Computational Geometry 47 (2014) 240–247

Contents lists available at ScienceDirect

Computational Geometry: Theory and Applications

www.elsevier.com/locate/comgeo

Streaming and dynamic algorithms for minimum enclosing balls in high dimensions ☆

Timothy M. Chan, Vinayak Pathak *

School of Computer Science, University of Waterloo, Ontario, Canada

ARTICLE INFO

Article history:
Received 21 September 2011
Accepted 21 May 2013
Available online 24 May 2013
Communicated by F. Dehne

Keywords:
Streaming algorithms
Dynamic algorithms
High dimension

ABSTRACT

At SODA'10, Agarwal and Sharathkumar presented a streaming algorithm for approximating the minimum enclosing ball of a set of points in d-dimensional Euclidean space. Their algorithm requires one pass, uses $O(d)$ space, and was shown to have approximation factor at most $(1+\sqrt{3})/2+\varepsilon \approx 1.3661$. We prove that the same algorithm has approximation factor less than 1.22, which brings us much closer to a $(1+\sqrt{2})/2 \approx 1.207$ lower bound given by Agarwal and Sharathkumar.

We also apply this technique to the dynamic version of the minimum enclosing ball problem (in the non-streaming setting). We give an $O(dn)$-space data structure that can maintain a 1.22-approximate minimum enclosing ball in $O(d \log n)$ expected amortized time per insertion/deletion.

1. Introduction

In this paper, we study a fundamental and well-known problem in computational geometry: Given a set P of points in $\mathbb{R}^d$, find the ball with the smallest radius that contains all points in P. This is known as the *minimum enclosing ball* or the *1-center* problem and has various applications, for example, in clustering and facility location. We will not survey the many known exact algorithms for the problem, as the focus of the paper is on *approximation* algorithms in the streaming and the dynamic setting.

In the standard *streaming model*, we consider algorithms that are allowed one pass over the input and can store only a small (usually polylogarithmic) amount of information at any time, as points arrive one at a time. In low constant dimensions, it is not difficult to devise a streaming algorithm that computes a $(1+\varepsilon)$-approximation to the minimum enclosing ball using $O(1/\varepsilon^{(d-1)/2})$ space, by maintaining extreme points along a number of different directions. In fact, streaming techniques for ε-*kernels* [1,8,3,12] allow many other similar geometric optimization problems to be solved with approximation factor $1+\varepsilon$ using $1/\varepsilon^{O(d)}$ space. However, these techniques do not work well in high dimensions because of the exponential dependencies on d.

In high dimensions, there is a trivial streaming algorithm with approximation factor 2: fix the center of the ball B at an arbitrary input point p_0 (say the first point), and whenever a new point p arrives that lies outside B, expand B to include p while keeping the center unchanged (see Section 2.1). Zarrabi-Zadeh and Chan [13] gave a nontrivial analysis showing that another equally simple streaming algorithm achieves approximation factor 3/2: whenever a new point p arrives that lies outside the current ball B, replace B by the smallest ball enclosing $B \cup \{p\}$. An advantage of these simple algorithms is that

☆ A preliminary version of the paper has appeared in Proc. 12th Algorithms and Data Structures Symposium (WADS), Lecture Notes in Computer Science, vol. 6844, 2011, pp. 195–206.

* Corresponding author.
E-mail addresses: tmchan@uwaterloo.ca (T.M. Chan), vpathak@uwaterloo.ca (V. Pathak).

0925-7721/$ – see front matter
http://dx.doi.org/10.1016/j.comgeo.2013.05.007

they avoid the exponential dependencies on d, using asymptotically the minimal amount of storage, namely, $O(d)$ space. (We assume that a unit of space can hold one coordinate value.)

Most recently, Agarwal and Sharathkumar [2] described a new streaming algorithm that also required just $O(d)$ space but with an even better approximation factor. They proved that the factor is upper-bounded by $(1+\sqrt{3})/2+\varepsilon \approx 1.3661$, where as usual, ε denotes an arbitrarily small positive constant. They also proved a complementary lower-bound result: any algorithm in the one-pass streaming model with space polynomially bounded in d has worst-case approximation factor at least $(1+\sqrt{2})/2 > 1.207$. The gap between 1.3661 and 1.207 raises an interesting question of what the best constant could be. It also reveals our current lack of general understanding on high-dimensional geometric problems in the streaming model, as the minimum enclosing ball problem is one of the most basic and simplest to consider.

In this paper, we describe an improved upper bound of 1.22 for minimum enclosing ball in the streaming model. The improvement is actually achieved by the same algorithm as Agarwal and Sharathkumar's; our contribution lies in a better analysis of their algorithm. As intermediate solutions, we first present two simple proofs, one yielding upper bounds of $4/3+\varepsilon \approx 1.333\ldots$[1] and another yielding $16/13+\varepsilon \approx 1.2307$. The 1.22 bound is obtained through more involved numerical calculations done with the assistance of a computer program.

In the second part of the paper, we investigate the *dynamic* version of the approximate minimum enclosing ball problem. Here, we are interested in data structures that support insertions and deletions of input points efficiently. Unlike in the streaming model, linear space is acceptable. As before, the problem is not difficult in low dimensions: one can maintain a $(1+\varepsilon)$-approximation to the minimum enclosing ball in $O(1/\varepsilon^{(d-1)/2}\log n)$ time with a data structure using $O(n/\varepsilon^{(d-1)/2})$ space, by keeping track of extreme points along various directions. The $\log n$ factor in the update time can be reduced to constant in the word RAM model [9].

In the high-dimensional case, it is possible to dynamize the trivial factor-2 method by using a simple randomization trick (see Section 3.1), but we are not aware of any prior work on efficient dynamic data structures that achieve approximation factor smaller than 2 and avoid exponential dependency on d.

We show that Agarwal and Sharathkumar's approach, which was originally intended for streaming, can be applied to the dynamic problem as well, if combined in the right way with ideas from other known techniques: specifically, Bădoiu and Clarkson's static method for high-dimensional minimum enclosing ball [6], and Chan's dynamization strategy for low-dimensional ε-kernels [9]. The resulting data structure requires $O(dn)$ space and supports updates in $O(d\log n)$ expected amortized time. Our analysis of Agarwal and Sharathkumar's technique implies that the same 1.22 upper bound carries over to this dynamic data structure.

2. Streaming MEB

2.1. Preliminaries and Agarwal and Sharathkumar's algorithm

Let P be a set of points in $\mathbb{R}^d$. We use $\mathrm{MEB}(P)$ to denote the minimum enclosing ball of the set P. For a ball B, we use $r(B)$ and $c(B)$ to denote its radius and center respectively. αB stands for the ball with center at $c(B)$ and radius equal to $\alpha r(B)$.

A very simple factor-2 streaming algorithm for approximating the MEB works as follows. Let the first point be p_0. Find the point p_1 in P that is farthest away from p_0. This can be implemented by a one-pass streaming algorithm. Return the ball centered at p_0 of radius $\|p_0p_1\|$. This ball clearly encloses P. The approximation factor is at most 2, since the MEB of P must enclose p_0 and p_1, and any ball that encloses p and q must have radius at least $\|p_0p_1\|/2$.

If more than one pass is allowed, we can get better ratios. In particular, Bădoiu and Clarkson [6] (building on prior work by Bădoiu, Har-Peled, and Indyk [7]) proved that we can achieve an approximation factor of $1+\varepsilon$ in $O(1/\varepsilon)$ passes. The algorithm works as follows. Pick a point $p_0 \in P$. Next, pick p_1 to be the point farthest from p_0 in P. In general, pick p_j to be the point farthest from $c(\mathrm{MEB}(\{p_0,\ldots,p_{j-1}\}))$ in P. It was shown that after $\lceil 2/\varepsilon\rceil$ iterations, the set K of $O(1/\varepsilon)$ chosen points satisfies the following *coreset* property, which implies that $r(\mathrm{MEB}(K))$ (computable by brute force) is a $(1+\varepsilon)$-approximation:

Definition 1. Given a set P of points in $\mathbb{R}^d$, an *ε-coreset* of P is a subset $K \subseteq P$ such that $P \subseteq (1+\varepsilon)\mathrm{MEB}(K)$.

Using Bădoiu and Clarkson's algorithm as a subroutine, Agarwal and Sharathkumar [2] gave a streaming algorithm for finding a $((1+\sqrt{3})/2+\varepsilon)$-factor MEB of a given set of points. The algorithm works as follows. Let the first point in the input stream be its own coreset and call the coreset K_1. Next, as long as the new arriving points lie inside $(1+\varepsilon)\mathrm{MEB}(K_1)$, do nothing. Otherwise, if p_i denotes the new point, call Bădoiu and Clarkson's algorithm on the set $K_1 \cup \{p_i\}$. This gives a new coreset K_2. In general, maintain a sequence of coresets $\mathcal{K} = \langle K_1,\ldots,K_u\rangle$ and whenever a new point p_i arrives that does not lie in $(1+\varepsilon)\mathrm{MEB}(K_j)$ for any j, call Bădoiu and Clarkson's algorithm on the set $\bigcup_{j=1}^{u} K_j \cup \{p_i\}$. However, doing this might make the sequence $\mathcal{K}$ too large. To reduce space, whenever a new call to the subroutine is made, the algorithm also

[1] Independent of our work, Agarwal and Sharathkumar (personal communication, Dec. 2010) have also found a proof of the 4/3 upper bound, which will appear in the journal version of their paper. Even compared against the 4/3 bound, our 1.22 bound is a substantial improvement.