Information Processing Letters 113 (2013) 518–521

Contents lists available at SciVerse ScienceDirect

Information Processing Letters

www.elsevier.com/locate/ipl

A note on searching line arrangements and applications

Danny Z. Chen [a,1], Haitao Wang [b,*]

[a] Department of Computer Science and Engineering, University of Notre Dame, Notre Dame, IN 46556, USA
[b] Department of Computer Science, Utah State University, Logan, UT 84322, USA

ARTICLE INFO

Article history:
Received 27 January 2012
Received in revised form 19 April 2013
Accepted 20 April 2013
Available online 23 April 2013
Communicated by F.Y.L. Chin

Keywords:
Line arrangement
Vertex searching
Points approximation
Computational geometry
Algorithm design

ABSTRACT

We study the problem of searching for a vertex with a desired property in the arrangement of a set of lines in the plane. We show that this problem can be solved efficiently by modifying (and simplifying) two slope selection algorithms without using parametric search. We apply this result to a points approximation problem and obtain an optimal solution for it without using parametric search. Since this line arrangement searching problem is quite natural, our result may find other applications as well.

1. Introduction

We consider the problem of searching for a vertex with a desired property in an arrangement of lines in the plane, called the *line arrangement searching problem*, defined as follows.

Suppose there is a function $f : \mathbb{R} \to \{0, 1\}$ and f is monotonically increasing, i.e., for any two values $x_1 \leqslant x_2$, $f(x_1) \leqslant f(x_2)$ always holds. Also, suppose the description of the function f is unknown to us; however, given any value x, we have an oracle that can evaluate $f(x)$ in $O(F)$ time, which we call the *f-oracle*. Let L be a set of n lines in the plane, and $\mathcal{A}_L$ denote the arrangement of the lines in L [9]. For simplicity of discussion, we assume without loss of generality that the lines in L are in general position: No two different lines are parallel, no three lines intersect at the same point, and no two distinct pairs of lines intersect at points that have the same x-coordinate. The intersection point of any two lines in L is called a *vertex* of $\mathcal{A}_L$. For any vertex $v \in \mathcal{A}_L$, let $x[v]$ denote the x-coordinate of v. The goal of the line arrangement searching problem is to find the *leftmost* vertex v of $\mathcal{A}_L$ such that $f(x[v]) = 1$, i.e., the vertex v with the smallest x-coordinate such that $f(x[v]) = 1$. We assume there exists at least one vertex v of $\mathcal{A}_L$ such that $f(x[v]) = 1$. Note that due to the monotonicity of the function f, this assumption can be easily verified by first finding the rightmost vertex v' of $\mathcal{A}_L$ (which can be done in $O(n \log n)$ time [9]) and then checking whether $f(x[v']) = 1$ by calling the f-oracle.

Although the line arrangement searching problem appears to be a quite natural problem, we are not aware of any previous result for it. A work with a somewhat similar flavor is in [3], where an oracle (called *touching oracle* [10]) was repeatedly used to determine the upper envelope of a set of unknown lines in the plane. In this paper, we solve the line arrangement searching problem in $O((n + F) \log n)$ time. This result may be particularly interesting when $F = O(n)$ because this likely yields an optimal solution. We achieve our result by modifying two optimal slope selection algorithms [4,16] without using parametric search [7,21]. Further, we show that our result can be applied to solve a points approximation problem optimally and practically in $O(n \log n)$ time (without using parametric search), which answers an open question raised

* Corresponding author.
E-mail addresses: dchen@cse.nd.edu (D.Z. Chen), haitao.wang@usu.edu (H. Wang).

[1] Chen's research was supported in part by NSF under Grants CCF-0916606 and CCF-1217906.

0020-0190/$ – see front matter © 2013 Elsevier B.V. All rights reserved.
http://dx.doi.org/10.1016/j.ipl.2013.04.011

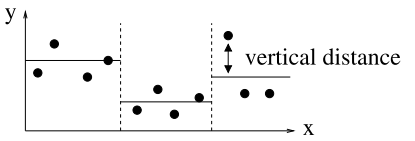

Fig. 1. Illustrating an example of approximating a set of points by a k-step function with $k = 3$.

in [12]. Since this line arrangement searching problem is quite natural, our result may find other applications as well. Below we introduce the points approximation problem.

1.1. The points approximation problem

Given a set $P = \{p_1, p_2, \ldots, p_n\}$ of n points in the plane with each point $p_i = (x_i, y_i)$ having a weight $w_i > 0$, we want to find a functional curve g to approximate P and minimize the approximation error, as defined below. The *vertical distance* between any point $p_i \in P$ and g is defined as $d(p_i, g) = |y_i - g(x_i)|$. The *approximation error* of g is defined as $e(P, g) = \max_{p_i \in P} w_i \cdot d(p_i, g)$. Further, we require g to be a *step function* of k steps for an input integer $k > 0$ (see Fig. 1). In summary, given P and k, the *points approximation problem* aims to compute a step function g of size k to approximate P, minimizing the approximation error.

The points approximation problem has been studied extensively. Guha and Shim [14] gave an $O(n \log n + k^2 \log^6 n)$ time algorithm. Lopez and Mayster [18] solved the problem in $O(n^2)$ time. Later, Fournier and Vigneron [12] (and its preliminary version [11]) presented an $O(n \log^4 n)$ time solution. Subsequently, Chen and Wang [6] gave an algorithm with $O(\min\{n \log^2 n, n \log n + k^2 \log \frac{n}{k} \log n \log \log n\})$ time. A randomized $O(n \log n)$ time algorithm was given by Liu [17]. Recently, Fournier and Vigneron [13] presented an $O(n \log n)$ time deterministic algorithm, which is optimal. However, the algorithm [13] utilizes Cole's parametric search [7], and thus, as discussed in [13], this algorithm is mainly of theoretical interest and unlikely to be practical. An open question was raised in [13] to design a practical $O(n \log n)$ time deterministic algorithm for this problem.

We show that our result for the line arrangement searching problem can be applied to develop an $O(n \log n)$ time deterministic algorithm for this points approximation problem without using parametric search.

The rest of this paper is organized as follows. In Section 2, we modify the two slope selection algorithms in [4,16] to solve the line arrangement searching problem. In Section 3, we discuss the application of our result to the points approximation problem.

2. Algorithms for searching line arrangements

In this section, we solve the line arrangement searching problem in $O((n + F) \log n)$ time by modifying the two slope selection algorithms in [4,16]. In fact, our modified algorithms become simpler than the original slope selection algorithms [4,16].

In the dual setting [9], the *slope selection* problem is to find the k-th leftmost vertex in an arrangement of n lines in the plane for a given integer k. Cole et al. [8] developed the first optimal $O(n \log n)$ time algorithm by using parametric search [7,21] and the AKS sorting network [1], which is fairly complicated. Later, Katz and Sharir [16] gave an $O(n \log n)$ time solution by using expander graphs [2,19]; Bönnimann and Chazelle [4] presented another $O(n \log n)$ time algorithm by using cuttings [5]. Both these algorithms [4,16] avoid using parametric search and are more practical.

In the following, we discuss the slope selection algorithms in [4,16] and our modified algorithms both on the line arrangement $\mathcal{A}_L$. Let v_k denote the vertex of $\mathcal{A}_L$ that is sought by the slope selection problem and let v^* denote the vertex of $\mathcal{A}_L$ that is sought by the line arrangement searching problem.

2.1. The algorithm via expanders

In this section, we modify the slope selection algorithm by Katz and Sharir [16], which we call the KS algorithm, to solve the line arrangement searching problem. Below, we first briefly sketch the KS algorithm and then discuss our modifications.

For two real values l and r with $l < r$, define a *vertical slab* $[l, r)$ as the set of points in the plane whose x-coordinates are at least l and less than r. The KS algorithm [16] has $O(\log n)$ steps. In each step, the algorithm works on a vertical slab $\sigma_j = [l_j, r_j)$ such that the sought vertex v_k is contained in σ_j. Along with σ_j, there is a (partial) partition $\mathcal{P}_j$ of σ_j into a collection of "vertical trapezoids". The output of this step is a new slab σ_{j+1} ($\subseteq \sigma_j$) and a partition $\mathcal{P}_{j+1}$ of σ_{j+1}. The algorithm maintains several algorithmic invariants (e.g., σ_{j+1} contains v_k). Using the expander graphs, the partition $\mathcal{P}_{j+1}$ is obtained by dividing each trapezoid of $\mathcal{P}_j$ into a constant number of smaller trapezoids. Then σ_{j+1} is obtained by narrowing σ_j in the following way. There are $O(n)$ intersections (called *critical points* in [16]) between the edges of the trapezoids in $\mathcal{P}_{j+1}$ and the lines in L. Divide σ_j into a constant number of vertical sub-slabs, each containing at most n critical points. Then, an $O(n)$ time oracle is used to perform a binary search among these sub-slabs, to determine which of them contains v_k. More specifically, given an x-coordinate x', the oracle can determine whether the x-coordinate of v_k is less than x'. The sub-slab containing v_k thus found is σ_{j+1}.

To solve our line arrangement searching problem, we can use the same algorithm as described above except that we replace its $O(n)$ time oracle by our $O(F)$ time f-oracle for evaluating the function $f(x)$. However, there is a little trick involved. Suppose we consider a slab $[l, r)$ that contains v^* and $[l, r)$ has two sub-slabs $[l, h)$ and $[h, r)$, and we want to determine which sub-slab contains v^*. To this end, we first use the f-oracle to determine the value of $f(h)$. If $f(h) = 0$, then our sought vertex v^* must be in $[h, r)$ due to the monotonicity of f. But if $f(h) = 1$, then v^* can be either in $[l, h)$ or in $[h, r)$. To see this, on one hand, it is obviously possible that $v^* \in [l, h)$ due to the monotonicity of f. On the other hand, it is possible that for any vertex v of $\mathcal{A}_L$ in $[l, h)$, we have $f(x[v]) = 0$ (recall that $x[v]$ is the x-coordinate of the vertex v), in which case