UNIVERSITÉ CÔTE D'AZUR

Approximate Polytope Membership Queries and Applications

Mémoire d'habilitation à diriger des recherches

Spécialité Informatique

Guilherme Dias da Fonseca

Université Clermont Auvergne, LIMOS, and INRIA Sophia Antipolis

fonseca@isima.fr

Jean Cardinal	Université Libre de Bruxelles	Rapporteur
Timothy Chan	University of Illinois at UC	Rapporteur
Nabil Mustafa	ESIEE Paris	Rapporteur
Jean-Daniel Boissonnat	INRIA, Sophia Antipolis	Garant
Victor Chepoi	Aix-Marseille Université	Examinateur
Bruno Martin	Université de Nice Sophia Antipolis	Examinateur
David M. Mount	University of Maryland	Examinateur

June 8, 2018

Abstract

In the *polytope membership* problem, we are given a convex polytope $P \subset \mathbb{R}^d$ for constant $d \geq 2$, and the objective is to preprocess P into a data structure so that, given any query point $q \in \mathbb{R}^d$, it is possible to determine efficiently whether $q \in P$. We consider this problem in an approximate setting. Given an approximation parameter $\varepsilon > 0$, an *ε-approximate polytope membership query* returns a positive result if $q \in P$, a negative result if the distance from q to P is greater than $\varepsilon \cdot \mathrm{diam}(P)$, and it may return either result otherwise.

We presented the first data structures especially tailored to *approximate polytope membership*. Initially, we showed how to answer queries in $O(1/\varepsilon^{(d-1)/4})$ time using optimal $O(1/\varepsilon^{(d-1)/2})$ storage. Later, we improved the analysis of the same data structure to $O(\log(1/\varepsilon)/\varepsilon^{(d-1)/8})$ query time for the same optimal storage. Switching to a different approach, we finally obtained an optimal data structure with $O(\log(1/\varepsilon))$ query time and $O(1/\varepsilon^{(d-1)/2})$ storage. Our data structures yield dramatic improvements to several well-studied geometric approximation problems, where the input is a set of n points and $\alpha > 0$ is an arbitrarily small constant:

- We reduce the storage needed to answer *approximate nearest neighbor* queries in polylogarithmic time from $O(n/\varepsilon^{d-1})$ to $O(n/\varepsilon^{d/2})$.
- We reduce the time to approximate the *diameter* from $O((n + 1/\varepsilon^{d-2}) \log(1/\varepsilon))$ to $O(n \log(1/\varepsilon) + 1/\varepsilon^{(d-1)/2+\alpha})$.
- We reduce the time to construct an *ε-kernel* from $O((n + 1/\varepsilon^{d-2}) \log(1/\varepsilon))$ to near-optimal $O(n \log(1/\varepsilon) + 1/\varepsilon^{(d-1)/2+\alpha})$.
- We reduce the expected time to approximate the *bichromatic closest pair* from $O(n/\varepsilon^{d/3})$ to $O(n/\varepsilon^{d/4+\alpha})$.
- We reduce the expected time to approximate the *minimum bottleneck Euclidean spanning tree* from $O((n \log n)/\varepsilon^{d/3})$ to $O((n \log n)/\varepsilon^{d/4+\alpha})$.

While our initial approach to approximate polytope membership was based on standard techniques of grids and quadtrees, our most efficient data structure presents the first algorithmic application of *Macbeath regions*, a classical structure from convex geometry. The data structure consists of a hierarchy of John ellipsoids of Macbeath regions, where each level of the hierarchy provides a certain degree of approximation of the boundary of the polytope.

Resumé

Dans le problème d'*appartenance à un polytope*, on a un polytope convexe $P \subset \mathbb{R}^d$ pour une constante $d \geq 2$, et l'objectif est de représenter P par une structure de données de façon que, si on reçoit un point de requête $q \in \mathbb{R}^d$, on peut rapidement déterminer si $q \in P$. On considère ce problème dans un contexte d'*approximation*. Étant donné un paramètre $\varepsilon > 0$, la requête doit renvoyer une réponse positive si $q \in P$, une réponse négative si la distance de q à P est supérieure à $\varepsilon \cdot \mathrm{diam}(P)$, et sinon on peut avoir n'importe quelle réponse, où $\mathrm{diam}(P)$ est le diamètre de P.

On a présenté les premières structures de données développées spécialement pour l'approximation de l'appartenance à un polytope. Initialement, on a montré comment répondre aux requêtes en temps $O(1/\varepsilon^{(d-1)/4})$ avec un stockage optimal de $O(1/\varepsilon^{(d-1)/2})$. Ensuite, on a raffiné l'analyse de la même structure de données pour obtenir temps de requête $O(\log(1/\varepsilon)/\varepsilon^{(d-1)/8})$ avec le même stockage optimal. Finalement, on a étudié une approche différente pour obtenir une structure de données optimale avec temps de requête $O(\log(1/\varepsilon))$ et stockage $O(1/\varepsilon^{(d-1)/2})$. Nos structures de données apportent d'énormes améliorations pour plusieurs problèmes classiques d'approximation géométrique, où l'entrée est un ensemble de n points et $\alpha > 0$ est une constante arbitrairement petite :

- On réduit le stockage pour approximer le *plus proche voisin* en temps polylogarithmique de $O(n/\varepsilon^{d-1})$ à $O(n/\varepsilon^{d/2})$.
- On réduit le temps pour approximer le *diamètre* de $O((n + 1/\varepsilon^{d-2}) \log(1/\varepsilon))$ à $O(n \log(1/\varepsilon) + 1/\varepsilon^{(d-1)/2+\alpha})$.
- On réduit le temps pour construire un *ε-noyau* de $O((n + 1/\varepsilon^{d-2}) \log(1/\varepsilon))$ à $O(n \log(1/\varepsilon) + 1/\varepsilon^{(d-1)/2+\alpha})$.
- On réduit le temps d'un algorithme probabiliste pour approximer les *deux points les plus rapprochés bichromatiques* de $O(n/\varepsilon^{d/3})$ à $O(n/\varepsilon^{d/4+\alpha})$.
- On réduit le temps d'un algorithme probabiliste pour approximer l'*arbre couvrant Euclidien d'étranglement minimal* de $O((n \log n)/\varepsilon^{d/3})$ à $O((n \log n)/\varepsilon^{d/4+\alpha})$.

Alors que notre solution initiale était basée sur de techniques bien connues comme les grilles et quadtrees, notre structure de données la plus efficace présente la première application algorithmique des *régions de Macbeath*, une structure classique de la géométrie convexe. La structure de donnés utilise une hiérarchie d'ellipsoïdes de John des régions de Macbeath, où chaque niveau de la hiérarchie fournit un certain degré d'approximation du bord du polytope.

Acknowledgments

I would like to thank Jean-Daniel Boissonnat for welcoming me multiple times at INRIA, Sophia Antipolis, where I prepared most of this dissertation, as well as for his valuable advice and encouragement.

I would like to thank David M. Mount for all the research we have done together and also for the enlightening experience that comes from spending time with him, both at personal and professional levels.

I would like to thank Suni Arya for all the research we have done together, for the multiple invitations to Hong Kong University of Science and Technology, and for the wisdom he always shares.

Finally, I would like to thank all committee members for making this work possible.

Contents

Chapter 1

Introduction

The field of computational geometry greatly succeeded its initial goal of designing efficient algorithms and data structures for numerous planar problems, such as convex hull, Voronoi diagram, point location, and range searching [51]. However, major barriers were encountered when trying to solve these problems in arbitrary dimensions. In fact, several lower bound results show that these problems require a processing time of roughly n^d, where n is the input size and d is the dimension. In order to surpass this barrier, called the *curse of dimensionality*, we resort to finding approximate solutions instead of exact ones.

Most of my current research topics are in the area of geometric approximation (see [65] for an introductory textbook). I'm particularly interested in approximations with respect to distances in spaces of constant dimension d. I focus this dissertation around one such problem: approximate polytope membership. Several fundamental problems are closely related to approximate polytope membership, and are also presented in this dissertation.

The work presented herein is based on four conference papers and one journal paper. The conference papers were presented in STOC 2011 [11], SODA 2012 [13], SODA 2017 [16], and SoCG 2017 [14]. The journal paper compiling the first two conference papers appeared at SIAM Journal on Computing [17]. All these papers were the fruit of my collaboration with Sunil Arya (Hong Kong University of Science and Technology) and my PhD advisor David Mount (University of Maryland) after the conclusion of my PhD. The papers cover a much greater level of detail than the present dissertation. The objective here is to survey the state of art of the problem, presenting some of the simplest proofs to give an idea of the type of argument used. Detailed proofs of all results can be found in the attached papers.

This dissertation is organized as follows. In Chapter 2, I consider the approximate polytope membership problem per se. After defining the problem and surveying previous work, I present two data structures that we proposed to solve the problem. The first data structure, called Split-Reduce, is based on a simple quadtree construction. However, it is not easy to analyze its complexity, and tight bounds remain an open problem. We prove some simple upper and lower bounds from [11] and give a high level idea of the improved bounds upper bounds from [13]. Then, I describe a completely different data structure [16], whose construction is much less elementary, but whose analyses is somewhat simpler. I present an informal definition of the data structure, and give one simple proof to hint at the kind of tools used.

In Chapter 3, I consider some of the several applications of approximate polytope membership. I start with our original motivating application [11] of approximate nearest neighbor searching. After surveying the rich history of the problem, I present the reduction from approximate nearest neighbor to approximate polytope membership. Then, I move to the problem of ε-kernels, giving an intuitive idea of our construction [14]. Next, I present the problems of diameter, bichromatic closest pair, and Euclidean trees, showing how these problems reduce to approximate polytope membership.

In Chapter 4, I briefly present some ongoing work. First I discuss the problem of approximating the Minkowski sum of two convex bodies, with the application of approximating the width of a convex body. Another ongoing work that I mention is the problem of approximate nearest neighbor searching in non-Euclidean metrics. I also discuss a related result in discrete geometry, where we consider the complexity of an approximating convex body [12, 15]. Finally, I list some open problems.

I use the standard computational geometry approach, which focuses on asymptotic complexities in the real RAM model of computation, unless otherwise specified. In most geometric approximation problems, a parameter $\varepsilon > 0$ is given to limit the approximation error allowed: the closer ε is to zero, the closer to an exact solution we get. Some works on geometric approximation assume that ε is a constant, while other works consider that ε is an asymptotic variable that approaches zero. The results presented herein are of the latter kind. The main goal of my work is to reduce ε-dependencies, for example, from $1/\varepsilon^d$ to $1/\varepsilon^{d/2}$. Furthermore, we assume that d is a constant, therefore hiding some often unavoidable exponential dependencies on d such as 2^d. Throughout, we use the term polytope to mean a bounded convex body with flat faces.

Chapter 2

Approximate Polytope Membership

Polytopes are key structures in many areas of mathematics and computation. In this dissertation, we consider a fundamental search problem related to polytopes. Let P denote a convex body in $\mathbb{R}^d$, that is, a closed, convex set of bounded diameter that has a nonempty interior. The *polytope membership problem* is that of preprocessing P so that it is possible to determine efficiently whether a given query point $q \in \mathbb{R}^d$ lies within P. Throughout, we assume that the dimension d is a fixed constant that is at least 2.

It follows from standard results in projective duality that deciding membership in a polytope defined as the intersection of n halfspaces is equivalent to answering halfspace emptiness queries for a set of n points. In dimension $d \leq 3$, it is possible to build a data structure of linear size that can answer such queries in logarithmic time [51]. In higher dimensions, however, all known exact data structures with roughly linear space have a query time of $\widetilde{\Omega}\left(n^{1-1/\lfloor d/2 \rfloor}\right)$[1] [72], which is unacceptably high for many applications. Polytope membership is a special case of polytope intersection queries [28, 46, 53]. Barba and Langerman [28] showed that for any fixed d, it is possible to preprocess polytopes in $\mathbb{R}^d$ so that given two such polytopes that have been translated and rotated, it can be determined whether they intersect each other in time that is logarithmic in their total combinatorial complexity. However, the preprocessing time and space grow as the combinatorial complexity of the polytope raised to the power $\lfloor d/2 \rfloor$. Since the combinatorial complexity of a polytope with n vertices can be as high as $\Theta(n^{\lfloor d/2 \rfloor})$, the storage upper bound is roughly $O(n^{d^2/4})$.

The lack of efficient exact solutions motivates the question of whether polytope membership queries can be answered approximately. We provide two strongly related definitions for approximate polytope membership. Let ε be a positive real parameter, and let $\mathrm{diam}(P)$ denote P's diameter. Given a query point $q \in \mathbb{R}^d$, a *weak ε-approximate polytope membership query* returns a positive result if $q \in P$, a negative result if the distance from q to its closest point in P is greater than $\varepsilon \cdot \mathrm{diam}(P)$, and it may return either result otherwise (see Figure 2.1(a)).

Sometimes it is useful to use a stronger definition of approximation, sensitive not only to the diameter, but to directional widths. Consider an expanded convex body $P' \supset P$. A query returns a positive result if the query point q is inside P, a negative result if q is outside P', and may return either result otherwise. In the *weak* version, we define P' as

[1] Throughout, we use $\widetilde{\mathrm{O}}(\cdot)$ and $\widetilde{\Omega}(\cdot)$ as variants of $O(\cdot)$ and $\Omega(\cdot)$, respectively, that ignore logarithmic factors.

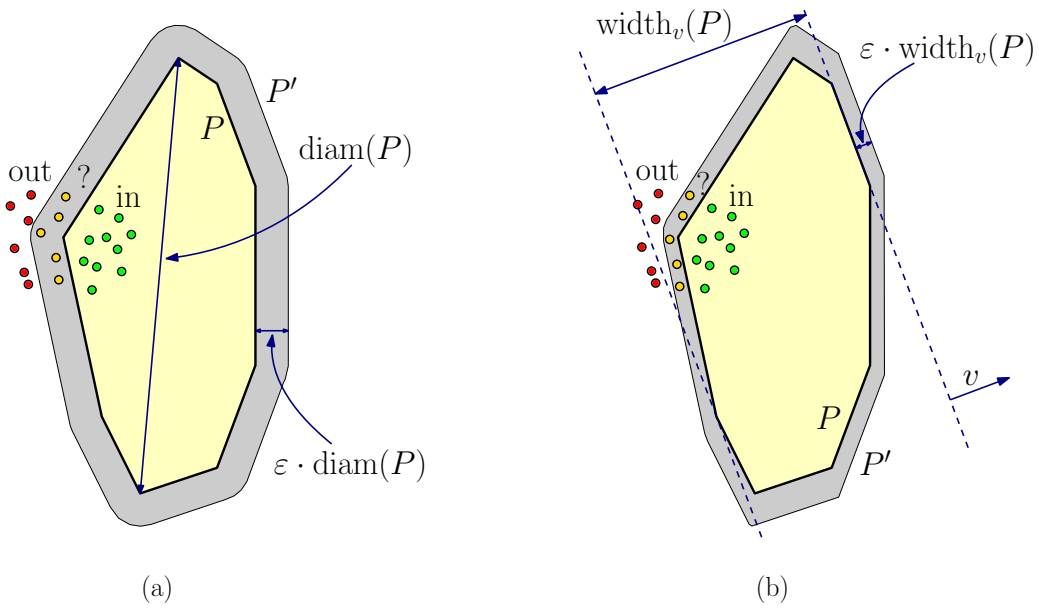


Figure 2.1: Approximate polytope membership: (a) weak problem formulation, (b) strong problem formulation.

the set of points that lie within distance $\varepsilon \cdot \text{diam}(P)$ of P, thus defining a body whose Hausdorff distance from P is $\varepsilon \cdot \text{diam}(P)$.

In the *strong ε-approximate polytope membership query*, we define P' as follows. For any nonzero vector $v \in \mathbb{R}^d$, consider the two supporting hyperplanes for P that are normal to v. Translate each of these hyperplanes outward by a distance of $\varepsilon \cdot \text{width}_v(P)$, and consider the closed slab-like region lying between them. Define P' to be the intersection of this (infinite) set of slabs (see Figure 2.1(b)). This is clearly a stronger approximation than the Hausdorff-based definition.

A solution for the weak version can be converted into a solution for the strong version using standard fattening techniques [5]. This reduction does not affect polynomial asymptotic complexities, since the value of ε is only changed by a constant factor. Therefore, we refer to the weak version when solving the problem, but we may apply the stronger version when using approximate polytope membership as a black box.

Polytope membership queries, both exact and approximate, arise in many application areas, such as linear-programming and ray-shooting queries [38, 44, 71, 73, 79], nearest neighbor searching and the computation of extreme points [39, 49], collision detection [58], and machine learning [37].

Existing solutions to approximate polytope membership queries have been based on straightforward applications of classic polytope approximation techniques. Given a polytope P, we say that a polytope P' is an *outer ε-approximation* of P if $P \subseteq P'$, and the Hausdorff distance between P' and P is at most $\varepsilon \cdot \text{diam}(P)$ (see Figure 2.1(a)). An *inner ε-approximation* is defined similarly but with $P' \subseteq P$. Dudley [54] showed that there exists an outer ε-approximating polytope for any bounded convex body in $\mathbb{R}^d$ formed by the intersection of $O\big(1/\varepsilon^{(d-1)/2}\big)$ halfspaces, and Bronshteyn and Ivanov [35] proved an analogous bound on the number of vertices needed to obtain an inner ε-approximation. Both bounds are known to be optimal in the worst case since $\Omega\big(1/\varepsilon^{(d-1)/2}\big)$ halfspaces

are necessary to approximate an Euclidean ball (see, e.g., [36]). These results have been applied to a number of problems, for example, the construction of coresets [5].

Dudley's construction provides a naïve solution to the approximate polytope membership problem. Construct an ε-approximation P', and determine whether q lies within all its bounding halfspaces. This approach takes $O(1/\varepsilon^{(d-1)/2})$ query time and space. An alternative simple solution was proposed by Bentley *et al.* [30]. Create a d-dimensional grid with cells of diameter ε and, for every column along the x_d-axis, store the two extreme x_d values where the column intersects P. This algorithm produces an approximation P' with $O(1/\varepsilon^{d-1})$ facets. Given a query point q, it is easy to determine if $q \in P'$ in logarithmic time (or even in constant time if we assume a model of computation that supports the floor function), but the space required by the approach is $O(1/\varepsilon^{d-1})$.

We are the first to present data structures especially tailored to approximate polytope membership. Our first data structure is based on a simple quadtree construction algorithm, called *split-reduce*. However, the analysis of the storage space of the data structure obtained by this algorithm is highly non-trivial. We present upper and lower bound to the storage. The data structure, upper and lower bounds to the storage, and applications have been presented in STOC 2011 [11]. An improved upper bound has been presented in SODA 2012 [13]. A journal version compiling both papers and additional details appears at SIAM Journal on Computing [17] and is attached to this dissertation. We present the split-reduce approach in Section 2.1.

While the split-reduce data structure is both simple and practical, the question of whether it is possible to achieve query time $O(\log \frac{1}{\varepsilon})$ with minimum storage $O(1/\varepsilon^{(d-1)/2})$ has remained open. We give an affirmative answer to this question. We abandon the quadtree-based approach in favor of a data structure involving a hierarchy of ellipsoids. These ellipsoids are selected through a sampling process that is inspired by a classical structure from the theory of convexity, called *Macbeath regions* [70]. This optimal data structure has been presented in SODA 2017 [16]. A faster preprocessing algorithm is presented in SoCG 2017 [14], together with many applications. Both papers are attached to this dissertation. We present the key ideas of the Macbeath region approach in Section 2.2.

2.1 Split-Reduce

As mentioned in the previous Section, Dudley's construction [54] gives a data structure with optimal storage space of $O(1/\varepsilon^{(d-1)/2})$, while Bentley *et al.* [30] gives optimal query time. These two extreme solutions raise the question of whether a tradeoff is possible between space and query time. Before presenting our results, it is illustrative to consider a very simple method for generating such a tradeoff. Given a polytope P, we say that a polytope P' is an *absolute ε-approximation* of P if the Hausdorff distance between P' and P is at most ε, regardless of the diameter of P. It follows immediately from Dudley's result that we can obtain an absolute ε-approximation to a polytope P of diameter $\mathrm{diam}(P)$ with $O((\mathrm{diam}(P)/\varepsilon)^{(d-1)/2})$ facets.

Given $r \in [\varepsilon, 1]$, subdivide the bounding hypercube into a regular grid of cells of diameter r and, for each cell that intersects the polytope's boundary, apply Dudley's absolute approximation to this portion of the polytope. By a straightforward packing argument, the number of occupied cells is $O(1/r^{d-1})$. Since each cell is of diameter $O(r)$,

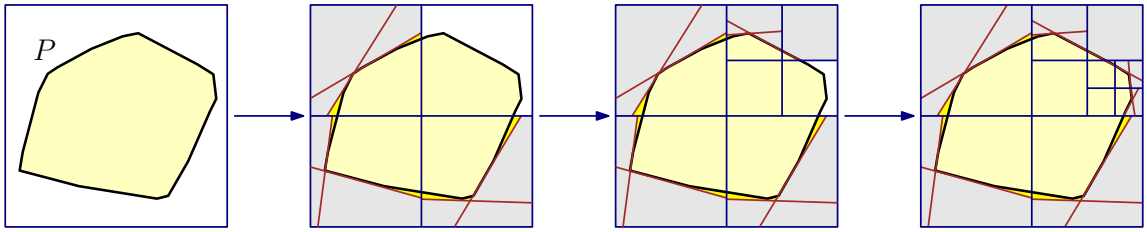


Figure 2.2: Example of the Split-Reduce algorithm with $t = 2$.

each can be approximated to absolute error ε using $O((r/\varepsilon)^{(d-1)/2})$ facets per cell. Subject to minor technical details, the result is a data structure of space $O(1/(\varepsilon r)^{(d-1)/2})$ and query time $O((r/\varepsilon)^{(d-1)/2})$. This interpolates nicely between the two extremes for $\varepsilon \leq r \leq 1$. The space-time trade-off is illustrated in Figure 2.4(a).

Given the optimality of Dudley's approximation, it may be tempting to think that the above tradeoff is optimal, but we will demonstrate that it is possible to do better. (To see why, observe that each piece of the polytope's boundary carries only a portion of the polytope's total curvature. Clarkson has shown that, when curvature is constrained, Dudley's bound is not tight [50].) We show that it is possible to achieve significantly better space-time tradeoffs for approximate polytope membership.

The data structure and its construction are both extremely simple. The data structure consists of a quadtree of height $O(\log \frac{1}{\varepsilon})$, where each leaf cell stores a set of halfspaces whose intersection approximates the polytope's boundary within this cell. A query is answered by performing a point location in the quadtree followed by a brute force inspection of the halfspaces in the node. The data structure is constructed by the following recursive algorithm, called Split-Reduce, illustrated in Figure 2.2. It is given the polytope $P \subseteq [-1, 1]^d$, the approximation parameter ε, and the desired query time t. The initial quadtree box is $Q = [-1, 1]^d$.

Split-Reduce(Q):

1. Let P' be an absolute ε-approximation of $Q \cap P$.
2. If the number of facets $|P'| \leq t$, then Q stores the hyperplanes bounding P'.
3. Otherwise, split Q into 2^d quadtree boxes and invoke Split-Reduce on each such box.

Step 1 consists of obtaining a polytope P' that absolute ε-approximates $Q \cap P$. Some polytopes can be approximated with far fewer than the $\Theta(1/\varepsilon^{(d-1)/2})$ facets generated by Dudley's construction. For example, a simplex can be represented exactly with $d + 1$ facets. The problem of generating a minimum-facet ε-approximation to a polytope can be reduced to a set cover problem [76]. By applying the well known greedy set-cover heuristic, it is possible to produce such an approximation in which the number of facets exceeds the optimum by a factor of $O(\log \frac{1}{\varepsilon})$. In particular, the set cover instance consists of one set for each facet f of the polytope, and the corresponding set contains all facets that are approximated by f's supporting hyperplane. Clarkson [48] presented a somewhat more complicated algorithm that does better. He showed that if c is the smallest number of facets required to approximate P, then it is possible to obtain an approximation with $O(c \log c)$ facets in $O(nc^2 \log n \log c)$ randomized time, where n is the number of facets of P.

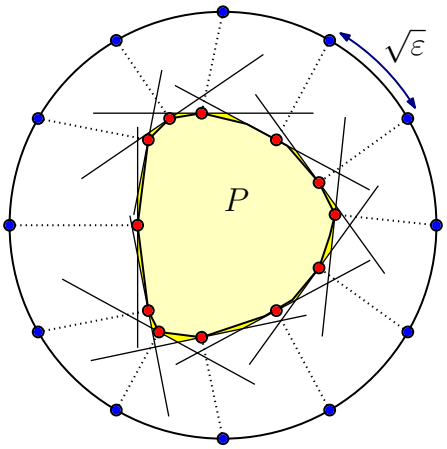


Figure 2.3: Dudley's polytope approximation.

For simplicity, we assume that the algorithm used in Step 1 produces an absolute approximation to $Q \cap P$ with a minimum number of facets. An alternative is to apply Chan's coreset construction [43] to P and then run Clarkson's approximation algorithm on the result. This yields a preprocessing time of $O(n + 1/\varepsilon^{3(d-1)/2})$ but increases the query time by a negligible factor of $O(\log \frac{1}{\varepsilon})$.

For completeness, let us now describe Dudley's algorithm. Given a bounded polytope P, let the *size* of P, denoted $\text{size}(P)$, be the side length of the smallest axis aligned box Q that contains P. Through an appropriate translation we may assume that the center of Q is the origin. Dudley's algorithm obtains an approximation P' as follows. Let B be a ball of radius $\text{size}(P)\sqrt{d}$ centered at the origin. (Note that $P \subset B$.) Place a set W of $\Theta((\text{size}(P)/\varepsilon)^{(d-1)/2})$ points on the surface of B such that every point on the surface of B is within distance $O(\sqrt{\varepsilon\,\text{size}(P)})$ of some point in W. For each point $w \in W$, let w' be its nearest point on the boundary of P. We call these points *samples*. For each sample point w', take the supporting halfspace passing through w' that is orthogonal to the vector from w' to w. P' is the intersection of these halfspaces (see Figure 2.3).

A key element in the analysis of Dudley's construction is that, for each point p on the boundary of P, there is a sample point w' in the above construction (subject to a suitable adjustment of the constant factors) whose distance from p is at most $\sqrt{\varepsilon\,\text{size}(P)}$, and the distance from p to the supporting hyperplane at w' is at most ε. In summary, we may view Dudley's approximation as producing a set of $\Theta((\text{size}(P)/\varepsilon)^{(d-1)/2})$ halfspaces, where each halfspace is responsible for approximating a region of P's boundary of diameter at most $\sqrt{\varepsilon\,\text{size}(P)}$. We exploit the limited range of each Dudley halfspace in the following lemma.

Lemma 2.1. *Given a polytope P and $\varepsilon > 0$, Dudley's method produces a collection of $O((\text{size}(P)/\varepsilon)^{(d-1)/2})$ halfspaces whose intersection ε-approximates P. Furthermore, given a square grid of side length $O(\sqrt{\varepsilon\,\text{size}(P)})$, we may associate each halfspace with $O(1)$ grid cells, such that this halfspace is used for approximating the boundary of P within only these cells.*

Recall that our algorithm takes four inputs: polytope P, box Q, query time t, and approximation error ε. For simplicity, we refer to the algorithm as Split-Reduce(Q), since

the parameters P, t, ε remain unchanged throughout the recursive calls. The output of our algorithm is a quadtree whose leaf cells induce a subdivision of Q. Each leaf cell L stores a set of $t_L \leq t$ halfspaces whose intersection approximates $P \cap L$, where t_L is the minimum number of halfspaces required to approximate $P \cap L$. The storage of this quadtree is defined as the total number of stored halfspaces over all the leaf cells. Before establishing the space-time tradeoff, we show that the algorithm produces a data structure with query time $t = \Theta(1/\varepsilon^{(d-1)/4})$ and the same storage as Dudley's algorithm, $O(1/\varepsilon^{(d-1)/2})$ (see Figure 2.4)(b).

Theorem 2.2. *The output of Split-Reduce*(Q) *for*

$$t \geq \left(\frac{\text{size}(Q)}{\varepsilon}\right)^{(d-1)/4} \geq 1$$

is a quadtree with storage $O((\text{size}(Q)/\varepsilon)^{(d-1)/2})$.

Proof. Let T denote the quadtree produced by the algorithm. For each leaf cell L of T, let t_L be the number of halfspaces stored in L. We will show that $\sum_L t_L \leq (\text{size}(Q)/\varepsilon)^{(d-1)/2}$, which establishes the desired storage bound.

Toward this end, we first prove a lower bound on the size of any leaf cell L. We assert that there exists a constant c_1 such that every leaf cell L has size $\text{size}(L) \geq \sqrt{\varepsilon\,\text{size}(Q)/c_1}$. The assertion follows from Lemma 2.1. In particular, the standard Dudley technique applied to a cell of size $\sqrt{\varepsilon\,\text{size}(Q)/c_1}$ produces at most $c_D(\text{size}(Q)/c_1\varepsilon)^{(d-1)/4}$ halfspaces, where c_D is the constant arising from Dudley's method. By choosing c_1 to be a sufficiently large constant, the number of halfspaces is at most t, and the termination condition of our algorithm implies that such a cell is not further subdivided.

Let P_D be the polytope obtained by applying Dudley's algorithm to $P \cap Q$. Combining our assertion with Lemma 2.1 (where $P \cap Q$ plays the role of P in the lemma), each bounding halfspace of P_D is used in approximating $O(1)$ leaf cells. We assign each halfspace of P_D to these leaf cells. The correctness of Dudley's algorithm implies that the halfspaces assigned to any cell L provides an approximation of $P \cap L$. Thus, t_L is no more than the number of Dudley halfspaces assigned to L. Since the number of halfspaces of P_D is $O((\text{size}(Q)/\varepsilon)^{(d-1)/2})$, it follows that $\sum_L t_L = O((\text{size}(Q)/\varepsilon)^{(d-1)/2})$, which completes the proof. □

Using much more sophisticated techniques we improve the upper bound in the previous theorem to the following space-time tradeoff. We omit the details, but the key element is a better understanding of the relationship between the curvature of the portion of a convex body inside a quadtree cell and the complexity to approximate that portion. We use "lg" to denote base-2 logarithm.

Theorem 2.3. *Given a polytope P in $\mathbb{R}^d$, an approximation parameter $0 < \varepsilon \leq 1$, and a real constant $\alpha \geq 4$, there is a data structure for ε-approximate polytope membership queries that achieves*

$$\textit{Query time: } O\left((\log \tfrac{1}{\varepsilon})/\varepsilon^{\frac{d-1}{\alpha}}\right)$$
$$\textit{Storage: } O\left(1/\varepsilon^{(d-1)\left(1-\frac{2\lfloor \lg \alpha \rfloor - 2}{\alpha}\right)}\right).$$

The constant factors in the space and the query time depend only on d and α (not on P or ε).

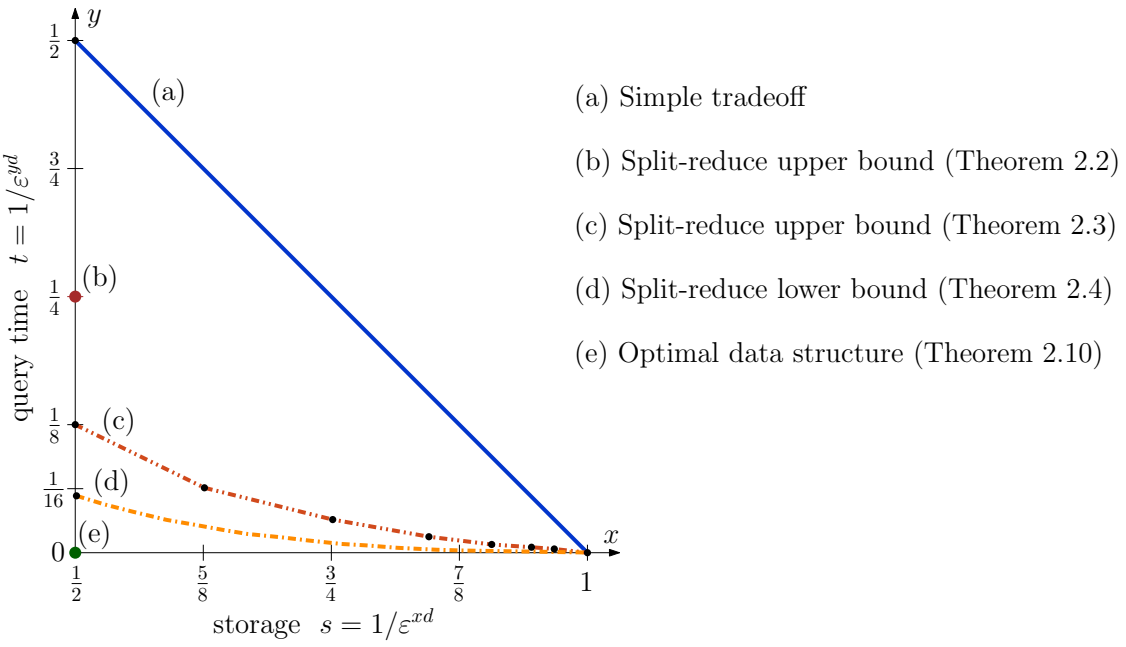


Figure 2.4: Query time as a function of the storage space for approximate polytope membership.

The above space bound is a simplification, and the exact bound is given in the paper. Both bounds are piecewise linear in $1/\alpha$ (with breakpoints at powers of two), but the exact bounds are continuous as a function of α. The resulting space-time trade-off is illustrated in Figure 2.4(c). (The plot reflects the more accurate bounds.)

We do not know whether the upper bounds presented in Theorem 2.3 are tight for our algorithm. In [11,17], we establish the following lower bound on the trade-off achieved by this algorithm. The lower bound is illustrated in Figure 2.4(d).

Theorem 2.4. *In any fixed dimension $d \geq 2$ and for any constant $\alpha \geq 4$, there exists a polytope such that for all sufficiently small positive ε, the data structure generated by split-reduce to achieve query time $O\big(1/\varepsilon^{(d-1)/\alpha}\big)$ has space*

$$\Omega\left(1/\varepsilon^{(d-1)\left(1-\frac{2\sqrt{2\alpha}-3}{\alpha}\right)-1}\right).$$

Our approach is similar to the lower bound proof of [21]. (Note that this is a lower bound on the performance of Split-Reduce, not on the problem complexity.) It is based on analyzing the performance of the algorithm on a particular convex body, a generalized hypercylinder that is curved in $k+1$ dimensions and flat in $d-1-k$ dimensions. We select the value of k that produces the best lower bound on the storage as a function of t, ε, and d.

As mentioned earlier, it is well known that $\Omega\big(1/\varepsilon^{(d-1)/2}\big)$ facets are required to ε-approximate a Euclidean ball of unit radius (see, e.g., [36]), and this holds for any polytope that is sufficiently close to a ball in terms of Hausdorff distance. The following utility lemma generalizes this observation to different diameters.

Lemma 2.5. *Let ε and Δ be real parameters, where $0 < \varepsilon \leq \Delta/4$. There exists a constant c_b and a polytope P in $\mathbb{R}^d$ of diameter at most Δ such that any outer ε-approximation of P requires at least $c_b(\Delta/\varepsilon)^{(d-1)/2}$ facets.*

Intuitively, in order to produce a polytope that is hard to approximate, it should have high curvature. If the curvature is high in all dimensions, however, the polytope will have a small surface area, and this will make it easier to approximate. Our approach is to consider polytopes based on generalized cylinders, which have constant curvature in some dimensions but are flat in others. Our next lemma introduces such a cylindrical polytope where the number of curved dimensions has been carefully chosen to maximize the space needed by our algorithm for a given query time. Theorem 2.4 is an immediate consequence.

Lemma 2.6. *There exists a polytope P in $\mathbb{R}^d$ such that for all sufficiently small positive ε (depending on d and α) and $t = 1/\varepsilon^{(d-1)/\alpha}$, the output of Split-Reduce(K, Q_0) on P has total space*

$$\Omega\left(1/\varepsilon^{(d-1)\left(1-\frac{2\sqrt{2\alpha}-3}{\alpha}\right)-1}\right).$$

Proof. To start, as a function of α, we wish to compute an integer dimension k in order to apply Lemma 2.5. Define reals $\delta = \sqrt{\alpha/2}/(d-1)$, $\kappa = (d-1)\sqrt{2/\alpha}$ and $\kappa' = \kappa(1+\delta)$. We observe first that

$$\kappa' - \kappa \;=\; \delta(d-1)\sqrt{2/\alpha} \;=\; 1.$$

Let $k = \lceil \kappa \rceil$, implying that $\kappa \leq k \leq \kappa'$. (Although we do not include the derivation here, κ has been chosen to produce the best lower bound, but since it is not necessarily an integer, k is obtained by rounding to a nearby integer.) Since $\alpha \geq 4$ and $d \geq 2$, we have $1 \leq k \leq d-1$.

Let c_b denote the constant of Lemma 2.5, and let $\Delta = \varepsilon((2^d+1)t/c_b)^{2/k}$. By our assumptions about d and α, we have $t = 1/\varepsilon^{\Theta(1)}$ and $\Delta = \varepsilon \cdot t^{\Theta(1)}$. It follows that for all sufficiently small ε, $\Delta/4 \geq \varepsilon$. Let h denote the linear subspace spanned by the first $k+1$ coordinate axes. We apply Lemma 2.5 in $\mathbb{R}^{k+1}$ for this value of Δ. The resulting polytope P (lying in h) has the property that the number of facets of any ε-approximation is at least

$$c_b\left(\frac{\Delta}{\varepsilon}\right)^{k/2} \;=\; c_b\left(\frac{\varepsilon\left(\frac{(2^d+1)t}{c_b}\right)^{2/k}}{\varepsilon}\right)^{k/2} \;=\; (2^d+1)t.$$

We can bound P's diameter by observing that for all sufficiently small ε

$$\begin{aligned}\operatorname{diam}(P) \;\leq\; \Delta \;&=\; \varepsilon\left(\frac{(2^d+1)t}{c_b}\right)^{2/k} \;\leq\; \varepsilon\left(\frac{2^d+1}{c_b\cdot\varepsilon^{(d-1)/\alpha}}\right)^{2/\kappa} \\ &=\; \varepsilon\left(\frac{2^d+1}{c_b\cdot\varepsilon^{(d-1)/\alpha}}\right)^{\sqrt{2\alpha}/(d-1)}.\end{aligned}$$

(Here we made use of the fact that for all sufficiently small ε, the quantity raised to power of $2/k$ is greater than 1.) Letting $c_b' = ((2^d+1)/c_b)^{\sqrt{2\alpha}/(d-1)}$, we obtain

$$\operatorname{diam}(P) \;\leq\; c_b'\varepsilon\left(\frac{1}{\varepsilon^{(d-1)/\alpha}}\right)^{\sqrt{2\alpha}/(d-1)} \;=\; c_b'\varepsilon^{1-\sqrt{2/\alpha}}.$$

Since $\alpha \geq 4$, for all sufficiently small ε, we have $\text{diam}(P) \leq 1/\sqrt{d}$. Therefore, P can be enclosed within $Q_0^{(k+1)}$.

Returning to $\mathbb{R}^d$, consider an infinite polyhedral hypercylinder whose "axis" is the $(d-1-k)$-dimensional orthogonal complement of h, and whose "cross-section" (i.e., intersection with any $(k+1)$-dimensional hyperplane parallel to h) is P. Define the polytope C to be the truncated cylinder obtained by intersecting the infinite hypercylinder with hypercube $Q_0^{(d)}$ (see Figure 2.5(a)). Let T denote the output of Split-Reduce$(K, Q_0^{(d)})$ for C, ε, and t. We will show that T's total space satisfies the bound given in the lemma's statement. To do this, let Σ denote any set of points placed on C's axis such that the distance between each pair of points is at least $2\Delta\sqrt{d}$. (In the degenerate case where $k = d-1$ the axis is 0-dimensional and Σ degenerates to a single point.) By a simple packing argument, there exists such a set having $\Omega(1/\Delta^{d-1-k})$ points.

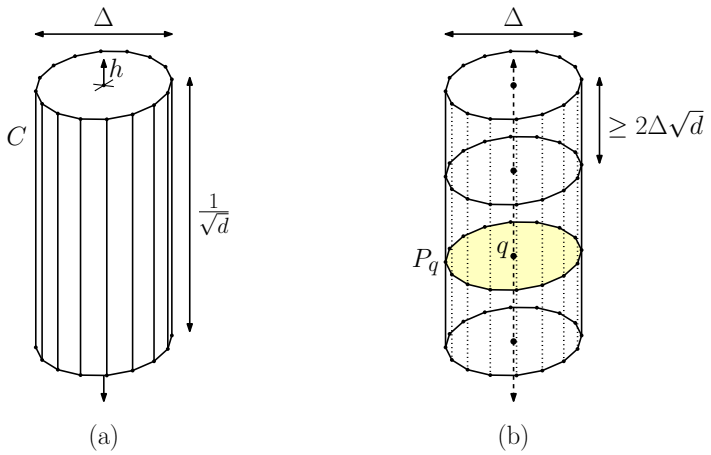
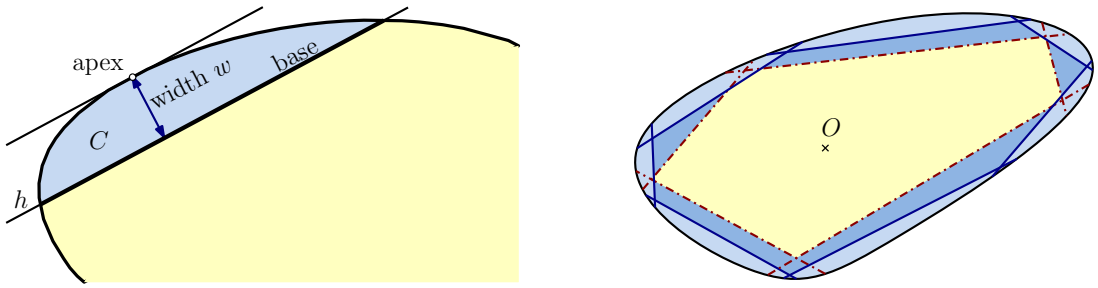


Figure 2.5: Lemma 2.6 for $d = 3$ and $k = 2$.

For any $q \in \Sigma$, let P_q denote the cross-section of C passing through q (see Figure 2.5(b)). Consider the set of leaf cells of T that intersect P_q. By applying Lemma 2.5 to the $(k+1)$-dimensional hyperplane on which P lies, it follows that these cells together must contain at least $(2^d+1)t$ halfspaces. We count the contributions of these cells by classifying them into two types. We say that a leaf cell of T is *large* if its side length is at least Δ, and otherwise it is *small*. By a simple packing argument, the number of large leaf cells intersecting P_q is at most 2^d. Since each leaf cell contains at most t halfspaces, the large leaf cells can together contain at most $2^d t$ halfspaces.

Therefore, the small leaf cells intersecting P_q together contain at least $(2^d+1)t - 2^d t = t$ halfspaces. Because the points of Σ are separated from each other by distance at least $2\Delta\sqrt{d}$, which is strictly larger than the diameter of any small leaf cell, each small leaf cell can intersect P_q for at most one $q \in \Sigma$. Therefore, the total space contribution of all the small leaf cells for all points of Σ is at least $t \cdot |\Sigma|$. Let $c_b'' = (c_b/(2^d+1))^{2(d-1-k)/k}$. T's total space can be asymptotically bounded from below as

$$\frac{t}{\Delta^{d-1-k}} = \frac{t}{\left(\varepsilon\left(\frac{(2^d+1)t}{c_b}\right)^{2/k}\right)^{d-1-k}} = \frac{c_b'' \cdot t}{\left(\varepsilon \cdot t^{2/k}\right)^{d-1-k}} = \frac{c_b'' \cdot t^{1-2(d-1-k)/k}}{\varepsilon^{d-1-k}}.$$



(a) (b)

Figure 2.6: (a) Cap concepts. (b) Covering the boundary of a convex body by caps at different levels.

Clearly, $c_b'' = \Theta(1)$. Recall that $t = 1/\varepsilon^{(d-1)/\alpha}$. Then, T's total space is asymptotically bounded from below as

$$\left(\frac{1}{\varepsilon}\right)^{(d-1)-k+\frac{d-1}{\alpha}\left(1-\frac{2(d-1-k)}{k}\right)} = \left(\frac{1}{\varepsilon}\right)^{(d-1)-k+\frac{d-1}{\alpha}\left(3-\frac{2(d-1)}{k}\right)} \tag{2.1}$$

Let $E(\alpha)$ denote this exponent. In order to complete the proof, we provide a lower bound on $E(\alpha)$. We use the fact that $\kappa \leq k \leq \kappa'$, apply the definitions of κ, κ', and δ, and straightforward manipulations to obtain

$$E(\alpha) \geq (d-1) - \kappa' + \frac{d-1}{\alpha}\left(3 - \frac{2(d-1)}{\kappa}\right) = (d-1)\left(1 - \frac{2\sqrt{2\alpha}-3}{\alpha}\right) - 1.$$

Substituting this value for the exponent in Eq. (2.1) completes the proof. □

2.2 Hierarchy of Macbeath Regions

In contrast to the previous construction, which is based on grids and quadtrees, our next construction employs a classical structure from the theory of convexity, called *Macbeath regions* [70]. Macbeath regions have found numerous uses in the theory of convex sets and the geometry of numbers (see Bárány [26] for an excellent survey). They have also been applied to several problems in the field of computational geometry. However, most previous results were either in the form of lower bounds [20,25,34] or focused on existential results [12,15,56,77]. Our work [14,16] presents the first explicit construction of Macbeath regions in algorithms and data structures.

Before we define Macbeath regions and present our actual construction, it is useful to consider a related construction that however does not give an efficient data structure. A *cap* C is defined to be the nonempty intersection of the convex body P with a halfspace H (see Figure 2.6(a)). Let h denote the hyperplane bounding H. We define the *base* of C to be $h \cap P$. The *apex* of C is any point in the cap such that the supporting hyperplane of P at this point is parallel to h. The *width* of C is the distance between h and this supporting hyperplane.

Caps may be used to define a hierarchical approximation of the boundary of the convex body of unit diameter. At level zero, we have a constant number of caps of constant width

w_0 that cover the boundary. At each level i we divide the previous width by two, covering the boundary with caps of width $w_i = w_0/2^i$ (see Figure 2.6(b)). We stop after $O(\log \frac{1}{\varepsilon})$ levels when the width of the caps is roughly ε. It follows from the works of Dudley [54] and Bronshteyn and Ivanov [35] that $O(1/w_i^{(d-1)/2})$ caps of width w_i are enough to cover the boundary. Therefore, the number of caps summed over all levels form a geometric progression with a total of $O(1/\varepsilon^{(d-1)/2})$ caps. To define the hierarchical relation, the children of each cap are the caps at the following level that intersect it.

Given such hierarchy of caps, we answer an approximate polytope membership query for a query point q as follows. Let the origin O be a centrally located point inside the convex body. We navigate through the hierarchy following the ray emanating from O toward q, starting at level zero. At each level we keep as an invariant a cap C that intersects the ray. To determine the invariant for the following level, it suffices to examine the children of C.

For this data structure to be potentially efficient, the number of children of any given cap must be small. Unfortunately, this is not the case. To see the issue, imagine a three-dimensional cone. Many caps of width ε are necessary to cover the rounded part of the cone (the number is $O(1/\varepsilon^{(d-2)/2})$ for a d-dimensional cone). The problem is that all these caps may contain the apex of the cone. Therefore the number of caps of width ε intersecting a given ray may be almost as large as the total number of caps.

The source of the problem is that caps do not provide good *packing properties*. For example, a very large number of caps may all intersect each other even if the volume of the intersection between any two caps is small, as seen in the cone. Notice that other geometric objects such as quadtree cells or congruent balls provide good packing properties. These objects however have roughly the same width in all directions, and therefore cannot adapt well to the different directional curvatures of the convex body. Next, we introduce Macbeath regions, which are not fat and provide the ability to adapt to different curvatures (like caps), but simultaneously provide good packing properties (like quadtree boxes).

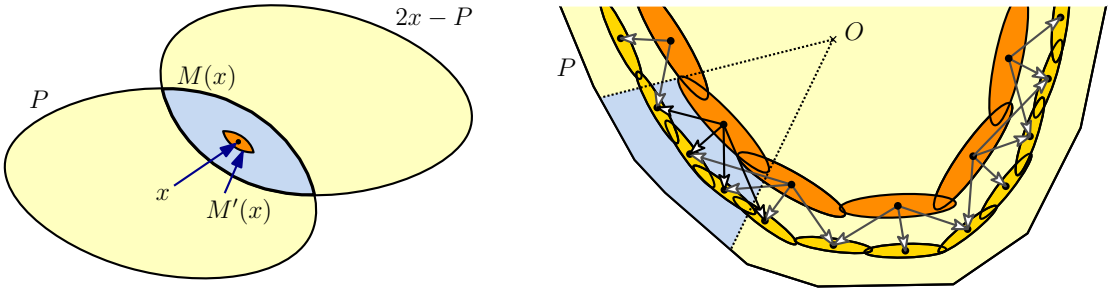
Let P be an arbitrary convex body. Given a point $x \in P$ and real parameter $\lambda \geq 0$, the *Macbeath region* $M^\lambda(x)$ (also called an *M-region*) is defined as:

$$M^\lambda(x) \;=\; x + \lambda((P - x) \cap (x - P)).$$

It is easy to see that $M^1(x)$ is the intersection of P and the reflection of P around x (see Figure 2.7(a)), and so $M^1(x)$ is centrally symmetric about x. $M^\lambda(x)$ is a scaled copy of $M^1(x)$ by the factor λ about x. We refer to x as the *center* of $M^\lambda(x)$ and to λ as its *scaling factor*. As a convenience, we define $M(x) = M^1(x)$ and $M'(x) = M^{1/5}(x)$. We refer to the latter as the *shrunken* Macbeath region. In fact $M(x)$ is a good approximation of the minimum volume cap containing point x. However, the shrunken Macbeath regions provide good packing properties.

We now present a lemma that illustrates the good packing properties of Macbeath regions. The lemma shows that if two shrunken Macbeath regions have a nonempty intersection, then a constant factor expansion of one contains the other [34, 59]. We give a proof as an example of the techniques used.

Lemma 2.7. *Let P be a convex body, and let $\lambda \leq 1/5$ be any positive real. If $x, y \in P$ such that $M^\lambda(x) \cap M^\lambda(y) \neq \emptyset$, then $M^\lambda(y) \subseteq M^{4\lambda}(x)$.*



(a) (b)

Figure 2.7: (a) Macbeath regions. (b) Two levels of the hierarchy of ellipsoids based on Macbeath regions.

Proof. Let z be a point in the intersection of $M^\lambda(x)$ and $M^\lambda(y)$. Then we can write z as:

$$z = x + \lambda(x - p_1) = y + \lambda(p_2 - y),$$

where $p_1, p_2 \in P$. Equating the two expressions for z above, we obtain

$$y = \frac{(1+\lambda)x - \lambda p_1 - \lambda p_2}{1-\lambda}.$$

Consider any point $w \in M^\lambda(y)$. We have

$$w = y + \lambda(y - p_3) = (1+\lambda)y - \lambda p_3,$$

where $p_3 \in P$. Substituting the expression obtained above for y, we have

$$w = \frac{(1+\lambda)((1+\lambda)x - \lambda p_1 - \lambda p_2)}{1-\lambda} - \lambda p_3,$$

which simplifies to

$$w = x + \frac{\lambda(3+\lambda)}{1-\lambda}(x - p),$$

where

$$p = \frac{1+\lambda}{3+\lambda}p_1 + \frac{1+\lambda}{3+\lambda}p_2 + \frac{1-\lambda}{3+\lambda}p_3.$$

As p is a convex combination of p_1, p_2 and p_3, $p \in P$. Thus, we have shown that

$$M^\lambda(y) \subseteq x + \frac{\lambda(3+\lambda)}{1-\lambda}(x - P). \tag{2.2}$$

In an analogous manner, we next show that

$$M^\lambda(y) \subseteq x + \frac{\lambda(3+\lambda)}{1-\lambda}(P - x). \tag{2.3}$$

Again, let z be any point in the intersection of $M^\lambda(x)$ and $M^\lambda(y)$. We can write z as:

$$z = x + \lambda(k_1' - x) = y + \lambda(y - k_2'),$$

where $k'_1, k'_2 \in P$. Equating the two expressions for z above, we obtain

$$y = \frac{(1-\lambda)x + \lambda k'_1 + \lambda k'_2}{1+\lambda}.$$

Consider any point $w \in M^\lambda(y)$. We have

$$w = y + \lambda(k'_3 - y) = (1-\lambda)y + \lambda k'_3,$$

where $k'_3 \in P$. Substituting the expression obtained above for y, we have

$$w = \frac{(1-\lambda)((1-\lambda)x + \lambda k'_1 + \lambda k'_2)}{1+\lambda} + \lambda k'_3,$$

which simplifies to

$$w = x + \frac{\lambda(3-\lambda)}{1+\lambda}(p' - x),$$

where

$$p' = \frac{1-\lambda}{3-\lambda}k'_1 + \frac{1-\lambda}{3-\lambda}k'_2 + \frac{1+\lambda}{3-\lambda}k'_3.$$

As p' is a convex combination of k'_1, k'_2 and k'_3, $p' \in P$. Letting p'' denote the point on segment xp' such that

$$\frac{\lambda(3-\lambda)}{1+\lambda}(p' - x) = \frac{\lambda(3+\lambda)}{1-\lambda}(p'' - x),$$

we can write

$$w = x + \frac{\lambda(3+\lambda)}{1-\lambda}(p'' - x),$$

where $p'' \in P$. Thus,

$$M^\lambda(y) \subseteq x + \frac{\lambda(3+\lambda)}{1-\lambda}(P - x),$$

which establishes Eq. (2.3). By combining this with Eq. (2.2), we obtain $M^\lambda(y) \subseteq M(x, \lambda(3+\lambda)/(1-\lambda))$. Since $\lambda \leq 1/5$, it is easy to see that $(3+\lambda)/(1-\lambda) \leq 4$. Thus $M^\lambda(y) \subseteq M(x, 4\lambda)$, completing the proof. □

As mentioned before, the fact that we can cover the boundary of a convex body by $O(1/\varepsilon^{(d-1)/2})$ caps of width ε follows from classic works. Such a bound is essential to obtain a data structure with low storage space. However, only recently did we prove an analogous bound using Macbeath regions [15]. Since Macbeath regions $M^\lambda(x)$ for $\lambda < 1$ do not reach the boundary of the convex body, we need to define a region close to the boundary that we cover. For any $\delta > 0$, define the *δ-erosion* of P, denoted $P(\delta)$, to be the closed convex body formed by removing from P all points lying within distance δ of ∂P. We showed the following.

Lemma 2.8. *Let P be a convex body of unit diameter. Let $\lambda > 0$ be a sufficiently small constant that depends only on d and $\varepsilon > 0$ be a sufficiently small parameter. Let X denote a maximal set of points lying on the boundary of the eroded body $P(\varepsilon)$ such that the associated Macbeath regions $M^\lambda(x)$ for $x \in X$ are pairwise disjoint. Then the collection of Macbeath regions $\{M^{4\lambda}(x) : x \in X\}$ covers $\partial P(\varepsilon)$ and $|X| = O(1/\varepsilon^{(d-1)/2})$.*

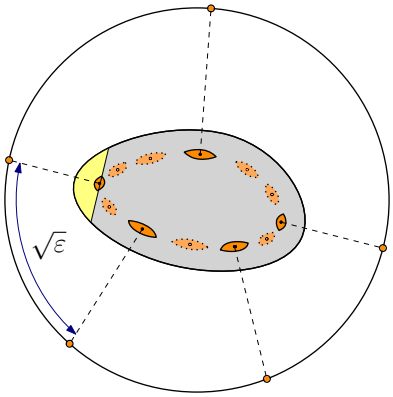


Figure 2.8: Sketch of the proof of Lemma 2.8.

The relation between a maximal packing of Macbeath regions and a covering of Macbeath regions follows from Lemma 2.7. The detailed proof that the number of Macbeath regions in a packing is $|X| = O(1/\varepsilon^{(d-1)/2})$ is presented in [15]. Next, I'll sketch the key ideas of the proof.

Let X be the set of centers of the disjoint Macbeath regions from Lemma 2.8. The proof begins with a pruning step that only reduces the size of X by a constant factor. We process the Macbeath regions $M^\lambda(x)$ for $x \in X$ ordered by increasing volume. For each region $M^\lambda(x)$ considered, we remove from X all Macbeath regions of larger volume that intersect $M^{c\lambda}(x)$, where c is a sufficiently large constant. It follows from packing properties that the number of regions removed is only a constant fraction, and therefore does not change the asymptotic value of $|X|$.

Next, we surround the convex body P by a ball of constant diameter, which we call the *Dudley ball*. We then perform and orthogonal projection of each point $x \in X$ onto the Dudley ball (orthogonal to the minimum volume cap containing x), as shown in Figure 2.8. Finally, we argue that the distance between any two projected points is $\Omega(\sqrt{\varepsilon})$. Since the total area of the Dudley ball is constant, the bound follows.

Another issue to obtain an efficient data structure is that Macbeath regions of a polytope are polytopes, but may have a large number of vertices. In order to efficiently use Macbeath regions in a data structure, we need to approximate the Macbeath region by a convex body of constant complexity. To this purpose, we use John's theorem [67]:

Lemma 2.9. *Let $P \subset \mathbb{R}^d$ be a centrally symmetric convex body centered at the origin. There exists an ellipsoid $E \supseteq P$ (*John ellipsoid*) such that $P \supseteq E/\sqrt{d}$, where $E/\sqrt{d}$ denotes the scaling of E by a factor of $1/\sqrt{d}$.*

By standard results, we can construct the John ellipsoid of a polytope in time that is linear in the number of its defining halfspaces [47]. For a Macbeath region $M^\lambda(x)$, we denote its circumscribing John ellipsoid by $E^\lambda(x)$, which we call a *Macbeath ellipsoid*. Applying John's theorem, we have $M^\lambda(x) \subseteq E^\lambda(x) \subseteq M^{\lambda\sqrt{d}}(x)$.

In [16] (attached), we show that Macbeath ellipsoids can be used to produce a hierarchy whose levels approximate the boundary of a convex body to different degrees of precision. This hierarchy has several good properties. First, the total number of ellipsoids is $O(1/\varepsilon^{(d-1)/2})$, matching the information theoretic lower bound of an ε-approximation of a convex body. Second, the number of levels is logarithmic and each node has $O(1)$

children. Finally, by performing a straightforward navigation in the hierarchy we can answer approximate polytope membership queries in $O(\log \frac{1}{\varepsilon})$ time. Next, we informally describe this hierarchy. For a more precise definition, which all constant explicitly given, please refer to [16].

Let P denote a polytope of unit diameter in $\mathbb{R}^d$. The hierarchy has $O(\log \frac{1}{\varepsilon})$ levels. Each level i corresponds to a covering of eroded body $P(\delta_i)$ by a set of $O(1/\delta_i^{(d-1)/2})$ Macbeath ellipsoids, where $\delta_i = \Theta(1/2^i)$. Each ellipsoid has $O(1)$ children, which correspond to the ellipsoids of the following level that approximate the same portion of the boundary (see Figure 2.7(b)). The hierarchy starts with $\delta_0 = \Theta(1)$ and stops after $O(\log \frac{1}{\delta})$ levels when $\delta_i = \Theta(\varepsilon)$, for a desired approximation ε.

To answer an ε-approximate polytope membership query for a given query point q, we consider the ray Oq. The query algorithm descends the DAG by starting at the root and visiting any node at level zero that intersects the ray. Letting u denote the current node, we next visit any child of u whose associated ellipsoid intersects the ray. (Such a child must exist.) We repeat the procedure until we reach the leaf level. Upon reaching the leaf level we intersect Oq with the leaf ellipsoid (which is a good approximation to the boundary of P near the ray Oq. Let p be an arbitrary point in the intersection. If p is between O and q, we return that q is outside of P. Otherwise, we return that q is inside P.

In [16], we present a simple algorithm to construct a δ-approximation hierarchy in $O(n + 1/\delta^{3(d-1)/2})$ time. We assume that the input polytope P is presented as the intersection of n halfspaces. Next, we describe the algorithm.

We first show how to construct the set of ellipsoids covering the eroded body $P(\delta)$, which form a level in the hierarchy. Translate each bounding halfspace of P toward the origin by amount δ. It is easy to see that the polytope $P(\delta)$ is the intersection of the translated halfspaces. This can be done in $O(n)$ time.

Consider a hypercube of constant size enclosing P. Superimpose a $\Theta(\delta)$-grid on each of the $2d$ facets of this hypercube. Intersect the segment joining the origin to each grid point with $\partial P(\delta)$, and let $X' \subset \partial P(\delta)$ denote the resulting set of intersection points. Note that $|X'| = O(1/\delta^{d-1})$. Using the fact that $P(\delta)$ is fat, a straightforward geometric calculation shows that for any point on $\partial P(\delta)$, there is a point of X' within distance $c\delta$ of it, where c is a suitable constant. (Adjusting the constant factor in the grid spacing, we can ensure that $c \leq \lambda_0\sqrt{d}$, which is a fact that we will use later in the proof.) As each point of X' can be determined in $O(n)$ time, X' can be computed in $O(n/\delta^{d-1})$ time.

For each $x' \in X'$, construct $M^\lambda(x')$. Let $\mathcal{M}'$ denote the resulting set of Macbeath regions. As a straightforward consequence of the definition of Macbeath regions, we can compute each Macbeath region in $O(n)$ time (i.e., in time proportional to the number of halfspaces that define P). Note that we represent each Macbeath region as the intersection of n halfspaces. For each Macbeath region $M^\lambda(x') \in \mathcal{M}'$, determine the circumscribing John ellipsoid. By standard results, we can construct the John ellipsoid of a polytope in time that is linear in the number of its defining halfspaces [47]. Thus, this step also takes time $O(n|\mathcal{M}'|) = O(n/\delta^{d-1})$.

Next, we will determine a maximal subset $\mathcal{M} \subseteq \mathcal{M}'$ such that the John ellipsoids associated with the Macbeath regions of $\mathcal{M}$ are disjoint. Initialize $\mathcal{M} = \emptyset$. Examine the Macbeath regions of $\mathcal{M}'$ one by one. Insert the Macbeath region into $\mathcal{M}$ if its associated John ellipsoid does not intersect the John ellipsoid of any Macbeath region of $\mathcal{M}$. Clearly,

this method yields a maximal subset $\mathcal{M} \subseteq \mathcal{M}'$ such that the associated John ellipsoids are disjoint. To bound the time required for this step, observe that the Macbeath regions of $\mathcal{M}$ are disjoint, and so by Lemma 2.8, $|\mathcal{M}| = O(1/\delta^{(d-1)/2})$. Since it takes constant time to check whether two ellipsoids intersect, it follows that the time required is $O(|\mathcal{M}'| \cdot |\mathcal{M}|) = O(1/\delta^{3(d-1)/2})$. The desired set of John ellipsoids covering $P(\delta)$ is the set of properly scaled John ellipsoids of the Macbeath regions in $\mathcal{M}$.

We build each level of the hierarchy independently. Since the running time for each level forms an geometric progression, the total running time is $O(1/\varepsilon^{3(d-1)/2})$. Given nodes u and v from levels i and $i+1$, respectively, v is a *child* of u if there exists a ray emanating from the origin that intersects both $E^{4\lambda_0\sqrt{d}}(x_u)$ and $E^{4\lambda_0\sqrt{d}}(x_v)$. Since an ellipsoid has constant complexity and the number of ellipsoids at each level is at most $O(1/\varepsilon^{(d-1)/2})$, computing the child/parent relationship does not add to the total running time, and therefore we build the whole hierarchy in $O(1/\varepsilon^{3(d-1)/2})$ time.

However, the exponent of $3(d-1)/2$ in the preprocessing time is higher than we would like. In [14] (attached), we show that we can build an approximate polytope membership data structure much faster. The key idea is to avoid building the whole hierarchy, but instead build multiple partial hierarchies and connecting them. For the algorithm to work in the desired running time, we need another result of independent interest, a faster algorithm to compute an ε-kernel, which is briefly explained in Section 3.2. As a result, we obtain the following theorem.

Theorem 2.10. *Given a polytope P in $\mathbb{R}^d$ represented as the intersection of n halfspaces and an approximation parameter $\varepsilon > 0$, there is a data structure that can answer ε-approximate polytope membership queries with*

$$\textit{Query time: } O\left(\log \frac{1}{\varepsilon}\right)$$
$$\textit{Storage: } O\left(1/\varepsilon^{\frac{d-1}{2}}\right).$$

Furthermore, the preprocessing time is

$$O\left(n \log \frac{1}{\varepsilon} + \left(\frac{1}{\varepsilon}\right)^{\frac{d-1}{2}+\alpha}\right),$$

where $\alpha > 0$ is an arbitrarily small constant.

Note that this optimal data structure corresponds to a point in Figure 2.4(e).

Chapter 3

Applications

In this chapter, we discuss several applications of the data structures presented in the previous chapter. We start with the classic problem of approximate nearest neighbor searching, showing how to reduce it to approximate polytope membership. Next, we move from data structure problems to static problems, where the efficient preprocessing of the approximate polytope membership data structure becomes a key element. The first static problem we consider is the ε-kernel. This problem is particularly tied to the approximate polytope membership data structure: we use kernel algorithms to efficiently build approximate polytope membership data structures and we use the approximate polytope membership data structure to build kernels. Next, we turn our attention to other static problems that are connected to approximate nearest neighbors and approximate polytope membership through previously known reductions.

3.1 Approximate Nearest Neighbor

Let S be a set of n points in $\mathbb{R}^d$. Given any $q \in \mathbb{R}^d$, an *ε-approximate nearest neighbor* (ANN) of q is any point of S whose distance from q is at most $(1+\varepsilon)$ times the distance to q's closest point in S (Figure 3.1). The objective is to preprocess S in order to answer such queries efficiently.

Approximate nearest neighbor searching in spaces of fixed dimension has been widely studied. Data structures with $O(n)$ storage and query times no better than $O(\log n + 1/\varepsilon^{d-1})$ have been proposed by several authors [24, 31, 42, 55]. In subsequent papers, it

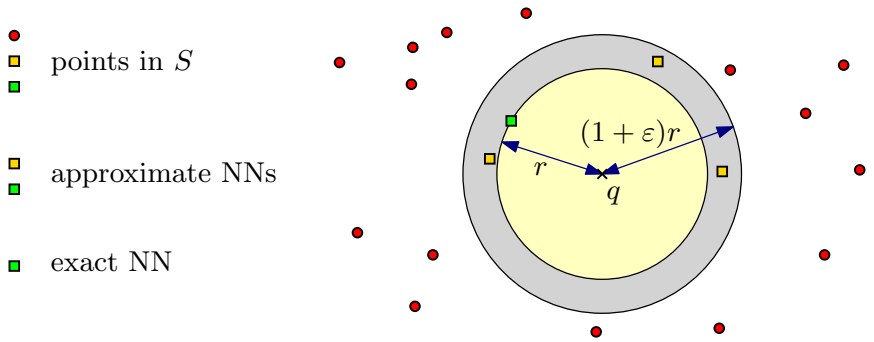


Figure 3.1: Exact and approximate nearest neighbor of q.

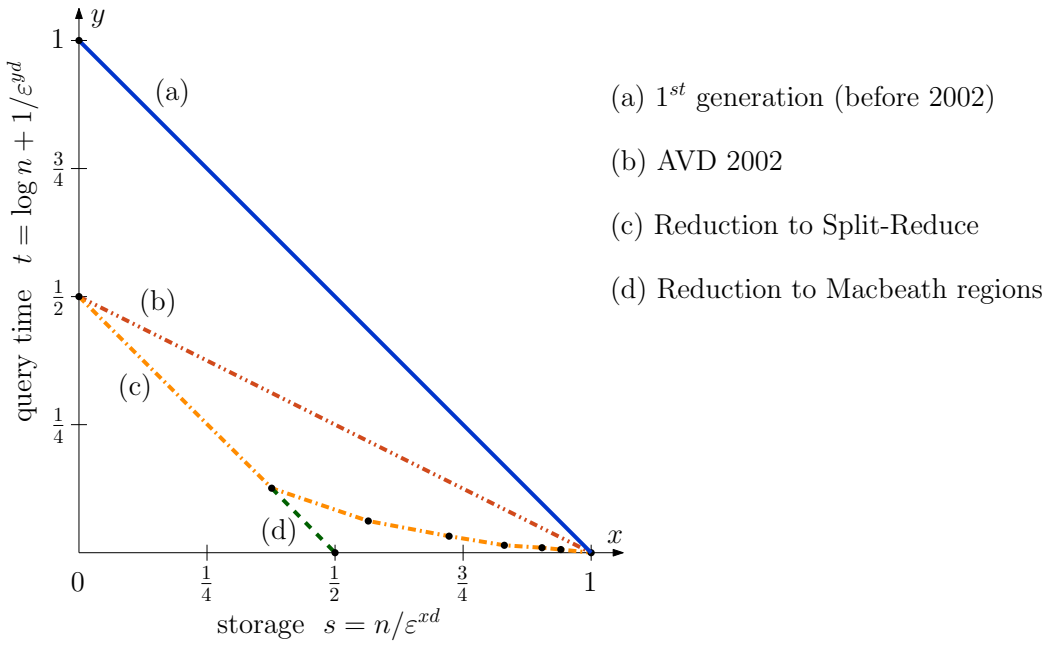


Figure 3.2: Query time as a function of the storage space for approximate nearest neighbor searching.

was shown that query times could be reduced at the expense of greater storage [40,49,64, 80]. Har-Peled introduced the AVD (approximate Voronoi diagram) data structure and showed that $O(\log \frac{n}{\varepsilon})$ query time could be achieved using roughly $O(n/\varepsilon^d)$ space [64]. The product of the ε-dependencies in the storage and query time remain close to $O(1/\varepsilon^d)$ throughout the tradeoff, as represented in Figure 3.2(a).

Space-time trade-offs were established for the AVD in a series of papers [10,18,19,21]. At one end of the spectrum, it was shown that with $O(n)$ storage, queries could be answered in time $O(\log n + 1/\varepsilon^{(d-1)/2})$. At the other end, queries could be answered in time $O(\log \frac{n}{\varepsilon})$ with space roughly $O(n/\varepsilon^d)$. The product of the ε-dependency in the storage and the *square* of the ε-dependency in the query time remain close to $O(1/\varepsilon^d)$ throughout the tradeoff (see Figure 3.2(b)).

In [11], we presented a reduction from Euclidean approximate nearest neighbor searching to polytope membership, which we explain later on. We established significant improvements to the best trade-offs throughout the middle of the spectrum, but the extremes were essentially unchanged [11,13]. See Figure 3.2(c). While the AVD is simple and practical, in [21] lower bounds were presented that imply that significant improvements at the extreme ends of the spectrum are not possible in this model. Through the use of our optimal data structure for polytope membership, we achieve the following improved trade-off, represented in Figure 3.2(d). The product of the ε-dependencies in the storage and query time remain close to $O(1/\varepsilon^{d/2})$ throughout the tradeoff.

Theorem 3.1. *Given a set S of n points in $\mathbb{R}^d$, an approximation parameter $\varepsilon > 0$, and m such that $\log \frac{1}{\varepsilon} \leq m \leq 1/(\varepsilon^{d/2} \log \frac{1}{\varepsilon})$, there is a data structure that can answer*

Euclidean ε-approximate nearest neighbor queries with

$$\text{Query time: } O\left(\log n + \frac{\log \frac{1}{\varepsilon}}{m \cdot \varepsilon^{\frac{d}{2}}}\right)$$
$$\text{Storage: } O\big(n\, m\big).$$

The preprocessing time is

$$O\left(n \log n \, \log \frac{1}{\varepsilon} + \frac{n\, m}{\varepsilon^{\alpha}}\right),$$

where $\alpha > 0$ is an arbitrarily small constant.

By setting m to its upper limit, the storage needed to answer ε-approximate nearest neighbor queries for a set of n points in polylogarithmic time is reduced to $O(n/\varepsilon^{d/2})$. This halves the exponent in the ε-dependency of the existing space bound of roughly $O(n/\varepsilon^d)$, which has stood for 15 years [64].

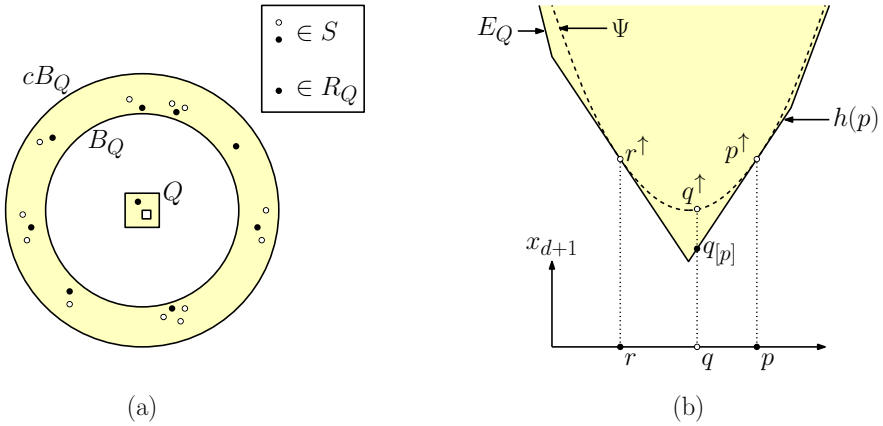
The connection between the polytope membership problem and ANN has been noted before by Clarkson [49]. Unlike Clarkson's, our results hold for point sets with arbitrary aspect ratios. The reduction from approximate nearest neighbor searching to approximate polytope membership is based on the approximate Voronoi diagram (AVD) construction from [21]. The AVD employs a height balanced variant of a quadtree, a balanced box decomposition (BBD) tree [22] to be precise. Each cell of a BBD tree corresponds to the set theoretic difference of two quadtree cells, an *outer box* and an optional *inner box*. Each leaf cell of the tree stores a set of *representative points* with the property that for any query point q lying within this cell, at least one of these representatives is an ε-nearest neighbor of q. A query is answered by locating the leaf cell that contains the query point and then determining the nearest representative from this cell (by brute force). The AVD's space is dominated by the total number of representatives over all the leaf cells. The query time is the height of the tree plus the number of representatives in the leaf cell.

The following lemma is central to our reduction and follows easily from the proofs of Lemmas 6.1 and 8.1 in [21]. Given a cell Q in a BBD tree, let $\text{size}(Q)$ denote the side length of Q and let B_Q be the ball of radius $2\sqrt{d}\,\text{size}(Q)$ whose center coincides with the center of Q's outer box. The lemma is illustrated in Figure 3.3(a).

Lemma 3.2. *Let $0 < \varepsilon \leq 1/2$ be a real parameter and S be a set of n points in $\mathbb{R}^d$. It is possible to construct a BBD tree T with $O(n \log \frac{1}{\varepsilon})$ nodes, where each leaf cell Q stores a subset $R_Q \subset S$ satisfying the following properties:*

- *(i) R_Q is an ε-representative set for Q (that is, for any point q in Q, one of the points in R_Q is an approximate nearest neighbor of q).*
- *(ii) At most one point of R_Q is contained in the ball B_Q, and the remaining points of R_Q are contained in the annulus $cB_Q \setminus B_Q$ for some constant c.*
- *(iii) The total number of representatives over all the leaf cells is $O(n \log \frac{1}{\varepsilon})$.*

Moreover, it is possible to compute the tree T and the sets R_Q for all the leaf cells in total time $O(n \log n \log \frac{1}{\varepsilon})$ and the cell that contains a query point can be located in $O(\log n + \log\log \frac{1}{\varepsilon})$ time.



(a) (b)

Figure 3.3: Approximate nearest neighbor searching: (a) Lemma 3.2 (black points are members of R_Q), (b) the lifting transformation. (Note that the figure is not drawn to scale, and the paraboloid in (b) has been translated to aid legibility.)

In order to connect Lemma 3.2 with approximate polytope membership queries, it is useful to define a special case of approximate nearest neighbor searching. The input for an approximate nearest neighbor searching data structure is a set S of data points and an approximation parameter ε. Given a constant $\sigma > 0$, *σ-well-separated* approximate nearest neighbor searching is defined as follows. Let Q_S and Q_q be two hypercubes of side length r and at distance at least σr from each other. In the σ-well-separated version we have the data points S inside Q_S and the query points inside Q_q. Data structures for the well-separated version are much more efficient than for the unrestricted version because we can remove the dependency on the number of points n altogether.

Lemma 3.3. *Let $0 < \varepsilon \leq 1/2$ be a real parameter, $\sigma > 0$ be a constant, and S be a set of n points in $\mathbb{R}^d$. Given a data structure for approximate polytope membership in d-dimensional space with query time $t_d(\varepsilon)$, storage $s_d(\varepsilon)$, and preprocessing time $O(n \log \frac{1}{\varepsilon} + b_d(\varepsilon))$ it is possible to preprocess S into a σ-well-separated ANN data structure with*

$$\textit{Query time: } O\left(t_{d+1}(\varepsilon) \cdot \log \frac{1}{\varepsilon}\right)$$
$$\textit{Storage: } O\big(s_{d+1}(\varepsilon)\big).$$

The preprocessing time is

$$O\left(n \log \frac{1}{\varepsilon} + b_{d+1}(\varepsilon)\right).$$

The reduction uses the well known *lifting transformation* [7, 57]. Let $(x_1, \ldots, x_{d+1})$ denote the coordinates of $\mathbb{R}^{d+1}$, and let us think of $(d+1)$st coordinate axis as being directed vertically upwards. Let Ψ denote the paraboloid $x_{d+1} = \sum_{i=1}^{d} x_i^2$. Given a point $p \in \mathbb{R}^d$, let $p^\uparrow$ denote the vertical projection of p onto Ψ (see Figure 3.3(b)), and let $h(p)$ denote the hyperplane tangent to Ψ at $p^\uparrow$. That is, the points of $h(p)$ satisfy $x_{d+1} = \sum_{i=1}^{d} 2p_i x_i - \|p\|^2$. Given $q \in \mathbb{R}^d$, let $q_{[p]}$ denote the point on $h(p)$ hit by a vertical ray shot downwards from $q^\uparrow$. A straightforward consequence of the definition of Ψ is that the squared distance between q and p in $\mathbb{R}^d$ is equal to the length of this vertical segment, that is, $\|qp\|^2 = \|q^\uparrow q_{[p]}\|$.

This suggests the following approach to computing the closest representative point through vertical ray shooting. Consider the (unbounded) convex polyhedron that results by taking the upper envelope of the hyperplanes $h(p)$ associated with the lifted representatives. Given the query point $q \in \mathbb{R}^d$, a ray shot vertically downward from $q^\uparrow$ hits some facet of this polyhedron. It follows from the above remarks, that the representative associated with this hyperplane is the closest to q. We can simulate ray shooting by applying polytope membership queries in concert with binary search. Of course, some care will be needed to map this problem into our context, which assumes a bounded polytope and approximation.

Combining the reduction from Lemma 3.3 with Theorem 2.10 we have:

Lemma 3.4. *Given a set S of n points in $\mathbb{R}^d$, an approximation parameter $\varepsilon > 0$, and a constant $\sigma > 0$, there is a data structure that can answer σ-well-separated Euclidean ε-approximate nearest neighbor queries with*

$$\textit{Query time: } O\left(\log^2 \frac{1}{\varepsilon}\right)$$
$$\textit{Storage: } O\left(\left(\frac{1}{\varepsilon}\right)^{\frac{d}{2}}\right).$$

The preprocessing time is

$$O\left(n \log \frac{1}{\varepsilon} + \left(\frac{1}{\varepsilon}\right)^{\frac{d}{2}+\alpha}\right),$$

where $\alpha > 0$ is an arbitrarily small constant.

Combining Lemma 3.2 with Lemma 3.3, we obtain the following theorem.

Theorem 3.5. *Let $0 < \varepsilon \leq 1/2$ be a real parameter and S be a set of n points in $\mathbb{R}^d$. Given a data structure for approximate polytope membership in d-dimensional space with query time at most $t_d(\varepsilon)$ and storage $s_d(\varepsilon)$, and preprocessing time $O(n \log \frac{1}{\varepsilon} + b_d(\varepsilon))$ it is possible to preprocess S into an ANN data structure with*

$$\textit{Query time: } O\left(\log n + t_{d+1}(\varepsilon) \cdot \log \frac{1}{\varepsilon}\right)$$
$$\textit{Storage: } O\left(n \log \frac{1}{\varepsilon} + n \frac{s_{d+1}(\varepsilon)}{t_{d+1}(\varepsilon)}\right)$$
$$\textit{Preprocessing: } O\left(n \log n \log \frac{1}{\varepsilon} + n \frac{b_{d+1}(\varepsilon)}{t_{d+1}(\varepsilon)}\right).$$

Proof. Construct a BBD-tree and the sets R_Q as in Lemma 3.2. For the nodes with $|R_Q| \leq t_{d+1}(\varepsilon) \log \frac{1}{\varepsilon}$, simply store the set R_Q and answer the corresponding queries by brute force. For the nodes with $|R_Q| > t_{d+1}(\varepsilon) \log \frac{1}{\varepsilon}$, use the construction from Lemma 3.4. The cell containing the query point can be located in $O(\log n + \log \log \frac{1}{\varepsilon})$ time. The bound for the total storage follows from the fact that the total number of representatives $\sum |R_Q| = O(n \log \frac{1}{\varepsilon})$ and the number of nodes with more than $t_{d+1}(\varepsilon) \log \frac{1}{\varepsilon}$ representatives is $O(n/t_{d+1}(\varepsilon))$. □

Combining Theorem 3.5 with Theorem 2.10, we obtain Theorem 3.1.

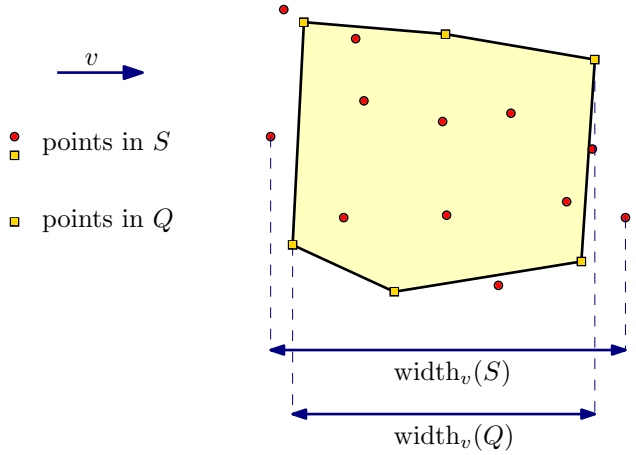


Figure 3.4: Illustration of an ε-kernel.

3.2 Kernel

Given a set S of n points in $\mathbb{R}^d$ and an approximation parameter $\varepsilon > 0$, an *ε-coreset* is an (ideally small) subset of S that approximates some measure over S (see [5] for a survey). Given a nonzero vector $v \in \mathbb{R}^d$, the *directional width* of S in direction v, $\text{width}_v(S)$ is the minimum distance between two hyperplanes that enclose S and are orthogonal to v. A *coreset for the directional width* (also known as an *ε-kernel* and as a *coreset for the extent measure*) is a subset $Q \subseteq S$ such that $\text{width}_v(Q) \geq (1-\varepsilon)\,\text{width}_v(S)$, for all $v \in \mathbb{R}^d$ (see Figure 3.4). Intuitively, the convex hull of an ε-kernel provides a good approximation to the convex hull of the entire set of points, and this fact explains their utility in approximating many extent measures efficiently by solving the problems on a much smaller subset. For example, kernels have been used to obtain approximation algorithms for several problems such as diameter, minimum width, convex hull volume, minimum enclosing cylinder, minimum enclosing annulus, and minimum-width cylindrical shell [4, 5].

The concept of ε-kernels was introduced by Agarwal et al. [4]. The existence of ε-kernels with $O(1/\varepsilon^{(d-1)/2})$ points is implied by the works of Dudley [54] and Bronshteyn and Ivanov [35], and this is known to be optimal in the worst case. Agarwal et al. [4] demonstrated how to compute such a kernel in $O(n + 1/\varepsilon^{3(d-1)/2})$ time, which reduces to $O(n)$ when $n = \Omega(1/\varepsilon^{3(d-1)/2})$. While less succinct ε-kernels with $O(1/\varepsilon^{d-1})$ points can be constructed in $O(n)$ time for all n [4, 30], no linear-time algorithm is known to build an ε-kernel of optimal size. Hereafter, we use the term ε-kernel to refer exclusively to an ε-kernel of size $O(1/\varepsilon^{(d-1)/2})$.

Chan [43] showed that an ε-kernel can be constructed in $O((n + 1/\varepsilon^{d-2}) \log \frac{1}{\varepsilon})$ time, which is nearly linear when $n = \Omega(1/\varepsilon^{d-2})$. He posed the open problem of obtaining a faster algorithm. A decade later, Arya and Chan [9] showed how to build an ε-kernel in roughly $O(n + \sqrt{n}/\varepsilon^{d/2})$ time using discrete Voronoi diagrams. In this paper, we attain the following near-optimal construction time.

Theorem 3.6. *Given a set S of n points in $\mathbb{R}^d$ and an approximation parameter $\varepsilon > 0$, it is possible to construct an ε-kernel of S with $O(1/\varepsilon^{(d-1)/2})$ points in $O(n \log \frac{1}{\varepsilon} + 1/\varepsilon^{(d-1)/2+\alpha})$ time, where α is an arbitrarily small positive constant.*

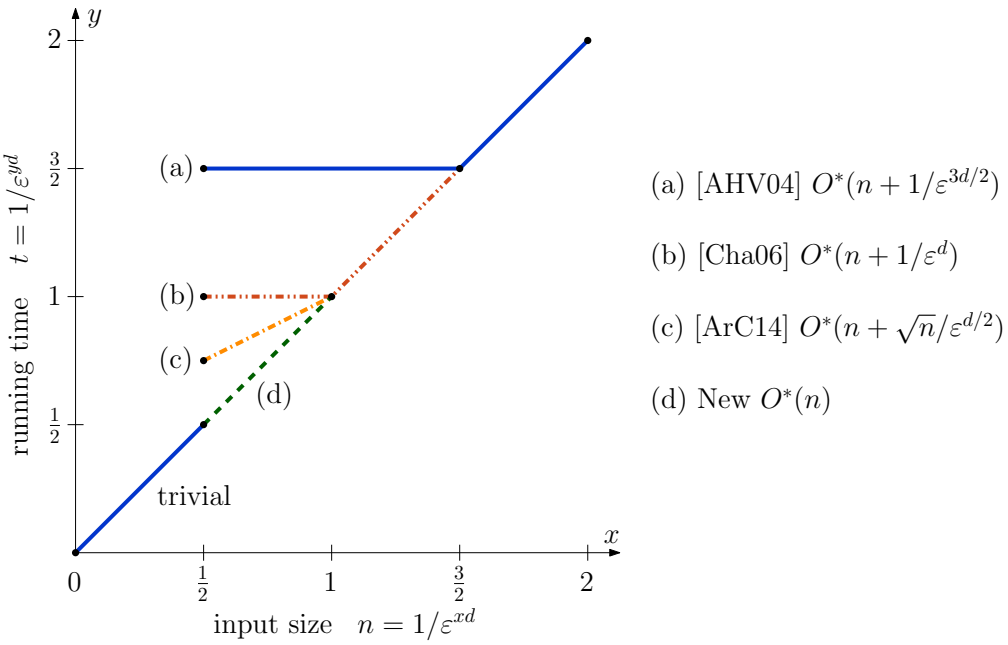


Figure 3.5: Running time of multiple ε-kernel algorithms as a function of the input size. The $O^*(\cdot)$ notation hides ε-dependencies of at most $1/\varepsilon^c$ for some constant c that does not depend on d.

Because the worst-case output size is $O(1/\varepsilon^{(d-1)/2})$, we may assume that n is at least this large, for otherwise we can simply take S itself to be the kernel. Since $1/\varepsilon^\alpha$ dominates $\log \frac{1}{\varepsilon}$, the above running time can be expressed as $O(n/\varepsilon^\alpha)$, which is nearly linear given that α can be made arbitrarily small. The history of the running times as a function of the input size can be visualized in Figure 3.5

Concurrently and independently, Timothy Chan has reported complexity bounds that are very similar to our results [45]. His ε-kernel construction takes roughly $O(n\sqrt{1/\varepsilon} + 1/\varepsilon^{(d-1)/2+3/2})$ time. Remarkably, the computational techniques are very different, based on Chebyshev polynomials.

Our algorithm to compute an ε-kernel in time $O(n \log \frac{1}{\varepsilon} + 1/\varepsilon^{(d-1)/2+\alpha})$ (Theorem 3.6) is conceptually quite simple. Since the fastest algorithm known to build the whole hierarchy of Macbeath regions takes $O(n + 1/\varepsilon^{3(d-1)/2})$ time, we cannot afford to build the whole hierarchy. Instead, we use an approximation parameter $\delta = \varepsilon^{1/3}$ to build a δ-approximation hierarchy in $O(n + 1/\delta^{3(d-1)/2}) = O(n + 1/\varepsilon^{(d-1)/2})$ time. Navigating through this coarser hierarchy, we partition the n points among the leaf Macbeath ellipsoids in $O(n \log \frac{1}{\varepsilon})$ time, discarding points that are too far from the boundary. We then compute an (ε/δ)-kernel for the set of points in each leaf ellipsoid and return the union of the kernels computed.

Given an algorithm to compute an ε-kernel in $O(n \log \frac{1}{\varepsilon} + 1/\varepsilon^{t(d-1)})$ time, the previous procedure produces an ε-kernel in $O(n \log \frac{1}{\varepsilon} + 1/\varepsilon^{t'(d-1)})$ time where $t' = (4t + 1)/6$. Bootstrapping the construction a constant number of times, the value of t goes down from 1 to a value that is arbitrarily close to $1/2$. This discrepancy accounts for the $O(1/\varepsilon^\alpha)$ factors in our running times.

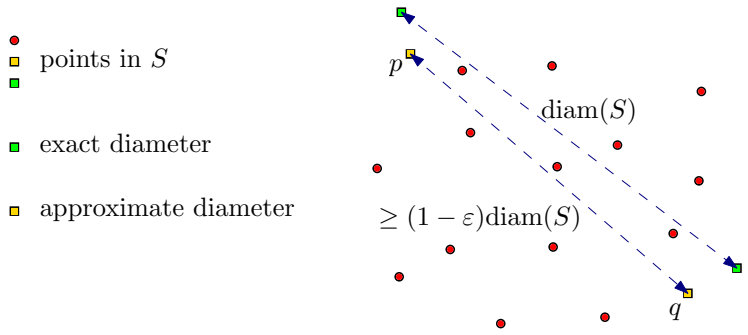


Figure 3.6: Diameter and approximate diameter.

3.3 Diameter

An important application of ε-kernels is to approximate the diameter of a point set. Given n data points, the *diameter* is defined to be the maximum distance between any two data points. An *ε-approximation* of the diameter is a pair of points whose distance is at least $(1-\varepsilon)$ times the exact diameter (see Figure 3.6). There are multiple algorithms to approximate the diameter [4,6,9,29,43]. The fastest running times are $O((n+1/\varepsilon^{d-2})\log\frac{1}{\varepsilon})$ [43] and roughly $O(n+\sqrt{n}/\varepsilon^{d/2})$ [9]. The algorithm from [43] essentially computes an ε-kernel Q and then determines the maximum value of $\text{width}_v(Q)$ among a set of $k = O(1/\varepsilon^{(d-1)/2})$ directions v by brute force [4]. Discrete Voronoi diagrams [9] permit this computation in roughly $O(n + \sqrt{n}/\varepsilon^{d/2})$ time. Therefore, combining the kernel construction of Theorem 3.6 with discrete Voronoi diagrams [9], we reduce n to $O(1/\varepsilon^{(d-1)/2})$ and obtain an algorithm to ε-approximate the diameter in roughly $O(n + 1/\varepsilon^{3d/4})$ time. However, we show that it is possible to obtain a much faster algorithm, as presented in the following theorem.

Theorem 3.7. *Given a set S of n points in $\mathbb{R}^d$ and an approximation parameter $\varepsilon > 0$, it is possible to compute an ε-approximation to the diameter of S in $O(n\log\frac{1}{\varepsilon}+1/\varepsilon^{(d-1)/2+\alpha})$ time.*

Applying approximate polytope membership in the dual space, we obtain a data structure for the following *ε-approximate directional width* problem, which is closely related to ε-kernels. Given a set S of n points in a constant dimension d and an approximation parameter $\varepsilon > 0$, the goal is to preprocess S to efficiently ε-approximate $\text{width}_v(S)$, for a nonzero query vector v. We prove the following result.

Lemma 3.8. *Given a set S of n points in $\mathbb{R}^d$ and an approximation parameter $\varepsilon > 0$, there is a data structure that can answer ε-approximate directional width queries with*

$$\textit{Query time: } O\left(\log^2\frac{1}{\varepsilon}\right)$$
$$\textit{Storage: } O\left(1/\varepsilon^{\frac{d-1}{2}}\right)$$

The preprocessing time is

$$O\left(n\log\frac{1}{\varepsilon}+\left(\frac{1}{\varepsilon}\right)^{\frac{d-1}{2}+\alpha}\right),$$

where $\alpha > 0$ is an arbitrarily small constant.

Proof. Given a polytope P (defined as the intersection of n halfspaces) that contains the origin O, we define a ray-shooting query (from the origin) as follows. Let v be a query direction and let r denote the ray emanating from O in direction v. The result of the query $q(P, v)$ is the length of $r \cap P$. In the ε-approximate version, any answer between $q(P, v)$ and $(1 + \varepsilon)q(P, v)$ is acceptable.

If we place the origin O in the center of the John ellipsoid of P, we have $q(P, -v) = \Theta(q(P, v))$ for all v. Thus, a constant approximation of $q(P, v)$ can be obtained by replacing P by its circumscribing John ellipsoid. We can then refine the approximation using binary search and approximate polytope membership queries. (To see this, consider the point $p \in \partial P$ that is hit by the ray, and let h be any supporting hyperplane at p. Consider the slab containing P that is bounded by this hyperplane and the parallel hyperplane on the opposite side of P. By properties of the John ellipsoid, the origin lies within a central region of the slab. It follows from basic geometry that if we expand the slab by ε times its width, the ratio between ray distances to the expanded slab boundary and the original slab boundary is $1 + O(\varepsilon)$. An ε-APM query with respect to P along this ray will achieve an approximation error that is no greater.) By a suitable adjustment to the constant factor, we can obtain an ε-approximation to $q(P, v)$ after $O(\log \frac{1}{\varepsilon})$ membership queries.

The polar body P^* (defined as the convex hull of n points) of P has the property that $\text{width}_v(P^*) = 1/q(P, v) + 1/q(P, -v)$. Therefore, we can ε-approximate the width of a set of points P^* using $O(\log \frac{1}{\varepsilon})$ approximate polytope membership queries on P and the lemma follows. □

Agarwal, Matoušek, and Suri [6] showed that the diameter of a point set S can be ε-approximated by computing the maximum width of S among $O(1/\varepsilon^{(d-1)/2})$ directions. Therefore, Theorem 3.7 follows immediately from Lemma 3.8.

3.4 Bichromatic Closest Pair

In the *bichromatic closest pair* (BCP) problem, we are given n points from two sets, designated red and blue, and we want to find the closest red-blue pair. In the *ε-approximate* version, the goal is to find a red-blue pair of points whose distance is at most $(1+\varepsilon)$ times the exact BCP distance. Approximations to the BCP problem were introduced in [69], and the most efficient randomized approximation algorithm runs in roughly $O(n/\varepsilon^{d/3})$ expected time [9]. The proof of the following result uses a reduction to well-separated approximate nearest neighbor searching that is based on [9, Theorem 3.2]. Since we invoke the algorithm from [69] which hashes the input points according to their coordinates in constant time, the model of computation needs to support integer division and randomization (e.g. unit-cost word RAM).

Theorem 3.9. *Given n red and blue points in $\mathbb{R}^d$ and an approximation parameter $\varepsilon > 0$, there is a randomized algorithm that computes an ε-approximation to the bichromatic closest pair in $O(n/\varepsilon^{d/4+\alpha})$ expected time.*

Proof. Let b denote the exact BCP distance. We obtain a constant approximation $b \leq a < 2b$ of the BCP distance in $O(n)$ expected time by running the randomized algorithm

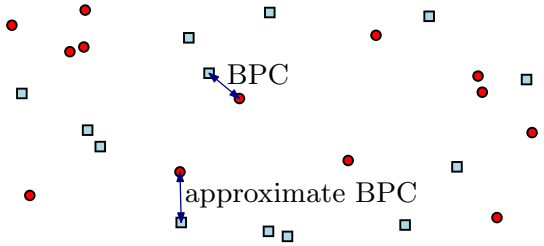


Figure 3.7: Exact and approximate bichromatic closest pair.

from [69]. Then, we build a grid with cells of diameter $a/4$ and partition the red points accordingly. Note that since $a/4 < b/2$, the BCP pair cannot be in the same grid cell, nor in two adjacent cells. The strategy of the algorithm is to partition the red points among the grid cells and to perform a constant number of well-separated approximate nearest neighbor queries for each blue point, returning the closest red-blue pair found. More precisely, for each blue point q, we perform an approximate nearest neighbor query among the grid cells Q_S that intersect the set theoretic difference of two balls of radii a and $a/2$ centered around q. These are the only grid cells that may contain the closest red point and, by a simple packing argument, the number of grid cells Q_S is constant. Since the grid cell Q_q that contains q cannot be adjacent to Q_S, it follows that the separation σ is at least 1.

To answer the queries efficiently, we separate the grid cells onto two types. If the number of red points in the cell is greater than $1/\varepsilon^{d/4}$, we say the cell is *heavy*, and otherwise we say the cell is *light*. Clearly, the number of heavy cells is $O(n \cdot \varepsilon^{d/4})$. We build well-separated approximate nearest-neighbor data structures for the heavy cells. Using Lemma 3.4, the total preprocessing time is $O(n/\varepsilon^{d/4+\alpha})$. For each light cell, we simply store the red points it contains and answer nearest neighbor queries by brute force in $O(1/\varepsilon^{d/4})$ time. Therefore, the total time spent answering queries is $O(n/\varepsilon^{d/4})$. □

3.5 Euclidean Trees

Given a set S of n points in $\mathbb{R}^d$, a *Euclidean minimum spanning tree* is the spanning tree with vertex set S that minimizes the *sum* of the edge lengths, while a *Euclidean minimum bottleneck tree* minimizes the *maximum* edge length. In the approximate version we respectively approximate the sum and the maximum of the edge lengths. A minimum spanning tree is a minimum bottleneck tree (although the converse does not hold). However, an approximation to the minimum spanning tree is not necessarily an approximation to the minimum bottleneck tree (see Figure 3.8). A recent approximation algorithm to the Euclidean minimum spanning tree takes roughly $O(n \log n + n/\varepsilon^2)$ time, regardless of the (constant) dimension [23]. On the other hand, the fastest algorithm to approximate the minimum bottleneck tree takes roughly $O((n \log n)/\varepsilon^{d/3})$ expected time [9]. The algorithm uses BCP to simultaneously attain an approximation to the minimum bottleneck and the minimum spanning trees.

An approximation to the Euclidean minimum spanning tree and minimum bottleneck tree can be computed by solving multiple BCP instances such that the sum of the number of points in all instances is $O(n \log n)$ [9, Theorem 4.1]. Applying this reduction together

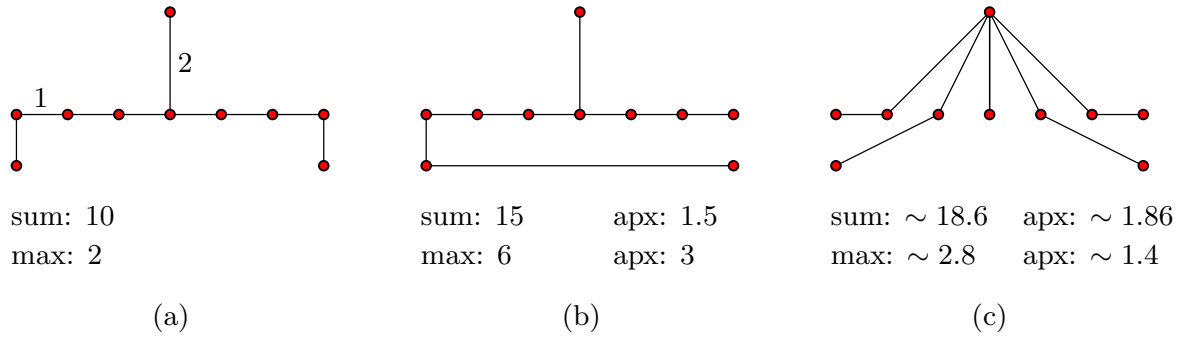


Figure 3.8: Let $\varepsilon = \frac{1}{2}$. (a) Exact Euclidean minimum spanning (and bottleneck) tree. (b) An ε-approximation of the minimum spanning tree that is not an ε-approximation of the minimum bottleneck tree. (c) An ε-approximation of the minimum bottleneck tree that is not an ε-approximation of the minimum spanning tree.

with Theorem 3.9, we prove the following theorem.

Theorem 3.10. *Given n points in $\mathbb{R}^d$ and an approximation parameter $\varepsilon > 0$, there is a randomized algorithm that computes a tree T that is an ε-approximation to both the Euclidean minimum bottleneck and the Euclidean minimum spanning trees in expected time $O((n \log n)/\varepsilon^{d/4+\alpha})$.*

Chapter 4

Conclusion

As seen in this dissertation, approximate polytope membership is a central problem in geometric approximation, both for its simple elegant definition and for its wide range of applications. Furthermore, Macbeath regions have a large algorithmic potential, which we are still discovering and I hope they will find many more applications in computational geometry.

The approximate membership formulation allowed us to obtain significant improvements to the complexity of several well studied problems. Ongoing work continues to confirm the utility of approximate polytope membership, with some applications that were not discussed in this dissertation. Hopefully many other applications will be discovered in the future. In Section 4.1, I describe ongoing work that was not finished in time to be part of this dissertation.

I decided to keep this dissertation centered on algorithmic problems, but during our research we obtained multiple discrete geometry results about the complexity of approximating polytopes. These results were not part of our original goals, but a fortunate product of the several techniques we developed. I briefly discuss these works in Section 4.2.

Finally I present open problems in Section 4.3.

4.1 Ongoing Work

Approximation problems involving a single convex body in d-dimensional space have received a great deal of attention in the computational geometry community [4,14–17,43,45]. In contrast, works involving multiple convex bodies are generally limited to dimensions $d \leq 3$ and/or do not consider approximation [2, 28, 60, 62, 84]. Next, we consider two problems involving multiple convex bodies: to detect if two polytopes intersect and to compute their Minkowski sum. We also present the impact of these results to the problem of approximating the width of a single convex body.

Polytope Intersection. Barba and Langerman [28] recently obtained a major result for exact polytope intersection. They showed how to preprocess convex polytopes in $\mathbb{R}^d$ so that given two such polytopes that have been subject to affine transformations, it can be determined whether they intersect each other in logaritmic time. However, the preprocessing time and storage grow as the combinatorial complexity of the polytope raised to the power $\lfloor d/2 \rfloor$. Since the combinatorial complexity of a polytope with n

vertices can be as high as $\Theta(n^{\lfloor d/2 \rfloor})$, the storage upper bound is roughly $O(n^{d^2/4})$. This high complexity motivates the study of approximations to the problem.

Let $K \subset \mathbb{R}^d$ be a convex polytope. Given a nonzero vector $v \in \mathbb{R}^d$, the *directional width* of K in direction v, $\text{width}_v(K)$ is the minimum distance between two hyperplanes that enclose K and are orthogonal to v. An *ε-approximation* of K is a polytope $K_\varepsilon \subset K$ such that $\text{width}_v(K_\varepsilon) \geq (1-\varepsilon)\,\text{width}_v(K)$, for all $v \in \mathbb{R}^d$. An *ε-kernel* of K is a an ε-approximation of K whose vertices are a subset of the vertices of K and the number of vertices is $O(1/\varepsilon^{(d-1)/2})$, which is sufficient and sometimes necessary.

In the approximate version of polytope intersection, we are given a parameter $\varepsilon > 0$ and independently preprocess two polytopes $A, B \subset \mathbb{R}^d$ into data structures such that we can *approximately* answer if $A \cap B = \emptyset$, where A and B have been subject to affine transformations. The answer to an *approximate polytope intersection query* is defined as follows. If *for all* valid ε-approximations $A_\varepsilon, B_\varepsilon$ of A, B respectively we have $A_\varepsilon \cap B_\varepsilon \neq \emptyset$, then we must answer *yes*. On the other hand, if *for all* valid ε-approximations $A_\varepsilon, B_\varepsilon$ we have $A_\varepsilon \cap B_\varepsilon = \emptyset$, then we must answer *no*. Otherwise, either answer is acceptable. We prove the following theorem.

Theorem 4.1. *Given a parameter $\varepsilon > 0$ and two polytopes $A, B \subset \mathbb{R}^d$, we can independently preprocess each polytope into a data structure in order to answer approximate polytope intersection queries with query time $O(\text{polylog}\,\frac{1}{\varepsilon})$, storage $O(1/\varepsilon^{(d-1)/2})$, and preprocessing time $O(n \log \frac{1}{\varepsilon} + 1/\varepsilon^{(d-1)/2+\alpha})$, where α is an arbitrarily small positive constant.*

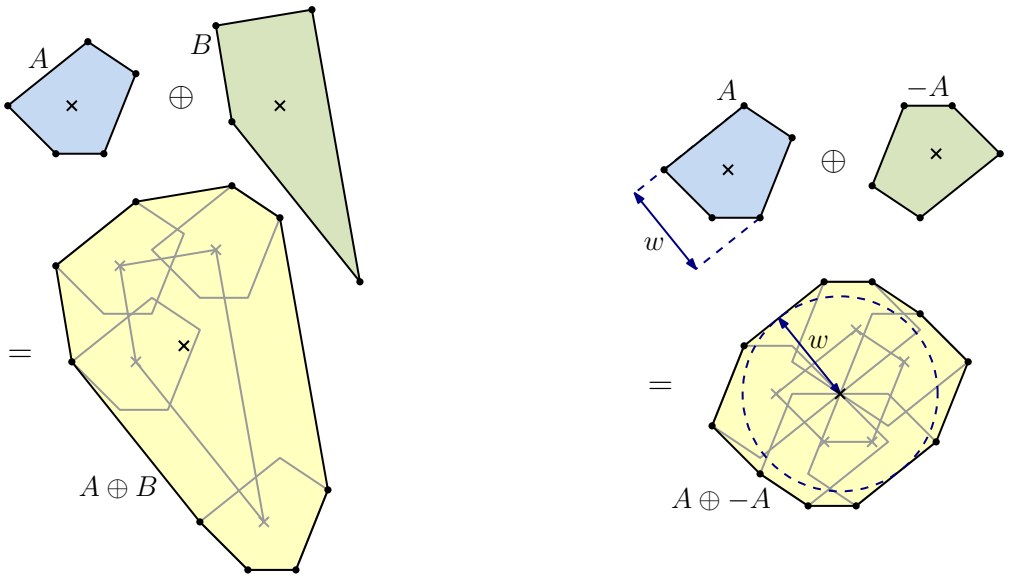
The data structure is nearly optimal since we need $\Omega(1/\varepsilon^{(d-1)/2})$ bits to store an ε-approximation of a polytope in the worst case [16].

Minkowski Sum. Given two convex bodies $A, B \subset \mathbb{R}^d$, the *Minkowski sum* $A \oplus B$ is defined as $\{p + q : p \in A,\ q \in B\}$ (see Figure 4.1(a) for an example). Minkowski sums are widely studied due to their applications in motion planning [63], computer aided design [84], computational biology [78], satellite layout [32], and image processing [68]. Furthermore, there is a large theoretical interest in Minkowski sums in both discrete geometry [3,60,66] and complexity of algorithms [1,82]. Since the time required to exactly compute the Minkowski sum of two convex polytopes with n vertices for $d \geq 3$ is $\Omega(n^2)$, different communities studied algorithms to compute approximations to the Minkowski sum in three dimensions [2, 62, 84].

In this paper, we show how to approximate the Minkowski sum of two convex polytopes in near-optimal time.

Theorem 4.2. *Given two convex polytopes $A, B \subset \mathbb{R}^d$ (represented by either n vertices or n bounding hyperplanes) and an approximation parameter $\varepsilon > 0$, it is possible to construct ε-approximation of $A \oplus B$ with $O(1/\varepsilon^{(d-1)/2})$ vertices or alternatively $O(1/\varepsilon^{(d-1)/2})$ facets in $O(n \log \frac{1}{\varepsilon} + 1/\varepsilon^{(d-1)/2+\alpha})$ time, where α is an arbitrarily small positive constant.*

Width. The *width* of a set of n points is the minimum over all directional widths, while the *diameter* is the maximum over all directional widths. After several subsequent improvements [4, 6, 9, 29, 43], near-optimal algorithms to ε-approximate the diameter in roughly $O(n + 1/\varepsilon^{d/2})$ time have been independently discovered by Chan [45] and the authors [14]. Surprisingly, these new works offered no improvement to the algorithms



(a) (b)

Figure 4.1: (a) Minkowski sum of two convex polygons. (b) Relationship between width and Minkowski sum.

that approximate the width, which remained a major open problem [45]. The fastest known algorithms date from over a decade ago and have a running time of roughly $O(n + 1/\varepsilon^{d-1})$ [41, 43].

Agarwal *et al.* [2] showed that the width of a convex body K is equal to the minimum distance from the origin to the boundary of the convex body $K \oplus (-K)$ (see Figure 4.1(b)). Using Theorem 4.2, we can approximate the width by computing an ε-kernel of $K \oplus (-K)$ represented by bounding hyperplanes and then determining the closest point to the origin among all bounding hyperplanes. The following theorem presents this result.

Theorem 4.3. *Given a set S of n points in $\mathbb{R}^d$ and an approximation parameter $\varepsilon > 0$, it is possible to compute an ε-approximation to the width of S in $O(n \log \frac{1}{\varepsilon} + 1/\varepsilon^{(d-1)/2+\alpha})$ time, where α is an arbitrarily small positive constant.*

On a different line of research, we consider approximate nearest neighbor searching in non-Euclidean metrics. The current reduction from approximate nearest neighbor searching to approximate polytope membership is based on a lifting transformation. This lifting transformation linearizes the Euclidean metric while increasing the dimension by only one unit. Such a transformation is not possible for non-Euclidean metrics, and therefore we need to rely on other techniques.

We noticed that in order to use our polytope membership data structure, we need convexity, but except for the preprocessing algorithm, no assumption of the convex body actually being a polytope is necessary. Therefore, we are free to use a transformation that produces a convex body with curved faces. In this ongoing work we consider the problems that can be solved by this key insight: convexify instead of linearize. When dealing with approximate nearest neighbor searching, smooth metrics seem to allow data structures

with lower complexity than metrics with corners (such as L_1 and L_∞). We believe that we should even be able to assign different metrics to different data points.

4.2 Polytope Approximation

Approximating convex bodies by polytopes is a fundamental problem, which has been extensively studied in the literature. (See Bronstein [36] for a recent survey.) At issue is the minimum number of vertices (alternatively, the minimum number of facets) needed in an approximating polytope for a given error $\varepsilon > 0$. Consider a convex body K in Euclidean d-dimensional space. A polytope P is said to *ε-approximate* K if the Hausdorff distance [36] between K and P is at most ε. Throughout, we will restrict attention to the Hausdorff metric, and we assume that the dimension d is a constant.

Our interest is in establishing bounds on the number of facets needed to approximate general convex bodies. Approximation bounds are of two common types. In both cases, it is shown that there exists $\varepsilon_0 > 0$ such that the bounds hold for all $\varepsilon \leq \varepsilon_0$. In the first type, which we call *nonuniform bounds*, the value of ε_0 depends on K (for example, on K's maximum curvature). Such bounds are often stated as holding "in the limit" as ε approaches zero, or equivalently as the complexity of the approximating polytope approaches infinity. Examples include bounds by Gruber [61], Clarkson [50], and others [33, 74, 81, 83].

In the second type, which we call *uniform bounds*, the value of ε_0 is independent of K. For example, these include the results of Dudley [54] and Bronshteyn and Ivanov [35]. These bounds hold without any smoothness assumptions. Dudley showed that, for $\varepsilon \leq 1$, any convex body K can be ε-approximated by a polytope P with $O((\mathrm{diam}(K)/\varepsilon)^{(d-1)/2})$ facets. Bronshteyn and Ivanov showed the same bound holds for the number of vertices. Constants hidden in the O-notation depend only on d.

The approximation bounds of both Dudley and Bronshteyn and Ivanov are tight up to constant factors (specifically when K is a Euclidean ball). These bounds may be significantly suboptimal if K is skinny, however. Let $\mathrm{area}(K)$ denote the $(d-1)$-dimensional Hausdorff measure of ∂K. We show that, under the assumption that the width of the body in any direction is at least ε, there exists an ε-approximating polytope whose number of facets is $O(\sqrt{\mathrm{area}(K)}/\varepsilon^{(d-1)/2})$. For a given diameter, the surface area of a convex body is maximized for a Euclidean ball, implying that $\mathrm{area}(K) = O(\mathrm{diam}(K)^{d-1})$. Thus, this bound is tight in the worst case.

Theorem 4.4. *Consider real d-space, $\mathbb{R}^d$. There exists a positive ε_0 and constant c_d such that for any convex body $K \subset \mathbb{R}^d$ and any ε, $0 < \varepsilon \leq \varepsilon_0$, if the width of K in any direction is at least ε, then there exists an ε-approximating polytope P whose number of facets is at most*

$$c_d \sqrt{\mathrm{area}(K)}/\varepsilon^{(d-1)/2}.$$

Note that the width assumption seems to be a technical necessity. For example, consider a $(d-2)$-dimensional unit ball B embedded within $\mathbb{R}^d$, and let B' denote its Minkowski sum with a d-dimensional ball of radius $\delta \ll \varepsilon$. By the optimality of Dudley's bound for Euclidean balls, $\Omega(1/\varepsilon^{(d-3)/2})$ facets are needed to approximate B and hence to approximate B'. But, the surface area of B' can be made arbitrarily small as a function of δ.

The width assumption is not a fundamental impediment, however. If the body is of width less than ε in some direction, then by projecting the body onto a hyperplane orthogonal to this direction, it is possible to reduce the problem to a convex approximation problem in one lower dimension. This can be repeated until the body's width is sufficiently large in all remaining dimensions, and the stated bound can be applied in this lower dimensional subspace.

The Upper-Bound Theorem [75] implies that a polytope with n vertices (resp., facets) has total combinatorial complexity $O(n^{\lfloor d/2 \rfloor})$. By combinatorial complexity we mean the total number of faces of all dimensions of the polytope. It will simplify matters to assume that K has been uniformly scaled to unit diameter. Applying the upper bound theorem to the results of either Dudley or Bronshteyn and Ivanov directly yields a bound of $O(1/\varepsilon^{(d^2-d)/4})$ on the combinatorial complexity of an ε-approximating polytope. Better uniform bounds without d^2 in the exponent are known, however. Consider a uniform grid Ψ of points with spacing $\Theta(\varepsilon)$, and let P denote the convex hull of $\Psi \cap K$. It is easy to see that P is an ε-approximating polytope for K. The combinatorial complexity of any lattice polytope[1] is known to be $O(V^{(d-1)/(d+1)})$, where V is the volume of the polytope [8, 27]. This implies that P has combinatorial complexity $O(1/\varepsilon^{d(d-1)/(d+1)}) \approx O(1/\varepsilon^{d-2})$. While this is significantly better than the bound provided by the Upper-Bound Theorem, it is still much larger than the lower bound of $\Omega(1/\varepsilon^{(d-1)/2})$.

We show that this gap can be dramatically reduced. In particular, we establish an upper bound on the combinatorial complexity of convex approximation that is optimal up to a polylogarithmic factor in $1/\varepsilon$.

Theorem 4.5. *Let $K \subset \mathbb{R}^d$ be a convex body of unit diameter, where d is a fixed constant. For all sufficiently small positive ε (independent of K) there exists an ε-approximating convex polytope P to K of combinatorial complexity $O(1/\widehat{\varepsilon}^{(d-1)/2})$, where $\widehat{\varepsilon} = \varepsilon / \log \frac{1}{\varepsilon}$.*

This is within a factor of $O(\log^{(d-1)/2} \frac{1}{\varepsilon})$ of the aforementioned lower bound. Our construction of the approximating polytope uses a stratified placement of Macbeath regions, in which Macbeath regions of larger volume are kept closer to the boundary and Macbeath regions of smaller volume are moved towards the center of the convex body. To prove the bounds, we also employ a deterministic version of the witness-collector technique, developed recently by Devillers *et al.* [52], in the context of our stratified construction.

4.3 Open Problems

Macbeath Regions

- Lemma 2.7 gives an upper bound of $O(1/\varepsilon^{(d-1)/2})$ to the number of disjoint Macbeath regions of width ε inside a fat convex body of unit diameter. These Macbeath regions may however have very different volumes, ranging from $\Theta(\varepsilon^d)$ to $\Theta(\varepsilon)$. Because the regions are disjoint and at distance ε from the boundary, the total volume cannot exceed ε. Therefore, for $v \geq \varepsilon^{(d+1)/2}$, we have that the number of regions of volume v is $O(\varepsilon/v)$. We believe that the number of Macbeath regions of small volume should also be small since they are generated by portions of high curvature and a

[1] A *lattice polytope* is the convex hull of any set of points with integer coordinates.

convex body cannot have high curvature everywhere. Is it true that the number of Macbeath regions of volume $v < \varepsilon^{(d+1)/2}$ is $O(v/\varepsilon)$?

- The proof of Lemma 2.7 uses a pruning step. Is this pruning step really necessary, or can we show that the projections onto the Dudley ball of the center points of any two disjoint Macbeath regions will have distance $\Omega(\sqrt{\varepsilon})$?

Polytope Membership

- The best upper bound for the query time of the data structure produced by the split-reduce algorithm with storage $O(1/\varepsilon^{(d-1)/2})$ is roughly $O(1/\varepsilon^{(d-1)/8})$, while the best lower bound is roughly $\Omega(1/\varepsilon^{(d-1)/18})$. Can the analysis of the split-reduce algorithm be improved or a better lower bound be obtained?

- The data structure based on Macbeath regions for polytope membership has optimal storage space of $O(1/\varepsilon^{(d-1)/2})$. However, some polytopes may require much less storage. In particular, can we bound the storage as a function of the number of facets or the combinatorial complexity of the input polytope? This sensitive bound could potentially be used to obtain improved bounds for approximate nearest neighbor searching, since the reduction produces several polytopes with few facets.

Approximate Nearest Neighbor Searching

- Previous lower bounds for approximate nearest neighbor searching were based on a model of computation that only allowed fat decompositions of space (the AVD Model). These lower bounds have been beaten by the use of Macbeath regions. Can we prove lower bounds in a more general model of computation?

- A black-box construction can be used to reduce approximate k-nearest neighbor for constant k to approximate nearest neighbor. However, can we obtain better results for the k-nearest neighbor problem for non-constant values of k? Also, can we deal with the k-nearest neighbor problem directly, even for constant k?

Kernel

- The algorithm to compute an ε-kernel takes $O(n \log \frac{1}{\varepsilon} + 1/\varepsilon^{(d-1)/2+\alpha})$ time for any $\alpha > 0$. Can we get rid of the α term or the $\log \frac{1}{\varepsilon}$ factor?

Diameter Approximation

- Is it possible to approximate the diameter faster than $O(n + 1/\varepsilon^{(d-1)/2})$ time or can we prove some lower bound?

Bibliography

[1] P. K. Agarwal, E. Flato, and D. Halperin. Polygon decomposition for efficient construction of Minkowski sums. *Comput. Geom. Theory Appl.*, 21(1):39 – 61, 2002.

[2] P. K. Agarwal, L. J. Guibas, S. Har-Peled, A. Rabinovitch, and M. Sharir. Penetration depth of two convex polytopes in 3d. *Nordic J. of Computing*, 7(3):227–240, 2000.

[3] P. K. Agarwal, S. Har-Peled, H. Kaplan, and M. Sharir. Union of random Minkowski sums and network vulnerability analysis. *Discrete Comput. Geom.*, 52(3):551–582, 2014.

[4] P. K. Agarwal, S. Har-Peled, and K. R. Varadarajan. Approximating extent measures of points. *J. Assoc. Comput. Mach.*, 51:606–635, 2004.

[5] P. K. Agarwal, S. Har-Peled, and K. R. Varadarajan. Geometric approximation via coresets. In J. E. Goodman, J. Pach, and E. Welzl, editors, *Combinatorial and Computational Geometry*. MSRI Publications, 2005.

[6] P. K. Agarwal, J. Matoušek, and S. Suri. Farthest neighbors, maximum spanning trees and related problems in higher dimensions. *Comput. Geom. Theory Appl.*, 1(4):189–201, 1992.

[7] P. K. Agarwal and J. Matoušek. Ray shooting and parametric search. *SIAM J. Comput.*, 22(4):794–806, 1993.

[8] G. E. Andrews. A lower bound for the volumes of strictly convex bodies with many boundary points. *Trans. Amer. Math. Soc.*, 106:270–279, 1963.

[9] S. Arya and T. M. Chan. Better ε-dependencies for offline approximate nearest neighbor search, Euclidean minimum spanning trees, and ε-kernels. In *Proc. 30th Annu. Sympos. Comput. Geom.*, pages 416–425, 2014.

[10] S. Arya, G. D. da Fonseca, and D. M. Mount. A unified approach to approximate proximity searching. In *Proc. 18th Annu. European Sympos. Algorithms*, pages 374–385, 2010.

[11] S. Arya, G. D. da Fonseca, and D. M. Mount. Approximate polytope membership queries. In *Proc. 43rd Annu. ACM Sympos. Theory Comput.*, pages 579–586, 2011.

[12] S. Arya, G. D. da Fonseca, and D. M. Mount. Optimal area-sensitive bounds for polytope approximation. In *Proc. 28th Annu. Sympos. Comput. Geom.*, pages 363–372, 2012.

[13] S. Arya, G. D. da Fonseca, and D. M. Mount. Polytope approximation and the Mahler volume. In *Proc. 23rd Annu. ACM-SIAM Sympos. Discrete Algorithms*, pages 29–42, 2012.

[14] S. Arya, G. D. da Fonseca, and D. M. Mount. Near-optimal ε-kernel construction and related problems. In *Proc. 33rd Internat. Sympos. Comput. Geom.*, pages 10:1–15, 2017.

[15] S. Arya, G. D. da Fonseca, and D. M. Mount. On the combinatorial complexity of approximating polytopes. *Discrete Comput. Geom.*, 58(4):849–870, 2017.

[16] S. Arya, G. D. da Fonseca, and D. M. Mount. Optimal approximate polytope membership. In *Proc. 28th Annu. ACM-SIAM Sympos. Discrete Algorithms*, pages 270–288, 2017.

[17] S. Arya, G. D. da Fonseca, and D. M. Mount. Approximate polytope membership queries. *SIAM J. Comput.*, 47(1):1–51, 2018.

[18] S. Arya and T. Malamatos. Linear-size approximate Voronoi diagrams. In *Proc. 13th Annu. ACM-SIAM Sympos. Discrete Algorithms*, pages 147–155, 2002.

[19] S. Arya, T. Malamatos, and D. M. Mount. Space-efficient approximate Voronoi diagrams. In *Proc. 34th Annu. ACM Sympos. Theory Comput.*, pages 721–730, 2002.

[20] S. Arya, T. Malamatos, and D. M. Mount. The effect of corners on the complexity of approximate range searching. *Discrete Comput. Geom.*, 41:398–443, 2009.

[21] S. Arya, T. Malamatos, and D. M. Mount. Space-time tradeoffs for approximate nearest neighbor searching. *J. Assoc. Comput. Mach.*, 57:1–54, 2009.

[22] S. Arya and D. M. Mount. Approximate range searching. *Comput. Geom. Theory Appl.*, 17:135–163, 2000.

[23] S. Arya and D. M. Mount. A fast and simple algorithm for computing approximate Euclidean minimum spanning trees. In *Proc. 27th Annu. ACM-SIAM Sympos. Discrete Algorithms*, pages 1220–1233, 2016.

[24] S. Arya, D. M. Mount, N. S. Netanyahu, R. Silverman, and A. Y. Wu. An optimal algorithm for approximate nearest neighbor searching fixed dimensions. *J. Assoc. Comput. Mach.*, 45(6):891–923, 1998.

[25] S. Arya, D. M. Mount, and J. Xia. Tight lower bounds for halfspace range searching. *Discrete Comput. Geom.*, 47:711–730, 2012.

[26] I. Bárány. The technique of M-regions and cap-coverings: A survey. *Rend. Circ. Mat. Palermo*, 65:21–38, 2000.

[27] I. Bárány. Extremal problems for convex lattice polytopes: A survey. *Contemp. Math.*, 453:87–103, 2008.

[28] L. Barba and S. Langerman. Optimal detection of intersections between convex polyhedra. In *Proc. 26th Annu. ACM-SIAM Sympos. Discrete Algorithms*, pages 1641–1654, 2015.

[29] G. Barequet and S. Har-Peled. Efficiently approximating the minimum-volume bounding box of a point set in three dimensions. *J. Algorithms*, 38(1):91–109, 2001.

[30] J. L. Bentley, M. G. Faust, and F. P. Preparata. Approximation algorithms for convex hulls. *Commun. ACM*, 25(1):64–68, 1982.

[31] S. N. Bespamyatnikh. Dynamic algorithms for approximate neighbor searching. In *Proc. Eighth Canad. Conf. Comput. Geom.*, pages 252–257, 1996.

[32] J.-D. Boissonnat, E. De Lange, and M. Teillaud. Minkowski operations for satellite antenna layout. In *Proc. 13th Annu. Sympos. Comput. Geom.*, pages 67–76, 1997.

[33] K. Böröczky, Jr. Approximation of general smooth convex bodies. *Adv. Math.*, 153:325–341, 2000.

[34] H. Brönnimann, B. Chazelle, and J. Pach. How hard is halfspace range searching. *Discrete Comput. Geom.*, 10:143–155, 1993.

[35] E. M. Bronshteyn and L. D. Ivanov. The approximation of convex sets by polyhedra. *Siberian Math. J.*, 16:852–853, 1976.

[36] E. M. Bronstein. Approximation of convex sets by polytopes. *J. Math. Sci.*, 153(6):727–762, 2008.

[37] C. J. C. Burges. A tutorial on support vector machines for pattern recognition. *Data Min. Knowl. Discov.*, 2(2):121–167, 1998.

[38] T. M. Chan. Fixed-dimensional linear programming queries made easy. In *Proc. 12th Annu. Sympos. Comput. Geom.*, pages 284–290, 1996.

[39] T. M. Chan. Output-sensitive results on convex hulls, extreme points, and related problems. *Discrete Comput. Geom.*, 16:369–387, 1996.

[40] T. M. Chan. Approximate nearest neighbor queries revisited. *Discrete Comput. Geom.*, 20:359–373, 1998.

[41] T. M. Chan. Approximating the diameter, width, smallest enclosing cylinder, and minimum-width annulus. *Internat. J. Comput. Geom. Appl.*, 12:67–85, 2002.

[42] T. M. Chan. Closest-point problems simplified on the RAM. In *Proc. 13th Annu. ACM-SIAM Sympos. Discrete Algorithms*, pages 472–473, 2002.

[43] T. M. Chan. Faster core-set constructions and data-stream algorithms in fixed dimensions. *Comput. Geom. Theory Appl.*, 35(1):20–35, 2006.

[44] T. M. Chan. Optimal partition trees. In *Proc. 26th Annu. Sympos. Comput. Geom.*, pages 1–10, 2010.

[45] T. M. Chan. Applications of Chebyshev polynomials to low-dimensional computational geometry. In *Proc. 33rd Internat. Sympos. Comput. Geom.*, pages 26:1–15, 2017.

[46] B. Chazelle and D. P. Dobkin. Intersection of convex objects in two and three dimensions. *J. Assoc. Comput. Mach.*, 34:1–27, 1987.

[47] B. Chazelle and J. Matoušek. On linear-time deterministic algorithms for optimization problems in fixed dimension. *J. Algorithms*, 21:579–597, 1996.

[48] K. L. Clarkson. Algorithms for polytope covering and approximation. In *Proc. Third Internat. Workshop Algorithms Data Struct.*, pages 246–252, 1993.

[49] K. L. Clarkson. An algorithm for approximate closest-point queries. In *Proc. Tenth Annu. Sympos. Comput. Geom.*, pages 160–164, 1994.

[50] K. L. Clarkson. Building triangulations using ε-nets. In *Proc. 38th Annu. ACM Sympos. Theory Comput.*, pages 326–335, 2006.

[51] M. de Berg, O. Cheong, M. van Kreveld, and M. Overmars. *Computational Geometry: Algorithms and Applications*. Springer, 3rd edition, 2010.

[52] O. Devillers, M. Glisse, and X. Goaoc. Complexity analysis of random geometric structures made simpler. In *Proc. 29th Annu. Sympos. Comput. Geom.*, pages 167–176, 2013.

[53] D. P. Dobkin and D. G. Kirkpatrick. Fast detection of polyhedral intersection. *Theo. Comp. Sci.*, 27:241–253, 1983.

[54] R. M. Dudley. Metric entropy of some classes of sets with differentiable boundaries. *J. Approx. Theory*, 10(3):227–236, 1974.

[55] C. A. Duncan, M. T. Goodrich, and S. Kobourov. Balanced aspect ratio trees: Combining the advantages of k-d trees and octrees. *J. Algorithms*, 38:303–333, 2001.

[56] K. Dutta, A. Ghosh, B. Jartoux, and N. H. Mustafa. Shallow packings, semialgebraic set systems, Macbeath regions and polynomial partitioning. In *Proc. 33rd Internat. Sympos. Comput. Geom.*, pages 38:1–15, 2017.

[57] H. Edelsbrunner. *Algorithms in Combinatorial Geometry*. Springer-Verlag, 1987.

[58] J. Erickson, L. J. Guibas, J. Stolfi, and L. Zhang. Separation-sensitive collision detection for convex objects. In *Proc. Tenth Annu. ACM-SIAM Sympos. Discrete Algorithms*, pages 327–336, 1999.

[59] G. Ewald, D. G. Larman, and C. A. Rogers. The directions of the line segments and of the r-dimensional balls on the boundary of a convex body in Euclidean space. *Mathematika*, 17:1–20, 1970.

[60] E. Fogel, D. Halperin, and C. Weibel. On the exact maximum complexity of Minkowski sums of polytopes. *Discrete Comput. Geom.*, 42(4):654–669, 2009.

[61] P. M. Gruber. Asymptotic estimates for best and stepwise approximation of convex bodies I. *Forum Math.*, 5:521–537, 1993.

[62] X. Guo, L. Xie, and Y. Gao. Optimal accurate Minkowski sum approximation of polyhedral models. *Advanced Intelligent Computing Theories and Applications. With Aspects of Theoretical and Methodological Issues*, pages 179–188, 2008.

[63] D. Halperin, O. Salzman, and M. Sharir. Algorithmic motion planning. In C. D. Toth, J. O'Rourke, and J. E. Goodman, editors, *Handbook of Discrete and Computational Geometry*, Discrete Mathematics and its Applications. CRC Press, 2017.

[64] S. Har-Peled. A replacement for Voronoi diagrams of near linear size. In *Proc. 42nd Annu. IEEE Sympos. Found. Comput. Sci.*, pages 94–103, 2001.

[65] S. Har-Peled. *Geometric approximation algorithms*. Number 173 in Mathematical surveys and monographs. American Mathematical Society, 2011.

[66] S. Har-Peled, T. M. Chan, B. Aronov, D. Halperin, and J. Snoeyink. The complexity of a single face of a Minkowski sum. In *Proc. Seventh Canad. Conf. Comput. Geom.*, pages 91–96. Citeseer, 1995.

[67] F. John. Extremum problems with inequalities as subsidiary conditions. In *Studies and Essays Presented to R. Courant on his 60th Birthday*, pages 187–204. Interscience Publishers, Inc., New York, 1948.

[68] A. Kaul and J. Rossignac. Solid-interpolating deformations: construction and animation of pips. *Computers & graphics*, 16(1):107–115, 1992.

[69] S. Khuller and Y. Matias. A simple randomized sieve algorithm for the closest-pair problem. *Information and Computation*, 118(1):34–37, 1995.

[70] A. M. Macbeath. A theorem on non-homogeneous lattices. *Ann. of Math.*, 56:269–293, 1952.

[71] J. Matoušek and O. Schwarzkopf. On ray shooting in convex polytopes. *Discrete Comput. Geom.*, 10:215–232, 1993.

[72] J. Matoušek. Reporting points in halfspaces. *Comput. Geom. Theory Appl.*, 2:169–186, 1992.

[73] J. Matoušek. Linear optimization queries. *J. Algorithms*, 14(3):432–448, 1993.

[74] D. E. McClure and R. A. Vitalie. Polygonal approximation of plane convex bodies. *J. Math. Anal. Appl.*, 51:326–358, 1975.

[75] P. McMullen. The maximum numbers of faces of a convex polytope. *Mathematika*, 17:179–184, 1970.

[76] J. S. B. Mitchell and S. Suri. Separation and approximation of polyhedral objects. *Comput. Geom. Theory Appl.*, 5:95–114, 1995.

[77] N. H. Mustafa and S. Ray. Near-optimal generalisations of a theorem of Macbeath. In *Proc. 31st Internat. Sympos. on Theoret. Aspects of Comp. Sci.*, pages 578–589, 2014.

[78] L. Pachter and B. Sturmfels. *Algebraic statistics for computational biology*, volume 13. Cambridge university press, 2005.

[79] E. A. Ramos. Linear programming queries revisited. In *Proc. 16th Annu. Sympos. Comput. Geom.*, pages 176–181, 2000.

[80] Y. Sabharwal, S. Sen, and N. Sharma. Nearest neighbors search using point location in balls with applications to approximate Voronoi decompositions. *J. Comput. Sys. Sci.*, 72:955–977, 2006.

[81] R. Schneider. Polyhedral approximation of smooth convex bodies. *J. Math. Anal. Appl.*, 128:470–474, 1987.

[82] H. R. Tiwary. On the hardness of computing intersection, union and Minkowski sum of polytopes. *Discrete Comput. Geom.*, 40(3):469–479, 2008.

[83] L. F. Toth. Approximation by polygons and polyhedra. *Bull. Amer. Math. Soc.*, 54:431–438, 1948.

[84] G. Varadhan and D. Manocha. Accurate Minkowski sum approximation of polyhedral models. *Graphical Models*, 68(4):343–355, 2006.

Attached Article 1

Next, we include the arXiv version of the following journal paper.

S. Arya, G. D. da Fonseca, and D. M. Mount. Approximate polytope membership queries. *SIAM Journal on Computing (SICOMP)*, 47(1):1–51, 2018.

The paper combines the following two conference papers.

Sunil Arya, Guilherme D. da Fonseca, and David M. Mount. Polytope approximation and the Mahler volume. In *ACM-SIAM Symposium on Discrete Algorithms (SODA)*, pages 29–42, 2012.

Sunil Arya, Guilherme D. da Fonseca, and David M. Mount. Approximate polytope membership queries. In *ACM Symposium on Theory of Computing (STOC)*, pages 579–586, 2011.

SIAM J. COMPUT.
Vol. 47, No. 1, pp. 1–51

APPROXIMATE POLYTOPE MEMBERSHIP QUERIES*

SUNIL ARYA†, GUILHERME D. DA FONSECA‡, AND DAVID M. MOUNT§

Abstract. In the polytope membership problem, a convex polytope K in $\mathbb{R}^d$ is given, and the objective is to preprocess K into a data structure so that, given any query point $q \in \mathbb{R}^d$, it is possible to determine efficiently whether $q \in K$. We consider this problem in an approximate setting. Given an approximation parameter ε, the query can be answered either way if the distance from q to K's boundary is at most ε times K's diameter. We assume that the dimension d is fixed, and K is presented as the intersection of n halfspaces. Previous solutions to approximate polytope membership were based on straightforward applications of classic polytope approximation techniques by Dudley [*Approx. Theory*, 10 (1974), pp. 227–236] and Bentley, Faust, and Preparata [*Commun. ACM*, 25 (1982), pp. 64–68]. The former is optimal in the worst case with respect to space, and the latter is optimal with respect to query time. We present four main results. First, we show how to combine the two above techniques to obtain a simple space-time trade-off. Second, we present an algorithm that dramatically improves this trade-off. In particular, for any constant $\alpha \geq 4$, this data structure achieves query time roughly $O(1/\varepsilon^{(d-1)/\alpha})$ and space roughly $O(1/\varepsilon^{(d-1)(1-\Omega(\log\alpha)/\alpha)})$. We do not know whether this space bound is tight, but our third result shows that there is a convex body such that our algorithm achieves a space of at least $\Omega(1/\varepsilon^{(d-1)(1-O(\sqrt{\alpha})/\alpha)})$. Our fourth result shows that it is possible to reduce approximate Euclidean nearest neighbor searching to approximate polytope membership queries. Combined with the above results, this provides significant improvements to the best known space-time trade-offs for approximate nearest neighbor searching in $\mathbb{R}^d$. For example, we show that it is possible to achieve a query time of roughly $O(\log n + 1/\varepsilon^{d/4})$ with space roughly $O(n/\varepsilon^{d/4})$, thus reducing by half the exponent in the space bound.

Key words. polytope membership, nearest neighbor searching, geometric retrieval, space-time trade-offs, approximation algorithms, convex approximation, Mahler volume

AMS subject classifications. 52B99, 68W25, 68P05

DOI. 10.1137/16M1061096

1. Introduction.

Convex polytopes are key structures in many areas of mathematics and computation. In this paper, we consider a fundamental search problem related to convex polytopes. Let K denote a convex body in $\mathbb{R}^d$, that is, a closed, convex set of bounded diameter that has a nonempty interior. We assume that K is presented as the intersection of n closed halfspaces. (Our results generally hold for any representation that satisfies the access primitives given at the start of section 3.) The *polytope membership problem* is that of preprocessing K so that it is possible to determine efficiently whether a given query point $q \in \mathbb{R}^d$ lies within K. Throughout, we assume that the dimension d is a fixed constant that is at least 2.

It follows from standard results in projective duality that polytope membership is equivalent to answering halfspace emptiness queries for a set of n points in $\mathbb{R}^d$. In dimension $d \leq 3$, it is possible to build a data structure of linear size that can answer such queries in logarithmic time [30]. In higher dimensions, however, all known exact

*Received by the editors February 11, 2016; accepted for publication (in revised form) June 2, 2017; published electronically January 2, 2018.

http://www.siam.org/journals/sicomp/47-1/M106109.html

Funding: The first author was supported by the Research Grants Council of Hong Kong, China, under projects 610108 and 16200014. The second author was supported by grants from CNPq and FAPERJ. The third author was supported by NSF grant CCF-1618866.

†Department of Computer Science and Engineering, Hong Kong University of Science and Technology, Clear Water Bay, Kowloon, Hong Kong (arya@cse.ust.hk).

‡Université Clermont Auvergne and LIMOS, Clermont-Ferrand, France (fonseca@isima.fr).

§Department of Computer Science and Institute for Advanced Computer Studies, University of Maryland, College Park, MD 20742 (mount@umd.edu).

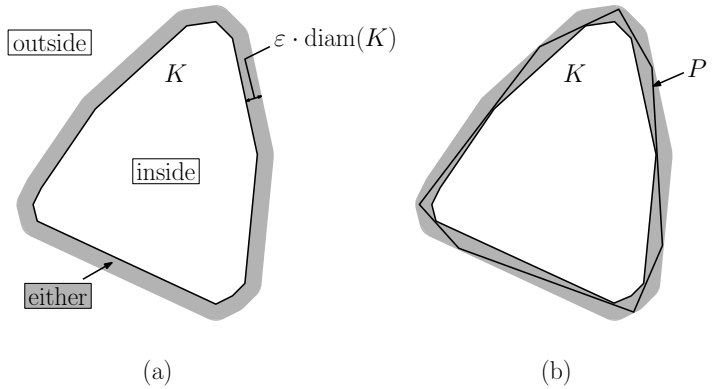


FIG. 1. *Approximate polytope membership:* (a) *problem formulation,* (b) *outer ε-approximation.*

data structures with roughly linear space have a query time of $\widetilde{\Omega}(n^{1-1/\lfloor d/2 \rfloor})$[1] [43], which is unacceptably high for many applications. Polytope membership is a special case of polytope intersection queries [25, 31, 13]. Barba and Langerman [13] showed that for any fixed d, it is possible to preprocess polytopes in $\mathbb{R}^d$ so that given two such polytopes that have been translated and rotated, it can be determined whether they intersect each other in time that is logarithmic in their total combinatorial complexity. However, the preprocessing time and space grow as the combinatorial complexity of the polytope raised to the power $\lfloor d/2 \rfloor$.

The lack of efficient exact solutions motivates the question of whether polytope membership queries can be answered approximately. Let ε be a positive real parameter, and let $\mathrm{diam}(K)$ denote K's diameter. Given a query point $q \in \mathbb{R}^d$, an *ε-approximate polytope membership query* returns a positive result if $q \in K$ and a negative result if the distance from q to its closest point in K is greater than $\varepsilon \cdot \mathrm{diam}(K)$, and it may return either result otherwise (see Figure 1(a)). Polytope membership queries, both exact and approximate, arise in many application areas, such as linear-programming and ray-shooting queries [20, 24, 46, 44, 42], nearest neighbor searching and the computation of extreme points [21, 28], collision detection [36], and machine learning [19].

Existing solutions to approximate polytope membership queries have been based on straightforward applications of classic polytope approximation techniques. We say that a polytope P is an *outer ε-approximation* of K if $K \subseteq P$, and the Hausdorff distance between P and K is at most $\varepsilon \cdot \mathrm{diam}(K)$ (see Figure 1(b)). An *inner ε-approximation* is defined similarly but with $P \subseteq K$. Dudley [32] showed that there exists an outer ε-approximating polytope for any bounded convex body in $\mathbb{R}^d$ formed by the intersection of $O(1/\varepsilon^{(d-1)/2})$ halfspaces, and Bronshteyn and Ivanov [17] proved an analogous bound on the number of vertices needed to obtain an inner ε-approximation. Both bounds are known to be asymptotically tight in the worst case (see, e.g., [18]). These results have been applied to a number of problems, for example, the construction of coresets [2]. By checking that a given query point lies within each of the halfspaces of Dudley's approximation, ε-approximate polytope membership queries can be answered with space and query time of $O(1/\varepsilon^{(d-1)/2})$.

[1]Throughout, we use $\widetilde{O}(\cdot)$ and $\widetilde{\Omega}(\cdot)$ as variants of $O(\cdot)$ and $\Omega(\cdot)$, respectively, that ignore logarithmic factors. We use "lg" to denote base-2 logarithm.

The principal contribution of this paper is to show that it is possible to achieve nontrivial space-time trade-offs for approximate polytope membership. In order to motivate our methods, in section 2 we present a simple space-time trade-off (stated in the following theorem), which is based on a straightforward combination of the approximations of Dudley [32] and Bentley, Faust, and Preparata [15]. Throughout, we will treat n and ε as asymptotic quantities, while the dimension d is a constant.

THEOREM 1.1 (simple trade-off). *Given a convex polytope K in $\mathbb{R}^d$, a positive approximation parameter ε, and a real parameter $\alpha \geq 2$, there is a data structure for ε-approximate polytope membership queries that achieves*

$$\textit{Query time: } O\left(1/\varepsilon^{\frac{d-1}{\alpha}}\right) \qquad \textit{Space: } O\left(1/\varepsilon^{(d-1)\left(1-\frac{1}{\alpha}\right)}\right).$$

The constant factors in the space and query time depend only on d (not on K, α, or ε).

We will strengthen this trade-off significantly in sections 3 and 4. We will show that it is possible to build a data structure with $O(1/\varepsilon^{(d-1)/2})$ space that allows polytope membership queries to be answered in roughly $O(1/\varepsilon^{(d-1)/8})$ time, thus reducing the exponent in the query time of Theorem 1.1 (for $\alpha = 2$) by $1/4$. Further, we will show that by iterating a suitable generalization of this construction, we can obtain the following trade-offs.

THEOREM 1.2. *Given a convex polytope K in $\mathbb{R}^d$, an approximation parameter $0 < \varepsilon \leq 1$, and a real constant $\alpha \geq 4$, there is a data structure for ε-approximate polytope membership queries that achieves*

$$\textit{Query time: } O\left(\left(\log \frac{1}{\varepsilon}\right)/\varepsilon^{\frac{d-1}{\alpha}}\right) \qquad \textit{Space: } O\Big(1/\varepsilon^{(d-1)\left(1-\frac{2\lfloor \lg \alpha \rfloor - 2}{\alpha}\right)}\Big).$$

The constant factors in the space and query time depend only on d and α (not on K or ε).

The above space bound is a simplification, and the exact bound is given in Lemma 6.4. Both bounds are piecewise linear in $1/\alpha$ (with breakpoints at powers of two), but the bounds of Lemma 6.4 are continuous as a function of α. The resulting space-time trade-off is illustrated in Figure 2(a). (The plot reflects the more accurate bounds.)

The above theorem is intentionally presented in a purely existential form. This is because our construction algorithm assumes the existence of a procedure that computes an ε-approximating polytope whose number of bounding hyperplanes is at most a constant factor larger than optimal. Unfortunately, we know of no efficient solution to this problem. In Lemma 7.6 we will show that if the input polytope is expressed as the intersection of n halfspaces, it is possible to build such a structure in time $O(n + 1/\varepsilon^{O(1)})$, such that the space and query times of the above theorem increase by an additional factor of $O(\log \frac{1}{\varepsilon})$.

Note that in contrast to many complexity bounds in the area of convex approximation, which hold only in the limit as ε approaches zero (see, e.g., [37, 40]), Theorems 1.1 and 1.2 hold for any positive $\varepsilon \leq 1$. The data structure of Theorem 1.2 is quite simple. It is based on a quadtree subdivision of space in which each cell is repeatedly subdivided until the combinatorial complexity of the approximating polytope within the cell is small enough to achieve the desired query time.

We do not know whether the upper bounds presented in Theorem 1.2 are tight for our algorithm. In section 8, we establish the following lower bound on the trade-off achieved by this algorithm.

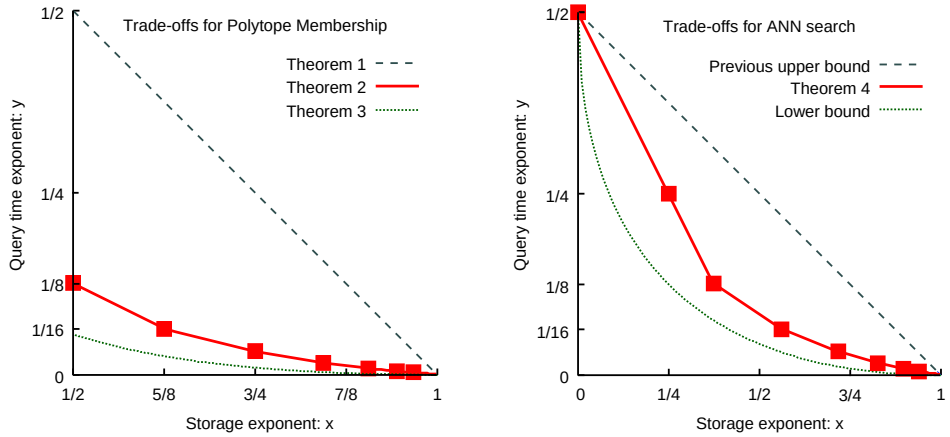


FIG. 2. *The multiplicative factors in the exponent of the* $1/\varepsilon$ *terms for* (a) *polytope membership queries and* (b) *approximate nearest neighbor (ANN) queries. Each point* (x, y) *represents a term of* $1/\varepsilon^{x(d\pm O(1))}$ *for storage and* $1/\varepsilon^{y(d\pm O(1))}$ *for query time, where the* $O(1)$ *term does not depend on* d.

THEOREM 1.3. *In any fixed dimension* $d \geq 2$ *and for any constant* $\alpha \geq 4$, *there exists a polytope such that for all sufficiently small positive* ε, *the data structure described in Theorem* 1.2 *when generated to achieve query time* $O(1/\varepsilon^{(d-1)/\alpha})$ *has space*

$$\Omega\left(1/\varepsilon^{(d-1)\left(1-\frac{2\sqrt{2\alpha}-3}{\alpha}\right)-1}\right).$$

Although α is not an asymptotic quantity, for the sake of comparing the upper and lower bounds, let us imagine that it is. For roughly the same query time, the α dependencies appearing in the exponents of the upper bounds on space are $(1-\frac{1}{\alpha})$ for Theorem 1.1 and $(1-\frac{\Omega(\log\alpha)}{\alpha})$ for Theorem 1.2, and the lower bound of Theorem 1.3 is roughly $(1-\frac{O(\sqrt{\alpha})}{\alpha})$. The trade-offs provided in these theorems are illustrated in Figure 2(a).

The second major contribution of this paper is to demonstrate that our trade-offs for approximate polytope membership queries imply significant improvements to the best known space-time trade-offs for approximate nearest neighbor searching. We are given a set X of n points in $\mathbb{R}^d$. Given any $q \in \mathbb{R}^d$, an *ε-approximate nearest neighbor* of q is any point of X whose distance from q is at most $(1+\varepsilon)$ times the distance to q's closest point in X. The objective is to preprocess X in order to answer such queries efficiently. Data structures for approximate nearest neighbor searching (in fixed dimensions) have been proposed by several authors [22, 33, 38, 28, 47]. The best space-time trade-offs [10] have query times roughly $O(1/\varepsilon^{d/\alpha})$ with storage roughly $O(n/\varepsilon^{d(1-2/\alpha)})$ for $\alpha \geq 2$ (see the dashed line in Figure 2(b)).

These results are based on a data structure called an *approximate Voronoi diagram* (AVD). In general, a data structure for approximate nearest neighbor searching is said to be in the *AVD model* if it has the general form of decomposition of space (generally a covering) by hyperrectangles of bounded aspect ratio, each of which is associated with a set of representative points. Given any hyperrectangle that contains the query point, at least one of these representatives is an ε-approximate nearest neighbor of the query point [10]. The AVD model is of interest because it is possible to prove lower bounds on the performance of such a data structure. In particular, the lower

bounds proved in [10] are shown in the dotted curve in Figure 2(b). By violating the AVD model, small additional improvements were obtained in [6].

Our improvements to approximate nearest neighbor searching are given in the following theorem.

THEOREM 1.4. *Let $0 < \varepsilon \leq 1$ be a real parameter, $\alpha \geq 1$ be a real constant, and X be a set of n points in $\mathbb{R}^d$. There is a data structure in the AVD model for approximate nearest neighbor searching that achieves*

$$\begin{aligned} \textit{Query time: } & O\left(\log n + (1/\varepsilon^{d/2\alpha}) \cdot \log^2 \frac{1}{\varepsilon}\right), \\ \textit{Space: } & O\left(n \cdot \max\left(\log \frac{1}{\varepsilon}, 1/\varepsilon^{d\left(\frac{1}{2}-\frac{1}{2\alpha}\right)}\right)\right) \textit{ for } 1 \leq \alpha < 2, \textit{ and} \\ & O\left(n/\varepsilon^{d\left(1-\frac{\lfloor \lg \alpha \rfloor}{\alpha}-\frac{1}{2\alpha}\right)}\right) \textit{ for } \alpha \geq 2. \end{aligned}$$

The constant factors in the space and query time depend only on d and α (not on ε).

The above space bound is a simplification of the more accurate bound given in Lemma 9.5. (Also see the remarks following the proof of this lemma for further minor improvements achievable by forgoing the AVD model.) As before, both bounds are piecewise linear in $1/\alpha$ (with breakpoints at powers of two), but the bounds of Lemma 9.5 are continuous as a function of α. The resulting space-time trade-off is illustrated in Figure 2(b). (The plot reflects the more accurate bounds of Lemma 9.5.)

As an example of the strength of the improvement that this offers, observe that in order for the existing AVD-based results to yield a query time of $\widetilde{O}(1/\varepsilon^{d/4})$ the required space would be roughly $\Omega(n/\varepsilon^{d/2})$. The exponent in the space bound is nearly twice that given by Theorem 1.4, which arises by setting $\alpha = 2$. The connection between the polytope membership problem and approximate nearest neighbor searching has been noted before by Clarkson [28]. Unlike Clarkson's, our results hold for point sets with arbitrary aspect ratios.

Our data structure is based on a simple quadtree-based decomposition of space. Let t denote the desired query time. We begin by preconditioning K so that it is fat and has at most unit diameter. We then employ a quadtree that hierarchically subdivides space into hypercube cells. The decomposition stops whenever we can declare that a cell is either entirely inside or outside of K, or (if it intersects K's boundary) it is locally approximable by at most t halfspaces. This procedure, called SplitReduce, is presented in section 3. Queries are answered by descending the quadtree to determine the leaf cell containing the query point, and (if not inside or outside) testing whether the query point lies within the approximating halfspaces.

Although the algorithm itself is very simple, the analysis of its space requirements is quite involved. In section 4, we begin with a simple analysis, which shows that it is possible to obtain a significant improvement over the Dudley-based approach (in particular, reducing the exponent in the query time by half with no increase in space). While this simple analysis introduces a number of useful ideas, it is not tight nor does it provide space-time trade-offs.

Our final analysis requires a deeper understanding of the local structure of the convex body's boundary. In section 5 we introduce local surface patches of K's boundary, called ε-dual caps. We relate the data structure's space requirements to the existence of a low cardinality hitting set of the dual caps. We present a two-pronged strategy for generating such a hitting set, one focused on dual caps of large surface area (intuitively corresponding to boundary regions of low curvature) and the other focused on

dual caps of small surface area (corresponding to boundary regions of high curvature). We show that simple random sampling suffices to hit dual caps of high surface area, and so the challenge is to hit the dual caps of low surface area. To do this, we show that dual caps of low surface area generate Voronoi patches on a hypersphere enclosing K of large surface area. We refer to this result as the *area-product bound*, which is stated in Lemma 5.2. This admits a strategy based on sampling points randomly on this hypersphere, and then projecting them back to their nearest neighbor on the surface of K.

The area-product bound is proved with the aid of a classical concept from the theory of convexity, called the *Mahler volume* [16, 48]. The Mahler volume of a convex body is a dimensionless quantity that involves the product of the body's volume and the volume of its polar body. We demonstrate that dual caps and their Voronoi patches exhibit a similar polar relationship. The proof of the area-product bound is quite technical and is deferred to section 10.

Armed with the area-product bound, in section 6 we establish our final bound on the space-time trade-offs of SplitReduce, which culminates in the proof of Theorem 1.2. In section 7 we present details on how the data structure is built and discuss preprocessing time. In section 8 we establish the lower bound result, which is stated in Theorem 1.3.

Finally, in section 9 we show how these results can be applied to improve the performance of approximate nearest neighbor searching in Euclidean space. It is well known that (exact) nearest neighbor searching can be reduced to vertical ray shooting to a polyhedron that results by lifting points in dimension d to tangent hyperplanes for a paraboloid in dimension $d+1$ [3, 34]. We show how to combine approximate vertical ray shooting (based on approximate polytope membership) with AVDs to establish Theorem 1.4.

2. Preliminaries. Throughout, we will use asymptotic notation to eliminate constant factors. In particular, for any positive real x, let $O(x)$ denote a quantity that is at most cx for some constant c. Define $\Omega(x)$ and $\Theta(x)$ analogously. We will sometimes introduce constants within a local context (e.g., within the statement of single lemma). To simplify notation, we will often use the same symbol "c" to denote such generic constants. Recall that we use "lg" to denote the base-2 logarithm. We will use "log" when the base does not matter. Some of our search algorithms involve integer grids, and for these we assume a model of computation that supports integer division.

Let K denote a full-dimensional convex body in $\mathbb{R}^d$, and let ∂K denote its boundary. For concreteness, we assume that K is represented as the intersection of n closed halfspaces. Our data structure can generally be applied to any representation that supports access primitives (i)–(iii) given at the start of section 3.

2.1. Absolute and relative approximations. Earlier, we defined approximation relative to K's diameter, but it will be convenient to define the approximation error in absolute terms. Given a positive real r, define $K \oplus r$ to be a set of points that lie within Euclidean distance r of K. We say that a polytope P is an *absolute ε-approximation* of a convex body K if

$$K \subseteq P \subseteq K \oplus \varepsilon.$$

When we wish to make the distinction clear, we refer to the definition in the introduction as a *relative approximation*. Henceforth, unless otherwise stated approximations are in the absolute sense.

In order to reduce the general approximation problem into a more convenient absolute form, we will transform K into a "fattened" body of bounded diameter. Given a parameter $0 < \gamma \leq 1$, we say that a convex body K is *γ-fat* if there exist concentric Euclidean balls B and B', such that $B \subseteq K \subseteq B'$, and $\text{radius}(B)/\text{radius}(B') \geq \gamma$. We say that K is *fat* if it is γ-fat for a constant γ (possibly depending on d, but not on n or ε). The following lemma shows that K can be fattened without significantly altering the approximation parameter. Let $Q_0^{(d)}$ denote the d-dimensional axis-aligned hypercube of unit diameter centered at the origin. When d is clear, we refer to this as Q_0.

LEMMA 2.1. *Given a convex body K in $\mathbb{R}^d$ and $0 < \varepsilon \leq 1$, there exists an affine transformation T such that $T(K)$ is $(1/d)$-fat and $T(K) \subseteq Q_0$. If P is an absolute $(\varepsilon/d\sqrt{d})$-approximation of $T(K)$, then $T^{-1}(P)$ is a relative ε-approximation of K.*

We omit the proof of this lemma for now, since it is subsumed by Lemma 7.1 below. Our approach will be to map K to $T(K)$, set $\varepsilon' \leftarrow \varepsilon/d\sqrt{d}$, and then apply an absolute ε'-approximation algorithm to $T(K)$ (or more accurately, to the result of applying T to each of K's defining halfspaces). Since ε' is within a constant factor of ε, the asymptotic complexity bounds that we will prove for the absolute case will apply to the original (relative) approximation problem case as well.

2.2. Concepts from quadtrees. By the above reduction, it suffices to consider the problem of computing an absolute ε-approximation to a fat convex body K that lies within Q_0. Our construction will be based on a quadtree decomposition of Q_0. More formally, we define a *quadtree cell* by the following well-known recursive decomposition. Q_0 is a quadtree cell, and given any quadtree cell Q, each of the 2^d hypercubes that result by bisecting each of Q's sides by an axis-orthogonal hyperplane is also a quadtree cell. A cell Q' that results from subdividing Q is a *child* of Q. Clearly, the child's diameter is half that of its parent. The subdivision process defines a (2^d)-ary tree whose nodes are quadtree cells and whose leaves are cells that are not subdivided.

It will be useful to define a notion of approximation that is local to a quadtree cell Q. An obvious definition would be to approximate $K \cap Q$. The problem with this is that a point $p \in Q$ that is close to K need not be close to $K \cap Q$ (see Figure 3(a)). To remedy this we say that a polytope P is an *ε-approximation of K within Q* if

$$K \cap Q \;\subseteq\; P \cap Q \;\subseteq\; (K \oplus \varepsilon) \cap Q$$

(see Figure 3(b)). This definition implies that for any query point $q \in Q$, we can correctly answer ε-approximate polytope membership queries with respect to K by checking whether $q \in P$. We do not care what happens outside of Q, and indeed P may even be unbounded.

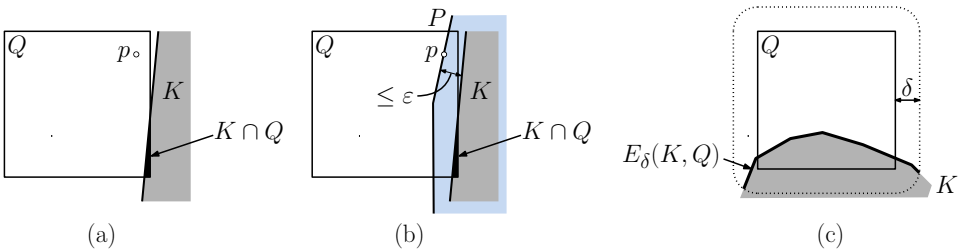


FIG. 3. (a), (b) *ε-approximation of K within Q, and* (c) $E_\delta(K, Q)$.

As we shall see later, computing an ε-approximation of K within a quadtree cell Q will generally require that we consider ∂K in a region that extends slightly beyond Q. We define $E_\delta(K, Q)$ to be the portion of ∂K that lies within distance δ of Q (see Figure 3(c)). Because $\delta = \sqrt{\varepsilon}$ will be of particular interest, we use $E(K, Q)$ as shorthand for $E_{\sqrt{\varepsilon}}(K, Q)$.

In order to apply constructions on quadtree cells of various sizes, it will be convenient to transform all such constructions into a common form. Given a quadtree cell Q, we define *standardization* to be the application of an affine transformation that uniformly scales and translates space so that Q is aligned with the standard quadtree cell Q_0. We transform K using this same transformation and apply the same scale factor to ε. Although we assume that the input body is contained within Q_0, after standardization, the transformed image of K need not be contained within Q_0.

2.3. Polarity and the Mahler volume. Some of our analysis will involve the well-known concept of *polarity*. Let us recall some general facts (see, e.g., Eggleston [35]). Given vectors $u, v \in \mathbb{R}^d$, let $\langle u, v\rangle$ denote their inner product, and let $\|v\| = \sqrt{\langle v, v\rangle}$ denote v's Euclidean length. Given a convex body $K \in \mathbb{R}^d$ define its *polar* to be the convex set

$$\text{polar}(K) \;=\; \{u : \langle u, v\rangle \leq 1 \text{ for all } v \in K\}.$$

If K contains the origin, then polar(K) is bounded. Given $v \in \mathbb{R}^d$, polar(v) is simply the closed halfspace that contains the origin whose bounding hyperplane is orthogonal to v and at distance $1/\|v\|$ from the origin (on the same side of the origin as v). The polar has the inclusion-reversing property that v lies within polar(u) if and only if u lies within polar(v). We may equivalently define polar(K) as the intersection of polar(v) for all $v \in K$.

Generally, given $r > 0$, define

$$\text{polar}_r(K) \;=\; \{u : \langle u, v\rangle \leq r^2 \text{ for all } v \in K\}.$$

It is easy to see that for any $v \in \mathbb{R}^d$, polar$_r(v)$ is the closed halfspace at distance $r^2/\|v\|$ (see Figure 4(a)). Thus, polar$_r(K)$ is a uniform scaling of polar(K) by a factor of r^2. In particular, if B is a Euclidean ball of radius x centered at the origin, then polar$_r(B)$ is a concentric ball of radius r^2/x.

An important concept related to polarity is the *Mahler volume*, which is defined to be the product of the volumes of a convex body and its polar. There is a large literature on the Mahler volume, mostly for centrally symmetric bodies. Later in the paper we will make use of the following bound on the Mahler volume for arbitrary

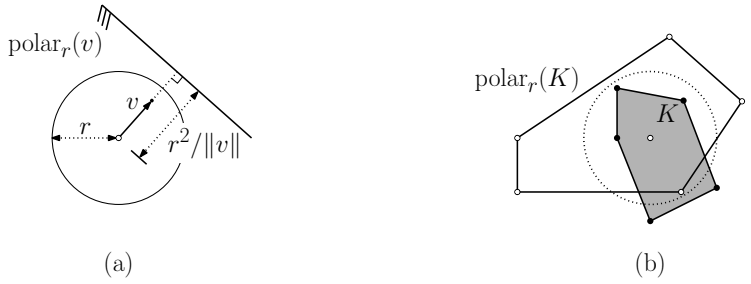


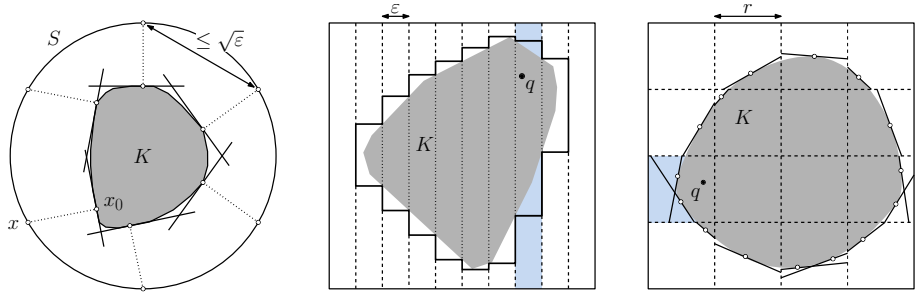
FIG. 4. *The generalized polar transform and polar body.*

convex bodies (see, e.g., Kuperberg [41]). Given a convex body K in $\mathbb{R}^d$, let $\mathrm{vol}(K)$ denote its volume or, more formally, its d-dimensional Hausdorff measure.

LEMMA 2.2 (Mahler volume). *There is a constant c_m depending only on d such that given a convex body K in $\mathbb{R}^d$, $\mathrm{vol}(K)\cdot\mathrm{vol}(\mathrm{polar}(K)) \geq c_m$. More generally, given $r > 0$, $\mathrm{vol}(K)\cdot\mathrm{vol}(\mathrm{polar}_r(K)) \geq c_m r^{2d}$.*

2.4. Simple approximation trade-off. Before presenting our results, it will be illuminating to see how to obtain simple data structures for approximate polytope membership by combining two existing approximation methods. Let us begin by describing Dudley's approximation. Assuming that K is contained within Q_0, let S denote the $(d-1)$-dimensional sphere of radius 3 centered at the origin, which we call the *Dudley hypersphere*. (The value 3 is not critical; any sufficiently large constant suffices.) For $\delta > 0$, a set Σ of points on S is said to be *δ-dense* if every point of S lies within distance δ of some point of Σ. Let Σ be a $\sqrt{\varepsilon}$-dense set of points on S (see Figure 5(a)). By a simple packing argument there exists such a set of cardinality $\Theta(1/\varepsilon^{(d-1)/2})$. For each point $x \in \Sigma$, let x_0 be its nearest point on K's boundary. For each such point x_0, consider the halfspace containing K that is defined by the supporting hyperplane passing through x_0 that is orthogonal to the line segment $\overline{xx}_0$. Dudley shows that the intersection of these halfspaces is an outer ε-approximation of K. We can answer approximate membership queries by testing whether q lies within all these halfspaces (by brute force). This approach takes $O(1/\varepsilon^{(d-1)/2})$ query time and space.

An alternative solution is related to a grid-based approximation by Bentley, Faust, and Preparata [15]. Again, we assume that K is contained within Q_0. For the sake of illustration, let us think of the dth coordinate axis as pointing upward. Partition the upper facet of Q_0 into a $(d-1)$-dimensional square grid with cells of diameter ε. A packing argument implies that the number of cells is $O(1/\varepsilon^{d-1})$. Extend each of these cells downward to form a subdivision of Q_0 into vertical columns (see Figure 5(b)). Trim each column at the highest and lowest points at which it intersects K. Together, these trimmed columns define a collection of hyperrectangles whose union contains K. The resulting data structure has $O(1/\varepsilon^{d-1})$ space. Given a query point q, in $O(1)$ time we can determine the vertical column containing q (assuming a model of computation that supports integer division), and we then test whether q lies within the trimmed column. In contrast to the method based on Dudley's construction, this method provides a better query time of $O(1)$ but with higher space of $O(1/\varepsilon^{d-1})$.



(a) (b) (c)

FIG. 5. *The ε-approximations of* (a) *Dudley and* (b) *Bentley et al., and* (c) *the simple trade-off. (Not drawn to scale.)*

It is possible to combine these two solutions into a simple unified approach that achieves a trade-off between space and query time. Given a parameter r, where $\varepsilon \leq r \leq 1$, subdivide Q_0 into a grid of hypercube cells each of diameter $\Theta(r)$. For each cell Q that intersects K's boundary, apply Dudley's approximation to this portion of the polytope. By a straightforward packing argument, the number of grid cells that intersect K's boundary is $O(1/r^{d-1})$ (see, for example, Lemma 3 of [11]). We apply standardization to Q (thus mapping Q to Q_0 and scaling ε to $\Omega(\varepsilon/r)$) and apply Dudley's construction. By Dudley's results, the number of halfspaces needed to approximate K within Q is $O((r/\varepsilon)^{(d-1)/2})$. To answer a query, in $O(1)$ time we determine which hypercube of the grid contains the query point (assuming a model of computation that supports integer division). We then apply brute-force search to determine whether the query point lies within all the associated halfspaces in time $O((r/\varepsilon)^{(d-1)/2})$. The query time is dominated by this latter term. The space is dominated by the total number of halfspaces, which is $O((1/r^{d-1}) \cdot (r/\varepsilon)^{(d-1)/2}) = O(1/(\varepsilon r)^{(d-1)/2})$. If we express r in terms of a parameter α, where $r = \varepsilon^{1-2/\alpha}$, then Theorem 1.1 follows as an immediate consequence. Note that the resulting trade-off interpolates nicely between the two extremes for $\varepsilon \leq r \leq 1$.

3. The data structure and construction. In this section we show how to improve the approach from the previous section by replacing the grid with a quadtree. The data structure is constructed by the recursive procedure, called SplitReduce, whose inputs consist of a convex body K and a quadtree cell Q. We are also given the approximation parameter $0 < \varepsilon \leq 1$ and a parameter $t \geq 1$ that controls the query time. Although we assume that K is presented as the intersection of n halfspaces, this procedure can be applied to any representation that supports the following access primitives:

(i) Determine whether Q is disjoint from K.
(ii) Determine whether Q is contained within $K \oplus \varepsilon$.
(iii) Determine whether there exists a set of at most t halfspaces whose intersection ε-approximates K within Q, and if so generate such a set.

Recall that we assume that K has been transformed so it is $(1/d)$-fat and lies within Q_0 (the hypercube of unit diameter centered at the origin). The data structure is built by the call SplitReduce(K, Q_0). In general, SplitReduce(K, Q) checks whether any of the above access primitives returns a positive result, and if so it terminates the decomposition and Q is declared a *leaf cell*. Otherwise, it makes a recursive call on the children of Q (see Figure 6(a)). On termination, each leaf cell is labeled as either "inside" or "outside" or is associated with a set of at most t approximating halfspaces (see Figure 6(b)).

SplitReduce(K, Q).
1. If $Q \cap K = \emptyset$, label Q as "outside."
2. If $Q \subseteq K \oplus \varepsilon$, label Q as "inside."
3. If there exists a set at most t halfspaces whose intersection provides an ε-approximation to K within Q, associate Q with such a set $P(Q)$ of minimum size.
4. Otherwise, split Q into 2^d quadtree cells and recursively invoke SplitReduce on each.

For the sake of our space-time trade-offs, we will usually assume that t is reasonably large, say, $t = \Omega(\log \frac{1}{\varepsilon})$. Under our assumption that $t \geq 1$, steps 1 and 2 are not needed, since it is possible to ε-approximate any cell satisfying these conditions with a single halfspace. This assumption on the value of t is mainly a convenience to simplify

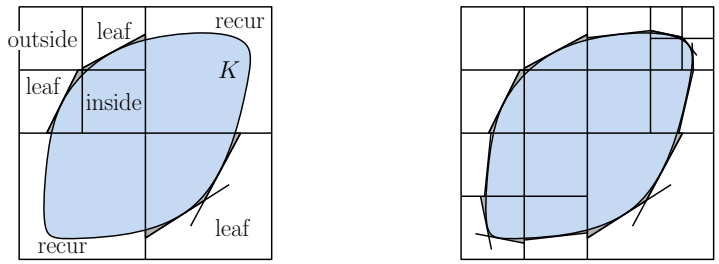


FIG. 6. (a) *Cases arising in SplitReduce for* $t = 2$ *and* (b) *the final subdivision.*

the formulas of our mathematical analysis. Observe that even if $t = 0$, the procedure will terminate and provide a correct answer once the cell diameter falls below ε.

It is easy to see that the recursion must terminate as soon as $\text{diam}(Q) \leq \varepsilon$ (since, irrespective of whether it intersects ∂K, any such cell can be labeled either as "inside" or "outside"). Of course, it may terminate much sooner. Since Q_0 is of unit diameter, it follows that the height of the quadtree is $O(\log \frac{1}{\varepsilon})$. The total space used by the data structure is the sum of the space needed to store the quadtree and the space needed to store the approximating halfspaces for the cells that intersect K's boundary. Our next lemma shows that if the query time is sufficiently large, the latter quantity dominates the space asymptotically. For each leaf cell Q generated by step 3, define $t(Q) = |P(Q)|$, and define $t(Q) = 1$ for all the other leaf cells.

LEMMA 3.1. *Given a convex body* $K \subseteq Q_0$ *in* $\mathbb{R}^d$*. If* t *is* $\Omega(\log \frac{1}{\varepsilon})$*, then the total space of the data structure produced by SplitReduce*(K, Q_0) *on* K *for query time* t *is asymptotically dominated by the sum of* $t(Q)$ *over all the leaf cells* Q *that intersect* K*'s boundary.*

Proof. Let T denote the quadtree produced by running SplitReduce on K. As mentioned above, T is of height $O(\log \frac{1}{\varepsilon})$. By our hypothesis that t is $\Omega(\log \frac{1}{\varepsilon})$, there exists a constant c such that $\text{height}(T) \leq ct$. Let L denote the set of leaves of T that intersect the boundary of K, and let M denote the internal nodes of T that have the property that all their children are leaves. (These are the lowest internal nodes of the tree.) Let $t(L)$ denote the sum of $t(Q)$ over all $Q \in L$.

The fact that each node $u \in M$ was subdivided by SplitReduce implies that more than t halfspaces are needed to approximate K within u's cell. Therefore, the children of u that intersect K's boundary together require at least t halfspaces. In addition to $P(Q)$, each quadtree leaf Q can (implicitly) contribute its $2d$ bounding hyperplanes to the approximation. Therefore, $t(L) + 2d|L|$ hyperplanes suffice to approximate K in all the cells of M, implying that $t(L) + 2d|L| \geq t \cdot |M|$. Since $t(Q) \geq 1$, we have $t(L) \geq |L|$, and thus $(1 + 2d)t(L) \geq t \cdot |M|$.

Each internal node of T either is in M or is an ancestor of a node in M. Thus, the total number of internal nodes of T is at most $|M| \cdot \text{height}(T)$. Since each internal node of a quadtree has 2^d children, the total number of nodes in the tree, excluding the root, is at most

$$2^d \cdot |M| \cdot \text{height}(T) \;\leq\; 2^d \cdot |M| \cdot (ct) \;\leq\; 2^d c(1 + 2d)t(L) \;=\; O(t(L)).$$

Each internal node of T and each leaf node that does not intersect K's boundary contributes only a constant amount to the total space. Therefore, the space contribution

of the nodes other than those of L is at most a constant factor larger than the total number of nodes of T, which we have shown is $O(t(L))$. Therefore, the total space is $O(t(L))$, as desired. ∎

A query is answered by performing a point location in the quadtree to determine the leaf cell containing the query point. If the leaf cell is not labeled as being "inside" or "outside," we test whether the query point lies within all the associated halfspaces, and if so, we declare the point to be inside K. Otherwise it is declared to be outside. Clearly, the query time is $O(\log \frac{1}{\varepsilon} + t)$.

The algorithm is correct provided that the set of halfspaces $P(Q)$ computed in step 3 defines any ε-approximation of K within Q, but our analysis of the data structure's space requirements below (see the proof of Lemma 4.3) relies on the assumption that the size of $P(Q)$ is within a constant factor of the minimum number of halfspaces of any ε-approximating polytope within Q. Unfortunately, we know of no constant-factor approximation to the problem of computing such a polytope. Thus, strictly speaking, the bounds stated in Theorem 1.2 are purely existential. In section 7 we will show that through a straightforward modification of the greedy set-cover heuristic, it is possible to compute an approximation in which the number of defining halfspaces exceeds the optimum (for slightly smaller approximation parameter) by a factor of at most $\rho = O(\log \frac{1}{\varepsilon})$. From the following result it follows that this increases our space and query time bounds by $O(\log \frac{1}{\varepsilon})$.

LEMMA 3.2. *Given any $\rho \geq 1$ and any constant $0 < \beta \leq 1$, if the number of halfspaces of $P(Q)$ computed in step 3 of SplitReduce is within a factor ρ of the minimum number of facets of any $(\beta\varepsilon)$-approximating polytope within Q, then Theorems 1.2 and 1.4 hold but with the asymptotic space and query time bounds larger by a factor of ρ.*

Proof. Let us refer to the hypothesized version of SplitReduce whose step 3 is suboptimal as SplitReduce$'$. Consider an execution of SplitReduce$'$ using ρt as the desired query time and ε as the approximation parameter, and let us compare this to an execution of SplitReduce using t and $\beta\varepsilon$, respectively. Since β is a constant, the asymptotic dependencies on ε are unaffected, and therefore the space and query times stated in Theorems 1.2 and 1.4 apply without modification to the execution of SplitReduce. In this execution, if the subdivision declares some quadtree cell Q to be a leaf, then t halfspaces suffice to $(\beta\varepsilon)$-approximate K within Q, and so by our hypothesis in the corresponding execution of SplitReduce$'$, step 3 returns at most ρt halfspaces, implying this execution also declares Q to be a leaf. Therefore, the tree generated by SplitReduce$'$ is a subtree of the tree generated by SplitReduce, but each leaf node may contain up to a factor of ρ more halfspaces. Thus, the asymptotic space and query time bounds for SplitReduce$'$ are larger than those of SplitReduce by this same factor. ∎

4. Simple upper bound. In this section, we present a simple upper bound of $O(1/\varepsilon^{(d-1)/2})$ on the storage of the data structure obtained by the SplitReduce algorithm for any given query time $t \geq 1/\varepsilon^{(d-1)/4}$. The tools developed in this section will be useful for the more comprehensive upper bounds, which will be presented in subsequent sections.

Throughout this section we do not necessarily assume that K has been scaled to lie within Q_0 and may generally be much larger. Recall that S denotes a hypersphere of radius 3 centered at the origin. Let X denote a surface patch of K that lies within S. Let $\mathrm{Vor}(X)$ denote the set of points exterior to K whose closest point on ∂K lies

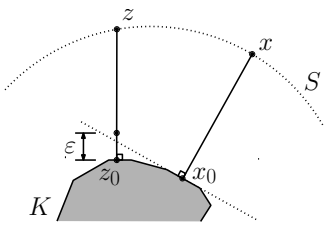


FIG. 7. *Lemma* 4.2.

within X. We refer to the surface patch $\mathrm{Vor}(X) \cap S$ (the points of S whose closest point on ∂K lies within X) as the *Voronoi patch* of X. Voronoi patches are related to Dudley's construction. In particular, a sample point $x \in S$ from Dudley's construction generates a supporting halfspace at a point of X if and only if $x \in \mathrm{Vor}(X) \cap S$. The following two lemmas are straightforward adaptations of Dudley's analysis [32]. The first is just a restatement of Dudley's result.

LEMMA 4.1. *Given a convex body K in $\mathbb{R}^d$ that lies within Q_0 and $0 < \varepsilon \leq 1$, there exists an ε-approximating polytope P bounded by at most $c/\varepsilon^{(d-1)/2}$ facets, where c is a constant depending only on d.*

The second lemma is a technical result that is implicit in Dudley's analysis. Given two points $x, y \in \mathbb{R}^d$, let $\overline{xy}$ denote the segment between them, and let $\|xy\|$ denote the Euclidean length of this segment.

LEMMA 4.2. *Let K be a convex body, let $0 < \varepsilon \leq 1$, and let z and x be two points of S such that $\|zx\| \leq \sqrt{\varepsilon}/4$. Let z_0 and x_0 be the points on ∂K that are closest to z and x, respectively. If z_0 is within unit distance of the origin, then*

(i) $\|z_0 x_0\| \leq \sqrt{\varepsilon}/4$ *and*

(ii) *the supporting hyperplane at x_0 orthogonal to the segment $\overline{xx}_0$ intersects segment $\overline{zz}_0$ at distance less than ε from z_0 (see Figure* 7*).*

The following lemma is an extension of Dudley's results, which allows us to bound the complexity of an ε-approximation of K within a quadtree cell Q. Recall from section 2.2 that $E(K, Q)$ denotes the portion of ∂K that lies within distance $\sqrt{\varepsilon}$ of Q.

LEMMA 4.3. *Let K be a convex body, $Q \subseteq Q_0$ be a quadtree cell that intersects ∂K, and $0 < \varepsilon \leq 1/2$. Let Σ denote a set of $(\sqrt{\varepsilon}/4)$-dense points on the Dudley sphere S. Then $t(Q) \leq |\Sigma \cap \mathrm{Vor}(E(K, Q))|$ (see Figure* 8(a)*).*

Proof. We construct an approximating polytope P by the following local variant of Dudley's construction. For each point $x \in \Sigma \cap \mathrm{Vor}(E(K, Q))$, let x_0 be its nearest point on the boundary of K. (Note that $x_0 \in E(K, Q)$.) For each point x_0, take the supporting halfspace to K passing through x_0 that is orthogonal to the segment $\overline{xx}_0$. Let P be the (possibly unbounded) intersection of these halfspaces.

First, we show that $\Sigma \cap \mathrm{Vor}(E(K, Q))$ is nonempty. Consider any point z_0 on $\partial K \cap Q$. Let z denote any point of $S \cap \mathrm{Vor}(E(K, Q))$ whose closest point on ∂K is z_0. By definition of Σ, there is a point $x \in \Sigma$ whose distance from z is at most $\sqrt{\varepsilon}/4$. Letting x_0 denote x's closest point on ∂K, by Lemma 4.2(i), $\|z_0 x_0\| \leq \sqrt{\varepsilon}/4 < \sqrt{\varepsilon}$. Thus, x_0 lies within $E(K, Q)$, which implies that $x \in \Sigma \cap \mathrm{Vor}(E(K, Q))$. It follows that P is bounded by at least one halfspace.

We now show that P is an (outer) ε-approximation of K within Q. Since P is defined by supporting hyperplanes, K is contained within P. Consider any $q \in Q$ that

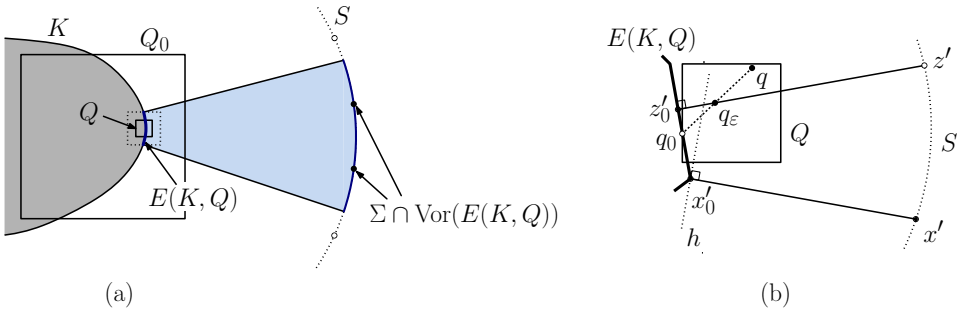


FIG. 8. *Lemma* 4.3. *(Not drawn to scale.)*

is at distance greater than ε from K. It suffices to show that $q \notin P$, that is, there exists a bounding hyperplane for P that separates q from K. Let q_0 denote the point of $K \cap Q$ that is closest to q (see Figure 8(b)). Note that q_0 is constrained to lie within Q, and hence this may not be the closest point to q on ∂K. By continuity, there must be a point on the segment $\overline{qq}_0$ that is at distance exactly ε from ∂K, which we denote by q_ε. Since Q is convex, this segment is contained in Q, and, hence, so is q_ε.

Let z'_0 be the point on ∂K that is closest to q_ε. (Note that z'_0 need not lie within Q.) Because Q_0 is centered at the origin, z'_0's distance from the origin is at most $\mathrm{diam}(Q_0)/2 + \|q_\varepsilon z'_0\| \leq 1/2 + \varepsilon \leq 1$. Let z' denote the point of intersection with the Dudley hypersphere S of the ray emanating from z'_0 and passing through q_ε. Let x' be a point of Σ that lies within distance $\sqrt{\varepsilon}/4$ of z', and let x'_0 be its closest point on ∂K. By Lemma 4.2(i) $\|x'_0 z'_0\| \leq \sqrt{\varepsilon}/4$, and by (ii) the supporting hyperplane h at x'_0 orthogonal to the segment $x'x'_0$ intersects segment $\overline{z'z'_0}$ at distance less than ε from z'_0. Thus, h separates q_ε from K, and therefore it separates q from K.

To complete the proof that $q \notin P$, it suffices to show that h is indeed included in our construction of P. By the triangle inequality and our assumption that $\varepsilon \leq 1/2$, the distance from x'_0 to Q is at most

$$\|x'_0 z'_0\| + \|z'_0 q_\varepsilon\| \;\leq\; \frac{\sqrt{\varepsilon}}{4} + \varepsilon \;\leq\; \sqrt{\varepsilon}.$$

It follows that $x'_0 \in E(K,Q)$, and so h is included in the construction of P. By our hypothesis that the set $P(Q)$ constructed in step 3 of SplitReduce is the minimum-sized set of halfspaces needed to ε-approximate K within Q, we have $t(Q) = |P(Q)| \leq |P| = |\Sigma \cap \mathrm{Vor}(E(K,Q))|$. (Note that this works even if Q is an "inside" cell that intersects K's boundary. In such a case $t(Q) = 1$ by definition, and as argued above, $\Sigma \cap \mathrm{Vor}(E(K,Q))$ is nonempty.) This completes the proof. ∎

Next, we prove a useful technical lemma, which bounds the total complexity of a set of leaves whose cells are of a given minimum size. Recalling the definition of Σ from the previous lemma, we may assume that $|\Sigma| = \Theta(1/\varepsilon^{(d-1)/2})$.

LEMMA 4.4. *Let K be a convex body in $\mathbb{R}^d$, let $0 < \varepsilon \leq 1/2$, and let L denote a set of disjoint quadtree cells contained within Q_0 such that each intersects ∂K and is of diameter $\Omega(\sqrt{\varepsilon})$. Then $\sum_{Q \in L} t(Q) = O(1/\varepsilon^{(d-1)/2})$.*

Proof. By applying Lemma 4.3 to each $Q \in L$ we have

$$\sum_{Q \in L} t(Q) \;\leq\; \sum_{Q \in L} |\Sigma \cap \mathrm{Vor}(E(K,Q))|.$$

Since $|\Sigma| = O(1/\varepsilon^{(d-1)/2})$, to complete the proof it suffices to show that each $x \in \Sigma$ lies within $\mathrm{Vor}(E(K,Q))$ for at most constant number of $Q \in L$. To see this, let x_0 be the point on ∂K that is closest to x. Since each cell $Q \in L$ has size at least $\Omega(\sqrt{\varepsilon})$, by disjointness and a packing argument it follows that at most a constant number (depending on dimension) of such cells can lie within distance $\sqrt{\varepsilon}$ of x_0, which establishes the claim. ∎

Combining the above results, we obtain the main result of this section.

LEMMA 4.5. *Let K be a convex body in $\mathbb{R}^d$ and $0 < \varepsilon \leq 1/2$. The output of SplitReduce(K, Q_0) for $t \geq 1/\varepsilon^{(d-1)/4}$ has total space $O(1/\varepsilon^{(d-1)/2})$.*

Proof. Let c_2 be the constant of Lemma 4.1, and define $c_1 = (1/c_2)^{2/(d-1)}$. We may assume that $\varepsilon \leq c_1^2$, for otherwise $\varepsilon = \Omega(1)$ and clearly SplitReduce will not generate more than a constant number of cells.

Let T denote the quadtree produced by the algorithm, and let L denote the set of leaf cells of T that intersect the boundary of K. Recall from Lemma 3.1 that the data structure's total space is asymptotically bounded by the sum of $t(Q)$ for all $Q \in L$. Thus, it suffices to prove that

$$\sum_{Q \in L} t(Q) \;=\; O(1/\varepsilon^{(d-1)/2}).$$

Toward this end, we first prove a lower bound on the size of any leaf cell Q. We assert that the cell Q associated with any internal node has diameter at least $\delta = c_1\sqrt{\varepsilon}$. It will then follow that each leaf cell has diameter at least $\delta/2$. Suppose to the contrary that $\mathrm{diam}(Q) < \delta$. Recall the standardization transformation from section 2.2, which maps Q to Q_0 and scales ε to at least $\varepsilon/\delta = \sqrt{\varepsilon}/c_1$. Let us denote this value by ε'. Since $\varepsilon \leq c_1^2$, we have $\varepsilon' \leq 1$. By applying Lemma 4.1 to the transformed body (with ε' playing the role of ε), it follows that the polytope $K \cap Q$ can be ε-approximated by a polytope P defined by the intersection of at most

$$\frac{c_2}{(\varepsilon')^{(d-1)/2}} \;=\; c_2\left(\frac{c_1}{\sqrt{\varepsilon}}\right)^{(d-1)/2} \;\leq\; \frac{1}{\varepsilon^{(d-1)/4}}$$

halfspaces. Since $K \cap Q \subseteq P$, it is easy to see that P is an ε-approximation of K within Q. Since $t \geq 1/\varepsilon^{(d-1)/4}$, the termination condition of our algorithm implies that such a cell is not further subdivided, contradicting our hypothesis that this is an internal node. Therefore, the cells of L satisfy the conditions of Lemma 4.4. The desired bound follows by applying this lemma. ∎

It is useful to contrast this with the Dudley-based approach described in section 2.4. For $t = 1/\varepsilon^{(d-1)/4}$, we obtain the same $O(1/\varepsilon^{(d-1)/2})$ space in each case, but the exponent in the query time of SplitReduce is only half that of the Dudley-based approach. Later, in Lemma 6.3, we will present a more refined analysis showing that it is possible to reduce this further, achieving a query time of only $\widetilde{O}(1/\varepsilon^{(d-1)/8})$.

It will be useful in later sections to generalize the above lemma to quadtree cells of arbitrary size. By a direct application of standardization, we obtain the following.

LEMMA 4.6. *Let K be a convex body in $\mathbb{R}^d$, let Q be a quadtree cell contained within Q_0, and let $0 < \varepsilon \leq \mathrm{diam}(Q)/2$. The output of the call SplitReduce(K, Q) for $t \geq (\mathrm{diam}(Q)/\varepsilon)^{(d-1)/4}$ has total space $O((\mathrm{diam}(Q)/\varepsilon)^{(d-1)/2})$.*

5. Dual caps and approximation. The bounds proved in the previous section apply to query times $t \geq 1/\varepsilon^{(d-1)/4}$. In section 6 we will show how to obtain good

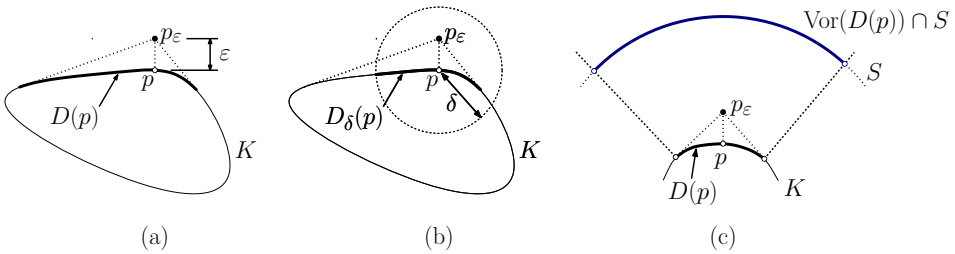


FIG. 9. (a) *Dual caps,* (b) *restricted dual caps, and* (c) *the Voronoi patch of a dual cap.*

space bounds for smaller query times. This will involve analyzing the local geometry about small boundary patches of the convex body. In this section, we introduce the principal geometric underpinnings that will be needed for this more refined analysis. In particular, we discuss the concepts of dual caps and restricted dual caps and their role in polytope approximation.

Although we do not assume that K is smooth, it will simplify the presentation to imagine that each boundary point has a unique supporting hyperplane and a unique normal vector. To achieve this, we employ an *augmented representation* of the boundary points of K. In particular, each boundary point $p \in \partial K$ will be expressed as a pair (p, h), where h is a supporting hyperplane at p. We will often refer to h as $h(p)$. When h is clear from context or unimportant, we avoid explicit reference to it.

We first observe that computing an outer ε-approximation of a convex body K by halfspaces can be reduced to a hitting-set problem. Consider any point p_ε that is external to K at distance ε from its boundary, and let (p, h) denote the augmented boundary point consisting of the closest point $p \in \partial K$ to p_ε and the supporting hyperplane through p that is orthogonal to the segment pp_ε (see Figure 9(a)). We define the *ε-dual cap* of p, denoted $D(p)$, to be the set of augmented boundary points (q, h') such that the supporting hyperplane h' through q intersects the closed line segment pp_ε. (Equivalently, these are the points of ∂K that are visible to p_ε.)

Any outer ε-approximation of K by halfspaces must contain at least one halfspace that separates p from p_ε, and this can be achieved by including h' for any pair (q, h') within $D(p)$. A set of augmented points $\Sigma \subseteq \partial K$ is said to be an *ε-hitting set* for K if for every $p \in \partial K$, $\Sigma \cap D(p) \neq \emptyset$. It follows directly that the intersection of the supporting halfspaces for any ε-hitting set is an outer ε-approximation of K. This observation will be formalized within our quadtree-based context in our next lemma. Before stating the lemma, we need to introduce one additional concept. In order to approximate K within a given quadtree cell Q, we are interested only in the geometry of K's boundary that lies close to Q. For this reason, it will be desirable to limit the diameter of dual caps. Given $\delta > 0$, let $B_\delta(p)$ denote the closed Euclidean ball of radius δ centered at p. Define the *δ-restricted dual cap*, denoted $D_\delta(p)$, to be the intersection of $D(p)$ with $B_\delta(p)$ (see Figure 9(b)).

LEMMA 5.1. *Let K be a convex body, let $Q \subseteq Q_0$ be a quadtree cell that intersects ∂K, and $0 < \varepsilon \leq 1/2$. Let Σ be any set of augmented points on $E(K, Q)$ that hits the set of all $\sqrt{\varepsilon}$-restricted ε-dual caps whose defining point is in $E(K, Q)$ (see Figure* 10(a)*). Then there is a polytope P defined as the intersection of $|\Sigma|$ halfspaces that ε-approximates K within Q.*

Proof. Let P be the polytope defined by the intersection of the supporting halfspaces associated with each augmented point of Σ (see Figure 10(b)). Clearly,

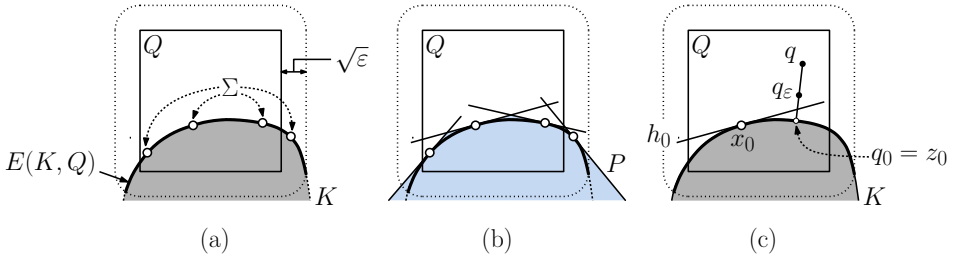


FIG. 10. *Lemma* 5.1.

$K \subseteq P$. Consider any point $q \in Q$ that is at distance greater than ε from $K \cap Q$. It suffices to show that $q \notin P$, that is, there exists a bounding hyperplane for P that separates q from K.

We apply a similar argument to the one that we used in the proof of Lemma 4.3. Consider any $q \in Q$ that is at distance greater than ε from K (see Figure 10(c)). It suffices to show that there exists a bounding hyperplane for P that separates q from K. Let q_0 denote the point of $K \cap Q$ that is closest to q. By continuity, there must be a point on the segment $\overline{qq}_0$ that is at distance exactly ε from ∂K, which we denote by q_ε. Since Q is convex, this segment must be contained in Q, and, hence, so is q_ε.

Let z_0 be the point on ∂K that is closest to q_ε. (In our figure $z_0 = q_0$, but generally z_0 need not lie within Q.) Since $\varepsilon \leq 1$, we have $\|q_\varepsilon z_0\| = \varepsilon \leq \sqrt{\varepsilon}$. It follows that $z_0 \in E(K, Q)$. Therefore, there exists an augmented point $(x_0, h_0) \in \Sigma$ that hits the $\sqrt{\varepsilon}$-restricted ε-dual cap defined by z_0 (whose apex is at q_ε). The supporting hyperplane h_0 separates q_ε (and therefore q) from K, as desired. ∎

Our analysis of the space bounds of SplitReduce is based on the combined sizes of the ε-hitting sets for K within each quadtree cell Q. Dudley's construction can be viewed as one method of computing ε-hitting sets. Unfortunately, Dudley's construction does not lead to the best bounds because it tends to oversample in regions of very low or very high curvature. Our analysis will be based on a more refined, area-based approach to bounding the sizes of hitting sets. The key geometric observation is that the product of the areas of any ε-dual cap and its associated Voronoi patch on the Dudley sphere S must be large. Intuitively, if the surface area of an ε-dual cap is small, then the total curvature of the patch must be high, and so the associated Voronoi patch must have relatively large area (see Figure 9(c)). More precisely, we show that (under certain conditions) the product of the areas of an ε-dual cap and its Voronoi patch is $\Omega(\varepsilon^{d-1})$. This result is stated formally in Lemma 5.2 below. Given a $(d-1)$-dimensional manifold, let area(Y) denote its $(d-1)$-dimensional Hausdorff measure. Given a convex body X in $\mathbb{R}^d$, we use area(X) as a shorthand for area(∂X).

LEMMA 5.2 (area-product bound). *Let K be a convex body in $\mathbb{R}^d$, and let $0 < \varepsilon \leq 1/8$. Consider a pair $(p, h(p))$, where $p \in \partial K$ and $h(p)$ is a supporting hyperplane passing through p. Let D denote the $\sqrt{\varepsilon}$-restricted ε-dual cap whose defining point is p. If K is fat and of diameter at least 2ε, there exists a constant c_a (depending only on d) such that if p lies within a unit ball centered at the origin, then* area$(D) \cdot$ area$(\mathrm{Vor}(D) \cap S) > c_a \cdot \varepsilon^{d-1}$.

The proof of the lemma is quite technical and will be deferred to section 10. The geometric basis of the proof involves the Mahler volume, which was introduced

in section 2.3. The bound stated in the lemma holds if K is γ-fat for any γ in the interval $(0, 1]$ under the assumption that γ does not depend on ε. In particular, the proof will reveal that $c_a = \Omega(\gamma^{d-1})$.

We will exploit this observation to demonstrate the existence of smaller ε-hitting sets than those given by Dudley's construction. We will hit the restricted ε-dual caps that have large surface area by sampling points randomly on the boundary of K, and we will hit those with small surface area by sampling points randomly on the Dudley hypersphere and then selecting their nearest neighbors on ∂K. In order to prove that such a random sampling strategy works to stab all the dual caps, we need to establish bounds on the VC-dimension of an appropriate range space based on dual caps. This is not surprising given that dual caps and restricted dual caps are defined by a constant number of parameters. The result is stated in the following lemma. The proof involves a straightforward application of basic geometric principles. (Details can be found in [8].)

LEMMA 5.3. *Let K be a convex body in $\mathbb{R}^d$ that lies within Q_0, and let ε and δ be positive real parameters. The following range spaces (X_i, R_i) have constant VC-dimension (where the constant depends only on d):*

1. *$X_1 = \partial K$ and R_1 is the set of ε-dual caps.*
2. *$X_2 = S$ and R_2 is the set of Voronoi patches of the ε-dual caps.*
3. *$X_3 = \partial K$ and R_3 is the set of δ-restricted ε-dual caps.*
4. *$X_4 = S$ and R_4 is the set of Voronoi patches of the δ-restricted ε-dual caps.*

In the next section we will exploit this result to establish the existence of small ε-nets for these range spaces. Note that the range spaces defined in this lemma are defined over ∂K, a domain of infinite cardinality. However, for our purposes, it suffices to consider dual caps and restricted dual caps whose defining points are drawn from any sufficiently dense set of points on ∂K (depending on ε), and therefore the domains of the range spaces can be treated as finite sets.

6. Final upper bound. In this section, we use the tools developed in sections 4 and 5 to obtain better upper bounds for approximate polytope membership. In particular, we present a proof of Theorem 1.2. We will first show how to apply the area-based techniques described in the previous section to improve the simple upper bound from Lemma 4.5 at the low-space end of the trade-off spectrum. (This will be presented in Lemma 6.3.) We will then apply this improvement repeatedly in an inductive manner to establish trade-offs throughout the spectrum. For technical reasons, many of the lemmas of this section assume constant upper bounds on the value of ε. There is no loss of generality in doing so, since it is easy to show that if ε is bounded below by any fixed constant, the asymptotic space and query times of SplitReduce are both $O(1)$.

Throughout this section, recall that $E_\delta(K, Q)$ is the portion of ∂K that is within distance δ of Q, and $E(K, Q) = E_{\sqrt{\varepsilon}}(K, Q)$. Also, define $E^+(K, Q) = E_{2\sqrt{\varepsilon}}(K, Q)$. We will assume that $\mathrm{diam}(K) \geq 2\varepsilon$, for otherwise it is trivial to compute an ε-approximation of constant size. Our first result establishes an area-based bound on the number of halfspaces needed to approximate K within a quadtree cell Q.

LEMMA 6.1. *Let K be a fat convex body in $\mathbb{R}^d$, let $0 < \varepsilon \leq 1/8$, and let $Q \subseteq Q_0$ be a quadtree cell that intersects ∂K. Letting c_a denote the constant of Lemma 5.2, define*

$$r \;=\; \left(\frac{\mathrm{area}(E^+(K, Q)) \cdot \mathrm{area}\left(\mathrm{Vor}(E^+(K, Q)) \cap S\right)}{c_a \cdot \varepsilon^{d-1}} \right)^{1/2}.$$

There is a polytope P defined as the intersection of $O(r \log r)$ halfspaces that is an ε-approximation of K within Q.

Proof. Letting $A_K = \text{area}(E^+(K,Q))$ and $A_S = \text{area}\,(\text{Vor}(E^+(K,Q)) \cap S)$, we can express the value of r more succinctly as $(A_K A_S / c_a \varepsilon^{d-1})^{1/2}$. First, we assert that $r = \Omega(1)$. To see this, we consider two cases. First, if K lies entirely within distance $2\sqrt{\varepsilon}$ of Q, then $\text{Vor}(E^+(K,Q)) \cap S = S$, which implies that $A_S = \Omega(1)$. Since K is fat and by our assumption that $\text{diam}(K) \geq 2\varepsilon$, it follows that $A_K = \Omega(\varepsilon^{d-1})$. Therefore, $r = \Omega(1)$. On the other hand, if some part of K lies at distance greater than $2\sqrt{\varepsilon}$ from Q, $E^+(K,Q)$ is a boundary patch of K of diameter $\Omega(\sqrt{\varepsilon})$. Since both K and Q are fat, it follows that $A_K = \Omega(\varepsilon^{(d-1)/2})$. By convexity, as we go from a boundary patch on K to its Voronoi cell on S, distances cannot decrease. Therefore $A_S \geq A_K$, and again we have $r = \Omega(1)$. Through a minor adjustment to constant factor c_a in r's definition, we may assume that $\log r \geq 1$.

By Lemma 5.1, in order to show the existence of an ε-approximating polytope P for K within Q, it suffices to show that it is possible to hit all $\sqrt{\varepsilon}$-restricted ε-dual caps whose defining point lies in $E(K,Q)$ (not to be confused with $E^+(K,Q)$) using $O(r \log r)$ points. To do this, we distinguish between two types of such restricted dual caps. A restricted dual cap D is of *type* 1 if $\text{area}(D) \geq (c_a \varepsilon^{d-1} A_K / A_S)^{1/2}$, and otherwise it is of *type* 2.

By assertions 3 and 4 of Lemma 5.3, we know that $\sqrt{\varepsilon}$-restricted ε-dual caps and their Voronoi patches both have constant VC-dimension. The VC-dimension is no larger if we restrict the domain of the range space. Therefore, by standard machinery (see, e.g., [4]) we can build a $(1/r)$-net for any restriction of these range spaces of size $O(r \log r)$ each by random sampling.

For type-1 dual caps, consider the restriction $(E^+(K,Q), R_3)$ of the range space given in Lemma 5.3.3. Let Σ_1 denote a $(1/r)$-net. Consider any type-1 dual cap D. Since D's defining point lies within $E(K,Q)$ and it is $\sqrt{\varepsilon}$-restricted, it lies entirely within $E^+(K,Q)$. Thus, we have

$$\frac{\text{area}(D \cap E^+(K,Q))}{\text{area}(E^+(K,Q))} = \frac{\text{area}(D)}{\text{area}(E^+(K,Q))} \geq \frac{\left(c_a \cdot \varepsilon^{d-1} A_K / A_S\right)^{1/2}}{A_K}$$
$$= \left(\frac{c_a \cdot \varepsilon^{d-1}}{A_K A_S}\right)^{1/2} = \frac{1}{r}.$$

Therefore D contains at least one point of Σ_1. It follows that Σ_1 hits all type-1 dual caps.

For type-2 dual caps, let us consider the restriction $(\text{Vor}(E^+(K,Q)) \cap S, R_4)$ of the range space of Lemma 5.3.4. Let Σ_2 denote a $(1/r)$-net. Because $\varepsilon \leq 1/8$ and $Q \subseteq Q_0$, $E(K,Q)$ lies within a ball centered at the origin of radius $\text{diam}(Q_0)/2 + \sqrt{\varepsilon} \leq 1$. Given any type-2 dual cap D whose defining (augmented) point lies in $E(K,Q)$, we may apply Lemma 5.2 to obtain

$$\text{area}(\text{Vor}(D) \cap S) \geq \frac{c_a \cdot \varepsilon^{d-1}}{\text{area}(D)} \geq \frac{c_a \cdot \varepsilon^{d-1}}{\left(c_a \cdot \varepsilon^{d-1} A_K / A_S\right)^{1/2}} = \left(\frac{c_a \cdot \varepsilon^{d-1} A_S}{A_K}\right)^{1/2}.$$

As before, since D's defining point lies within $E(K,Q)$, $D \subseteq E^+(K,Q)$. From this we have

$$\frac{\text{area}(\text{Vor}(D \cap E^+(K,Q)) \cap S)}{\text{area}(\text{Vor}(E^+(K,Q)) \cap S)} = \frac{\text{area}(\text{Vor}(D) \cap S)}{\text{area}(\text{Vor}(E^+(K,Q)) \cap S)}$$
$$\geq \frac{\left(c_a \cdot \varepsilon^{d-1} A_S / A_K\right)^{1/2}}{A_S} = \left(\frac{c_a \cdot \varepsilon^{d-1}}{A_K A_S}\right)^{1/2}$$
$$= \frac{1}{r}.$$

Therefore $\text{Vor}(D) \cap S$ contains at least one point of Σ_2, implying that Σ_2 hits the Voronoi patches of all type-2 dual caps. For each point of Σ_2, we select its nearest neighbor on ∂K, obtaining a set $\Sigma_2' \subset E^+(K,Q)$. It follows directly that the set Σ_2' hits all type-2 dual caps. Therefore, the union $\Sigma_1 \cup \Sigma_2'$ forms the desired set of size $O(r \log r)$ that hits all $\sqrt{\varepsilon}$-restricted ε-dual caps whose defining point lies within $E(K,Q)$. ∎

In order to establish our storage bounds, we analyze the behavior of the algorithm at a particular level of the decomposition. Given the query-time parameter t, recall that we stop the subdivision process in SplitReduce(K,Q) if the number of hyperplanes needed to approximate K within Q falls below t. Also recall that $t(Q)$ denotes the number of approximating halfspaces associated with Q. Let us consider the state of the subdivision process when the cell sizes reach roughly $\sqrt{\varepsilon}$. Cells that have stopped subdividing by this point are "good," since we can bound the total space requirements for all such cells by appealing to Lemma 4.4. For the remaining "bad" cells, we will bound their space requirements on a cell-by-cell basis by using the simple upper bound from Lemma 4.6. For our approach to work well, it is crucial to obtain a good bound on the number of such bad cells. We exploit the area bound of Lemma 6.1 for this purpose. Whenever SplitReduce subdivides a cell of size $O(\sqrt{\varepsilon})$, we can infer that more than t hyperplanes are required to approximate K within this cell. Since the portion of ∂K lying within this cell is small, the area of its Voronoi patch on the Dudley sphere must be large. A packing argument applied on the Dudley sphere will be used to limit the number of these bad cells.

In order to formalize the notion of good and bad cells, let T denote the quadtree produced by SplitReduce(K,Q_0), and let T' denote the subtree of T induced by cells of diameter at least $\sqrt{\varepsilon}/2$. For the remainder of this section, let L_1 denote the (good) leaf cells of T' that are not subdivided further by the algorithm, and let L_2 be the remaining (bad) leaf cells of T'. The cells of L_1 and L_2 are all of diameter $\Omega(\sqrt{\varepsilon})$. Each cell in L_1 can be approximated using at most t halfspaces, and those in L_2 require more. In our next lemma, we bound the total number of approximating halfspaces over all the good leaf cells and the total number of bad leaf cells.

LEMMA 6.2. *Let K be a fat convex body in $\mathbb{R}^d$, and let $0 < \varepsilon \leq 1/8$. Let T denote the quadtree produced by SplitReduce(K,Q_0) for $t \geq 1$, and let L_1 and L_2 be as defined above. Then*

(i) $\sum_{Q \in L_1} t(Q) = O(1/\varepsilon^{(d-1)/2})$,
(ii) $|L_2| = O((1+\log t)/t)^2 (1/\varepsilon)^{(d-1)/2})$.

Proof. Because the cells of L_1 are disjoint and each is of diameter $\Omega(\sqrt{\varepsilon})$, assertion (i) follows as a direct consequence of Lemma 4.4. Thus, it remains to prove assertion (ii). Let Q be any cell of L_2. Since any child of a cell of L_2 is of diameter smaller than $\sqrt{\varepsilon}/2$ and Q's diameter is twice this, we have $\sqrt{\varepsilon}/2 \leq \text{diam}(Q) < \sqrt{\varepsilon}$. Recall that $E^+(K,Q) = E_{2\sqrt{\varepsilon}}(K,Q)$. Also, let $A_K(Q)$ and $A_S(Q)$ denote the values of A_K and A_S, respectively, from the proof of Lemma 6.1, when applied to Q.

Because $\text{diam}(Q) \leq \sqrt{\varepsilon}$ and $E^+(K,Q)$ involves a boundary patch of K that intersects Q and includes an additional expansion by distance $2\sqrt{\varepsilon}$, it follows that this boundary patch has diameter $O(\sqrt{\varepsilon})$. Therefore, $A_K(Q) = O(\varepsilon^{(d-1)/2})$. By applying Lemma 6.1 (and recalling the constant c_a from Lemma 5.2), we have $t(Q) = O(r \log r)$, where

$$r = \left(\frac{\text{area}(E^+(K,Q)) \cdot \text{area}\left(\text{Vor}(E^+(K,Q)) \cap S\right)}{c_a \cdot \varepsilon^{d-1}}\right)^{1/2}$$
$$= \left(\frac{A_K(Q) A_S(Q)}{c_a \cdot \varepsilon^{d-1}}\right)^{1/2} = O\left(\sqrt{\frac{A_S(Q)}{\varepsilon^{(d-1)/2}}}\right).$$

In Lemma 6.1 we showed that (after a suitable adjustment to c_a) we have $\log r \geq 1$. Since Q is subdivided further, we know that $t(Q) > t$, which implies that $t = O(r \log r)$. Because $t \geq 1$, by simple manipulations we have $t/(1 + \log t) = O(r)$. By combining this with the upper bound on r from above, we obtain $A_S(Q) = \Omega((t/(1 + \log t))^2 \varepsilon^{(d-1)/2})$, which yields the lower bound

$$\sum_{Q \in L_2} A_S(Q) = |L_2| \cdot \Omega\left(\left(\frac{t}{1 + \log t}\right)^2 \varepsilon^{\frac{d-1}{2}}\right).$$

As shown in the proof of Lemma 4.4, given any set of disjoint quadtree cells of diameter $\Omega(\sqrt{\varepsilon})$ a point of S can be in $\text{Vor}(E^+(K,Q))$ for at most a constant number of these cells. Since the quadtree cells of L_2 satisfy these conditions,

$$\sum_{Q \in L_2} A_S(Q) = \sum_{Q \in L_2} \text{area}(\text{Vor}(E^+(K,Q)) \cap S) = O(\text{area}(S)).$$

Combining this with our lower bound, we have

$$|L_2| = O\left(\text{area}(S) \cdot \left(\frac{1 + \log t}{t}\right)^2 \cdot \left(\frac{1}{\varepsilon}\right)^{\frac{d-1}{2}}\right).$$

Since S is a hypersphere of constant radius, its area is bounded, and assertion (ii) follows immediately. □

Recall that we showed in Lemma 4.5 that it is possible to answer approximate membership queries in $1/\varepsilon^{(d-1)/4}$ time using space $O(1/\varepsilon^{(d-1)/2})$. By using the above lemma, we show next that we can improve this to achieving query time roughly $O(1/\varepsilon^{(d-1)/8})$ for the same space.

LEMMA 6.3. *Let K be a fat convex body in $\mathbb{R}^d$, and let $0 < \varepsilon \leq 1/16$. For $t \geq (\lg \frac{1}{\varepsilon})/\varepsilon^{(d-1)/8}$, the output of SplitReduce(K, Q_0) has total space $O(1/\varepsilon^{(d-1)/2})$.*

Proof. Let T denote the quadtree produced by the algorithm. By Lemma 3.1, the data structure's total space is dominated by the space needed to store the hyperplanes in the leaf cells. Thus, it suffices to show that the sum of $t(Q)$ over all leaf cells Q of T is $O(1/\varepsilon^{(d-1)/2})$. Let T', L_1, and L_2 be as defined just prior to Lemma 6.2. By Lemma 6.2(i), the total contribution of $t(Q)$ for all cells in L_1 is $O(1/\varepsilon^{(d-1)/2})$. So, it suffices to bound the contribution due to L_2.

Let Q be any cell of L_2. Recall from the proof of Lemma 6.2 that $\sqrt{\varepsilon}/2 \leq \text{diam}(Q) \leq \sqrt{\varepsilon}$. Since $t \geq 1/\varepsilon^{(d-1)/8}$, it follows that $t \geq (\text{diam}(Q)/\varepsilon)^{(d-1)/4}$. Because $\varepsilon \leq 1/16$, we have $\varepsilon \leq \sqrt{\varepsilon}/4 \leq \text{diam}(Q)/2$. By Lemma 4.6, the output of SplitReduce(K, Q) has total space at most

$$O\left(\left(\frac{\mathrm{diam}(Q)}{\varepsilon}\right)^{\frac{d-1}{2}}\right) \;=\; O\left(\left(\frac{1}{\varepsilon}\right)^{\frac{d-1}{4}}\right).$$

By Lemma 6.2(ii), $|L_2| = O(((1+\log t)/t)^2(1/\varepsilon)^{(d-1)/2})$. Since $t \geq (\lg \frac{1}{\varepsilon})/\varepsilon^{(d-1)/8}$, we have $|L_2| = O(1/\varepsilon^{(d-1)/4})$. Summing up the space contributions of all $Q \in L_2$, the total space for these cells is

$$|L_2| \cdot O\big(1/\varepsilon^{(d-1)/4}\big) \;=\; O\big(1/\big(\varepsilon^{(d-1)/4} \cdot \varepsilon^{(d-1)/4}\big)\big) \;=\; O\big(1/\varepsilon^{(d-1)/2}\big),$$

as desired. □

In order to extend the space-time trade-off to other query times, we will apply the previous result as the basis case in an induction argument. The induction will be controlled by a parameter α, which we assume to be a constant. The proof is rather technical, but it involves a straightforward application of the earlier results of this section.

Lemma 6.4. *Let K be a fat convex body in $\mathbb{R}^d$, and let $0 < \varepsilon \leq 1/16$. Let $\alpha \geq 4$ be a real-valued constant. For $t \geq (\lg \frac{1}{\varepsilon})/\varepsilon^{(d-1)/\alpha}$, the output of SplitReduce(K, Q_0) has total space*

$$O\left(1/\varepsilon^{(d-1)\left(1-2\left(\frac{\lfloor \lg \alpha \rfloor - 2}{\alpha} + \frac{1}{2^{\lfloor \lg \alpha \rfloor}}\right)\right)}\right).$$

Proof. Define $k = \lfloor \lg \alpha \rfloor$, which implies that $k \geq 2$, and $2^k \leq \alpha < 2^{k+1}$. Expressed as a function of k, the desired space bound can be expressed as

$$c_k \cdot \left(1/\varepsilon^{(d-1)\left(1-2\left(\frac{k-2}{\alpha} + \frac{1}{2^k}\right)\right)}\right) \tag{1}$$

for a constant c_k (depending on k but not on ε).

We begin exactly as in the proof of the previous lemma. Let T denote the quadtree produced by the algorithm, and by Lemma 3.1, it suffices to bound the sum of $t(Q)$ over all leaf cells of T. Given T', L_1, and L_2 defined prior to Lemma 6.2, the space contribution due to the cells of L_1 is $O(1/\varepsilon^{(d-1)/2})$. To see that this satisfies our space bound, observe that since $k \geq 2$ and $\alpha \geq 2^k$, we have

$$\frac{1}{4} \;\geq\; \frac{k-1}{2^k} \;=\; \frac{k-2}{2^k} + \frac{1}{2^k} \;\geq\; \frac{k-2}{\alpha} + \frac{1}{2^k}.$$

Therefore, the total contribution of $t(Q)$ for all cells in L_1 is

$$O\big(1/\varepsilon^{(d-1)/2}\big) \;=\; O\Big(1/\varepsilon^{(d-1)\left(1-2\left(\frac{1}{4}\right)\right)}\Big) \;\leq\; O\Big(1/\varepsilon^{(d-1)\left(1-2\left(\frac{k-2}{\alpha} + \frac{1}{2^k}\right)\right)}\Big), \tag{2}$$

which matches the desired bound given in (1).

It remains to bound the contribution to the space of the cells of L_2. We do this by induction on k. For the basis case $k = 2$, we have $4 \leq \alpha < 8$. Therefore $t > (\lg \frac{1}{\varepsilon})/\varepsilon^{(d-1)/8}$. By applying Lemma 6.3, the total space of the data structure (which includes the contribution of L_2) is $O(1/\varepsilon^{(d-1)/2})$. It follows from (2) (for the case $k = 2$) that this satisfies our storage bound.

For the induction step, we assume that the lemma holds for $k-1$ (that is, $2^{k-1} \leq \alpha/2 < 2^k$), and our objective is to prove it for k. It will be convenient to express the induction hypothesis in a form that holds for an arbitrary quadtree cell $Q \subseteq Q_0$.

By applying standardization to Q (thus mapping Q to Q_0 and scaling ε to $\varepsilon/\text{diam}(Q)$), the induction hypothesis states that for

$$0 < \varepsilon \leq \frac{\text{diam}(Q)}{16} \text{ and } t \geq \left(\lg \frac{\text{diam}(Q)}{\varepsilon}\right) \cdot \left(\frac{\text{diam}(Q)}{\varepsilon}\right)^{\frac{d-1}{\alpha/2}}, \tag{3}$$

there is a constant c_{k-1} such that the output of SplitReduce(K, Q) has total space at most

$$c_{k-1} \cdot (\text{diam}(Q)/\varepsilon)^{(d-1)\left(1-2\left(\frac{k-3}{\alpha/2}+\frac{1}{2^{k-1}}\right)\right)}. \tag{4}$$

Let Q be any cell of L_2. In the proof of Lemma 6.2 we showed that $\sqrt{\varepsilon}/2 \leq \text{diam}(Q) < \sqrt{\varepsilon}$. By the bound on t from the statement of this lemma, we have

$$t \geq \left(\lg \frac{1}{\varepsilon}\right)\left(\frac{1}{\varepsilon}\right)^{\frac{d-1}{\alpha}} \geq \left(\lg \frac{1}{\sqrt{\varepsilon}}\right)\left(\frac{1}{\sqrt{\varepsilon}}\right)^{\frac{2(d-1)}{\alpha}} \geq \left(\lg \frac{\text{diam}(Q)}{\varepsilon}\right)\left(\frac{\text{diam}(Q)}{\varepsilon}\right)^{\frac{d-1}{\alpha/2}},$$

implying that t satisfies (3). If ε is at most $\text{diam}(Q)/16$, we may apply the induction hypothesis, yielding the space bound given in (4). Since $\text{diam}(Q) < \sqrt{\varepsilon}$, this can be simplified to

$$c_{k-1} \cdot (1/\sqrt{\varepsilon})^{(d-1)\left(1-2\left(\frac{k-3}{\alpha/2}+\frac{1}{2^{k-1}}\right)\right)} = c_{k-1} \cdot 1/\varepsilon^{(d-1)\left(\frac{1}{2}-\frac{2(k-3)}{\alpha}-\frac{1}{2^{k-1}}\right)}. \tag{5}$$

By combining Lemma 6.2(ii) with the lower bound on t given in the statement of this lemma, the number of cells in L_2 satisfies

$$|L_2| = O\left(\left(\frac{\lg t}{t}\right)^2\left(\frac{1}{\varepsilon}\right)^{\frac{d-1}{2}}\right) = O\left(\varepsilon^{\frac{2(d-1)}{\alpha}}\left(\frac{1}{\varepsilon}\right)^{\frac{d-1}{2}}\right) = O\left(\left(\frac{1}{\varepsilon}\right)^{(d-1)\left(\frac{1}{2}-\frac{2}{\alpha}\right)}\right). \tag{6}$$

The total contribution to the space by the cells of L_2 is the product of the space requirements for each cell of L_2, given in (5), and the number of such cells, given in (6). There exists a constant c_k (depending on k but not on ε) such that the total space is at most

$$c_k \cdot \left(1/\varepsilon^{(d-1)\left(\left(\frac{1}{2}-\frac{2(k-3)}{\alpha}-\frac{1}{2^{k-1}}\right)+\left(\frac{1}{2}-\frac{2}{\alpha}\right)\right)}\right) = c_k \cdot \left(1/\varepsilon^{(d-1)\left(1-2\left(\frac{k-2}{\alpha}+\frac{1}{2^k}\right)\right)}\right).$$

On the other hand, if ε exceeds $\text{diam}(Q)/16$, then since $\text{diam}(Q) \geq \sqrt{\varepsilon}/2$ it follows that ε is $\Omega(1)$, and we can adjust to c_k to satisfy this bound. In either case, we achieve the bound in (1). □

Observe that the exponent in the space bound in the preceding lemma is a piecewise linear function in $1/\alpha$, whose breakpoints coincide with powers of two. It is easily verified that the exponent is a continuous function of α. (In particular, observe that $\lim_{\delta \to 0} f(2^{k-\delta}) = f(2^k)$, where $f(\alpha) = 1/2^{\lfloor \lg \alpha \rfloor} + (\lfloor \lg \alpha \rfloor - 2)/\alpha$.)

We can now present the proof of Theorem 1.2. Recall that K is a convex polytope in $\mathbb{R}^d$. By Lemma 2.1, we can precondition K so that it is $(1/d)$-fat and is contained within Q_0, thus allowing us to approximate K absolutely. Also, if $1/16 < \varepsilon \leq 1$, we set $\varepsilon = 1/16$. (Both of these changes result in a constant factor decrease to ε, which will not affect the asymptotic bounds.) We then set $t = (\lg \frac{1}{\varepsilon})/\varepsilon^{(d-1)/\alpha}$ and invoke SplitReduce(K, Q_0). Let T denote the resulting data structure. Given the

preconditioning of K and the alteration of ε, we may apply Lemma 6.4 to show that the total space for T is

$$O\left(1/\varepsilon^{(d-1)\left(1-2\left(\frac{\lfloor \lg \alpha \rfloor -2}{\alpha}+\frac{1}{2^{\lfloor \lg \alpha \rfloor}}\right)\right)}\right).$$

Using the fact that $1/2^{\lfloor \lg \alpha \rfloor} \geq 1/\alpha$, this is

$$O\left(1/\varepsilon^{(d-1)\left(1-2\left(\frac{\lfloor \lg \alpha \rfloor -2}{\alpha}+\frac{1}{\alpha}\right)\right)}\right) \;=\; O\left(1/\varepsilon^{(d-1)\left(1-\frac{2\lfloor \lg \alpha \rfloor -2}{\alpha}\right)}\right),$$

which matches the space bound of Theorem 1.2.

Recall that a query is answered by locating the leaf node of T that contains the query point, followed by an inspection of the (at most) t halfspaces stored in this leaf node. By our remarks following the presentation of SplitReduce, T is of height $O(\log \frac{1}{\varepsilon})$, which implies that the query time is dominated by the value of t. This completes the proof of Theorem 1.2.

7. Preprocessing. Our principal focus so far has been on establishing the existence of trade-offs between space and query time, without considering how to construct the data structure. In this section we discuss preprocessing issues. We first discuss the preconditioning of K as described in Lemma 2.1 and then discuss the implementation of the access primitives (i)–(iii) needed for SplitReduce as presented at the start of section 3. We assume that the input convex body K is presented as the intersection of a set $\mathcal{H}$ of n halfspaces in $\mathbb{R}^d$. Throughout, let Q denote an arbitrary quadtree cell.

Let t denote the query-time parameter in SplitReduce. As observed in section 3, under our assumption that $t \geq 1$, steps 1 and 2 are not needed, since we can rely entirely on step 3, and therefore access primitives (i) and (ii) are not needed.[2] The remainder of this section will be focused on preconditioning (section 7.1) and the implementation of access primitive (iii), which locally approximates K within Q (section 7.2).

7.1. Preconditioning. Recall that we assume that K is a (full-dimensional) convex polytope in $\mathbb{R}^d$ that is presented as the intersection of a set of n closed halfspaces. Also recall that Q_0 is the axis-aligned hypercube of unit diameter that is centered at the origin. Our objective is to precondition K by computing an affine transformation that both fattens K and maps it to lie within Q_0. Q_0 has a side length of $1/\sqrt{d}$, and therefore it contains a ball of radius $1/2\sqrt{d}$ centered at the origin. Let B_0 denote this ball, and let r_0 denote its radius. For $0 < \gamma \leq 1$, let γB_0 denote the concentric ball of radius $\gamma r_0 = \gamma/2\sqrt{d}$. We say that a polytope is in *γ-canonical position* if it is nested between γB_0 and B_0 (see Figure 11). Clearly, a polytope that is in canonical position is contained within Q_0 and is γ-fat. The following lemma shows that K can be efficiently mapped into this form, and furthermore an absolute approximation to the transformed body can be easily mapped to a relative approximation of K. (Lemma 2.1 follows as an immediate consequence of this.) Such fattening operations are commonplace in geometric approximation algorithms (see, e.g., [1, 23, 39, 14]), and we employ the standard approach based on minimum enclosing volumes, the John ellipsoid in particular.

[2]If we wished to we could implement access primitive (i) in linear time by linear programming. Also, by testing the membership of each of Q's vertices in K, we could implement a stronger version of access primitive (ii), namely, that of determining whether $Q \subseteq K$ (as opposed to $K \oplus \varepsilon$).

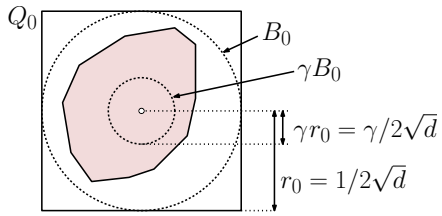


FIG. 11. *A polytope in γ-canonical position.*

LEMMA 7.1. *Let K be a convex polytope in $\mathbb{R}^d$ defined as the intersection of a set $\mathcal{H}$ of n halfspaces, and let $0 < \varepsilon \leq 1$. There is an algorithm that, given $\mathcal{H}$ and ε, in $O(n)$ time computes an affine transformation T that maps K into $(1/d)$-canonical position, such that if P is an absolute $\varepsilon/(d\sqrt{d})$-approximation of $T(K)$, then $T^{-1}(P)$ is a relative ε-approximation of K.*

Proof. Chazelle and Matoušek [26] show that in any fixed dimension, there exists an $O(n)$ time algorithm that, given a convex polytope K presented as the intersection of n halfspaces, computes an ellipsoid E of maximum volume contained within K, also known as the *John ellipsoid* [12]. (At the expense of an increase in the constant factors, we can apply the simpler construction by Barequet and Har-Peled [14].) It is well known from John's theorem (see, e.g., [12]) that K is contained within a uniform scaling of E by a factor of d. It follows from basic linear algebra that the transformation T that maps E to a Euclidean ball $\frac{1}{d}B_0$ achieves the desired result. (Details can be found in [8].) □

In order to make subsequent processing more efficient, we adapt a standard coreset construction to reduce the number of halfspaces to a function depending only on ε and d. The process will involve some further scaling, which will slightly modify the parameters.

LEMMA 7.2. *Let K be a convex polytope in $\mathbb{R}^d$ defined as the intersection of a set $\mathcal{H}$ of n halfspaces, and let $0 < \varepsilon \leq 1$. There is an algorithm that, given $\mathcal{H}$ and ε, in $O(n + 1/\varepsilon^{d-1})$ time computes an affine transformation T' and a subset $\mathcal{H}' \subseteq \mathcal{H}$ of size $O(1/\varepsilon^{(d-1)/2})$ such that*

(i) *applying T' to the intersection of $\mathcal{H}'$ results in a convex polytope K' that is in $(1/2d)$-canonical position;*
(ii) *furthermore, if P is an absolute $\varepsilon/(4d\sqrt{d})$-approximation of K', then $T'^{-1}(P)$ is a relative ε-approximation of K.*

Proof. Given $\mathcal{H}$, we begin by computing the transformation T of Lemma 7.1 in $O(n)$ time. Let $T(K)$ denote the resulting polytope, which is in $(1/d)$-canonical position (see Figure 12(a)). Given a set S of points $\mathbb{R}^d$, the *extent measure* associates each unit vector $u \in \mathbb{R}^d$ with the minimum distance between two hyperplanes orthogonal to u that contain S between them. More formally, define $w_u(S) = \max_{p,q \in S} \langle p - q, u \rangle$ (recalling that $\langle \cdot, \cdot \rangle$ denotes inner product). A subset $S' \subseteq S$ is said to be an *ε-coreset* for the extent measure if for all unit vectors u, $w_u(S') \geq (1 - \varepsilon) w_u(S)$. Agarwal, Har-Peled, and Varadarajan [1] showed that, given a set of n points in $\mathbb{R}^d$, it is possible to construct an ε-coreset for the extent measure of size $O(1/\varepsilon^{(d-1)/2})$. Chan presented an algorithm to compute such a coreset in $O(n + (1/\varepsilon)^{d-1})$ time [23].

In order to apply the coreset construction, we first employ the polar dual transformation (recall section 2.3) to $T(K)$, resulting in an n-element point set S of size n

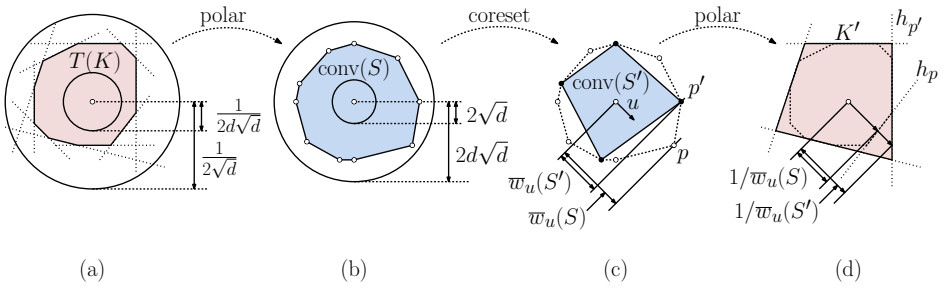


FIG. 12. *Proof of Lemma* 7.2. *(Not drawn to scale.)*

such that $\text{conv}(S) = \text{polar}(T(K))$ (see Figure 12(b)). It is easy to verify that $\text{conv}(S)$ is nested between an inner ball of radius $2\sqrt{d}$ and an outer ball of radius $2d\sqrt{d}$. Let $\varepsilon' = \varepsilon/4d^2$. We then apply Chan's algorithm to compute an ε'-coreset $S' \subseteq S$ in time $O(n + (1/\varepsilon')^{d-1}) = O(n + (1/\varepsilon)^{d-1})$ (see Figure 12(c)). Let $\mathcal{H}'$ be the subset of $\mathcal{H}$ that results by taking the polar duals of the points of S', and let K' be the convex body that results from intersecting these halfspaces (see Figure 12(d)).

Clearly, $T(K) \subseteq K'$ and $|\mathcal{H}'| = O(1/\varepsilon^{(d-1)/2})$. It follows from a straightforward geometric argument that the Hausdorff distance between $T(K)$ and K' is at most $\varepsilon/2d\sqrt{d}$. (For details, see [8]). Therefore, if P is any absolute $(\varepsilon/2d\sqrt{d})$-approximation to K', then by the triangle inequality (applied to the Hausdorff distance) P is an absolute $(\varepsilon/2d\sqrt{d}) + (\varepsilon/2d\sqrt{d}) = \varepsilon/d\sqrt{d}$ approximation to $T(K)$. By Lemma 7.1, P is a relative ε-approximation of K. We are almost done, but the canonical-position condition fails, because K' need not lie within B_0 of radius $r_0 = 1/2\sqrt{d}$ (even though $T(K)$ does). Since the Hausdorff distance between K' and $T(K)$ is at most $\varepsilon/2d\sqrt{d} \leq 1/2\sqrt{d} = r_0$, K' lies within $2B_0$. The simple fix is to apply a uniform scaling of space by a factor of $1/2$ combined with a suitable constant-factor adjustment of ε. The desired conclusion follows as a direct consequence of canonical position. ∎

7.2. Efficient local approximations. Next, we consider the implementation of access primitive (iii), which given a convex body K in γ-canonical position, a quadtree cell Q, and query-time t determines whether there exist t halfspaces whose intersection ε-approximates K within Q. The space and query times stated in Theorem 1.2 are based on the assumption that the number of bounding halfspaces of this local approximating polytope is within a constant factor of optimal. However, we know of no efficient algorithm that can achieve this. In this section we show how to efficiently implement step 3 of SplitReduce approximately in the sense that the number of halfspaces in the approximation exceeds the optimum (for a slightly smaller approximation parameter) by a factor of $O(\log \frac{1}{\varepsilon})$. As shown in Lemma 3.2, this will lead to an increase in the space and query times stated in Theorem 1.2 by a factor of only $O(\log \frac{1}{\varepsilon})$.

A natural approach would be to adapt Clarkson's algorithm for polytope approximation [27]. There are a few messy technical issues involved with such an adaptation. (For example, Clarkson's algorithm applies to the convex hull of a set of points, rather than the intersection of halfspaces.) Since we do not require the strong approximation bounds provided by Clarkson's algorithm, we will instead present a simple direct solution based on a reduction to the set-cover problem. Our approach is to construct a set system where the point set consists of a dense set of points of spacing $\Theta(\varepsilon)$ that covers the portion of Q that is external to $K \oplus c'\varepsilon$ for a suitable constant $c' < 1$.

We associate each bounding halfspace of K with the set of grid points that lie *outside* of this halfspace. We will show that the halfspaces associated with a minimum set cover for this system produce the desired local approximation. We use the greedy set cover heuristic to construct this cover.

Recall that $K \oplus r$ denotes the set of points that lie within Euclidean distance r of K. In order to avoid the complexities of determining whether a point lies outside of $K \oplus c'\varepsilon$, it will suffice for our purposes to perform the simpler test of whether a point lies outside a scaled copy of K. The following lemma follows from a straightforward geometric argument (see [8] for details).

LEMMA 7.3. *For $0 < \gamma \leq 1$ and $0 < \varepsilon \leq 1$, let K be a polytope in $\mathbb{R}^d$ that is in γ-canonical position, and let $K^+ = (1 + 2\sqrt{d}\varepsilon)K$. Then*

$$K \oplus \gamma\varepsilon \;\subseteq\; K^+ \;\subseteq\; K \oplus \varepsilon.$$

While access primitive (iii) does not place any restrictions on the halfspaces used when computing an ε-approximation to K within Q, when the query point q lies outside of K, it may be useful to add further restrictions. In particular, when the query point lies outside of K, it is desirable to obtain a *witness* to nonmembership in the form of a bounding halfspace of K that does not contain q. (This will be exploited in section 9 in the reduction of approximate nearest neighbor searching to approximate polytope membership. The witness hyperplane is used to identify the approximate nearest neighbor.) To achieve this, we would like to use bounding halfspaces from the original polytope in our approximation. By a simple application of Carathéodory's theorem, we can show that we sacrifice only a constant factor by adding this restriction. The following is a straightforward generalization of Lemma 3.1 from Mitchell and Suri [45]. (See [8] for details.)

LEMMA 7.4. *Let K be a convex polytope in $\mathbb{R}^d$ defined as the intersection of a set $\mathcal{H}$ of halfspaces, and let $Q \subseteq Q_0$ be a quadtree cell. If there exists an ε-approximation of K within Q bounded by m halfspaces, then there exists a subset of $\mathcal{H}$ of size at most dm that ε-approximates K within Q.*

We are now in a position to present our set-cover-based local approximation. This is a bicriteria approximation since it is suboptimal with respect to both the number of bounding halfspaces and the approximation parameter.

LEMMA 7.5. *For $0 < \gamma \leq 1$ and $0 < \varepsilon \leq 1$, let K be a polytope in $\mathbb{R}^d$ in γ-canonical position that is given as the intersection of a set $\mathcal{H}$ of n halfspaces. Let $Q \subseteq Q_0$ be a quadtree cell. In $O(n/\varepsilon^d)$ time, it is possible to compute a subset $\mathcal{H}' \subseteq \mathcal{H}$ such that*

(i) *the intersection of the halfspaces of $\mathcal{H}'$ is an ε-approximation of K within Q,*
(ii) *if m denotes the minimum number of halfspaces needed to $(\gamma\varepsilon/2)$-approximate K within Q, then $|\mathcal{H}'|$ is $O(m \log \frac{1}{\varepsilon})$.*

Proof. First, we may assume without loss of generality that $\varepsilon \leq 2/\sqrt{d}$. Otherwise, setting $\varepsilon = 2/\sqrt{d}$ will certainly satisfy (i) and will only affect the constant factors in the asymptotic bounds of claim (ii) and the construction time. Define $\beta = \sqrt{d}\varepsilon/2$. By the above assumption, we have

$$(1+\beta)^2 \;=\; \left(1 + \sqrt{d}\varepsilon + \frac{d\varepsilon^2}{4}\right) \;\leq\; 1 + \frac{3}{2}\sqrt{d}\varepsilon.$$

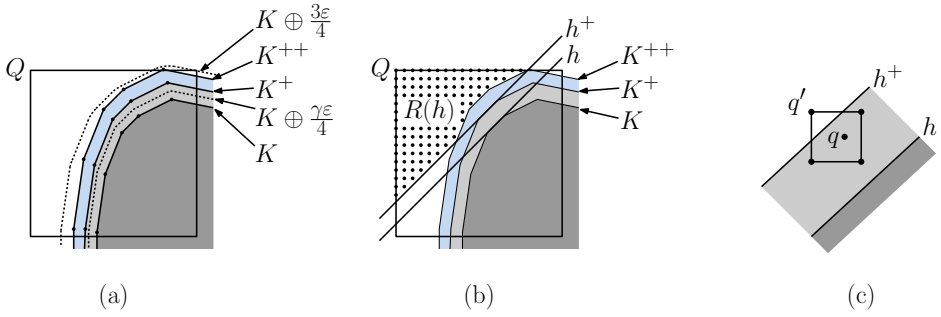


FIG. 13. *Proof of Lemma* 7.5.

Let $K^+ = (1+\beta)K$ and let $K^{++} = (1+\beta)K^+ = (1+\beta)^2 K$. By applying Lemma 7.3 but with ε taking on the values $\varepsilon/4$ and $3\varepsilon/4$, respectively, we have

$$K \oplus \frac{\gamma\varepsilon}{4} \;\subseteq\; K^+ \;\subseteq\; K^{++} \;\subseteq\; K \oplus \frac{3\varepsilon}{4} \tag{7}$$

(see Figure 13(a)). Let $\delta = \gamma\varepsilon/4$ and let G denote the vertices of a hypercube grid of diameter δ. Let R be the set of grid points that lie within Q but outside of K^{++}, that is, $R = G \cap (Q \setminus K^{++})$. Since $Q \subseteq Q_0$, the resulting set is of size $O(1/\varepsilon^d)$, and hence it can be computed in time $O(1/\varepsilon^d) \cdot |\mathcal{H}| = O(n/\varepsilon^d)$, by testing each grid point against each halfspace of $\mathcal{H}$. Because $\gamma \leq 1$, we have $\delta \leq \varepsilon/4$.

Next, we define a set system to model the approximation process. For each $h \in \mathcal{H}$ we define a subset $R(h) \subseteq R$ as follows. First, let $h^+ = (1+\beta)h$ denote the corresponding bounding halfspace of the scaled body K^+ (see Figure 13(b)). Define $R(h)$ to be the subset of points of R that lie outside of h^+. Consider a set system consisting of the points of R and the sets $R(h)$ for all $h \in \mathcal{H}$. Since every point of R lies outside of K^{++}, and hence outside of K^+, together these sets cover R. The resulting collection of sets has total cardinality $O(n/\varepsilon^d)$.

Consider any set cover C of the resulting set system. Let $P(C)$ denote the polyhedron that results by intersecting the halfspaces h whose associated set $R(h)$ is included in this cover. (Note that the sets $R(h)$ are based on the halfspaces bounding the scaled body K^+, while $P(C)$ is based on the halfspaces bounding the original body K.) We assert that $P(C)$ ε-approximates K within Q. It suffices to show that for any point $q \in Q \setminus (K \oplus \varepsilon)$, q is not in $P(C)$. First, observe that for such a point q, all the vertices of the grid cell in which it lies are within distance δ of q. Therefore, by the triangle inequality, each such vertex is at distance at least $\varepsilon - \delta \geq 3\varepsilon/4$ from K. Since by (7), $K^{++} \subseteq K \oplus 3\varepsilon/4$, these vertices are all exterior to K^{++}, which implies that they are all members of R. Let q' be any of these vertices. Since C is a cover, there exists a halfspace $h \in \mathcal{H}$ such that $R(h)$ is in the cover and contains this point. This implies that q' lies outside the associated halfspace h^+ (see Figure 13(c)). Because K is in γ-canonical position, the minimum distance between h's bounding hyperplane and the origin is at least $\gamma/2\sqrt{d}$. Therefore the distance between any point in h to any point exterior to h^+ is at least

$$\frac{\gamma}{2\sqrt{d}}((1+\beta)-1) \;=\; \frac{\gamma}{2\sqrt{d}} \cdot \frac{\sqrt{d}\varepsilon}{2} \;=\; \frac{\gamma\varepsilon}{4} \;=\; \delta.$$

It follows by the triangle inequality that q is exterior to h, and therefore it lies outside of $P(C)$, as desired.

Let C' denote a set cover that results by running the greedy heuristic [29] on the aforementioned set system. By standard results on the greedy heuristic, the size of the resulting cover exceeds that of an optimal cover by a factor of at most $\ln |R| = O(\log \frac{1}{\varepsilon})$. C' can be computed in time that is proportional to the total cardinality of the sets of the set system, which is $O(n/\varepsilon^d)$. Let $\mathcal{H}'$ denote the associated set of halfspaces, and let $P(C')$ denote the intersection of these halfspaces. By the above remarks, $P(C')$ is an ε-approximation to K within Q, which establishes claim (i).

To establish (ii), consider a $(\gamma\varepsilon/4)$-approximation of K within Q that is bounded by the minimum number m of halfspaces. By Lemma 7.4 there exists such an approximation that uses only the bounding halfspaces of K, such that the number of halfspaces is larger by a factor of at most d. Let P^+ denote this approximation, and let $\mathcal{H}^+ \subseteq \mathcal{H}$ denote its bounding halfspaces. By (7), we have $P^+ \subseteq K \oplus \gamma\varepsilon/4 \subseteq K^+$. Let $P^{++} = (1+\beta)P^+$. Clearly, $P^{++} \subseteq (1+\beta)K^+ = K^{++}$. Therefore, every point of R lies outside of P^{++}. It follows that the sets $R(h)$ associated with the halfspaces h that bound P^+ form a set cover of R within our system. Letting C^{++} denote this cover, we have $|C'| \leq O(\log \frac{1}{\varepsilon}) \cdot |C^{++}| \leq O(\log \frac{1}{\varepsilon}) \cdot dm = O(m \log \frac{1}{\varepsilon})$, as desired. ∎

We can now present the main result of this section, which summarizes the preprocessing time.

LEMMA 7.6. *Given a full-dimensional convex polytope K in $\mathbb{R}^d$ defined as the intersection of a set of n halfspaces, approximation parameter $0 < \varepsilon \leq 1$, and query time parameter $t \geq 1$, there is an algorithm that runs in time $O(n + 1/\varepsilon^{c_p d})$ for some constant c_p (which does not depend on d) that constructs a data structure satisfying Theorem 1.2 but with an additional factor of $O(\log \frac{1}{\varepsilon})$ in both the space and query times.*

Proof. Given K's bounding halfspaces, we apply Lemma 7.2. In $O(n + 1/\varepsilon^{d-1})$ time we obtain a polytope K', such that K' is in γ-canonical position for $\gamma = 1/2d$. K' is bounded by a subset $\mathcal{H}'$ of halfspaces of size $n' = O(1/\varepsilon^{(d-1)/2})$, and the problem of computing a relative ε-approximation of K reduces to the problem of computing an absolute ε'-approximation of K', where $\varepsilon' = \varepsilon/4d\sqrt{d}$.

Ideally, we would like to invoke SplitReduce on K' using ε' as the approximation parameter and t as the query time parameter. Since we do not know how to determine minimum-sized convex approximations efficiently, we will need to relax our expectations. For any quadtree cell Q generated by SplitReduce, we apply Lemma 7.5 on the set $\mathcal{H}'$ of halfspaces. By claim (i) of this lemma, after $O(n'/(\varepsilon')^d) = O(1/\varepsilon^{3d/2})$ time, a subset $\mathcal{H}'' \subseteq \mathcal{H}'$ can be computed that is an ε'-approximation of K' within Q. Irrespective of the choice of the query time, the maximum number of quadtree cells generated by SplitReduce is $O(1/\varepsilon^d)$, and therefore (after preconditioning) the overall running time of SplitReduce is $O(1/\varepsilon^{5d/2})$. Combined with the $O(n + 1/\varepsilon^d)$ time for preconditioning, the algorithm's overall running time is $O(n + 1/\varepsilon^{c_p d})$, where $c_p = 5/2$.

Let $\varepsilon'' = \gamma\varepsilon'/2 = \gamma\varepsilon/8d\sqrt{d}$. By Lemma 7.5(ii) the number of halfspaces in $\mathcal{H}''$ is within a factor of $\rho = O(\log \frac{1}{\varepsilon'}) = O(\log \frac{1}{\varepsilon})$ of the size of the minimum-sized ε''-approximation of K' within Q. Since $\varepsilon'' = \beta\varepsilon'$ for a constant β, Lemma 3.2 implies that the conclusions of Theorem 1.2 hold but with an additional factor of $\rho = O(\log \frac{1}{\varepsilon})$ in both the space and query times. ∎

8. Lower bound. In this section, we establish lower bounds on the space-time trade-offs obtained by SplitReduce for polytope membership. In particular, we will prove Theorem 1.3. Our approach is similar to the lower bound proof of [10]. (Note

that this is a lower bound on the performance of SplitReduce, not on the problem complexity. It applies to the stronger existential version of the algorithm.) It is based on analyzing the performance of the algorithm on a particular convex body, a generalized hypercylinder that is curved in $k+1$ dimensions and flat in $d-1-k$ dimensions. We select the value of k that produces the best lower bound on the storage as a function of t, ε, and d. Throughout, we use the term ε-approximation in the absolute sense, as defined in section 2.1.

As mentioned earlier, it is well known that $\Omega(1/\varepsilon^{(d-1)/2})$ facets are required to ε-approximate a Euclidean ball of unit radius (see, e.g., [18]), and this holds for any polytope that is sufficiently close to a ball in terms of Hausdorff distance. The following utility lemma generalizes this observation to different diameters. The proof is a straightforward exercise in geometry. (Details can be found in [8].)

LEMMA 8.1. *Let ε and Δ be real parameters, where $0 < \varepsilon \leq \Delta/4$. There exists a constant c_b and a polytope P in $\mathbb{R}^d$ of diameter at most Δ such that any outer ε-approximation of P requires at least $c_b(\Delta/\varepsilon)^{(d-1)/2}$ facets.*

Intuitively, in order to produce a polytope that is hard to approximate, it should have high curvature. If the curvature is high in all dimensions, however, the polytope will have a small surface area, and this will make it easier to approximate. Our approach is to consider polytopes based on generalized cylinders, which have constant curvature in some dimensions but are flat in others. Our next lemma introduces such a cylindrical polytope where the number of curved dimensions has been carefully chosen to maximize the space needed by our algorithm for a given query time. Theorem 1.3 is an immediate consequence.

LEMMA 8.2. *There exists a polytope P in $\mathbb{R}^d$ such that for all sufficiently small positive ε (depending on d and α) and $t = 1/\varepsilon^{(d-1)/\alpha}$, the output of SplitReduce(K, Q_0) on P has total space*

$$\Omega\left(1/\varepsilon^{(d-1)\left(1-\frac{2\sqrt{2\alpha}-3}{\alpha}\right)-1}\right).$$

Proof. To start, as a function of α, we wish to compute an integer dimension k in order to apply Lemma 8.1. Define reals $\delta = \sqrt{\alpha/2}/(d-1)$, $\kappa = (d-1)\sqrt{2/\alpha}$, and $\kappa' = \kappa(1+\delta)$. We observe first that

$$\kappa' - \kappa \;=\; \delta(d-1)\sqrt{2/\alpha} \;=\; 1.$$

Let $k = \lceil \kappa \rceil$, implying that $\kappa \leq k \leq \kappa'$. (Although we do not include the derivation here, κ has been chosen to produce the best lower bound, but since it is not necessarily an integer, k is obtained by rounding to a nearby integer.) Since $\alpha \geq 4$ and $d \geq 2$, we have $1 \leq k \leq d-1$.

Let c_b denote the constant of Lemma 8.1, and let $\Delta = \varepsilon((2^d+1)t/c_b)^{2/k}$. By our assumptions about d and α, we have $t = 1/\varepsilon^{\Theta(1)}$ and $\Delta = \varepsilon \cdot t^{\Theta(1)}$. It follows that for all sufficiently small ε, $\Delta/4 \geq \varepsilon$. Let h denote the linear subspace spanned by the first $k+1$ coordinate axes. We apply Lemma 8.1 in $\mathbb{R}^{k+1}$ for this value of Δ. The resulting polytope P (lying in h) has the property that the number of facets of any ε-approximation is at least

$$c_b\left(\frac{\Delta}{\varepsilon}\right)^{k/2} \;=\; c_b\left(\frac{\varepsilon\left(\frac{(2^d+1)t}{c_b}\right)^{2/k}}{\varepsilon}\right)^{k/2} \;=\; (2^d+1)t.$$

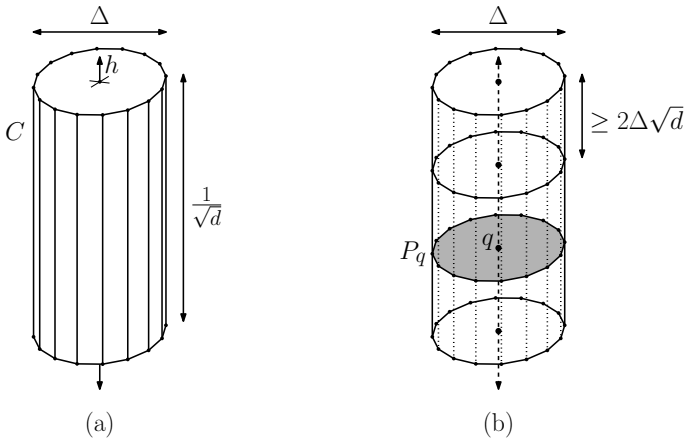


FIG. 14. *Lemma* 8.2 *for* $d = 3$ *and* $k = 2$.

We can bound P's diameter by observing that for all sufficiently small ε

$$\begin{aligned}\operatorname{diam}(P) \leq \Delta \;&=\; \varepsilon\left(\frac{(2^d+1)t}{c_b}\right)^{2/k} \;\leq\; \varepsilon\left(\frac{2^d+1}{c_b \cdot \varepsilon^{(d-1)/\alpha}}\right)^{2/\kappa} \\ &= \varepsilon\left(\frac{2^d+1}{c_b \cdot \varepsilon^{(d-1)/\alpha}}\right)^{\sqrt{2\alpha}/(d-1)}.\end{aligned}$$

(Here we made use of the fact that for all sufficiently small ε, the quantity raised to power of $2/k$ is greater than 1.) Letting $c_b' = ((2^d+1)/c_b)^{\sqrt{2\alpha}/(d-1)}$, we obtain

$$\operatorname{diam}(P) \;\leq\; c_b'\varepsilon\left(\frac{1}{\varepsilon^{(d-1)/\alpha}}\right)^{\sqrt{2\alpha}/(d-1)} \;=\; c_b'\varepsilon^{1-\sqrt{2/\alpha}}.$$

Since $\alpha \geq 4$, for all sufficiently small ε, we have $\operatorname{diam}(P) \leq 1/\sqrt{d}$. Therefore, P can be enclosed within $Q_0^{(k+1)}$.

Returning to $\mathbb{R}^d$, consider an infinite polyhedral hypercylinder whose "axis" is the $(d-1-k)$-dimensional orthogonal complement of h and whose "cross section" (i.e., intersection with any $(k+1)$-dimensional hyperplane parallel to h) is P. Define the polytope C to be the truncated cylinder obtained by intersecting the infinite hypercylinder with hypercube $Q_0^{(d)}$ (see Figure 14(a)). Let T denote the output of $\text{SplitReduce}(K, Q_0^{(d)})$ for C, ε, and t. We will show that T's total space satisfies the bound given in the lemma's statement. To do this, let Σ denote any set of points placed on C's axis such that the distance between each pair of points is at least $2\Delta\sqrt{d}$. (In the degenerate case where $k = d-1$ the axis is 0-dimensional and Σ degenerates to a single point.) By a simple packing argument, there exists such a set having $\Omega(1/\Delta^{d-1-k})$ points.

For any $q \in \Sigma$, let P_q denote the cross-section of C passing through q (see Figure 14(b)). Consider the set of leaf cells of T that intersect P_q. By applying Lemma 8.1 to the $(k+1)$-dimensional hyperplane on which P lies, it follows that these cells together must contain at least $(2^d+1)t$ halfspaces. We count the contributions of these cells by classifying them into two types. We say that a leaf cell of T is *large* if its side length is at least Δ, and otherwise it is *small*. By a simple packing

argument, the number of large leaf cells intersecting P_q is at most 2^d. Since each leaf cell contains at most t halfspaces, the large leaf cells can together contain at most $2^d t$ halfspaces.

Therefore, the small leaf cells intersecting P_q together contain at least $(2^d + 1)t - 2^d t = t$ halfspaces. Because the points of Σ are separated from each other by distance at least $2\Delta\sqrt{d}$, which is strictly larger than the diameter of any small leaf cell, each small leaf cell can intersect P_q for at most one $q \in \Sigma$. Therefore, the total space contribution of all the small leaf cells for all points of Σ is at least $t \cdot |\Sigma|$. Let $c_b'' = (c_b/(2^d + 1))^{2(d-1-k)/k}$. T's total space can be asymptotically bounded from below as

$$\frac{t}{\Delta^{d-1-k}} = \frac{t}{\left(\varepsilon\left(\frac{(2^d+1)t}{c_b}\right)^{2/k}\right)^{d-1-k}} = \frac{c_b'' \cdot t}{\left(\varepsilon \cdot t^{2/k}\right)^{d-1-k}} = \frac{c_b'' \cdot t^{1-2(d-1-k)/k}}{\varepsilon^{d-1-k}}.$$

Clearly, $c_b'' = \Theta(1)$. Recall that $t = 1/\varepsilon^{(d-1)/\alpha}$. Then, T's total space is asymptotically bounded from below as

$$\left(\frac{1}{\varepsilon}\right)^{(d-1)-k+\frac{d-1}{\alpha}\left(1-\frac{2(d-1-k)}{k}\right)} = \left(\frac{1}{\varepsilon}\right)^{(d-1)-k+\frac{d-1}{\alpha}\left(3-\frac{2(d-1)}{k}\right)}. \tag{8}$$

Let $E(\alpha)$ denote this exponent. In order to complete the proof, we provide a lower bound on $E(\alpha)$. We use the fact that $\kappa \le k \le \kappa'$, apply the definitions of κ, κ', and δ, and apply straightforward manipulations to obtain

$$E(\alpha) \ge (d-1) - \kappa' + \frac{d-1}{\alpha}\left(3 - \frac{2(d-1)}{\kappa}\right) = (d-1)\left(1 - \frac{2\sqrt{2\alpha} - 3}{\alpha}\right) - 1.$$

Substituting this value for the exponent in (8) completes the proof. ∎

9. Approximate nearest neighbor searching. In this section, we present a reduction from approximate nearest neighbor searching to approximate polytope membership, which will allow us to prove Theorem 1.4. Our reduction will involve the following additional assumptions regarding the implementation of SplitReduce. First (as in section 7), we assume that K is presented as the intersection of n halfspaces. Second, we assume that a leaf node is labeled as "inside" only if it lies entirely within K (as opposed to lying within $K \oplus \varepsilon$ as described in SplitReduce). Third, we assume that leaf cells that store halfspaces use only bounding halfspaces of K.

Clearly, these assumptions do not affect the data structure's correctness. We assert that they do not affect the data structure's asymptotic query time or space bounds. Regarding the second assumption, observe that for any cell Q that lies within $K \oplus \varepsilon$, K can be ε-approximated within Q using a single halfspace (any halfspace that contains Q suffices). Regarding the third assumption, recall that Lemma 7.4 shows that we may assume that the approximating halfspaces for each node are drawn from the input halfspaces at the expense of a constant factor increase in the query time.

The reduction from approximate nearest neighbor searching to approximate polytope membership is based on the AVD construction from [10]. The AVD employs a height balanced variant of a quadtree, a balanced box decomposition (BBD) tree [11] to be precise. Each cell of a BBD tree corresponds to the set theoretic difference of two quadtree cells, an *outer box* and an optional *inner box*. Each leaf cell of the tree stores a set of *representative points* with the property that for any query point q lying

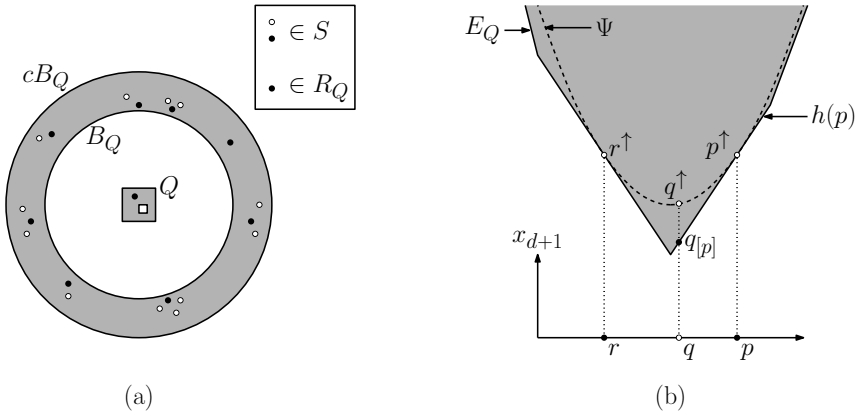


FIG. 15. *Approximate nearest neighbor searching:* (a) *Lemma* 9.1 *(black points are members of* R_Q*),* (b) *the lifting transformation. (Note that the figure is not drawn to scale, and the paraboloid in* (b) *has been translated to aid legibility.)*

within this cell, at least one of these representatives is an ε-nearest neighbor of q. A query is answered by locating the leaf cell that contains the query point and then determining the nearest representative from this cell (by brute force). The AVD's space is dominated by the total number of representatives over all the leaf cells. The query time is the height of the tree plus the number of representatives in the leaf cell. A data structure for nearest neighbor searching is said to be in the *AVD model* if it has this general form, that is, a covering of the query region by hyperrectangles of bounded aspect ratio, each of which is associated with a set of representative points [10]. Lower bounds on the performance of any data structure in the AVD model were given in [10].

The reader need not be familiar with the details of the AVD data structure. The next lemma encapsulates the important technical information needed for our reduction. It follows easily from the proofs of Lemmas 6.1 and 8.1 in [10]. Given a cell Q in a BBD tree, let B_Q denote the ball of radius $2 \cdot \mathrm{diam}(Q)$ whose center coincides with the center of Q's outer box (see Figure 15(a)). Given a Euclidean ball B of radius r and positive c, let cB denote the ball concentric with B of radius cr.

LEMMA 9.1. *Let* $0 < \varepsilon \leq 1/2$ *be a real parameter and* X *be a set of* n *points in* $\mathbb{R}^d$*. It is possible to construct a BBD tree* T *with* $O(n \cdot \log \frac{1}{\varepsilon})$ *nodes, where each leaf cell* Q *stores a subset* $R_Q \subset X$ *satisfying the following properties:*

(i) *For any point* q *in* Q*, one of the points in* R_Q *is an* ε*-approximate nearest neighbor of* q*.*
(ii) *At most one point of* R_Q *is contained in the ball* B_Q*, and the remaining points of* R_Q *are contained in* $c_q B_Q \setminus B_Q$ *for some constant* c_q *(which depends on the dimension).*
(iii) *The total number of representative points over all the leaf cells of* T *is* $O(n \cdot \log \frac{1}{\varepsilon})$*.*

Moreover, it is possible to compute the tree T *and the sets* R_Q *for all the leaf cells in total time* $O(n \cdot \log n \cdot \log \frac{1}{\varepsilon})$*, and the cell that contains a query point can be located in time* $O(\log n + \log \log \frac{1}{\varepsilon})$*.*

In the AVD data structure of [10] the closest representative point to a query point is determined by brute-force enumeration of the elements of R_Q. We consider whether it is possible search them more efficiently by reduction to polytope approximation.

The following lemma explains how to connect Lemma 9.1 with approximate polytope membership queries. Our construction uses the well-known *lifting transformation* [3, 34]. Let $(x_1, \ldots, x_{d+1})$ denote the coordinates of $\mathbb{R}^{d+1}$, and let us think of the $(d+1)$st coordinate axis as being directed vertically upward. Let Ψ denote the paraboloid $x_{d+1} = \sum_{i=1}^{d} x_i^2$. Given a point $p \in \mathbb{R}^d$, let $p^\uparrow$ denote the vertical projection of p onto Ψ (see Figure 15(b)), and let $h(p)$ denote the hyperplane tangent to Ψ at $p^\uparrow$. That is, the points of $h(p)$ satisfy $x_{d+1} = \sum_{i=1}^{d} 2p_i x_i - \|p\|^2$. Given $q \in \mathbb{R}^d$, let $q_{[p]}$ denote the point on $h(p)$ hit by a vertical ray shot downward from $q^\uparrow$. A straightforward consequence of the definition of Ψ is that the squared distance between q and p in $\mathbb{R}^d$ is equal to the length of this vertical segment, that is, $\|qp\|^2 = \|q^\uparrow q_{[p]}\|$.

This suggests the following approach to computing the closest representative point through vertical ray shooting. Consider the (unbounded) convex polyhedron that results by taking the upper envelope of the hyperplanes $h(p)$ associated with the lifted representatives. Given the query point $q \in \mathbb{R}^d$, a ray shot vertically downward from $q^\uparrow$ hits some facet of this polyhedron. It follows from the above remarks that the representative associated with this hyperplane is the closest to q. We can simulate ray shooting by applying polytope membership queries in concert with binary search. Of course, some care will be needed to map this problem into our context, which assumes a bounded polytope and approximation.

LEMMA 9.2. *Let $0 < \varepsilon \leq 1/2$ be a real parameter and consider a quadtree cell Q and a set of representative points R_Q as in Lemma 9.1. Given a data structure for ε-approximate polytope membership in d-dimensional space with query time $t_d(\varepsilon)$ and space $s_d(\varepsilon)$, it is possible to preprocess R_Q into an approximate nearest neighbor data structure for query points in Q with query time $O(t_{d+1}(\varepsilon) \cdot \log \frac{1}{\varepsilon})$ and space $O(s_{d+1}(\varepsilon))$.*

Proof. Since at most one point of R_Q is contained in B_Q, the corresponding point may be inspected separately without increasing the complexity bounds. Therefore, we may assume that all points of R_Q are contained in $c_q B_Q \setminus B_Q$.

Although we assume that the errors in polytope membership are absolute (because of standardization), errors in approximate nearest neighbor searching are relative. That is, a point r is an ε-approximate nearest neighbor of q if $\|qr\| \leq (1+\varepsilon)\|qp\|$, where p is q's true nearest neighbor. Because errors are relative, we may assume that space has been translated and uniformly scaled so that Q is mapped to $Q_0^{(d)}$, the hypercube of unit diameter centered at the origin in $\mathbb{R}^d$. As a result, B_Q is mapped to a ball of radius 2. It follows that the distance from any point of Q to any point of R_Q is greater than 1. Therefore, an absolute error of ε implies a relative error of at most ε.

In order to reduce nearest neighbor searching among the points of R_Q to polytope membership, let E_Q denote the upper envelope, that is, the intersection of the upper halfspaces, of the hyperplanes $h(p)$, for all $p \in R_Q$ (the shaded region in Figure 15(b)). As mentioned above, the facet of E_Q hit by shooting a ray vertically downward from $q^\uparrow$ corresponds to the closest point of R_Q to q.

Since the upper envelope is unbounded, we first compute a bounded convex polytope on which to perform approximate membership queries. Because the query points lie in Q, we are only interested in the portion of E_Q that projects vertically onto Q. Given that the distance of any point $p \in R_Q$ to the origin is at most $2c_q = O(1)$, it follows that the portion of E_Q of interest fits within an axis-aligned $(d+1)$-dimensional hypercube of constant diameter that is centered at the origin. Let Q' denote such a hypercube, let $K_Q = E_Q \cap Q'$, and let $\varepsilon' = \varepsilon/6c_q$. We invoke SplitReduce to construct an ε'-approximate membership data structure for K_Q. (More formally, we first scale Q' into standard form, and we scale ε' by the same factor. We then apply SplitReduce

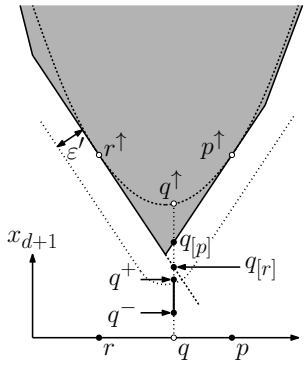


FIG. 16. *Proof of Lemma* 9.2. *(Not drawn to scale.)*

with the scaled value of ε'. Since Q' is of constant diameter, the scale factor will also be a constant, and therefore only the constant factors in the analysis will be affected. We then apply an inverse scaling to obtain the desired ε'-approximating polytope for K_Q.)

We simulate the ray shooting process by a binary search to locate the contact point approximately. Consider the vertical segment formed by intersecting Q' with the vertical line passing through $q^{\uparrow}$. The upper endpoint of this segment is clearly inside K_Q and its lower endpoint is outside. We repeatedly split the segment at its midpoint, perform an approximate polytope membership query, and retain the subsegment whose upper endpoint is (approximately) inside K_Q and whose lower endpoint is (approximately) outside. We terminate the search when the length of the segment falls below ε'. Since Q' is of constant diameter, the search terminates after $O(\log \frac{1}{\varepsilon})$ membership queries. Let us denote the endpoints of this final segment as q^+ (upper) and q^- (lower).

Recall our assumption that cells are labeled by SplitReduce as "inside" or "outside" only if they lie entirely inside or outside K_Q, respectively. It follows that as we traverse the cells that intersect the segment q^+q^- from top to bottom, we cannot transition directly from an "inside" cell to an "outside" cell. Therefore, at least one of these cells must contain a set of representative hyperplanes. Let $h(r)$ denote the hyperplane having the topmost intersection with the vertical ray. We return r as the approximate nearest neighbor (see Figure 16). It is easy to see that this algorithm satisfies the desired time and space bounds.

All that remains is to establish correctness, by showing that r is indeed an ε-approximate nearest neighbor of q. In order to do this, let p be q's true nearest neighbor in R_Q. Due to the nature of the binary search, q^+ lies within distance ε' of K_Q. (Note that it might lie within K_Q.) Thus, the distance from q^+ to the upper halfspace bounded by $h(p)$ is at most ε'. By the triangle inequality, the distance from q^- to this halfspace is at most $\varepsilon' + \varepsilon' = 2\varepsilon'$. Since p is q's true nearest neighbor, $q_{[p]}$ lies on ∂K_Q, and so the hyperplane $h(p)$ separates q^- from K_Q. This implies that the distance from q^- to $h(p)$ is also not greater than $2\varepsilon'$.

We claim that the vertical distance from q^- to $q_{[p]}$ is at most ε. To see why, recall that p lies within a ball of radius $2c_q$ centered at the origin. This implies that $h(p)$ cannot be too steep, that is, the angle formed between $h(p)$'s normal vector and the vertical axis can be bounded away from $\pi/2$ by a constant. By basic linear algebra, it can be shown that the ratio of the vertical and orthogonal distances of any point to $h(p)$ is bounded above by $\sqrt{4c_q^2+1} < 3c_q$. Therefore, we have $\|q_{[p]}q^-\| \leq 3\,c_q\,(2\varepsilon') = \varepsilon$, as desired.

Because r is the witness produced by the algorithm, $h(r)$ separates q^- from K_Q, which implies that $q_{[r]}$ lies above q^-. Thus, we have $\|q_{[p]}q_{[r]}\| \leq \|q_{[p]}q^-\| \leq \varepsilon$. Therefore,

$$\|qr\|^2 \;=\; \|q^\uparrow q_{[r]}\| \;=\; \|q^\uparrow q_{[p]}\| + \|q_{[p]}q_{[r]}\| \;\leq\; \|q^\uparrow q_{[p]}\| + \varepsilon.$$

By the lifting transformation, we have $\|q^\uparrow q_{[p]}\| = \|qp\|^2$, and combining this with the fact that $\|qp\| \geq 1$, we have

$$\|qr\|^2 \;\leq\; \|qp\|^2 + \varepsilon \;\leq\; \|qp\|^2 + \|qp\|^2\varepsilon \;=\; \|qp\|^2(1+\varepsilon) \;\leq\; \left(\|qp\|(1+\varepsilon)\right)^2.$$

Therefore, r is an ε-approximate nearest neighbor of p, which completes the proof. ∎

The above lemma shows how to apply approximate polytope membership to efficiently answer approximate nearest neighbor queries within each cell of the AVD. To obtain a complete data structure for approximate nearest neighbor searching we apply this to every leaf cell of the AVD.

LEMMA 9.3. *Let $0 < \varepsilon \leq 1/2$ be a real parameter and X be a set of n points in $\mathbb{R}^d$. Given a data structure for approximate polytope membership in d-dimensional space with query time at most $t_d(\varepsilon)$ and storage $s_d(\varepsilon)$, it is possible to preprocess X into an approximate nearest neighbor searching data structure with query time $O(\log n + t_{d+1}(\varepsilon) \cdot \log \frac{1}{\varepsilon})$ and space*

$$O\left(n \log \frac{1}{\varepsilon} + n \frac{s_{d+1}(\varepsilon)}{t_{d+1}(\varepsilon)}\right).$$

Proof. Following Lemma 9.1, construct a BBD-tree T, and for each leaf cell Q of T, construct the set of representative points R_Q. For each leaf cell such that $|R_Q| \leq t_{d+1}(\varepsilon) \cdot \lg \frac{1}{\varepsilon}$, simply store the set R_Q and answer the corresponding queries by brute force. For the nodes with $|R_Q| > t_{d+1}(\varepsilon) \cdot \lg \frac{1}{\varepsilon}$, use the construction from Lemma 9.2.

To answer an approximate nearest neighbor query we search the AVD of Lemma 9.1 to find the leaf cell containing the query point and then apply Lemma 9.2. Thus, the query time is

$$O\left(\log n + \log\log \frac{1}{\varepsilon} + t_{d+1}(\varepsilon) \cdot \log \frac{1}{\varepsilon}\right) \;=\; O\left(\log n + t_{d+1}(\varepsilon) \cdot \log \frac{1}{\varepsilon}\right).$$

To bound the total space, observe from Lemma 9.1(iii) that the total number of representative points is $O(n \log \frac{1}{\varepsilon})$. Thus, by a simple counting argument, the number of leaf cells with more than $t_{d+1}(\varepsilon) \cdot \lg \frac{1}{\varepsilon}$ representatives is $O(n/t_{d+1}(\varepsilon))$. Therefore, the total space of the data structure is $O(n \log \frac{1}{\varepsilon} + n(s_{d+1}(\varepsilon)/t_{d+1}(\varepsilon)))$. ∎

Because of its reliance on binary search, the generic reduction given in Lemmas 9.2 and 9.3 is not formally in the AVD model. Recall that the AVD model is important because lower bounds have been established in this model [10], and thus these lower bounds do not apply here. However, by sacrificing generality and a factor of $O(\log \frac{1}{\varepsilon})$ in the space bound, we can exploit the properties of SplitReduce to obtain a data structure that is in the AVD model.

LEMMA 9.4. *Let $0 < \varepsilon \leq 1/2$ be a real parameter and X be a set of n points in $\mathbb{R}^d$. Given a split-reduce data structure for approximate polytope membership in*

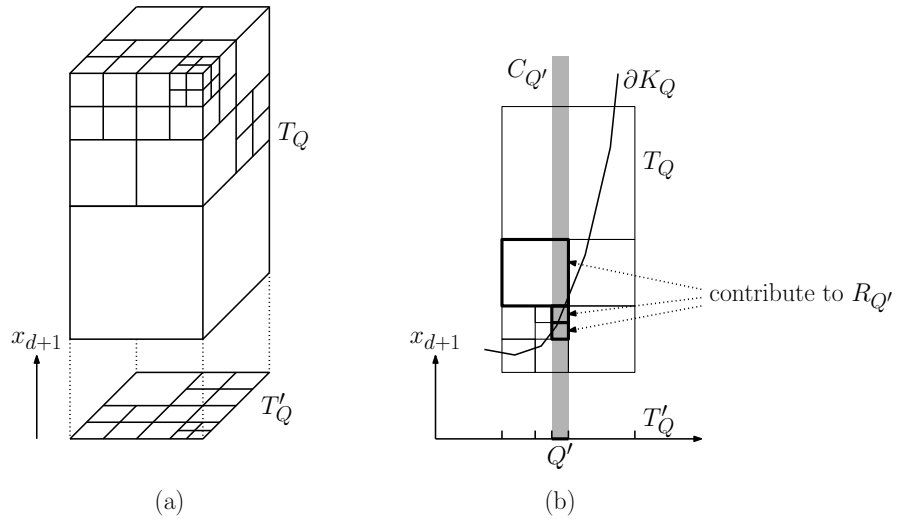


FIG. 17. *Producing an approximate nearest neighbor data structure in the AVD model.*

d-dimensional space with query time at most $t_d(\varepsilon)$ and storage $s_d(\varepsilon)$, it is possible to preprocess X into an approximate nearest neighbor data structure in the AVD model with query time $O(\log n + t_{d+1}(\varepsilon) \cdot \log \frac{1}{\varepsilon})$ and space

$$O\left(n\left(1 + \frac{s_{d+1}(\varepsilon)}{t_{d+1}(\varepsilon)}\right)\log\frac{1}{\varepsilon}\right).$$

Proof. As in Lemma 9.3, construct a BBD-tree T, and for each leaf cell Q of T, construct the set of representative points R_Q. We may assume that $|R_Q| > t_{d+1}(\varepsilon) \cdot \lg \frac{1}{\varepsilon}$, since otherwise we just use the points of R_Q as the representatives. In order to handle query points lying within Q, we apply Lemma 9.2, where queries are answered using the tree produced by SplitReduce. Let T_Q denote the resulting tree. We exploit the fact that the SplitReduce data structure associates a collection of hyperplanes with each leaf cell of T_Q, and by the nature of our reduction, each of these hyperplanes corresponds to a lifted point of R_Q. These lifted points will play the role of nearest neighbor representatives. Intuitively, our approach is to "undo" the lifting transformation by projecting the leaf cells of T_Q vertically from $\mathbb{R}^{d+1}$ down to $\mathbb{R}^d$ and then building a d-dimensional AVD structure based on this projection.

The projection of the cells of T_Q onto $\mathbb{R}^d$ naturally defines a quadtree subdivision of $\mathbb{R}^d$, which we denote by T'_Q (see Figure 17(a)). For each leaf cell Q' of T'_Q, let $C_{Q'}$ denote the infinite vertical cylinder in $\mathbb{R}^{d+1}$ whose cross section is Q' (see Figure 17(b)). Because Q' is a leaf, any leaf cell of T_Q that intersects this cylinder projects onto a hypercube that contains Q'.

Recall the lifted polytope K_Q of Lemma 9.2. For each leaf cell of T_Q that contains a point whose vertical distance from ∂K_Q is at most ε, we create a representative point corresponding to each of the hyperplanes that SplitReduce associates with this leaf cell. We denote the resulting collection of representatives by $R_{Q'}$. These are the only hyperplanes that are relevant to the binary search of Lemma 9.2, and therefore one of them will provide the final witness in the binary search (the point r in the proof of Lemma 9.2). This implies that $R_{Q'}$ constitutes a valid representative set for

ε-approximate nearest neighbor searching for any query point that lies in Q'. Thus, the resulting data structure is a valid AVD structure.

In order to bound the query time we recall some of the observations made in the proof of Lemma 9.2. Since K_Q is contained within a hypercube of constant diameter centered at the origin, the absolute slopes of the hyperplanes of the approximating polytope are bounded above by some constant. Recall that the leaf cells of T_Q that contribute a point to $R_{Q'}$ have side lengths at least as large as that of Q'. By the same reasoning used in Lemma 3 of [11], the number of such quadtree leaf cells that can intersect ∂K_Q is bounded by a constant, which we denote by c_ℓ. (This constant depends on the dimension d and the largest possible slope.) Therefore, the total number of cells contributing a representative to $R_{Q'}$ is at most c_ℓ. Since each cell contributes at most $t_{d+1}(\varepsilon)$ representatives, the total number of representatives associated with any leaf cell of T'_Q is at most $c_\ell \cdot t_{d+1}(\varepsilon) = O(t_{d+1}(\varepsilon))$.

The bound on the total space is complicated by the fact that a large cell that intersects ∂K_Q may overlap the columns of many small leaf cells, and hence a large cell's representatives may be replicated many times. Let M denote the set of internal nodes of T_Q all of whose children are leaves. We encountered this set earlier in the proof of Lemma 3.1. As we saw in that earlier lemma, because each node of M was split by SplitReduce, it follows that each such cell requires more than $t_{d+1}(\varepsilon)$ halfspaces to approximate $K(Q)$, and thus, the children of M together require at least as many representatives. Therefore we have $|M| \cdot t_{d+1}(\varepsilon) \leq s_{d+1}(\varepsilon)$. Reasoning as we did in Lemma 3.1, every internal node of T_Q either is in M or is an ancestor of a node in M. Thus, the number of internal nodes is at most $|M| \cdot \text{height}(T_Q)$. Since every internal node has 2^d children, the total number of nodes in T_Q is at most $2^d \cdot |M| \cdot \text{height}(T_Q)$. Clearly, the number of leaf cells of T'_Q can be no larger. As we saw in the previous paragraph, each leaf cell of T'_Q is associated with at most $c_\ell \cdot t_{d+1}(\varepsilon)$ representatives. Since the tree is of height $O(\log \frac{1}{\varepsilon})$, the total number of representatives over all these cells is at most

$$\begin{aligned}(2^d \cdot |M| \cdot \text{height}(T_Q))(c_\ell \cdot t_{d+1}(\varepsilon)) &= c_\ell \cdot 2^d \cdot \text{height}(T_Q) \cdot (|M| \cdot t_{d+1}(\varepsilon)) \\ &\leq \left(c_\ell \cdot 2^d \cdot \log \frac{1}{\varepsilon}\right) \cdot s_{d+1}(\varepsilon) \\ &= O\left(s_{d+1}(\varepsilon) \cdot \log \frac{1}{\varepsilon}\right).\end{aligned}$$

By Lemma 9.1(iii), the total number of representatives in T_Q is $O(n \log \frac{1}{\varepsilon})$. By a counting argument, the number of leaf cells with more than $t_{d+1}(\varepsilon) \cdot \log \frac{1}{\varepsilon}$ representatives is $O(n/t_{d+1}(\varepsilon))$. Therefore, the total space is

$$O\left(n \log \frac{1}{\varepsilon} + \frac{n}{t_{d+1}(\varepsilon)} \cdot s_{d+1}(\varepsilon) \cdot \log \frac{1}{\varepsilon}\right) \;=\; O\left(n\left(1 + \frac{s_{d+1}(\varepsilon)}{t_{d+1}(\varepsilon)}\right) \log \frac{1}{\varepsilon}\right)$$

as desired. ∎

By combining this with Theorem 1.2 (applying the more accurate space bounds from Lemma 6.4) we obtain the main result of this section.

LEMMA 9.5. *Let $0 < \varepsilon \leq 1$ be a real parameter, $\alpha \geq 1$ be a real constant, and X be a set of n points in $\mathbb{R}^d$. There is a data structure in the AVD model for approximate nearest neighbor searching that achieves*

$$\textit{Query time: } O\left(\log n + (1/\varepsilon^{d/2\alpha}) \cdot \log^2 \frac{1}{\varepsilon}\right),$$

$$\textit{Space: } O\left(n \cdot \max\left(\log \frac{1}{\varepsilon}, 1/\varepsilon^{d\left(\frac{1}{2}-\frac{1}{2\alpha}\right)}\right)\right) \quad \textit{for } 1 \le \alpha < 2, \textit{ and}$$

$$O\left(n/\varepsilon^{d\left(1-\frac{\lfloor \lg \alpha \rfloor}{\alpha}-\frac{1}{2^{\lfloor \lg \alpha \rfloor}}+\frac{1}{2\alpha}\right)}\right) \quad \textit{for } \alpha \ge 2.$$

The constant factors in the space and query time depend only on d and α (not on ε). At the expense of increasing the query time and space by a factor of $O(\log \frac{1}{\varepsilon})$ it is possible to construct the data structure in time $O(n(\log n + 1/\varepsilon^{cd}) \log \frac{1}{\varepsilon})$ for some constant c (that does not depend on d or α).

Proof. Given X and ε, we first observe that if $1/16 < \varepsilon \le 1$, we may set $\varepsilon = 1/16$, since this will only affect the constant factors in the asymptotic bounds. We consider two cases based on the value of α.

If $1 \le \alpha < 2$, we will apply Theorem 1.2 with the values of d and α of the theorem set to $d' = d + 1$ and $\alpha' = 4$, respectively. The theorem states that there is a data structure that achieves query time

$$O\left(\left(\log \frac{1}{\varepsilon}\right)/\varepsilon^{\frac{d'-1}{\alpha'}}\right) = O\left((1/\varepsilon^{d/4}) \cdot \log \frac{1}{\varepsilon}\right) = O\left((1/\varepsilon^{d/2\alpha}) \cdot \log \frac{1}{\varepsilon}\right) \tag{9}$$

and space

$$O\left(1/\varepsilon^{(d'-1)\left(1-\frac{2\lfloor \lg \alpha' \rfloor - 2}{\alpha'}\right)}\right) = O\big(1/\varepsilon^{d/2}\big). \tag{10}$$

Letting $t_{d+1}(\varepsilon)$ and $s_{d+1}(\varepsilon)$ denote the quantities of (9) and (10), respectively, we apply Lemma 9.4 to obtain a data structure in the AVD model with query time $O(\log n + (1/\varepsilon^{d/2\alpha}) \cdot \log^2 \frac{1}{\varepsilon})$ and space

$$O\left(n\left(1 + \frac{1/\varepsilon^{d/2}}{(1/\varepsilon^{d/2\alpha}) \cdot \log \frac{1}{\varepsilon}}\right) \log \frac{1}{\varepsilon}\right) = O\left(n \cdot \max\left(\log \frac{1}{\varepsilon}, 1/\varepsilon^{d\left(\frac{1}{2}-\frac{1}{2\alpha}\right)}\right)\right),$$

as desired.

Otherwise, if $\alpha \ge 2$, we apply Theorem 1.2 (but using the more accurate space bounds from Lemma 6.4) in dimension $d' = d + 1$ and with trade-off parameter $\alpha' = 2\alpha$. (Observe that $\alpha' \ge 4$, as required by Theorem 1.2 and Lemma 6.4.) This yields an approximate polytope membership data structure with query time $t_{d+1}(\varepsilon) = O((1/\varepsilon^{d/2\alpha}) \cdot \log \frac{1}{\varepsilon})$ and space

$$s_{d+1}(\varepsilon) = O\left(1/\varepsilon^{d\left(1-2\left(\frac{\lfloor \lg(2\alpha) \rfloor - 2}{2\alpha}+\frac{1}{2^{\lfloor \lg(2\alpha) \rfloor}}\right)\right)}\right) = O\left(1/\varepsilon^{d\left(1-\frac{\lfloor \lg \alpha \rfloor - 1}{\alpha}-\frac{1}{2^{\lfloor \lg \alpha \rfloor}}\right)}\right).$$

By Lemma 9.4 this implies the existence of a data structure in the AVD model with the desired query time of $O(\log n + (1/\varepsilon^{d/2\alpha}) \cdot \log^2 \frac{1}{\varepsilon})$ and space

$$O\left(n\left(1 + \frac{1/\varepsilon^{d\left(1-\frac{\lfloor \lg \alpha \rfloor - 1}{\alpha}-\frac{1}{2^{\lfloor \lg \alpha \rfloor}}\right)}}{(\log \frac{1}{\varepsilon})/\varepsilon^{d/2\alpha}}\right) \log \frac{1}{\varepsilon}\right).$$

Since $\alpha \geq 2$, we may ignore the "1+" term in the inner parenthetical factor. After some simplification we obtain the desired space bound of

$$O\left(n/\varepsilon^{d\left(1-\frac{\lfloor \lg \alpha \rfloor}{\alpha}-\frac{1}{2^{\lfloor \lg \alpha \rfloor}}+\frac{1}{2\alpha}\right)}\right).$$

The preprocessing involves first computing the AVD, which by Lemma 9.1 takes $O(n \cdot \log n \cdot \log \frac{1}{\varepsilon})$ time. For each of the $O(n \log \frac{1}{\varepsilon})$ leaf cells Q of the AVD, we apply SplitReduce in dimension $d+1$ to its associated set R_Q of representatives. By Lemma 7.6 this takes $O(n_Q + 1/\varepsilon^{c_p(d+1)})$ time, where $n_Q = |R_Q|$, and c_p is a constant that does not depend on d. Summing over all the leaf cells of the AVD and recalling that the total number of representatives is $O(n \cdot \log \frac{1}{\varepsilon})$, it follows that the total preprocessing time is on the order of

$$\begin{aligned} n \cdot \log n \cdot \log \frac{1}{\varepsilon} \;+\; \sum_Q \left(n_Q + 1/\varepsilon^{c_p(d+1)}\right) &= n \cdot \log n \cdot \log \frac{1}{\varepsilon} + n \cdot \log \frac{1}{\varepsilon} \cdot \left(\frac{1}{\varepsilon}\right)^{c_p(d+1)} \\ &= n \left(\log n + \left(\frac{1}{\varepsilon}\right)^{c\,d}\right) \log \frac{1}{\varepsilon}, \end{aligned}$$

where $c = c_p(d+1)/d$, as desired. Because of the reliance on approximate set cover in the processing of Lemma 7.6, the query time and space are larger by a factor of $O(\log \frac{1}{\varepsilon})$. □

Note that the above proof uses the AVD-based reduction given in Lemma 9.4. If instead we had used Lemma 9.3, we would obtain a slight improvement in the space, by a factor of $\Theta(\log \frac{1}{\varepsilon})$, at the loss of having a data structure in the AVD model. By the simple observation that $1/2^{\lfloor \lg \alpha \rfloor} \geq 1/\alpha$, the above space bound for the $\alpha \geq 2$ case simplifies to $O(n/\varepsilon^{d(1-\frac{\lfloor \lg \alpha \rfloor}{\alpha}-\frac{1}{2\alpha})})$, and this establishes Theorem 1.4.

10. Proof of the area-product bound. In this section, we present lower bounds for the product of the area of (restricted) ε-dual caps and the associated Voronoi patches, and in particular, we present a proof of Lemma 5.2, which appeared at the end of section 5.

We begin by recalling some notation. We are given a convex body K in $\mathbb{R}^d$ and a pair $(p, h(p))$, where $p \in \partial K$ and $h(p)$ is a supporting hyperplane passing through p, such that p lies within a unit ball centered at the origin. Also recall that p_ε denotes the point lying at distance ε from p in the direction of the outward normal orthogonal to $h(p)$ at p. S denotes the Dudley hypersphere, which is centered at the origin and is of radius 3. For $y \geq 1$, let $H^{(y)}(p)$ be any hyperplane that is parallel to $h(p)$ and translated away from K by distance y. (This is illustrated in Figure 18. Note that the figures of this section are not drawn to scale.) To simplify our descriptions, we consider the directed line segment from p to p_ε to be "vertically downward," so that the hyperplanes $h(p)$ and $H^{(y)}(p)$ are "horizontal" with $h(p)$ above $H^{(y)}(p)$.

Recall that the *ε-dual cap* defined by p, denoted $D(p)$, is the portion of ∂K that is visible from p_ε (see Figure 19(a)). Also, recall that $\mathrm{Vor}(D(p))$ consists of the points that are exterior to K whose closest point on ∂K lies within $D(p)$. Define the *base* of $D(p)$, denoted $\Gamma(p)$, to be the intersection of $h(p)$ with the convex hull of $K \cup \{p_\varepsilon\}$.

For $\delta > 0$, recall that the *δ-restricted ε-dual cap* defined by p, denoted $D_\delta(p)$, is $D(p) \cap B_\delta(p)$, where $B_\delta(p)$ is the Euclidean ball of radius δ centered at p

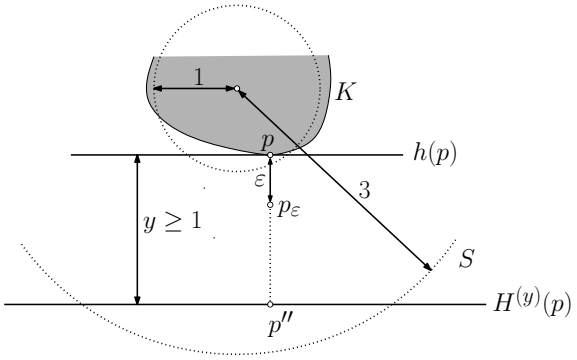


FIG. 18. *Definitions of* $h(p)$, $H^{(y)}(p)$, *and* S.

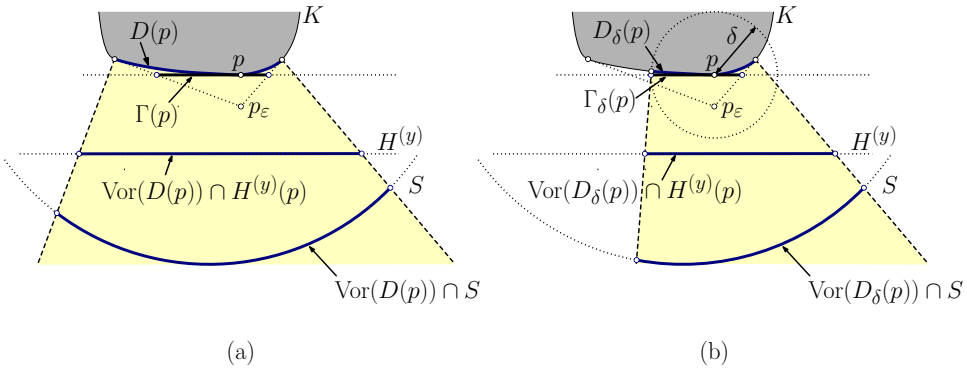


FIG. 19. *Dual caps, bases, and Voronoi regions for the* (a) *unrestricted and* (b) *restricted cases.*

(see Figure 19(b)). As before, $\mathrm{Vor}(D_\delta(p))$ is the set of points that are exterior to K whose closest point on ∂K lies within $D_\delta(p)$. Also, the *δ-restricted base*, denoted $\Gamma_\delta(p)$ is $\Gamma(p) \cap B_\delta(p)$.

Our objective in this section is to establish bounds on the product of the area of a $\sqrt{\varepsilon}$-restricted ε-dual cap and its Voronoi patch on the Dudley hypersphere. It will be easier to start with hyperplane patches on $H^{(y)}(p)$ and then generalize to spherical patches on S. The main result of this section is given in the following lemma. Part (ii) is equivalent to Lemma 5.2, which is our main objective. Part (i) is a useful intermediate result.

LEMMA 10.1. *Let K be a convex body in $\mathbb{R}^d$, and let $0 < \varepsilon \leq 1/8$ and $\delta = \sqrt{\varepsilon}$. There are constants c_a and c'_a (depending only on d) such that for any point $p \in \partial K$,*

(i) *given any $y \geq 1$,* $\mathrm{area}(D_\delta(p)) \cdot \mathrm{area}(\mathrm{Vor}(D_\delta(p)) \cap H^{(y)}(p)) \geq c'_a \cdot \varepsilon^{d-1}$;

(ii) *if K is fat and has diameter at least 2ε, and p lies within a unit ball centered at the origin, then* $\mathrm{area}(D_\delta(p)) \cdot \mathrm{area}(\mathrm{Vor}(D_\delta(p)) \cap S) \geq c_a \cdot \varepsilon^{d-1}$.

This lemma holds generally for any $\delta \geq \sqrt{\varepsilon}$, but it suffices for our purposes to consider the restricted case of $\delta = \sqrt{\varepsilon}$. Note that the additional assumptions on fatness and diameter of part (ii) are necessary for establishing a lower bound. If K is not fat or not of sufficiently large diameter, then $\mathrm{area}(D_\delta(p))$ can be arbitrarily small. Since the Dudley hypersphere is bounded, it would not be possible to establish any lower bound on the product of their areas.

The remainder of this section is devoted to proving this lemma. Because p will be fixed throughout, in order to simplify the notation, we will drop references to p. For example, we will use h, $H^{(y)}$, D_δ, Γ_δ, and B_δ in place of $h(p)$, $H^{(y)}(p)$, $D_\delta(p)$, $\Gamma_\delta(p)$, and $B_\delta(p)$, respectively.

Since it will be useful to relate sets on h with sets on $H^{(y)}$, we observe that each of these hyperplanes can be consistently identified with $\mathbb{R}^{d-1}$ by endowing them with parallel coordinate frames, one centered at p (for h) and one centered at p's orthogonal projection onto $H^{(y)}$. Thus, a point on h and its vertical projection onto $H^{(y)}$ have the same coordinates.

We start by proving Lemma 10.1(i). Since the value of y will be fixed throughout this part of the proof, we refer to $H^{(y)}$ simply as H. Let p'' denote the origin of H's coordinate system (the vertical projection of p onto H). (See Figure 18.) In order to exploit Lemma 2.2 on the Mahler volume, rather than considering $\mathrm{Vor}(D_\delta) \cap H$ directly, we will find it convenient to instead analyze the polar dual of the base Γ_δ. Using the aforementioned coordinate frame, we can think of Γ_δ as a body in $\mathbb{R}^{d-1}$. For $r = \sqrt{\varepsilon/8}$, consider the generalized polar of the dual base, $\mathrm{polar}_r(\Gamma_\delta)$, which we can think of as a convex subset of H. Because Γ_δ contains the origin of h (namely, p), it follows directly that $\mathrm{polar}_r(\Gamma_\delta)$ is bounded and convex and also contains the origin of H (namely, p''). In order to obtain a lower bound on $\mathrm{area}(\mathrm{Vor}(D_\delta) \cap H)$, we will first show that $\mathrm{polar}_r(\Gamma_\delta)$ is a subset of $\mathrm{Vor}(D_\delta) \cap H$ and then derive a lower bound on $\mathrm{area}(\mathrm{polar}_r(\Gamma_\delta))$. The first assertion is established by the following lemma.

LEMMA 10.2. *Given the preconditions of Lemma* 10.1 *and* $r = \sqrt{\varepsilon/8}$*, we have* $\mathrm{polar}_r(\Gamma_\delta) \subseteq \mathrm{Vor}(D_\delta) \cap H$.

The proof is rather technical and involves a reduction to the problem in two-dimensional space. Before giving the proof, it will help to provide some intuition regarding the relationship between $\mathrm{Vor}(D_\delta) \cap H$ and the polar of Γ_δ.

For the sake of simplicity, let us consider just the two-dimensional setting. Let t denote a point of tangency on ∂K with respect to p_ε (see Figure 20), and let v be the intersection of the line segment $p_\varepsilon t$ with h. Shoot a ray from t perpendicular to ∂K until it intersects H. Let q denote this intersection point. Since K is convex, all the points on the segment $p''q$ have their nearest neighbor on the portion of ∂K between p and t, that is, they all lie within $\mathrm{Vor}(D)$. Observe that if we translate this perpendicular line so that it emanates from p_ε instead of t, it will hit H at a point q' that is closer to p''. Therefore, the segment $p''q'$ also lies within $\mathrm{Vor}(D)$. Let ℓ denote the distance between p_ε and p''. By similar triangles, it is easy to see that the length of $p''q'$ is $\ell \cdot \varepsilon / \|pv\|$. Since $v \in \Gamma$, q' lies within $\mathrm{polar}_{r'}(\Gamma)$, where $r' = \sqrt{\ell \cdot \varepsilon}$.

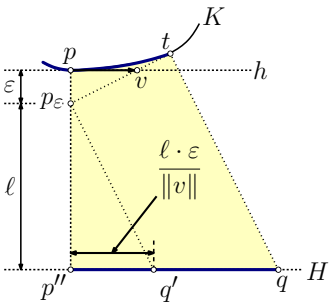


FIG. 20. *The relationship between* $\mathrm{Vor}(D_\delta) \cap H$ *and the polar of* Γ_δ.

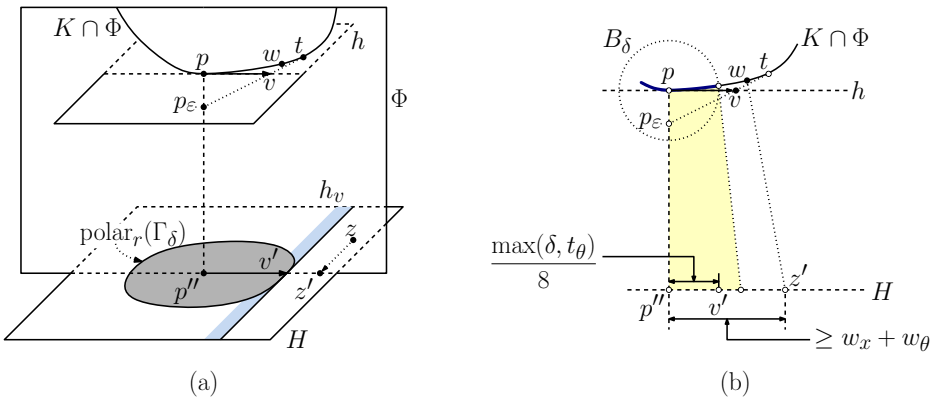


FIG. 21. *The reduction to the plane* Φ.

Because $y \geq 1$ and $\varepsilon \leq 1/8$, we have $r' = \Omega(\sqrt{\varepsilon})$. This observation generalizes readily to higher dimensions, and it follows that $\text{polar}_{r'}(\Gamma) \subseteq \text{Vor}(D) \cap H$. We will show how to generalize this intuition to higher dimensions and the δ-restricted setting.

For any $z \in H$ let w denote its nearest neighbor on ∂K. In order to prove Lemma 10.2, it suffices to show that if $w \notin D_\delta$ (implying that $z \notin \text{Vor}(D_\delta) \cap H$), then $z \notin \text{polar}_r(\Gamma_\delta)$. By our assumption that H lies below K it follows that w lies on the "lower surface" of ∂K (meaning that a vertical ray directed downward from w does not intersect the interior of K). Since w is not in the restricted cap, we know that either $w \notin D$ or $w \notin B_\delta$.

It will simplify the analysis to reduce the problem to a two-dimensional setting. Consider the plane Φ that contains the points p, p_ε, and w. (Note that these points are not collinear.) Let t be the point of tangency on $\partial K \cap \Phi$ with respect to p_ε that lies on the same side as w (see Figure 21(a)). Let v be the intersection of the line segment $p_\varepsilon t$ with h. We may identify Φ with $\mathbb{R}^2$ by imposing a coordinate system on Φ where the origin is at p, the y-axis is directed upward away from p_ε, and the x-axis is parallel to the vector from p to v. Given a point $u \in \Phi$, let u_x and u_y denote its coordinates relative to this coordinate system. Further, if $u \in \partial K \cap \Phi$, let u_θ denote the slope of the (unique) supporting line on Φ passing through u. Note that z need not lie on Φ. Let z' be the orthogonal projection of z onto Φ. Observe that $t_\theta = \varepsilon / \|pv\|$, and therefore $\|pv\| = \varepsilon / t_\theta$. By our choice of coordinate system and assumptions about orientations, the coordinates of w, t and the slopes w_θ and t_θ are all nonnegative quantities.

The point v lies on the base Γ of p's unrestricted dual cap. By employing our coordinate system on h, we can identify v with a vector in $\mathbb{R}^{d-1}$ (emanating from p). If $\|pv\| \leq \delta$, then v contributes a bounding halfspace to $\text{polar}_r(\Gamma_\delta)$. This halfspace is bounded by a hyperplane that is orthogonal to v and lies at distance $r^2/\|pv\|$ from the origin. Let us think of this halfspace, which we denote by h_v, as lying on H (see Figure 21(a)). Recalling that $r = \sqrt{\varepsilon/8}$, the distance of h_v's bounding hyperplane to the origin p'' is $(\varepsilon/8)/\|pv\| = t_\theta/8$. On the other hand, if $\|pv\| > \delta$, then v lies outside the restricted base. In this case v's subvector of length δ lies on the boundary of the restricted base and contributes to $\text{polar}_r(\Gamma_\delta)$ a halfspace whose bounding hyperplane is at distance $(\varepsilon/8)/\delta$ from the origin. Recalling that $\delta = \sqrt{\varepsilon}$, this is equal to $\delta/8$. Thus, in either case, $\text{polar}_r(\Gamma_\delta)$ is bounded by a halfspace whose defining hyperplane is orthogonal to v and lies at distance $\max(\delta, t_\theta)/8$ from the origin. This hyperplane

intersects the horizontal line $H \cap \Phi$ at some point v' that lies to the right of p'' at distance $v'_x = \max(\delta, t_\theta)/8$ (see Figure 21(b)).

Because the hyperplane passing through v' is orthogonal to v, in order to show that $z \notin \text{polar}_r(\Gamma_\delta)$, it suffices to show that z' does not lie within h_v, which is equivalent to showing that $z'_x > v'_x$. We have thus reduced the problem to a two-dimensional setting.

Recall that w is the closest point to z on ∂K. We assert that w is also the closest point to z' on $\partial K \cap \Phi$. The reason is that the squared distance from z to any point on $\partial K \cap \Phi$ can be expressed as the sum of the squared distance from z' to this point and the squared distance from z to z'. Since the latter quantity is the same for all points on Φ, the closest point to z is also the closest point to z'. From basic properties of convexity, it follows that the line wz' is orthogonal to the support line passing through w on $\partial K \cap \Phi$. Therefore, the slope of wz' (in Φ's coordinate system) is $-1/w_\theta$, and in particular we have $(z'_x - w_x)/(z'_y - w_y) = -w_\theta$. Since h and H are separated by at least unit distance (with h above H), we have $w_y - z'_y \geq 1$, and so $z'_x \geq w_x + w_\theta$.

Thus, to complete the proof of Lemma 10.2, it suffices to show that if $w \notin D_\delta$, then $v'_x < w_x + w_\theta$. We first establish two useful technical results. These results will be applied in a context where w lies within the unrestricted dual cap but outside the restricted dual cap, that is, when $w_x \leq t_x$ but $w \notin B_\delta$. The first result shows that if t_θ is sufficiently small, the slope of the line pw is at most unity. The second shows that if t_θ is sufficiently large, the slope of pw is not much smaller than the slope of t's supporting line.

LEMMA 10.3. *Given the preconditions of Lemma* 10.1 *and the aforementioned two-dimensional reduction, and given w and t as introduced above, where $w_x \leq t_x$ and $w \notin B_\delta$,*

(i) *if $t_\theta \leq \delta\sqrt{8}$, then $w_y/w_x \leq 1$, and*
(ii) *if $t_\theta > \delta\sqrt{8}$, then $w_y/w_x \geq t_\theta/2$.*

The proof is a straightforward geometric exercise and has been omitted. (See [8] for the full proof.)

We are now in position to complete the proof of Lemma 10.2. Recall that our objective is to show that if $w \notin D_\delta$, then $v'_x < w_x + w_\theta$, where $v'_x = \max(\delta, t_\theta)/8$. We consider two cases, depending on t_θ. First, if $t_\theta \leq \delta\sqrt{8}$, then $v'_x \leq \max(\delta, \delta\sqrt{8})/8 = \delta/\sqrt{8}$. Since the line $p_\varepsilon t$ has slope t_θ and $t_y \geq 0$, we have $t_x = (t_y + \varepsilon)/t_\theta \geq \varepsilon/t_\theta \geq \delta/\sqrt{8}$. We consider two subcases. If $w_x > t_x$, then we have $w_x + w_\theta > t_x \geq \delta/\sqrt{8} \geq v'_x$, as desired. On the other hand, if $w_x \leq t_x$, then w is inside the unrestricted cap D. Since by our hypothesis, w is not in the restricted cap, it must be that $w \notin B_\delta$, that is, $w_x^2 + w_y^2 > \delta^2$. By Lemma 10.3(i), we have $w_x \geq w_y$. Therefore, $2w_x^2 \geq w_x^2 + w_y^2 > \delta^2$, which implies that $w_x > \delta/\sqrt{2}$. Therefore, $w_x + w_\theta \geq w_x > \delta/\sqrt{2} > v'_x$, as desired.

For the second case, assume that $t_\theta > \delta\sqrt{8}$. In this case $v'_x = t_\theta/8$. As before, we consider two subcases. If $w_x > t_x$, then by convexity $w_\theta \geq t_\theta$, and so $w_x + w_\theta \geq t_\theta > v'_x$, as desired. On the other hand, if $w_x \leq t_x$, then since w lies within the unrestricted cap, we may infer that $w \notin B_\delta$. By Lemma 10.3(ii), we have $w_y/w_x \geq t_\theta/2$. Because the support line at w passes below the origin, we also have $w_\theta \geq w_y/w_x$. Therefore $w_x + w_\theta \geq w_y/w_x \geq t_\theta/2 > v'_x$. This completes the proof of Lemma 10.2.

Because it is easier to deal with flat objects than curved ones, before returning to the proof of Lemma 10.1(i), we show that the area of the restricted dual cap is, up to a constant factor, bounded below by the area of its base. This result is straightforward for unrestricted caps, since it is easy to show that the base is contained within the orthogonal projection of the dual cap onto h. However, restriction complicates

the analysis. The proof involves a rather technical but straightforward geometric argument. We have omitted it, but it is given in full in [8].

LEMMA 10.4. *Given the preconditions of Lemma* 10.1*, it follows that* $\text{area}(D_\delta) \geq \text{area}(\Gamma_\delta)/2^{d-1}$.

We are now ready to prove Lemma 10.1(i). Recall that $r = \sqrt{\varepsilon/8}$. As observed earlier, $\text{polar}_r(\Gamma_\delta)$ is a scaled copy of $\text{polar}(\Gamma_\delta)$ by a factor of r^2, and therefore (since these are $(d-1)$-dimensional bodies) we have $\text{area}(\text{polar}_r(\Gamma_\delta)) = r^{2(d-1)} \cdot \text{area}(\text{polar}(\Gamma_\delta))$. By applying Lemma 10.2, we have

$$\text{area}(\text{Vor}(D_\delta) \cap H) \;\geq\; \text{area}(\text{polar}_r(\Gamma_\delta)) \;=\; r^{2(d-1)} \cdot \text{area}(\text{polar}(\Gamma_\delta)).$$

By Lemma 10.4, $\text{area}(D_\delta) \geq \text{area}(\Gamma_\delta)/2^{d-1}$, and therefore

$$\begin{aligned}\text{area}(D_\delta) \cdot \text{area}(\text{Vor}(D_\delta) \cap H) &\geq \frac{\text{area}(\Gamma_\delta)}{2^{d-1}} \cdot r^{2(d-1)} \cdot \text{area}(\text{polar}(\Gamma_\delta)) \\ &\geq \left(\frac{r^2}{2}\right)^{d-1} \text{area}(\Gamma_\delta) \cdot \text{area}(\text{polar}(\Gamma_\delta)).\end{aligned}$$

We now apply the Mahler-volume bound. By Lemma 2.2 (in $\mathbb{R}^{d-1}$), there exists a constant c_m (depending only on d) such that $\text{area}(\Gamma_\delta) \cdot \text{area}(\text{polar}(\Gamma_\delta)) \geq c_m$. Therefore,

$$\text{area}(D_\delta) \cdot \text{area}(\text{Vor}(D_\delta) \cap H) \;\geq\; c_m \left(\frac{r^2}{2}\right)^{d-1} \;=\; c_m \left(\frac{\varepsilon}{16}\right)^{d-1}.$$

Selecting any $c'_a \leq c_m/16^{d-1}$ establishes Lemma 10.1(i).

Next, let us establish Lemma 10.1(ii). Recall that we assume that K is fat and of diameter at least 2ε. In particular, let us assume that K is γ-fat, where γ is a constant independent of n and ε that lies in the interval $(0, 1]$. (As a result of Lemma 2.1, we may assume that γ is $1/d$ when applying this result.)

It is natural to try to generalize the approach used in part (i). First, we would show that

$$\text{area}(\text{Vor}(D_\delta) \cap S) = \Omega\big(r^{2(d-1)} \cdot \text{area}(\text{polar}(\Gamma_\delta))\big) \quad \text{and} \quad \text{area}(D_\delta) = \Omega(\text{area}(\Gamma_\delta)),$$

and then we would apply the Mahler-volume bound to yield a lower bound on the product $\text{area}(\Gamma_\delta) \cdot \text{area}(\text{polar}(\Gamma_\delta))$. A problem arises, however, if K is not smooth. In particular, if some portion of the boundary of K in p's vicinity is nearly vertical, then the boundary of Γ_δ can be arbitrarily close to the origin (namely, p), implying that $\text{polar}(\Gamma_\delta)$ cannot be bounded, and hence its area can be arbitrarily large. This was not an issue in part (i), because H is also unbounded. But since S is bounded, $\text{area}(\text{Vor}(D_\delta) \cap S)$ cannot be arbitrarily large. We will remedy this by smoothing K by taking its Minkowski sum with a small Euclidean ball of radius $O(\varepsilon)$. We shall see (in the proof of Lemma 10.7) that this allows us to constrain the area of $\text{polar}(\Gamma_\delta)$. This smoothing operation requires us to adapt many of the prior results of this section to this new context.

To construct the smoothed body, for the remainder of this section define $\varepsilon' = \varepsilon/2$, and let $K' = K \oplus \varepsilon'$ (see Figure 22(a)). Recall that h denotes the supporting hyperplane at p and p_ε is the point at distance ε from p in the direction orthogonal to h. As before, for the sake of illustration, let us assume that p_ε is vertically below p.

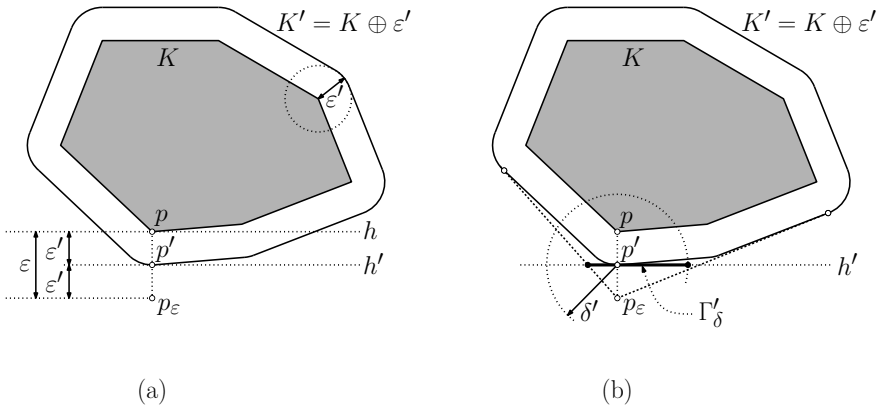


FIG. 22. *The smoothed body K'.*

Let p' be the midpoint of the segment pp_ε. Clearly, $p' \in \partial K'$, and the parallel hyperplane h' passing through p' is a supporting hyperplane for K'.

Let us also define the dual base in this smoothed context. Define Γ' to be the intersection of h' and $\text{conv}(K' \cup \{p_\varepsilon\})$. Let $\delta' = \sqrt{\varepsilon'} = \delta/\sqrt{2}$, and define the restricted base Γ'_δ to be the intersection of Γ' and a ball of radius δ' centered at p' (see Figure 22(b)). Our analysis will be based on K' and Γ'_δ, as opposed to K and Γ_δ. Our first objective will be to show that the area of Γ'_δ is not significantly larger than that of Γ_δ. As before, we endow h and h' with parallel coordinate frames whose origins are located at p and p', respectively. Then we can think of Γ_δ and Γ'_δ as convex sets in $\mathbb{R}^{d-1}$. The following lemma relates these two bodies.

LEMMA 10.5. *Given a convex body K that is γ-fat and of diameter at least 2ε and given Γ_δ and Γ'_δ as defined above, there exists a constant c (depending on γ and the dimension d) such that* $\text{area}(\Gamma'_\delta) \leq c \cdot \text{area}(\Gamma_\delta)$.

This result is not surprising, given that K is fat and $\|pp'\|$ is within a constant factor of $\|pp_\varepsilon\|$. The proof involves straightforward geometric reasoning, but (as always) restriction complicates the analysis. It is presented in full in [8].

Recall that $\text{Vor}(D_\delta) \cap S$ consists of the set of points on the sphere S whose closest point on ∂K lies within the restricted dual cap D_δ. Let D'_δ denote the corresponding restricted dual cap for K', that is, the set of points of $\partial K'$ that are visible from p_ε and lie within the ball $B_{\delta'}(p')$. Our analysis will be based on establishing a lower bound on the area of $\text{Vor}(D'_\delta) \cap S$. The following lemma shows that this will provide a lower bound on the area of $\text{Vor}(D_\delta) \cap S$.

LEMMA 10.6. *Given the preconditions of Lemma* 10.1(ii), $\text{area}(\text{Vor}(D'_\delta) \cap S) \leq \text{area}(\text{Vor}(D_\delta) \cap S)$.

Proof. We sketch the proof here, but complete details can be found in [8]. We prove the stronger result that $\text{Vor}(D'_\delta) \cap S \subseteq \text{Vor}(D_\delta) \cap S$. First observe that (by our restriction on ε and the definition of δ) both D_δ and D'_δ lie within the Dudley hypersphere S. Consider any point $q'' \in \text{Vor}(D'_\delta) \cap S$. It suffices to show that $q'' \in \text{Vor}(D_\delta) \cap S$. Let q and q' be the closest points on ∂K and $\partial K'$, respectively, to q''. It follows from basic properties of the Minkowski sum that these three points are collinear and there are supporting hyperplanes at q and q' that are orthogonal to this line. Since $q'' \in \text{Vor}(D'_\delta) \cap S$, we have $q' \in D'_\delta$, which implies that q' is visible

from p_ε. By the existence of parallel supporting hyperplanes, q is also visible from p_ε. Therefore, q lies in p's unrestricted dual cap D. By basic properties of convexity, $\|pq\| \leq \|p'q'\|$, which implies that q lies within p's restricted dual cap, and therefore $q'' \in \text{Vor}(D_\delta) \cap S$, as desired. □

Before completing the proof of Lemma 10.1(ii), we exploit the smoothness of K' to establish a relationship between the areas of $\text{Vor}(D'_\delta) \cap S$ and $\text{polar}_{r'}(\Gamma'_\delta)$, where the polar radius r' is suitably modified for the smoothed context. This is given in the next lemma.

LEMMA 10.7. *Given the preconditions of Lemma* 10.1(ii) *and* $r' = \sqrt{\varepsilon'/8}$, *we have* $\text{area}(\text{Vor}(D'_\delta) \cap S) \geq \text{area}(\text{polar}_{r'}(\Gamma'_\delta))$.

Proof. The proof involves a rather technical geometric analysis. We present a sketch here, but complete details can be found in [8]. First, we use the fact that since p is in K, there is a ball of radius $\varepsilon' = \varepsilon/2$ centered at p that lies within K'. It follows that Γ'_δ contains a $(d-1)$-dimensional Euclidean ball (centered at p') of radius $\varepsilon'/\sqrt{3}$.

Let H' denote the hyperplane that is at unit distance below p'. Let p'' denote the vertical projection of p' onto H'. By the definition of the polar transformation, $\text{polar}_{r'}(\Gamma'_\delta)$ (when viewed as a subset of H') is contained within a $(d-1)$-dimensional unit ball centered at p''. Let C denote the semi-infinite generalized cylinder whose horizontal cross section is $\text{polar}_{r'}(\Gamma'_\delta)$, whose upper surface lies on H', and which extends vertically downward. Lemma 10.2 (applied now to K', ε', $\text{polar}_{r'}(\Gamma'_\delta)$ and $\text{Vor}(D'_\delta) \cap H'$) implies that $\text{polar}_{r'}(\Gamma'_\delta) \subseteq \text{Vor}(D'_\delta) \cap H'$. Since this applies not only to H' but to any hyperplane lying below H', it follows that $C \subseteq \text{Vor}(D'_\delta)$. The remainder of the proof involves showing that S is large enough that the orthogonal projection of $S \cap C$ onto H' is equal to $\text{polar}_{r'}(\Gamma'_\delta)$. Since $S \cap C \subseteq \text{Vor}(D'_\delta) \cap S$, and since the area of the orthogonal projection of a set cannot be larger than the area of the original set, we have

$$\text{area}(\text{Vor}(D'_\delta) \cap S) \;\geq\; \text{area}(S \cap C) \;\geq\; \text{area}(\text{polar}_{r'}(\Gamma'_\delta)),$$

as desired. □

We are now ready to prove Lemma 10.1(ii). Recall that $r' = \sqrt{\varepsilon'/8}$. By Lemmas 10.6 and 10.7, we have

$$\text{area}(\text{Vor}(D_\delta) \cap S) \;\geq\; \text{area}(\text{Vor}(D'_\delta) \cap S) \;\geq\; \text{area}(\text{polar}_{r'}(\Gamma'_\delta)).$$

As observed earlier, $\text{polar}_{r'}(\Gamma'_\delta)$ is a scaled copy of $\text{polar}(\Gamma'_\delta)$ by a factor of $(r')^2 = \varepsilon'/8 = \varepsilon/16$, and therefore (since these are $(d-1)$-dimensional bodies) we have

$$\text{area}(\text{Vor}(D_\delta) \cap S) \;\geq\; \left(\frac{\varepsilon}{16}\right)^{d-1} \cdot \text{area}(\text{polar}(\Gamma'_\delta)).$$

By Lemma 10.4, $\text{area}(D_\delta) \geq \text{area}(\Gamma_\delta)/2^{d-1}$. Also, by Lemma 10.5 there is a constant c'' (depending on the fatness parameter γ and d) such that $\text{area}(\Gamma'_\delta) \leq c'' \cdot \text{area}(\Gamma_\delta)$. Therefore, we have

$$\text{area}(D_\delta) \;\geq\; \frac{\text{area}(\Gamma_\delta)}{2^{d-1}} \;\geq\; \frac{\text{area}(\Gamma'_\delta)}{c'' \cdot 2^{d-1}}.$$

Combining these, we obtain

$$\begin{aligned}\text{area}(D_\delta) \cdot \text{area}(\text{Vor}(D_\delta) \cap S) &\geq \frac{\text{area}(\Gamma'_\delta)}{c'' \cdot 2^{d-1}} \cdot \left(\frac{\varepsilon}{16}\right)^{d-1} \cdot \text{area}(\text{polar}(\Gamma'_\delta)) \\ &= \frac{1}{c''}\left(\frac{\varepsilon}{32}\right)^{d-1} \text{area}(\Gamma'_\delta) \cdot \text{area}(\text{polar}(\Gamma'_\delta)).\end{aligned}$$

By applying Lemma 2.2 (in $\mathbb{R}^{d-1}$) to Γ'_δ, there exists a constant c_m (depending on d) such that

$$\text{area}(\Gamma'_\delta) \cdot \text{area}(\text{polar}(\Gamma'_\delta)) \;\geq\; c_m.$$

Therefore, we have

$$\text{area}(D_\delta) \cdot \text{area}(\text{Vor}(D_\delta) \cap S) \;\geq\; \frac{c_m}{c''} \left(\frac{\varepsilon}{32}\right)^{d-1}.$$

Selecting any $c_a \leq (c_m/c'')(1/32)^{d-1}$ establishes Lemma 10.1(ii). This concludes our proof of the area bounds.

11. Concluding remarks. In this paper we have presented an efficient data structure for determining approximately whether a given query point lies within a convex body. Our solution is based on a simple and natural quadtree-based algorithm, called SplitReduce. Our principal technical contribution has been an analysis of the space-time trade-offs for this algorithm. These are the first nontrivial space-time trade-offs for this problem. We do not know whether this analysis is tight, but we presented a lower bound example that demonstrates the limits of possible improvements. We also demonstrated the value of approximate polytope membership by showing that our data structure can be combined with an AVD data structure to produce significant improvements to the space-time trade-offs of approximate nearest neighbor searching in Euclidean space.

Our analysis of the trade-offs involved a combination of a number of novel techniques, which may be of broader interest. One notable example is the application of the Mahler volume as a means of analyzing the local structure of a convex body through consideration of both its primal and dual representations. This resulted in an efficient two-pronged sampling strategy for computing hitting sets of low cardinality for ε-dual caps. The Mahler volume has also been applied in [7] to derive an optimal area-sensitive bound on the number of facets needed to approximate a convex body.

This work provokes a number of questions for further research. The first involves extending approximate polytope membership queries to other approximate query problems involving convex bodies. For example, in section 9 we showed how to reduce approximate nearest neighbor searching in dimension d to vertical ray shooting queries in dimension $d+1$. However, the polytope involved had a very restricted structure. It would be interesting to know whether there is a data structure exhibiting similar trade-offs for answering approximate ray-shooting queries for general convex bodies. Another example is answering approximate linear-programming queries, where a convex body is preprocessed, and the problem is to determine an extreme point of the body approximately in a given query direction. A further generalization of this would be to extend the work of Barba and Langerman [13] to an approximate setting. It particular, is it possible to preprocess convex bodies so that given two such bodies that have been translated and rotated, it can be decided efficiently whether they intersect each other approximately?

Our result on approximate nearest neighbor searching relies on the lifting transformation to reduce the problem to approximate polytope membership. As a consequence, this approach is applicable only to Euclidean distances. This raises the question of whether there exists a more direct route to approximate nearest neighbor searching that achieves similar space-time improvements and yet avoids reliance on lifting. For example, Arya and Chan [5] have presented improvements to approximate

nearest neighbor searching that do not involve lifting. This raises the hope that generalizations to other norms may be possible. While their focus was different from ours (for example, space-time trade-offs are not considered), their results are inferior to our best bounds. These better bounds arise explicitly from concepts like the Mahler volume, which are applicable only in the context of convex approximation, and hence they rely crucially on lifting. A major challenge is whether it is possible to bypass this intermediate step in order to obtain analogous improvements for approximate nearest neighbor searching.

Note added in proof. After the original submission of this paper, the authors discovered a new approach to polytope membership that achieves query time $O(\log \frac{1}{\varepsilon})$ with storage of only $O(1/\varepsilon^{(d-1)/2})$ [9]. As a consequence, it is possible to answer ε-approximate nearest neighbor queries for a set of n points in $O(\log \frac{n}{\varepsilon})$ time with storage of only $O(n/\varepsilon^{d/2})$. While these new results surpass the results of this paper theoretically, the data structure presented there involves significantly larger constant factors and lacks the simplicity and practicality of the approach described here.

Acknowledgment. The authors would like to thank the anonymous reviewers for their many insightful comments.

REFERENCES

[1] P. K. Agarwal, S. Har-Peled, and K. R. Varadarajan, *Approximating extent measures of points*, J. Assoc. Comput. Mach., 51 (2004), pp. 606–635.

[2] P. K. Agarwal, S. Har-Peled, and K. R. Varadarajan, *Geometric approximation via coresets*, in Combinatorial and Computational Geometry, J. E. Goodman, J. Pach, and E. Welzl, eds., MSRI Publications, Cambridge, UK, 2005.

[3] P. K. Agarwal and J. Matoušek, *Ray shooting and parametric search*, SIAM J. Comput., 22 (1993), pp. 794–806, https://doi.org/10.1137/0222051.

[4] N. Alon and J. H. Spencer, *The Probabilistic Method*, Wiley-Interscience, New York, 2000.

[5] S. Arya and T. M. Chan, *Better ε-dependencies for offline approximate nearest neighbor search, Euclidean minimum spanning trees, and ε-kernels*, in Proceedings of the 30th Annual Symposium on Computational Geometry, 2014, pp. 416–425.

[6] S. Arya, G. D. da Fonseca, and D. M. Mount, *A unified approach to approximate proximity searching*, in Proceedings of the 18th Annual European Symposium on Algorithms, 2010, pp. 374–385.

[7] S. Arya, G. D. da Fonseca, and D. M. Mount, *Optimal area-sensitive bounds for polytope approximation*, in Proceedings of the 28th Annual Symposium on Computational Geometry, 2012, pp. 363–372, https://doi.acm.org/10.1145/2261250.2261305.

[8] S. Arya, G. D. da Fonseca, and D. M. Mount, *Approximate Polytope Membership Queries*, arXiv:1604.01183 [cs.CG], 2016.

[9] S. Arya, G. D. da Fonseca, and D. M. Mount, *Optimal approximate polytope membership*, in Proceedings of the 28th Annual ACM-SIAM Symposium on Discrete Algorithms, 2017, pp. 270–288.

[10] S. Arya, T. Malamatos, and D. M. Mount, *Space-time tradeoffs for approximate nearest neighbor searching*, J. Assoc. Comput. Mach., 57 (2009), pp. 1–54, https://doi.acm.org/10.1145/1613676.1613677.

[11] S. Arya and D. M. Mount, *Approximate range searching*, Comput. Geom. Theory Appl., 17 (2000), pp. 135–163.

[12] K. Ball, *An elementary introduction to modern convex geometry*, in Flavors of Geometry, S. Levy, ed., Math Sci. Res. Inst. Publ. 31, Cambridge University Press, Cambridge, UK, 1997, pp. 1–58.

[13] L. Barba and S. Langerman, *Optimal detection of intersections between convex polyhedra*, in Proceedings of the 26th Annual ACM-SIAM Symposium on Discrete Algorithms, 2015, pp. 1641–1654.

[14] G. Barequet and S. Har-Peled, *Efficiently approximating the minimum-volume bounding box of a point set in three dimensions*, J. Algorithms, 38 (2001), pp. 91–109.

[15] J. L. Bentley, M. G. Faust, and F. P. Preparata, *Approximation algorithms for convex hulls*, Commun. ACM, 25 (1982), pp. 64–68, https://doi.acm.org/10.1145/358315.358392.
[16] J. Bourgain and V. D. Milman, *New volume ratio properties for convex symmetric bodies*, Invent. Math., 88 (1987), pp. 319–340, https://doi.org/10.1007/BF01388911.
[17] E. M. Bronshteyn and L. D. Ivanov, *The approximation of convex sets by polyhedra*, Siberian Math. J., 16 (1976), pp. 852–853.
[18] E. M. Bronstein, *Approximation of convex sets by polytopes*, J. Math. Sci., 153 (2008), pp. 727–762.
[19] C. J. C. Burges, *A tutorial on support vector machines for pattern recognition*, Data Min. Knowl. Discov., 2 (1998), pp. 121–167, https://dx.doi.org/10.1023/A:1009715923555.
[20] T. M. Chan, *Fixed-dimensional linear programming queries made easy*, in Proceedings of the 12th Annual Symposium on Computational Geometry, 1996, pp. 284–290, https://doi.acm.org/10.1145/237218.237397.
[21] T. M. Chan, *Output-sensitive results on convex hulls, extreme points, and related problems*, Discrete Comput. Geom., 16 (1996), pp. 369–387.
[22] T. M. Chan, *Closest-point problems simplified on the RAM*, in Proceedings of the 13th Annual ACM-SIAM Symposium on Discrete Algorithms, 2002, pp. 472–473.
[23] T. M. Chan, *Faster core-set constructions and data-stream algorithms in fixed dimensions*, Comput. Geom. Theory Appl., 35 (2006), pp. 20–35, https://dx.doi.org/10.1016/j.comgeo.2005.10.002.
[24] T. M. Chan, *Optimal partition trees*, in Proceedings of the 26th Annual Symposium on Computational Geometry, 2010, pp. 1–10, https://doi.acm.org/10.1145/1810959.1810961.
[25] B. Chazelle and D. P. Dobkin, *Intersection of convex objects in two and three dimensions*, J. Assoc. Comput. Mach., 34 (1987), pp. 1–27.
[26] B. Chazelle and J. Matoušek, *On linear-time deterministic algorithms for optimization problems in fixed dimension*, J. Algorithms, 21 (1996), pp. 579–597, https://doi.org/10.1006/jagm.1996.0060.
[27] K. L. Clarkson, *Algorithms for polytope covering and approximation*, in Proceedings of the 3rd International Workshop on Algorithms of Data Structure, 1993, pp. 246–252.
[28] K. L. Clarkson, *An algorithm for approximate closest-point queries*, in Proceedings of the 10th Annual Symposium of Computational Geometry, 1994, pp. 160–164.
[29] T. H. Cormen, C. E. Leiserson, R. L. Rivest, and C. Stein, *Introduction to Algorithms*, MIT Press, Cambridge, MA, 2001.
[30] M. de Berg, O. Cheong, M. van Kreveld, and M. Overmars, *Computational Geometry: Algorithms and Applications*, 3rd ed., Springer, New York, 2010.
[31] D. P. Dobkin and D. G. Kirkpatrick, *Fast detection of polyhedral intersection*, Theoret. Comput. Sci., 27 (1983), pp. 241–253.
[32] R. M. Dudley, *Metric entropy of some classes of sets with differentiable boundaries*, Approx. Theory, 10 (1974), pp. 227–236.
[33] C. A. Duncan, M. T. Goodrich, and S. Kobourov, *Balanced aspect ratio trees: Combining the advantages of k-d trees and octrees*, J. Algorithms, 38 (2001), pp. 303–333.
[34] H. Edelsbrunner, *Algorithms in Combinatorial Geometry*, Springer, New York, 1987.
[35] H. G. Eggleston, *Convexity*, Cambridge University Press, Cambridge, UK, 1958.
[36] J. Erickson, L. J. Guibas, J. Stolfi, and L. Zhang, *Separation-sensitive collision detection for convex objects*, in Proceedings of the 10th Annual ACM-SIAM Symposium of Discrete Algorithms, 1999, pp. 327–336.
[37] P. M. Gruber, *Asymptotic estimates for best and stepwise approximation of convex bodies* I, Forum Math., 5 (1993), pp. 521–537.
[38] S. Har-Peled, *A replacement for Voronoi diagrams of near linear size*, in Proceedings of the 42nd Annual IEEE Symposium on Foundations of Computer Science, 2001, pp. 94–103.
[39] S. Har-Peled, *Geometric Approximation Algorithms*, AMS, Providence, RI, 2011.
[40] J. K. Boroczky, *Approximation of general smooth convex bodies*, Adv. Math., 153 (2000), pp. 325–341.
[41] G. Kuperberg, *From the Mahler conjecture to Gauss linking integrals*, Geom. Funct. Anal., 18 (2008), pp. 870–892.
[42] J. Matoušek and O. Schwarzkopf, *On ray shooting in convex polytopes*, Discrete Comput. Geom., 10 (1993), pp. 215–232.
[43] J. Matoušek, *Reporting points in halfspaces*, Comput. Geom. Theory Appl., 2 (1992), pp. 169–186.
[44] J. Matoušek, *Linear optimization queries*, J. Algorithms, 14 (1993), pp. 432–448, https://dx.doi.org/10.1006/jagm.1993.1023.
[45] J. S. B. Mitchell and S. Suri, *Separation and approximation of polyhedral objects*, Comput. Geom. Theory Appl., 5 (1995), pp. 95–114.

[46] E. A. Ramos, *Linear programming queries revisited*, in Proceedings of the 16th Annual Symposium on Computational Geometry, 2000, pp. 176–181, https://doi.acm.org/10.1145/336154.336198.

[47] Y. Sabharwal, S. Sen, and N. Sharma, *Nearest neighbors search using point location in balls with applications to approximate Voronoi decompositions*, J. Comput. System Sci., 72 (2006), pp. 955–977.

[48] L. A. Santaló, *An affine invariant for convex bodies of n-dimensional space*, Port. Math., 8 (1949), pp. 155–161 (in Spanish).

Attached Article 2

Next, we include the expanded arXiv version of the following conference paper.

S. Arya, G. D. da Fonseca, and D. M. Mount. Optimal approximate polytope membership. In *Proceedings of the 28th Annual ACM-SIAM Symposium on Discrete Algorithms (SODA)*, pages 270–288, 2017.

http://arxiv.org/abs/1612.01696

Optimal Approximate Polytope Membership

Sunil Arya*
Department of Computer Science and Engineering
The Hong Kong University of
Science and Technology
Clear Water Bay, Kowloon, Hong Kong
arya@cse.ust.hk

Guilherme D. da Fonseca
Université Clermont Auvergne and
LIMOS
Clermont-Ferrand, France
fonseca@isima.fr

David M. Mount†
Department of Computer Science and
Institute for Advanced Computer Studies
University of Maryland
College Park, Maryland 20742
mount@cs.umd.edu

Abstract

In the polytope membership problem, a convex polytope K in $\mathbb{R}^d$ is given, and the objective is to preprocess K into a data structure so that, given a query point $q \in \mathbb{R}^d$, it is possible to determine efficiently whether $q \in K$. We consider this problem in an approximate setting and assume that d is a constant. Given an approximation parameter $\varepsilon > 0$, the query can be answered either way if the distance from q to K's boundary is at most ε times K's diameter. Previous solutions to the problem were on the form of a space-time trade-off, where logarithmic query time demands $O(1/\varepsilon^{d-1})$ storage, whereas storage $O(1/\varepsilon^{(d-1)/2})$ admits roughly $O(1/\varepsilon^{(d-1)/8})$ query time. In this paper, we present a data structure that achieves logarithmic query time with storage of only $O(1/\varepsilon^{(d-1)/2})$, which matches the worst-case lower bound on the complexity of any ε-approximating polytope. Our data structure is based on a new technique, a hierarchy of ellipsoids defined as approximations to Macbeath regions.

As an application, we obtain major improvements to approximate Euclidean nearest neighbor searching. Notably, the storage needed to answer ε-approximate nearest neighbor queries for a set of n points in $O(\log \frac{n}{\varepsilon})$ time is reduced to $O(n/\varepsilon^{d/2})$. This halves the exponent in the ε-dependency of the existing space bound of roughly $O(n/\varepsilon^d)$, which has stood for 15 years (Har-Peled, 2001).

1 Introduction

Convex polytopes are key structures in many areas of mathematics and computation. In this paper, we consider a fundamental search problem related to these objects. Let K denote a convex polytope in $\mathbb{R}^d$, that is, the bounded intersection of n halfspaces. The *polytope membership problem* is that of preprocessing K so that it is possible to determine efficiently whether a given query point $q \in \mathbb{R}^d$ lies within K. Throughout, we assume that the dimension d is a fixed constant and that K is full dimensional.

It follows from standard results in projective duality that polytope membership is equivalent to answering halfspace emptiness queries for a set of n points in $\mathbb{R}^d$. In dimension $d \leq 3$, it is possible to build a data structure of linear size that can answer such queries in logarithmic time [29, 30]. In higher dimensions, however, the fastest exact data structures with near-linear space have a query time of roughly $O\big(n^{1-1/\lfloor d/2 \rfloor}\big)$ [41], which is unacceptably high for many applications.

Polytope membership is a special case of polytope intersection queries [16, 27, 30]. Recently, Barba and Langerman [16] showed that for any fixed d, it is possible to preprocess polytopes in $\mathbb{R}^d$ so that given

*Research supported by the Research Grants Council of Hong Kong, China under project number 610012.
†Research supported by NSF grant CCF–1618866.

two such polytopes that have been translated and rotated, it can be determined whether they intersect each other in time that is logarithmic in their total combinatorial complexity. The preprocessing time and space are quite high, growing as the combinatorial complexity of the polytope (which can be as high as $\Theta(n^{\lfloor d/2 \rfloor})$) raised to the power $\lfloor d/2 \rfloor$.

The lack of efficient exact solutions has motivated consideration of approximate solutions. Let ε be a positive real parameter, and let $\mathrm{diam}(K)$ denote K's diameter. Given a query point $q \in \mathbb{R}^d$, an *ε-approximate polytope membership query* returns a positive result if $q \in K$, a negative result if the distance from q to its closest point in K is greater than $\varepsilon \cdot \mathrm{diam}(K)$, and it may return either result otherwise. Polytope membership queries, both exact and approximate, arise in many application areas, such as linear-programming and ray-shooting queries [22, 26, 40, 42, 43], nearest neighbor searching and the computation of extreme points [23, 28], collision detection [35], and machine learning [21].

Dudley [31] showed that, for any convex body K in $\mathbb{R}^d$, it is possible to construct an ε-approximating polytope P with $O(1/\varepsilon^{(d-1)/2})$ facets. This bound is asymptotically tight in the worst case, even when K is a Euclidean ball. This construction implies a (trivial) data structure for approximate polytope membership problem with space and query time $O(1/\varepsilon^{(d-1)/2})$. Another simple solution arises from the approximation proposed by Bentley *et al.* [17]. A d-dimensional grid with cells of size $\Theta(\varepsilon \cdot \mathrm{diam}(K))$ is created and for every column along the x_d-axis, the two extreme x_d values where the column intersects K are stored. Given a query point q, it is easy to determine if $q \in P$ in constant time (assuming a model of computation that supports the floor function). The storage required by the approach is $O(1/\varepsilon^{d-1})$.

In [4], the authors presented a simple and practical data structure for the approximate polytope membership problem, called *SplitReduce*. Given a parameter t, space is subdivided hierarchically using a quadtree until each cell either (1) lies entirely inside K, (2) entirely outside K, or (3) intersects K's boundary and is locally approximable by at most t halfspaces. In the latter case, the leaf node associated with such a cell stores such a set of hyperplanes. To answer a query, the quadtree is descended until arriving at the leaf node whose cell contains the query point. If this node is not labeled as inside or outside, the query is answered by testing whether the query point lies within all the halfspaces stored in the leaf node. In [4] it is shown that the quadtree height is $O(\log \frac{1}{\varepsilon})$, and therefore the overall query time is $O(\log \frac{1}{\varepsilon} + t)$.

A more refined analysis is presented in [6], showing that the minimum storage of $O(1/\varepsilon^{(d-1)/2})$ is attained for query time $t = \Theta\big((\log \frac{1}{\varepsilon})/\varepsilon^{(d-1)/8}\big)$. Furthermore, a space-time trade-off is presented that involves a piecewise linear function. Obtaining a tight analysis remains an open problem. A lower-bound proof shows that the storage requirement increases when the query time t drops down to roughly $O(1/\varepsilon^{(d-1)/18})$ [4]. Furthermore, the data structure provides no improvement over the storage in [17] when the query time is polylogarithmic.

While the SplitReduce data structure is both simple and practical, the question of whether it is possible to achieve query time $O(\log \frac{1}{\varepsilon})$ with minimum storage $O(1/\varepsilon^{(d-1)/2})$ has remained open. In this paper, we give an affirmative answer to this question. We abandon the quadtree-based approach of [4] and [6] in favor of a data structure involving a hierarchy of ellipsoids. These ellipsoids are selected through a sampling process that is inspired by a classical structure from the theory of convexity, called *Macbeath regions* [39]. Here is our main result.

Theorem 1.1 *Given a convex polytope K in $\mathbb{R}^d$ and an approximation parameter $0 < \varepsilon \leq 1$, there is a data structure that can answer ε-approximate polytope membership queries with*

$$\textit{Query time: } O\left(\log \frac{1}{\varepsilon}\right) \textit{ and Space: } O\left(\frac{1}{\varepsilon^{(d-1)/2}}\right).$$

Our focus is on the existence of this data structure. Preprocessing will be discussed in future work, but assuming that K is represented as the intersection of h halfspaces, the construction described in Section 3.1 can be implemented in time $O(n + \mathrm{poly}(1/\varepsilon))$, with polynomial exponents depending on d. The principal contribution of this paper is to show that through the use of a more "shape-sensitive" approach, it is possible to achieve dramatic improvements over the space requirements of the data structure.

As evidence of the importance of this result, we show that it can be applied to produce significant improvements in the efficiency of *approximate nearest-neighbor searching* in Euclidean space. Approximate nearest neighbor searching in spaces of fixed dimension has been widely studied. Data structures

with $O(n)$ storage and query times no better than $O(\log n + 1/\varepsilon^{d-1})$ have been proposed by several authors [13, 18, 25, 32]. In subsequent papers, it was shown that query times could be reduced at the expense of greater storage [24, 28, 37, 44]. Har-Peled introduced the AVD (approximate Voronoi diagram) data structure and showed that $O(\log \frac{n}{\varepsilon})$ query time could be achieved using $\widetilde{O}(n/\varepsilon^d)$ space [37]. (The notation $\widetilde{O}(\cdot)$ conceals logarithmic factors.)

Space-time trade-offs were established for the AVD in a series of papers [3, 8, 9, 11]. At one end of the spectrum, it was shown that with $O(n)$ storage, queries could be answered in time $O(\log n + 1/\varepsilon^{(d-1)/2})$. At the other end, queries could be answered in time $O(\log \frac{n}{\varepsilon})$ with space $\widetilde{O}(n/\varepsilon^d)$. In [4], the authors presented a reduction from Euclidean approximate nearest neighbor searching to polytope membership. They established significant improvements to the best trade-offs throughout the middle of the spectrum, but the extremes were essentially unchanged [4, 6]. While the AVD is simple and practical, in [11] lower bounds were presented that imply that significant improvements at the extreme ends of the spectrum are not possible in this model. Through the use of our new data structure for polytope membership, we achieve the following improved trade-off.

Theorem 1.2 *Given a set X of n points in $\mathbb{R}^d$, an approximation parameter $0 < \varepsilon \leq 1$, and m such that $\log \frac{1}{\varepsilon} \leq m \leq 1/(\varepsilon^{d/2} \log \frac{1}{\varepsilon})$, there is a data structure that can answer Euclidean ε-approximate nearest neighbor queries with*

$$\textit{Query time: } O\left(\log n + \frac{1}{m \cdot \varepsilon^{d/2}}\right) \quad \textit{and}$$

$$\textit{Space: } O(n\,m)\,.$$

By setting m to its upper limit it is possible to achieve logarithmic query time while roughly *halving* the exponent in the ε-dependency of the previous best bound, as expressed in the following corollary.

Corollary 1.1 *Given a set X of n points in $\mathbb{R}^d$ and an approximation parameter $0 < \varepsilon \leq 1$, there is a data structure that can answer Euclidean ε-approximate nearest neighbor queries with*

$$\textit{Query time: } O\left(\log \frac{n}{\varepsilon}\right) \quad \textit{and} \quad \textit{Space: } O\left(\frac{n}{\varepsilon^{d/2}}\right).$$

The rest of the paper is organized as follows. In the next section we present definitions and preliminary results. In Section 3 we present the data structure and analyze its performance. Section 4 discusses the application to approximate nearest-neighbor searching.

2 Geometric Preliminaries

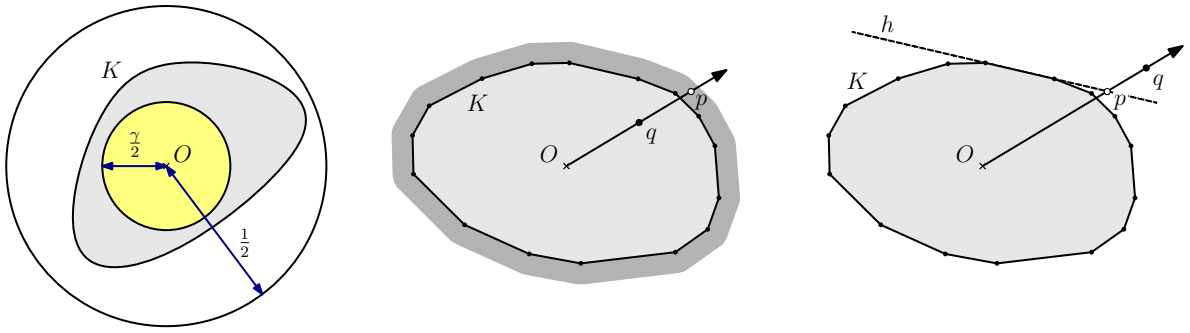
Throughout, we assume that K is presented as the intersection of halfspaces. Note however that our results are largely insensitive to the exact representation or the combinatorial complexity of K. (The exceptions are our remarks on the construction of the data structure and choice of hyperplane witnesses to non-membership). For this reason, we will often refer to K simply as a convex body.

It will be convenient to define the approximation error in absolute terms. Given a query point $q \in \mathbb{R}^d$, an *absolute* ε-approximate polytope membership query returns a positive result if $q \in K$, a negative result if the distance from q to its closest point in K is greater than ε, and it may return either result otherwise. We may assume throughout that $d \geq 4$, since polytope membership queries (which may be applied to the Dudley approximation) can be answered exactly in logarithmic time for $d \leq 3$ [30].

2.1 Canonical Position and Ray Shooting.

Let ∂K denote the boundary of K. Let O denote the origin of $\mathbb{R}^d$, and for $x \in \mathbb{R}^d$ and $r \geq 0$, let $B^r(x)$ denote the Euclidean ball of radius r centered at x. Given a parameter $0 < \gamma \leq 1$, we say that a convex body K is *γ-fat* if there exist concentric Euclidean balls B and B', such that $B \subseteq K \subseteq B'$, and $\text{radius}(B)/\text{radius}(B') \geq \gamma$. We say that K is *fat* if it is γ-fat for a constant γ (possibly depending on d, but not on ε).

Let B_0 denote a ball of radius $r_0 = 1/2$ centered at the origin. For $0 < \gamma \leq 1$, let γB_0 denote the concentric ball of radius $\gamma r_0 = \gamma/2$. We say that a convex body K is in *γ-canonical form* if its



(a) (b) (c)

Figure 1: (a) γ-canonical form, (b) ε-approximate ray-shooting query, (c) witness.

boundary is nested between γB_0 and B_0 (see Figure 1(a)). A body in γ-canonical form is γ-fat, and $\text{diam}(K) \in [\gamma, 1]$. We will refer to point O as the *center* of K.

The next lemma shows that, up to constant factors, the problem of answering relative ε-approximate polytope membership queries can be reduced to the problem of answering absolute (ε/d)-approximate queries with respect to a convex body in $(1/d)$-canonical form. The proof follows from a combination of John's Theorem [38] and Lemma 3.1 of Agarwal *et al.* [1]. (Also, see Lemma 2.1 of the arXiv version of [7].)

Lemma 2.1 *Let $K \subset \mathbb{R}^d$ be a convex body. There exists a non-singular affine transformation T such that $T(K)$ is in $(1/d)$-canonical form. Further, if q is a point at distance greater than $\varepsilon \cdot \text{diam}(K)$ from K, then $T(q)$ is at distance greater than ε/d from $T(K)$.*

In light of this result, we may assume henceforth that K is presented in γ-canonical form, for any constant γ (depending on dimension), and that ε has been appropriately scaled. (This scaling will affect the constant factors hidden in our asymptotic bounds.) Henceforth, we focus on the problem of answering absolute ε-approximate polytope membership queries with respect to K.

Our query algorithm solves a slightly more general problem, which will be exploited later in Section 4. Given a convex body in γ-canonical form and any point $q \in \mathbb{R}^d \setminus \{O\}$, consider the (infinite) ray with origin at O and passing through q, which we denote as Oq. An *ε-approximate ray shooting query* returns a point p that lies on this ray and is not internal to K but lies within distance ε of K[1] (see Figure 1(b)). Given the answer to such a ray-shooting query, we can answer approximate membership queries for a query point q by applying the query to the ray Oq and testing whether q lies on the portion of the ray between O and p. If so, then (by convexity and the fact that O is interior to K) q lies within distance ε of K. If not, q does not lie within K. In Section 3 we will show the following.

Lemma 2.2 *Given an arbitrary constant γ, a convex polytope K in $\mathbb{R}^d$ that is in γ-canonical form, and an approximation parameter $0 < \varepsilon \leq 1$, there is a data structure that can answer ε-approximate ray-shooting queries in $O(\log \frac{1}{\varepsilon})$ time and $O(1/\varepsilon^{(d-1)/2})$ space.*

Theorem 1.1 follows directly from Lemmas 2.1 and 2.2. Our ray-shooting algorithm satisfies the additional property that, when K is given as the intersection of halfspaces, the reported point p lies on the bounding hyperplane h of one of these halfspaces (see Figure 1(c)). The query returns not only p but h as well. As such, if q is reported to lie outside of K, then h serves as a witness to q's non-membership. This fact will be exploited in Section 4.

[1] In light of Lemma 2.1, approximate ray-shooting queries also can be defined for an arbitrary convex body. The ray's origin is chosen to be the center of the John ellipsoid and the distance to the point p is relative to K's diameter. In general the ray's central point may be located at any point in K's interior with the property that K's boundary is sandwiched between two uniformly scaled copies of an ellipsoid, both centered at this point. As in Lemma 2.1, the value of ε needs to be adjusted based on the scale factor.

2.2 Caps and Macbeath Regions.

Much of the material in this section has been presented in [7]. We include it here for the sake of completeness.

Given a convex body K, a *cap* C is defined to be the nonempty intersection of the convex body K with a halfspace (see Figure 2(a)). Let h denote the hyperplane bounding this halfspace. We define the *base* of C to be $h \cap K$. The *apex* of C is any point in the cap such that the supporting hyperplane of K at this point is parallel to h. The *width* of C, denoted width(C), is the distance between h and this supporting hyperplane. Given any cap C of width w and a real $\rho \geq 0$, we define its *ρ-expansion*, denoted C^ρ, to be the cap of K cut by a hyperplane parallel to and at distance ρw from this supporting hyperplane. (Note that $C^\rho = K$, if ρw exceeds the width of K along the defining direction.) An easy consequence of convexity is that, for $\rho \geq 1$, C^ρ is a subset of the region obtained by scaling C by a factor of ρ about its apex. This implies the following lemma.

Lemma 2.3 *Let $K \subset \mathbb{R}^d$ be a convex body and $\rho \geq 1$. For any cap C of K, $\text{vol}(C^\rho) \leq \rho^d \cdot \text{vol}(C)$.*

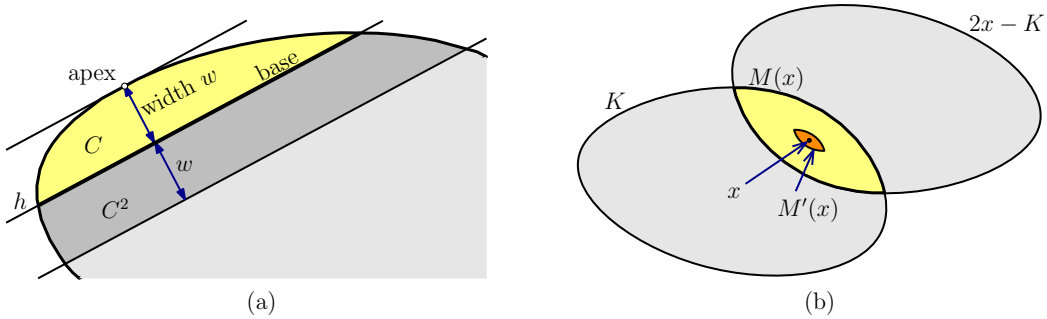


Figure 2: (a) Cap concepts and (b) Macbeath regions.

Given a point $x \in K$ and real parameter $\lambda \geq 0$, the *Macbeath region* $M^\lambda(x)$ (also called an *M-region*) is defined as:

$$M^\lambda(x) \;=\; x + \lambda((K - x) \cap (x - K)).$$

It is easy to see that $M^1(x)$ is the intersection of K and the reflection of K around x (see Figure 2(b)), and so $M^1(x)$ is centrally symmetric about x. $M^\lambda(x)$ is a scaled copy of $M^1(x)$ by the factor λ about x. We refer to x as the *center* of $M^\lambda(x)$ and to λ as its *scaling factor*. As a convenience, we define $M(x) = M^1(x)$ and $M'(x) = M^{1/5}(x)$. We refer to the latter as the *shrunken* Macbeath region.

Macbeath regions have found numerous uses in the theory of convex sets and the geometry of numbers (see Bárány [15] for an excellent survey). They have also been applied to a growing number of results in the field of computational geometry, particularly to construct lower bounds for range searching [10,14,19] and to bound the complexity of an ε-approximating polytope [5,7].

Given any point $x \in K$, we define a *minimal cap* $C(x)$ to be the cap with minimum volume that contains x. Clearly, the base of the minimal cap must pass through x. In fact, a standard variational argument implies x is the centroid of the base (otherwise, we could decrease the cap volume by an infinitesimal rotation of the base about x [36]). If the minimal cap is not unique, the notation $C(x)$ will refer to any one of these caps fixed arbitrarily. Define $v(x) = \text{vol}(C(x))$ and $\text{width}(x) = \text{width}(C(x))$. It will be convenient to use $C^\rho(x)$ to refer to the ρ-expansion of $C(x)$, that is, $C^\rho(x) = (C(x))^\rho$.

We now present two lemmas that encapsulate key properties of Macbeath regions, which will be useful in the development of our data structure. The first lemma shows that if two shrunken Macbeath regions have a nonempty intersection, then a constant factor expansion of one contains the other [19,36]. Since the statement we need is slightly different from that proved in earlier papers, we give a proof in the appendix.

Lemma 2.4 *Let K be a convex body, and let $\lambda \leq 1/5$ be any real. If $x, y \in K$ such that $M^\lambda(x) \cap M^\lambda(y) \neq \emptyset$, then $M^\lambda(y) \subseteq M^{4\lambda}(x)$.*

The next lemma shows that the minimal cap associated with a point is contained within a suitable constant factor expansion of the associated Macbeath region. It is a straightforward adaptation of a lemma proved by Ewald, Larman and Rogers [36] (see proof of Lemma 4 in [36]).

Lemma 2.5 *Let $K \subset \mathbb{R}^d$ be a convex body in γ-canonical form, and let $\Delta_0 = \frac{1}{2}(\gamma^2/(4d))^d$ be a constant. If x is a point in K that lies within distance Δ_0 of ∂K, then $C(x) \subseteq M^{3d}(x)$.*

The following lemma is an immediate consequence of the definition of Macbeath region.

Lemma 2.6 *Let K be a convex body and $\lambda > 0$. If x is a point in a cap C of K, then $M^\lambda(x) \cap K \subseteq C^{1+\lambda}$. Furthermore, if $\lambda \leq 1$, then $M^\lambda(x) \subseteq C^{1+\lambda}$.*

The next lemma is useful in situations when we know that a shrunken Macbeath region partially overlaps a cap of K. It allows us to conclude that a constant factor expansion of the cap will fully contain the Macbeath region. The proof appears in [7].

Lemma 2.7 *Let K be a convex body. Let C be a cap of K and x be a point in K such that $C \cap M'(x) \neq \emptyset$. Then $M'(x) \subseteq C^2$.*

2.3 Relating Distances and Widths.

In this section we present a number of geometric results demonstrating the relationship between three notions of the distance from a point lying within a convex body to body's boundary. Throughout, let K be a convex body in γ-canonical form where γ is a constant and let $x \in K$. Recall that width(x) is the width of x's minimum cap. Define $\delta(x)$ to be the minimum distance from x to any point on ∂K. For the sake of ray-shooting queries, we define a ray-based notion of distance as well. Given $x \in K$, consider the intersection point p of ∂K and the ray emanating from O and passing through x. Define x's *ray-distance*, denoted ray(x), to be $\|xp\|$ (see Figure 3).

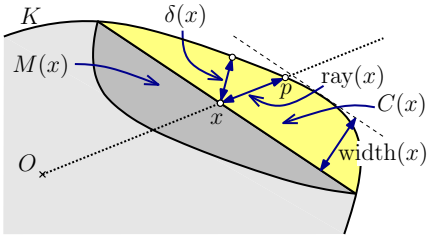


Figure 3: Relating $\delta(x)$, width(x), and ray(x).

First we relate ray(x) and $\delta(x)$. The lower bound on ray(x) is trivial and the upper bound follows by a straightforward adaptation of Lemma 4.2 of [7].

Lemma 2.8 *Let K be a convex body in γ-canonical form. For any point $x \in K$, $\delta(x) \leq \text{ray}(x) \leq \delta(x)/\gamma$.*

Next, let us relate width(x) and $\delta(x)$. Clearly, width$(x) \geq \delta(x)$. In Lemma 2.10, we show that close to the boundary, width(x) cannot exceed $\delta(x)$ by more than a constant factor. Its proof is based standard properties of Macbeath regions and the following lemma.

Lemma 2.9 *Let K be a convex body in γ-canonical form. Let C_1 and C_2 be two caps of K such that $C_1 \subseteq C_2$. Then* width$(C_1) \leq 2 \cdot$ width$(C_2)/\gamma$.

Proof. We consider two cases depending on whether the origin O is inside C_1 or not. First, if $O \in C_1$, then $O \in C_2$. Since K contains the ball $B^{\gamma/2}(O)$, it follows that width$(C_2) \geq \gamma/2$. Since K is contained within the ball $B^{1/2}(O)$, we have width$(C_1) \leq 1$. Thus, width$(C_1) \leq 2 \cdot$ width$(C_2)/\gamma$.

Otherwise, we have $O \notin C_1$. Consider the segment joining O to t, where t is the apex of C_1. Let x denote the point of intersection of this segment with the base of C_1. Clearly, width$(C_1) \leq$ ray(x). By Lemma 2.8, ray$(x) \leq \delta(x)/\gamma$. Thus, width$(C_1) \leq \delta(x)/\gamma$. Also, since $x \in C_2$, we have $\delta(x) \leq$ width(C_2). Thus, width$(C_1) \leq$ width$(C_2)/\gamma$, completing the proof. □

Lemma 2.10 *Let $K \subset \mathbb{R}^d$ be a convex body in γ-canonical form, and let Δ_0 be the constant of Lemma 2.5. If x is a point in K such that $\delta(x) \leq \Delta_0$, then* $\text{width}(x) \leq (2/\gamma)(3d+1)\delta(x)$.

Proof. Let t denote the point on ∂K that is closest to x. Consider the supporting hyperplane of K at t that is orthogonal to segment xt. Consider the halfspace bounded by this hyperplane which does not contain K in its interior. Translate this halfspace such that the bounding hyperplane passes through x. Let C denote the cap formed by intersecting this halfspace with K. Note that the width of cap C is $\delta(x)$. By Lemma 2.6, $M^{3d}(x) \cap K \subseteq C^{3d+1}$. Since $\delta(x) \leq \Delta_0$, it follows from Lemma 2.5 that $C(x) \subseteq M^{3d}(x)$. By definition, $C(x) \subseteq K$, so we have

$$C(x) \;\subseteq\; M^{3d}(x) \cap K \;\subseteq\; C^{3d+1}.$$

By Lemma 2.9, it follows that

$$\begin{aligned} \text{width}(x) &= \text{width}(C(x)) \\ &\leq \frac{2}{\gamma}\text{width}(C^{3d+1}) \;=\; \frac{2}{\gamma}(3d+1)\delta(x), \end{aligned}$$

as desired. □

The following lemma, illustrated in Figure 4, will be useful to analyze the ray shooting performed by our data structure.

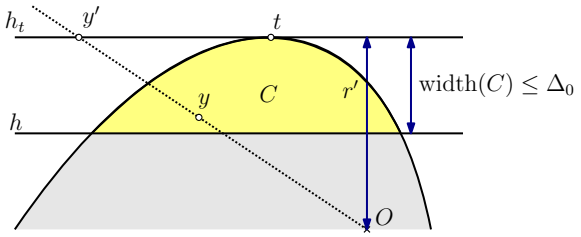


Figure 4: Statement of Lemma 2.11.

Lemma 2.11 *Let K be a convex body in γ-canonical form, and let Δ_0 be the constant of Lemma 2.5. Let C be a cap of width at most Δ_0 defined by a hyperplane h, and let y be any point in C. Let t be C's apex, and let h_t be the hyperplane parallel to h that passes through t. Letting y' denote the intersection of line Oy and h_t, we have $\|yy'\| \leq 2 \cdot \text{width}(C)/\gamma$.*

Proof. Given that $y \in C$, $\|yy'\|$ is maximized when y lies on $C \cap h$, and so let us assume this. Since K is in γ-canonical form, it is nested between two balls of radii $r = \gamma/2$ and $R = 1/2$ centered at O. Let r' denote the perpendicular distance from O to h_t. Clearly, h_t is a supporting hyperplane of K, and so $r' \geq r$. By definition of Δ_0 and since $\gamma \leq 1$, we have $\Delta_0 \leq \gamma/4 = r/2$. Let $R' = \|Oy\|$. Since $y \in K$, $R' \leq R$. Letting $w = \text{width}(C)$, by similar triangles we have $R'/(r' - w) = \|yy'\|/w$. Therefore,

$$\begin{aligned} \|yy'\| &= \frac{R'}{r'-w}w \;\leq\; \frac{R'}{r'-\Delta_0}w \;\leq\; \frac{R'}{r-(r/2)}w \\ &\leq \frac{R}{r/2}w \;=\; \frac{2\cdot\text{width}(C)}{\gamma}, \end{aligned}$$

as desired. □

Finally, we establish a monotonicity relationship between $\delta(x)$ and $\text{ray}(x)$ that holds close to the boundary. For any $\delta > 0$, define the *δ-erosion* of K, denoted $K(\delta)$, to be the closed convex body formed by removing from K all points lying within distance δ of ∂K. We can define $K(\delta)$ equivalently as follows. Let $\mathcal{H}$ denote the set of supporting halfspaces of K, so that $K = \bigcap_{H \in \mathcal{H}} H$. Letting $\mathcal{H}(\delta)$ denote the set of halfspaces obtained by translating each halfspace of $\mathcal{H}$ towards O by δ, we have $K(\delta) = \bigcap_{H \in \mathcal{H}(\delta)} H$. Recalling that $B^{\gamma/2}(O) \subseteq K$, the next lemma follows from elementary geometry.

Lemma 2.12 *Let K be a convex body in γ-canonical form. The following hold:*

(a) if $\delta < \gamma/2$, then $O \in K(\delta)$.

(b) Consider any ray emanating from O. Let x and y denote the points of intersection of this ray with the boundaries of $K(\gamma/2)$ and K, respectively. As point p moves along this ray from x to y, $\delta(p)$ decreases strictly monotonically.

2.4 Further Properties of Macbeath Regions.

Finally, we identify some useful novel properties of Macbeath regions. The first lemma is a useful utility. Lemma 2.14 shows that all the points in a shrunken Macbeath region have similar distances from the boundary of K, and Lemma 2.15 shows that the minimal caps associated with these points have similar volumes.

Lemma 2.13 *Let K be a convex body. If $x \in K$ and $x' \in M'(x)$, then $x \in M^{1/4}(x')$.*

Proof. Recalling that $M'(x) = M^{1/5}(x)$, it follows that there exist points $p_1, p_2 \in K$ such that $x' = x + \frac{1}{5}(p_1 - x)$ and $x' = x + \frac{1}{5}(x - p_2)$. After simple algebraic manipulations, the first equation is equivalent to

$$x \;=\; x' + \frac{1}{4}(x' - p_1). \tag{1}$$

Letting $p_3 = \frac{2}{3}p_2 + \frac{1}{3}x'$, the second equation is equivalent to

$$x \;=\; x' + \frac{1}{4}\left(\frac{2}{3}p_2 + \frac{1}{3}x' - x'\right) \;=\; x' + \frac{1}{4}\left(p_3 - x'\right). \tag{2}$$

As p_3 is a convex combination of p_2 and x', we have $p_3 \in K$. Eq. (1) shows that $x \in x' + (1/4)(x' - K)$, and Eq. (2) shows that $x \in x' + (1/4)(K - x')$. Thus $x \in M^{1/4}(x')$. □

Lemma 2.14 *Let K be a convex body. If $x \in K$ and $x' \in M'(x)$, then $4\delta(x)/5 \le \delta(x') \le 4\delta(x)/3$.*

Proof. To prove the lower bound on $\delta(x')$, let z denote the point of ∂K that is closest to x', and let h be a supporting hyperplane passing through z (see Figure 5). Let ℓ denote the (perpendicular) distance from x to h, and let h' be the translate of h by distance $4\ell/5$ towards x. Because $M(x)$ lies entirely within the halfspace bounded by h that contains the origin, it follows that $M'(x)$ lies entirely within the corresponding halfspace bounded by h'. This implies that $\delta(x') \ge 4\ell/5$. Clearly, $\delta(x) \le \ell$, and hence $\delta(x') \ge 4\ell/5 \ge 4\delta(x)/5$.

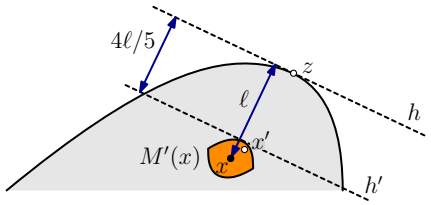


Figure 5: Proof of Lemma 2.14.

To prove the upper bound on $\delta(x')$ observe that, by Lemma 2.13, $x \in M^{1/4}(x')$. A symmetrical argument to the above shows that $\delta(x) \ge 3\delta(x')/4$, as desired. □

Recall that $C(x)$ is the cap of minimum volume that contains x and $v(x) = \mathrm{vol}(C(x))$.

Lemma 2.15 *Let $K \subset \mathbb{R}^d$ be a convex body. If $x \in K$ and $x' \in M'(x)$, then $2^d v(x) \ge v(x') \ge v(x)/2^d$.*

Proof. By Lemma 2.6, $M'(x) \subseteq C^{6/5}(x)$. Therefore, $x' \in C^{6/5}(x)$, implying that the minimum volume cap containing x' has volume at most $\mathrm{vol}(C^{6/5}(x))$. By Lemma 2.3, $\mathrm{vol}(C^{6/5}(x)) \leq (6/5)^d \mathrm{vol}(C(x))$. Thus

$$v(x') \;\leq\; \mathrm{vol}(C^{6/5}(x)) \;\leq\; \left(\frac{6}{5}\right)^d v(x) \;\leq\; 2^d v(x),$$

which proves the first inequality. To prove the second inequality observe that, by Lemma 2.13, $x \in M^{1/4}(x')$. Arguing as in the proof of the first inequality, we obtain $v(x) \leq 2^d v(x')$, as desired. □

3 The Data Structure

Recall that we are given a convex polytope $K \subset \mathbb{R}^d$ in γ-canonical form, where γ is a constant, and our objective is to construct a data structure that can answer ε-approximate ray-shooting queries. Our approach is to compute a series of nested rings within K, each of which surrounds the origin. Each ring is the union of a collection of appropriately scaled Macbeath regions such that any ray shot from the origin hits at least one Macbeath region from each ring (see Figure 6). The rings extend outwards towards the boundary of K. To simplify query processing, we will replace each Macbeath region with a containing ellipsoid whose volume is larger by at most a constant factor. With each successive level these "Macbeath ellipsoids" define successively better approximations to ∂K, with the last ring forming an ε-approximation to ∂K.

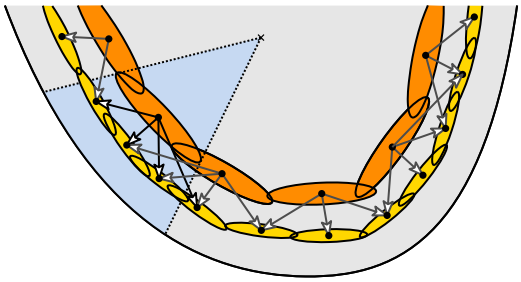


Figure 6: Illustration of two levels of the data structure.

These rings naturally define a layered DAG structure whose nodes correspond to Macbeath ellipsoids. A Macbeath ellipsoid at level i is the child of a Macbeath ellipsoid at level $i-1$ if there is a ray from the origin that intersects both of them. (It will in fact hit the ellipsoid at level $i-1$ before the one at level i.) We will show that each ellipsoid has a constant number of children, and that the overall depth of this DAG is $O(\log \frac{1}{\varepsilon})$.

To define the structure more formally, let Δ_0 be the constant of Lemma 2.5, and for $i \geq 0$ define $\Delta_i = \Delta_0/2^i$. The levels are indexed from 0 to ℓ, where ℓ is the smallest integer such that $\Delta_\ell \leq \gamma^2\varepsilon/(8(3d+1))$. Since γ is a constant, $\ell = O(\log \frac{1}{\varepsilon})$. Recall that $K(\delta)$ denotes the body that results by eroding K by distance δ, and let $\lambda_0 = 1/(20\sqrt{d})$ be a constant. By Lemma 2.12(a), $K(\Delta_0)$ contains the origin O and $K(\Delta_i) \subset K(\Delta_{i+1})$. The nodes at level i of our data structure correspond to a maximal set of disjoint Macbeath regions $M^{\lambda_0}(x)$ whose centers x lie on the boundary of the eroded body $K(\Delta_i)$. For any node u, let x_u denote the center of the associated Macbeath region $M^{\lambda_0}(x_u)$. Define the associated *Macbeath ellipsoid*, denoted $E(x_u)$, to be the circumscribing John ellipsoid of $M^{4\lambda_0}(x_u)$. (Since $M^{\lambda_0}(x_u)$ is centrally symmetric about x_u, $E(x_u)$ will be centered about this point.) We will show that the union of the Macbeath ellipsoids at level i cover $\partial K(\Delta_i)$, implying that any ray emanating from the origin must intersect at least one ellipsoid of each level.

As mentioned above, given nodes u and v from levels i and $i+1$, respectively, v is a child of u if there exists a ray emanating from the origin that intersects both $E(x_u)$ and $E(x_v)$. We can *root* the DAG by creating a special node whose children are all the nodes of level zero. In order to produce a witness for approximate ray-shooting queries, we associate each leaf node with a constant number of supporting hyperplanes of K that locally approximate the boundary of K near the leaf's Macbeath ellipsoid. (This will be discussed in detail in Section 3.1).

Given a ray Oq, the query algorithm descends the DAG by starting at the root and visiting any node at level zero that intersects the ray. Letting u denote the current node, we next visit any child of u whose associated ellipsoid intersects the ray. (Such a child must exist.) Upon reaching the leaf level we intersect Oq with all of its associated supporting hyperplanes and return the intersection point p that is closest to O as the answer to the query (along with the identity of the hyperplane containing p).

In the subsections below, we present a formal analysis of the structure and its properties. In Section 3.1 we sketch its construction. In Section 3.2 we show that each node has $O(1)$ children. In Section 3.3, we show that the total storage required is $O(1/\varepsilon^{(d-1)/2})$. Finally, in Section 3.4 we show that the query algorithm is correct and has query time $O(\log \frac{1}{\varepsilon})$.

3.1 Construction.

Since our focus is on the existential properties of the data structure, we will discuss its construction only at a high level. We are given the convex body K and approximation parameter ε. Due to the approximate nature of the queries, most of the steps can be implemented approximately subject to a suitable adjustment of the constant factors.

The construction begins by converting K into canonical form as described in Lemma 2.1. Next, for $0 \leq i \leq \ell$, the eroded bodies $K(\Delta_i)$ are computed. Recalling the constant λ_0 earlier, for each body $K(\Delta_i)$ we greedily compute a maximal set of points X_i on its boundary such that the Macbeath regions $M^{\lambda_0}(x)$ for $x \in X_i$ are pairwise disjoint. For each point $x \in X_i$, we construct the associated Macbeath region $M^{4\lambda_0}(x)$ and the associated Macbeath ellipsoid $E(x)$. We also create a node for this point at level i of the DAG. Finally, for each pair of nodes at consecutive levels of the DAG, we determine whether there exists a ray emanating from the origin that intersects both of their associated Macbeath ellipsoids. If so, we create a parent-child link between them. We create a special root node, which we connect to all the nodes of level zero. This defines the layered DAG structure.

Next, let us consider the assignment of supporting hyperplanes to the leaves of the data structure. Let u be a leaf node, and let $E(x_u)$ denote the associated Macbeath ellipsoid with center point x_u (see Figure 7). Let $C(x_u)$ denote the corresponding minimum volume cap. Let t be the apex of this cap, and let h_t denote the hyperplane (which is a supporting hyperplane of K) passing through t and parallel to the base of the cap. In Lemma 3.7, we will show that h_t can serve as the desired witness, but in some applications it is desirable that the witness be chosen from K's bounding hyperplanes. By Carathéodory's theorem [34], there is a set of at most d of K's bounding halfspaces whose intersection defines an unbounded simplex that contains K, and this simplex is contained within the halfspace bounded by h_t containing K (shaded in blue in Figure 7). The leaf node u stores this set of hyperplanes, which we denote by H_u.

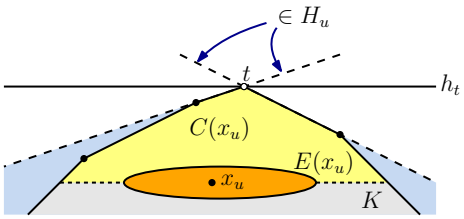


Figure 7: A leaf node in the data structure.

3.2 Bounding the Out-degree.

In this section we show that each node has $O(1)$ children. Intuitively, this involves showing that the set of rays emanating from the origin that pass through a Macbeath ellipsoid for a point on the boundary of $K(\Delta_i)$ can intersect at most a constant number of Macbeath ellipsoids for points on the boundary $K(\Delta_{i+1})$. This is because the points x defining the nodes of each level have disjoint Macbeath regions $M^{\lambda_0}(x)$, which permits us to employ a packing argument.

For any point $x \in K$, recall that $v(x)$ denotes the volume of the minimal cap $C(x)$. Our first lemma considers how $v(x)$ changes as the point x moves towards the boundary of K along a ray emanating from

O. The lemma shows that if the distance to the boundary, $\delta(x)$, decreases by at most a constant factor, then $v(x)$ decreases by no more than some constant factor.

Lemma 3.1 *Let $K \subset \mathbb{R}^d$ be a convex body in γ-canonical form. Let y be a point on the ray Ox, such that $\text{ray}(y) \leq \text{ray}(x)$. If $\delta(y) \geq \delta(x)/\alpha$ for any $\alpha \geq 1$, then $v(y) \geq (\gamma/\alpha)^d\, v(x)$.*

Proof. If $C(y)$ contains O then, by convexity, it would follow that $x \in C(y)$. This would imply that $v(y) = \text{vol}(C(y)) \geq v(x)$, which would prove the lemma. We may assume therefore that $C(y)$ does not contain O.

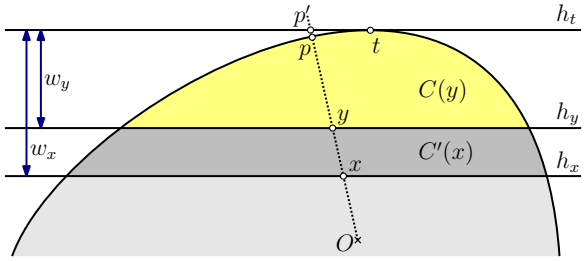


Figure 8: Proof of Lemma 3.1.

Let h_y denote the hyperplane passing through the base of $C(y)$, and let t denote the apex of $C(y)$. Let h_t and h_x denote the hyperplanes parallel to h_y passing through t and x, respectively. Note that h_t is a supporting hyperplane of K. Let $C'(x)$ denote the (not necessarily minimal) cap with apex t, whose base lies on h_x. Let w_y and w_x denote the widths of the caps $C(y)$ and $C'(x)$, respectively. Clearly, $C'(x)$ is a (w_x/w_y)-expansion of the cap $C(y)$, and so by Lemma 2.3, $\text{vol}(C'(x)) \leq (w_x/w_y)^d \cdot \text{vol}(C(y))$. Thus

$$v(x) \;\leq\; \text{vol}(C'(x)) \;\leq\; \left(\frac{w_x}{w_y}\right)^d v(y). \tag{3}$$

Next we show that w_y is not much smaller than w_x. Let p and p' denote the points of intersection of the ray Ox with ∂K and h_t, respectively. Using elementary geometry and the facts that $\text{ray}(y) \geq \delta(y)$ and $\text{ray}(x) \leq \delta(x)/\gamma$ (Lemma 2.8), we obtain

$$\begin{aligned}\frac{w_x}{w_y} \;&=\; \frac{\|xp'\|}{\|yp'\|} \;=\; \frac{\text{ray}(x) + \|pp'\|}{\text{ray}(y) + \|pp'\|} \\ &\leq\; \frac{\text{ray}(x)}{\text{ray}(y)} \;\leq\; \frac{\delta(x)/\gamma}{\delta(y)} \;\leq\; \frac{\alpha}{\gamma}.\end{aligned}$$

Substituting this bound in Equation 3, we obtain $v(x) \leq (\alpha/\gamma)^d\, v(y)$, which completes the proof. □

The following lemma relates the Macbeath regions associated with a node and any of its children.

Lemma 3.2 *Let $K \subset \mathbb{R}^d$ be a convex body in γ-canonical form for some constant γ, and let Δ_0 be the constant of Lemma 2.5. Let x be a point within distance at most Δ_0 of the boundary of K. Consider the generalized cone formed by rays emanating from the center O of K and intersecting $M'(x)$. Consider any Macbeath region $M'(y)$ that overlaps this cone where $\delta(y) = \delta(x)/2$. Then*

(a) $M'(y) \subseteq C^4(x)$, and

(b) There exists a constant c (depending on d and γ) such that $\text{vol}(M(y)) \geq v(x)/c$.

Proof. We claim that $M'(y)$ overlaps $C^2(x)$. By Lemma 2.7, this will imply that $M'(y) \subseteq C^4(x)$ and so will establish (a). To see the claim, consider any ray that emanates from O and intersects both $M'(x)$ and $M'(y)$. Let x' and y' be any two points on this ray that are contained in $M'(x)$ and $M'(y)$, respectively (see Figure 9). Applying Lemma 2.14 to points x and x', we obtain $\delta(x') \geq 4\delta(x)/5$. Applying the same

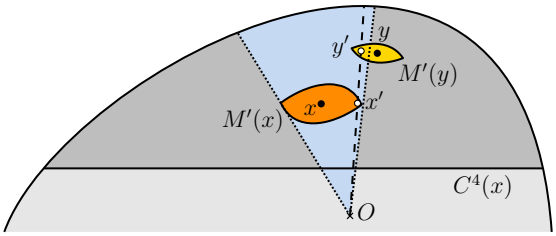


Figure 9: Proof of Lemma 3.2.

lemma to points y and y', we obtain $\delta(y') \leq 4\delta(y)/3$. Recalling that $\delta(y) = \delta(x)/2$ and putting this all together, we obtain

$$\delta(y') \;\leq\; \frac{4}{3} \cdot \delta(y) \;=\; \frac{4}{3} \cdot \frac{\delta(x)}{2} \;\leq\; \frac{4}{3} \cdot \frac{1}{2} \cdot \frac{5}{4} \cdot \delta(x') \;<\; \delta(x').$$

Applying Lemma 2.14 to points x and x', we have $\delta(x') \leq 4\delta(x)/3 \leq 4\Delta_0/3$. Substituting the value of Δ_0, it is easy to verify that $\delta(x') < \gamma/2$. Since $\delta(y') < \delta(x')$, we can now apply Lemma 2.12(b) to conclude that $\text{ray}(y') < \text{ray}(x')$. In other words, y' occurs after x' along the ray emanating from O. Also, by Lemma 2.6, we have $M'(x) \subseteq C^{6/5}(x) \subseteq C^2(x)$. Therefore, $x' \in C^2(x)$, and so $y' \in C^2(x)$. Thus, we have shown that $M'(y)$ intersects $C^2(x)$, which proves (a).

Next we prove (b). Applying Lemma 2.14 to points y and y', we obtain $\delta(y') \geq 4\delta(y)/5$. Recalling that $\delta(x') \leq 4\delta(x)/3$, we have

$$\begin{aligned} \delta(y') \;&\geq\; \frac{4}{5} \cdot \delta(y) \;=\; \frac{4}{5} \cdot \frac{\delta(x)}{2} \\ &\geq\; \frac{4}{5} \cdot \frac{1}{2} \cdot \frac{3}{4} \cdot \delta(x') \;\geq\; \frac{1}{4} \cdot \delta(x'). \end{aligned}$$

Applying Lemma 3.1 to x' and y', we obtain $v(y') \geq (\gamma/4)^d v(x')$.

Applying Lemma 2.15 to x and x', we have $v(x') \geq v(x)/2^d$. Analogously, we have $v(y) \geq v(y')/2^d$. Also, since $\delta(y) = \delta(x)/2 \leq \Delta_0/2 \leq \Delta_0$, the precondition of Lemma 2.5 is satisfied for point y. Applying Lemma 2.5, it follows that $C(y) \subseteq M^{3d}(y)$. Thus

$$\text{vol}(M(y)) \;\geq\; \frac{\text{vol}(C(y))}{(3d)^d} \;=\; \frac{v(y)}{(3d)^d}.$$

Putting it all together, we obtain

$$\begin{aligned} \text{vol}(M(y)) \;&\geq\; \frac{v(y)}{(3d)^d} \;\geq\; \frac{1}{(3d)^d} \cdot \frac{1}{2^d} \cdot v(y') \\ &\geq\; \frac{1}{(3d)^d} \cdot \frac{1}{2^d} \cdot \left(\frac{\gamma}{4}\right)^d v(x') \\ &\geq\; \frac{1}{(3d)^d} \cdot \frac{1}{2^d} \cdot \left(\frac{\gamma}{4}\right)^d \cdot \frac{1}{2^d} \cdot v(x) \\ &\geq\; \left(\frac{\gamma}{48\,d}\right)^d \cdot v(x). \end{aligned}$$

This yields $\text{vol}(M(y)) \geq v(x)/c$ for any constant $c \geq (48\,d/\gamma)^d$, which proves (b). □

The previous lemma implies the following.

Lemma 3.3 *Let $K \subset \mathbb{R}^d$ be a convex body, and let Δ_0 be the constant of Lemma 2.5. Also, let $\lambda \leq 1/5$ be any constant. Let $x \in K$ such that $\delta(x) \leq \Delta_0$. Consider the generalized cone formed by rays emanating from the center O of K and intersecting $M'(x)$. Let Y denote any set of points y such that $\delta(y) = \delta(x)/2$ and the set of Macbeath regions $M^\lambda(y)$ are disjoint. Let $Y' \subseteq Y$ denote the set of points y such that $M'(y)$ overlaps the aforementioned cone. Then $|Y'| = O(1)$.*

Proof. Let y denote any point of Y'. Applying Lemma 3.2, it follows that (a) $M'(y) \subseteq C^4(x)$, and (b) $\text{vol}(M(y)) \geq v(x)/c$, for a suitable constant c. Since $\lambda \leq 1/5$, it follows that $M^\lambda(y)$ is contained in $C^4(x)$. By Lemma 2.3, the volume of $C^4(x)$ is at most $4^d v(x) = O(v(x))$ and the volume of $M^\lambda(y)$ is $\lambda^d \cdot \text{vol}(M(y)) \geq \lambda^d \cdot v(x)/c = \Omega(v(x))$. Since the Macbeath regions $M^\lambda(y)$ for $y \in Y'$ are disjoint, by a straightforward packing argument, it follows that $|Y'| = O(1)$. □

We are now ready to show that the number of children of any non-root node u in our data structure is $O(1)$. (We will analyze the number of children of the root node later. See the remarks following Lemma 3.5.) Consider any node u at level $i \geq 0$. Recall that $E(x_u)$ denotes the associated Macbeath ellipsoid, which encloses $M^{4\lambda_0}(x_u)$. The children of u are those nodes v at level $i+1$ whose ellipsoid $E(x_v)$ intersects the generalized cone formed by rays emanating from the origin that intersect $E(x_u)$. The child condition is expressed in terms of Macbeath ellipsoids (for the sake of efficient query processing), but the above lemma is stated in terms of Macbeath regions.

Since $x_u \in \partial K(\Delta_i)$, we have $\delta(x_u) = \Delta_i \leq \Delta_0$. Macbeath regions are centrally symmetric, and the constant in John's Theorem [38] is $\sqrt{d}$ for centrally symmetric bodies. Recalling that $\lambda_0 = 1/(20\sqrt{d})$ we have

$$M^{4\lambda_0}(x_u) \;\subseteq\; E(x_u) \;\subseteq\; M^{4\lambda_0\sqrt{d}}(x_u) \;=\; M'(x_u). \tag{4}$$

Thus, the generalized cone of rays that intersect $M'(x_u)$ includes all the rays used to define the children of x_u. The points x_v that form level $i+1$ of the structure lie on $\partial K(\Delta_{i+1})$ and thus satisfy $\delta(x_v) = \delta(x_u)/2$. Since by our construction they have disjoint Macbeath regions $M^{\lambda_0}(x_v)$, they constitute a set Y as described in the preconditions of Lemma 3.3. Each child v of u corresponds to a point x_v such that the ellipsoid $E(x_v)$ intersects the generalized cone. Reasoning as we did above for x_u, we have $E(x_v) \subseteq M'(x_v)$. Therefore, the points x_v associated with the children of u constitute a subset of the set Y' given in the lemma. Therefore, the number of children of x_u is $O(1)$, as desired.

3.3 Storage Space.

In this section, we show that the total number of nodes in the data structure is $O(1/\varepsilon^{(d-1)/2})$. Since each node has $O(1)$ children, it will follow that the total storage is also $O(1/\varepsilon^{(d-1)/2})$.

Recall the constants Δ_0 and $\lambda_0 = 1/(20\sqrt{d})$ defined earlier. The number of nodes at level i is bounded above by the cardinality of a maximal set of disjoint Macbeath regions $M^{\lambda_0}(x)$, such that the centers x lie on the boundary of $K(\Delta_i)$, where $\Delta_i = \Delta_0/2^i$. Our analysis will make use of the following lemma, which is a straightforward adaptation of Lemma 3.2, which is proved in the arXiv version of [7].

Lemma 3.4 *Let $K \subset \mathbb{R}^d$ be a convex body in γ-canonical form. Let $0 < \lambda \leq 1/5$ be any fixed constant and let $\Delta \leq \gamma/12$ be a real parameter. Let $\mathcal{C}$ be a set of caps, whose widths lie between Δ and 2Δ, such that the Macbeath regions $M^\lambda(x)$ centered at the centroids x of the bases of these caps are disjoint. Then $|\mathcal{C}| = O(1/\Delta^{(d-1)/2})$.*

We apply this to bound the number of Macbeath regions that define the nodes of each layer.

Lemma 3.5 *Let $K \subset \mathbb{R}^d$ be a convex body in γ-canonical form for some constant γ. Let Δ_0 be the constant of Lemma 2.5 and $0 < \lambda \leq 1/5$ be any fixed constant. Let $\Delta \leq \Delta_0$ be a real parameter. Let $\mathcal{M}$ be a set of disjoint Macbeath regions, each of which has scaling factor λ and whose centers lie on the boundary of $K(\Delta)$. Then $|\mathcal{M}| = O(1/\Delta^{(d-1)/2})$.*

Proof. Let X denote the set of center points of $\mathcal{M}$. By Lemma 2.10, for any $x \in X$, width(x) is between Δ and $(2/\gamma)(3d+1)\Delta$. We can partition X (and by extension $\mathcal{M}$) into $O(1)$ groups such that the points in any group have same width to within a factor of two. Let X' denote one of these groups, and let its associated widths be between w and $2w$. Since $\Delta \leq \Delta_0$, we have $w \leq (2/\gamma)(3d+1)\Delta_0$. Under our assumption that $d \geq 3$, it is easy to verify that the latter quantity does not exceed $\gamma/12$. Thus, the set of caps $C(x)$ for the points of this group satisfy the precondition of Lemma 3.4. Applying this lemma yields $|X'| = O(1/w^{(d-1)/2})$. Summing over all the groups, it follows that the total size of X (and hence the number of regions in $\mathcal{M}$) is $O(1/\Delta^{(d-1)/2})$. □

By Lemma 3.5, the number of nodes at level i is $O(1/\Delta_i^{(d-1)/2}) = O((2^i/\Delta_0)^{(d-1)/2})$. Recall that Δ_0 depends only on d and γ, and both d and γ are constants. It follows that the number of nodes at level

zero is $O(1)$. (This bounds the out-degree of the root node, as alluded to in Section 3.2.) Also, observe that the number of nodes grows geometrically with each level. Therefore, the total number of nodes is dominated by the number of leaves. The leaves are located at level ℓ, where Δ_ℓ is $\Omega(\varepsilon)$. Therefore, the number of leaves, and hence the total number of nodes, is $O(1/\varepsilon^{(d-1)/2})$.

3.4 Query Processing.

Finally, let us present the query algorithm for answering ε-approximate ray-shooting queries. Let Oq denote the query ray. As mentioned earlier, the query algorithm descends the layered DAG structure, visiting a node u at each level such that the associated Macbeath ellipsoid $E(x_u)$ intersects the query ray, until arriving at the leaf level. In order to show that this is well defined, it is necessary to demonstrate that such a node exists at each level of the data structure. Since all the eroded bodies $K(\Delta_i)$ contain the origin, it suffices to show that the union of the Macbeath ellipsoids associated with the nodes of level i cover the boundary of $K(\Delta_i)$. This is established by the following lemma.

Lemma 3.6 *For any $\Delta \le \Delta_0$, let X denote a maximal set of points lying on the boundary of the eroded body $K(\Delta)$ such that the associated Macbeath regions $M^{\lambda_0}(x)$ are pairwise disjoint. Then the collection of Macbeath ellipsoids $\{E(x) \mid x \in X\}$ covers $\partial K(\Delta)$.*

Proof. Consider any point $x' \in \partial K(\Delta)$. Because X is maximal, there must exist $x \in X$ such that $M^{\lambda_0}(x)$ has a nonempty intersection with $M^{\lambda_0}(x')$. By Lemma 2.4, $M^{\lambda_0}(x') \subseteq M^{4\lambda_0}(x)$. Recalling that $M^{4\lambda_0}(x) \subseteq E(x)$, it follows that $x' \in E(x)$. □

Since $\Delta_i \le \Delta_0$ for each level i of the data structure, it follows from the above lemma that the query procedure will succeed in finding a suitable child for each node visited until it reaches the leaf level. Since each node has a constant number of children, it takes $O(\ell) = O(\log \frac{1}{\varepsilon})$ time to perform this descent.

Recall from Section 3.1 that each leaf node u stores a set H_u of at most d supporting hyperplanes of K whose intersection defines an unbounded simplex that contains K (see Figure 10(a)). The query algorithm computes the intersection of the query ray with each of these hyperplanes and returns the closest intersection point p to the origin. The following lemma establishes the correctness of the query processing.

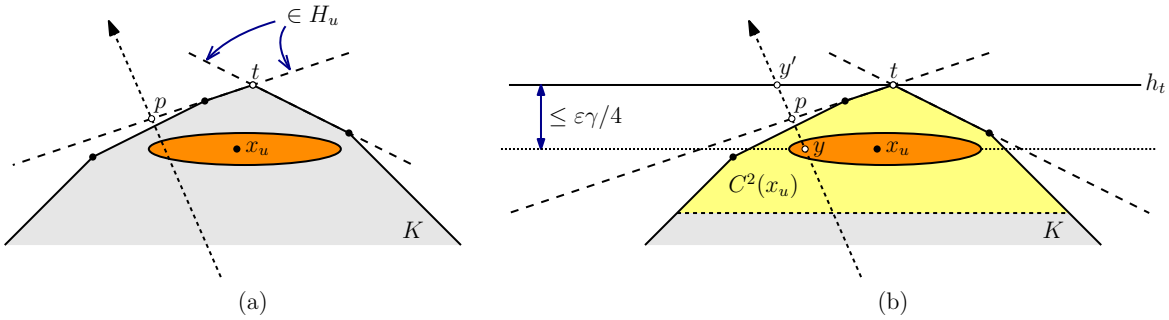


Figure 10: Query processing for a leaf node.

Lemma 3.7 *Given a query ray Oq, the point p returned by the query procedure is a valid answer to the ε-approximate ray-shooting query, and it lies on a supporting hyperplane of K.*

Proof. Observe that p lies at the intersection of the query ray and a supporting hyperplane of K. Clearly, p is not internal to K, so all that remains is to show that p lies within distance ε of K. Recall that a leaf node u satisfies $\delta(x_u) \le \gamma^2\varepsilon/(8(3d+1))$, and therefore by Lemma 2.10, $\text{width}(x_u) \le \gamma\varepsilon/4$. Since the search procedure arrived at node u, the ray Oq intersects $E(x_u)$. By Eq. (4) and Lemma 2.6,

$$E(x_u) \;\subseteq\; M'(x_u) \;\subseteq\; C^{6/5}(x_u) \;\subseteq\; C^2(x_u).$$

Let t denote the apex of $C^2(x_u)$, and let h_t denote the hyperplane passing through t that is parallel to the base of this cap (see Figure 10(b)). By construction, the intersection of the halfspaces H_u associated

with u lies within the halfspace bounded by h_t that contains K. Let y be any point in $E(x_u) \cap Oq$, and let y' denote the intersection of the ray Oq and h_t. By Lemma 2.11, $\|y'y\| \leq 2 \cdot \text{width}(C^2(x_u))/\gamma = 4 \cdot \text{width}(x_u)/\gamma \leq \varepsilon$. Therefore y' lies within distance ε of K, implying that p does as well. □

Summarizing the results of this section, we have shown that, given a convex polytope K in γ-canonical form, where γ is a constant, and given $\varepsilon > 0$, there exists a data structure that uses $O(1/\varepsilon^{(d-1)/2})$ space and answers ε-approximate ray-shooting queries in time $O(\log \frac{1}{\varepsilon})$. This establishes Lemma 2.2, and Theorem 1.1 follows immediately. The following lemma justifies our assertion that these bounds are asymptotically optimal.

Lemma 3.8 *For all sufficiently small $\varepsilon > 0$, any data structure for answering ε-approximate polytope membership queries in $\mathbb{R}^d$ requires $\Omega(1/\varepsilon^{(d-1)/2})$ bits of storage, and if the data structure operates in the decision tree model, the query time is $\Omega(\log \frac{1}{\varepsilon})$ in the worst case.*

Proof. Consider a Euclidean ball of unit diameter in $\mathbb{R}^d$, and let p be any point on the boundary of this ball. For any $0 < \varepsilon < \frac{1}{2}$, it follows from a simple application of the Pythagorean Theorem that a cap of width ε whose apex is at p has diameter at most $c\sqrt{\varepsilon}$, for some constant c depending only on d. By a simple packing argument there exists a set P of points of size $\Omega((1/\sqrt{\varepsilon})^{d-1}) = \Omega(1/\varepsilon^{(d-1)/2})$ on the boundary of the ball such that the ε-width caps centered at these points are pairwise disjoint. For any two distinct subsets P' and P'' of P, consider a point p that lies in one subset, say P', but not in the other. It is easy to see that for the query point p, the answer to the ε-approximate membership query at q is "yes" for P' and "no" for P''. Therefore, the two data structures for these subsets must differ. It follows that there are $2^{|P|}$ distinct data structures needed to represent the various subsets of P. By an information-theoretic argument, such a data structure requires $\Omega(1/\varepsilon^{(d-1)/2})$ bits in the worst case. Assuming that queries are answered in the decision-tree model, such a structure requires depth $\Omega(\log \frac{1}{\varepsilon})$. □

4 Approximate Nearest-Neighbor Searching

In this section we present a reduction from approximate Euclidean nearest-neighbor searching to approximate polytope membership, or more accurately, to approximate ray-shooting. The reduction is based on the approximate Voronoi diagram (AVD) construction from [11]. The AVD for an n-element point set X employs a height-balanced variant of a quadtree, a balanced box decomposition (BBD) tree [12] to be precise. Each leaf cell Q of the tree stores a set $R \subseteq X$ of *representative points*, which have the property that for any query point $q \in Q$, at least one of these representatives is an ε-nearest neighbor of q. We will employ a version of this data structure where the total number of representatives over all the nodes is $O(n \log \frac{1}{\varepsilon})$.

In the data structure of [11] a query is answered by locating the leaf cell that contains the query point in $O(\log n)$ time, and then selecting the nearest representative from this cell to the query (by simple brute force). Later in [4] it was shown that queries can be answered more efficiently by replacing the brute-force search with an approach based on using approximate polytope membership queries. (This will be discussed below.) These membership queries were applied within the context of a binary search in order to simulate approximate ray shooting. In light of Lemma 2.2, we can forgo the binary search, which saves a factor of $O(\log \frac{1}{\varepsilon})$ in the query time.

The approximate ray shooting queries used in [4] were of a different nature than those presented here. First, the rays are vertical (parallel to one of the coordinate axes). Second, the hyperplanes near the portion of the polytope's boundary where the ray might hit are not too sharply sloped with respect to the query ray. (More formally, for any ε-approximating convex polytope P of K, the angle between the vertical ray and the normal vector of the hyperplane of P hit by this ray is bounded away from $\pi/2$ by a constant.) We refer to this as *vertical slope-restricted approximate ray shooting*. The first part of the following result is proved in [4], and the slope-restricted variant follows directly by eliminating the binary search.

Lemma 4.1 *Let $0 < \varepsilon \leq 1/2$ be a real parameter and X be a set of n points in $\mathbb{R}^d$. Given a data structure for approximate polytope membership in d-dimensional space with query time at most $t_d(\varepsilon)$ and storage $s_d(\varepsilon)$, it is possible to preprocess X into an ε-approximate nearest neighbor data structure with*

$$\textit{Query time: } O\left(\log n + t_{d+1}(\varepsilon) \cdot \log \tfrac{1}{\varepsilon}\right) \quad \textit{and}$$

$$\textit{Space: } O\left(n \log \frac{1}{\varepsilon} + n \, \frac{s_{d+1}(\varepsilon)}{t_{d+1}(\varepsilon)}\right).$$

If vertical slope-restricted approximate ray shooting queries are supported, then the query time is $O(\log n + t_{d+1}(\varepsilon))$.

Observe that the space bound varies inversely with the query-time bound. While the query time presented here is $O(\log 1/\varepsilon)$, we can artificially generate higher query times by employing brute-force search. Indeed, this lemma exploits the fact that when brute-force search is used on subsets of size at most $t_{d+1}(\varepsilon)$, the data structure need only be constructed for subsets of size at least $t_{d+1}(\varepsilon)$, of which there are at most $O(n/t_{d+1}(\varepsilon))$.

4.1 Lifting Transformation.

In order to adapt Lemma 4.1 to our context, we will need to understand a bit more about how it works. It is based on a well-known transformation that maps a point in $\mathbb{R}^d$ to $\mathbb{R}^{d+1}$ by projecting it vertically onto a paraboloid. More formally, we can embed a point $p = (x_1, \ldots, x_d)$ in $\mathbb{R}^d$ into $\mathbb{R}^{d+1}$ by adding an additional $(d+1)$st coordinate whose value is zero. Let us visualize the $(d+1)$st coordinate axis as being directed vertically upwards. Let Ψ denote the paraboloid $x_{d+1} = \sum_{i=1}^{d} x_i^2$. Given a point $p \in \mathbb{R}^d$, the *lifting transformation* projects $p \in \mathbb{R}^d$ vertically to a point $p^\uparrow$ lying on Ψ. Define $h(p)$ to be the hyperplane tangent to Ψ at $p^\uparrow$, that is,

$$h(p) \;=\; \left\{(x_1, \ldots, x_{d+1}) \;\middle|\; x_{d+1} = \sum_{i=1}^{d} 2p_i x_i - \|p\|^2\right\}.$$

For any $q \in \mathbb{R}^d$, let q_p denote the point of intersection between $h(p)$ and a vertical ray shot upwards from q. Letting $\|pq\|$ denote the Euclidean distance between points p and q, it is easily verified that $\|q_p q^\uparrow\| = \|pq\|^2$. (See [4] for details.)

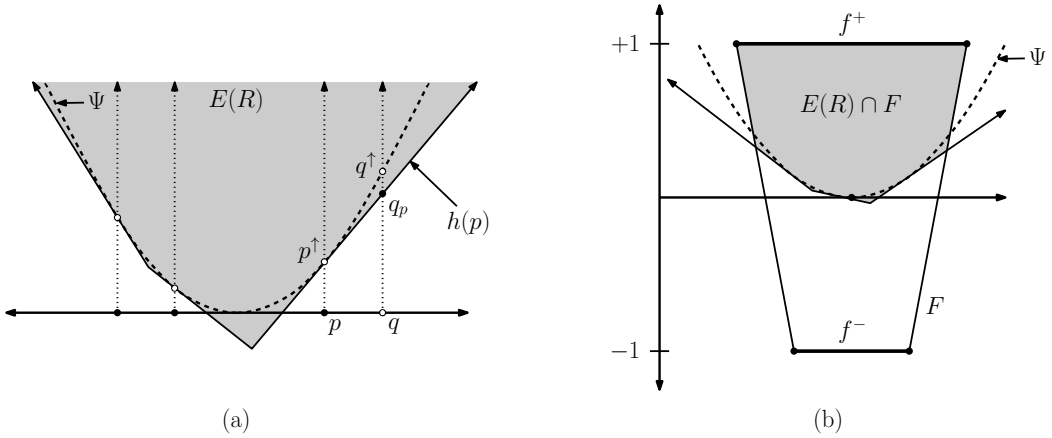


Figure 11: (a) The lifting transformation and (b) the restriction $E(R) \cap F$.

Given a finite point set R in $\mathbb{R}^d$, let $E(R)$ denote the upper envelope of the hyperplanes $h(p)$ for each $p \in R$ (shaded in Figure 11(a)). A vertical line through any point $q \in \mathbb{R}^d$ intersects a facet of $E(R)$. (If the line intersects the boundary between multiple facets, we select one facet arbitrarily.) It follows directly that the nearest neighbor in R of any query point q is the point $p \in R$ whose associated hyperplane $h(p)$ is hit by the vertical line segment passing through q. That is, nearest neighbor queries in $\mathbb{R}^d$ can be reduced to vertical ray-shooting queries against $E(R)$ in $\mathbb{R}^{d+1}$ [2, 33].

While this applies to exact nearest neighbors, it is shown in [4] that the ε-approximate closest representative in R can be determined by simulating vertical ray shooting against a suitable approximation to $E(R)$. In particular, after a normalizing transformation, it can be assumed that the cell Q is centered at the origin, and both Q and the points of R all lie within some constant distance of the origin. The choice of this constant is arbitrary (depending possibly on d but not on ε), and it only affects the constant factors in the query time. We will assume henceforth that this constant is chosen to be $1/2$.

$E(R)$ is unbounded, and it will be necessary to define a bounded polytope that contains the relevant portion of $E(R)$. Because the distance between any point $q \in \mathbb{R}^d$ and its closest representative in R is at most one, it follows that for the sake of answering nearest neighbor queries, the relevant portion of $E(R)$ lies within the region bounded by two horizontal hyperplanes $-1 \leq x_{d+1} \leq +1$. For reasons that will be apparent later, it will be convenient to define this bounded region to be a frustum. Let f^- be the d-dimensional hypercube on the hyperplane $x_{d+1} = -1$ satisfying $-1/2 \leq x_i \leq 1/2$, for $1 \leq i \leq d$, and let f^+ be the d-dimensional hypercube on the hyperplane $x_{d+1} = +1$ satisfying $-5/6 \leq x_i \leq 5/6$, for $1 \leq i \leq d$ (see Figure 11(b)). Let F denote the frustum defined by the convex hull of f^- and f^+. Clearly, the relevant portion of $E(R)$ lies within F, and so we may restrict attention to the polytope $E(R) \cap F$.

In [4] it is shown that after normalization, answering vertical ray-shooting queries approximately with respect to $E(R)$ is sufficient to answer approximate nearest neighbor queries with respect to R. The following lemma restates this result in a manner that is suitable for our context. The proof follows directly from the analysis of [4], but with the constant factors adjusted accordingly.

Lemma 4.2 *Given an AVD cell Q and representative set R that have been normalized as specified above, there exists a positive constant c (depending possibly on d but not on ε) such that following holds. Let R' be any subset of R such that the Hausdorff distance between $E(R') \cap F$ and $E(R) \cap F$ is at most ε/c. Then for any $q \in Q$, if $p' \in R'$ is the defining point of the facet of $E(R')$ that is hit by a vertical line through q, then p' is an ε-approximate nearest neighbor of q within R.*

4.2 From Vertical to Central Ray Shooting.

The principal impediment to applying this result to the polytope membership data structure described in Section 3 is that the ray-shooting used in Lemma 4.2 is vertical, and here it is targeted towards a point at the center of the polytope. In the remainder of this section we will show how to adapt vertical ray shooting to central ray shooting. Our approach involves defining a projective transformation that maps vertical lines to lines passing through a given point.

Before giving the transformation, let us recall some basic facts from projective geometry and homogeneous coordinates. A point $p = (x_1, \ldots, x_{d+1}) \in \mathbb{R}^{d+1}$ can be represented using homogeneous coordinates as a $(d+2)$-vector $[x_0, x_1, \ldots, x_{d+1}]$, where $x_0 = 1$. (We use square brackets for homogeneous coordinates and parentheses for Cartesian coordinates.) Two nonzero homogeneous vectors represent the same point in space if they are equal up to a nonzero scale factor. The point at infinity in the direction given by the nonzero vector $(x_1, \ldots, x_{d+1})$ is represented by the homogeneous coordinates $[0, x_1, \ldots, x_{d+1}]$. Any projective transformation can be defined by applying a linear transformation to the homogeneous coordinates followed by a normalization step in which all the coordinates are divided by the x_0 coordinate (assuming that it is nonzero).

Given a point $p = [x_0, x_1, \ldots, x_{d+1}]$, consider the projective transformation

$$\begin{aligned} T(p) &= [4x_0 + x_{d+1}, 4x_1, \ldots, 4x_d, 2x_{d+1}] \\ &\equiv \left(\frac{4x_1}{4 + x_{d+1}}, \ldots, \frac{4x_d}{4 + x_{d+1}}, \frac{2x_{d+1}}{4 + x_{d+1}} \right). \end{aligned}$$

Let S denote a hypersphere of unit radius that is centered one unit above the origin. Let $p_0 = (0, \ldots, 0, 2)$ denote the topmost point of S. The following lemma states the important properties of T for our purposes.

Lemma 4.3 *The projective transformation T satisfies the following:*

(1) *T maps horizontal hyperplanes to horizontal hyperplanes and it fixes the hyperplane $x_{d+1} = 0$, that is, for any $p \in \mathbb{R}^d$, $T(p) = p$.*

(2) *T maps the point at vertical infinity (having homogeneous coordinates $[0, \ldots, 0, 1]$) to p_0 (having the homogeneous coordinates $[1, 0, \ldots, 0, 2]$). Therefore, the vertical line through any point $p \in \mathbb{R}^d$ is mapped to the line $\overline{pp_0}$ (see Figure 12(a)).*

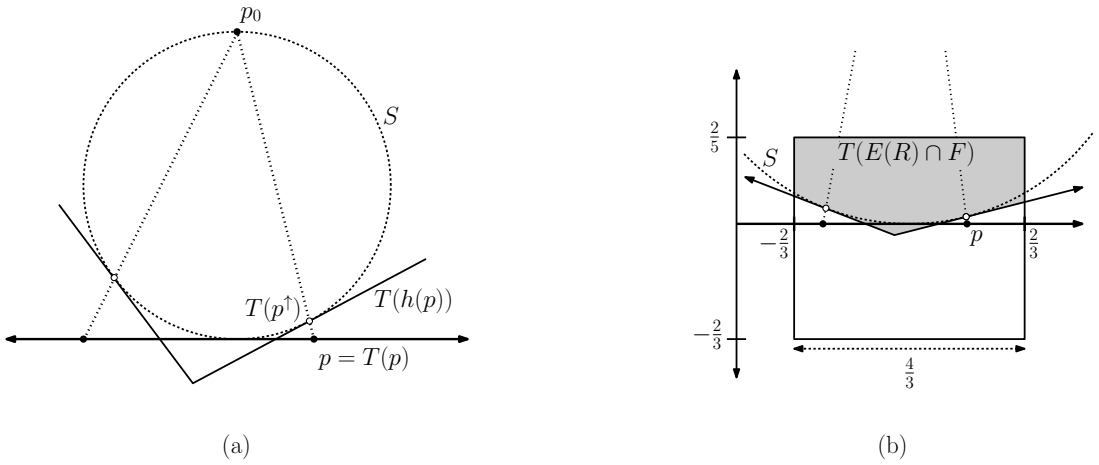
(3) *If $x_{d+1} > -4$, then T preserves the signs of the coordinates of the transformed point.*

(4) *T maps F to an axis-aligned hyperrectangle whose vertical projection is a hypercube of side length $4/3$ centered at the origin and whose vertical extent is $-2/3 \leq x_{d+1} \leq 2/5$ (see Figure 12(b)).*

(5) *T maps the paraboloid Ψ to the punctured sphere $S \setminus \{p_0\}$. Therefore, for any $p \in \mathbb{R}^d$, $T(p^\uparrow)$ is the intersection of the line $\overline{pp_0}$ and $S \setminus \{p_0\}$. Because projective transformations preserve flatness, $T(h(p))$ is the hyperplane tangent to S at this point.*

(6) *The inverse of T is*

$$\begin{aligned} T^{-1}(p) &= \frac{1}{8}[2x_0 - x_{d+1}, 2x_1, \ldots, 2x_d, 4x_{d+1}] \\ &\equiv \left(\frac{2x_1}{2 - x_{d+1}}, \ldots, \frac{2x_d}{2 - x_{d+1}}, \frac{4x_{d+1}}{2 - x_{d+1}}\right). \end{aligned}$$



(a) (b)

Figure 12: The projective transformation and Lemma 4.3.

Proof. Assertions (1)–(3) and (6) are straightforward to verify. Assertion (4) can be verified by transforming the corner points of F, $(\pm 1/2, \ldots, \pm 1/2, -1)$ and $(\pm 5/6, \ldots, \pm 5/6, 1)$. To see assertion (5), observe that the points of S can be described as the zero set of the function

$$\varphi(x_1, \ldots, x_{d+1}) = \sum_{i=1}^{d} x_i^2 + (x_{d+1} - 1)^2 - 1.$$

Let $p = [1, x_1, \ldots, x_{d+1}] \equiv (x_1, \ldots, x_{d+1})$ denote the coordinates of any point on Ψ. Letting $\sigma = \sum_{i=1}^{d} x_i^2$, we have $x_{d+1} = \sigma$. Applying T yields

$$\begin{aligned} T(p) &= [4 + \sigma, 4x_1, \ldots, 4x_d, 2\sigma] \\ &\equiv \left(\frac{4x_1}{4 + \sigma}, \ldots, \frac{4x_d}{4 + \sigma}, \frac{2\sigma}{4 + \sigma}\right). \end{aligned}$$

It is straightforward to verify that $\varphi(T(p)) = 0$ and $\lim_{\sigma \to \infty} T(p) = p_0$. □

Assertion (4) is the reason for defining F in the manner that we did. Projective transformations preserve flatness, and hence $T(E(R) \cap F)$ is a polytope. It follows that for any $q \in Q$, we can compute its exact nearest neighbor in R by determining the lower facet of $T(E(R) \cap F)$ that is hit by the line $\overleftrightarrow{qp_0}$. (There is an obvious connection with the relationship observed by Brown [20] between the stereographic projection and the Voronoi diagram.)

4.3 Preserving Distances.

In order to show that this transformation can be used for approximate nearest neighbor searching, we show that T does not significantly distort the distances between points of interest. In particular, we show that if two points of $T(F)$ are close then their preimages are also close.

Lemma 4.4 *There exists a constant c' such that for any two points $p, q \in T(F)$ such that $\|pq\| \leq 1/4$, $\|T^{-1}(p)T^{-1}(q)\| \leq c'\|pq\|$.*

Proof. Let $\|v\|_\infty$ denote the L_∞ length of a vector v. Consider any two points $p, q \in T(F)$ such that $\|pq\| \leq 1/4$. We can express q as $p + \vec{\delta}$, where $\|\vec{\delta}\|_\infty \leq 1/4$. Let $\delta_\infty = \|\vec{\delta}\|_\infty$. We will show that the L_∞ distance between $T^{-1}(p)$ and $T^{-1}(p + \vec{\delta})$ is at most $c''\delta_\infty$ for some constant c''. It will follow that T^{-1} increases Euclidean distances for the points of interest by a factor of at most $c' = c''(d+1)$.

In order to establish the above assertion, let $p = (x_1, \ldots, x_{d+1})$ and let $\vec{\delta} = (\delta_1, \ldots, \delta_{d+1})$. We begin with the following easy inequalities. Given $1 \leq i \leq d+1$, by our bounds on p and $\vec{\delta}$ we have

$$\begin{aligned} (i)\ |x_i + \delta_i| &< 1 \\ (ii)\ 2 - x_{d+1} &> 1 \\ (iii)\ 2 - x_{d+1} - \delta_{d+1} &> 1. \end{aligned} \tag{5}$$

(The worst case for the first inequality arises when $x_i = 2/3$ and $\delta_i = 1/4$, and the worst case for the second and third inequalities occur when $x_{d+1} = 2/3$ and $\delta_{d+1} = 1/4$.) We will also make use of the identity $a/(b-c) = a/b + ac/b(b-c)$, assuming b and $b - c$ are both nonzero.

Consider the transformed point $T^{-1}(q) = T^{-1}(p + \vec{\delta})$. By applying Lemma 4.3(6) and the above identity, for $1 \leq i \leq d$, we find that the ith coordinate is mapped to

$$\begin{aligned} \frac{2(x_i + \delta_i)}{2 - (x_{d+1} + \delta_{d+1})} &= \frac{2(x_i + \delta_i)}{(2 - x_{d+1}) - \delta_{d+1}} \\ &= \frac{2(x_i + \delta_i)}{2 - x_{d+1}} + \frac{2(x_i + \delta_i)\delta_{d+1}}{(2 - x_{d+1})(2 - x_{d+1} - \delta_{d+1})} \\ &= \frac{2x_i}{2 - x_{d+1}} + \frac{2\delta_i}{2 - x_{d+1}} + \frac{2(x_i + \delta_i)\delta_{d+1}}{(2 - x_{d+1})(2 - x_{d+1} - \delta_{d+1})}. \end{aligned}$$

After some expansion, this is equal to

$$\frac{2x_i}{2 - x_{d+1}} + \frac{2\delta_i}{2 - x_{d+1}} + \frac{2(x_i + \delta_i)\delta_{d+1}}{(2 - x_{d+1})(2 - x_{d+1} - \delta_{d+1})}.$$

The first term is the ith coordinate of $T^{-1}(p)$. By Eq. (5), the second term has absolute value at most $2\delta_\infty$. The third term has absolute value at most $2\delta_\infty$. Therefore the ith coordinate of $T^{-1}(p+\vec{\delta})$ is within distance $4\delta_\infty$ of the corresponding coordinate of $T^{-1}(p)$. By applying a similar analysis to the $(d+1)$st coordinate of $T^{-1}(p + \vec{\delta})$, it follows that this coordinate is within distance $8\delta_\infty$ of the corresponding coordinate of $T^{-1}(p)$. Therefore, by setting $c'' = 8$, it follows that the L_∞ distance between $T^{-1}(p)$ and $T^{-1}(p + \vec{\delta})$ is at most $c''\delta_\infty$. □

By combining Lemmas 4.2 and 4.4, it follows that in order to answer ε-approximate nearest neighbor queries in $\mathbb{R}^d$ for an AVD leaf cell Q and a set R of representatives, it suffices to first apply the normalizing transformation to Q, construct an approximate ray-shooting data structure for $T(E(R) \cap F)$ that answers queries to within an absolute error of $\varepsilon' = \varepsilon/cc'$, where c and c' are the constant factors of these respective lemmas. It follows from Lemma 4.4 that the result is an absolute (ε/c)-approximation to the corresponding vertical ray shooting query in $T^{-1}(T(E(R) \cap F)) = E(R) \cap F$. From Lemma 4.2 such an approximation suffices to answer ε-approximate nearest neighbor queries.

In order to apply the results of Section 3, we require that the polytope in question be in γ-canonical form for a suitable constant γ and that rays be directed towards the origin. To do this, we modify $T(E(R) \cap F)$. Recall that it is contained within an axis-aligned hyperrectangle whose topmost facet is at $x_{d+1} = 2/5$. We move this topmost facet up to $x_{d+1} = 8/3$ (see Figure 13). The point p_0 lies within

the interior of the modified polytope. Further a ball of radius $2/3$ centered at p_0 is contained entirely within this polytope, and the polytope is completely contained within a ball of radius less than $3 + d$. By translating this modified polytope so that p_0 coincides with the origin, the result is in γ-canonical form for $\gamma > 2/3(3 + d)$.

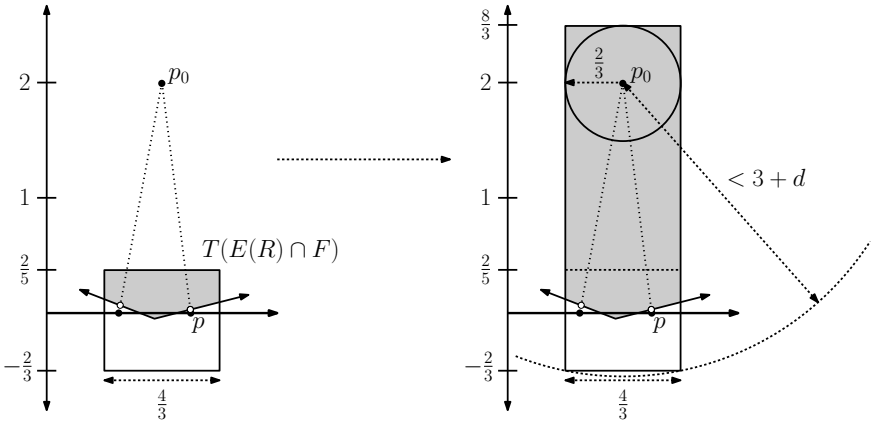


Figure 13: The modified polytope.

Now, the data structure described in Section 3 can be applied to the modified polytope. The witness hyperplane (as described in Section 3.1) that is hit by the ray provides the identity of the desired nearest-neighbor representative, that is, the approximate nearest neighbor of the query point.

Given a parameter m such that

$$\log \frac{1}{\varepsilon} \;\leq\; m \;\leq\; \frac{1}{\varepsilon^{d/2} \log \frac{1}{\varepsilon}},$$

we set $t_{d+1}(\varepsilon) = 1/(m \cdot \varepsilon^{d/2})$ and $s_{d+1}(\varepsilon) = 1/\varepsilon^{d/2}$. Note that for m in this range we have $t_{d+1}(\varepsilon) \geq \log \frac{1}{\varepsilon}$ and so our data structure can achieve this query time for ε-approximate ray-shooting queries. By the results of this section, these bounds apply to vertical slope-restricted approximate ray shooting queries as well. By applying Lemma 4.1 we obtain a data structure for approximate Euclidean nearest-neighbor searching with query time $O(\log n + 1/(m \cdot \varepsilon^{d/2}))$ and space $O(n \log \frac{1}{\varepsilon} + nm) = O(nm)$. This establishes Theorem 1.2.

References

[1] P. K. Agarwal, S. Har-Peled, and K. R. Varadarajan. Approximating extent measures of points. *J. Assoc. Comput. Mach.*, 51:606–635, 2004.

[2] P. K. Agarwal and J. Matoušek. Ray shooting and parametric search. *SIAM J. Comput.*, 22(4):794–806, 1993.

[3] S. Arya, G. D. da Fonseca, and D. M. Mount. A unified approach to approximate proximity searching. In *Proc. 18th Annu. European Sympos. Algorithms*, pages 374–385, 2010.

[4] S. Arya, G. D. da Fonseca, and D. M. Mount. Approximate polytope membership queries. In *Proc. 43rd Annu. ACM Sympos. Theory Comput.*, pages 579–586, 2011. (Full version available from http://arxiv.org/abs/1604.01183).

[5] S. Arya, G. D. da Fonseca, and D. M. Mount. Optimal area-sensitive bounds for polytope approximation. In *Proc. 28th Annu. Sympos. Comput. Geom.*, pages 363–372, 2012.

[6] S. Arya, G. D. da Fonseca, and D. M. Mount. Polytope approximation and the Mahler volume. In *Proc. 23rd Annu. ACM-SIAM Sympos. Discrete Algorithms*, pages 29–42, 2012.

[7] S. Arya, G. D. da Fonseca, and D. M. Mount. On the combinatorial complexity of approximating polytopes. In *Proc. 32nd Internat. Sympos. Comput. Geom.*, pages 11:1–11:15, 2016. (Expanded version in http://arxiv.org/abs/1604.01175).

[8] S. Arya and T. Malamatos. Linear-size approximate Voronoi diagrams. In *Proc. 13th Annu. ACM-SIAM Sympos. Discrete Algorithms*, pages 147–155, 2002.

[9] S. Arya, T. Malamatos, and D. M. Mount. Space-efficient approximate Voronoi diagrams. In *Proc. 34th Annu. ACM Sympos. Theory Comput.*, pages 721–730, 2002.

[10] S. Arya, T. Malamatos, and D. M. Mount. The effect of corners on the complexity of approximate range searching. *Discrete Comput. Geom.*, 41:398–443, 2009.

[11] S. Arya, T. Malamatos, and D. M. Mount. Space-time tradeoffs for approximate nearest neighbor searching. *J. Assoc. Comput. Mach.*, 57:1–54, 2009.

[12] S. Arya and D. M. Mount. Approximate range searching. *Comput. Geom. Theory Appl.*, 17:135–163, 2000.

[13] S. Arya, D. M. Mount, N. S. Netanyahu, R. Silverman, and A. Y. Wu. An optimal algorithm for approximate nearest neighbor searching fixed dimensions. *J. Assoc. Comput. Mach.*, 45(6):891–923, 1998.

[14] S. Arya, D. M. Mount, and J. Xia. Tight lower bounds for halfspace range searching. *Discrete Comput. Geom.*, 47:711–730, 2012.

[15] I. Bárány. The technique of M-regions and cap-coverings: A survey. *Rend. Circ. Mat. Palermo*, 65:21–38, 2000.

[16] L. Barba and S. Langerman. Optimal detection of intersections between convex polyhedra. In *Proc. 26th Annu. ACM-SIAM Sympos. Discrete Algorithms*, pages 1641–1654, 2015.

[17] J. L. Bentley, M. G. Faust, and F. P. Preparata. Approximation algorithms for convex hulls. *Commun. ACM*, 25(1):64–68, 1982.

[18] S. N. Bespamyatnikh. Dynamic algorithms for approximate neighbor searching. In *Proc. Eighth Canad. Conf. Comput. Geom.*, pages 252–257, 1996.

[19] H. Brönnimann, B. Chazelle, and J. Pach. How hard is halfspace range searching. *Discrete Comput. Geom.*, 10:143–155, 1993.

[20] K. Q. Brown. Voronoi diagrams from convex hulls. *Inform. Process. Lett.*, 9:223–228, 1979.

[21] C. J. C. Burges. A tutorial on support vector machines for pattern recognition. *Data Min. Knowl. Discov.*, 2(2):121–167, 1998.

[22] T. M. Chan. Fixed-dimensional linear programming queries made easy. In *Proc. 12th Annu. Sympos. Comput. Geom.*, pages 284–290, 1996.

[23] T. M. Chan. Output-sensitive results on convex hulls, extreme points, and related problems. *Discrete Comput. Geom.*, 16:369–387, 1996.

[24] T. M. Chan. Approximate nearest neighbor queries revisited. *Discrete Comput. Geom.*, 20:359–373, 1998.

[25] T. M. Chan. Closest-point problems simplified on the RAM. In *Proc. 13th Annu. ACM-SIAM Sympos. Discrete Algorithms*, pages 472–473, 2002.

[26] T. M. Chan. Optimal partition trees. In *Proc. 26th Annu. Sympos. Comput. Geom.*, pages 1–10, 2010.

[27] B. Chazelle and D. P. Dobkin. Intersection of convex objects in two and three dimensions. *J. Assoc. Comput. Mach.*, 34:1–27, 1987.

[28] K. L. Clarkson. An algorithm for approximate closest-point queries. In *Proc. Tenth Annu. Sympos. Comput. Geom.*, pages 160–164, 1994.

[29] M. de Berg, O. Cheong, M. van Kreveld, and M. Overmars. *Computational Geometry: Algorithms and Applications*. Springer, 3rd edition, 2010.

[30] D. P. Dobkin and D. G. Kirkpatrick. Fast detection of polyhedral intersection. *Theo. Comp. Sci.*, 27:241–253, 1983.

[31] R. M. Dudley. Metric entropy of some classes of sets with differentiable boundaries. *J. Approx. Theory*, 10(3):227–236, 1974.

[32] C. A. Duncan, M. T. Goodrich, and S. Kobourov. Balanced aspect ratio trees: Combining the advantages of k-d trees and octrees. *J. Algorithms*, 38:303–333, 2001.

[33] H. Edelsbrunner. *Algorithms in Combinatorial Geometry*. Springer-Verlag, 1987.

[34] H. G. Eggleston. *Convexity*. Cambridge University Press, 1958.

[35] J. Erickson, L. J. Guibas, J. Stolfi, and L. Zhang. Separation-sensitive collision detection for convex objects. In *Proc. Tenth Annu. ACM-SIAM Sympos. Discrete Algorithms*, pages 327–336, 1999.

[36] G. Ewald, D. G. Larman, and C. A. Rogers. The directions of the line segments and of the r-dimensional balls on the boundary of a convex body in Euclidean space. *Mathematika*, 17:1–20, 1970.

[37] S. Har-Peled. A replacement for Voronoi diagrams of near linear size. In *Proc. 42nd Annu. IEEE Sympos. Found. Comput. Sci.*, pages 94–103, 2001.

[38] F. John. Extremum problems with inequalities as subsidiary conditions. In *Studies and Essays Presented to R. Courant on his 60th Birthday*, pages 187–204. Interscience Publishers, Inc., New York, 1948.

[39] A. M. Macbeath. A theorem on non-homogeneous lattices. *Ann. of Math.*, 56:269–293, 1952.

[40] J. Matoušek and O. Schwarzkopf. On ray shooting in convex polytopes. *Discrete Comput. Geom.*, 10:215–232, 1993.

[41] J. Matoušek. Reporting points in halfspaces. *Comput. Geom. Theory Appl.*, 2:169–186, 1992.

[42] J. Matoušek. Linear optimization queries. *J. Algorithms*, 14(3):432–448, 1993.

[43] E. A. Ramos. Linear programming queries revisited. In *Proc. 16th Annu. Sympos. Comput. Geom.*, pages 176–181, 2000.

[44] Y. Sabharwal, S. Sen, and N. Sharma. Nearest neighbors search using point location in balls with applications to approximate Voronoi decompositions. *J. Comput. Sys. Sci.*, 72:955–977, 2006.

A Appendix

For the sake of completeness, we give a proof of Lemma 2.4.

Lemma 2.4 *Let K be a convex body, and let $\lambda \leq 1/5$ be any real. If $x, y \in K$ such that $M^\lambda(x) \cap M^\lambda(y) \neq \emptyset$, then $M^\lambda(y) \subseteq M^{4\lambda}(x)$.*

Proof. Let z be a point in the intersection of $M^\lambda(x)$ and $M^\lambda(y)$. Then we can write z as:

$$z = x + \lambda(x - p_1) = y + \lambda(p_2 - y),$$

where $p_1, p_2 \in K$. Equating the two expressions for z above, we obtain

$$y = \frac{(1+\lambda)x - \lambda p_1 - \lambda p_2}{1 - \lambda}.$$

Consider any point $w \in M^\lambda(y)$. We have

$$w = y + \lambda(y - p_3) = (1+\lambda)y - \lambda p_3,$$

where $p_3 \in K$. Substituting the expression obtained above for y, we have

$$w = \frac{(1+\lambda)((1+\lambda)x - \lambda p_1 - \lambda p_2)}{1-\lambda} - \lambda p_3,$$

which simplifies to

$$w = x + \frac{\lambda(3+\lambda)}{1-\lambda}(x - p),$$

where

$$p = \frac{1+\lambda}{3+\lambda}p_1 + \frac{1+\lambda}{3+\lambda}p_2 + \frac{1-\lambda}{3+\lambda}p_3.$$

As p is a convex combination of p_1, p_2 and p_3, $p \in K$. Thus, we have shown that

$$M^\lambda(y) \;\subseteq\; x + \frac{\lambda(3+\lambda)}{1-\lambda}(x - K). \tag{6}$$

In an analogous manner, we next show that

$$M^\lambda(y) \;\subseteq\; x + \frac{\lambda(3+\lambda)}{1-\lambda}(K - x). \tag{7}$$

Again, let z be any point in the intersection of $M^\lambda(x)$ and $M^\lambda(y)$. We can write z as:

$$z = x + \lambda(k_1' - x) = y + \lambda(y - k_2'),$$

where $k_1', k_2' \in K$. Equating the two expressions for z above, we obtain

$$y = \frac{(1-\lambda)x + \lambda k_1' + \lambda k_2'}{1+\lambda}.$$

Consider any point $w \in M^\lambda(y)$. We have

$$w = y + \lambda(k_3' - y) = (1-\lambda)y + \lambda k_3',$$

where $k_3' \in K$. Substituting the expression obtained above for y, we have

$$w = \frac{(1-\lambda)((1-\lambda)x + \lambda k_1' + \lambda k_2')}{1+\lambda} + \lambda k_3',$$

which simplifies to

$$w = x + \frac{\lambda(3-\lambda)}{1+\lambda}(p' - x),$$

where

$$p' = \frac{1-\lambda}{3-\lambda}k_1' + \frac{1-\lambda}{3-\lambda}k_2' + \frac{1+\lambda}{3-\lambda}k_3'.$$

As p' is a convex combination of k_1', k_2' and k_3', $p' \in K$. Letting p'' denote the point on segment xp' such that

$$\frac{\lambda(3-\lambda)}{1+\lambda}(p' - x) = \frac{\lambda(3+\lambda)}{1-\lambda}(p'' - x),$$

we can write

$$w = x + \frac{\lambda(3+\lambda)}{1-\lambda}(p'' - x),$$

where $p'' \in K$. Thus,

$$M^\lambda(y) \;\subseteq\; x + \frac{\lambda(3+\lambda)}{1-\lambda}(K - x),$$

which establishes Eq. (7). By combining this with Eq. (6), we obtain $M^\lambda(y) \subseteq M(x, \lambda(3+\lambda)/(1-\lambda))$. Since $\lambda \leq 1/5$, it is easy to see that $(3+\lambda)/(1-\lambda) \leq 4$. Thus $M^\lambda(y) \subseteq M(x, 4\lambda)$, completing the proof. □

Attached Article 3

Next, we include the expanded arXiv version of the following conference paper.

S. Arya, G. D. da Fonseca, and D. M. Mount. Near-optimal ε-kernel construction and related problems. In *Proceedings of the 33rd International Symposium on Computational Geometry*, pages 10:1–15, 2017.

http://arxiv.org/abs/1703.10868

Near-Optimal ε-Kernel Construction and Related Problems

Sunil Arya*
Department of Computer Science and Engineering
The Hong Kong University of Science and Technology
Clear Water Bay, Kowloon, Hong Kong
arya@cse.ust.hk

Guilherme D. da Fonseca
Université Clermont Auvergne and LIMOS
Clermont-Ferrand, France
fonseca@isima.fr

David M. Mount†
Department of Computer Science and Institute for Advanced Computer Studies
University of Maryland
College Park, Maryland 20742
mount@cs.umd.edu

Abstract

The computation of (i) ε-kernels, (ii) approximate diameter, and (iii) approximate bichromatic closest pair are fundamental problems in geometric approximation. In this paper, we describe new algorithms that offer significant improvements to their running times. In each case the input is a set of n points in $\mathbb{R}^d$ for a constant dimension $d \geq 3$ and an approximation parameter $\varepsilon > 0$. We reduce the respective running times
(i) from $O((n + 1/\varepsilon^{d-2}) \log \frac{1}{\varepsilon})$ to $O(n \log \frac{1}{\varepsilon} + 1/\varepsilon^{(d-1)/2+\alpha})$,
(ii) from $O((n + 1/\varepsilon^{d-2}) \log \frac{1}{\varepsilon})$ to $O(n \log \frac{1}{\varepsilon} + 1/\varepsilon^{(d-1)/2+\alpha})$, and
(iii) from $O(n/\varepsilon^{d/3})$ to $O(n/\varepsilon^{d/4+\alpha})$,
for an arbitrarily small constant $\alpha > 0$. Result (i) is nearly optimal since the size of the output ε-kernel is $\Theta(1/\varepsilon^{(d-1)/2})$ in the worst case.

These results are all based on an efficient decomposition of a convex body using a hierarchy of Macbeath regions, and contrast to previous solutions that decompose space using quadtrees and grids. By further application of these techniques, we also show that it is possible to obtain near-optimal preprocessing time for the most efficient data structures to approximately answer queries for (iv) nearest-neighbor searching, (v) directional width, and (vi) polytope membership.

1 Introduction

In this paper we present new faster algorithms to several fundamental geometric approximation problems involving point sets in d-dimensional space. In particular, we present approximation algorithms for ε-kernels, diameter, bichromatic closest pair, and the minimum bottleneck spanning tree. Our results arise from a recently developed shape-sensitive approach to approximating convex bodies, which is based on the classical concept of Macbeath regions. This approach has been applied to computing area-sensitive bounds for polytope approximation [6], polytope approximations with low combinatorial complexity [7], answering approximate polytope-membership queries [8], and approximate nearest-neighbor searching [8]. The results of [8] demonstrated the existence of data structures for these query problems but did not discuss preprocessing in detail. We complete the story by presenting efficient algorithms for building data structures for three related queries: approximate polytope membership, approximate directional width, and approximate nearest-neighbors.

*Research supported by the Research Grants Council of Hong Kong, China under project number 16200014.
†Research supported by NSF grant CCF–1618866.

Throughout, we assume that the dimension d is a constant. Our running times will often involve expressions of the form $1/\varepsilon^{\alpha}$. In such cases, $\alpha > 0$ is constant that can be made arbitrarily small. The approximation parameter ε is treated as an asymptotic variable that approaches 0. We assume throughout that $\varepsilon < 1$, which guarantees that $\log \frac{1}{\varepsilon} > 0$.

In Section 1.1, we present our results for ε-kernels, diameter, bichromatic closest pair, and minimum bottleneck tree. In Section 1.2, we present our results for the data structure problems. In Section 1.3, we give an overview of the techniques used.

Concurrently and independently, Timothy Chan has reported complexity bounds that are very similar to our results [20]. Remarkably, the computational techniques seem to be very different, based on Chebyshev polynomials.

1.1 Static Results

Kernel. Given a set S of n points in $\mathbb{R}^d$ and an approximation parameter $\varepsilon > 0$, an *ε-coreset* is an (ideally small) subset of S that approximates some measure over S (see [2] for a survey). Given a nonzero vector $v \in \mathbb{R}^d$, the *directional width* of S in direction v, $\text{width}_v(S)$ is the minimum distance between two hyperplanes that enclose S and are orthogonal to v. A *coreset for the directional width* (also known as an *ε-kernel* and as a *coreset for the extent measure*) is a subset $Q \subseteq S$ such that $\text{width}_v(Q) \geq (1-\varepsilon)\,\text{width}_v(S)$, for all $v \in \mathbb{R}^d$. Kernels are among the most fundamental constructions in geometric approximation, playing a role similar to that of convex hulls in exact computations. Kernels have been used to obtain approximation algorithms to several problems such as diameter, minimum width, convex hull volume, minimum enclosing cylinder, minimum enclosing annulus, and minimum-width cylindrical shell [1, 2].

The concept of ε-kernels was introduced by Agarwal et al. [1]. The existence of ε-kernels with $O(1/\varepsilon^{(d-1)/2})$ points is implied in the works of Dudley [22] and Bronshteyn and Ivanov [18], and this is known to be optimal in the worst case. Agarwal et al. [1] demonstrated how to compute such a kernel in $O(n + 1/\varepsilon^{3(d-1)/2})$ time, which reduces to $O(n)$ when $n = \Omega(1/\varepsilon^{3(d-1)/2})$. While less succinct ε-kernels with $O(1/\varepsilon^{d-1})$ points can be constructed in time $O(n)$ time for all n [1, 16], no linear-time algorithm is known to build an ε-kernel of optimal size. Hereafter, we use the term ε-kernel to refer exclusively to an ε-kernel of size $O(1/\varepsilon^{(d-1)/2})$.

Chan [19] showed that an ε-kernel can be constructed in $O((n + 1/\varepsilon^{d-2}) \log \frac{1}{\varepsilon})$ time, which is nearly linear when $n = \Omega(1/\varepsilon^{d-2})$. He posed the open problem of obtaining a faster algorithm. A decade later, Arya and Chan [4] showed how to build an ε-kernel in roughly $O(n + \sqrt{n}/\varepsilon^{d/2})$ time using discrete Voronoi diagrams. In this paper, we attain the following near-optimal construction time.

Theorem 1.1. *Given a set S of n points in $\mathbb{R}^d$ and an approximation parameter $\varepsilon > 0$, it is possible to construct an ε-kernel of S with $O(1/\varepsilon^{(d-1)/2})$ points in $O(n \log \frac{1}{\varepsilon} + 1/\varepsilon^{(d-1)/2+\alpha})$ time, where α is an arbitrarily small positive constant.*

We note that when $n = o(1/\varepsilon^{(d-1)/2})$, the input S is already an ε-kernel and therefore an $O(n)$ time algorithm is trivial. Because the worst-case output size is $O(1/\varepsilon^{(d-1)/2})$, we may assume that n is at least this large, for otherwise we can simply take S itself to be the kernel. Since $1/\varepsilon^{\alpha}$ dominates $\log \frac{1}{\varepsilon}$, the above running time can be expressed as $O(n/\varepsilon^{\alpha})$, which is nearly linear given that α can be made arbitrarily small.

Diameter. An important application of ε-kernels is to approximate the diameter of a point set. Given n data points, the *diameter* is defined to be the maximum distance between any two data points. An *ε-approximation* of the diameter is a pair of points whose distance is at least $(1-\varepsilon)$ times the exact diameter. There are multiple algorithms to approximate the diameter [1, 3, 4, 15, 19]. The fastest running times are $O((n + 1/\varepsilon^{d-2}) \log \frac{1}{\varepsilon})$ [19] and roughly $O(n + \sqrt{n}/\varepsilon^{d/2})$ [4]. The algorithm from [19] essentially computes an ε-kernel Q and then determines the maximum value of $\text{width}_v(Q)$ among a set of $k = O(1/\varepsilon^{(d-1)/2})$ directions v by brute force [1]. Discrete Voronoi diagrams [4] permit this computation in roughly $O(n + \sqrt{n}/\varepsilon^{d/2})$ time. Therefore, combining the kernel construction of Theorem 1.1 with discrete Voronoi diagrams [4], we reduce n to $O(1/\varepsilon^{(d-1)/2})$ and obtain an algorithm to ε-approximate the diameter in roughly $O(n + 1/\varepsilon^{3d/4})$ time. However, we show that it is possible to obtain a much faster algorithm, as presented in the following theorem.

Theorem 1.2. *Given a set S of n points in $\mathbb{R}^d$ and an approximation parameter $\varepsilon > 0$, it is possible to compute an ε-approximation to the diameter of S in $O(n \log \frac{1}{\varepsilon} + 1/\varepsilon^{(d-1)/2+\alpha})$ time.*

Bichromatic Closest Pair. In the *bichromatic closest pair* (BCP) problem, we are given n points from two sets, designated red and blue, and we want to find the closest red-blue pair. In the *ε-approximate* version, the goal is to find a red-blue pair of points whose distance is at most $(1 + \varepsilon)$ times the exact BCP distance. Approximations to the BCP problem were introduced in [26], and the most efficient randomized approximation algorithm runs in roughly $O(n/\varepsilon^{d/3})$ expected time [4]. We present the following result.

Theorem 1.3. *Given n red and blue points in $\mathbb{R}^d$ and an approximation parameter $\varepsilon > 0$, there is a randomized algorithm that computes an ε-approximation to the bichromatic closest pair in $O(n/\varepsilon^{d/4+\alpha})$ expected time.*

Euclidean Trees. Given a set S of n points in $\mathbb{R}^d$, a *Euclidean minimum spanning tree* is the spanning tree with vertex set S that minimizes the sum of the edge lengths, while a *Euclidean minimum bottleneck tree* minimizes the maximum edge length. In the approximate version we respectively approximate the sum and the maximum of the edge lengths. A minimum spanning tree is a minimum bottleneck tree (although the converse does not hold). However, an approximation to the minimum spanning tree is not necessarily an approximation to the minimum bottleneck tree. A recent approximation algorithm to the Euclidean minimum spanning tree takes roughly $O(n \log n + n/\varepsilon^2)$ time, regardless of the (constant) dimension [11]. On the other hand, the fastest algorithm to approximate the minimum bottleneck tree takes roughly $O((n \log n)/\varepsilon^{d/3})$ expected time [4].The algorithm uses BCP to simultaneously attain an approximation to the minimum bottleneck and the minimum spanning trees. We prove the following theorem.

Theorem 1.4. *Given n points in $\mathbb{R}^d$ and an approximation parameter $\varepsilon > 0$, there is a randomized algorithm that computes a tree T that is an ε-approximation to both the Euclidean minimum bottleneck and the Euclidean minimum spanning trees in $O((n \log n)/\varepsilon^{d/4+\alpha})$ expected time.*

1.2 Data Structure Results

Polytope membership. Let P denote a convex polytope in $\mathbb{R}^d$, represented as the bounded intersection of n halfspaces. The *polytope membership problem* consists of preprocessing P so that it is possible to determine efficiently whether a given query point $q \in \mathbb{R}^d$ lies within P. In the ε-approximate version, we consider an expanded convex body $K \supset P$. A natural way to define this expansion would be to consider the set of points that lie within distance $\varepsilon \cdot \mathrm{diam}(P)$ of P, thus defining a body whose Hausdorff distance from P is $\varepsilon \cdot \mathrm{diam}(P)$. However, this definition has the shortcoming that it is not sensitive to the directional width of P. Instead, we define K as follows. For any nonzero vector $v \in \mathbb{R}^d$, consider the two supporting hyperplanes for P that are normal to v. Translate each of these hyperplanes outward by a distance of $\varepsilon \cdot \mathrm{width}_v(P)$, and consider the closed slab-like region lying between them. Define K to be the intersection of this (infinite) set of slabs. This is clearly a stronger approximation than the Hausdorff-based definition. An *ε-approximate polytope membership query* (ε-APM query) returns a positive result if the query point q is inside P, a negative result if q is outside K, and may return either result otherwise.[1]

We recently proposed an optimal data structure to answer approximate polytope membership queries, but efficient preprocessing remained an open problem [8]. In this paper, we present a similar data structure that not only attains optimal storage and query time, but can also be preprocessed in near-optimal time.

Theorem 1.5. *Given a convex polytope P in $\mathbb{R}^d$ represented as the intersection of n halfspaces and an approximation parameter $\varepsilon > 0$, there is a data structure that can answer ε-approximate polytope membership queries with*

$$\textit{Query time: } O\left(\log \frac{1}{\varepsilon}\right) \quad \textit{Space: } O\left(1/\varepsilon^{\frac{d-1}{2}}\right) \quad \textit{Preprocessing: } O\left(n \log \frac{1}{\varepsilon} + \left(\frac{1}{\varepsilon}\right)^{\frac{d-1}{2}+\alpha}\right).$$

[1] Our earlier works on ε-APM queries [5, 8] use the weaker Hausdorff form to define the problem, but the solutions presented there actually achieve the stronger direction-sensitive form.

Directional width. Applying the previous data structure in the dual space, we obtain a data structure for the following *ε-approximate directional width* problem, which is closely related to ε-kernels. Given a set S of n points in a constant dimension d and an approximation parameter $\varepsilon > 0$, the goal is to preprocess S to efficiently ε-approximate $\text{width}_v(S)$, for a nonzero query vector v. We present the following result.

Theorem 1.6. *Given a set S of n points in $\mathbb{R}^d$ and an approximation parameter $\varepsilon > 0$, there is a data structure that can answer ε-approximate directional width queries with*

$$\textit{Query time: } O\left(\log^2 \frac{1}{\varepsilon}\right) \textit{ Space: } O\left(1/\varepsilon^{\frac{d-1}{2}}\right) \textit{ Preprocessing: } O\left(n \log \frac{1}{\varepsilon} + \left(\frac{1}{\varepsilon}\right)^{\frac{d-1}{2}+\alpha}\right).$$

Nearest Neighbor. Let S be a set of n points in $\mathbb{R}^d$. Given any $q \in \mathbb{R}^d$, an *ε-approximate nearest neighbor* (ANN) of q is any point of S whose distance from q is at most $(1+\varepsilon)$ times the distance to q's closest point in S. The objective is to preprocess S in order to answer such queries efficiently. Data structures for approximate nearest neighbor searching (in fixed dimensions) have been proposed by several authors, offering space-time tradeoffs (see [8] for an overview of the tradeoffs). Applying the reduction from approximate nearest neighbor to approximate polytope membership established in [5] together with Theorem 1.5, we obtain the following result, which matches the best bound [8] up to an $O(\log \frac{1}{\varepsilon})$ factor in the query time, but offers faster preprocessing time.

Theorem 1.7. *Given a set S of n points in $\mathbb{R}^d$, an approximation parameter $\varepsilon > 0$, and m such that $\log \frac{1}{\varepsilon} \leq m \leq 1/(\varepsilon^{d/2} \log \frac{1}{\varepsilon})$, there is a data structure that can answer Euclidean ε-approximate nearest neighbor queries with*

$$\textit{Query time: } O\left(\log n + \frac{\log \frac{1}{\varepsilon}}{m \cdot \varepsilon^{\frac{d}{2}}}\right) \textit{ Space: } O(nm) \textit{ Preprocessing: } O\left(n \log n \log \frac{1}{\varepsilon} + \frac{nm}{\varepsilon^{\alpha}}\right).$$

1.3 Techniques

In contrast to previous kernel constructions, which are based on grids and the execution of Bronshteyn and Ivanov's algorithm, our construction employs a classical structure from the theory of convexity, called *Macbeath regions* [27]. Macbeath regions have found numerous uses in the theory of convex sets and the geometry of numbers (see Bárány [14] for an excellent survey). They have also been applied to several problems in the field of computational geometry. However, most previous results were either in the form of lower bounds [9, 12, 17] or focused on existential results [6, 7, 23, 30].

In [8] the authors introduced a data structure based on a hierarchy of ellipsoids based on Macbeath regions to answer approximate polytope membership queries, but the efficient computation of the hierarchy was not considered. In this paper, we show how to efficiently construct the Macbeath regions that form the basis of this hierarchy.

Let P denote a convex polytope in $\mathbb{R}^d$. Each level i in the hierarchy corresponds to a δ_i-approximation of the boundary of P by a set of $O(1/\delta_i^{(d-1)/2})$ ellipsoids, where $\delta_i = \Theta(1/2^i)$. Each ellipsoid is sandwiched between two Macbeath regions and has $O(1)$ children, which correspond to the ellipsoids of the following level that approximate the same portion of the boundary (see Figure 1). The hierarchy starts with $\delta_0 = \Theta(1)$ and stops after $O(\log \frac{1}{\delta})$ levels when $\delta_i = \delta$, for a desired approximation δ. We present a simple algorithm to construct the hierarchy in $O(n + 1/\delta^{3(d-1)/2})$ time. The polytope P can be presented as either the intersection of n halfspaces or the convex hull of n points. We present the relevant background in Section 3.

Our algorithm to compute an ε-kernel in time $O(n \log \frac{1}{\varepsilon} + 1/\varepsilon^{(d-1)/2+\alpha})$ (Theorem 1.1) is conceptually quite simple. Since the time to build the ε-approximation hierarchy for the convex hull is prohibitively high, we use an approximation parameter $\delta = \varepsilon^{1/3}$ to build a δ-approximation hierarchy in $O(n + 1/\varepsilon^{(d-1)/2})$ time. By navigating through this hierarchy, we partition the n points among the leaf Macbeath ellipsoids in $O(n \log \frac{1}{\varepsilon})$ time, discarding points that are too far from the boundary. We then compute an (ε/δ)-kernel for the set of points in each leaf ellipsoid and return the union of the kernels computed.

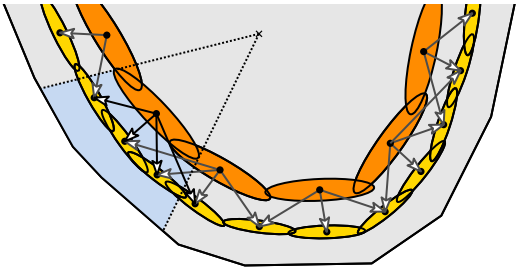


Figure 1: Two levels of the hierarchy of ellipsoids based on Macbeath regions.

Given an algorithm to compute an ε-kernel in $O(n \log \frac{1}{\varepsilon} + 1/\varepsilon^{t(d-1)})$ time, the previous procedure produces an ε-kernel in $O(n \log \frac{1}{\varepsilon} + 1/\varepsilon^{t'(d-1)})$ time where $t' = (4t+1)/6$. Bootstrapping the construction a constant number of times, the value of t goes down from 1 to a value that is arbitrarily close to 1/2. This discrepancy accounts for the $O(1/\varepsilon^{\alpha})$ factors in our running times. In Section 4, we present the complete algorithm and its analysis, proving Theorem 1.1.

In Section 5, we use our kernel construction in the dual space to efficiently build a polytope membership data structure, proving Theorem 1.5. The key idea is to compute multiple kernels in order to avoid examining the whole polytope when building each Macbeath region. Again, we use bootstrapping to obtain a near-optimal preprocessing time. The remaining theorems follow from Theorems 1.1 and 1.5, together with several known reductions (Section 6).

2 Geometric Preliminaries

Consider a convex body K in d-dimensional space $\mathbb{R}^d$. Let ∂K denote the boundary of K. Let O denote the origin of $\mathbb{R}^d$. Given a parameter $0 < \gamma \leq 1$, we say that K is γ-*fat* if there exist concentric Euclidean balls B and B', such that $B \subseteq K \subseteq B'$, and radius$(B)$/radius$(B') \geq \gamma$. We say that K is *fat* if it is γ-fat for a constant γ (possibly depending on d, but not on ε).

Unless otherwise specified, the notion of ε-*approximation* between convex bodies will be based on the direction-sensitive definition given in Section 1.2. We say that a convex body K' is an *absolute* ε-*approximation* to another convex body K if they are within Hausdorff error ε of each other. Further, we say that K' is an *inner* (resp., *outer*) approximation if $K' \subseteq K$ (resp., $K' \supseteq K$).

Let B_0 denote a ball of radius $r_0 = 1/2$ centered at the origin. For $0 < \gamma \leq 1$, let γB_0 denote the concentric ball of radius $\gamma r_0 = \gamma/2$. We say that a convex body K is in γ-*canonical form* if it is nested between γB_0 and B_0. A body in γ-canonical form is γ-fat and has diameter $\Theta(1)$. We will refer to point O as the *center* of P.

For any point $x \in K$, define $\delta(x)$ to be minimum distance from x to any point on ∂K. For the sake of ray-shooting queries, it is useful to define a ray-based notion of distance as well. Given $x \in K$, define the *ray-distance* of x to the boundary, denoted ray(x), as follows. Consider the intersection point p of ∂K and the ray emanating from O and passing through x. We define ray$(x) = \|xp\|$. The following utility lemma will be helpful in relating distances to the boundary.

Lemma 2.1. *Given a convex body K in γ-canonical form:*

(a) *For any point $x \in P$, $\delta(x) \leq \text{ray}(x) \leq \delta(x)/\gamma$.*

(b) *Let h be a supporting hyperplane of K. Let p be any point inside K at distance at most w from h, where $w \leq \gamma/4$. Let p' denote the intersection of the ray Op and h. Then $\|pp'\| \leq 2w/\gamma$.*

(c) *Let p be any point on the boundary of K, and let h be a supporting hyperplane at p. Let h' denote the hyperplane obtained by translating h in the direction of the outward normal by w. Let p' denote the intersection of the ray Op with h'. Then $\|pp'\| \leq w/\gamma$.*

We omit the straightforward proof. The lower bound on ray(x) for part (a) is trivial, and the upper bound follows by a straightforward adaption of Lemma 4.2 of [7]. Part (b) is an adaptation of Lemma 2.11 of [8], and part (c) is similar.

For any centrally symmetric convex body A, define A^λ to be the body obtained by scaling A by a factor of λ about its center. The following lemma appears in Barany [13].

Lemma 2.2. *Let $\lambda \geq 1$. Let A and B be centrally symmetric convex bodies such that $A \subseteq B$. Then $A^\lambda \subseteq B^\lambda$.*

2.1 Caps and Macbeath Regions

Much of the material in this section has been presented in [7, 8]. We include it here for the sake of completeness. Given a convex body K, a *cap* C is defined to be the nonempty intersection of K with a halfspace (see Figure 2(a)). Let h denote the hyperplane bounding this halfspace. We define the *base* of C to be $h \cap K$. The *apex* of C is any point in the cap such that the supporting hyperplane of K at this point is parallel to h. The *width* of C, denoted width(C), is the distance between h and this supporting hyperplane. Given any cap C of width w and a real $\lambda \geq 0$, we define its λ-*expansion*, denoted C^λ, to be the cap of K cut by a hyperplane parallel to and at distance λw from this supporting hyperplane. (Note that $C^\lambda = K$, if λw exceeds the width of K along the defining direction.)

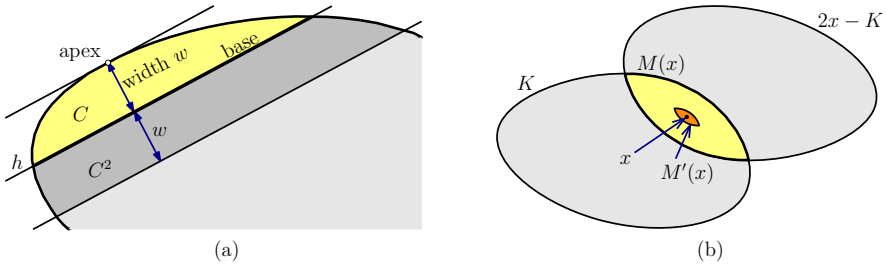


Figure 2: (a) Cap concepts and (b) Macbeath regions.

Given a point $x \in K$ and real parameter $\lambda \geq 0$, the *Macbeath region* $M^\lambda(x)$ (also called an *M-region*) is defined as:

$$M^\lambda(x) \;=\; x + \lambda((K - x) \cap (x - K)).$$

It is easy to see that $M^1(x)$ is the intersection of K and the reflection of K around x (see Figure 2(b)). Clearly, $M^1(x)$ is centrally symmetric about x, and $M^\lambda(x)$ is a scaled copy of $M^1(x)$ by the factor λ about x. We refer to x as the *center* of $M^\lambda(x)$ and to λ as its *scaling factor*. As a convenience, we define $M(x) = M^1(x)$ and $M'(x) = M^{1/5}(x)$. We refer to the latter as the *shrunken* Macbeath region.

We now present a few lemmas that encapsulate key properties of Macbeath regions. The first lemma shows that if two shrunken Macbeath regions have a nonempty intersection, then a constant factor expansion of one contains the other [8, 17, 24].

Lemma 2.3. *Let K be a convex body, and let $\lambda \leq 1/5$ be any real. If $x, y \in K$ such that $M^\lambda(x) \cap M^\lambda(y) \neq \emptyset$, then $M^\lambda(y) \subseteq M^{4\lambda}(x)$.*

The next lemma is useful in situations when we know that a shrunken Macbeath region partially overlaps a cap of K. It allows us to conclude that a constant factor expansion of the cap will fully contain the Macbeath region. The proof appears in [7].

Lemma 2.4. *Let K be a convex body. Let C be a cap of K and x be a point in K such that $C \cap M'(x) \neq \emptyset$. Then $M'(x) \subseteq C^2$.*

The following lemma shows that all points in a shrunken Macbeath region have similar distances from the boundary of K. The proof appears in [8].

Lemma 2.5. *Let K be a convex body. If $x \in K$ and $x' \in M'(x)$, then $4\delta(x)/5 \leq \delta(x') \leq 4\delta(x)/3$.*

For any $\delta > 0$, define the δ-erosion of a convex body K, denoted $K(\delta)$, to be the closed convex body formed by removing from K all points lying within distance δ of ∂K. The next lemma bounds the number of disjoint Macbeath regions that can be centered on the boundary of $K(\delta)$. The proof appears in [8].

Lemma 2.6. *Consider a convex body $K \subset \mathbb{R}^d$ in γ-canonical form for some constant γ. Define $\Delta_0 = \frac{1}{2}(\gamma^2/(4d))^d$. For any fixed constant $0 < \lambda \leq 1/5$ and real parameter $\delta \leq \Delta_0$, let $\mathcal{M}$ be a set of disjoint λ-scaled Macbeath regions whose centers lie on the boundary of $K(\delta)$. Then $|\mathcal{M}| = O(1/\delta^{(d-1)/2})$.*

2.2 Shadows of Macbeath regions

Shrunken Macbeath regions reside within the interior of the convex body, but it is useful to identify the portion of the body's boundary that this Macbeath region will be responsible for approximating. For this purpose, we introduce the shadow of a Macbeath region. Given a convex body K that contains the origin O and a region $R \subseteq K$, we define the *shadow* of R (with respect to K), denoted shadow(R), to be the set of points $x \in K$ such that the line segment Ox intersects R.

We also define a set of *normal directions* for R, denoted normals(R). Consider the set of all hyperplanes that support K at some point in the shadow of R. Define normals(R) to be the set of outward unit normals to these supporting hyperplanes. Typically, the region R in our constructions will be a (scaled) Macbeath region or an associated John ellipsoid (as defined in Section 3), close to the boundary of K. The following lemma captures a salient feature of these shadows, namely, that the shadow of a Macbeath region $M'(x)$ can be enclosed in an ellipsoid whose width in all normal directions is $O(\delta(x))$.

Lemma 2.7. *Let $K \subset \mathbb{R}^d$ be a convex body in γ-canonical form for some constant γ. Let x be a point at distance δ from the boundary of K, where $\delta \leq \Delta_0$. Let $M = M'(x), S = \text{shadow}(M)$, $N = \text{normals}(M)$, and $\widehat{M} = M^{4/\gamma}(x)$. Then:*

(a) $S \subseteq \widehat{M}$.

(b) $\text{width}_v(S) \leq c_1\delta$ *for all* $v \in N$. *Here* c_1 *is the constant* $8/(3\gamma)$.

(c) $\text{width}_v(\widehat{M}) \leq c_2\delta$ *for all* $v \in N$. *Here* c_2 *is the constant* $160/(3\gamma^2)$.

Proof. We first prove (a). Consider any point $p \in \partial K \cap S$. Let y denote the first point of intersection of the ray Op with the Macbeath region M. To prove (a), it suffices to show that the segment yp is contained in $\widehat{M}$ which, by convexity, is equivalent to showing that both points y and p are contained in $\widehat{M}$. Since $y \in M$, we have $y \in \widehat{M}$. To prove that $p \in \widehat{M}$, observe that a straightforward consequence of the definition of Macbeath regions is that $M(y)$ must contain a ball of radius $\delta(y)$ centered at y. Further, by Lemma 2.1(a), $\|yp\| = \text{ray}(y) \leq \delta(y)/\gamma$. It follows that $p \in M^{1/\gamma}(y)$. Also, since $y \in M'(x)$, it follows trivially that $M'(y)$ overlaps with $M'(x)$. Thus, by Lemma 2.3, $M'(y) \subseteq M^{4/5}(x)$. Applying Lemma 2.2 to $M'(y)$ and $M^{4/5}(x)$ with $\lambda = 5/\gamma$, we obtain $M^{1/\gamma}(y) \subseteq M^{4/\gamma}(x) = \widehat{M}$. Thus $p \in \widehat{M}$, which proves (a).

To prove (b), let p be any point of $\partial K \cap S$ and let y denote any point in the intersection of the ray Op with $M'(x)$. Let h denote any hyperplane supporting K at p. Let v denote the outward normal to h. We translate hyperplane h to pass through y and let C denote the cap of K cut by the resulting hyperplane. Clearly $\text{width}(C) \leq \|py\|$. By Lemma 2.1(a), $\|py\| = \text{ray}(y) \leq \delta(y)/\gamma$. Since $y \in M'(x)$, by Lemma 2.5, $\delta(y) \leq 4\delta(x)/3 = 4\delta/3$. Thus $\text{width}(C) \leq \delta(y)/\gamma \leq 4\delta/(3\gamma)$. Also, by Lemma 2.4, since $M'(x)$ overlaps C, $M'(x) \subseteq C^2$. It follows from convexity that $S \subseteq C^2$ and thus

$$\text{width}_v(S) \;\leq\; \text{width}(C^2) \;\leq\; 2\,\text{width}(C) \;\leq\; 8\delta/(3\gamma),$$

which proves (b).

To prove (c), recall that $M'(x) \subseteq S$, which implies that $\text{width}_v(M'(x)) \leq 8\delta/(3\gamma)$. Thus $\text{width}_v(\widehat{M}) = (20/\gamma)\text{width}_v(M'(x)) \leq 160\,\delta/(3\gamma^2)$. □

2.3 Representation Conversions

Convex sets are naturally described in two ways, as the convex hull of a discrete set of points and as the intersection of a discrete set of halfspaces. Some computational tasks are more easily performed using one operation or the other. For this reason, it will be useful to be able to convert between one representation and the other. Also, when approximate representations suffice, it will be useful

to prune a large set down to a smaller size. In this section we will present a few technical utilities to perform these conversions.

Given an n-element point set in $\mathbb{R}^d$, Chan showed that it is possible to construct an ε-kernel of size $O(1/\varepsilon^{(d-1)/2})$ in time $O(n + 1/\varepsilon^{d-1})$ [19]. The following lemma shows that, by applying Chan's construction, is is possible to efficiently approximate the convex hull of n points as the intersection of halfspaces.

Lemma 2.8. *Let $\gamma < 1$ be a positive constant, and $\varepsilon > 0$ be a real parameter. Let P be a polytope in γ-canonical form represented as the convex hull of n points. In $O(n + 1/\varepsilon^{d-1})$ time it is possible to compute a polytope P' represented as the intersection of $O(1/\varepsilon^{(d-1)/2})$ halfspaces such that P' is an inner absolute ε-approximation of P.*

Proof. Throughout the proof, to avoid tracking the numerous constant factors, we use the notation $O(\varepsilon)$ to denote a quantity that is a suitable scaling of ε by a constant factor. Let S be the input set of n points such that $P = \text{conv}(S)$. Since P is in γ-canonical form, we have $S \subseteq B_0$, where B_0 denotes the ball of radius $r_0 = 1/2$ centered at the origin. By applying Chan's kernel construction [19], in time $O(n + 1/\varepsilon^{d-1})$ we can compute a point set S'' of size $O(1/\varepsilon^{(d-1)/2})$ such that $\text{conv}(S'')$ is an absolute $O(\varepsilon)$-approximation of $\text{conv}(S)$.

We then apply the polar transformation to the points of S'' yielding a set of $O(1/\varepsilon^{(d-1)/2})$ halfspaces in the dual space. It follows from standard properties of the polar transformation that the polytope $\widehat{P}$ defined by the intersection of these halfspaces is fat and has constant diameter (see, e.g., Lemma 7.2 of the journal version of [5]). This can be performed in time $O(1/\varepsilon^{(d-1)/2})$.

Next, take a sufficiently large hypercube centered at the origin that contains $\widehat{P}$ and there is constant separation between the boundary of this hypercube and $\widehat{P}$. (Side length $O(1/\gamma)$ suffices.) We superimpose a grid of side length $\Theta(\sqrt{\varepsilon})$ on each of the $2d$ facets of this hypercube. Letting G denote the resulting set of grid points on the boundary of the hypercube, we have $|G| = O(1/\varepsilon^{(d-1)/2})$. Through the use of quadratic programming we compute the nearest neighbor of each point of G on the boundary of $\widehat{P}$. For each grid point this can be done in time linear in the number of halfspaces that define $\widehat{P}$ [28, 29]. Thus, the total time for computing all the nearest neighbors is $O(|S''| \cdot |G|) = O(1/\varepsilon^{d-1})$. Letting $\widehat{S}$ denote the set of nearest neighbors so obtained, we have $|\widehat{S}| = O(1/\varepsilon^{(d-1)/2})$. By standard results, $\text{conv}(\widehat{S})$ is an absolute $O(\varepsilon)$-approximation of $\widehat{P}$ [18, 22].

We again apply the polar transformation, mapping the set $\widehat{S}$ back to the primal space to obtain a set H of $O(1/\varepsilon^{(d-1)/2})$ halfspaces. Let P'' be the polytope formed by intersecting these halfspaces. Recalling that $\text{conv}(\widehat{S})$ is an absolute $O(\varepsilon)$-approximation of $\widehat{P}$, it follows from standard results that P'' is an absolute $O(\varepsilon)$-approximation of $\text{conv}(S'')$. This step takes time $O(1/\varepsilon^{(d-1)/2})$.

Since P'' is an absolute $O(\varepsilon)$-approximation of $\text{conv}(S'')$, and $\text{conv}(S'')$ is an absolute $O(\varepsilon)$-approximation of P, it follows that (subject to a suitable choice of constant factors) P'' is an absolute $O(\varepsilon)$-approximation of P. Define P' to be the polytope obtained by first translating the bounding halfspaces of P'' (i.e., the halfspaces of H) towards the origin by an amount $\Theta(\varepsilon)$ and then intersecting the resulting halfspaces. Clearly, P' is then an absolute inner $O(\varepsilon)$-approximation of P, as desired.

The overall running time is dominated by the time needed to compute the kernel, and the time needed to compute the nearest neighbors for the points of G. □

The following lemma is useful when representing polytopes by the intersection of halfspaces.

Lemma 2.9. *Let $\gamma < 1$ be a positive constant, and $\varepsilon > 0$ be a real parameter. Let P be a polytope in γ-canonical form represented as the intersection of n halfspaces. In $O(n + 1/\varepsilon^{d-1})$ time it is possible to compute a polytope P' represented as the intersection of $O(1/\varepsilon^{(d-1)/2})$ halfspaces such that P' is an outer absolute ε-approximation of P.*

Proof. Let H denote the set of n halfspaces defining P. We apply the polar transformation to the halfspaces of H obtaining a set S of n points in the dual space. It follows from standard properties of the polar transformation that $\text{conv}(S)$ is fat and has constant diameter (see, e.g., the journal version of [5]). This step can be performed in $O(n)$ time. By applying Chan's kernel construction [19], in time $O(n+1/\varepsilon^{d-1})$, we can compute a point set S' of size $O(1/\varepsilon^{(d-1)/2})$ such that $\text{conv}(S')$ is an inner absolute $O(\varepsilon)$-approximation of $\text{conv}(S)$. We again apply the polar transformation, mapping the set S' back to the primal space to obtain a set H' of $O(1/\varepsilon^{(d-1)/2})$ halfspaces. Let P' be the polytope

formed by intersecting these halfspaces. Since $\text{conv}(S')$ is an inner absolute $O(\varepsilon)$-approximation of $\text{conv}(S)$, it follows that (subject to a suitable choice of constant factors), P' is an absolute outer $O(\varepsilon)$-approximation of P, as desired. The total time is dominated by the time to compute the kernel. □

Remark: Theorem 1.1 shows that an ε-kernel of size $O(1/\varepsilon^{(d-1)/2})$ can be computed in time $O(n \log \frac{1}{\varepsilon} + 1/\varepsilon^{(d-1)/2+\alpha})$. The construction time in Lemma 2.9 is asymptotically the same as the time needed to construct an ε-kernel. Therefore, the construction time can be reduced to this quantity.

3 Hierarchy of Macbeath Ellipsoids

The data structure presented in [8] for the approximate polytope membership problem is based on constructing a hierarchy of ellipsoids. In this section, we present a variant of this structure, which will play an important role in our constructions.

For a Macbeath region $M^\lambda(x)$, we denote its circumscribing John ellipsoid by $E^\lambda(x)$, which we call a *Macbeath ellipsoid*. Since Macbeath regions are centrally symmetric and the constant in John's Theorem [25] is $\sqrt{d}$ for centrally symmetric bodies, we have $E^\lambda(x) \subseteq M^{\lambda\sqrt{d}}(x)$. Recall the constant $\Delta_0 = \frac{1}{2}(\gamma^2/4d)^d$ defined in the statement of Lemma 2.6, and define $\lambda_0 = 1/(20d)$. Each level of our structure is based on the following lemma. (We caution the reader that in the lemmas of this section, the value of n used in the application of the lemma may differ from the original input size.)

Lemma 3.1. *Let $\gamma < 1$ be a positive constant, and let $0 < \delta \leq \Delta_0$ be a real parameter. Let P be a polytope in γ-canonical form, represented as the intersection of n halfspaces. There exists a set $X \subseteq \partial P(\delta)$ consisting of $O(1/\delta^{(d-1)/2})$ points such that the following properties hold:*

(a) *The set of Macbeath regions $\{M^{\lambda_0}(x) : x \in X\}$ are pairwise disjoint.*

(b) *The set of Macbeath ellipsoids $\{E^{4\lambda_0\sqrt{d}}(x) : x \in X\}$ together cover $\partial P(\delta)$.*

Furthermore, in $O(n/\delta^{d-1} + 1/\delta^{3(d-1)/2})$ time, we can construct the set of Macbeath ellipsoids $\{E^{4\lambda_0\sqrt{d}}(x) : x \in X\}$.

Proof. We first show how to construct the required set of Macbeath ellipsoids. Translate each bounding halfspace of P towards the origin by amount δ. It is easy to see that the polytope $P(\delta)$ is the intersection of the translated halfspaces. This can be done in $O(n)$ time.

Recalling that $P \subseteq B_0$, where B_0 is the ball of radius $r_0 = 1/2$, consider the hypercube just enclosing B_0. Superimpose a $\Theta(\delta)$-grid on each of the $2d$ facets of this hypercube. Intersect the segment joining the origin to each grid point with $\partial P(\delta)$, and let $X' \subset \partial P(\delta)$ denote the resulting set of intersection points. Note that $|X'| = O(1/\delta^{d-1})$. Using the fact that $P(\delta)$ is fat, a straightforward geometric calculation shows that for any point on $\partial P(\delta)$, there is a point of X' within distance $c\delta$ of it, where c is a suitable constant. (Adjusting the constant factor in the grid spacing, we can ensure that $c \leq \lambda_0\sqrt{d}$, which is a fact that we will use later in the proof.) As each point of X' can be determined in $O(n)$ time, X' can be computed in $O(n/\delta^{d-1})$ time.

For each $x' \in X'$, construct $M^{\lambda_0}(x')$. Let $\mathcal{M}'$ denote the resulting set of Macbeath regions. As a straightforward consequence of the definition of Macbeath regions, we can compute each Macbeath region in $O(n)$ time (i.e., in time proportional to the number of halfspaces that define P). Note that we represent each Macbeath region as the intersection of n halfspaces. For each Macbeath region $M^{\lambda_0}(x') \in \mathcal{M}'$, determine the circumscribing John ellipsoid $E^{\lambda_0}(x')$. By standard results, we can construct the John ellipsoid of a convex polytope in time that is linear in the number of its defining halfspaces [21]. Thus, this step also takes time $O(n|\mathcal{M}'|) = O(n/\delta^{d-1})$.

Next, we will determine a maximal subset $\mathcal{M} \subseteq \mathcal{M}'$ such that the John ellipsoids associated with the Macbeath regions of $\mathcal{M}$ are disjoint. Initialize $\mathcal{M} = \emptyset$. Examine the Macbeath regions of $\mathcal{M}'$ one by one. Insert the Macbeath region into $\mathcal{M}$ if its associated John ellipsoid does not intersect the John ellipsoid of any Macbeath region of $\mathcal{M}$. Clearly, this method yields a maximal subset $\mathcal{M} \subseteq \mathcal{M}'$ such that the associated John ellipsoids are disjoint. To bound the time required for this step, observe that the Macbeath regions of $\mathcal{M}$ are disjoint, and so by Lemma 2.6, $|\mathcal{M}| = O(1/\delta^{(d-1)/2})$. Since it takes constant time to check whether two ellipsoids intersect, it follows that the time required is $O(|\mathcal{M}'| \cdot |\mathcal{M}|) = O(1/\delta^{3(d-1)/2})$.

Finally, to obtain the desired ellipsoids let X denote the set of centers of the Macbeath regions of $\mathcal{M}$. For each $x \in X$, we scale the associated ellipsoid $E^{\lambda_0}(x)$ constructed above about its center by a factor of $4\sqrt{d}$ to obtain the ellipsoid $E^{4\lambda_0\sqrt{d}}(x)$. This step can be done in time $O(|\mathcal{M}|) = O(1/\delta^{(d-1)/2})$.

By combining the time of the above steps we obtain the desired overall construction time of $O(n/\delta^{d-1} + 1/\delta^{3(d-1)/2})$. The bound on $|X|$ follows from the bound on $\mathcal{M}$. By our earlier remarks, the set of Macbeath regions $\{M^{\lambda_0}(x) : x \in X\}$ are pairwise disjoint, which proves Property (a).

It remains to establish Property (b), that is, to show that the union of the ellipsoids $\{E^{4\lambda_0\sqrt{d}}(x) : x \in X\}$ covers $\partial P(\delta)$. Towards these end, consider any point $p \in \partial P(\delta)$. We will show that there is an $x \in X$ such that the ellipsoid $E^{4\lambda_0\sqrt{d}}(x)$ contains p. Recall that there is a point $x' \in X'$ that is within distance $c\delta$ of p, where c is a constant no more than $\lambda_0\sqrt{d}$. A straightforward consequence of the definition of Macbeath regions is that, for any point $y \in P$, the Macbeath region $M(y)$ contains a ball of radius $\delta(y)$ centered at y. It follows that $M^{\lambda_0\sqrt{d}}(x')$ contains a ball of radius $\lambda_0\sqrt{d}\,\delta$ centered at x'. Hence, $p \in M^{\lambda_0\sqrt{d}}(x') \subseteq M^{4\lambda_0\sqrt{d}}(x') \subseteq E^{4\lambda_0\sqrt{d}}(x')$. Thus, if $x' \in X$, we are done.

We next consider the case when $x' \notin X$. In this case, it follows from our construction that there is an $x \in X$ such that $E^{\lambda_0}(x)$ intersects $E^{\lambda_0}(x')$. Recall that $E^{\lambda_0}(x) \subseteq M^{\lambda_0\sqrt{d}}(x)$ and $E^{\lambda_0}(x') \subseteq M^{\lambda_0\sqrt{d}}(x')$. Thus $M^{\lambda_0\sqrt{d}}(x)$ intersects $M^{\lambda_0\sqrt{d}}(x')$. Applying Lemma 2.3, it follows that $M^{\lambda_0\sqrt{d}}(x') \subseteq M^{4\lambda_0\sqrt{d}}(x)$. Thus $p \in M^{4\lambda_0\sqrt{d}}(x) \subseteq E^{4\lambda_0\sqrt{d}}(x)$. It follows that the ellipsoids $E^{4\lambda_0\sqrt{d}}(x)$, for $x \in X$, together cover $\partial P(\delta)$, which completes the proof of the lemma. □

Based on the above lemma, we are now ready to describe our hierarchical data structure. Let P be a polytope in γ-canonical form, where γ is a constant. Recall the constant $\Delta_0 = \frac{1}{2}(\gamma^2/(4d))^d$, and for $i \geq 0$ define $\Delta_i = \Delta_0/2^i$. The data structure consists of levels $0, 1, \ldots, \ell$, where ℓ is the smallest integer such that $\Delta_\ell \leq \delta$. Since γ is a constant, $\ell = O(\log \frac{1}{\delta})$. Since P is in γ-canonical form, the origin O is at distance at least $\gamma/2$ from ∂P. Since $\Delta_0 < \gamma/2$, it follows that $P(\Delta_0)$ contains O and $P(\Delta_i) \subset P(\Delta_{i+1})$ for $0 \leq i \leq \ell - 1$. For each level i, we apply Lemma 3.1, setting δ in the lemma to Δ_i. In $O(n/\Delta_i^{d-1} + 1/\Delta_i^{3(d-1)/2})$ time, we obtain a set of $O(1/\Delta_i^{(d-1)/2})$ ellipsoids that cover $\partial P(\Delta_i)$. Summing over all levels, the number of ellipsoids is $O(1/\delta^{(d-1)/2})$ and the time taken is $O(n/\delta^{d-1} + 1/\delta^{3(d-1)/2})$.

Our data structure is a directed acyclic graph (DAG), where the nodes at level i correspond to the ellipsoids computed for level i. The children of an ellipsoid E at level i are the ellipsoids E' at level $i+1$ such that there exists a ray from the origin that simultaneously intersects E and E'. Since this is a constant time operation for any two given ellipsoids, it takes $O(1/\delta^{d-1})$ time to find the children of all the nodes in the DAG. It is convenient to root the DAG by creating a special node whose children are all the nodes of level zero. An important property of the hierarchy is that each node only has $O(1)$ children. The proof of this property is similar to that given in [8]. We include the proof for the sake of completeness. For the root, this follows from the fact that the number of nodes of level zero is $O(1/\Delta_0^{(d-1)/2})$ and Δ_0 is a constant. For non-root nodes, the proof is based on the following lemma, which is proved in [8].

Lemma 3.2. *Let $K \subset \mathbb{R}^d$ be a convex body in γ-canonical form for some constant γ, and let $\lambda \leq 1/5$ be any constant. Let $x \in K$ such that $\delta(x) \leq \Delta_0$. Consider the generalized cone formed by rays emanating from the center O of K and intersecting $M'(x)$. Let Y denote any set of points y such that $\delta(y) = \delta(x)/2$ and the set of Macbeath regions $M^\lambda(y)$ are disjoint. Let $Y' \subseteq Y$ denote the set of points y such that $M'(y)$ overlaps the aforementioned cone. Then $|Y'| = O(1)$.*

For any node w, we let x_w denote the center of the associated ellipsoid. Consider any node u at level $i \geq 0$. Also, consider the generalized cone formed by rays emanating from the origin that intersect the ellipsoid $E^{4\lambda_0\sqrt{d}}(x_u)$ associated with u. The children of u are those nodes v at level $i+1$ whose ellipsoid $E^{4\lambda_0\sqrt{d}}(x_v)$ intersects this generalized cone.

Since $x_u \in \partial P(\Delta_i)$, we have $\delta(x_u) = \Delta_i \leq \Delta_0$. Recall that $E^{4\lambda_0\sqrt{d}}(x_u) \subseteq M^{4\lambda_0 d}(x_u) = M'(x_u)$. Thus, the generalized cone of rays that intersect $M'(x_u)$ includes all the rays used to define the children of u. The points x_v that form level $i+1$ of the structure lie on $\partial P(\Delta_{i+1})$ and thus satisfy $\delta(x_v) = \delta(x_u)/2$. By Property (a) of Lemma 3.1, the Macbeath regions $M^{\lambda_0}(x_v)$ are pairwise disjoint, thus they constitute a set Y as described in the preconditions of Lemma 3.2. Each child

v of u corresponds to a point x_v such that the ellipsoid $E^{4\lambda_0\sqrt{d}}(x_v)$ intersects the generalized cone. Reasoning as we did above for x_u, we have $E^{4\lambda_0\sqrt{d}}(x_v) \subseteq M'(x_v)$. Therefore, the points x_v associated with the children of u constitute a subset of the set Y' given in the lemma. Therefore, the number of children of u is $O(1)$, as desired.

Given a query ray Oq, our data structure allows us to quickly find a leaf node such that the associated ellipsoid intersects this ray. The query algorithm descends the DAG by starting at the root and visiting any node at level zero that intersects the ray. Letting u denote the current node, we next visit any child of u whose associated ellipsoid intersects the ray. We repeat this procedure until a leaf node is reached. As the number of levels is $O(\log \frac{1}{\delta})$, this quantity bounds the time taken by this procedure. We summarize the main result of this section.

Lemma 3.3. *Let $\gamma < 1$ be a positive constant, and let $0 < \delta \leq \Delta_0$ be a real parameter. Let P be a polytope in γ-canonical form, represented as the intersection of n halfspaces. In $O(n/\delta^{d-1} + 1/\delta^{3(d-1)/2})$ time, we can construct the DAG structure described above. In particular, the DAG satisfies the following properties:*

(a) *The total number of nodes (including leaves), and the total space used by the DAG are both $O(1/\delta^{(d-1)/2})$.*

(b) *Each leaf is associated with an ellipsoid $E^{4\lambda_0\sqrt{d}}(x)$, where $x \in \partial P(\delta)$. The union of the ellipsoids associated with all the leaves covers $\partial P(\delta)$.*

(c) *Given a query ray Oq, in $O(\log \frac{1}{\delta})$ time, we can find a leaf node such that the associated ellipsoid intersects this ray.*

Given a convex body K and query point q, an *absolute ε-APM* query returns a positive result if q lies within K, a negative result if q is at distance at least ε from K, and otherwise it may return either result. After a small enhancement, this DAG can be used for answering absolute ε-APM queries for a polyope P in γ-canonical form. We assume that P is represented as the intersection of a set H of n halfspaces. We invoke the above lemma for $\delta = \varepsilon\gamma/(2c_1)$, where c_1 is the constant of Lemma 2.7(b). We then associate each leaf of the DAG with a halfspace as follows. Let x denote the center of the leaf ellipsoid and let p denote the intersection of the ray Ox with ∂P. Let $h \in H$ denote any supporting halfspace of P (containing P) at p. We store h with this leaf. By exhaustive search, we can determine h in $O(n)$ time, so the total time for this step is $O(n/\varepsilon^{(d-1)/2})$. Asymptotically, this does not affect the time it takes to construct the data structure. Given a query point q, we answer queries by first determining a leaf whose ellipsoid intersects the ray Oq. By Lemma 3.3(c), this takes $O(\log \frac{1}{\varepsilon})$ time. We return a positive answer if and only if q is contained in the associated halfspace. We establish the correctness of this method in the following lemma.

Lemma 3.4. *Given a query point q, the query procedure returns a valid answer to the absolute ε-APM query.*

Proof. Consider the leaf whose ellipsoid intersects the ray Oq. Let $E^{4\lambda_0\sqrt{d}}(x)$ denote the associated ellipsoid and let h be the halfspace stored with this leaf. Recall that h is a supporting halfspace at the point p where the ray Ox intersects ∂P. If $q \in P$ then clearly $q \in h$ and such a query point is correctly declared as lying inside P. To complete the proof, we need to show that if $q \notin P$ and the distance of q from ∂P is greater than ε, then $q \notin h$. In this case, q would be correctly declared as lying outside P.

Let y denote any point in the intersection of the ray Oq with the leaf ellipsoid. Let y' denote the intersection of the ray Oq with the hyperplane bounding h. To prove the claim, it suffices to show that $\|yy'\| \leq \varepsilon$. Recall that $E^{4\lambda_0\sqrt{d}}(x) \subseteq M^{4\lambda_0 d}(x) = M'(x)$. Let $M = M'(x), S = \text{shadow}(M)$, and $N = \text{normals}(M)$. Clearly $y, p \in S$ and the normal vector v to the hyperplane bounding h belongs to N. By Lemma 2.7, $\text{width}_v(S) \leq c_1\delta(x) = \varepsilon\gamma/2$. Note that the distance of y from the hyperplane bounding h is at most $\text{width}_v(S)$. Applying Lemma 2.1(b), we obtain $\|yy'\| \leq (2/\gamma)(\varepsilon\gamma/2) = \varepsilon$, as desired. □

We summarize the result below.

Lemma 3.5. *Let $\gamma < 1$ be a positive constant, and let $\varepsilon > 0$ be a real parameter. Let P be a polytope in γ-canonical form, represented as the intersection of n halfspaces. In $O(n/\varepsilon^{d-1} + 1/\varepsilon^{3(d-1)/2})$ time, we can construct a data structure that uses $O(1/\varepsilon^{(d-1)/2})$ space and answers absolute ε-APM queries in $O(\log \frac{1}{\varepsilon})$ time.*

4 Kernel Construction

In this section we show how to build an ε-kernel efficiently, proving Theorem 1.1. The input to an ε-kernel construction consists of the approximation parameter ε and a set S of n points. Our algorithm is based on a bootstrapping strategy. We assume that we have access to an algorithm that can construct an ε-kernel of $O(1/\varepsilon^{(d-1)/2})$ size in time $O(n \log \frac{1}{\varepsilon} + 1/\varepsilon^{(1/2+\beta)(d-1)})$, where $\beta > 0$ is a parameter. Recall that the size of the kernel is asymptotically optimal in the worst case. We will present a method for improving the running time of this algorithm. Recall that Chan [19] gave an algorithm for constructing kernels of optimal size which runs in time $O(n \log \frac{1}{\varepsilon} + 1/\varepsilon^{d-1})$. We will use this algorithm to initialize our bootstrapping scheme with $\beta = 1/2$.

Our method is based on executing the following steps. It uses a parameter $\delta = \varepsilon^{1/3}$.

1. We begin by "fattening" the input point set S. Formally, we compute an affine transformation that maps S to S', such that $\mathrm{conv}(S')$ is in γ-canonical form for some constant γ. By standard results (see, e.g., the journal version of [5]), this affine transformation and the set S' can be computed in $O(n)$ time.

2. Use Lemma 2.8 to build a polytope P, represented as the intersection of $O(1/\delta^{(d-1)/2})$ halfspaces, such that P is an inner absolute δ-approximation of $\mathrm{conv}(S')$. This step takes $O(n + 1/\delta^{d-1}) = O(n + 1/\varepsilon^{(d-1)/3})$ time.

3. Construct the DAG structure of Lemma 3.3 for polytope P using the parameter δ. Replacing n in the statement of the lemma by $O(1/\delta^{(d-1)/2})$, it follows that this step takes $O(1/\delta^{3(d-1)/2}) = O(1/\varepsilon^{(d-1)/2})$ time.

4. For each point $p \in S'$, in $O(\log \frac{1}{\delta})$ time, we find a leaf of the DAG such that the associated ellipsoid $E^{4\lambda_0\sqrt{d}}(x)$ intersects the ray Op. Recall that $x \in \partial P(\delta)$. In $O(1)$ additional time, we can determine whether p lies in the shadow of this ellipsoid (with respect to $\mathrm{conv}(S')$). If so, we associate p with this ellipsoid, otherwise we discard it. By Lemma 3.3(c), it takes $O(\log \frac{1}{\delta})$ time to process each point, thus the time taken for processing all the points of S' is $O(n \log \frac{1}{\delta}) = O(n \log \frac{1}{\varepsilon})$.

5. For each leaf ellipsoid of the DAG, we build a $(c_3\varepsilon/\delta)$-kernel for the points of S' that lie in its shadow, where c_3 is a suitably small constant that will be selected later. This kernel computation is done using the aforementioned algorithm that computes ε-kernels of point sets of size n in time $O(n \log \frac{1}{\varepsilon} + 1/\varepsilon^{(1/2+\beta)(d-1)})$. The size of the $O(\varepsilon/\delta)$-kernel computed for each shadow is $O((\delta/\varepsilon)^{(d-1)/2})$ and the time required is $O(n_i \log \frac{\delta}{\varepsilon} + (\delta/\varepsilon)^{(1/2+\beta)(d-1)})$, where n_i denotes the number of points of S' in the shadow. Summed over all the shadows, it follows that the total time required is

$$O\left(n \log \frac{\delta}{\varepsilon} + \left(\frac{1}{\delta}\right)^{\frac{d-1}{2}} \left(\frac{\delta}{\varepsilon}\right)^{\left(\frac{1}{2}+\beta\right)(d-1)}\right) = O\left(n \log \frac{1}{\varepsilon} + \left(\frac{1}{\varepsilon}\right)^{\left(\frac{1}{2}+\frac{2\beta}{3}\right)(d-1)}\right).$$

   Here we have used the facts that each point of S' is assigned to at most one shadow and the total number of shadows, which is bounded by the number of leaves in the DAG, is $O(1/\delta^{(d-1)/2})$.

6. Let $S'' \subseteq S'$ be the union of the kernels computed in the previous step. Since the number of shadows is $O(1/\delta^{(d-1)/2})$ and the size of the kernel for each shadow is $O((\delta/\varepsilon)^{(d-1)/2})$, it follows that $|S''| = O(1/\varepsilon^{(d-1)/2})$. We apply the inverse of the affine transformation computed in Step 1 to the points of S'', and output the resulting set of points as the desired ε-kernel for S.

We have shown that the size of the output kernel is $O(1/\varepsilon^{(d-1)/2})$, as desired. The running time of Step 5 dominates the time complexity. The next lemma establishes the correctness of this construction.

Lemma 4.1. *The construction yields an ε-kernel.*

Proof. Throughout this proof, for a given convex body K, we use $M_K(x)$, $E_K(x)$, and $\delta_K(x)$ to denote the quantities $M(x)$, $E(x)$, and $\delta(x)$ with respect to K. Let $P' = \text{conv}(S')$. By standard results on fattening, it suffices to show that $\text{conv}(S'')$ is an absolute $O(\varepsilon)$-approximation of P'. Let v be an arbitrary direction. Consider the extreme point p of S' in direction v. Clearly $p \in \partial P'$. Recall that P is an inner δ-approximation of P', and the ellipsoids associated with the leaves of the DAG cover the boundary of $P(\delta)$. Thus, there must be an ellipsoid $E = E_P^{4\lambda_0\sqrt{d}}(x)$, $x \in \partial P(\delta)$, such that p is assigned to the shadow of E in Step 4. Note that this shadow and all shadows throughout this proof are assumed to be with respect to the polytope P' (and not P). We claim that $\text{width}_v(\text{shadow}(E)) \leq 2c_1\delta$, where c_1 is the constant of Lemma 2.7(b). Assuming this claim for now, let us complete the proof of the lemma. Recall that in Step 5, we built a $(c_3\varepsilon/\delta)$-kernel for all the points of S' that are assigned to the shadow of E, and S'' includes all the points of this kernel. It follows that the distance between the supporting hyperplanes of $\text{conv}(S')$ and $\text{conv}(S'')$ in direction v is at most $(c_3\varepsilon/\delta) \cdot \text{width}_v(\text{shadow}(E)) \leq (c_3\varepsilon/\delta) \cdot (2c_1\delta) = 2c_1c_3\varepsilon$. By choosing c_3 sufficiently small, we can ensure that this quantity is smaller than any desired constant times ε, which proves the lemma.

It remains to show that $\text{width}_v(\text{shadow}(E)) \leq 2c_1\delta$. Recall that

$$E \;=\; E_P^{4\lambda_0\sqrt{d}}(x) \;\subseteq\; M_P^{4\lambda_0 d}(x) \;=\; M'_P(x).$$

Furthermore, since $P \subseteq P'$, a straightforward consequence of the definition of Macbeath regions is that $M'_P(x) \subseteq M'_{P'}(x)$. To simplify the notation, let M denote $M'_{P'}(x)$. Putting it together, we obtain $E \subseteq M$. Thus $\text{shadow}(E) \subseteq \text{shadow}(M)$, which implies that $\text{width}_v(\text{shadow}(E)) \leq \text{width}_v(\text{shadow}(M))$. By Lemma 2.7(b),

$$\text{width}_v(\text{shadow}(M)) \;\leq\; c_1\delta_{P'}(x).$$

Using the triangle inequality and the fact that P is an inner δ-approximation of P', we obtain $\delta_{P'}(x) \leq \delta_P(x) + \delta = 2\delta$. Thus $\text{width}_v(\text{shadow}(E)) \leq \text{width}_v(\text{shadow}(M)) \leq 2c_1\delta$, as desired. □

We are now ready to prove Theorem 1.1.

Proof. Our proof is based on a constant number of applications of the algorithm from this section. It suffices to show that there is an algorithm that can construct an ε-kernel of $O(1/\varepsilon^{(d-1)/2})$ size in time $O(n \log \frac{1}{\varepsilon} + 1/\varepsilon^{(1/2+\beta')(d-1)})$, where $\beta' = \alpha/(d-1)$.

We initialize the bootstrapping process by Chan's algorithm [19], which has $\beta = 1/2$. Observe that the value of β is initially $1/2$ and falls by a factor of $2/3$ with each application of the algorithm. It follows that after $O(\log \frac{1}{\alpha})$ applications, we will obtain an algorithm with the desired running time. This completes the proof. □

5 Approximate Polytope Membership

In this section we show how to obtain a data structure for approximate polytope membership, proving Theorem 1.5. Our best data structure for APM achieves query time $O(\log \frac{1}{\varepsilon})$ with storage $O(1/\varepsilon^{(d-1)/2})$ and preprocessing time $O(n \log \frac{1}{\varepsilon} + 1/\varepsilon^{(d-1)/2+\alpha})$. As with kernels, our construction here is again based on a bootstrapping strategy. To initialize the process, we will use a data structure that achieves the aforementioned query time with the same storage but with preprocessing time $O(n + 1/\varepsilon^{3(d-1)/2})$. The data structure is based on Lemma 3.5. Recall that the input is a polytope represented as the intersection of n halfspaces.

We begin by "fattening" the input polytope. Formally, we use an affine transformation to map the input polytope to a polytope P' that is in γ-canonical form. This step takes $O(n)$ time [5]. By standard results, it suffices to build a data structure for answering absolute $O(\varepsilon)$-APM queries with respect to P' (see, e.g., Lemma 7.1 of the journal version of [5]).

Next, we apply Lemma 2.9 to construct an outer absolute $O(\varepsilon)$-approximation P of P', where P is represented as the intersection of $O(1/\varepsilon^{(d-1)/2})$ halfspaces. This step takes $O(n + 1/\varepsilon^{d-1})$ time. Finally, we use Lemma 3.5 to construct a data structure for answering absolute $O(\varepsilon)$-APM queries

with respect to P. Replacing n in the statement of the lemma by $O(1/\varepsilon^{(d-1)/2})$, it follows that this step takes $O(1/\varepsilon^{3(d-1)/2}))$ time.

The total construction time is $O(n+1/\varepsilon^{3(d-1)/2})$. To answer a query, we map the query point using the same transformation used to fatten the polytope, and then use the data structure constructed above to determine whether the resulting point lies in polytope P. Subject to an appropriate choice of constant factors, the correctness of this method follows from the fact that P is an outer absolute $O(\varepsilon)$-approximation of P'.

We summarize this result in the following lemma.

Lemma 5.1. *Let $\varepsilon > 0$ be a real parameter and let P be a polytope, represented as the intersection of n halfspaces. In $O(n + 1/\varepsilon^{3(d-1)/2})$ time, we can construct a data structure that uses $O(1/\varepsilon^{(d-1)/2})$ space and answers ε-APM queries in $O(\log \frac{1}{\varepsilon})$ time.*

We can now present the details of our bootstrapping approach. We assume that we have access to a data structure that can answer ε-APM queries in $O(\log \frac{1}{\varepsilon})$ time with $O(1/\varepsilon^{(d-1)/2})$ storage and $O(n \log \frac{1}{\varepsilon} + 1/\varepsilon^{(1/2+\beta)(d-1)})$, where $\beta > 0$ is a parameter. We present a method for constructing a new data structure which matches the given data structure in space and query time, but has a lower preprocessing time. Our method uses a parameter $\delta = \varepsilon^{\beta/(1+\beta)}$.

1. As in the construction given above, we first fatten the input polytope. Formally, we use an affine transformation to map the input polytope to a polytope P' that is in γ-canonical form. This step takes $O(n)$ time. By standard results, it suffices to build a data structure for answering absolute $O(\varepsilon)$-APM queries with respect to P'.

2. Use Lemma 2.9 to construct an outer absolute $O(\varepsilon)$-approximation P of P', where P is represented as the intersection of $O(1/\varepsilon^{(d-1)/2})$ halfspaces. By the remark following Lemma 2.9, this step takes $O(n \log \frac{1}{\varepsilon} + 1/\varepsilon^{(d-1)/2+\alpha})$ time.

3. Construct the DAG of Lemma 3.3 for polytope P using the parameter δ. Replacing n in the statement of the lemma by $O(1/\varepsilon^{(d-1)/2})$, it follows that this step takes $O((1/\delta)^{d-1} \cdot (1/\varepsilon)^{(d-1)/2})$ time.

4. For each leaf of the DAG, we construct an APM data structure as follows. Let $E = E^{4\lambda_0\sqrt{d}}(x)$ denote the ellipsoid associated with the leaf. Let R denote the minimum enclosing hyperrectangle of the ellipsoid $E^{4/\gamma}(x)$. We will see later that R contains the shadow of E (with respect to P), and its width in any direction in normals(E) is at most $c_2 d\delta = O(\delta)$, where c_2 is the constant in Lemma 2.7(c). We use the aforementioned algorithm for constructing an APM data structure for this region with approximation parameter $c_3\varepsilon/\delta$, where c_3 is a sufficiently small constant that we will select later. Note that each such region can be expressed as the intersection of $n_i = O(1/\varepsilon^{(d-1)/2})$ halfspaces, namely, all the halfspaces defining P together with the $2d$ halfspaces defined by the facets of R. The construction time of the APM data structure for each leaf is

$$O\left(n_i \log \frac{\delta}{\varepsilon} + \left(\frac{\delta}{\varepsilon}\right)^{\left(\frac{1}{2}+\beta\right)(d-1)}\right) \;=\; O\left(\left(\frac{1}{\varepsilon}\right)^{\frac{d-1}{2}} \log \frac{\delta}{\varepsilon} + \left(\frac{\delta}{\varepsilon}\right)^{\left(\frac{1}{2}+\beta\right)(d-1)}\right),$$

   and the space used is $O((\delta/\varepsilon)^{(d-1)/2})$. Since there are $O(1/\delta^{(d-1)/2})$ leaves, it follows that the total space is $O(1/\varepsilon^{(d-1)/2})$, and the total construction time is the product of $O(1/\delta^{(d-1)/2})$ and the above construction time for each leaf.

Summing up the time over all the four steps, we get a total construction time on the order of

$$n \log \frac{1}{\varepsilon} + \left(\frac{1}{\varepsilon}\right)^{\frac{d-1}{2}+\alpha} + \left(\frac{1}{\delta}\right)^{d-1}\left(\frac{1}{\varepsilon}\right)^{\frac{d-1}{2}} + \left(\frac{1}{\delta}\right)^{\frac{d-1}{2}} \cdot \left(\left(\frac{1}{\varepsilon}\right)^{\frac{d-1}{2}} \log \frac{\delta}{\varepsilon} + \left(\frac{\delta}{\varepsilon}\right)^{\left(\frac{1}{2}+\beta\right)(d-1)}\right).$$

Recalling that $\delta = \varepsilon^{\beta/(1+\beta)}$ and assuming that the constant α is much smaller than β, it follows that the construction time is

$$O\left(n \log \frac{1}{\varepsilon} + \left(\frac{1}{\varepsilon}\right)^{\left(\frac{1}{2}+\frac{\beta}{1+\beta}\right)(d-1)}\right).$$

We answer queries as follows. Recall the affine transformation used to fatten the input polytope. We apply this transformation on the input query point to obtain a point q. Recall that it suffices to answer absolute $O(\varepsilon)$-APM queries for q with respect to P'. As P is an outer absolute $O(\varepsilon)$-approximation of P', it suffices to answer absolute $O(\varepsilon)$-APM queries for q with respect to P. To answer this query, we identify a leaf of the DAG such that the associated ellipsoid E intersects the ray Oq. This takes time $O(\log \frac{1}{\delta})$. Let y denote an intersection point of this ray with the ellipsoid E. If q lies on the segment Oy, then q is declared as lying inside P. Otherwise we return the answer we get for query q using the APM data structure we built for this leaf. It takes time $O(\log \frac{\delta}{\varepsilon})$ to answer this query. Including the time to locate the leaf, the total query time is $O(\log \frac{1}{\varepsilon})$.

In Lemma 5.2, we show that queries are answered correctly.

Lemma 5.2. *The query procedure returns a valid answer to the ε-APM query.*

Proof. We borrow the terminology from the query procedure given above. As mentioned, it suffices to show that our algorithm correctly answers absolute $O(\varepsilon)$-APM queries for q with respect to the polytope P. Recall that we identify a leaf of the DAG whose associated ellipsoid $E = E^{4\lambda_0\sqrt{d}}(x)$ intersects the ray Oq. Recall that y is a point on the intersection of the ray Oq with E. Clearly, if q lies on segment Oy, then $q \in P$ and q is correctly declared as lying inside P.

It remains to show that queries are answered correctly when $\|Oq\| > \|Oy\|$. In this case, we handle the query using the APM data structure we built for the leaf. Recall that this structure is built for the polytope formed by intersecting P with the smallest enclosing hyperrectangle R of the ellipsoid $E^{4/\gamma}(x)$. We claim that (i) $\text{shadow}(E) \subseteq R$ and (ii) $\text{width}_v(R) \leq c_2 d\delta$ for all $v \in \text{normals}(E)$, where c_2 is the constant in Lemma 2.7(c).

To see this claim, recall that $M^{\lambda}(x) \subseteq E^{\lambda}(x) \subseteq M^{\lambda\sqrt{d}}(x)$ for any $\lambda > 0$. Using this fact, it follows that $M^{4/\gamma}(x) \subseteq E^{4/\gamma}(x) \subseteq M^{4\sqrt{d}/\gamma}(x)$. By Lemma 2.7(a), $\text{shadow}(E) \subseteq M^{4/\gamma}(x)$. Thus $\text{shadow}(E) \subseteq E^{4/\gamma}(x) \subseteq R$, which proves (i). To prove (ii), note that $R \subseteq E^{4\sqrt{d}/\gamma}(x)$, since R is the smallest enclosing hyperrectangle of $E^{4/\gamma}(x)$. Also $E^{4\sqrt{d}/\gamma}(x) \subseteq M^{4d/\gamma}(x)$. Thus $R \subseteq M^{4d/\gamma}(x)$. By Lemma 2.7(c), $\text{width}_v(M^{4/\gamma}(x)) \leq c_2\delta$ for all $v \in \text{normals}(M'(x))$. Since $R \subseteq M^{4d/\gamma}(x)$ and $E \subseteq M'(x)$, it follows that $\text{width}_v(R) \leq c_2 d\delta$ for all $v \in \text{normals}(E)$.

We return to showing that queries are correctly answered when $\|Oq\| > \|Oy\|$. We consider two possibilities depending on whether q is inside or outside P. If $q \in P$ then $q \in \text{shadow}(E)$. By part (i) of the above claim, $\text{shadow}(E) \subseteq R$. Thus $q \in P \cap R$. It follows that the APM structure built for the leaf will declare this point as lying inside $P \cap R$, and hence the overall algorithm will correctly declare that q lies in P.

Finally, we consider the case when $q \notin P$. To complete the proof, we need to show that if the distance of q from the boundary of P is greater than ε, then q is declared as lying outside P. Let p denote the point of intersection of the ray Oq with ∂P, let h denote a hyperplane supporting P at p, and let v denote the outward normal to h. Recall by part (i) of the claim that $\text{shadow}(E) \subseteq R$. It follows that h is a supporting hyperplane of $P \cap R$ at p. By part (ii) of the claim, $\text{width}_v(R) \leq c_2 d\delta$. It follows that $\text{width}_v(P \cap R) \leq c_2 d\delta$. Recall that the APM data structure for the leaf is built using approximation parameter $c_3\varepsilon/\delta$ for some constant c_3. By definition of APM query (in the standard, direction-sensitive sense), the absolute error allowed in direction v is at most $(c_3\varepsilon/\delta) \cdot \text{width}_v(P \cap R) \leq (c_3\varepsilon/\delta)(c_2 d\delta)$. By choosing c_3 sufficiently small we can ensure that this error is at most $\varepsilon\gamma$. To make this more precise, let h' denote the hyperplane parallel to h (outside P), and at distance $\varepsilon\gamma$ from it. Consider the halfspace bounded by h' and containing P. By definition of APM query, if q is not contained in this halfspace, then q would be declared as lying outside $P \cap R$, and the overall algorithm would declare q as lying outside P. Let p' denote the point of intersection of the ray Oq with h'. By Lemma 2.1(c), $\|pp'\| \leq (\varepsilon\gamma)/\gamma = \varepsilon$. Thus, if the distance of q from ∂P is greater than ε, then q cannot lie on segment pp' and q is correctly declared as lying outside P. This completes the proof of correctness. □

We are now ready to prove Theorem 1.5

Proof. Our proof is based on a constant number of applications of the method presented in this section. It suffices to show that there is a data structure with space and query time as in the theorem and preprocessing time $O(n \log \frac{1}{\varepsilon} + 1/\varepsilon^{(1/2+\beta')(d-1)})$, where $\beta' = \alpha/(d-1)$.

We initialize the bootstrapping process by the data structure described in the beginning of this section, which has $\beta = 1$. Recall that applying the method once changes the value of β to $\beta/(1+\beta)$. It is easy to show that after i applications, the value of β will fall to $1/(i+1)$. Thus, after $O(1/\alpha)$ applications, we will obtain a data structure with the desired preprocessing time. □

6 Reductions

In this section, we show how the remaining problems reduce to polytope membership. We start with a useful variation of approximate nearest neighbor searching.

The input for an approximate nearest neighbor searching data structure is a set S of data points and an approximation parameter ε. Given a constant $\sigma > 0$, *σ-well-separated* approximate nearest neighbor searching is defined as follows. Let Q_S and Q_q be two hypercubes of side length r and at distance at least σr from each other. In the σ-well-separated version we have the data points S inside Q_S and the query points inside Q_q. Data structures for the well-separated version are much more efficient than for the unrestricted version. The following reduction from well-separated approximate nearest neighbor searching to approximate polytope membership is presented in [5, Lemma 9.2 of the journal version].

Lemma 6.1. *Let $0 < \varepsilon \leq 1/2$ be a real parameter, $\sigma > 0$ be a constant, and S be a set of n points in $\mathbb{R}^d$. Given a data structure for approximate polytope membership in d-dimensional space with query time $t_d(\varepsilon)$, storage $s_d(\varepsilon)$, and preprocessing time $O(n \log \frac{1}{\varepsilon} + b_d(\varepsilon))$ it is possible to preprocess S into a σ-well-separated ANN data structure with*

$$\textit{Query time: } O\left(t_{d+1}(\varepsilon) \cdot \log \frac{1}{\varepsilon}\right) \quad \textit{Space: } O\big(s_{d+1}(\varepsilon)\big) \quad \textit{Preprocessing: } O\left(n \log \frac{1}{\varepsilon} + b_{d+1}(\varepsilon)\right).$$

Combining the previous reduction with Theorem 1.5 we have:

Lemma 6.2. *Given a set S of n points in $\mathbb{R}^d$, an approximation parameter $\varepsilon > 0$, and a constant $\sigma > 0$, there is a data structure that can answer σ-well-separated Euclidean ε-approximate nearest neighbor queries with*

$$\textit{Query time: } O\left(\log^2 \frac{1}{\varepsilon}\right) \quad \textit{Space: } O\left(\left(\frac{1}{\varepsilon}\right)^{\frac{d}{2}}\right) \quad \textit{Preprocessing: } O\left(n \log \frac{1}{\varepsilon} + \left(\frac{1}{\varepsilon}\right)^{\frac{d}{2}+\alpha}\right).$$

Next, we prove Theorem 1.3 using a reduction to well-separated approximate nearest neighbor searching that is based on [4, Theorem 3.2].

Proof. Let b denote the exact BCP distance. We obtain a constant approximation $b \leq a < 2b$ of the BCP distance in $O(n)$ expected time by running the randomized algorithm from [26]. Then, we build a grid with cells of diameter $a/4$ and partition the red points accordingly. Note that since $a/4 < b/2$, the BCP pair cannot be in the same grid cell, nor in two adjacent cells. The strategy of the algorithm is to partition the red points among the grid cells and to perform a constant number of well-separated approximate nearest neighbor queries for each blue point, returning the closest red-blue pair found. More precisely, for each blue point q, we perform an approximate nearest neighbor query among the grid cells Q_S that intersect the set theoretic difference of two balls of radii a and $a/2$ centered around q. These are the only grid cells that may contain the closest red point and, by a simple packing argument, the number of grid cells Q_S is constant. Since the grid cell Q_q that contains q cannot be adjacent to Q_S, it follows that the separation σ is at least 1.

To answer the queries efficiently, we separate the grid cells onto two types. If the number of red points in the cell is greater than $1/\varepsilon^{d/4}$, we say the cell is *heavy*, and otherwise we say the cell is *light*. Clearly, the number of heavy cells is $O(n \cdot \varepsilon^{d/4})$. We build well-separated approximate nearest-neighbor data structures for the heavy cells. Using Lemma 6.2, the total preprocessing time is $O(n/\varepsilon^{d/4+\alpha})$. For each light cell, we simply store the red points it contains and answer nearest neighbor queries by brute force in $O(1/\varepsilon^{d/4})$ time. Therefore, the total time spent answering queries is $O(n/\varepsilon^{d/4})$. □

An approximation to the Euclidean minimum spanning tree and minimum bottleneck tree can be computed by solving multiple BCP instances such that the sum of the number of points in all instances is $O(n \log n)$ [4, Theorem 4.1]. Applying this reduction together with Theorem 1.3, we prove Theorem 1.4.

The following reduction from ANN to APM is presented in [5, Lemma 9.3 of the journal version]. For the preprocessing time see [10, Lemma 8.3].

Lemma 6.3. *Let $0 < \varepsilon \leq 1/2$ be a real parameter and S be a set of n points in $\mathbb{R}^d$. Given a data structure for approximate polytope membership in d-dimensional space with query time at most $t_d(\varepsilon)$ and storage $s_d(\varepsilon)$, and preprocessing time $O(n \log \frac{1}{\varepsilon} + b_d(\varepsilon))$ it is possible to preprocess S into an ANN data structure with*

$$\textit{Query time: } O\left(\log n + t_{d+1}(\varepsilon) \cdot \log \frac{1}{\varepsilon}\right) \quad \textit{Space: } O\left(n \log \frac{1}{\varepsilon} + n \frac{s_{d+1}(\varepsilon)}{t_{d+1}(\varepsilon)}\right)$$
$$\textit{Preprocessing: } O\left(n \log n \log \frac{1}{\varepsilon} + n \frac{b_{d+1}(\varepsilon)}{t_{d+1}(\varepsilon)}\right).$$

Applying this reduction with the data structure from Theorem 1.5 and setting $t_{d+1}(\varepsilon) = 1/(m \cdot \varepsilon^{d/2})$ for $\log \frac{1}{\varepsilon} \leq m \leq 1/(\varepsilon^{d/2} \log \frac{1}{\varepsilon})$, we obtain Theorem 1.7.

Next, we show how to obtain a data structure for approximate directional width queries (Theorem 1.6) using the data structure for approximate polytope membership from Theorem 1.5. The proof uses standard duality and binary search techniques.

Proof. Given a polytope P (defined as the intersection of n halfspaces) that contains the origin O, we define a ray-shooting query (from the origin) as follows. Let v be a query direction and let r denote the ray emanating from O in direction v. The result of the query $q(P, v)$ is the length of $r \cap P$. In the ε-approximate version, any answer between $q(P, v)$ and $(1 + \varepsilon)q(P, v)$ is acceptable.

If we place the origin O in the center of the John ellipsoid of P, we have $q(P, -v) = \Theta(q(P, v))$ for all v. Thus, a constant approximation of $q(P, v)$ can be obtained by replacing P by its circumscribing John ellipsoid. We can then refine the approximation using binary search and approximate polytope membership queries. (To see this, consider the point $p \in \partial P$ that is hit by the ray, and let h be any supporting hyperplane at p. Consider the slab containing P that is bounded by this hyperplane and the parallel hyperplane on the opposite side of P. By properties of the John ellipsoid, the origin lies within a central region of the slab. It follows from basic geometry that if we expand the slab by ε times its width, the ratio between ray distances to the expanded slab boundary and the original slab boundary is $1 + O(\varepsilon)$. An ε-APM query with respect to P along this ray will achieve an approximation error that is no greater.) By a suitable adjustment to the constant factor, we can obtain an ε-approximation to $q(P, v)$ after $O(\log \frac{1}{\varepsilon})$ membership queries.

The polar body P^* (defined as the convex hull of n points) of P has the property that $\text{width}_v(P^*) = 1/q(P, v) + 1/q(P, -v)$. Therefore, we can ε-approximate the width of a set of points P^* using $O(\log \frac{1}{\varepsilon})$ approximate polytope membership queries on P and Theorem 1.6 follows. □

Agarwal, Matoušek, and Suri [3] showed that the diameter of a point set S can be ε-approximated by computing the maximum width of S among $O(1/\varepsilon^{(d-1)/2})$ directions. Therefore, Theorem 1.2 follows immediately from Theorem 1.6.

References

[1] P. K. Agarwal, S. Har-Peled, and K. R. Varadarajan. Approximating extent measures of points. *J. Assoc. Comput. Mach.*, 51:606–635, 2004.

[2] P. K. Agarwal, S. Har-Peled, and K. R. Varadarajan. Geometric approximation via coresets. In J. E. Goodman, J. Pach, and E. Welzl, editors, *Combinatorial and Computational Geometry*. MSRI Publications, 2005.

[3] P. K. Agarwal, J. Matoušek, and S. Suri. Farthest neighbors, maximum spanning trees and related problems in higher dimensions. *Comput. Geom. Theory Appl.*, 1(4):189–201, 1992.

[4] S. Arya and T. M. Chan. Better ε-dependencies for offline approximate nearest neighbor search, Euclidean minimum spanning trees, and ε-kernels. In *Proc. 30th Annu. Sympos. Comput. Geom.*, pages 416–425, 2014.

[5] S. Arya, G. D. da Fonseca, and D. M. Mount. Approximate polytope membership queries. In *Proc. 43rd Annu. ACM Sympos. Theory Comput.*, pages 579–586, 2011. (Full version available from http://arxiv.org/abs/1604.01183, to appear on SIAM J. Computing).

[6] S. Arya, G. D. da Fonseca, and D. M. Mount. Optimal area-sensitive bounds for polytope approximation. In *Proc. 28th Annu. Sympos. Comput. Geom.*, pages 363–372, 2012.

[7] S. Arya, G. D. da Fonseca, and D. M. Mount. On the combinatorial complexity of approximating polytopes. In *Proc. 32nd Internat. Sympos. Comput. Geom.*, pages 11:1–11:15, 2016. (Full version http://arxiv.org/abs/1604.01175, to appear on Discrete Comput. Geom.).

[8] S. Arya, G. D. da Fonseca, and D. M. Mount. Optimal approximate polytope membership. In *Proc. 28th Annu. ACM-SIAM Sympos. Discrete Algorithms*, pages 270–288, 2017. (Also available as http://arxiv.org/abs/1612.01696).

[9] S. Arya, T. Malamatos, and D. M. Mount. The effect of corners on the complexity of approximate range searching. *Discrete Comput. Geom.*, 41:398–443, 2009.

[10] S. Arya, T. Malamatos, and D. M. Mount. Space-time tradeoffs for approximate nearest neighbor searching. *J. Assoc. Comput. Mach.*, 57:1–54, 2009.

[11] S. Arya and D. M. Mount. A fast and simple algorithm for computing approximate Euclidean minimum spanning trees. In *Proc. 27th Annu. ACM-SIAM Sympos. Discrete Algorithms*, pages 1220–1233, 2016.

[12] S. Arya, D. M. Mount, and J. Xia. Tight lower bounds for halfspace range searching. *Discrete Comput. Geom.*, 47:711–730, 2012.

[13] I. Bárány. Intrinsic volumes and f-vectors of random polytopes. *Math. Ann.*, 285:671–699, 1989.

[14] I. Bárány. The technique of M-regions and cap-coverings: A survey. *Rend. Circ. Mat. Palermo*, 65:21–38, 2000.

[15] G. Barequet and S. Har-Peled. Efficiently approximating the minimum-volume bounding box of a point set in three dimensions. *J. Algorithms*, 38(1):91–109, 2001.

[16] J. L. Bentley, M. G. Faust, and F. P. Preparata. Approximation algorithms for convex hulls. *Commun. ACM*, 25(1):64–68, 1982.

[17] H. Brönnimann, B. Chazelle, and J. Pach. How hard is halfspace range searching. *Discrete Comput. Geom.*, 10:143–155, 1993.

[18] E. M. Bronshteyn and L. D. Ivanov. The approximation of convex sets by polyhedra. *Siberian Math. J.*, 16:852–853, 1976.

[19] T. M. Chan. Faster core-set constructions and data-stream algorithms in fixed dimensions. *Comput. Geom. Theory Appl.*, 35(1):20–35, 2006.

[20] T. M. Chan. Applications of Chebyshev polynomials to low-dimensional computational geometry. In *Proc. 33rd Internat. Sympos. Comput. Geom.*, pages 26:1–15, 2017.

[21] B. Chazelle and J. Matoušek. On linear-time deterministic algorithms for optimization problems in fixed dimension. *J. Algorithms*, 21:579–597, 1996.

[22] R. M. Dudley. Metric entropy of some classes of sets with differentiable boundaries. *J. Approx. Theory*, 10(3):227–236, 1974.

[23] K. Dutta, A. Ghosh, B. Jartoux, and N. H. Mustafa. Shallow packings, semialgebraic set systems, Macbeath regions and polynomial partitioning. In *Proc. 33rd Internat. Sympos. Comput. Geom.*, pages 38:1–15, 2017.

[24] G. Ewald, D. G. Larman, and C. A. Rogers. The directions of the line segments and of the r-dimensional balls on the boundary of a convex body in Euclidean space. *Mathematika*, 17:1–20, 1970.

[25] F. John. Extremum problems with inequalities as subsidiary conditions. In *Studies and Essays Presented to R. Courant on his 60th Birthday*, pages 187–204. Interscience Publishers, Inc., New York, 1948.

[26] S. Khuller and Y. Matias. A simple randomized sieve algorithm for the closest-pair problem. *Information and Computation*, 118(1):34–37, 1995.

[27] A. M. Macbeath. A theorem on non-homogeneous lattices. *Ann. of Math.*, 56:269–293, 1952.

[28] N. Megiddo. Linear-time algorithms for linear programming in R^3 and related problems. *SIAM J. Comput.*, 12:759–776, 1983.

[29] N. Megiddo. Linear programming in linear time when the dimension is fixed. *J. Assoc. Comput. Mach.*, 31:114–127, 1984.

[30] N. H. Mustafa and S. Ray. Near-optimal generalisations of a theorem of Macbeath. In *Proc. 31st Internat. Sympos. on Theoret. Aspects of Comp. Sci.*, pages 578–589, 2014.

Reports

Next, we include the reports of the the dissertation committee members:

- Jean Cardinal, Université Libre de Bruxelles (in French)
- Timothy Chan, University of Illinois at Urbana-Champaign
- Nabil Mustafa, Université de Paris-Est, ESIEE Paris

Followed by the report of the defense committee members:

- Nabil Mustafa, ESIEE Paris
- Jean-Daniel Boissonnat, INRIA, Sophia Antipolis
- Victor Chepoi, Aix-Marseille Université
- Bruno Martin, Université de Nice Sophia Antipolis
- David M. Mount, University of Maryland

UNIVERSITÉ LIBRE DE BRUXELLES, UNIVERSITÉ D'EUROPE

ULB

FACULTÉ DES SCIENCES
Département d'Informatique
Jean Cardinal, Professeur

Boulevard du Triomphe
T 02 650 56 08 – **F** 02 650 56 09
M jcardin@ulb.ac.be

Département d'Informatique
Localisation : Campus de la Plaine, Boulevard du Triomphe, accès 2, bâtiment NO, niveau 8
Adresse postale : ULB CP 212, avenue F.D. Roosevelt 50, 1050 Bruxelles, tél. 32 2 650 56 14, fax. 32 2 650 56 09

Concerne Rapport HDR Guilherme Dias da Fonseca

Bruxelles, le 25 mai 2018

Ceci est un rapport d'évaluation de l'Habilitation à Diriger des Recherches (HDR) de Guilherme Dias da Fonseca, Maître de Conférence en Informatique à l'Université de Clermont Auvergne, en délégation à l'INRIA Sophia Antipolis, rédigé par Jean Cardinal, Professeur à l'Université libre de Bruxelles (ULB), rapporteur.

Le présent rapport se compose de deux parties : (i) un résumé des apports principaux décrits dans le document soumis, (ii) une évaluation en termes d'importance et variété des résultats, de qualité et de visibilité, et une appréciation de l'autonomie du candidat.

Apports principaux

Contexte. Le candidat est spécialiste en géométrie algorithmique, et plus particulièrement des aspects théoriques liés aux problèmes de détection de proximité (appartenance à un polytope, plus proches voisins, diamètre) en haute dimension. Ces problèmes forment un corpus spécifique de la géométrie algorithmique, dans lesquels des outils théoriques sont nécessaires pour lutter contre le "fléau de la dimension", qui rend souvent impossible l'adaptation directe de méthodes en dimension deux ou trois à des dimensions plus hautes. Pour cette raison, on a souvent recours à des technique d'approximation.

Le problème principal étudié est celui de l'appartenance à un polytope : Etant donné un polytope P, construire une structure de données permettant de décider efficacement si un point q donné appartient à P. Cette réponse est ici approximative : on se donne en outre une petite constante ϵ et la réponse est arbitraire dans le cas où q est à l'extérieur mais à une distance inférieure à ϵ de P.

Partie 1 : Appartenance à un polytope par la méthode Split-Reduce. La première partie du document concerne des travaux publiés par le candidat en 2011 et 2012 dans les conférences STOC et SODA, et compilés dans un article de la revue SIAM Journal on Computing publié en 2018. Ces travaux établissent l'existence de solutions intermédiaires interpolant entre deux méthodes connues depuis plusieurs décennies pour le problème d'appartenance, à savoir une ϵ-approximation par un polytope due à Dudley, et l'utilisation de *quadtrees* par Bentley. La méthode proposée est naturelle mais son analyse est difficile : les bornes inférieures et supérieures connues sur la complexité de requête ne correspondent pas. En particulier, à espace fixé, l'écart entre la borne inférieure et supérieure sur le temps de requête est encore superquadratique. Une discussion concise mais précise des progrès effectués sur les bornes inférieures et supérieures est donnée dans le document, et résumée dans la figure 2.4.

Cette méthode ne permet néanmoins pas de résoudre le problème en espace $O(1/\epsilon^{(d-1)/2})$ avec un temps de requête $O(\log \frac{1}{\epsilon})$.La seconde partie présente une méthode complètement différente permettant ces performances.

Partie 2 : Appartenance à un polytope par la méthode des régions de MacBeath. La seconde méthode, présentée dans des articles des conférences SODA et SoCG en 2017, utilise les notions de régions de MacBeath. Celles-ci constituent un outil fondamental en théorie de la convexité, qui est ici intelligemment employé à des fins algorithmiques. En combinant un lemme de *packing* avec des résultats classiques d'approximation d'un polytope par un ellipsoïde, il est possible de construire une structure hiérarchique possédant les deux qualités désirables d'avoir un degré borné d'une part, et de couvrir le bord de P avec un nombre asymptotiquement minimum de régions d'autre part. L'algorithme de requête est une marche dans cette hiérarchie, qui consomme un nombre d'étapes logarithmique en $1/\epsilon$.

Partie 3 : Applications. La troisième partie résume les applications des résultats à d'autres problèmes fondamentaux en géométrie algorithmique. Loin d'être de simples corollaires, ces applications nécessitent parfois un traitement spécifique. L'application à la recherche de plus proches voisins repose sur les propriétés du relevé des points sur un paraboloïde en dimension $d+1$. Le calcul d'un ϵ-kernel repose sur la construction approximative de la hiérarchie de régions de MacBeath. Ces deux résultats peuvent être utilisés à leur tour dans un algorithme d'approximation du diamètre et de plus proches paires bichromatiques (BCP). Finalement, le résultat sur BCP peut être utilisé dans un algorithme d'approximation d'arbre couvrant Euclidien minimum.

Evaluation

Pertinence des problèmes étudiés. Les applications de la troisième partie mettent très clairement en évidence le caractère fondamental et omniprésent du problème étudié. Loin de résoudre uniquement un problème ponctuel, les méthodes présentées sont des outils génériques permettant d'attaquer un grand nombre de questions fondamentales en géométrie algorithmique sous l'angle de l'approximation.

Qualité des résultats et visibilité. Les résultats sont conséquents et bien exposés. Le résultat concernant les hiérarchies de régions de MacBeath est brillant, tant dans l'originalité de l'utilisation du concept de MacBeath, que dans le traitement non-trivial des propriétés relatives à son efficacité dans ce cadre précis. Un indicateur fiable de la qualité et de la visibilité de ces résultats est la liste

des conférences dans lesquelles ils ont été publiés. STOC, SODA et SoCG font effet partie des conférences du domaines les plus sélectives.

Une caractéristique peu commune du travail de Guilherme da Fonseca présenté dans cette HDR est la progression constante sur un problème de recherche fondamental et bien défini. Loin d'être un assemblage de résultats disparates, le document convoie efficacement les étapes successives d'une réflexion sur un problème dans une direction. Les questions ouvertes présentées à la fin du document indiquent en outre que le sujet est loin d'être épuisé.

Autonomie et collaborations. Les travaux présentés dans le document ont tous été réalisés en collaboration avec les professeurs Sunil Arya (HKUST) et David Mount (UMD), tous les deux spécialistes des problèmes de proximité géométrique. David Mount est en outre l'ancien directeur de thèse du candidat. En revanche, d'autres travaux non présentés dans ce document et réalisés avec une douzaine d'autres coauteurs internationaux attestent de la qualité de son réseau de collaborateurs.

Conclusion

Il ne fait aucun doute, à la lecture du document soumis, que Guilherme da Fonseca est un chercheur confirmé, ayant contribué à la résolution de nombreuses questions pertinentes et fondamentales, et qu'il possède toutes les capacités à contribuer au progrès des connaissances en informatique théorique et en géométrie discrète, ainsi qu'à encadrer des recherches dans ce domaine.

Je soutiens fermement et sans réserve l'organisation de la défense de cette HDR.

Jean Cardinal

UNIVERSITY OF ILLINOIS
AT URBANA-CHAMPAIGN

Department of Computer Science

201 N. Goodwin
Urbana, IL 61801

Ecole Doctorale STIC
Campus SophiaTech
Bâtiment Forum Haut
930 route des Colles
06410 Biot
France

May 16, 2018

Report on the HDR of Guilherme Dias da Fonseca

This dissertation studies efficient algorithms and data structures for approximate polytope membership queries, which are of fundamental importance and lie at the heart of several central problems in low-dimensional computational geometry, including approximate nearest neighbor search, approximate diameter, width, minimum-bottleneck spanning tree, etc.

In the mid-1990s, data structures were found by Arya, Mount, et al. that can answer $(1+\varepsilon)$-approximate nearest neighbor queries in logarithmic time and linear space, in any constant dimension, for any positive constant ε. For nearly two decades since, researchers have sought methods with better dependencies of the query time and space on the quality of approximation ε, with work by Clarkson [SoCG'94], myself [SoCG'97], Har-Peled [FOCS'01], and Arya, Malamatos, and Mount [SODA'02 and STOC'02], ... The work described in this dissertation, from a remarkable series of papers Guilherme wrote with Sunil Arya and David Mount [STOC'11, SODA'12, SODA'17, SoCG'17] (all done after his PhD), contains the current best results on this fundamental problem, introducing new techniques and pushing them to their limit.

The STOC'11 and SODA'12 papers (Article 1) by Guilherme and co-authors are truly seminal work. These papers are the first to formulate the approximate polytope membership problem, explain why this is the key to understanding the ε-dependency of approximate nearest neighbor search, and present improved results for a wide range of space/query-time trade-offs. Their solution is obtained by combining an elegant recursive quadtree algorithm (called "SplitReduce") with standard constructions (by Dudley) in the base cases. The analysis of the algorithm is highly nontrivial, and requires techniques from convexity theory, including the so-called "Mahler volume."

The SODA'17 paper (Article 2) describes further improvements. But these are not merely "incremental" improvements—Guilherme and co-authors in fact achieve the ultimate, *optimal* space/query-time bounds for the approximate polytope membership problem. Their data structure, consisting of a hierarchy of ellipsoids, requires $O((1/\varepsilon)^{(d-1)/2})$ space with $O(\log(1/\varepsilon))$ query time, and is obtained via a powerful new technique, based on so-called "Macbeath regions".

telephone 217-333-3426 *fax* 217-333-3501 *email* admin@cs.uiuc.edu

Finally, the SoCG'17 paper (Article 3) continues the investigation into the preprocessing time needed to build such data structures, and obtained the current best preprocessing-time/query-time trade-offs, which are paramount to algorithmic applications. The construction is closely linked to a type of coresets called *ε-kernels*, which form another fundamental family of problems in computational geometry and have been extensively studied by many researchers in the past (Agarwal, Har-Peled, and Varadarajan [JACM'04], myself [SoCG'04], and Arya and myself [SoCG'14]). Guilherme and co-authors obtained significantly better algorithms, in fact, almost optimal algorithms, for constructing ε-kernels. The new time bound, near $O(n + (1/\varepsilon)^{(d-1)/2})$, improves the previous bound, near $O(n + (1/\varepsilon)^{d-2})$. Applications include well known problems such as approximating the diameter of a point set, and the bichromatic closest pair (for the latter, the new time bound, near $O((1/\varepsilon)^{(d-1)/4}n)$, improves the previous time bound, near $O((1/\varepsilon)^{(d-1)/3}n)$). In my opinion, these are fantastic results! In an independent SoCG'17 paper, I recently discovered similar bounds, using very different techniques; Guilherme's bounds are actually slightly better than mine.

The concluding chapter of the dissertation hints at still more applications of these techniques, for example, to approximating the width of a point set in near $O(n + (1/\varepsilon)^{(d-1)/2})$ time. These new results also look intriguing (the width problem in particular is one that I have posed and was unable to solve), and I eagerly look forward to reading about them in the future.

In summary, this dissertation contains impressive original results, of fundamental importance, and with great technical depth. In addition, it is very well written. Clearly, this HDR deserves to be defended.

Sincerely,

Timothy Chan
Founder Professor in Computer Science
University of Illinois at Urbana-Champaign
`http://tmc.web.engr.illinois.edu/`

P.S. Some minor comments:

- The $O(n)$-time algorithm for approximate bichromatic closest pair uses constant-time hashing, and requires the unit-cost word RAM model, despite the largest paragraph on page 7 specifying the real RAM as the main computational model.
- In [1,3,66] of the bibliography, "minkowski" should be capitalized.

UNIVERSITÉ PARIS-EST

Report on HDR thesis of Guilherme Dias da Fonseca.

Nabil Mustafa
Professor

ESIEE Paris
Laboratoire d'Informatique Gaspard-Monge
Université Paris-Est,
Champs-sur-Marne, France.

Telephone: +33 1 92 45 66 07
Email: mustafan@esiee.fr

18 May, 2018.

The HDR thesis of Guilherme Dias da Fonseca is entitled "Approximate Polytope Membership Queries and Applications". Polytopes are key structures in the study of convexity, and succinct approximations of polytopes is one of the basic questions in the area. The contents of the thesis can be divided into two parts: the first part deals with aspects—both combinatorial and computational—of polytope approximation and approximate polytope membership problem in $\mathbb{R}^d$. The second part then presents applications of the results of the first part to many basic geometric optimization problems.

The basic setting is that, given a polytope $\mathcal{C}$ in $\mathbb{R}^d$, one would like to construct a small-sized 'approximate polytope' of $\mathcal{C}$, i.e., a simpler polytope $\mathcal{C}'$ such that the following approximate polytope membership query can be answered efficiently: given a point $q \in \mathbb{R}^d$, does q lie inside $\mathcal{C}$? The notion of approximation can

be parameterized by a parameter $\epsilon \in [0,1]$, so that the distance between $\mathcal{C}$ and $\mathcal{C}'$ must be upper-bounded by ϵ. $\mathcal{C}'$ is then called an ϵ-approximating polytope of $\mathcal{C}$. There are several ways to define the size of $\mathcal{C}'$—e.g., number of faces, number of vertices—as well as several different distance measures between $\mathcal{C}$ and $\mathcal{C}'$—Hausdorff distance being the main one studied in this thesis.

A classical result of Dudley from 1974 (building on earlier results) shows that for polytopes of constant diameter, there exist an ϵ-approximating polytope $\mathcal{C}'$ with $O\left(\frac{1}{\epsilon^{\frac{d-1}{2}}}\right)$ faces. Furthermore, this is tight as can be seen when $\mathcal{C}$ is a Euclidean ball. Unfortunately, algorithmically it is not very efficient in answering the approximate membership query; the naïve solution takes time $O\left(\frac{1}{\epsilon^{\frac{d-1}{2}}}\right)$.

The first part presents near-optimal solutions to this membership problem, returning back to add new and original insights to this classical problem. A first data-structure, called *split-reduce* data structure uses a clever combination of computational geometry techniques—gridding, quadtrees, hierarchical partitioning, coresets, ϵ-nets—together with ideas from convex geometry (curvature, Mahler volume etc.). In particular, the idea of analyzing the structure of convex bodies using Mahler volume—so that one can take random samples either with respect to curvature or surface area—is beautiful, and no doubt will turn out to be influential in future years.

A second improved bound is obtained using the idea of *Macbeath regions* from convex geometry. Roughly (ellipsoids of) Macbeath regions were used by Guilherme and co-authors to define a new way to hierarchically decompose the boundary of $\mathcal{C}$ with few ellipsoids. Besides the clever use of Macbeath regions, there are several technical issues that arise that had to be solved. Then the membership query can be answered by a variant of ray-shooting in this hierarchy. This is a crowning achievement in this area: an algorithm that answers approximate polytope membership queries in time $O\left(\log \frac{1}{\epsilon}\right)$ with storage $O\left(\frac{1}{\epsilon^{\frac{d-1}{2}}}\right)$.

The second part of the thesis turns towards applications of this and related structures to many of the most basic problems in computational geometry:

- nearest neighbors for point sets in $\mathbb{R}^d$, improving the storage bound from $O\left(\frac{n}{\epsilon^d}\right)$ to roughly $O\left(\frac{n}{\epsilon^{d/2}}\right)$, a very strong improvement on a bound from over 15 years ago.
- small-sized kernels and coresets among points in $\mathbb{R}^d$. This problem has be-

come important in recent years with massive data, and the need to compute small 'sketches' of data that capture various geometric and combinatorial properties.

- approximation algorithms for computing the diameter of point sets in $\mathbb{R}^d$, as well as approximating the directional width of point sets.
- other problems such as approximation algorithms for bichromatic closest pair and minimum spanning Euclidean trees.

The thesis

The thesis has been prepared with great care: there are many figures throughout that clearly illustrate the technical ideas and the involved proofs. I especially appreciated Chapters 2, 3 and 4, which are written elegantly and showed Guilherme's capacity for very good exposition balancing both clarity of thought as well as obeying the constraints of formal correctness. He explains clearly why certain problems are studied, and how they were proved over the years. He also demonstrates very clearly that his results truly advance the state of the art, and that some of these results answer long-standing and heavily researched problems.

The candidate

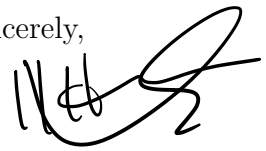
I consider Guilherme de Fonseca to be a very successful researcher in discrete and computational geometry. He has produced masterful works in the area of discrete and computational geometry, obtaining many important and original results. His command of classical mathematical subjects—convex geometry, discrete geometry—and their unexpected applications in designing algorithms for basic computational geometry problems is impressive. It is evident that Guilherme has clearly mastered these techniques, and built on them in a productive, difficult and useful manner.

Besides the results presented in this thesis, Guilherme has obtained other results in range searching, matching theory, data structures and combinatorial optimization problems in geometric intersection graphs. All these results have been published in very selective and excellent journals and conferences—SoCG, SODA,

STOC, Discrete & Computational Geometry, SIAM Journal on Computing, Algorithmica. Overall, Guilherme de Fonseca is the author of more than 29 published articles.

For all these reasons, I am very enthusiastically in favor of this thesis; the thesis can be accepted without reserve for the "Diplome national d'Habilitation à diriger des recherches".

Sincerely,



Nabil Mustafa

Laboratoire d'Informatique Gaspard-Monge, Université de Paris-Est, ESIEE Paris,
Champs-sur-Marne. http://ligm.u-pem.fr/.

UNIVERSITE COTE D'AZUR

ECOLE DOCTORALE DES SCIENCES ET TECHNOLOGIES DE L'INFORMATION ET DE LA COMMUNICATION

Rapport de soutenance

d'Habilitation à Diriger les Recherches de Monsieur Guilherme DIAS DA FONSECA

Titre du mémoire : Approximation de l'appartenance à un polytope avec des applications. Approximate Polytope Membership Queries and Applications.

Membres du jury : N. Mustafa, J.D. Boissonnat, V. Chepoi, B. Martin (président), David D. Mount.

Guilherme's work is at the boundary of convex geometry, algorithmics and computational geometry.

The candidate presented optimal solutions to approximate membership queries on polytopes and numerous applications of these algorithms and data structures. One of these prominent applications is approximate nearest neighbour searching.

The talk presented a beautiful overview of his main technical results delivered in a pedagogical manner that made it accessible to a broad audience.

With his complete and convincing answers to the questions, Guilherme proved his expertise on the subject and his broad culture in algorithmics and discrete geometry.

The quality of the results is attested by publications in the leading journals (SICOMP) and conferences (STOC, SODA, SoCG,...).

The committee unanimously and strongly delivers to Guilherme Dias da Fonseca the "Habilitation à diriger les recherches" from Université Côte d'Azur.

Signature des membres du jury : **date :**

8 JUNE 2018

V. Chepoi