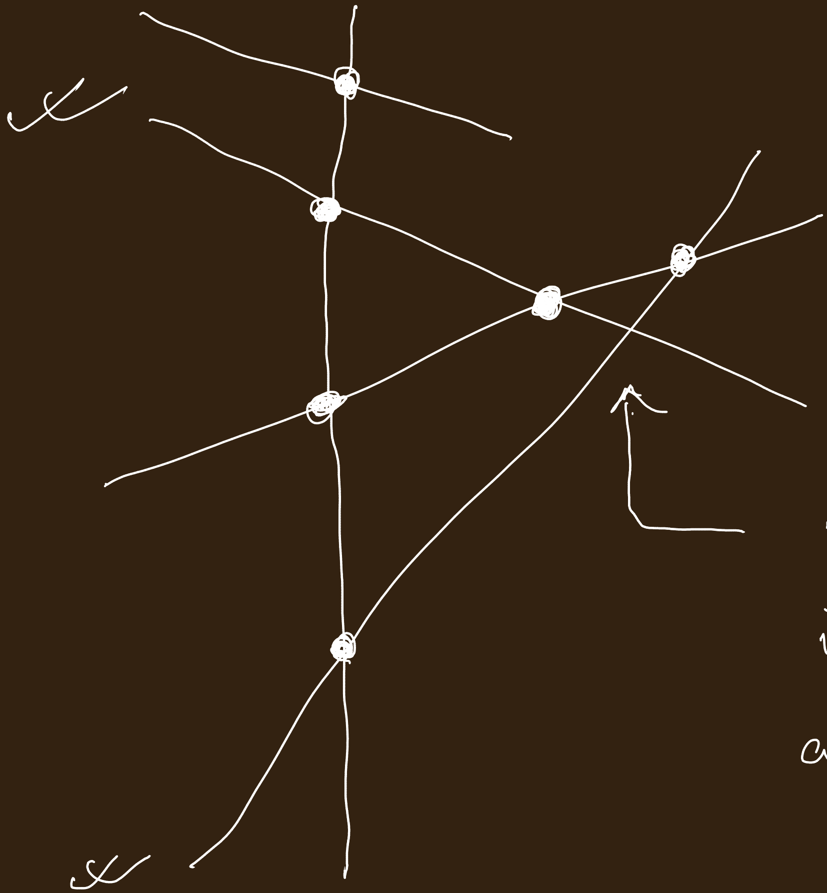
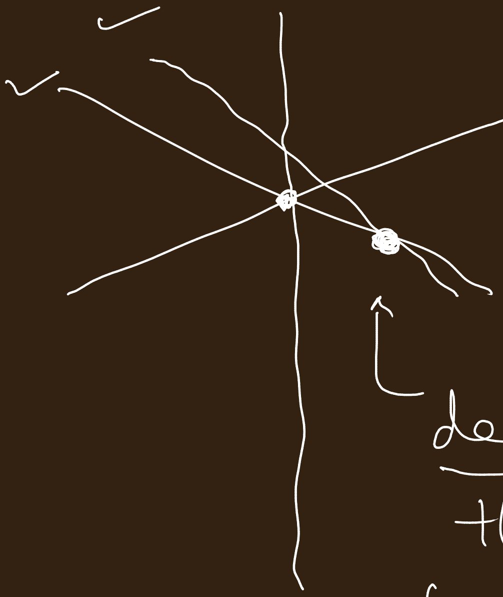
Line-sweep algorithm (LSI)

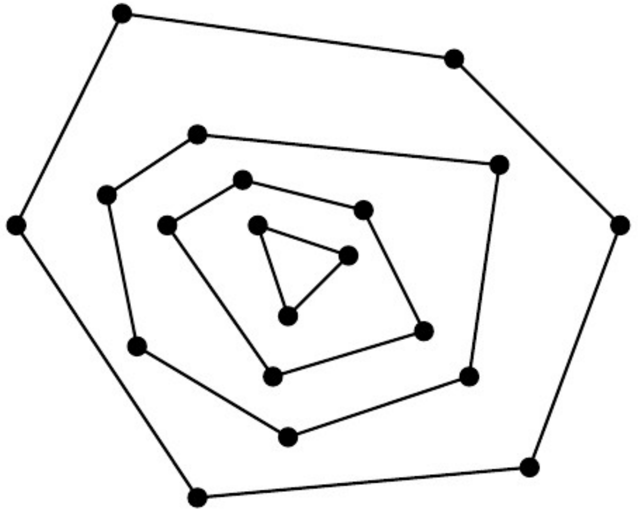
What are consecutive segments on the sweep line?





To ensure
space complexity is $O(n)$.

1. Let S be a set of n points in the plane. Let L_1 denote the set of vertices of the convex hull $\mathrm{CH}(S)$. We remove the points in L_1 from S, and compute the convex hull again. Now, let L_2 denote the set of vertices of $\mathrm{CH}(S \setminus L_1)$. Then, we delete the points of L_2 from $S \setminus L_1$, and repeat the same process until all points are removed. $L_1, L_2, L_3, \ldots$ are called the *onion layers* of S. The following figure describes this concept.



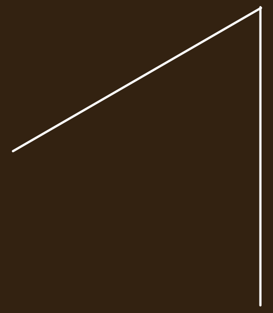
Suppose that we use Jarvis's march for each convex-hull construction. What is the worst-case running time for computing all the onion layers of S? Select the tightest bound.

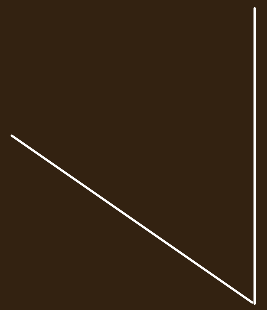
$$O(nh_1 + (n-h_1)h_2 + (n-h_1-h_2)h_3 + \cdots) = O(nh_1 + nh_2 + nh_3 + \cdots) = O(n^2)$$

This is a tight bound because we may have a single layer with n vertices.

2. Suppose that in Graham's scan, pairs of points (not three or more) may have the same x-coordinates. How can you modify the algorithm to handle this degeneracy?

For the upper hull:
Sort first with respect to increasing x-coordinates, and then (in case of ties) with respect to decreasing y-coordinates


Right turn


Wrong turn

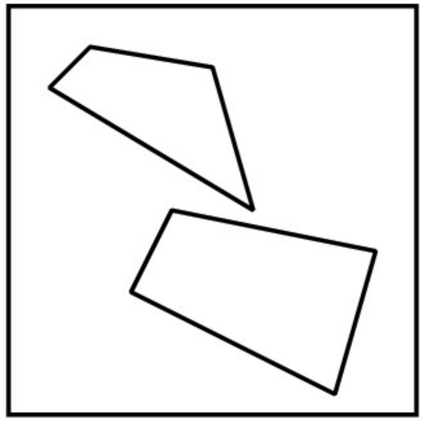
3. Let S and T be two disjoint sets of points in the Euclidean plane. S and T need not be horizontally separated. You have computed the two convex hulls CH(S) and CH(T).

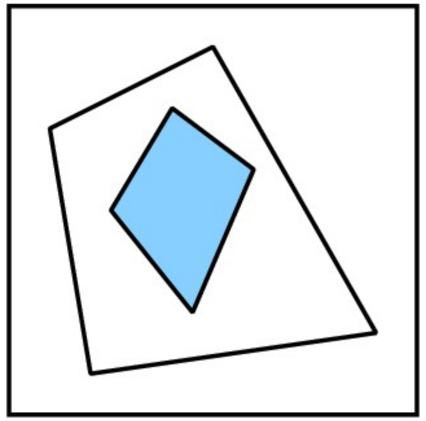
   (a) Propose an O(n)-time algorithm to merge these two hulls to $CH(S \cup T)$, where $n = | S \cup T |$.

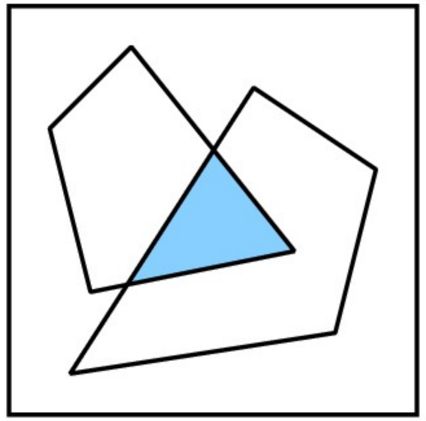
As the exercise on quadrilateral intersections illustrates, we may have to construct many tangents. We can instead do the following.

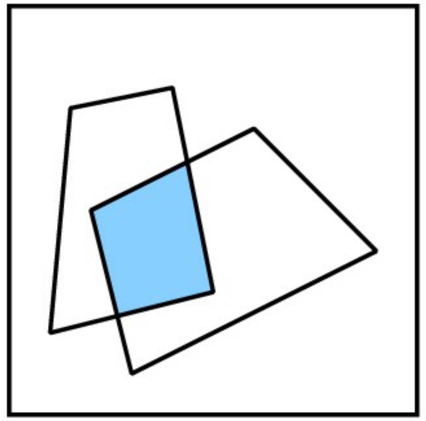
From CH(S) generate two sorted lists: L_1 (in the upper hull) and L_2 (in the lower hull). Likewise, generate two sorted lists L_3 and L_4 from CH(T). Merge the four lists.

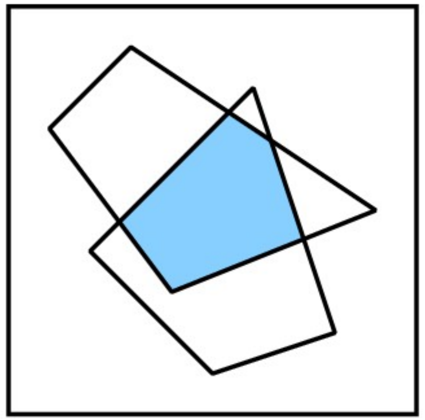
Run Graham's scan on the merged list.

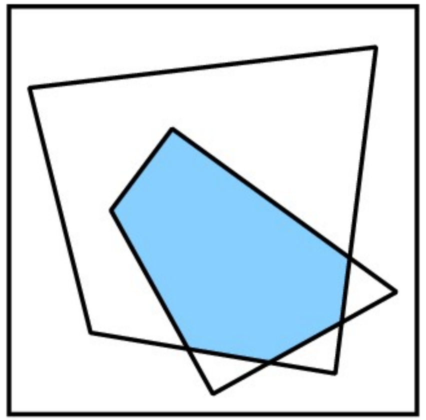

(a) No intersection

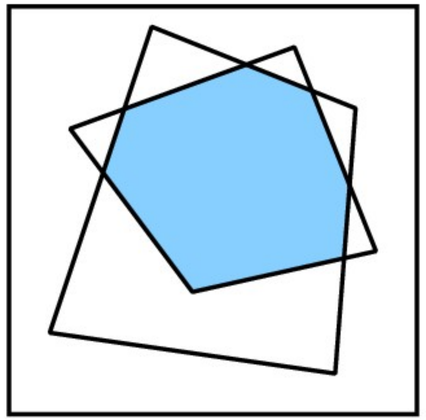

(b) One inside other

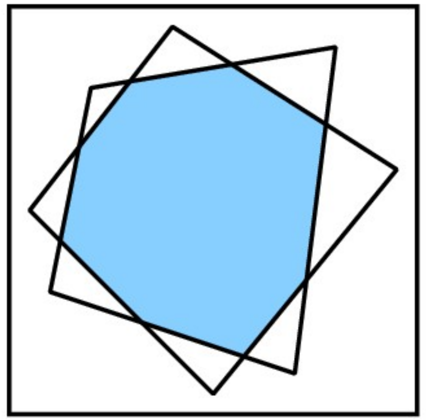

(c) Three sides


(d) Four sides


(e) Five sides


(f) Six sides


(g) Seven sides

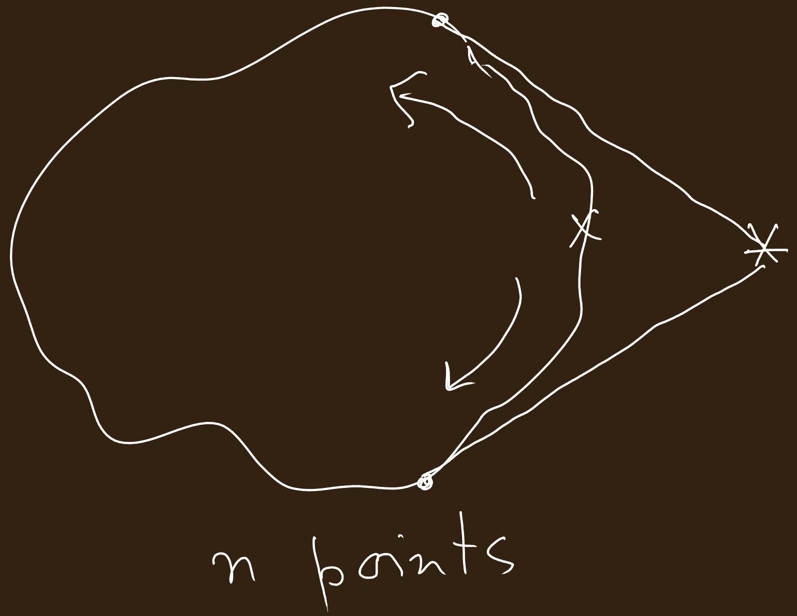

(h) Eight sides

(b) Prove that the problem of Part (a) cannot be solved in $o(n)$ time in the worst case.

If we have such an algorithm for merging, we have a convex-hull algorithm with running time

$T(n) = 2\,T(n/2) + o(n) = o(n \log n)$, a contradiction.

4. Suppose that the points are coming in increasing x-coordinates. How can you convert a hull of n points to a hull of n+1 points? Running time?



Find the two tangents, and discard the points between the two points of tangency.

O(n)

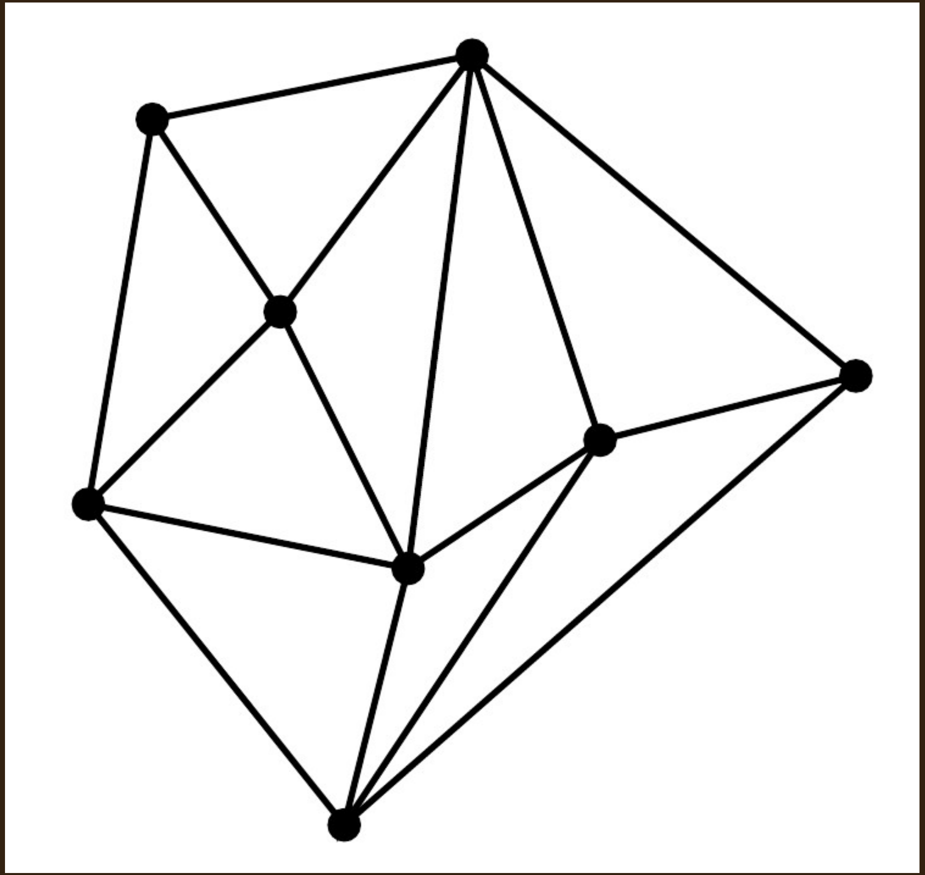
5. [Incremental Convex-Hull construction] In the construction of CH(S), we first sort the points in increasing x-coordinates. Then we apply the previous construction by introducing one point at a time. What will be the overall running time?

Initial sorting takes O(n log n) time.

The remaining work takes O(n^2) time. But this bound is not tight. The amortized running time is O(n).

So the overall running time is again O(n log n).

6. (a) Prove that the number e of edges in any triangulation of CH(S) with |S| = n satisfies $2n - 3 \leq e \leq 3n - 6$.



Let t be the number of triangles, and h the number of vertices on CH(S). By Euler's formula, we have

$$n - e + (t + 1) = 2.$$

Each edge belongs to two faces, so

$$2e = 3t + h.$$

Eliminate t. Then use $3 \leqslant h \leqslant n$.

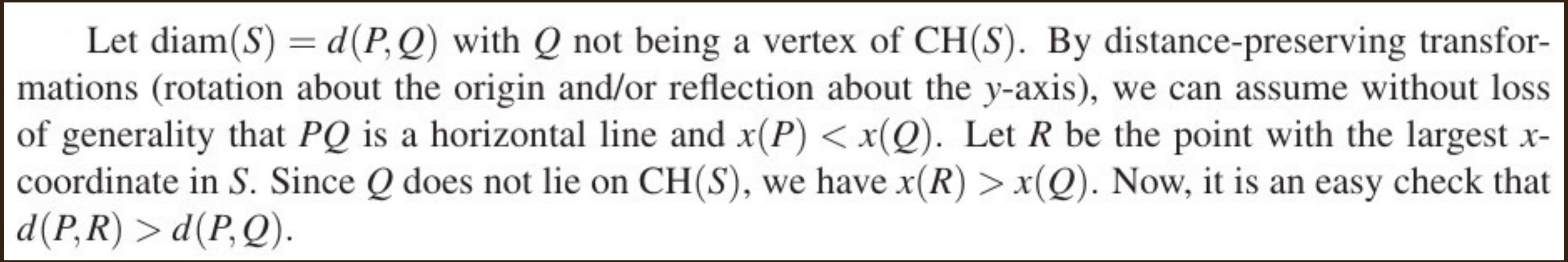
(b) How can you use the incremental convex-hull construction to triangulate CH(S)? Running time?

Join the new point with the points of tangency, and also with all the discarded points.

Initial sorting: O(n log n)
Triangulation: O(n) amortized time

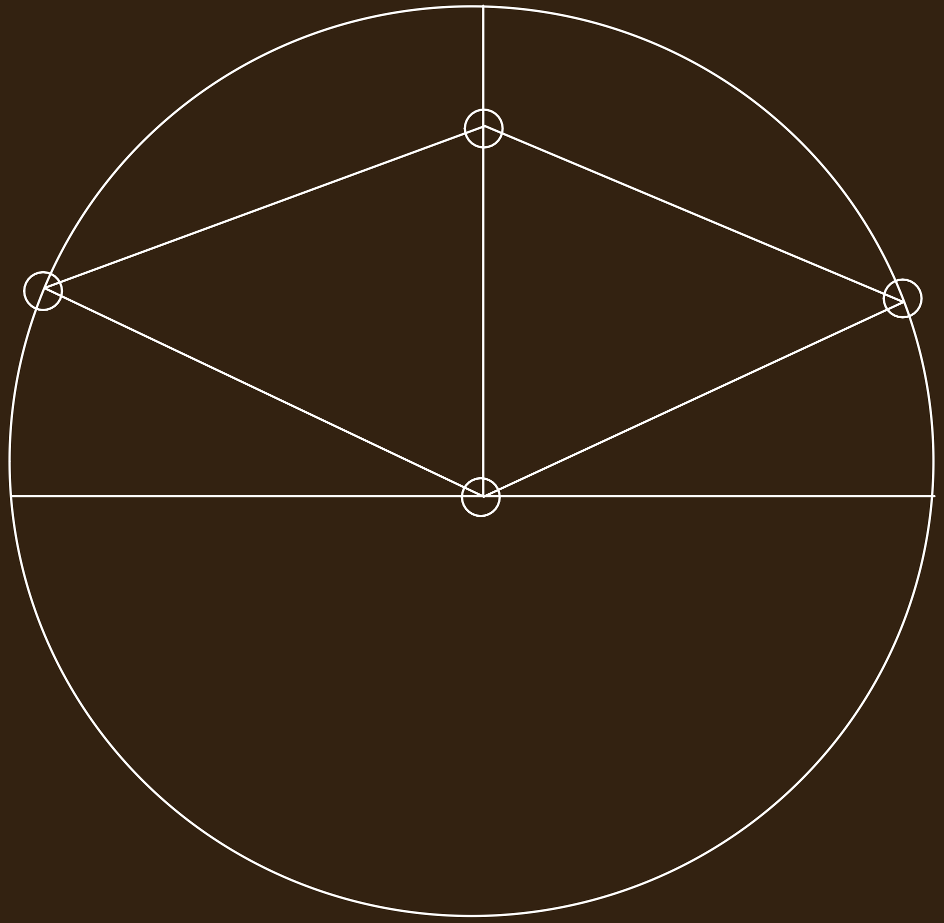
7. [Farthest Pair] Let S be a set of n points in general position in the plane. We want to find P,Q ϵ S such that d(P,Q) is the maximum.

(a) Prove that P and Q are vertices of CH(S).



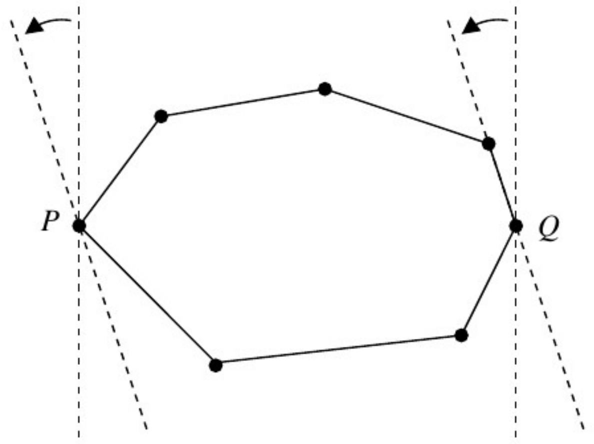
Let $\text{diam}(S) = d(P,Q)$ with Q not being a vertex of $\text{CH}(S)$. By distance-preserving transformations (rotation about the origin and/or reflection about the y-axis), we can assume without loss of generality that PQ is a horizontal line and $x(P) < x(Q)$. Let R be the point with the largest x-coordinate in S. Since Q does not lie on $\text{CH}(S)$, we have $x(R) > x(Q)$. Now, it is an easy check that $d(P,R) > d(P,Q)$.

(b) Let P be a point on CH(S). Demonstrate that the distances of the points on CH(S) from P need not be unimodal.



(c) Let Q,Q' be consecutive vertices on CH(S). Let L be the line QQ'. The perpendicular distances of the vertices of CH(S) from L are unimodal. Let P be the farthest point from L. Build a collection C of pairs (P,Q) and (P,Q'). Prove that the farthest pair (P,Q) can be found in C.

Let $\operatorname{diam}(S) = d(P,Q)$. We can assume, without loss of generality, that PQ is horizontal, and $x(P) < x(Q)$. But then, P (resp. Q) is the point in S with the minimum (resp. maximum) x-coordinate. Therefore, $\mathrm{CH}(S)$ lies completely between two vertical lines passing through P and Q. Rotate these two lines simulltaneously (clockwise or counter-clockwise) until one of these lines coincide with an (the first) edge of $\mathrm{CH}(S)$. This procedure is demonstrated in the following figure. It is now an easy check that $(P,Q) \in C$ (in the figure, for example, we have $Q = Q_{i+1}$ and P is a farthest Q_k from the line L_i passing through Q_i and Q_{i+1}).



(d) If the points in S are in general position, what is the maximum size of C?

2n

(e) Propose an O(n log n)-time algorithm for computing the farthest pair in S.

For each edge on CH(S), the vertices most distant from the edge can be obtained by binary search in the unimodal sequence of distances. We do not need to compute the entire sequence, but compute only those distances that are needed during the search.