Computational Geometry 47 (2014) 329–347

Contents lists available at ScienceDirect

Computational Geometry: Theory and Applications

www.elsevier.com/locate/comgeo

Data structures for range-aggregate extent queries ☆

Prosenjit Gupta [a,b,1], Ravi Janardan [c,*,2], Yokesh Kumar [c,2], Michiel Smid [d,3]

[a] Mentor Graphics, Hyderabad 5000082, India
[b] International Institute of Information Technology, Gachibowli, Hyderabad 500032, India
[c] Department of Computer Science & Engineering, University of Minnesota, Minneapolis, MN 55455, USA
[d] School of Computer Science, Carleton University, Ottawa, Ontario, K1S 5B6, Canada

ARTICLE INFO

Article history:
Received 28 August 2008
Received in revised form 27 July 2009
Accepted 3 August 2009
Available online 30 March 2013
Communicated by Pat Morin

Keywords:
Computational geometry
Data structures
Closest pair
Diameter
Width

ABSTRACT

A fundamental and well-studied problem in computational geometry is *range searching*, where the goal is to preprocess a set, S, of geometric objects (e.g., points in the plane) so that the subset $S' \subseteq S$ that is contained in a query range (e.g., an axes-parallel rectangle) can be reported efficiently. However, in many situations, what is of interest is to generate a more informative "summary" of the output, obtained by applying a suitable *aggregation function* on S'. Examples of such aggregation functions include *count*, *sum*, *min*, *max*, *mean*, *median*, *mode*, and *top-k* that are usually computed on a set of weights defined suitably on the objects. Such *range-aggregate* query problems have been the subject of much recent research in both the database and the computational geometry communities.

In this paper, we further generalize this line of work by considering aggregation functions on point-sets that measure the extent or "spread" of the objects in the retrieved set S'. The functions considered here include *closest pair*, *diameter*, and *width*. The challenge here is that these aggregation functions (unlike, say, *count*) are not efficiently decomposable in the sense that the answer to S' cannot be inferred easily from answers to subsets that induce a partition of S'. Nevertheless, we have been able to obtain space- and query-time-efficient solutions to several such problems including: closest pair queries with axes-parallel rectangles on point sets in the plane and on random point-sets in $\mathbb{R}^d$ ($d \geqslant 2$), closest pair queries with disks on random point-sets in the plane, diameter queries on point-sets in the plane, and guaranteed-quality approximations for diameter and width queries in the plane. Our results are based on a combination of geometric techniques, including multilevel range trees, Voronoi Diagrams, Euclidean Minimum Spanning Trees, sparse representations of candidate outputs, and proofs of (expected) upper bounds on the sizes of such representations.

☆ A preliminary version of this paper appeared in the Proceedings of the 20th Canadian Conference on Computational Geometry, Montreal, Aug. 13–15, 2008, pp. 7–10.

* Corresponding author.
E-mail addresses: prosenjit_gupta@acm.org (P. Gupta), janardan@cs.umn.edu (R. Janardan), kumaryo@cs.umn.edu (Y. Kumar), michiel@scs.carleton.ca (M. Smid).

[1] Research supported in part by grants SR/S3/EECE/22/2004 and DST/INT/US/NSF-RPO-0155/04 from the Department of Science and Technology, Government of India.
[2] Research supported, in part, by the National Science Foundation under grants INT–0422775 and CCF-0514950.
[3] Research supported by NSERC.

0925-7721/$ – see front matter
http://dx.doi.org/10.1016/j.comgeo.2009.08.001

1. Introduction

Range searching is an important and well-studied class of problems in computational geometry. In a typical instance of this problem, called *range reporting*, we are given a set, S, of geometric objects (say, points in the plane) that we wish to preprocess into a data structure, so that given any query object Q (say, an axes-parallel rectangle), the subset $S' \subseteq S$ that is contained in Q can be reported efficiently. (Thus, $S' = S \cap Q$.) The paper by Agarwal and Erickson [2] provides a comprehensive survey of geometric range searching.

There are situations where it is not sufficient to merely report the objects of S'; instead, what is desired is a more informative "summary" of the output, such as an order-statistic on S'. For instance, a realtor would be interested in knowing the average or the median price of homes (the "objects") in different neighborhoods (the "queries") of a large city. This can be accomplished by applying a suitable function, called an *aggregation function*, on S'. Examples of aggregation functions include *count*, *sum*, *min*, *max*, *mean*, *median*, *mode*, and *top-k* that are usually computed on a set of weights defined suitably on the objects (in the above example, the weights are house prices). Such *range-aggregate* query problems have been the subject of much recent research in both the database and the computational geometry communities; see, for instance, [4,9,12,16,22,24–27].

In this paper, we make further contributions to range-aggregate query processing by considering aggregation functions that measure the extent or "spread" of the objects in S'. These functions include *closest pair*, *diameter* (or *farthest pair*), and *width*. Extent measures find applications in collision detection, shape-fitting, clustering, etc. [3]. Often, instead of computing the measure on the entire set—which can be both expensive and unnecessary—it is more useful to "zoom in" on a region of interest that is specified by a query range and compute quickly the desired measure only for this region. (For example, given the instantaneous positions of all aircraft over a busy airport, it is important that an air-traffic controller be able to determine rapidly the closest pair within any prescribed region of airspace, in order to identify potential collisions.)

A major challenge in working with these aggregation functions is that they are not decomposable efficiently, in the sense that the answer for S', under one of these functions, cannot be inferred quickly from answers for subsets that form a partition of S'. (For instance, given a partition of S' into sets S'_1 and S'_2, the closest pair in S' cannot be inferred in sublinear time from the closest pair information for S'_1 and S'_2.) Despite this, however, we have been able to obtain space- and query-time-efficient solutions (either exact or approximate solutions with guaranteed error bounds) to several range-aggregate extent queries, as summarized in Table 1. Our results are based on a combination of techniques, including the use of multilevel range trees, Voronoi Diagrams, Euclidean Minimum Spanning Trees, generating sparse representations of candidate output sets, and establishing proofs of (expected) upper bounds on the sizes of such representations.

Shan et al. [19] were among the first to consider the range-aggregate closest pair problem. For axes-parallel query rectangles, they gave a solution based on R-trees and showed that this performed well in practice; however, they did not provide a theoretical analysis of their method. Gupta [10] obtained a solution to this problem in one-dimension (resp., two-dimensions), again for axes-parallel query ranges, where the query time was $O(1)$ using $O(n)$ space (resp., $O(\log^3 n)$ query time using $O(n^2 \log^3 n)$ space). The two-dimensional result was improved recently by Sharathkumar and Gupta [21] to $O(\log^3 n)$ query time using $O(n \log^3 n)$ space. In [20,21], they also considered a variant (motivated by applications in VLSI design rule checking [23]), where the goal is to determine if the closest pair in an axes-parallel query rectangle is within a user-specified tolerance; their approach answered queries in $O(\log^2 n)$ time using $O(n \log^{2+\epsilon} n)$ space, for any constant

Table 1
Summary of results. All results are big-O and, unless noted otherwise, worst-case. Query rectangles are axes-parallel. By "random points" we mean that the points are chosen independently and uniformly at random in the unit-hypercube. For the result in Section 5, by "partly in" we mean that one point in the closest pair is in the query and the other is outside. Here k is a tunable integer-valued parameter, $1 \leqslant k \leqslant n$, δ is a real-valued error tolerance parameter $0 < \delta < 1$, $\epsilon > 0$ is a real-valued constant, and $d \geqslant 2$ is an integer constant. By "variable δ", we mean that it is part of the query; otherwise, it is fixed.

Objects	Query	Aggregation function	Query time	Space	Section
Points in $\mathbb{R}^2$	Rect.	Closest pair	$\log^2 n$	$n \log^5 n$	2
Random points in $\mathbb{R}^d$ ($d \geqslant 2$)	Hyper-rect.	Closest pair	$\log^{2d} n$	$n \log^{3d-2} n$ (expected)	3
Random points in $\mathbb{R}^2$	Disk	Closest pair	$n^{2/3+\epsilon}$ (expected)	$n^{1+\epsilon}$ (expected)	4
Points in $\mathbb{R}^d$ ($d \geqslant 2$)	Hyper-rect. Halfspace Ball	Closest pair ("partly in" query)	$\log^d n$ $n^{1-1/d+\epsilon}$ $n^{1-1/(d+1)+\epsilon}$	$n \log^d n$ $n^{1+\epsilon}$ $n^{1+\epsilon}$	5
Points in $\mathbb{R}^2$	Rect.	Farthest pair	$k \log^5 n$, $1 \leqslant k \leqslant n$	$(n + (n/k)^2) \log^2 n$	6
Points in $\mathbb{R}^2$	Rect.	$(1-\delta)$-farthest pair $(1-\delta)$-farthest pair; variable δ	$\frac{1}{\sqrt{\delta}} \log^2 n$ $\frac{1}{\sqrt{\delta}} \log n + \log^3 n$	$\frac{1}{\sqrt{\delta}} n \log n$ $n \log^2 n$	7
Points in $\mathbb{R}^2$	Rect.	$(1+\delta)$-width	$\frac{1}{\sqrt{\delta}} \log^3 n$	$\frac{1}{\sqrt{\delta}} n \log^2 n$	8