SIAM J. COMPUT.
Vol. 15, No. 1, February 1986

CONSTRUCTING BELTS IN TWO-DIMENSIONAL ARRANGEMENTS WITH APPLICATIONS*

H. EDELSBRUNNER† AND E. WELZL†

Abstract. For H a set of lines in the Euclidean plane, $A(H)$ denotes the induced dissection, called the arrangement of H. We define the notion of a belt in $A(H)$, which is bounded by a subset of the edges in $A(H)$, and describe two algorithms for constructing belts. All this is motivated by applications to a host of seemingly unrelated problems including a type of range search and finding the minimum area triangle with the vertices taken from some finite set of points.

Key words. computational geometry, arrangements of lines, plane-sweep, maintenance of convex hulls, efficient algorithms, halfplanar range search

1. Introduction. The systematic study of algorithms for low-dimensional geometric problems started with the doctoral dissertation of Shamos [S] around 1975. Since then, this area of research experienced a steady increase in activity which can be explained by the host of theoretically interesting problems in the field and also by the relevance of the developed results to more practically oriented branches in computer science.

This paper is concerned with several seemingly unrelated problems dealing with finite sets of points in the Euclidean plane. The problems are discussed in §§ 4.1 through 4.5. All solutions rely heavily on the more basic developments of §§ 2 and 3. Section 2 introduces the concept of a dual plane which hosts a line for each point in the original space. Such a set of lines cuts the (dual) plane into convex regions, edges, and vertices which express certain convexity properties of the (original) point-set fairly explicitly. The set of lines also turns out to favour the development of algorithmic solutions for the (original) problems, once the correspondences between both settings are reasonably understood. Section 3 presents two algorithms for constructing belts, a fundamental concept defined for sets of lines in § 2.

We explicate one problem to illustrate the general character of our results: Let S be a set of n points in the plane. The *halfplanar range search problem* requires a data structure for S such that the number of points which lie in a later specified query halfplane can be determined efficiently. All known solutions either take a lot of space ($O(n^2)$ space suffices to achieve $O(\log n)$ time for answering a query [EKM]) or suboptimal time (with $O(n)$ space, the currently best solutions answers a query in $O(n^{0.695})$ time [EW2]). Recent arguments of Fredman [F] also support the thesis that halfplanar range search is inherently more complex than classical orthogonal range search (see Bentley and Friedman [BF] for an early survey of solutions for the latter). Motivated by these results, we relax the problem and ask for rough ideas of the number of points in a query halfplane rather than for the exact answer. Sections 4.1 and 4.2 offer families of solutions which realize varying degrees of accuracy. Most striking, Theorem 4.6 shows the existence of a constant space structure which discriminates between halfplanes that contain less than one third and more than two thirds of all points.

2. Geometric preliminaries. This section introduces the geometric concepts and facts which are needed in the forthcoming discussions. The primary concern are so-called k-sets of planar point-sets and their appearance under a dualizing geometric transform.

* Received by the editors October 4, 1982, and in revised form October 29, 1984.

† Institutes for Information Processing, Technical University of Graz, A-8010 Graz, Austria.

Let S denote a set of n points in the plane, for some positive integer n. We call a subset S' of S a *k-set of S*, for $0 \leqq k \leqq n$, if it contains k points and there exists a halfplane which intersects S in S'. Let $f_k(S)$ denote the number of k-sets realized by S and let $f_k(n)$ denote the maximum of $f_k(S)$, for all sets S of n points in the plane. Then $f_0(n) = f_n(n) = 1$ and obviously $f_k(n) = f_{n-k}(n)$, for $0 \leqq k \leqq n$. Bounds on the asymptotic behaviour of $f_k(n)$, for $1 \leqq k \leqq n-1$, are developed in Erdös, Lovasz, Simmons, and Straus [ELSS] and in Edelsbrunner and Welzl [EW1]:

PROPOSITION 2.1. *Let k and n denote two positive integers with $k \leqq \lfloor n/2 \rfloor$. Then $f_k(n)$ and $f_{n-k}(n)$ are in $\Omega(n \log(k+1))$ and in $O(nk^{1/2})$.*

It is often the case that geometric problems formulated for point-sets are more conveniently solved in dual space, which can, e.g., be obtained by application of the following geometric transform T which was also used in Brown [Br] to solve geometric problems different from ours.

(1) A point $p = (p_x, p_y)$ is mapped into the line $T(p)$ whose points (x, y) satisfy $y = p_x x + p_y$, and

(2) a nonvertical line L whose points (x, y) satisfy $y = L_x x + L_y$ is mapped into the point $T(L) = (-L_x, L_y)$ (see Fig. 2-1).

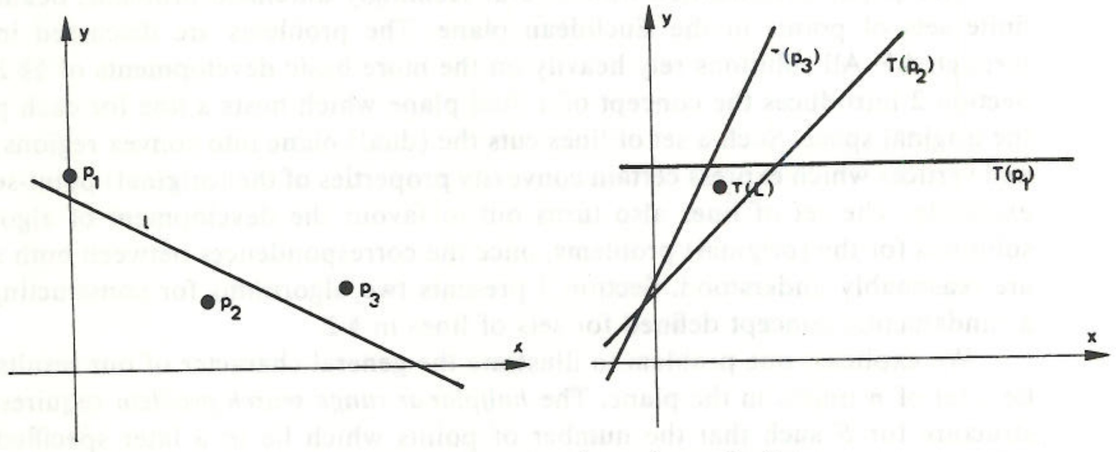

FIG. 2.1. *T applied to three points and a line.*

A set S of points is transformed into a set $T(S)$ of nonvertical lines. The nice property of T is the maintenance of the relative position between a point and a line. Let $p = (p_x, p_y)$ be a point and L: $y = L_x x + L_y$ a nonvertical line. Let $y_0 = L_x p_x + L_y$. Then we say that *p lies below*, *on*, or *above* L depending on whether $p_y < y_0$, $p_y = y_0$, or $p_y > y_0$. We also say that *L lies above p*, *contains p*, or *lies below p* in these cases.

Observation 2.2. Let p denote a point and L a nonvertical line in the plane. Then p lies below (or above) L if and only if $T(p)$ lies below (or above) $T(L)$.

This effect can be observed in Fig. 2.1 where p_2 is the only point below L and $T(p_2)$ is the only line below $T(L)$.

The set $H = T(S)$ of lines induces a dissection of the plane called the *arrangement $A(H)$ of H*. $A(H)$ consists of *vertices* (intersections of lines), *edges* (maximal connected subsets of the lines which contain no vertex), and *regions* (maximal connected subsets of the plane which contain no edge or vertex). For convenience, we say that two parallel lines intersect in a vertex at infinity. Also for convenience, we define the notions of complete edges and regions: A *complete edge of $A(H)$* is a bounded edge, an unbounded edge on a line that has a parallel line in H, or the union of two unbounded edges both supported by the same line which has no parallel line in H. A *complete* *region of* $A(H)$ is a bounded region, an unbounded region enclosed between two parallel lines, or the union of two unbounded regions R_1 and R_2 such that any line in H separates R_1 from R_2. Notice the similarity of these notions to the concepts of edges and regions in the projective plane. There exists an important correspondence between the k-sets of S and the complete regions of $A(H)$:

Observation 2.3. Let S' be some k-set of S. Then there exists a complete region R in $A(H)$ such that a line L separates S' from $S-S'$ if and only if $T(L)$ lies in R.

For each point p in the plane let $b(p)$, $o(p)$, and $a(p)$ denote the number of lines in H which lie below p, contain p, and lie above p, respectively. Evidently, $b(p)+o(p)+a(p)=n$ (n the cardinality of H), for each point p. We define the *k-belt of H*, for $0 \leqq k \leqq \lceil n/2 \rceil$, as the set of points p in the plane such that $b(p)+o(p) \geqq k$ and $a(p)+o(p) \geqq k$ (see Fig. 2.2). The 0-belt is the whole plane, and the k-belt, for $k \geqq 1$, is bounded below and above by an unbounded polygonal chain, respectively. These polygonal chains are monotone in x, that is, any one of them intersects each vertical line in exactly one point. For $k=(n+1)/2$, the two boundaries of the k-belt coincide. Obviously, the k'-belt is contained in the k''-belt if $0 \leqq k'' < k' \leqq \lceil n/2 \rceil$.

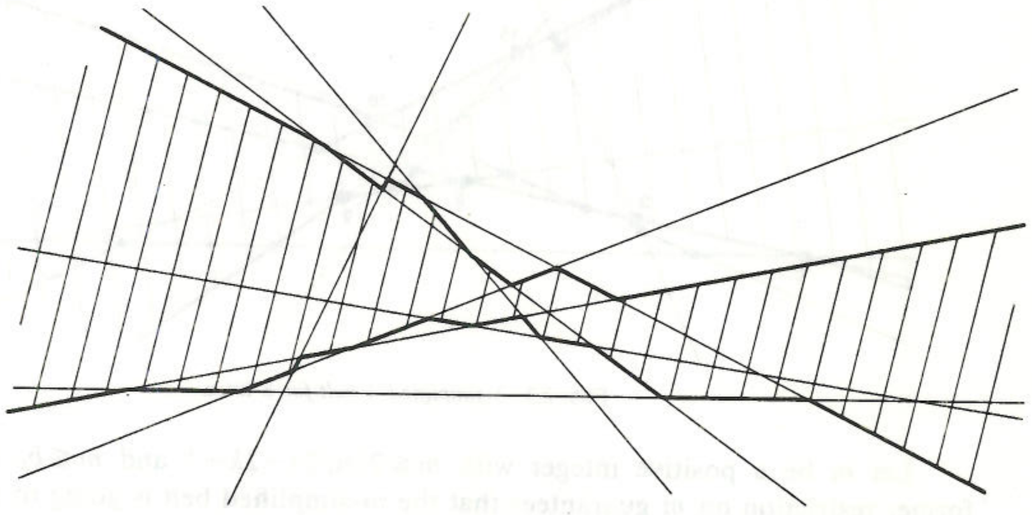

FIG. 2.2. *The 3-belt for 8 lines.*

Let $p=(p_x, p_y)$ be an arbitrary point in the plane and let B be the k-belt of H, for some integer k. Let y_b and y_a denote the y-coordinates of the intersections of the vertical line through p and the lower and upper boundary of B, respectively. We say that p lies *below*, *in*, or *above* B if $p_y < y_b$, $y_b \leqq p_y \leqq y_a$, or $y_a < p_y$ is true, respectively.

In the following sections, a k-belt will be represented by the sequences of edges of its boundaries. Let $b_k(H)$ denote the number of complete edges bounding the k-belt of H, and let $b_k(n)$ denote the maximum of $b_k(H)$, for all sets H of n lines in the plane. The strong relationship between the k-sets of S and the complete regions of $A(H)$ imply that $b_k(n)$ and $b_{n-k}(n)$ are in $\Theta(f_{k-1}(n)+f_k(n))$ and therefore in $\Omega(n \log (k+1))$ and in $O(nk^{1/2})$. The interested reader is invited to verify the following:

LEMMA 2.4. *Let S denote a set of n points in the plane with no three collinear, and define* $H=T(S)$. *Then*

$$b_0(H)=0,$$

$$b_1(H)=f_1(S),$$

$$b_k(H)=f_{k-1}(S)+f_k(S), \quad \textit{for } 2 \leqq k \leqq \lceil (n-1)/2 \rceil.$$

In § 4 of this paper, so-called simplified belts of sets of lines are used to decrease the space requirements needed for solving halfplanar range estimation problems. These simplified belts are obtained from ordinary belts by replacing sequences of complete edges by single complete edges. To this end, let B denote the k-belt of a set H of $n \geqq 2$ lines and $1 \leqq k \leqq \lceil n/2 \rceil$. For convenience, we assume that no three lines of H are concurrent, that is, no three intersect in a common vertex. The unrestricted case will be discussed later. We call a vertex of the lower and upper boundary of B a *lower* and an *upper vertex*, respectively. Note that two parallel lines define a vertex at infinity which is defined a lower vertex. If a vertex belongs to both boundaries then it is considered as two vertices, one of each kind. We assign the numbers $0, 1, \cdots, b_k(H)-1$ to the $b_k(H)$ vertices of B such that $i<j$ if i is a lower and j an upper vertex, and vertex i is to left of the vertical line through vertex j, otherwise (see Fig. 2.3).

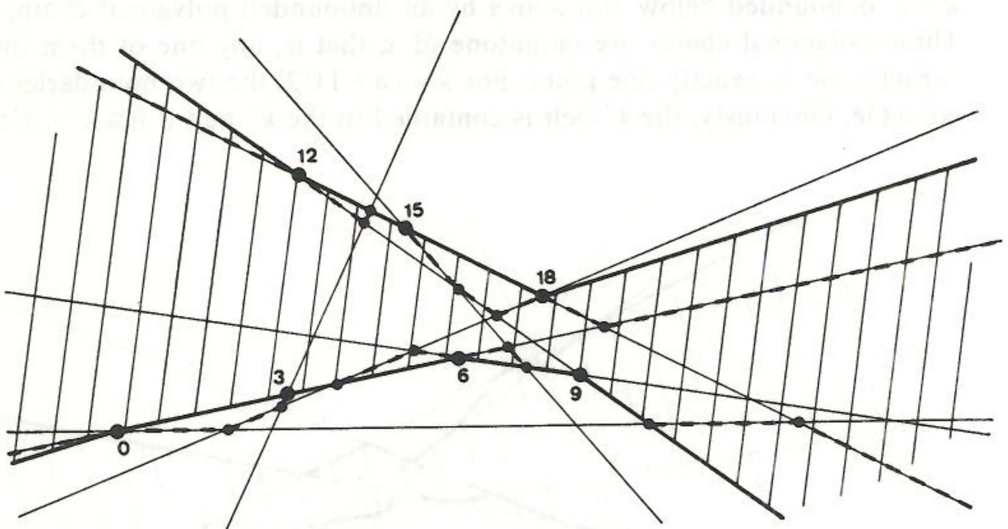

FIG. 2.3. *3-simplified 3-belt for 8 lines.*

Let m be a positive integer with $m \leqq 2\lceil n/2 \rceil - 2k+1$ and $m \leqq b_k(H)-1$. The former restriction on m guarantees that the m-simplified belt is going to be bounded by two noncrossing polygonal chains (see Lemma 2.6). The latter restriction implies that the m-simplified belt will avoid over-simplification. The *complete edges* of the boundaries of the m-simplified k-belt connect the vertices 0 and m, m and $2m, \cdots, rm$ and 0, with $r = \lfloor (b_k(H)-1)/m \rfloor$. A complete edge which connects two lower or two upper vertices is the line segment which has the two vertices as endpoints. A complete edge connecting a lower and an upper vertex consists of two rays on the line through the two vertices which emanate from the two vertices such that they do not overlap. The *m-simplified k-belt* is the set of points in between the lower and upper boundary including the boundary points. The reason for the introduction of simplified belts is the smaller number of vertices and edges they consist of as compared to ordinary belts.

Observation 2.5. Let S denote a set of n points in the plane such that no three are collinear. Then the m-simplified k-belt of $H = T(S)$, for $1 \leqq k \leqq \lceil n/2 \rceil$, $1 \leqslant m \leqslant 2\lceil n/2 \rceil - 2k+1$, and $m \leqq b_k(H)-1$, has at most $\lceil b_k(H)/m \rceil$ vertices.

The nice property of simplified k-belts is the fact that they are reasonable approximations of ordinary k-belts.

LEMMA 2.6. *Let S be a set of n points in the plane such that no three are collinear, and define $H = T(S)$. Then the m-simplified k-belt, for $1 \leqq m \leqq 2\lceil n/2 \rceil - 2k+1$ and $m \leqq b_k(H)-1$, contains the $(k+\lfloor m/2 \rfloor)$-belt and is contained in the* $\max\{0, k-\lfloor m/2 \rfloor\}$*-belt of H.*

Proof. Consider an arbitrary complete edge e of the m-simplified k-belt which we call M; see Fig. 2.4. Let L denote a line in H which properly intersects e, that is, it does not support e. It is readily seen that at least one complete edge of the k-belt of H which is supported by L is in the sequence of complete edges replaced by e (see Fig. 2-4). (Note that this is true only because no three lines intersect in a common point.) Consequently, at most $m-2$ lines of H properly intersect e.

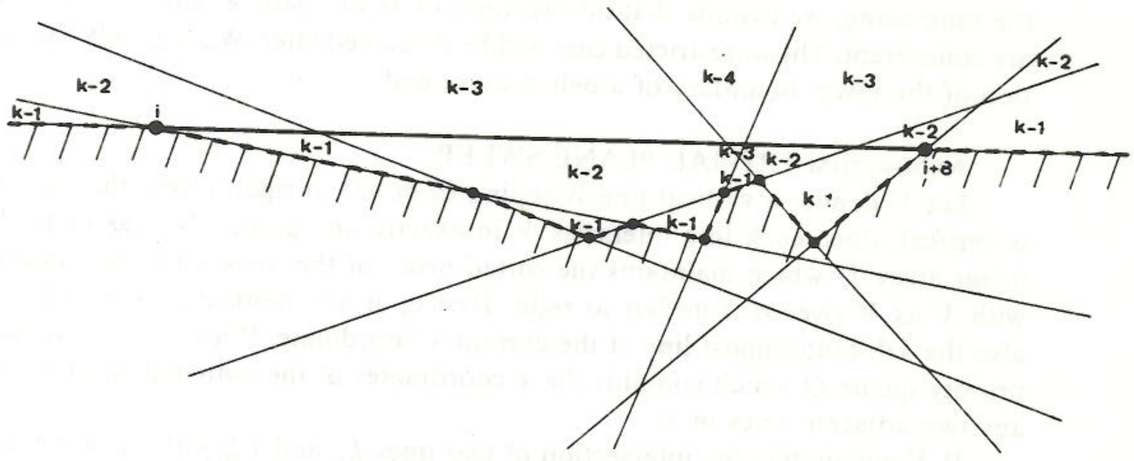

FIG. 2.4. *Simplification with $m = 8$.*

From the correspondence between the k-sets of S and the complete regions of $A(H)$ (see Observations 2.2 and 2.3) and the definition of k-belts we know that the regions which share an edge with the k-belt correspond to $(k-1)$-sets or k-sets. W.l.o.g. let the regions above the considered sequence of edges correspond to $(k-1)$-sets. Those regions above the latter regions which share edges with them correspond to $(k-2)$-sets, etc., etc. Let k' be minimal such that there is a complete region R which corresponds to a k'-set and R intersects e or lies between e and the sequence of edges replaced by e. Since e properly intersects only $m-2$ lines, we have $k' \geqq k - \lfloor m/2 \rfloor$. (The details of the verification of this fact are left to the interested reader.) This implies the second part of Lemma 2.6 while the analogous reasoning implies the first part. This completes the argument.

As an immediate consequence of Lemma 2.6, the m-simplified k-belt intersects each vertical line in a single and nonempty interval provided $k + \lfloor m/2 \rfloor \leqq \lceil n/2 \rceil$ which is equivalent to $m \leqq 2\lceil n/2 \rceil - 2k + 1$. Thus, we can extend the below-in-above relation defined for points and ordinary belts to points and simplified belts without ambiguity.

Let us now return to the general case of a point-set S which may contain an arbitrary number of points on a single line. Difficulties arise when a belt of $H = T(S)$ is simplified which contains a vertex common to more than two lines. If the simplification is carried out as described then Lemma 2.6 need not be true. This can be remedied by taking care that a new complete edge properly intersects at most $m-2$ lines of H. A simple way to achieve this is to consider a vertex v which is common to $i \geqq 2$ lines as a sequence of $\min\{m-1, i-2\}$ degenerate edges and one more vertex. This invalidates the definition given for simplified belts, and Lemma 2.6 is true again. Observation 2.5 is no longer true in the strength it is stated. Nevertheless we have:

Observation 2.7. Let S denote an arbitrary set of n points in the plane. Then the m-simplified k-belt of $H = T(S)$, with $1 \leqq k \leqq \lceil n/2 \rceil$ and $1 \leqq m \leqq 2\lceil n/2 \rceil - 2k + 1$, contains at most $\lceil b_k(n)/m \rceil$ complete edges.

Although there may be new complete edges which replace less than m old ones, the number is compensated by the loss of edges caused by the concurrency of lines.

3. Constructing belts. Let S be a set of n points in the plane. Throughout this section, we will assume that n and k are positive integers such that $k \leqq \lceil n/2 \rceil$. We will introduce two algorithms for constructing a belt of $H = T(S)$ which is represented by two linear lists storing the lower and upper vertices ordered from left to right, respectively. While the first algorithm is reasonably simple, the second is reasonably efficient. In fact, no better algorithm is currently known except for the special case $k = 1$. For the time being, we assume that no two lines of H are parallel and that no three lines are concurrent. The unrestricted case will be discussed later. W.l.o.g. only the construction of the lower boundary of a belt is described.

ALGORITHM TRIVIAL PLANE SWEEP:

Let V denote a vertical line sweeping from left to right. (Note that no line in H is vertical, thus, each line intersects V in exactly one point.) V is associated with a linear array D which maintains the sorted order of the lines w.r.t. their intersections with V as V sweeps from left to right. Hence, at any moment, the kth line in D is also the kth bottommost line at the current x-coordinate. With V we associate also a priority queue Q which contains the x-coordinates of the anticipated intersections of any two adjacent lines in D.

If V encounters the intersection of two lines L_b and L_a, with L_b below L_a to the left of the intersection, then the following actions are taken: (1) Delete the topmost element from Q, that is, the x-coordinate of the intersection. (2) Let L_{bb} and L_{aa} denote the lines immediately below L_b and above L_a, respectively. If L_b and L_{bb}, or L_a and L_{aa} intersect to the right of V then delete the x-coordinate of the respective intersection from Q. (3) Reverse the order of L_b and L_a in D. (4) If L_b and L_{aa}, or L_a and L_{bb} intersect to the right of V, then insert the x-coordinates of those intersections into Q. (5) If L_b or L_a is the kth line before the intersection is encountered, then the intersection point completes an edge of the k-belt which is reflected in the structure of the k-belt.

Trivial modifications enable the algorithm to handle degeneracies like parallel and concurrent lines. In the former case, intersection points in infinity are created. In the latter case, the order of all lines which meet in a vertex encountered by V is reversed in D.

LEMMA 3.1. *Let H be a set of n nonvertical lines in the plane. Algorithm* TRIVIAL PLANE SWEEP *constructs the k-belt of H in $O(n^2 \log n)$ time and $O(n + b_k(H))$ space.*

Proof. There are at most $\binom{n}{2}$ intersections to be handled with the data structures D and Q. Each intersection of two lines requires $O(\log n)$ time, see Aho, Hopcroft, and Ullman [AHU]. Common intersections of $i > 2$ lines require $O(i \log n)$ time which is less than the amount that would be needed if all pairs of the i lines intersect in unique points. (Deletions from Q can be handled by maintaining a pointer from each pair of adjacent lines in D for which an intersection is anticipated to the x-coordinate of this intersection in Q.) The space required by D and Q is $O(n)$ and the one for the k-belt is $O(b_k(H))$. This completes the argument.

It is worth noting that Algorithm TRIVIAL PLANE SWEEP can construct all belts of H within the same asymptotic time bounds. The algorithm is also an interesting general method for certain types of point problems if it neglects the construction of belts and performs other actions instead. In fact, § 4.5 demonstrates one such example.

For computing a single belt, a considerably more efficient method is obtained by refining Algorithm TRIVIAL PLANE SWEEP: Instead of considering all $\binom{n}{2}$ intersections, only those coinciding with vertices of the belt are processed. This is made possible by maintaining the lines below and above the kth line in separate data structures of a particular kind. For this algorithm we need the following result due to Overmars and van Leeuwen [OvL1]:

PROPOSITION 3.2. *The intersection of a set S of n halfplanes can be computed in $O(n \log n)$ time and $O(n)$ space. This intersection can then be maintained with $O(\log^2 n)$ time per insertion and deletion such that the adjacent edges of a given edge are available in constant time. All bounds state the requirements in the worst case and n denotes the current number of halfplanes stored.*

Again only the construction of the lower boundary of the belt is described, and for the time being, we assume that no two lines are parallel and no three are concurrent.

ALGORITHM SOPHISTICATED PLANE SWEEP:

Let V denote a vertical line sweeping from left to right. At some moment in time let L in H cause the kth bottommost intersection V_L with V. The lines below V_L are interpreted as lower boundaries of halfplanes which are stored in a data structure H_B. The lines above V_L are interpreted as upper boundaries of halfplanes which are stored in a similar data structure H_A. H_B and H_A are instances of the data structure referred to in Proposition 3.2.

The next intersection of L to the right of V is determined as follows: (1) The halfplane bounded below by L is inserted into H_B. The adjacent edge in counterclockwise order of the edge caused by this halfplane (if it exists) gives the leftmost intersection to the right of V of L with a line L_B below V_L (see Fig. 3.1, where no such edge exists). (2) The halfplane bounded above by L is inserted into H_A. The adjacent edge in clockwise order of the edge caused by this halfplane (if it exists) gives the leftmost intersection to the right of V of L with a line L_A above V_L (see Fig. 3.1). (3) W.l.o.g. let L_A intersect L above L_B (as illustrated in Fig. 3.1). Then L is deleted from H_B and L_A is deleted from H_A. (If neither L_A nor L_B exists, then L supports the rightmost and unbounded edge of the lower boundary.) In addition, the new edge of the k-belt created is reflected in the data structure of the k-belt, and L_A is the new kth line.

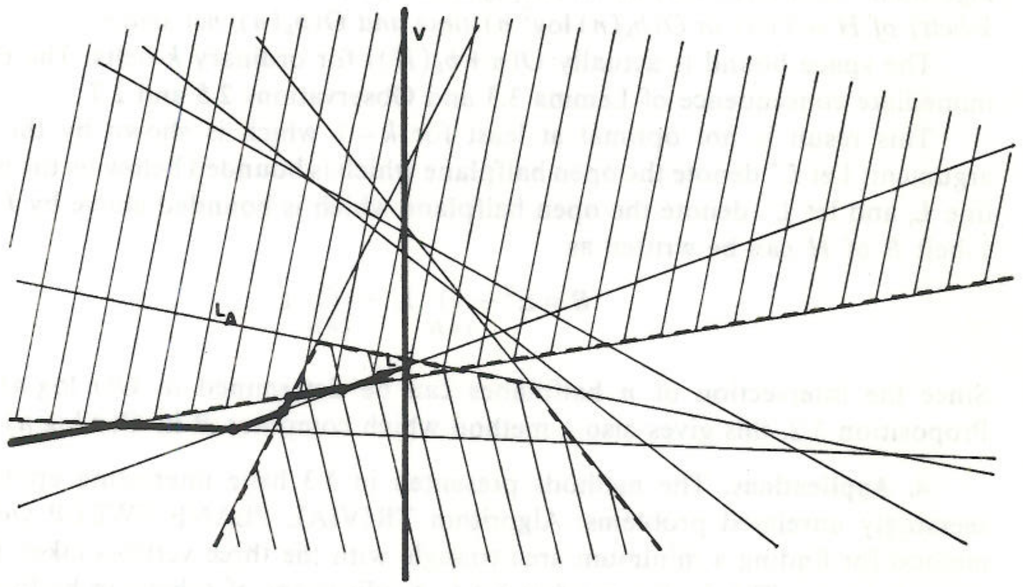

FIG. 3.1. *Constructing the 3-belt of 8 lines.*

A short moment of reflection shows that for the correctness of the algorithm it must be guaranteed that L contributes an edge to the intersection of halfplanes in H_B and H_A, respectively. In addition, this edge must intersect V at its current position. We will argue that this is true for H_B as the case of H_A is completely analogous.

Let I_B denote the intersection of the halfplanes in H_B. Since the bounding lines of those halfplanes are all below L at the current position of V, L has at least the intersection with V in common with I_B. This point lies also on the boundary of the halfplane which is bounded below by L. This implies what has to be guaranteed.

Similarly to Algorithm TRIVIAL PLANE SWEEP, degeneracies like parallel and concurrent lines can be incorporated easily. While the former case is completely trivial, the latter requires some care. It occurs when L intersects I_B or I_A in a vertex or both in the same point. If L intersects i lines in a single point, then up to $i+3$ insertions and deletions have to be performed.

LEMMA 3.3. *Algorithm* SOPHISTICATED PLANE SWEEP *constructs the k-belt of H in $O(b_k(n)\log^2 n)$ time and $O(n+b_k(H))$ space.*

Proof. At the far left, the line with the kth smallest slope is also the kth bottommost line. This line L can be determined in $O(n)$ time and the data structures H_B and H_A for the lines below and above L, respectively, can be constructed in $O(n\log n)$ time. These activities require $O(n)$ space.

Each intersection point of two lines on the boundaries of the k-belt can be determined in $O(\log^2 n)$ time and the structures H_B and H_A can then be adjusted in $O(\log^2 n)$ time, see Proposition 3.2. A careful implementation processes a vertex common to $i>1$ lines in $O(i\log^2 n)$ time. Note that this is less than the amount which is required in the worst case if those i lines define $\binom{i}{2}$ intersection points. This implies that $O(b_k(n)\log^2 n)$ time is spent for the computation and $O(b_k(H))$ space is occupied by the description of the k-belt. This completes the argument.

Both algorithms can easily be adapted to construct simplified belts instead of ordinary ones. For computing the m-simplified k-belt, only each mth vertex of the k-belt gives rise to a new edge to be created. Recall that a vertex common to $i\geqq 2$ lines is interpreted as $\min\{m, i-1\}$ vertices. This implies:

THEOREM 3.4. *Let S denote a set of n points in the plane. Then there exists an algorithm which constructs the m-simplified k-belt (the ordinary k-belt is the 1-simplified k-belt) of $H=T(S)$ in $O(b_k(n)\log^2 n)$ time and $O(b_k(n)/m)$ space.*

The space bound is actually $O(n+b_k(H))$ for ordinary k-belts. The proof is an immediate consequence of Lemma 3.3 and Observations 2.5 and 2.7.

This result is not optimal at least for $k=1$ which is shown by the following argument: Let L^+ denote the open halfplane which is bounded below by the nonvertical line L, and let L^- denote the open halfplane which is bounded above by L. Then the 1-belt B of H can be written as

$$B=E^2-\bigcap_{L\in H} L^+ - \bigcap_{L\in H} L^-.$$

Since the intersection of n halfplanes can be determined in $O(n\log n)$ time, see Proposition 3.2, this gives also a method which computes B in $O(n\log n)$ time.

4. Applications.

The methods presented in § 3 have interesting applications to seemingly unrelated problems. Algorithm TRIVIAL PLANE SWEEP yields a new method for finding a minimum area triangle with the three vertices taken from some given finite set. This is discussed in § 4.5. Applications of k-belts to halfplanar range estimation, finding so-called centerpoints, k-nearest neighbour search, and a problem with points moving on a line are discussed in §§ 4.1 through 4.4.

4.1. Halfplanar range estimation.

This section describes the use of ordinary and simplified belts for halfplanar range estimation. Representative for a variety of conceivable variants, we define three of the most basic halfplanar range estimation problems and show how to solve them.

Let S denote a set of n points in the plane and let k be some positive integer, with $k \leqq \lceil n/2 \rceil$. The simplest halfplanar range estimation problem, termed *simple* $EP(k)$, reads as follows: Let h denote a query halfplane which contains $A(h)$ points of S. Decide whether $0 \leqq A(h) < k$, $k \leqq A(h) \leqq n-k$, or $n-k < A(h) \leqq n$.

THEOREM 4.1. *Let S be a set of n points in the plane. There exists a data structure which requires* $O(b_k(n))$ *space and* $O(b_k(n) \log^2 n)$ *time for construction such that* $O(\log n)$ *time suffices to answer a query of the simple* $EP(k)$, *with* $k \leqq \lceil n/2 \rceil$.

Proof. The asserted bounds can be achieved with the k-belt of $H = T(S)$ as will be shown. The required amount of space and time for the construction follows immediately from Theorem 3.4. Thus, let us consider a query of the simple $EP(k)$. W.l.o.g. let the query halfplane h be bounded above by the line L. Then

(1) h contains less than k points if and only if $T(L)$ lies below the k-belt of H,

(2) h contains no less than k and no more than $n-k$ points if and only if $T(L)$ is contained in the k-belt, and

(3) h contains more than $n-k$ points if and only if $T(L)$ lies above the k-belt of H.

Which one of the three cases applies can be decided in $O(\log n)$ time by binary search since the two boundaries of the k-belt are monotone in x. This completes the argument.

Considerably more accurate answers can be obtained using several belts instead of a single one. We define the *uniform* $EP(\varepsilon)$, with $0 < \varepsilon < 1$, as follows: Let $j = \lfloor \lceil n/2 \rceil / \lfloor n^\varepsilon \rfloor \rfloor$ and let $A(h)$ denote the number of points of S in the query halfplane h. Determine an integer i, with $-j \leqq i \leqq j$, such that $(j-i)\lfloor n^\varepsilon \rfloor \leqq A(h) < (j-i+1)\lfloor n^\varepsilon \rfloor$ if $i<0$, $j\lfloor n^\varepsilon \rfloor \leqq A(h) \leqq n - j\lfloor n^\varepsilon \rfloor$ if $i=0$, and $n-(j-i+1)\lfloor n^\varepsilon \rfloor < A(h) \leqq n-(j-i)\lfloor n^\varepsilon \rfloor$ if $i>0$. This somewhat ugly definition of the problem has been chosen since the most simple combination of belts is able to solve it. Similar but differently defined halfplanar range estimation problems can be dealt with in more complicated but essentially equivalent combinations of belts. For convenience, we define $B_K(H) = \sum_{k \in K} b_k(H)$ for $H = T(S)$ and K subset of $\{1, 2, \cdots, \lceil n/2 \rceil\}$. $B_K(n)$ is then defined as the maximum of $B_K(H)$ for all sets S of n points.

THEOREM 4.2. *Let S be a set of n points in the plane and write H for* $T(S)$. *For the uniform* $EP(\varepsilon)$, *with* $0 < \varepsilon < 1$, *there exists a data structure which requires* $O(B_K(n))$ *space and* $O(B_K(n) \log^2 n)$ *time for construction, where* $K = \{\lfloor n^\varepsilon \rfloor, 2\lfloor n^\varepsilon \rfloor, \cdots, j\lfloor n^\varepsilon \rfloor\}$, *such that a query can be answered in* $O(\log n)$ *time.*

Before proceeding to the proof of this assertion let us comment on $B_K(n)$. No reasonable upper bounds are known for these sums. The best bound follows from Proposition 2.1 and claims that $B_K(n)$ is in $O(n^2)$ and in $O(|K| n^{3/2})$, with $|K|$ the number of indices in K.

Proof of Theorem 4.2. The data structure used for the uniform $EP(\varepsilon)$ consists of the k-belts of H, for k in K. Let h denote a query halfplane which is, say, bounded above by the line L. In order to answer the query, we determine the largest i' in $\{0, 1, \cdots, j\}$ such that $T(L)$ is contained in the $i\lfloor n^\varepsilon \rfloor$-belt of H. Then the answer is $-j+i'$ if $i'<j$ and $T(L)$ lies below the $(i'+1)\lfloor n^\varepsilon \rfloor$-belt, and it is $j-i'$, otherwise.

Notice that binary search testing $T(L)$ against the boundaries of the various belts is no longer appropriate in order to achieve $O(\log n)$ query time. In the place of that strategy we exploit a method due to Kirkpatrick [K]. Let D be a partition of the plane with a total of $|D|$ regions, straight edges, and vertices. Kirkpatrick's method permits us to determine in $O(\log |D|)$ time the region of D that contains a query point. $O(|D|)$ space and $O(|D| \log |D|)$ time for construction is required. The assertion follows since the collection of k-belts used induces a partition D of the plane with $|D| = O(B_K(n))$. This completes the argument.

Before considering simplified belts for halfplanar range estimation we note that k-belts of a line arrangement can be combined almost arbitrarily leading to all sorts of answers reflecting the approximate number of points in the respective query halfplane.

A serious drawback of ordinary belts is the superlinear space required in the worst case. In particular, the combination of a large number of belts needs a rather large amount of space. This drawback is bypassed by exploiting simplified belts as introduced in § 2.

Let S denote a set of n points in the plane, define $H = T(S)$, let k denote an integer with $1 \leqq k \leqq \lceil n/2 \rceil$, and let m denote an integer with $2 \leqq m \leqq 2\lceil n/2 \rceil - 2k + 1$ and $m \leqq b_k(H) - 1$. Note that the simple $EP(k)$ cannot be solved with the m-simplified k-belt of H, if $m > 1$. We define the *simple* $EP(k, m)$ as follows: Let h be a query halfplane and let $A(h)$ designate the number of points of S which are contained in h. Decide whether $0 \leqq A(h) < k + \lfloor m/2 \rfloor$, $k - \lfloor m/2 \rfloor \leqq A(h) \leqq n - k + \lfloor m/2 \rfloor$, or $n - k - \lfloor m/2 \rfloor < A(h) \leqq n$. Note that the three cases are not exclusive which implies that $A(h)$ does not uniquely determine the answer.

THEOREM 4.3. *Let S be a set of n points in the plane and $H = T(S)$. For the simple $EP(k, m)$ with $1 \leqq k \leqq \lceil n/2 \rceil$, $2 \leqq m \leqq 2\lceil n/2 \rceil - 2k + 1$, and $m \leqq b_k(H) - 1$, there exists a data structure which requires $O(b_k(n)/m)$ space and $O(b_k(n) \log^2 n)$ time for construction such that $O(\log n)$ time suffices to answer a query.*

Proof. The bounds for the required space and time for construction are trivially achieved by the m-simplified k-belt of H which we call M, see Observation 2.7 and Theorem 3.4. By Lemma 2.6, M is contained in the $\max\{0, k - \lfloor m/2 \rfloor\}$-belt of H and contains the $(k + \lfloor m/2 \rfloor)$-belt of H. Since $k - \lfloor m/2 \rfloor \geqq 0$ and $k + \lfloor m/2 \rfloor \leqq \lceil n/2 \rceil$ by definition of m, a query with halfplane h bounded above by L can be answered by deciding whether $T(L)$ lies below, in, or above M. This can be done in $O(\log n)$ time using binary search which completes the argument.

We say that a halfplanar range estimation problem has *accuracy* m if the overlap between two different answers is at most m. E.g. the problem solved by the $\lceil n^{1/2} \rceil$-simplified $\lfloor n/3 \rfloor$-belt M of H has accuracy $2\lfloor n^{1/2}/2 \rfloor$. That is, M decides with accuracy $2\lfloor n^{1/2}/2 \rfloor$ whether there are less than $\lfloor n/3 \rfloor$, from $\lfloor n/3 \rfloor$ to $n - \lfloor n/3 \rfloor$, or more than $n - \lfloor n/3 \rfloor$ of the points in the query halfplane. Due to Proposition 2.1 and Observation 2.7, M is known to require $O(n)$ space.

As for ordinary belts, almost arbitrary combinations of simplified belts can be used to obtain different solutions for different halfplanar range estimation problems. For example, a relaxed version of the uniform $EP(\varepsilon)$ can be solved with $\lceil n^{\varepsilon} \rceil$-simplified belts yielding:

THEOREM 4.4. *Let S be a set of n points in the plane and let ε be a real number, with $0 < \varepsilon < 1$. Then there exists a data structure which requires $O(n^{1-2\varepsilon} b_{n/2}(n))$ space and $O(n^{1-\varepsilon} b_{n/2}(n) \log^2 n)$ time for construction such that $O(\log n)$ time suffices to answer the uniform $EP(\varepsilon)$ with accuracy $\lfloor n^{\varepsilon} \rfloor$.*

4.2. Centerpoints. Let S be a set of n points in the plane. A point c not necessarily in S is called a *centerpoint of* S if the two closed halfplanes of any line through c contain both at least $\lceil n/3 \rceil$ points. The existence of a centerpoint for every set S is a consequence of Helly's theorem [H]. In dual space, $T(c)$ is a line such that for any point p of $T(c)$ there are respective at least $\lceil n/3 \rceil$ lines of $H = T(S)$ which are not below and not above p. So, $T(c)$ is contained in the $\lceil n/3 \rceil$-belt of H. The construction of the region of all centerpoints of S thus might proceed as follows:

1. Construct the $\lceil n/3 \rceil$-belt B of H.

2. Compute the intersection C of all closed halfplanes h defined as follows: Let line L bound h, then $T(L)$ is a vertex of B, and there are at most $\lceil n/3 \rceil - 2$ points outside of h.

Theorem 3.4 and a method for constructing the intersection of m halfplanes in $O(m \log m)$ time (Brown [Br]) imply

THEOREM 4.5. *$O(b_{\lceil n/3 \rceil}(n) \log^2 n)$ time and $O(b_{\lceil n/3 \rceil}(n))$ space suffices to construct the region of all centerpoints of a set of n points in the plane.*

A line $T(c)$ dual to a centerpoint c of S can be used for a surprisingly space efficient solution of a weak halfplanar range estimation problem.

THEOREM 4.6. *Let S be a set of n points in the plane. There is a data structure which takes constant space and $O(b_{\lceil n/3 \rceil}(n) \log^2 n)$ time for its construction such that constant time suffices to decide whether a given halfplane contains at most $n - \lceil n/3 \rceil$ or at least $\lceil n/3 \rceil$ points of S.*

We mention that developments following the ones reported in this paper lead to new algorithms which compute a single centerpoint in $O(n \log^4 n)$ time [CSY], [C].

4.3. *k*-nearest neighbours search. Let S denote a set of n points in the plane and let k be some integer with $1 \leqq k \leqq n-1$. The *k-nearest neighbours search problem* [SH], [L] requires the accommodation of S such that the k nearest to a later specified query point can be determined efficiently. We consider the problem restricted to query points in infinity which are better interpreted as directions. We assume that k is small compared to n which are the only interesting cases.

Let B denote the k-belt of $H = T(S)$. To each edge e of the lower boundary of B we assign the list of lines below including the line which supports e. Similarly, to each edge e of the upper boundary we assign the list of lines above including the line which supports e. Note that two unbounded edges whose union is a complete edge have assigned the same list which is now assigned to the complete edge. By construction, two adjacent complete edges can have the same list of lines. In such a case both complete edges are replaced by a single complete edge as described for simplified belts in § 2. The new complete edge has assigned the same list as the two complete edges it replaces. Let B' denote the modified belt which is obtained by repeatedly replacing adjacent complete edges with identical lists of lines. (Note that Algorithm SOPHISTICATED PLANE SWEEP can be used to construct B' without creating B.)

A direction in the original space transforms under T into a vertical line. The query is answered by binary search to identify the two complete edges of B' which intersect this vertical line. The k-nearest neighbours are those points whose corresponding lines are assigned to one of the two complete edges determined.

Some amount of space can be saved, in particular for large k, via the following method: Instead of assigning a list to each complete edge of B', we assign a list only to every kth of its complete edges. Let e denote a complete edge which has assigned a list of lines. The complete edge f immediately to the right of e stores the lines that must be deleted from, resp. inserted into, the list of e in order to obtain the one of f. In nondegenerate cases, f stores two lines. To cope with cases where f stores $2i > 2$ lines, f is effectively treated like a sequence of i complete edges. The analogous information is stored for each of the next $k-2$ complete edges to the right of f. The space required in the worst case is reduced by a factor k while the query time remains the same by the following strategy: For a vertical query line q, the lower (or upper) complete edge of B' which intersects q is determined. Then the first complete edge to the left of the latter is identified which has assigned a full list of lines. This list is updated during the walk back to the originally determined complete edge. The lists are stored as doubly linked lists and the updates to be performed are indicated by pointers to what has to be changed. A careful implementation of this method finally implies:

THEOREM 4.7. *Let S denote a set of n points in the plane. There exists a data structure which requires $O(f_k(S))$ space and $O(b_k(n)\log^2 n)$ time for construction such that $O(\log n+k)$ time suffices to report the k first points of S in a query direction.*

4.4. Moving points on a line. Let S be a set of n moving points on a vertical line. Each point p of S is specified by its location p_0 at time 0 and by its constant speed p_s which is positive if p moves upwards and negative if p moves downwards. We say that a point p is *at position k at time t* if it is the kth point from above at this moment t. Let $P_k(S)$ denote the sequence of points at position k during the time interval from minus infinity to plus infinity. This model of moving points is also considered by Ottmann and Wood [OW] who examine the construction of $P_1(S)$, for example.

THEOREM 4.8. *Let S be a set of n moving points on a line. Then there exists an algorithm which computes $P_k(S)$ in $O(b_k(n)\log^2 n)$ time and $O(n+|P_k(S)|)$ space, where $|P_k(S)|$ denotes the length of $P_k(S)$.*

Proof. Each point p of S is transformed into a line by introducing time as second coordinate with horizontal axis, say. Then the line corresponding to p intersects the vertical coordinate axis at location p_0 and has slope p_s. Thus, our problem transforms to computing the upper boundary of the k-belt of the line arrangement obtained. The assertion follows from Theorem 3.4 which completes the argument.

4.5. Minimum area triangle. Let S denote a set of n points in the plane, with $n \geqq 3$. A *minimum area triangle of S* is a triangle with minimum area whose vertices are chosen from S. Dobkin and Munro [DM] were the first to come up with a nontrivial solution that takes $O(n^2\log^2 n)$ time and $O(n^2\log n)$ space. Their method is based on

Observation 4.9. Let TR denote a minimum area triangle of S with vertices p, q, and r. Then r is a point in S different from p and q which is nearest to the line through p and q.

Observation 4.9 can be exploited for the determination of TR: for each line through two points of S compute a nearest of the remaining points. Let us consider the scenario in the dual space obtained by application of the transform T. The line through p and q corresponds to the intersection v of $T(p)$ and $T(q)$ and r corresponds to a line immediately below or above this intersection point, that is, r is hit first when v moves upwards or downwards in the vertical direction. Note that the line immediately below (or above) this intersection point is not unique if several lines intersect exactly immediately below (or above) this point. A trivial modification of Algorithm TRIVIAL PLANE SWEEP described in § 3 yields:

THEOREM 4.10. *Let S denote a set of $n \geqq 3$ points in the plane. Then there exists an algorithm which computes the minimum area triangle of S in $O(n^2\log n)$ time and $O(n)$ space.*

We note that following our developments [CGL], [EOS] developed an $O(n^2)$ time and space algorithm for minimum area triangles which is based on the same geometric observations as the algorithm above. The improvement in time is achieved by a new method for constructing a complete arrangement.

5. Discussions and dynamization. The authors consider the introduction of k-belts in arrangements of lines as the main contribution of this paper. This concept has applications to several problems defined for lines or, in dual space, for points. Among these applications are

(i) space efficient data structures for halfplanar range estimation in the plane,

(ii) finding centerpoints in the plane,
(iii) determining k first points for query directions, and
(iv) computing the sequence of kth points of a dynamically changing set on a line.

All these applications rely on an efficient algorithm which computes a belt with $O(\log^2 n)$ time per edge. A less efficient but more powerful method which constructs belts in $O(n^2 \log n)$ time can also be used to find a triangle with minimum area whose vertices are taken from a finite set of points in the plane.

All data structures presented for halfplanar range estimation are inherently *static*, that is, there is no way to perform insertions of new points or deletions of points efficiently. Data structures which accommodate those two operations are called *dynamic*. There exists an extensive literature about general methods which convert static data structures for so-called decomposable searching problems into dynamic ones, see e.g. Bentley [B], Maurer and Ottmann [MO], and Overmars and van Leeuwen [OvL2]. A searching problem is said to be *decomposable* if the answer for any set S of objects and any query objects q can be derived in constant time from the answers for S_1 and q, and for S_2 and q, respectively, for every partition of S into S_1 and S_2. It is easily verified that the halfplanar range search problem is decomposable while the halfplanar range estimation problems considered in § 4.1 are not. Nevertheless, the general dynamization methods can be applied to the data structures of § 4.1. The strange effect of those conversions is that the data structures as well as the problems are changed in a meaningful way.

Representative for other and similar cases, let us consider the simple $EP(\lfloor n/3 \rfloor)$ for a set S of n points in the plane. Let $S_1, S_2, \cdots, S_m$ be an arbitrary partition of S into subsets of about $n^{1/2}$ points each. Hence, m is also about $n^{1/2}$. Let $|S_i|$ denote the number of points in S_i. Instead of the $\lfloor n/3 \rfloor$-belt for $T(S)$ we maintain an $\lfloor |S_i|/3 \rfloor$-belt for each subset S_i. For details of the maintenance strategies of this collection of belts we refer to Maurer and Ottmann [MO]. A point is inserted into (or deleted from) the system by reconstructing only one belt which takes $O(b_k(n/m) \log^2 n)$ time, with k about $n^{1/2}/3$. A query is performed on each data structure which takes $O(n^{1/2} \log n)$ time. Let $A(h)$ denote the exact number of points of S contained in h. The m answers can be used to derive an interval of length about $n/3$ which contains $A(h)$. This interval can be derived from the m answers since the answer for S_i, with $1 \leqq i \leqq m$, decides for about two thirds of the points in S_i whether or not they are contained in the query halfplane.

Let us finally give a number of open problems which come naturally from the investigations presented. (1) Give tighter bounds on $f_k(n)$, that is, improve Proposition 2.1. It is worthwhile to note that Erdös, Lovasz, Simmons and Straus [ELSS] and Edelsbrunner and Welzl [EW1] conjecture that the derived lower bounds are closer to the truth than the upper bounds. (2) Section 4.1 has exploited collections of k-belts as data structures. Let K be some fixed subset of $\{1, 2, \cdots, \lceil n/2 \rceil\}$ and let $F_K(n)$ denote the maximum of $\sum_{k \in K} f_k(S)$ for all sets S of n points in the plane. Calculate a lower and an upper bound for $F_K(n)$. (3) Let $a(S)$ denote the maximal number of edges of an arbitrary x-monotone polygonal chain which is a subset of $\bigcup_{p \in S} T(p)$. Let $a(n)$ denote the maximum of $a(S)$ for all sets S of n points in the plane. Note that Proposition 2.1 implies that $a(n)$ is in $\Omega(n \log n)$. Derive an upper bound for $a(n)$. (4) Can the construction of a k-belt of a set of n lines be accomplished in $O(b_k(n) \log n)$ time? (Compare Theorem 3.4.) (5) The data structure described in § 4.3 has similarities with the order-k Voronoi diagram, see Shamos and Hoey [SH] or Lee [L]. This diagram for a set S of n planar points is a partition of the plane into convex regions. Each region R is associated with a subset S' of S which contains k points such that an arbitrary point falls into R if and only if the points in S' are nearer to p

than the points in $S-S'$. Does there exist an algorithm which constructs the order-k Voronoi diagram of S based on the method described in § 4.3? It is tempting to conjecture that such an algorithm can improve the algorithm due to Lee [L] which constructs, successively, all order-i Voronoi diagrams, for $1 \leqq i \leqq k$. (6) At last, it is also interesting to pose the same questions in a higher dimensional environment. Most important: How many k-sets can be realized by a set of n points in three dimensions?

Note added in proof. It has been proved that $F_K(n) \in O(n(\sum_{k \in K} k)^{1/2})$ (E. Welzl, *More on k-sets of finite sets in the plane*, Rep. 204, Inst. for Information Processing, Technical Univ. Graz, Graz, Austria, 1985). Moreover, we observed that $a(n) \in \Omega(n^2)$.

REFERENCES

[AHU] A. V. AHO, J. E. HOPCROFT AND J. D. ULLMAN, *The Design and Analysis of Computer Algorithms*, Addison-Wesley, Reading, MA, 1974.

[B] J. L. BENTLEY, *Decomposable searching problems*, Inform. Process. Lett., 8 (1979), pp. 244–251.

[BF] J. L. BENTLEY AND J. H. FRIEDMAN, *Data structures for range searching*, ACM Comput. Surveys, 11 (1979), pp. 397–409.

[Br] K. Q. BROWN, *Geometric transforms for fast geometric algorithms*, Ph.D. thesis, Rep. CMU-CS-80-101, Dept. Computer Science, Carnegie-Mellon Univ., Pittsburgh, PA, 1980.

[CGL] B. M. CHAZELLE, L. J. GUIBAS AND D. T. LEE, *The power of geometric duality*, Proc. 24th Annual IEEE Symposium on Foundations of Computer Science, 1983, pp. 217–225.

[C] R. COLE, *Slowing down sorting networks to obtain faster sorting algorithms*, Rep. 117, Dept. Comput. Sci., New York Univ., NY, 1984.

[CSY] R. COLE, M. SHARIR AND C. K. YAP, *On k-hulls and related problems*, Proc. 16th Annual ACMS Symposium on Theory of Computing 1984, pp. 154–166.

[DM] D. P. DOBKIN AND J. I. MUNRO, private communication.

[EKM] H. EDELSBRUNNER, D. G. KIRKPATRICK AND H. A. MAURER, *Polygonal intersection searching*, Inform. Process. Lett., 14 (1982), pp. 74–79.

[EOS] H. EDELSBRUNNER, J. O'ROURKE AND R. SEIDEL, *Constructing arrangements of lines and hyperplanes with applications*, Proc. 24th Annual IEEE Symposium on Foundations of Computer Sciences, 1983, pp. 83–91; this Journal, 15 (1986), to appear.

[EW1] H. EDELSBRUNNER AND E. WELZL, *On the number of line-separations of a finite set in the plane*, J. Combin. Theory Ser. A, 38 (1985), pp. 15–29.

[EW2] ——— *Halfplanar range search in linear space and $O(n^{0.695})$ query time*, Rep. F111, Inst. for Information Processing, Technical Univ. Graz, Graz, Austria, 1983.

[ELSS] P. ERDÖS, L. LOVASZ, A. SIMMONS AND E. G. STRAUS, *Dissection graphs of planar point sets*, in A Survey of Combinatorial Theory, J. N. Srivastava et al., eds., North-Holland, Amsterdam, 1973, pp. 139–149.

[F] M. L. FREDMAN, *The inherent complexity of dynamic data structures which accommodate range queries*, Proc. 21st Annual IEEE Symposium on Foundations of Computer Science, 1980, pp. 191–199.

[H] E. HELLY, *Ueber Mengen konvexer Koerper mit gemeinschaftlichen Punkten*, Jber. deutsch. Math. Verein., 32 (1923), pp. 175–176.

[K] D. G. KIRKPATRICK, *Optimal search in planar subdivisions*, this Journal, 12 (1983), pp. 28–35.

[L] D. T. LEE, *On k-nearest neighbor Voronoi diagrams in the plane*, IEEE Trans. Comput. C-31 (1982), pp. 478–487.

[MO] H. A. MAURER AND TH. OTTMANN, *Dynamic solutions of decomposable searching problems*, in Discrete Structures and Algorithms, U. Pape, ed., Carl Hanser, 1979, pp. 17–24.

[OW] TH. OTTMANN AND D. WOOD, *Dynamical sets of points*, Rep. CS-82-56, Dept. Computer Science, Univ. Waterloo, Waterloo, Ontario, Canada, 1982.

[OvL1] M. H. OVERMARS AND L. VAN LEEUWEN, *Maintenance of configurations in the plane*, J. Comput. System Sci., 23 (1981), pp. 166–204.

[OvL2] M. H. OVERMARS AND J. VAN LEEUWEN, *Worst-case optimal insertion and deletion for decomposable searching problems*, Inform. Process. Lett., 12 (1981), pp. 168–173.

[S] M. I. SHAMOS, *Computational geometry*, Ph.D. Thesis, Dept. Computer Science, Yale Univ., New Haven, CT, 1978.

[SH] M. I. SHAMOS AND D. HOEY, *Closest-point problems*, Proc. 16th Annual IEEE Symposium on Foundations of Computer Science, 1976, pp. 208–215.