

COMP 4045 – Solution examples

Question 3 - Assignment 1: Any triangulation T of a polygon P with n vertices and h polygonal holes has $n - 2h - 2$ triangles.

PROOF: We will prove the theorem by induction on the number of holes of P .

Base case: If $h = 0$ then $|T| = n - 2 = n - 2h - 2$ and the statement holds.

Induction hypothesis: Assume that the statement is true for any polygon P with $k - 1$ holes.

Induction step: Prove that the statement is true for any polygon P with $h = k$ holes. Consider any polygonal hole H_1 of P , and let q be the rightmost vertex of H_1 , as illustrated in Fig. 1. Let r be any vertex of P that can be seen by q and lies to the right of q . Before we continue we prove the existence of r .

Consider the edge e_1 of P that can be seen by q and that lies vertically above q . Let t be the point on e_1 vertically above q . Slide t to the right along e_1 until:

1. t has reached the right endpoint of e_1 , or
2. t cannot be seen from q .

The two cases are illustrated in Fig. 1a and b. In the first case the existence of the r is trivial if we set r to be the right endpoint of e_1 . In the second case it holds that the segment (q, t) must pass through a vertex v of P , otherwise t could be moved further along e_1 . In this case the existence of r follows if we set r to be the vertex v , thus we have shown the existence of r .

Add the diagonal (q, r) to P . By splitting (q, r) into two edges and splitting q and r into two vertices each, it is easy to see that one can obtain a polygon P' with $k - 1$ holes and $n + 2$ vertices. According to the induction hypothesis it holds that any triangulation T' of P' has $(n + 2) - 2(k - 1) - 2 = n - 2k - 2$ triangles.

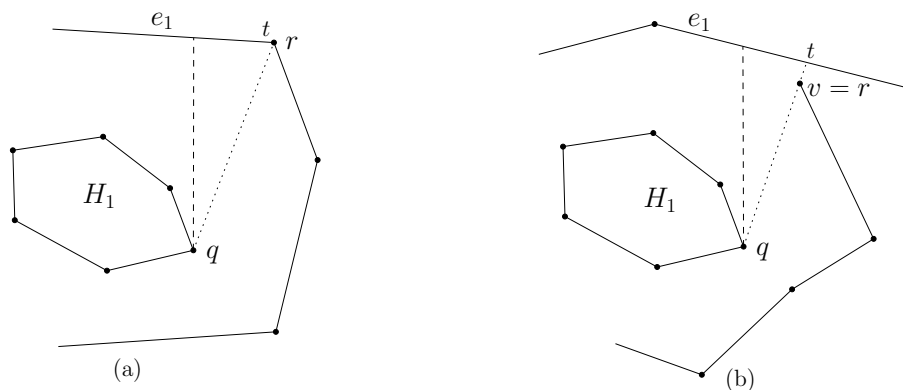


Figure 1:

Question 6 - Assignment 2:

The dual of triangulation can be viewed as a binary tree T . Select an arbitrary leaf of T to be the root. Let v be an arbitrary node of T and let $T(v)$ be

the subtree of T rooted at v . One can preprocess T in $O(n)$ time such that every node v stores the size $T(v)$, denoted $s(v)$. Consider the subpolygon $P(v)$ of P corresponding to $T(v)$. It is not hard to see that $P(v)$ has size $s(v) + 2$. As a result it holds that if there exists a diagonal of the triangulation that partitions P into two subpolygons of similar size then we can easily find that diagonal in $O(n)$ time. Thus it only remains to show the existence of such a diagonal.

Consider the following approach. Start from the root of T and walk down the tree. In each step choose the child with the largest subtree. The set of vertices traversed starting from the root are denoted v_0, v_1, v_2, \dots . Stop at the first vertex v_k for which it holds that $s(v_k) \leq \lfloor 2n/3 \rfloor$.

Let d be the diagonal of P that corresponds to the edge between v_k and v_{k-1} . Note that d partitions P into two subpolygons that we denote $P(v_k)$ and P' . If $s(v_k) \geq \lceil n/3 \rceil$ then we are done since P' will contain at most $(n-2) - (\lceil n/3 \rceil) + 2 \leq 2n/3$ vertices. Could it be that $s(v_k) < \lceil n/3 \rceil$, i.e., $s(v_k) \leq \lceil n/3 \rceil - 1$? The answer is 'No' and the reason is that $s(v_{k-1}) \geq \lfloor 2n/3 \rfloor + 1$. Thus v_{k-1} must have two children, v_k and a vertex v' . It holds that $s(v') + s(v_k) = s(v_{k-1}) - 1 \geq \lfloor 2n/3 \rfloor$ thus $s(v') \geq \lceil n/3 \rceil$. But this cannot be true since $s(v_k)$ must be greater than or equal to $s(v')$, otherwise we would have chosen v' as v_k .

As a result we know that $s(v_k) \geq \lceil n/3 \rceil$ thus this completes the proof, since the diagonal corresponding to the edge (v_{k-1}, v_k) in T is a "valid" diagonal.

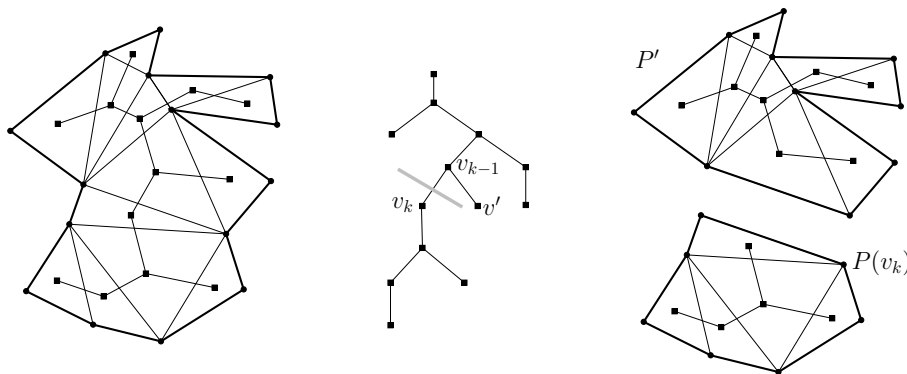


Figure 2:

Question 5 - Assignment 2:

A rectangle r in 2D can be described by four coordinates in 4D (x_1, x_2, x_3, x_4) , where x_1, x_2, x_3 and x_4 is the left, right, bottom and top coordinates of r .

Building a kd-tree T for a 4-dimensional point set can be done in $O(n \log n)$ time using $O(n)$ space. Note also that a standard rectangular query in T can be answered in $O(n^{3/4} + k)$ time.

How do we transform a point query among rectangles in 2D into a rectangular query among points in 4D? To build up an intuition about the problem we

consider the transformation from 1D to 2D, see Fig. 3. Every 1-dimensional rectangle becomes a point in 2D (left endpoint, right endpoint). Consider a point query q in 1D. All rectangles with left endpoint to the left of q and right endpoint to the right of q should be reported. All points to the left of q in 1D corresponds to all the points in 2D whose x -coordinate is smaller than q , and all points to the right of q in 1D corresponds to all the points in 2D whose y -coordinate is greater than q . As a result we get the query rectangle with lower left corner at $(0, q)$ and upper right corner at (q, ∞) .

The transformation generalises to higher dimensions. If the query point q in 2D has coordinates (q_x, q_y) then the query rectangle in 4D is defined by the two points $(0, q_x, 0, q_y)$ and $(q_x, \infty, q_y, \infty)$. According to Chapter 5 in the course book by de Berg et al., an orthogonal range query can be answered in time $O(n^{3/4} + k)$, where k is the number of reported points.

To prove the correctness of the algorithm and its complexity it remains to prove that the every reported rectangle is intersected by the query point and that no other rectangle is intersected by q . This is left as an exercise :-)

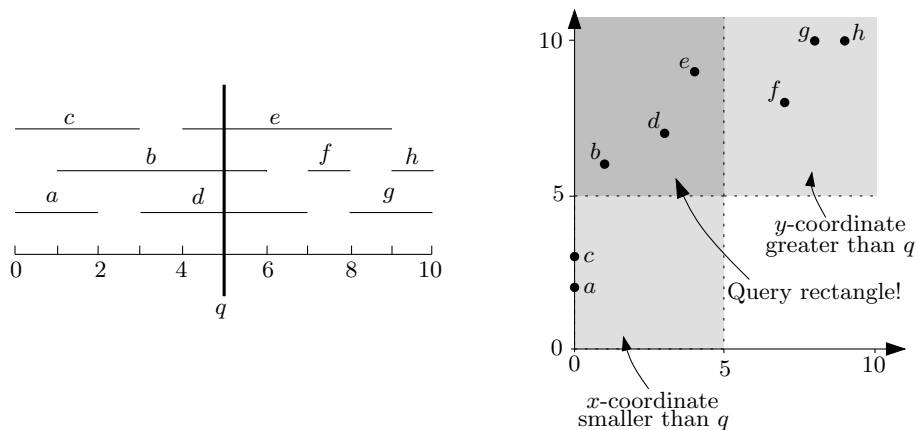


Figure 3: .