Karlsruher Institut für Technologie
Fakultät für Informatik
ITI Wagner

Computational Geometry
Summer term 2018

Exercises 1 - Convex Hulls & Line Segment Intersection

Discussion: Friday, May 4th, 2018

Lecture 1 – 18.04.2018

Exercise 1. Show that the last step of the algorithm FIRSTCONVEXHULL(P) can be implemented such that is has running time $O(n \log n)$.

```
FirstConvexHull(P)
  E ← ∅
  foreach (p, q) ∈ P × P with p ≠ q do
      valid ← true
      foreach r ∈ P do
          if not (r strictly right of →pq or r ∈ ‾pq) then
              valid ← false
      if valid then
          E ← E ∪ {(p, q)}
  Construct sorted node list L of CH(P) from E
  return L
```

Exercise 2. In the first lecture, we have seen the *gift wrapping* or *Jarvis march* algorithm for computing the convex hull of a set of n points. Sketch a proof of the following theorem, in particular the correctness of the algorithm.

Theorem 1. *The convex hull $CH(P)$ of a set of n points P in $\mathbb{R}^2$ can be computed in $O(n \cdot h)$ time using the gift wrapping algorithm, where $h = |CH(P)|$.*

What degeneracies may occur in the input? How can you handle them correctly?

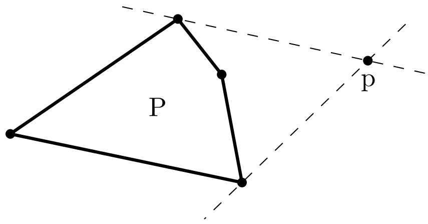

Figure 1: Convex polygon P and the two tangents from an external point p.

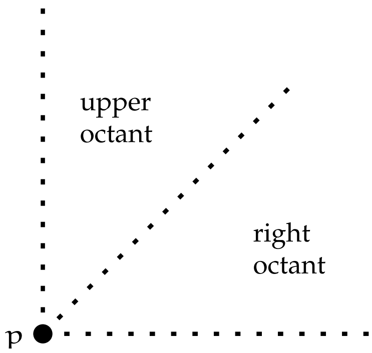

Figure 2: Largest top-right region of p

Exercise 3. Let P be a convex polygon with n vertices and let p be a point outside P as shown in Figure 1.

Show that the two tangent lines at P that pass through p can be computed in $O(\log n)$ time, assuming that the polygon is given as a clockwise sorted list of its vertices. In which part of Chan's convex hull algorithm is this subroutine needed?

Exercise 4. We require that any algorithm computing the convex hull of a given set of points returns the convex hull vertices as a clockwise sorted list of points.

1. Show that any algorithm for computing the convex hull of n points has a worst case running time of $\Omega(n \log n)$ and thus *Graham Scan* is worst-case optimal.
2. Why is the running time of the *gift wrapping* algorithm not in contradiction to part (a)?

Lecture 2 – 25.04.2018

Exercise 5. The algorithm seen in the lecture for finding the intersection points of n line segments required $O(n + I)$ space, where I is the number of intersection points. Modify the algorithm to use $O(n)$ space only and argue that the modified algorithm remains correct. Does this affect the asymptotic running time?

Exercise 6. Let P be a set of n points in the plane. The *largest top-right region* of a point $p \in P$ is the union of all open axis-aligned squares that touch p with their bottom left corner and contain no other point of P in their interior.

(1) Prove that the largest top-right region of a point is either a square or the intersection of two open half-planes.
(2) Let p be a point in P, let O_1 be the upper octant and O_2 be the right octant of p; see Fig. 2. Which point in $O_1 \cap P$ restricts the largest top-right region of p at most? Which point in $O_2 \cap P$ restricts the largest top-right region of p at most?
(3) Describe an algorithm that computes for all points in P the largest top-right region using $O(n \log n)$ running time in total.
Hint: Determine the points of question (2) using two sweeps.