CMSC 754: Spring 2020 Dave Mount

Homework 4: Arrangements and Approximations

Handed out Fri, May 1. Due at the start of class on Tuesday, May 12. Late homeworks are not accepted (unless an extension has been prearranged) so please turn in whatever you have completed by the due date. Unless otherwise specified, you may assume that all inputs are in *general position*. Whenever asked to give an algorithm running in $O(f(n))$ time, you may give a *randomized algorithm* whose expected running time is $O(f(n))$.

Problem 1. Given a set P of n points in $\mathbb{R}^2$ in general position, and given a nonvertical line ℓ, project the points orthogonally onto ℓ and consider their left-to-right order (see Fig. 1). The result is called an *allowable permutation*. (Let us assume that ℓ is chosen so no two points have the same projection.)

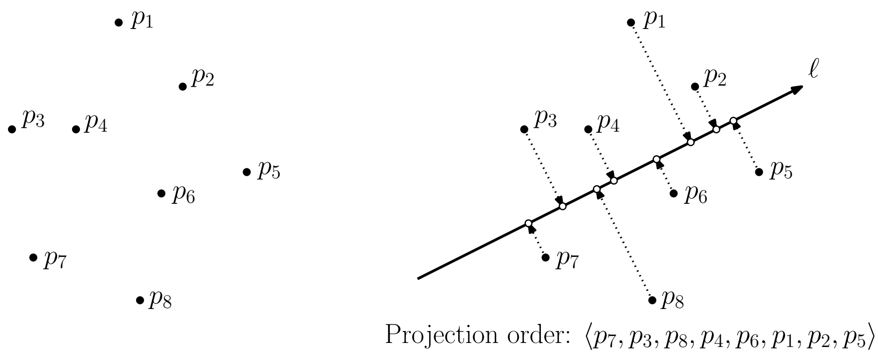

Figure 1: Allowable permutation

In general there are an exponential number ($n!$) of distinct permutations of n points. Prove that the number of allowable permutations is only $O(n^2)$. (Hint: Explain how each allowable permutation is manifested in the dual arrangement.)

Problem 2. You are given three sets of points R, G, and B (red, green, and blue) in $\mathbb{R}^2$. A *tricolor slab* is a pair of parallel lines such that the closed region bounded between these two lines contains at least one point from each of R, G, and B. Define the slab's *height* to be the vertical distance between these lines (see Fig. 2).

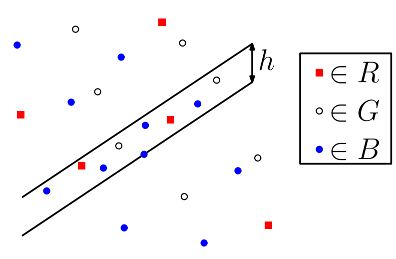

Figure 2: Tricolor slab of height h.

(a) There are infinitely many a tricolor slabs. Assuming that the point set $R \cup G \cup B$ is in general position, prove that a tricolor slab of minimum height has a point of each color on its boundary, with two points on one line and one point on the other.

(b) Assuming the standard dual transformation (mapping point (a, b) to line $y = ax - b$), explain what a tricolor slab of minimum vertical height h corresponds to in the dual setting.

(c) Present an algorithm, which given inputs R, G, and B, computes the tricolor slab of minimum vertical height. Your algorithm should run in $O(n^2 \log n)$ time, where $n = |R| + |G| + |B|$. Derive your algorithm's running time and justify its correctness.

Problem 3. You are given two sets of points in $\mathbb{R}^d$, called R (for red) and B (for blue). A *bichromatic pair* is any pair of points (p, q), where $p \in R$ and $q \in B$. The *bichromatic diameter* is defined to be the bichromatic pair of maximum distance: $\max_{p \in R, q \in B} \|p - q\|$.

Given $\varepsilon > 0$, an *ε-approximation* to the bichromatic diameter is a pair $p' \in R$ and $q' \in B$, such that

$$\frac{\|p - q\|}{1 + \varepsilon} \;\leq\; \|p' - q'\| \;\leq\; \|p - q\|,$$

where p and q are the true bichromatic diameter.

Present an efficient algorithm which, given R, B, and $\varepsilon > 0$, computes an ε-approximation to the bichromatic diameter. Your algorithm should run in time $O(n \log n + n/\varepsilon^d)$ time, where $n = |R| + |B|$. Derive your algorithm's running time and justify its correctness.

Problem 4. You are given a set $P = \{p_1, \ldots, p_n\}$ of n points in $\mathbb{R}^d$. Given a positive real r, the *proximity index*, denoted $\Psi_P(r)$ is the number of pairs of points of P that lie within distance r of each other. (Formally, $\Psi_P(r)$ is the number of pairs $1 \leq i < j \leq n$ such that $\|p_i - p_j\| \leq r$.)

Given P and r, we can easily compute $\Psi_P(r)$ in $O(n^2)$ time by inspecting all pairs of points, but since n is very large, this is too slow. Given a constant $\varepsilon > 0$, an *ε-approximation* to $\Psi_P(r)$ is an integer X from 0 to $\binom{n}{2}$, such that

$$\Psi_P\left(\frac{r}{1+\epsilon}\right) \;\leq\; X \;\leq\; \Psi_P((1+\epsilon)r).$$

Present an algorithm running in time $O(n \log n + n/\epsilon^d)$ which, given P, r, and ε, computes such an approximation. (Hint: You may assume that, when a quadtree is computed, each node u of the quadtree can be associated with an integer $\mathrm{wt}(u)$, which indicates the number of points of P lying within u's subtree.)

Explain your algorithm's running time and justify its correctness.

Challenge problems count for extra credit points. These additional points are factored in only after the final cutoffs have been set, and can only increase your final grade.

Challenge Problem 1. Modify your solution to Problem 2 (tricolor slab), so that instead of minimizing the vertical distance between the lines of the slab, it minimizes the *perpendicular distance* between these lines. The running time should be the same.

Challenge Problem 2. You are given two sets of points B and R (called, blue and red, respectively) in $\mathbb{R}^2$, each of size n. Give an $O(n^2 \log n)$ time algorithm which determines whether there exists a line ℓ such that the orthogonal projections of the points of $B \cup R$ onto this line alternate between blue and red (see Fig. 3). (It does not matter which color the sequence begins or ends with, just that there is no consecutive red-red or blue-blue in the projected order.)

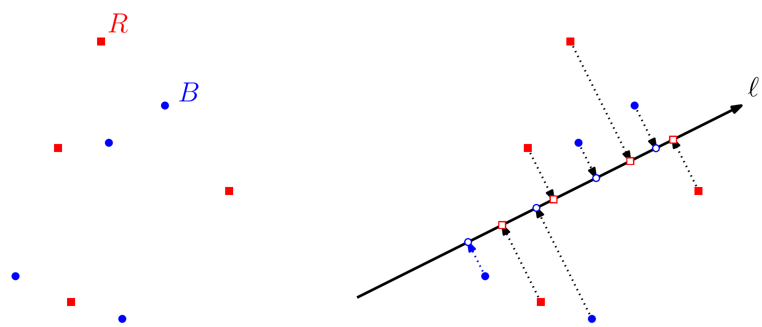

Figure 3: Red-blue alternating projection