OPTIMAL SOLUTIONS FOR A CLASS OF POINT RETRIEVAL PROBLEMS

Bernard Chazelle and Herbert Edelsbrunner

Brown Univ. and Techn. Univ. Graz

Abstract

Let $P$ be a set of $n$ points in the Euclidean plane and let $C$ be a convex figure. We study the problem of preprocessing $P$ so that for any query point $q$, the points of $P$ in $C + q$ can be retrieved efficiently. If constant time suffices for deciding the inclusion of a point in $C$, we then demonstrate the existence of an optimal solution: the algorithm requires $O(n)$ space and $O(k + \log n)$ time for a query with output size $k$. If $C$ is a disk, the problem becomes the well-known fixed radius neighbor problem, to which we thus provide the first known optimal solution.

1. Introduction

Let $P$ be a set of $n$ points in the Euclidean plane $E^2$, and let $C$ be a convex figure. We study the complexity of the following problem: preprocess $P$ so that for any query translate $C_q = C + q$ of $C$ the points in $P \cap C_q$ can be retrieved efficiently. Intuitively, a query corresponds to an arbitrary displacement of $C$ without rotation. We demonstrate the existence of a solution that is optimal in space and time, provided that $C$ satisfies certain weak computational conditions. Specifically, we describe a data structure that requires $O(n)$ space and $O(k + \log n)$ time to answer a query with output size $k$. The only assumption necessary to the validity of the algorithm is that constant time suffices for deciding whether a point is contained in $C$. A few other primitive operations must be assumed for the sake of preprocessing. If such operations can be executed in constant time, the data structure can be constructed in $O(n^2)$ time.

The generality of the setting allows a uniform solution of several problems which have been treated separately in the past. If $C$ is a disk, the problem becomes the well-known fixed-radius neighbor problem [BM,C1,CCPY]. The best solution to this problem achieves optimal retrieval time at the cost of $O(n(\log n \log \log n)^2)$ space [CCPY], but also handles queries with non-fixed radius. If $C$ is a triangle or a rectangle then we have restricted versions of the triangular and orthogonal range search problems.

Authors' current address: B. Chazelle: Dept. of Computer Science, Box 1910, Brown Univ., Providence, RI 02912, USA. H. Edelsbrunner: Inst. Inf. Proc., Techn. Univ. Graz, Schießstattg. 4a, A-8010, Graz, Austria. The first author was supported in part by NSF grant MCS 83-03925.
We look at the special case where \( C \) is a convex \( m \)-gon and \( m \) is considered a variable of the problem. For this case, we describe a solution requiring \( O(n + m) \) space and \( O(k + \log n \log m) \) time to compute a \( k \)-point answer.

2. The Geometric and Computational Backdrop

In this section we introduce relevant geometric notions and address the computational assumptions we have to make.

Let \( \mathbb{E}^2 \) denote the Euclidean plane and endow it with a system of Cartesian coordinates \( x \) and \( y \). The directions determined by the \( x \) and \( y \) axes are referred to as horizontal and vertical, respectively. Let \( A \) be a subset of \( \mathbb{E}^2 \). We assume that the reader is familiar with the concepts of interior \( \text{int} \ A \), closure \( \text{cl} \ A \) and boundary \( \partial A \). For two points \( a = (a_x, a_y) \) and \( b = (b_x, b_y) \), we have \( a + b = (a_x + b_x, a_y + b_y) \), and for a real \( \lambda \), \( \lambda a = (\lambda a_x, \lambda a_y) \). These operations are naturally extended to subsets \( A, B \) of \( \mathbb{E}^2 \), i.e. \( A + B = \{ a + b \mid a \in A, b \in B \} \) and \( \lambda A = \{ \lambda a \mid a \in A \} \). For any point \( q \), \( A + q = A + \{ q \} \) is called a translate of \( A \) and is denoted \( A_q \). A is convex if for any points \( a_1 \) and \( a_2 \) in \( A \), the point \( \lambda a_1 + (1 - \lambda)a_2 \) lies in \( A \) for each \( \lambda \) such that \( 0 \leq \lambda \leq 1 \). The smallest convex set that contains \( A \) is called the convex hull of \( A \), denoted \( \text{conv} A \). The convex hull of \( \{ a, b \} \) is called a segment.

The model of computation is the standard RAM with infinite real arithmetic — a traditional assumption in computational geometry. Let \( C \) be a convex closed figure with non-empty interior. We leave \( C \) essentially unspecified and therefore must make a minimum number of assumptions on the primitive operations allowed with respect to \( C \). First, we consider the intersection of the boundaries of two translates of \( C \). We define \( S(v, w) = \partial(-C)_v \cap \partial(-C)_w \), for two points \( v \) and \( w \) in \( \mathbb{E}^2 \). By convexity of \( C \), \( S(v, w) \) is either empty or consists of at most two possibly degenerate segments, and thus can be represented in a constant amount of space. This concept is naturally extended to the case where \( v \) and \( w \) are infinitesimally close: this gives \( S(v, w) = S(v, l) = \partial(-C)_v \cap \partial(-C)_{l + 1} \), for \( l \) the line that contains \( v \) and \( w \) (see Fig. 1 for an illustration of the two cases).

We call \( C \) computable if (i) constant time suffices to test for any point \( p \) in \( \mathbb{E}^2 \) whether or not \( p \) is contained in \( C \), and (ii) constant time suffices to compute \( S(v, w) \) for any two, potentially infinitesimally close, points \( v \) and \( w \) in \( \mathbb{E}^2 \). In the remainder of this section, we elaborate on the primitive operations needed and introduce the notion of silo. Let \( L \) (resp. \( R \)) in \( \partial C \) be the point with minimal (resp. maximal) \( z \)-coordinate, and maximal \( y \)-coordinate if not unique.

Lemma 1. If \( C \) is computable, \( L \) and \( R \) can be determined in constant time.

Proof: Let \( l \) be the vertical line through the origin \( O = (0, 0) \). Since \( C \) is computable, \( S(O, l) \) and the lower endpoints \( a \) and \( b \) of the two vertical segments that constitute \( S(O, l) \) can be determined in constant time. Let \( a_x < b_x \); we then have \( L = -b \) and \( R = -a \).