

CS 252B Assignment 6

Due at start of class on Tuesday, April 12th

1. *Angular sweep.* Let C be a circle centered at the origin that has n cords (line segments whose endpoints lie on C). Assume that no two cords share a common endpoint. Find an “angular sweep” algorithm that computes the total number of intersections between the cords. For example, if all cords intersect in the middle, the correct answer is $\binom{n}{2}$. Your algorithm has to run in (worst-case) time $O(n \log n)$. By angular sweep, I mean that the sweep line should be a ray starting at the origin and extending to infinity along, say, the x -axis, and the sweep should be a rotation of the sweep line around the origin.
2. *Line intersections.* Start with a set of n horizontal lines and m vertical lines in the plane. Describe (without too many details) a plane sweep algorithm that finds all intersections between those lines. (You can assume that no two horizontal lines overlap, and similarly for vertical lines.) This is very similar to what we did in class with rectangles, but simpler. In particular, you don’t need to use interval trees. Explain how you would need to augment a red-black tree (if at all) to make your algorithm work, and give pseudo-code for the function `All-Lines-Intersection(T,I)` which, given a vertical interval I , returns all y -coordinates in T that are contained in I (i.e. all horizontal lines in T that intersect the vertical line I).
3. On n parallel railway tracks n trains are going with constant speeds v_1, v_2, \dots, v_n . At time $t = 0$ the trains are at positions $k_1, k_2 \dots k_n$. Give an $O(n \log n)$ algorithm that detects all trains that at some moment in time are leading. To this end, use the algorithm for computing the intersection of half-planes.
4. *Constructive proofs are more work than proofs by contradiction, but they actually show you how to construct a solution!* Let’s revisit lemma 4.5, on page 73 of the Computational Geometry book by Berg et al. (The relevant chapter will be handed out on April 7.) Assume for simplicity that ℓ_i is the x -axis. Let $\vec{v} = (1, 0)$ be the usual vector along the x -axis, and for $1 \leq j \leq i$ let \vec{c}_j, h_j, C_j and v_j be as defined in class (and in the handout). Again, let x_j be the x -coordinate of the point where ℓ_i and ℓ_j intersect, and let

$$\begin{aligned}x_{\text{left}} &= \max_{1 \leq j < i} \{x_j \mid \ell_i \cap h_j \text{ is bounded to the left}\} \\x_{\text{right}} &= \min_{1 \leq j < i} \{x_j \mid \ell_i \cap h_j \text{ is bounded to the right}\}\end{aligned}$$

We want to restate part (ii) of the lemma so that it directly gives us our algorithm for 2DLP. Here is what you need to prove (proofs by contradiction are not allowed!) :

If $v_{i-1} \notin h_i$, then either $C_i \neq \emptyset$ or

$$v_i = \begin{cases} (x_{\text{left}}, 0) & \text{if } \vec{c} \cdot \vec{v} < 0 \\ (x_{\text{right}}, 0) & \text{if } \vec{c} \cdot \vec{v} > 0 \end{cases}$$

Note that $\vec{c} \cdot \vec{v} > 0$ just means that $f_{\vec{c}}(p)$ is increasing as p moves to the right along the x -axis. For simplicity, you can assume that \vec{c} is not perpendicular to the x -axis (that is, $\vec{c} \cdot \vec{v} \neq 0$).

5. *Closest-point heuristic for approximating ΔTSP .* Exercise 35.2-3, on page 1033 of CLRS.