CMSC 754: Lecture 15
Applications of Arrangements

Reading: Chapter 8 in the 4M's. Some of this material is not covered in the book.

Applications of Arrangements and Duality: Last time we introduced the concept of an arrangement of lines in the plane, and we showed how to construct such an arrangement in $O(n^2)$ time. Line arrangements, when combined with the dual transformation, make it possible to solve a number of geometric computational problems. A number of examples are given below. Unless otherwise stated, all these problems can be solved in $O(n^2)$ time and $O(n^2)$ space by constructing a line arrangement. Alternately, they can be solved in $O(n^2 \log n)$ time and $O(n)$ space by applying plane sweep to the arrangement.

General position test: Given a set of n points in the plane, determine whether any three are collinear.

Minimum area triangle: Given a set of n points in the plane, determine the minimum area triangle whose vertices are selected from these points.

Minimum k-corridor: Given a set of n points, and an integer k, determine the narrowest pair of parallel lines that enclose at least k points of the set. The distance between the lines can be defined either as the vertical distance between the lines or the perpendicular distance between the lines (see Fig. 1(a)).

Visibility graph: Given line segments in the plane, we say that two points are *visible* if the interior of the line segment joining them intersects none of the segments. Given a set of n non-intersecting line segments, compute the *visibility graph*, whose vertices are the endpoints of the segments, and whose edges a pairs of visible endpoints (see Fig. 1(b)).

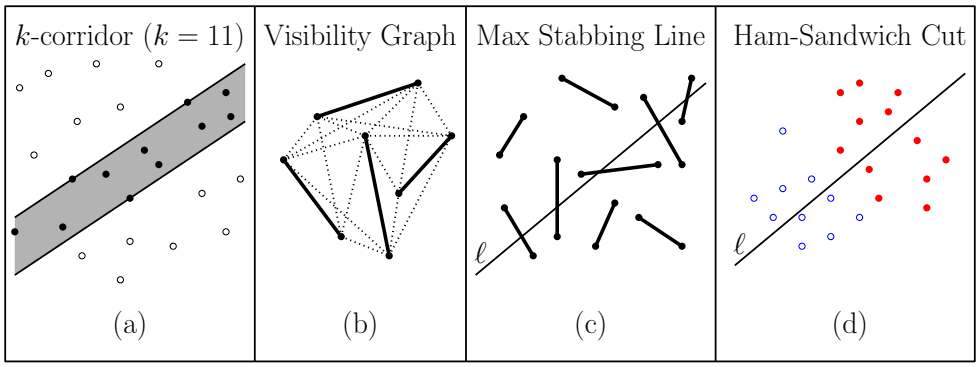

Fig. 1: Applications of arrangements.

Maximum stabbing line: Given a set of n line segments in the plane, compute the line ℓ that stabs (intersects) the maximum number of these line segments (see Fig. 1(c)).

Ham Sandwich Cut: Given n red points and m blue points, find a single line ℓ that simultaneously bisects these point sets. It is a famous fact from mathematics, called the *Ham-Sandwich Theorem*, that such a line always exists. If the two point sets are separable by a line (that is, the red convex hull and the blue convex hull do not intersect), then this can be solved in time $O(n + m)$ (see Fig. 1(d)).

In the remainder of the lecture, we'll see how problems like these can be solved through the use of arrangements.

Sweeping Arrangements: Since an arrangement of n lines is of size $\Theta(n^2)$, we cannot expect to solve problems through the explicit use of arrangements in less than quadratic time. Most applications involve first constructing the arrangement, and then traversing it in some manner. In many instances, the most natural traversal to use is based on a plane-sweep. (This is not the only way however. Since a planar arrangement is a graph, methods such as depth-first and breadth-first search can be used.)

If an arrangement is to be built just so it can be swept, then maybe you don't need to construct the arrangement at all. You can just perform the plane sweep on the lines, exactly as we did for the line segment intersection algorithm. Assuming that we are sweeping from left to right, the initial position of the sweep line is at $x = -\infty$ (which means sorting by slope). The sweep line status maintains the lines in, say, bottom to top order according to their intersection with the sweep line. The events are the vertices of the arrangement.

Note that the sweep-line status always contains exactly n entries. Whenever an intersection event occurs, we can update the sweep-line status by swapping two adjacent entries. Thus, instead of an ordered dictionary, it suffices to store the lines in a simple n-element array, sorted, say, from top to bottom. This means the **sweep-line updates can be performed in $O(1)$ time**, rather than $O(\log n)$(see Fig. 2(a)). We still need to maintain the priority queue, and these operations take $O(\log n)$ time each.

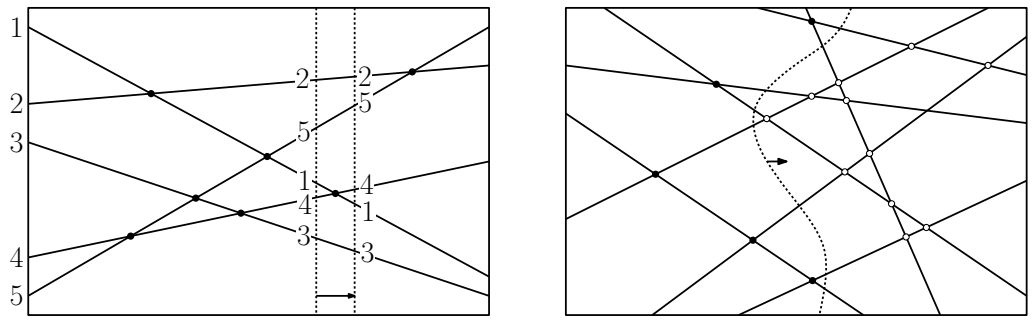

Fig. 2: Sweeping a line arrangement.

Sweeping an arrangement in this manner takes $O(n^2 \log n)$ time, and $O(n)$ space. Because it is more space-efficient, this is often an attractive alternative to constructing the entire subdivision.

Topological Plane Sweep: (Optional) As mentioned above, the priority queue is the slowest part of plane sweeping an arrangement. Remarkably, there is a way to save this $O(\log n)$ factor, but we must abandon hope of sweeping events in purely left-to-right order. There is a somewhat more "relaxed" version of plane sweep, which works for line arrangements in the plane. The method is called *topological plane sweep*. It runs in $O(n^2)$ time (thus, eliminating an $O(\log n)$ factor from the running time) and uses $O(n)$ space.

It achieves efficiency relaxing the requirement that vertices be swept in strict left-to-right order. Rather, it uses a more "local" approach for deciding which vertex of the arrangement to sweep next.[1] This local approach guarantees that the vertices along each line are swept in their proper order, even though vertices lying on different lines are not necessarily swept in their proper left-to-right order. Intuitively, we can think of the sweep line as a sort of *pseudoline*, that intersects each line of the arrangement exactly once (see Fig. 2(b)). Although we will not present any justification, it method applicable to all the problems we will discuss in today's lecture.

Duality: Many of our applications will involve the dual transformation, which we introduced earlier in the semester. Recall that the dual of a point $p = (a, b)$ is the line $p^* : y = ax - b$, and the dual of a line $\ell : y = ax - b$ is the point $\ell^* = (a, b)$. Also recall the *order-reversing property* that the point p lies above line ℓ (at vertical distance h) if and only if the dual line p^* lies below dual point ℓ^* (also at vertical distance h).

Narrowest k-corridor: We are given a set P of n points in the plane and an integer k, $1 \leq k \leq n$, and we wish to determine the narrowest pair of parallel lines that enclose at least k points of the set. (We call this a *slab* or *corridor*.) We define the *width* of the corridor to be the vertical distance between these. Our objective is to compute the corridor of minimum width that encloses k points (which may lie on the corridor's boundary.) It is straightforward to adapt the algorithm to minimize the perpendicular distance between the lines. We will make the usual general-position assumptions that no three points of P are collinear and no two points have the same x-coordinate.

Consider any corridor defined by parallel lines ℓ_a (above) and ℓ_b (below) (see Fig. 3(a)). Since the lines are parallel, these points share the same a-coordinate, which implies that the line segment $\overline{\ell_a^* \ell_b^*}$ is vertical (see Fig. 3(b)). The vertical distance between the lines (that is, the difference in their y-intercepts) is the same as the vertical distance between the dual points.

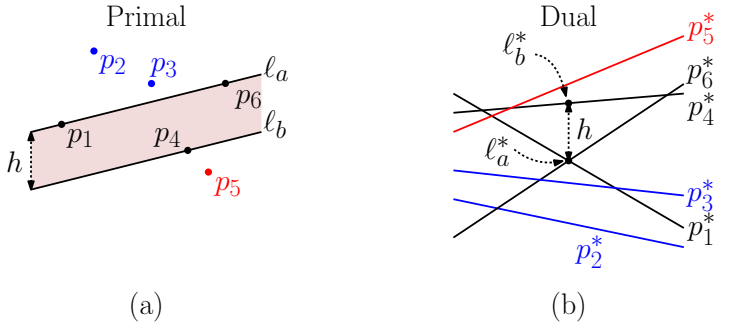

Fig. 3: A 3-corridor in primal and dual forms. (Note that the corridor is not as narrow as possible.)

By the order-reversing property, points that lie above/below/within the corridor (shown in blue, red, and black in Fig. 3(a), respectively) are mapped to dual lines that pass below/above/through this segment (see Fig. 3(b)). Thus, we have the following equivalent dual formulation of this problem:

[1] For details, see "Topologically sweeping an arrangement" by H. Edelsbrunner and L. J. Guibas, *J. Comput. Syst. Sci.*, 38 (1989), 165–194.

Shortest vertical k-stabber: Given an arrangement of n lines, determine the shortest vertical segment that stabs (intersects) k lines of the arrangement.

It is easy to show that the shortest vertical k-stabber may be assumed to have one of its endpoints on a vertex of the arrangement. (If the vertical line has endpoints in the interior of two edges, we can slide it left or right and decrease its length.)

3-stabber: The 3-stabber is the simplest to describe. A perform a simple plane sweep of the arrangement (using a vertical sweep line). Whenever we encounter a vertex of the arrangement, we consider the distance from this vertex to the edge of the arrangement lying immediately above this vertex and the edge lying immediately below. (These are illustrated by the blue broken lines in Fig. 4(a).) We can solve this problem by plane sweep in $O(n^2 \log n)$ time and $O(n)$ space. (By using topological plane sweep, the extra $\log n$ factor in the running time can be removed.)

Note that we can use this to test whether points are in general position. It is easy to prove that a set of n points has three or more collinear points if and only if the dual arrangement's minimum 3-stabber is of length zero.

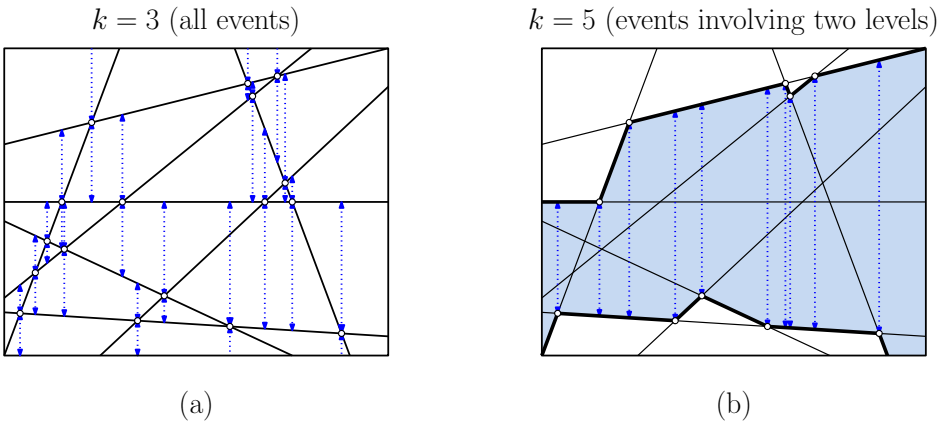

Fig. 4: The critical events in computing the shortest vertical 3-stabber (a) and 5-stabber (b).

k-stabber: Whenever we encounter a vertex in the plane sweep, we determine the distance to the lines of the arrangement that lie $k-2$ above and $k-2$ below (see the blue broken lines of Fig.4(b)). The reason for the "-2" is to account for two lines that pass through the vertex itself. Recalling that the sweep-line status can be stored in a simple n-element array, it is possible to access these entries in $O(1)$ time for any value of k.

Halfplane Discrepancy: Next we consider a problem derived geometric sampling. Suppose that we are given a collection of n points P lying in a unit square $U = [0,1]^2$. We want to use these points for random sampling purposes. In particular, the property that we would like these points to possess is that for any halfplane h, we the fraction of points of P that lie within h should be roughly equal to the area of intersection of h with U. More precisely, define $\mu(h)$ to be the area of $h \cap U$, and $\mu_P(h) = |P \cap h|/|P|$. A sample is good if $\mu(h) \approx \mu_P(h)$, for any choice of h.

To make this more formal, we define the *discrepancy* (or more accurately, the *halfplane discrepancy*) of a finite point set P with respect to a halfplane h to be

$$\Delta_P(h) = |\mu(h) - \mu_P(h)|.$$

For example, in Fig. 5(a), the area of $h \cap U$ is $\mu(h) = 0.625$, and there are 7 out of 13 points in h, thus $\mu_P(h) = 7/13 = 0.538$. Thus, the discrepancy of h is $|0.625 - 0.538| = 0.087$. Define the *halfplane discrepancy* of P to be the maximum (or more properly the supremum, or least upper bound) of this quantity over all halfplanes:

$$\Delta(P) = \sup_h \Delta_P(h).$$

Let's consider the problem of computing the discrepancy of a given point set P.

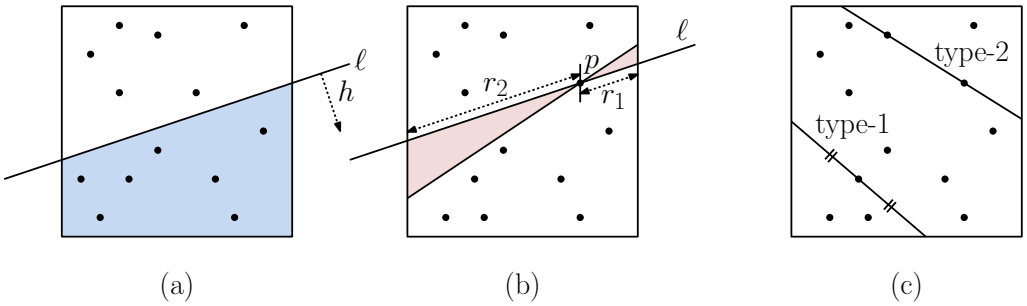

Fig. 5: Discrepancy of a point set.

Since there are an uncountably infinite number of halfplanes that intersect the unit square, we should first derive some sort of *finiteness criterion* on the set of halfplanes that might produce the greatest discrepancy.

Lemma: Let h denote the halfplane that generates the maximum discrepancy with respect to P, and let ℓ denote the line that bounds h. Then either:

(i) ℓ passes through one point of P, and this point is the midpoint of the line segment $\ell \cap U$, or

(ii) ℓ passes through two points of P.

Remark: If a line passes through one or more points of P, then should this point be included in $\mu_P(h)$? For the purposes of computing the maximum discrepancy, the answer is to either include or omit the point, whichever produces the larger discrepancy. The justification is that it is possible to perturb h infinitesimally so that it includes none or all of these points without altering $\mu(h)$.

Proof: We will show that any line can be moved until it satisfies either (i) or (ii) in such a manner that the discrepancy never decreases. First, if ℓ does not pass through any point of P, then (depending on which is larger $\mu(h)$ or $\mu_P(h)$) we can move the line up or down without changing $\mu_P(h)$ and increasing or decreasing $\mu(h)$ to increase their difference, until it does pass through a point of P. Next, if ℓ passes through a point $p \in P$, but is not the midpoint of the line segment $\ell \cap U$, then we claim that we can

rotate this line about p and hence increase or decrease $\mu(h)$ without altering $\mu_P(h)$, to increase their difference.

To establish the claim, consider Fig. 5(b). Suppose that the line ℓ passes through point p and let $r_1 < r_2$ denote the two lengths along ℓ from p to the sides of the square. Observe that if we rotate ℓ through a small angle θ, then to a first order approximation, the gain due to area of the triangle on the right is $r_1^2\theta/2$, since this triangle can be approximated by an angular sector of a circle of radius r_1 and angle θ. The loss due to the area of the triangle on the left is $r_2^2\theta/2$. Thus, since $r_1 < r_2$ this rotation will decrease the area of region lying below h infinitesimally. A rotation in the opposite increases the area infinitesimally. Since the number of points bounded by h does not change as a function of θ, the discrepancy cannot be achieved as long as such a rotation is possible.

We say that a line is *type-1* if it satisfies condition (i) and it is *type-2* if it satisfies condition (2) (see Fig. 5(c)). We will show that the discrepancy for each types of lines can be computed in $O(n^2)$ time.

Type-1: For each point $p \in P$, there are only a constant number of lines ℓ (at most two, I believe) through this point such that p is the midpoint of $\ell \cap U$. It follows that there are at most $O(n)$ type-1 lines. We can compute the discrepancy of each such line in $O(n)$ time, which leads to a total running time of $O(n^2)$.

Type-2: Consider a type-2 line ℓ that passes through two points $p_i, p_j \in P$. This line defines two halfplanes, one above and one below. We'll explain how to compute the discrepancy of the lower halfplane, h^-, and the upper halfplane, h^+, is symmetrical. First, observe that we can compute $\mu(h^-)$ in constant time, so all that remains is computing $\mu_P(h^-)$, that is, the number of points lying on or below ℓ.

If we apply our standard dual transformation, ℓ is mapped in the dual plane to a point ℓ^* at which the dual lines p_i^* and p_j^* intersect. This is just a vertex in the line arrangement $\mathcal{A}(P^*)$. By the order-reversing property of the dual transformation, the points lying on or below ℓ coincide with the dual lines that lie on or above this vertex.

We can compute this quantity in constant time for each vertex of the line arrangement. Recall that the sweep-line status is stored in a simple n-element array, sorted from top to bottom. The vertex in the arrangement corresponds to two consecutive entries in the sweep-line status, say at positions $k-1$ and k. The number of dual lines lying on or above the vertex is therefore just k (assuming that we index the array from 1 to n).

For example, consider the vertex being swept in Fig. 2(a). These intersecting lines are at indices $k-1 = 3$ and $k = 4$ in the sweep-line status, and hence there are 4 lines on or above this vertex, which implies that there are 4 points lying on or below the corresponding type-2 line $\overline{p_1p_4}$ in the primal configuration. Since we can compute the discrepancy for each type-2 line in $O(1)$ time, the overall time to compute the discrepancies of all type-2 lines is $O(n^2)$.

Levels: The analysis that was done above for type-2 lines suggests another useful structure within a line arrangement. We can classify each element of the arrangement according to the number of lines lying above and below it. We say that a point is at *level* k, denoted $\mathcal{L}_k$, in an arrangement if there are at most $k-1$ lines (strictly) above this point and at most $n-k$ lines

(strictly) below it. It is easy to see that $\mathcal{L}_k$ is an x-monotone polygonal curve (see Fig. 6(a)). For example, $\mathcal{L}_1$ is the upper envelope of the lines, and $\mathcal{L}_n$ is the lower envelope. Assuming general position, each vertex of the arrangement is on two levels, which meet at this vertex in a "knocked-knee" manner. Given the arrangement $\mathcal{A}(L)$ of a set of n lines, it is an easy matter to label every edge of the arrangement with its level number, by tracking its index in the sweep-line status.

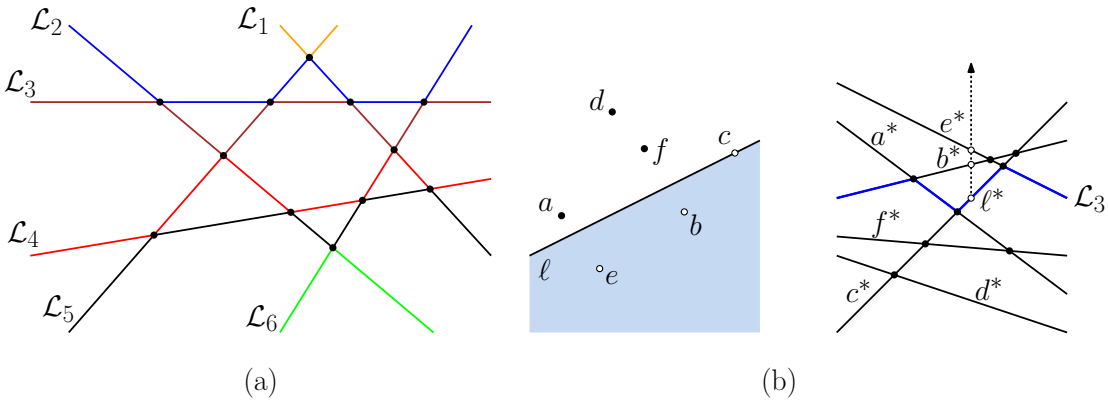

Fig. 6: Levels in an arrangement and k-sets.

There is a dual equivalence between a level in an arrangement and a concept called k-sets. Given an n-element point set P and integer $1 \leq k \leq n$, a k-element subset of $P' \subseteq P$ is called a *k-set* of P if there exists a halfplane h such that $P' = P \cap h$. For example, if (p_i, p_j) is an edge of the convex hull of P, then $P' = \{p_i, p_j\}$ is a 2-set of P. A classical question in combinatorial geometry is, as a function of n and k, what is the maximum number of possible k-sets that any n-element set can have. (The current best bounds range between $O(n \log k)$ and $O(nk^{1/3})$.)

There is a close relationship between the k-sets of P and level k of the dual arrangement $\mathcal{A}(P^*)$. To see this, let us first distinguish between two types of k-sets. We say that a k-set is a *lower k-set* if it lies in the halfplane beneath a line ℓ and otherwise it is an *upper k-set*. Let's just consider lower k-sets, since upper k-sets are symmetrical (by reflecting the points about the x-axis).

Consider any lower k-set defined by some line ℓ. We may assume that ℓ passes through a point of P, and hence there are $k-1$ points strictly below ℓ. The associated dual point ℓ^* lies on an edge of the dual arrangement, and by the order-reversing property of the dual transformation, there are $k-1$ lines of $\mathcal{A}(P^*)$ that pass strictly above ℓ^*. (For example, in Fig. 6(b), the lower 3-set $\{c, b, e\}$ is defined by a line ℓ, which passes through c. In the dual setting, the point ℓ^* lies on the dual line c^* and lies on level $\mathcal{L}_3$ because lines b^* and e^* lie above it.) The upper k-sets can be identified with level $\mathcal{L}_{n-k+1}$, because each point on this level has k lines passing on or below it, and hence $n-k+1$ lines on or above.

Sorting all angular sequences: Earlier, we introduced the problem of computing visibility graphs. We will not explicitly discuss the solution of that problem here, but we will discuss a fundamental subroutine in this algorithm. Consider a set of n points in the plane. For each

point p in this set we want to sort the remaining $n-1$ point in cyclic order. Clearly, we can compute the cyclically sorted order about any given point in $O(n \log n)$ time, and this leads to an overall running time of $O(n^2 \log n)$. We will show that we can do this $O(n^2)$ time. (This is rather surprising. Lower bounds on sorting imply that we cannot sort n sets of $n-1$ numbers faster than $\Omega(n^2 \log n)$ time. But here we are exploiting the special structure of the cyclically ordered point sets.)

Here is how we do it. Suppose that p is the point around which we want to sort, and let $\langle p_1, \ldots, p_n \rangle$ be the points in final angular order about p (see Fig. 7(a)). Consider the arrangement defined by the dual lines p_i^*. How does this order manifest itself in the arrangement?

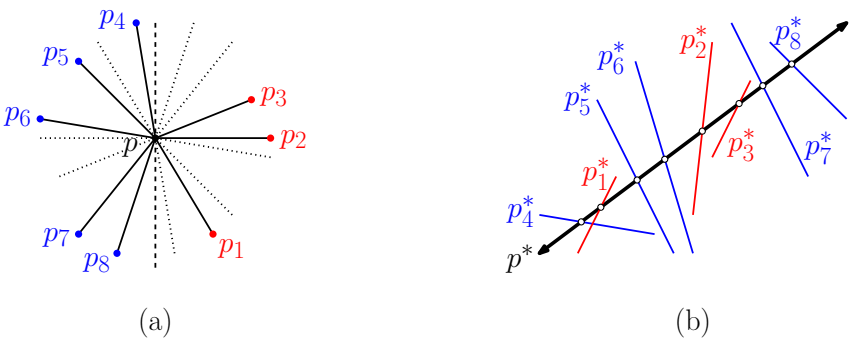

Fig. 7: Arrangements and angular sequences.

Consider the dual line p^*, and its intersection points with each of the dual lines p_i^*. These form a sequence of vertices in the arrangement along p^*. Consider this sequence ordered from left to right. It would be nice if this order were the desired circular order, but this is not quite correct. It follows from the definition of our dual transformation that the a-coordinate of each of these vertices in the dual arrangement is the slope of some line of the form $\overline{pp_i}$ in the primal plane. Thus, the sequence in which the vertices appear on the line is a *slope ordering* of the points about p_i, which is not quite the same as the *angular ordering*.

However, given this slope ordering, we can simply test which primal points lie to the left of p (shown in blue in Fig. 7(a)), and separate them from the points that lie to the right of p (shown in red in Fig. 7(a)). We partition the vertices into two sorted sequences, and then concatenate these two sequences, with the points on the right side first, and the points on the left side later. For example, in Fig. 7, we partition the slope-sorted sequence $\langle 4, 1, 5, 6, 2, 3, 7, 8 \rangle$ into the left sequence $\langle 4, 5, 6, 7, 8 \rangle$ and the right sequence $\langle 1, 2, 3 \rangle$, and then we concanate them to obtain the final angle-sorted sequence $\langle 1, 2, 3, 4, 5, 6, 7, 8 \rangle$.

Thus, once the arrangement has been constructed, we can reconstruct each of the angular orderings in $O(n)$ time, for a total of $O(n^2)$ time. (Topological plane sweep can also be used, but since the output size is $\Omega(n^2)$, there no real benefit to be achieved by using topological plane sweep.)