Proximate Planar Point Location

John Iacono* Stefan Langerman†

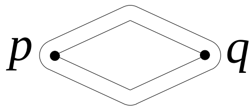

Figure 1: The diamond metric

Abstract

A new data structure is presented for planar point location that executes a point location query quickly if it is spatially near the previous query. Given a triangulation T of size n and a sequence of point location queries $A = q_1, \dots q_m$, the structure presented executes q_i in time $O(\log d(q_{i-1}, q_i))$. The distance metric, d, that is used is a two dimensional generalization of rank distance that counts the number of triangles in a region from the triangle that contains q_{i-1} to the triangle that contains q_i. The data structure uses $O(n \log \log n)$ space.

1 Introduction

Distribution-sensitive data structures have running times that can be expressed as a function of some distributional measure of the sequence of operations performed on the structure. Such structures can exploit sequences of operations that exhibit some desirable behavior. Because real-world access sequences are rarely uniformly random, if our structures are optimized to perform better on the types of distributions likely to be found in a given application, we can obtain running times that are faster than standard (distribution-insensitive) data structures.

Distribution-sensitive dictionaries. Distribution-sensitive structures are well-studied for the *dictionary* problem: maintain a collection of key-value pairs subject to queries for the value associated with a given key. For this problem, two major types of distributions have been studied: spatial and temporal distributions.

The first type of distribution, *spatial* or *locality of reference*, is where items are more likely to be accessed (searched) if they are close, in terms of rank, to the previous access. The level-linked trees of Brown and Tarjan [4], splay trees of Tarjan and Sleator [13, 6, 5], and the unified structure of Iacono [9] all achieve $O(\log |r(q_i) - r(q_{i-1})|)$ query time, where q_i is the ith accessed element, and $r(q_i)$ is the rank of element q_i, (the number of elements less than or equal to it in the dictionary). This is called the *dynamic finger property.*

The second type of distribution, *temporal*, is where items are more likely to be accessed if they have been accessed recently. Data structures with the *working set property* can access an element in time logarithmic in the number of distinct accesses since the last time that element was accessed. Splay trees [13], the working-set structure [9], and the unified structure [9] are all dictionaries with the working set property.

The *unified property* [9] integrates the dynamic finger and working set properties into one, by saying that an access is fast if it is close (in rank) to an item that was accessed recently. Splay trees are conjectured to have this property, but so far only the (complicated and impractical) unified structure [9] is known to have it.

Point Location In computational geometry, there are several planar point location structures that are temporally distribution sensitive. The three papers [2, 3, 10] all present roughly the same result: Given the access probability of each region of a triangulation (assumed to be independent), it is possible to create a data structure whose expected search time is the entropy of that probability distribution. Such a result is analogous to the one-dimensional structures known as *optimal search trees* (Knuth [11]), which date back to 1971. In contrast, splay trees, and in fact any structure with the working-set property, have the same amortized asymptotic runtime as optimal search trees [8], without having to know the access probabilities. Thus the state of the art in distribution-sensitive point location is equivalent to the results for one-dimensional point location obtained thirty years ago.

Point Searching Although the idea of having a sequence of point location queries where each is close to the previous one is a natural idea, there are no known spatially distribution sensitive planar point location structures previous to this paper. However, Demaine, Iacono and Langerman presented in [7] a spatially distribution sensitive data structure for the

*Department of Computer and Information Science, Polytechnic University, 5 MetroTech Center, Brooklyn NY 11201, USA. jiacono@poly.edu

†School of Computer Science, McGill University, 3480 University Street Suite 318, Montreal, Quebec H3A 2A7, Canada. sl@cgm.cs.mcgill.ca. Research is supported by grants from MITACS, FCAR and CRM.

related problem of *point searching*. In the point searching problem, n points $P = \{p_1, \ldots, p_n\}$ are given. The goal is to preprocess the points in order to support an online sequence of point search queries $q_1, \ldots, q_m$. For each query q_i, the data structure must return the index j of a point p_j such that $p_j = q_i$, or it must indicate that no such point exists. In order to obtain a $O(\log d(q_{i-1}, q_i))$, they had to decide on what the appropriate distance metric d was, as the notion of difference in rank used in one dimensional spatially distribution sensitive data structures does not have a clear generalization to higher dimensions. In two dimensions, a similar point-counting metric was developed, where the distance between p and q is the number of points in some two-dimensional region containing both p and q. Such a *region-counting metric* provided a bridge between the geometric distance between two points and the combinatorial complexity of the point set. They abstractly studied region counting metrics, and concluded that the sector metric $d_{s(\theta)}(p, q)$, which counts the points in a sector of angle θ with the apex at p and with q at the center of the arc, is the best metric under certain assumptions. They then developed a data structure for the point searching problem that executes point location queries in time $O(\log d(q_{i-1}, q_i))$, where d is is the fixed triangle metric, which is very close to the sector metric. However, their structure effectively restricts the queries to points in P, thus severely limiting the practical applications of the point searching problem.

Our Results Here we present a data structure for planar point location in a triangulation that has the spatially distribution sensitive runtime of $O(\log d(q_{i-1}, q_i))$ for query q_i. The distance metric d, described fully below, is a region-counting metric that is very similar to the sector metric described above. Our structure is completely new, and shares no elements with the structure for the weaker problem of point searching. The space used is $O(n \log\log n)$, in contrast to the $O(n \log n)$ used by the point searching structure.

The reader may question the use of the somewhat complex idea of a region counting metric as a measure of distance between two points in a triangulation. In a triangulation, the seemingly natural method to count the distance between two points p and q is the number of triangles crossed by the line segment connecting p and q. The problem with this metric, and related metrics such as the distance in the dual graph of the triangulation, is that there may be 2^i points q such that $d(p, q) = i$, thus information theoreticly precluding the possibility of a $O(\log d(q_{i-1}, q_i))$ runtime. This problem is caused by the attempted use of a one dimensional metric in two dimensions and can only be overcome through the use of a true two dimensional metric, which the region-counting metric is.

2 The Distance Metric

The distance $d(p, q)$ is defined as follows: Let $\Diamond(p, q)$ be the diamond shaped region with vertices p and q, symmetry axis pq and angle at p and q both $2\pi/k$, for some constant even integer k. The *granularity* $g(p, q)$ of the query is the length of the longest edge of the triangulation intersecting $\Diamond(p, q)$. Let $\widehat{\Diamond}(p, q)$ be the region of all points at a distance $\leq g(p, q)$ from some point in $\Diamond(p, q)$. We define $d(p, q)$ to be the number of edges intersecting $\widehat{\Diamond}(p, q)$.

3 Data Structure

We here give an overview of the data structure, details will appear in the full version.

Given some line ℓ in the plane, it would be easy to design a data structure allowing proximate point location from any triangle intersecting ℓ to any point q on ℓ, by maintaining the intersections of ℓ with every edge of the triangulation in a level-linked tree [4] and performing finger searches from the originating triangle to the query point. The running time of the query would be the logarithm of the number of edges encountered along the line, which is no larger than the distance reported by our region counting metric.

Our data structure is obtained by combining the above one dimensional technique with a persistent sweep line technique similar to the one in [12], and a doubly exponential data structure hierarchy similar to the one used in [9].

References

[1] S. Arya, S. Cheng, D. Mount, and H. Ramesh. Efficient expected-case algorithms for planar point location. In *Scandanavian Workshop on Algorithm Theory. (LNCS 1851)*, pp. 353–366, 2000.

[2] S. Arya, T. Malamatos, and D. M. Mount. Entropy-preserving cuttings and space-efficient planar point location. In *Symposium on Discrete Algorithms*, pp. 256–261, 2001.

[3] S. Arya, T. Malamatos, and D. M. Mount. A simple emtropy-based algorithm for planar point location. In *Symposium on Discrete Algorithms*, pp. 262–268, 2001.

[4] M. R. Brown and R. E. Tarjan. Design and analysis of a data structure for representing sorted lists. *SIAM J. Comput.*, 9:594–614, 1980.

[5] R. Cole. On the dynamic finger conjecture for splay trees. Part II: the proof. *SIAM J. Computing*, 30(1):44–85, 2000.

[6] R. Cole, B. Mishra, J. Schmidt, and A. Siegel. On the dynamic finger conjecture for splay trees. Part I: splay sorting log n-block sequences. *SIAM J. Computing*, 30(1):1–43, 2000.

[7] E. Demaine, J. Iacono, and S. Langerman. Proximate point searching. In *Canadian Conference on Computational Geometry*, pp. 1–4, 2002.

[8] J. Iacono. New upper bounds for pairing heaps. In *Scandanavian Workshop on Algorithm Theory (LNCS 1851)*, pp. 32–45, 2000.

[9] J. Iacono. Alternatives to splay trees with $O(\log n)$ worst-case access times. In *Symposium on Discrete Algorithms*, pp. 516–522, 2001.

[10] J.Iacono. Optimal planar point location. In *Symposium on Discrete Algorithms*, pp. 240–241, 2001.

[11] D. E. Knuth. Optimum binary search trees. *Acta Inf.*, 1:14–25, 1971.

[12] N. Sarnak and R. E. Tarjan. Planar point location using persistant search trees. *Comm. ACM*, 29:669–679, 1986.

[13] D. D. Sleator and R. E. Tarjan. Self-adjusting binary trees. *JACM*, 32:652–686, 1985.