Due: 15th of April 2010 at 10am

Please submit as PDF by email to `thomas.wolle@nicta.com.au`

As a first step go to `http://gudmundsson.biz/comp5045/course_information.htm` and read the sections: **Advice on how to do the home work** and **Academic Dishonesty**.

COMP 5045 – Assignment 1

1. Let E be an unordered set of n line segments that are the edges of a convex polygon. Describe an $O(n \log n)$ time algorithm that computes from E a list containing all the vertices of the convex polygon in clockwise, or counter-clockwise, order. Prove the correctness and the running time of your algorithm. [10 points]

2. (a) A rectilinear polygon is a simple polygon of which all edges are horizontal or vertical. Let P be a rectilinear polygon with n vertices. Show that for any n there is a rectilinear polygon such that $\lfloor n/4 \rfloor$ guards are necessary to guard it. [3 points]

   (b) Give a simple polygon P and a placement of guards, such that the guards see every point of the boundary of P, but there is at least one point interior to P not seen by any gurad. [3 points]

   (c) A watchman route is a closed walk in P such that every point on the boundary of P is visible from some point of the walk. Construct a polygon P that may contain polygonal holes and a watchman route such that the watchman sees every point of the boundary of P, but there is at least one point interior to P not seen by the watchman. [4 points]

3. A simple polygon P is called *star-shaped* if it contains a point q, such that for any point p in P, the straight line segment pq is contained in P. Give an algorithm whose expected running time is linear to decide whether a simple polygon is star-shaped. [10 points]

4. You are given two x-monotone polygonal chains P and Q. Prove that the number of times P and Q can intersect is $O(n)$, where n is the total number of vertices of P and Q, see Figure 1(a). [10 points]

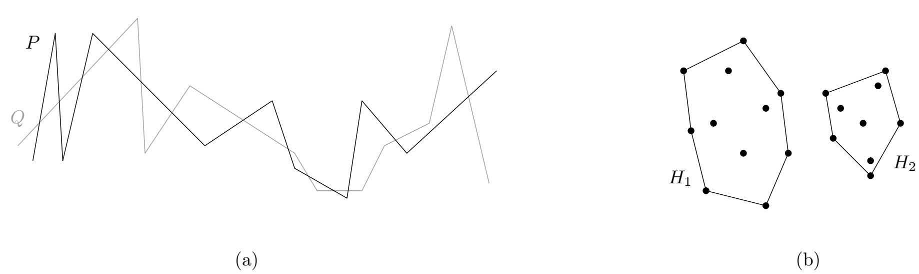

Figure 1: (a) Two x-monotone polygonal chains P and Q; (b) Illustrating the convex hulls H_1 and H_2

5. Consider the algorithm for reporting all intersections between line segments as given in the lecture.

   (a) In a straight forward analysis, we observed that there are in total $O(n + k)$ events, which will be stored in a queue Q during the algorithm. Each of those events takes $O(\log |Q|)$ time to handle. Hence, the algorithm runs in $O((n + k) \log(n + k))$ time. Prove that the algorithm, as given in the lecture, actually runs in $O((n+k) \log n)$ time. [2 points]

   (b) The number of events stored in Q at any time influences the running time and the space needed by the algorithm. In the algorithm given in the lecture, this number is $O(n+k)$. Your task is to modify this algorithm, such that the number of events stored in Q at any time is at most $O(n)$. (Note: This does not change the total number of events. Hint: During the algorithm, you might want to delete certain events from Q and add them later on again.) [5 points]

   (c) As mentioned in the lecture, during the sweep some line segments are intersected by the current sweep line ℓ. The ordered sequence of these line segments can be stored in a balanced binary search tree T. Consider the situation, where the sweep line ℓ encounters a new segment s. Explain how to find the position in T where s has to be inserted. What exactly has to be stored in such a tree? [3 points]

6. Consider the following algorithm to compute the convex hull of a set S of n points in the plane, as illustrated in Figure 1(b).

   Step 1: Sort the points in S by increasing x-coordinate.

   Step 2: Recursively compute the convex hull of the left half of the point set. The resulting convex hull is denoted H_1.

   Step 3: Recursively compute the convex hull of the right half of the point set. The resulting convex hull is denoted H_2.

   Step 4: From H_1 and H_2 compute the convex hull H of the entire point set.

   (a) Consider the edge e connecting the highest point in H_1 with the highest point in H_2. Will the edge e be an edge in H? Prove your answer. [3 points]

   (b) Consider the points clockwise along H_1 between the highest point of H_1 to the bottom point of H_1. Can any of these points be in the convex hull, H, of the entire set? Prove your answer. [2 points]

   (c) Give a correct implementation of step 4, that runs in $O(n)$ time. Prove the correctness and the running time of your algorithm. [5 points]