Computational Geometry 58 (2016) 110–123

Contents lists available at ScienceDirect

Computational Geometry: Theory and Applications

www.elsevier.com/locate/comgeo

Optimal randomized incremental construction for guaranteed logarithmic planar point location [☆]

Michael Hemmer [a,*], Michal Kleinbort [b,*], Dan Halperin [b,*]

[a] *Department of Computer Science, University of Technology Braunschweig, Braunschweig, Germany*
[b] *School of Computer Science, Tel Aviv University, Tel Aviv, Israel*

A R T I C L E I N F O

Article history:
Received 12 February 2015
Received in revised form 10 March 2016
Accepted 18 July 2016
Available online 21 July 2016

Keywords:
Point location
Randomized incremental construction

A B S T R A C T

Given a planar map of n segments in which we wish to efficiently locate points, we present the first randomized incremental construction of the well-known trapezoidal-map search-structure that only requires expected $O(n \log n)$ preprocessing time while deterministically guaranteeing worst-case linear storage space and worst-case logarithmic query time. The best previously known randomized construction time of the search structure, which is based on a directed acyclic graph, so-called the *history DAG*, and with the above worst-case space and query-time guarantees, was expected $O(n \log^2 n)$. The result is based on a deeper understanding of the structure of the history DAG, its depth in relation to the length of its longest search path, as well as its correspondence to the *trapezoidal search tree*. Our results immediately extend to planar maps induced by finite collections of pairwise interior disjoint well-behaved curves.

© 2016 Elsevier B.V. All rights reserved.

1. Introduction

The planar point location problem for a set S of n pairwise interior disjoint x-monotone curves inducing a planar subdivision (or a planar arrangement) $\mathcal{A}(S)$ is defined as follows: given a query point q, locate the feature of $\mathcal{A}(S)$ containing q, i.e., the face, edge or vertex of $\mathcal{A}(S)$ in which q lies. It is one of the fundamental problems in Computational Geometry and has numerous applications in a variety of domains, such as computer graphics, motion planning, computer aided design (CAD), geographic information systems (GIS), and many more.

In this work we revisit one of the most elegant and general algorithms for planar point location, namely the randomized incremental construction of the trapezoidal map and the related search structures [3,15,19]. It is also the only algorithm for general x-monotone curves that has an exact, complete and maintained implementation [7,10], which is available via CGAL, the Computational Geometry Algorithms Library [23].

[☆] This work has been supported in part by the 7th Framework Programme for Research of the European Commission, under FET–Open grant number 255827 (CGL—Computational Geometry Learning), by the Israel Science Foundation (grant no. 1102/11), by the German–Israeli Foundation (grant no. 1150-82.6/2011), by the Blavatnik Computer Science Research Fund, and by the Hermann Minkowski–Minerva Center for Geometry at Tel Aviv University. Michael Hemmer is currently affiliated with Google Inc.
[*] Corresponding authors.
E-mail addresses: mhsaar@gmail.com (M. Hemmer), balasmic@post.tau.ac.il (M. Kleinbort), danha@post.tau.ac.il (D. Halperin).

http://dx.doi.org/10.1016/j.comgeo.2016.07.006
0925-7721/© 2016 Elsevier B.V. All rights reserved.

1.1. Previous work

As a core problem in Computational Geometry, the planar point location problem has been well-studied for many years. Among the various solutions to the problem, some methods can only provide an *expected* query time of $O(\log n)$ but cannot guarantee logarithmic query time for all cases. It is particularly true for solutions that only require $O(n)$ space. In addition, certain solutions may only support linear subdivisions, while others are applicable to non-linear ones as well. Triangulation-based point location methods, such as Devillers's Delaunay Hierarchy [4] and the original[1] version of Kirkpatrick's approach [12] are restricted to linear subdivisions, since they build on a triangulation of the actual input. Kirkpatrick creates a hierarchy of $O(\log n)$ levels of triangulated faces (including the outer face), where at each level an independent-set of low-degree vertices is removed when creating the next level in the hierarchy. This approach guarantees that the data structure requires only $O(n)$ space and that a query takes only $O(\log n)$ time. The *Delaunay Hierarchy* of Devillers, on the other hand, does not guarantee logarithmic query time, and may have a linear query time in the worst case.

Most of the other methods can be summarized under the *trapezoidal search graph* model of computation, as pointed out by Seidel and Adamy [20]. The fundamental search structure used by this model is a directed acyclic graph $\mathbb{G}$ (which may even be just a tree for some methods) with one root and many leaves. Internal nodes in $\mathbb{G}$ have two outgoing edges each, and are either labeled with a vertical line and are therefore left–right nodes, or labeled by an input curve and in such a case are top–bottom nodes. In principal, all these solutions can be generalized to support well-behaved curves [8, Subsection 1.3.3], that is, curves that can be decomposed into a finite number of x-monotone pieces.

One of the earliest solutions that can be classified under this model is known as the *slabs method* introduced by Dobkin and Lipton [5]. Every endpoint induces a vertical wall giving rise to $2n+1$ vertical slabs. A point location query is performed by a binary search to locate the correct slab and another search within the slab in $O(\log n)$ time. Preparata [17] introduced the *Trapezoid Graph method* based on the slabs method. His method reduces the space bounds from $O(n^2)$, as required by Dobkin and Lipton's slabs method, to $O(n \log n)$ only, by uniquely decomposing each edge into $O(\log n)$ fragments. Sarnak and Tarjan [18] achieved a significant improvement in memory usage for a slabs-based method by using a *persistent* data structure. Their key observation is that the sequence of search structures in all slabs can be interpreted as one structure that changes over time, which can be stored as a persistent data structure requiring only $O(n)$ size. Another example for this model is the *separating chains method* by Lee and Preparata [14], which also requires linear space. Their algorithm expects a monotone subdivision and uses horizontal monotone chains to separate faces. It is based on the idea that faces of any monotone subdivision can be totally ordered preserving the above–below relation. Each chain is a node in a binary search tree (each edge is kept only once). Querying the structure is essentially deciding whether the query point is above or below $O(\log n)$ chains. However, for each chain this test takes $O(\log n)$, using a binary search. Therefore, the total query time is $O(\log^2 n)$. Another linear size data structure was proposed by Edelsbrunner et al. [6]. They used Fractional Cascading in order to create a layered chain tree as a search structure by copying every other x-value from a node to its parent and maintaining pointers from parent list to child lists. Querying this structure takes $O(\log n)$ time.

This work focuses on the trapezoidal-map randomized incremental construction (RIC), which was introduced by Mulmuley [15] and Seidel [19]. Its associated search structure is a Directed Acyclic Graph (DAG) recording the history of the construction. It achieves expected $O(n \log n)$ preprocessing time, expected $O(\log n)$ query time and expected $O(n)$ space. As pointed out by de Berg et al. [3], the latter two can even be guaranteed. However, their sketched solution would require $O(n \log^2 n)$ preprocessing time. A general major advantage of all variants of this approach is that they can also handle dynamic scenes to some extent, namely, it is possible to add or delete edges later on. The entire method is discussed in more detail in Section 2 below.

Snoeyink and van Kreveld [22] described an approach, based on Kirkpatrick's algorithm, which uses independent sets to determine a proper insertion order of segments. This allows for constructing a search structure with equivalent guarantees in deterministic $O(n \log n)$ time.

A variant of the randomized construction algorithm adds weights and thus gives expected query time satisfying entropy bounds [2]. Arya et al. also stated that entropy preserving cuttings can be used to give a method the query time of which approaches the optimal entropy bound, at the cost of increased space and programming complexity [1]. These methods guarantee a logarithmic query time, however maintaining the search structures requires a considerably large amount of memory and a significant increase in the preprocessing time. Therefore, these solutions are generally rather complicated to implement. For other methods and variants the reader is referred to a comprehensive overview given in [21].

1.2. Contribution

This article extends the theoretical aspects of the work presented in the European Symposium on Algorithms (ESA) 2012 [10], which also presented a major revamp of the exact implementation of the RIC in CGAL, which now guarantees $O(\log n)$ query time and $O(n)$ space.

[1] A generalization was given in [22].

Download English Version:

https://daneshyari.com/en/article/6868562

Download Persian Version:

https://daneshyari.com/article/6868562

Daneshyari.com