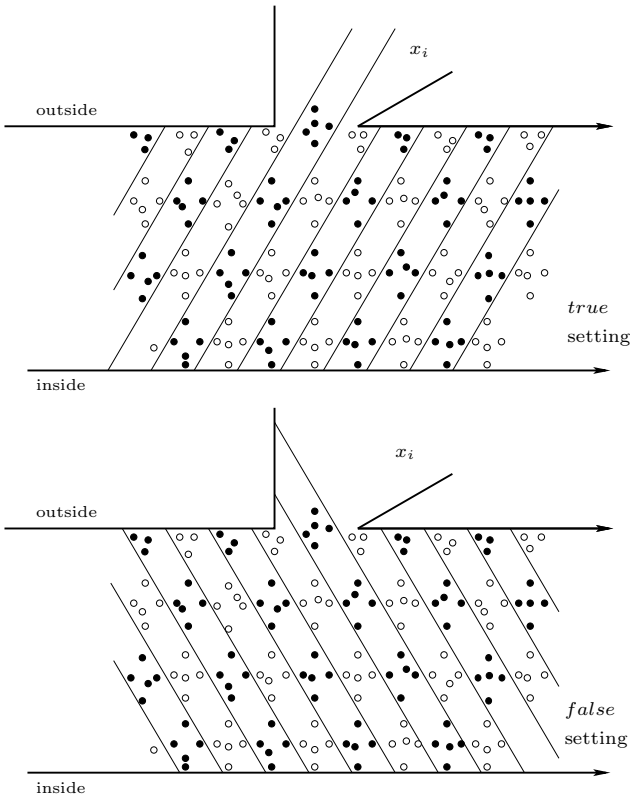

Figure 7: True and false settings of a variable.

Clauses are encoded as equilateral triangles that meet its three variables in the corners as shown in Figure 8. A bump is placed on each side of the triangle to prevent chords from intersecting more than one variable gadget. A variable gadget meets a clause gadget on its outside boundary. At the place where they touch, there is a little connection between the variable and the clause. If a clause contains a variable x_i, then

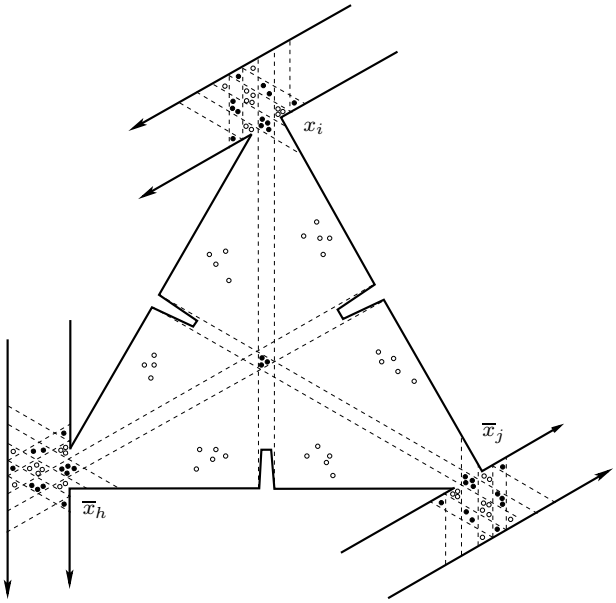

Figure 8: The clause $(\bar{x_h} \vee x_i \vee \bar{x_j})$.

the gadget for variable x_i approaches the triangle with an angle of 30°. If a clause contains $\bar{x}_i$, then the gadget for variable x_i approaches the triangle with an angle of 90°. Since G_{Exp} is planar, the variable and clause gadgets can be connected to form a polygon P with holes.

Notice that if the gadget of x_i is set to true in a clause containing x_i, or if the gadget of x_i is set to false in a clause containing $\bar{x}_i$, then two parallel chords from x_i's gadget cross the triangle. These two parallel chords separate the three blue points in the center of triangle from the remaining red points in the triangle.

We now show that $Exp(x)$ can be satisfied with exactly one true literal per clause if and only if the blue and red points in P can be separated by exactly kn non-intersecting chords.

First, suppose $Exp(x)$ can be satisfied with exactly one true literal per clause. We can separate the blue from the red points in each variable x_i with k chords, depending on the truth value of x_i. Since each clause has exactly one true literal, only the corresponding variable has two parallel chords that pass through through the triangle. So we have separated all blue from all red points in the polygon using kn chords.

On the other hand, suppose kn chords suffice the separate the blue and red points. Each variable requires at least k separating chords, and the shape of the clause gadget impose that chords intersect no more than one variable gadget. This implies there are exactly k chords per variable. Therefore each variable is set to true or false. The blue points at the center of each clause gadget are separated from the red points in that gadget, which implies that each clause contains exactly one true literal. □

We conclude this section remarking that between the results described in Theorems 4, 5 and 6 there is a gap raising an intriguing question:

Open Problem 1 *What is the complexity of finding a minimal set of non-crossing chords that weakly separate two point sets contained in the interior of*

(a) a convex k-gon? (with k as part of the input)

(b) a disk?

(c) a simple polygon?

Acknowledgement

This research was initiated at the McGill Workshop on Instance-Based Learning at Belairs Marine Biology Institute, Jan. 31–Feb.6, 2003. The authors would like to thank the workshop organizer Godfried Toussaint and the other workshop participants, namely, Greg Aloupis, Prosenjit Bose, David Bremner, Vida Dujmovic, Danny Krizanc, Pat Morin, Tom Shermer, and David Wood for helpful discussions and for providing a stimulating working environment.

References

[1] B. Aronov. On the geodesic Voronoi diagram of point sites in a simple polygon. *Algorithmica*, 4:109–140, 1989.

[2] B. Aronov, S. J. Fortune, and G. Wilfong. Furthest-site geodesic Voronoi diagram. *Discrete & Computational Geometry*, 9:217–255, 1993.

[3] B. K. Bhathacharya. Circular separability of planar point sets. In G. T. Toussaint, editor, *Computational Morphology*. North Holland, Amsterdam, 1988.

[4] J.-D. Boissonnat, J. Czyzowicz, O. Devillers, J. Urrutia, and M. Yvinec. Computing largest circles separating two sets of segments. *International Journal of Computational Geometry and Applications*, 10(1):41–53, 2000.

[5] P. Bose, E. D. Demaine, F. Hurtado, J. Iacono, S. Langerman, and P. Morin. Ham-sandwich cuts in polygonal environments. Manuscript, 2003.

[6] H. Edelsbrunner and F. P. Preparata. Minimum polygonal separation. *Information and Computation*, 77:218–232, 1988.

[7] R. W. Freimer. *Investigations in Geometric Subdivisions: Linear Shattering and Cartographic Map Coloring*. PhD thesis, Cornell Universit, 2000.

[8] L. J. Guibas and J. Hershberger. Optimal shortest path queries in a simple polygon. In *Proceedings of the third annual symposium on Computational geometry*, pages 50–63. ACM Press, 1987.

[9] F. Hurtado, M. Noy, P. A. Ramos, and C. Seara. Separating objects in the plane with wedges and strips. *Discrete Applied Mathematics*, 109:109–138, 2001.

[10] P. Laroche. Satisfiabilite de 1-parmi-3 planaire est np-complet. *C.R. Acad. Sci. Paris*, pages 389–392, 1993.

[11] N. Megiddo. Linear-time algorithms for linear programming in r^3 and related problems. *SIAM Journal on Computing*, 12(4):759–776, 1983.

[12] E. Papadopoulou and D. T. Lee. A new approach for the geodesic voronoi diagram of points in a simple polygon and other restricted polygonal domains. *Algorithmica*, 20(4):319–352, 1998.

[13] R. Pollack, M. Sharir, and G. Rote. Computing the geodesic center of a simple polygon. *Discrete & Computational Geometry*, 4:611–626, 1989.

[14] C. Seara. *On Geometric Separability*. PhD thesis, Univ. Politècnica de Catalunya, June 2002.

[15] S. Suri. Computing geodesic furthest neighbors in simple polygons. *Journal of Computer and System Sciences*, 39:220–235, 1989.

[16] G. Toussaint T. Asano. Computing the geodesic center of a simple polygon. In *Discrete Algorithms and Complexity, Perspectives in Computing*, pages 65–79. Academic Press, 1987.

[17] G. T. Toussaint. An optimal algorithm for computing the relative convex hull of a set of points in a polygon. In *Signal Processing III: Theories and Applications, Proc. EURASIP-86, Part 2*, pages 853–856, September 1986.

[18] A. C. Yao. Lower bounds for algebraic computation trees with integer inputs. *SIAM Journal on Computing*, 20(4):655–668, 1991.