A Lower Bound on the Complexity of Approximate Nearest-Neighbor Searching on the Hamming Cube

Amit Chakrabarti
Bernard Chazelle
Benjamin Gum
Alexey Lvov

Abstract

We consider the nearest-neighbor problem over the d-cube: given a collection of points in $\{0,1\}^d$, find the one nearest to a query point (in the L^1 sense). We establish a lower bound of $\Omega(\log\log d/\log\log\log d)$ on the worst-case query time. This result holds in the cell probe model with (any amount of) polynomial storage and word-size $d^{O(1)}$. The same lower bound holds for the approximate version of the problem, where the answer may be any point further than the nearest neighbor by a factor as large as $2^{\lfloor(\log d)^{1-\varepsilon}\rfloor}$, for any fixed $\varepsilon > 0$.

1 Introduction

For a variety of practical reasons ranging from molecular biology to web searching, nearest-neighbor searching has been a focus of attention lately [2–4,6–10,12–15,17,19,21,22,27]. In the applications considered, the dimension of the ambient space is usually high, and predictably, classical lines of attack based on space partitioning fail. To overcome the well-known "curse of dimensionality," it is typical to relax the search by seeking only approximate answers. Curiously, no lower bound has been established — to our knowledge — on the complexity of the approximate problem in its canonical setting, i.e., points on the hypercube. Our work is an attempt to remedy this.

We note that two recent results, due to Borodin et al. [8] and Barkol and Rabani [5] do give lower bounds on the *exact* version of the problem.

Given a *database* or *key-set* $S \subseteq \{0,1\}^d$, a *δ-approximate nearest neighbor* (*δ-ANN*) of a *query point* $x \in \{0,1\}^d$ is any key $y \in S$ such that $\|x-y\|_1 \leq \delta\|x-z\|_1$, for any $z \in S$. The parameter $\delta \geq 1$ is called the *approximation factor*[1] of the problem. Given some δ, the problem is to preprocess S so as to be able to find a *δ-ANN* of any query point efficiently. The data structure consists of a table T whose entries hold $d^{O(1)}$ bits each. This means that a point can be read in constant time. This assumption might be unrealistically

[1]Most of the literature about ANN is concerned with algorithms that achieve approximation factors close to 1 and sometimes they use the term "ε-ANN" (for small positive ε) to mean what we would call a $(1+\varepsilon)$-ANN.

generous when d is large, but note that this only strengthens our lower bound result.

Theorem 1.1 *Suppose the table T, constructed from preprocessing a database S of n points in $\{0,1\}^d$, is of size polynomial in n and d and holds $d^{O(1)}$-bit entries. Then, for any algorithm using T for δ-ANN searching, there exists some S such that the query time is $\Omega(\log\log d/\log\log\log d)$. This holds for any approximation factor $\delta \leq 2^{\lfloor(\log d)^{1-\varepsilon}\rfloor}$, for any fixed $\varepsilon > 0$.*

How good is the lower bound? First, note that the problem can be trivially solved *exactly* in constant time, by using a table of size 2^d. Moreover, recent algorithmic results [15, 17, 21], when adapted to our model of computation, show that for *constant* $\delta > 1$ there is a polynomial sized table with d-bit entries and a *randomized* algorithm that enables us to answer δ-*ANN* queries using $O(\log\log d)$ probes to the table. Although there seems to be only a small gap between this upper bound and our lower bound, the two bounds are in fact incomparable because of the randomization. An important open theoretical question regarding *ANN* searching is to extend our lower bound to allow randomization.

In this context it is worth mentioning that a stronger lower bound for *exact* nearest neighbor search is now known. Recent results of Borodin et al. [8] show that even randomized algorithms for this problem require $\Omega(\log d)$ query time in our model; Barkol and Rabani [5] improve this bound to $\Omega(d/\log n)$.

2 The Cell Probe Model

Yao's *cell probe model* [26] provides a framework for measuring the number of memory accesses required by a search algorithm. Because of its generality, any lower bound in that model can be trusted to apply to virtually any conceivable sequential algorithm. In his seminal paper [1], Ajtai established a nontrivial lower bound for predecessor queries in a discrete universe (for recent improvements, see [6, 23, 25]). Our proof begins with a similar adversarial scenario. Given a key-set of n points in $\{0,1\}^d$, a table T is built in preprocessing: its size is $(dn)^c$, for fixed (but arbitrary) $c > 0$ and each entry holds $d^{O(1)}$ bits. (For simplicity, we assume that an entry consists of exactly d bits; the proof is very easily generalized if this is number d is changed to $d^{O(1)}$.) To answer queries, the algorithm has at its disposal an infinite supply of functions f_1, f_2, etc. Given a query x, the algorithm evaluates the index $f_1(x)$ and looks up the table entry $T[f_1(x)]$. If $T[f_1(x)]$ is a δ-*ANN* of x, it can stop after this single round. Otherwise, it evaluates $f_2(x, T[f_1(x)])$ and looks up the entry $T[f_2(x, T[f_1(x)])]$. Again it stops if it this entry is the desired answer (at a cost of two rounds), else it goes on in this fashion. The query time of the algorithm is defined to be the maximum number of rounds, over all queries $x \in \{0,1\}^d$, required to find a δ-*ANN* of x in the table. Note that we do not charge the algorithm for the time it takes to compute the functions f_r or for the time it takes to decide whether or not to stop. Note also that we require the last entry of T fetched by the algorithm to be the answer that it will give; this might seem like an artificial requirement but it aids our proof and adds at most one to the query time.

We couch our cell-probe arguments in a communication-complexity setting as we model our adversarial lower bound proof as a game between Bob and Alice [11, 20]. The algorithm is modeled by a sequence of functions $f_1, f_2, \ldots$. Alice starts out with a set $P_1 \subseteq \{0,1\}^d$ of candidate queries and Bob holds a collection $K_1 \subseteq 2^{\{0,1\}^d}$ of candidate key-sets, each set in K_1 being of size n. The goal of Alice and Bob is to force as many communication rounds as possible, thereby exhibiting a hard problem instance. We remark that our language differs subtly from that of Miltersen [23] and Miltersen et al. [24] who instead invent communication *problems* that can be reduced to their cell-probe data structure problems, and then prove lower bounds for the communication problems.

The possible values of $f_1(x)$ (provided by the algorithm for every x) partition P_1 into equivalence classes. Alice chooses one such class and the corresponding value of $f_1(x)$, thus restricting the set of possible queries to $P_2 \subseteq P_1$. Given this fixed value of $f_1(x)$, which Alice communicates to Bob, the entry $T[f_1(x)]$ depends only on Bob's choice of key-set. All the possible values of that entry partition K_1 into equivalence classes. Bob picks one of them and communicates the corresponding value of $T[f_1(x)]$ to Alice, thus restricting the collection of possible key-sets to $K_2 \subseteq K_1$. Alice and Bob can then iterate on this process. This produces two nested sequences of *admissible* query sets,

$$P_1 \supseteq P_2 \supseteq \cdots \supseteq P_t\,,$$

and *admissible* key-set collections,

$$K_1 \supseteq K_2 \supseteq \cdots \supseteq K_t\,.$$

An element of $P_r \times K_r$ specifies a problem instance. The set $P_r \times K_r$ is called *nontrivial* if it contains at least two problem instances with distinct answers, meaning that no point can serve as a suitable *ANN* in both instances. If $P_r \times K_r$ is nontrivial, then obviously round r is needed, and possibly others as well.

We show that for some appropriate value of $n = n(d)$, there exists an admissible starting $P_1 \times K_1$, together with a strategy for Alice and Bob, that leads to a nontrivial $P_t \times K_t$, for $t = \Theta(\log\log d/\log\log\log d)$. What makes the problem combinatorially challenging is that a greedy strategy can be shown to fail. Certainly Alice and Bob must ensure that P_r and K_r do not shrink too fast; but just as important, they must choose a low-discrepancy strategy that keeps query sets and key-sets "entangled" together. To achieve this, we adapt to our needs a technique used by Ajtai [1] for probability amplification in product spaces.[2]

3 Combinatorial Preliminaries

Before getting into the proof of our lower bound, we introduce some notation and describe a combinatorial construction that will be central to our arguments. For the rest of this paper, we assume that d is large enough and that logarithms are to the base 2. The term "distance" refers to the Hamming distance between two points in $\{0,1\}^d$. A "ball of radius r centered at x" denotes the set $\{y \in \{0,1\}^d : \mathrm{dist}(x,y) \le r\}$. To begin with, we specify the size n of the admissible key-sets and the number t of rounds, and also define two auxiliary numbers h and β:

$$h \overset{\mathrm{def}}{=} 6ct \tag{1}$$

$$\beta \overset{\mathrm{def}}{=} 16 \cdot 2^{\lfloor (\log d)^{1-\varepsilon} \rfloor} \tag{2}$$

$$t = \left\lceil \frac{\varepsilon \log\log d}{2 \log\log\log d} \right\rceil \tag{3}$$

$$n = (h-1)^{t-1} d^{5t} \tag{4}$$

The significance of the above formulae will become clear in the proofs of Lemmas 3.4–4.3. Recall that the constant c parametrizes the size of the table in the cell-probe algorithm: the table has $(dn)^c$ entries, each consisting of d bits.

The main combinatorial object we construct is a hierarchy $\mathcal{H}$ of balls or, more formally, a rooted tree $\mathcal{H}$ and a family of balls, one associated with each node of $\mathcal{H}$, such that the parent–child relation in $\mathcal{H}$ translates into the inclusion relation in the family of balls. Our proof will rely crucially on three quantities being large enough: the height of $\mathcal{H}$, the degree of a node in $\mathcal{H}$, and the minimum distance between balls associated with sibling nodes in $\mathcal{H}$.

From this main tree $\mathcal{H}$, we derive additional trees $\mathcal{H}_r$, $1 \le r \le t$; the tree $\mathcal{H}_r$ is used in the r^{th} round of communication between Alice and Bob. The tree $\mathcal{H}_1$ is a certain contraction (in the graph theoretic sense) of $\mathcal{H}$. The trees $\mathcal{H}_2, \ldots, \mathcal{H}_r$ are "nondeterministic" in the following sense: roughly speaking, $\mathcal{H}_r$ is obtained by picking a node v of $\mathcal{H}_{r-1}$, "uncontracting" v to obtain a subtree of $\mathcal{H}$, and then contracting this subtree in a different way. Notice that the choice of v determines the resulting $\mathcal{H}_r$. In the proof, we shall fix the node v (and thus determine $\mathcal{H}_r$) only during the r^{th} round of the Alice–Bob game.

[2] This simple but powerful technique, which is described in §4.3, has been used elsewhere in communication complexity, for example by Karchmer and Wigderson [18].

The sets P_i, of admissible queries, and K_i, of admissible key-sets, will be constructed based on these trees $\mathcal{H}_r$.

We shall now describe the constructions of these trees precisely. We begin with a geometric fact about the hypercube.

Definition 3.1 *A family of balls is said to be γ-separated if the distance between any two points belonging to distinct balls in the family is more than γ times the distance between any two points belonging to any one ball in the family. Here γ is any positive real quantity.*

Lemma 3.2 *Let $B \subseteq \{0,1\}^d$ be a ball of radius $k \leq d$ large enough. For any $\gamma \geq 16$ there exists a $\gamma/16$-separated family of balls within B, such that the size of the family is at least $2^{k/13}$ and the radius of each ball in the family is k/γ.*

Proof: We use an argument similar to the proof of Shannon's theorem. Let V_r be the volume of (i.e. the number of points in) a ball in $\{0,1\}^d$ of radius r, centered at a point in $\{0,1\}^d$. (Notice that this number does not depend on the center). Clearly

$$V_r = \sum_{i=0}^{\lfloor r \rfloor} \binom{d}{i}.$$

Consider the ball B', concentric with B and of radius $k/3$, and call its points initially *unmarked*. We proceed to mark the points of B' as follows: while there is an unmarked point left in B', pick one and mark all the points at distance at most $k/4$ from that point. The number N of points we pick in B' satisfies

$$N \geq \frac{V_{k/3}}{V_{k/4}}.$$

We can bound N from below:

$$N \;\geq\; \frac{V_{k/3}}{V_{k/4}} \;=\; \frac{\sum_{i=0}^{\lfloor k/3 \rfloor} \binom{d}{i}}{\sum_{i=0}^{\lfloor k/4 \rfloor} \binom{d}{i}} \;\geq\; \frac{\binom{d}{\lfloor k/3 \rfloor}}{\sum_{i=0}^{\lfloor k/4 \rfloor} \binom{d}{i}}.$$

Note that in each term of the sum in the denominator i is at most $d/4$. For such i,

$$\frac{\binom{d}{i}}{\binom{d}{i-1}} \;=\; \frac{d-i+1}{i} \;\geq\; 3,$$

so

$$\sum_{i=0}^{\lfloor k/4 \rfloor} \binom{d}{i} \;\leq\; \frac{3}{2}\binom{d}{\lfloor k/4 \rfloor}.$$

This gives

$$N \;\geq\; \frac{2}{3} \cdot \frac{\binom{d}{\lfloor k/3 \rfloor}}{\binom{d}{\lfloor k/4 \rfloor}}$$

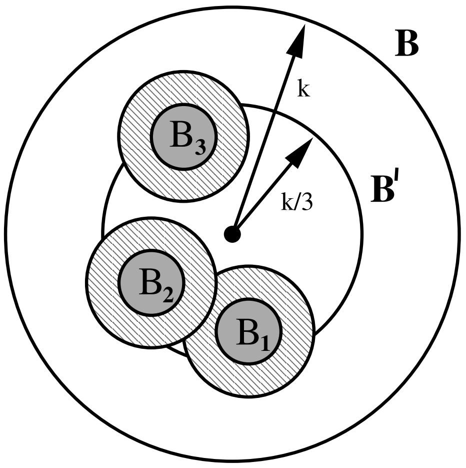

Fig. 1. Picking the separated family of balls $B_1, B_2, \ldots$. The marked points are indicated by hatching; the picked balls by solid fill.

$$\begin{aligned}
&= \frac{2}{3} \cdot \prod_{i=\lfloor k/4 \rfloor}^{\lfloor k/3 \rfloor} \frac{d-i+1}{i} \\
&\geq \frac{2}{3} \cdot \big(\underbrace{2 \times 2 \times \cdots \times 2}_{k/12-2 \text{ factors}}\big) \\
&\geq 2^{k/12-3},
\end{aligned}$$

and for large enough k, this implies $N \geq 2^{k/13}$.

Now pick balls of radius k/γ centered at the N picked points; their centers are in B' and their common radius is at most $k/16$, so these balls lie within B. Moreover, it is easy to see that they form a $\gamma/16$-separated family. To see why, suppose on the contrary that two points p and q in balls centered at distinct points p_0 and q_0 lie within $k/8$ of each other. Then,

$$\begin{aligned}
\operatorname{dist}(p_0, q_0) &\leq \operatorname{dist}(p_0, p) + \operatorname{dist}(p, q) + \operatorname{dist}(q, q_0) \\
&\leq k/\gamma + k/8 + k/\gamma \\
&\leq k/4,
\end{aligned}$$

since $\gamma \geq 16$. But this is a contradiction since, by construction, $\operatorname{dist}(p_0, q_0) > k/4$. ∎

Corollary 3.3 *For k divisible by β, there exists a $\beta/16$-separated family of radius-(k/β) balls within B, of size $2^{k/\beta}$.* ∎

Let $\mathcal{H}$ be the tree whose root is associated with the ball of radius d centered at $(0, \ldots, 0)$. The children of the root are each associated with one of the $2^{d/\beta}$ balls specified by the above corollary.[3] Their children, grandchildren, etc., are defined similarly. In general, a node of depth k (root being of depth 0) is associated with a ball of radius d/β^k and its number of children is $2^{d/\beta^{k+1}}$. We iterate this recursive construction until the leaves of $\mathcal{H}$ are of depth h^{t-1}. Note that the balls associated with the leaves of $\mathcal{H}$ are of radius at least $d/\beta^{h^{t-1}}$, and thus, by our choice of t, large enough for the application of Lemma 3.2; specifically, its corollary.

The tree $\mathcal{H}$ is used to build other trees, each one associated with a separate round. We begin with the round-one tree $\mathcal{H}_1$. Given $v \in \mathcal{H}$, let $\mathcal{H}_1^*(v)$ denote the subtree of depth h^{t-2} rooted at v. For each node v of $\mathcal{H}$ whose depth is divisible by h^{t-2}, remove from $\mathcal{H}$ all the nodes of $\mathcal{H}_1^*(v)$, except for its leaves, which we keep in $\mathcal{H}$ and make into the children of v: these operations transform $\mathcal{H}$ into a tree $\mathcal{H}_1$ of depth h. In this way, each node v of $\mathcal{H}_1$ (together with its children) forms a contraction of the tree $\mathcal{H}_1^*(v)$. We can easily check that a node of $\mathcal{H}_1$ of depth $k < h$ has exactly

$$2^{\nu d/\beta^{kh^{t-2}+1}}$$

children, where $\nu = (1 - 1/\beta^{h^{t-2}})/(1 - 1/\beta)$.

For $1 < r < t$, we define $\mathcal{H}_r$ by induction. We pick some internal node v of $\mathcal{H}_{r-1}$ and consider the tree $\mathcal{H}_{r-1}^*(v)$ of which it is the contraction. This tree now plays the role of $\mathcal{H}$ earlier: For $z \in \mathcal{H}_{r-1}^*(v)$, we let $\mathcal{H}_r^*(z)$ denote the subtree of $\mathcal{H}_{r-1}^*(v)$ of depth h^{t-r-1} rooted at z. If the depth of z in $\mathcal{H}_{r-1}^*(v)$ is divisible by h^{t-r-1}, we turn the leaves of $\mathcal{H}_r^*(z)$ into the children of z, which transforms $\mathcal{H}_{r-1}^*(v)$ into a tree of depth h that is the desired $\mathcal{H}_r$.

For $r = t$, we define $\mathcal{H}_r$ (with respect to an internal node $v \in \mathcal{H}_{r-1}$) as simply the tree formed by v and its children. We note once again that, for any $r > 1$, the definition of $\mathcal{H}_r$ is not deterministic, since the initial choice of v is left unspecified.

Lemma 3.4 *Any internal node v of any $\mathcal{H}_r$ satisfies $2^{\sqrt{d}} < \deg(\mathcal{H}_r, v) < 2^{2d/\beta}$, where $\deg(\mathcal{T}, v)$ denotes the number of children of node v in tree $\mathcal{T}$.*

Proof: Observe that $\deg(\mathcal{H}_t, v) = \deg(\mathcal{H}_{t-1}, v)$. So, it suffices to prove the lemma for $1 \le r \le t-1$. Pick any such r and consider any internal node v of $\mathcal{H}_r$: $\deg(\mathcal{H}_r, v)$ is the number of leaves of $\mathcal{H}_r^*(v)$, which itself is a subtree of $\mathcal{H}$ of depth h^{t-r-1}. So, if k is the depth of v in $\mathcal{H}$, then

$$\deg(\mathcal{H}_r, v) = \prod_{i=1}^{h^{t-r-1}} 2^{d/\beta^{k+i}}.$$

[3]To simplify the notation, we shall assume that d is a large enough power of 2. Note that β is already a power of 2.

It follows that the number $\deg(\mathcal{H}_r, v)$ is largest when $r = 1,\ k = 0$, and smallest when $r = t-1,\ k = h^{t-1} - 1$. Thus

$$\deg(\mathcal{H}_r, v) \le \prod_{i=1}^{h^{t-2}} 2^{d/\beta^i} = 2^{\nu d/\beta} < 2^{2d/\beta}.$$

On the other hand, $\deg(\mathcal{H}_r, v) \ge 2^{d/\beta^{h^{t-1}}}$, so it suffices to prove that

$$h^{t-1} \log \beta < \frac{1}{2} \log d. \tag{5}$$

But this follows after some routine algebra from (1), (3), and the fact that d is large enough. ∎

The association between balls and nodes of $\mathcal{H}_r$ is inherited from $\mathcal{H}$ in the obvious manner.

4 The Lower Bound

We now turn to the proof of the lower bound itself. Recall that the proof consists of an adversarial strategy that leads to well-structured sets P_r, of admissible queries, and K_r, of admissible key-sets. We shall require these sets to satisfy certain invariants and shall ensure that Alice's and Bob's strategies maintain these invariants. Finally, we shall show that if the invariants have been maintained for $t-1$ rounds, then the problem instance $P_t \times K_t$ is nontrivial, which would imply that round t is necessary, completing the proof.

4.1 Admissible Queries

Recall that Alice's message in round r partitions P_r into equivalence classes. This partitioning can be unwieldy, so we restrict the admissible query sets to be part of another, better-structured, nested sequence

$$P_1^* \supseteq P_2^* \supseteq \cdots \supseteq P_t^*,$$

where each $P_r^* \supseteq P_r$.

The centers of the balls at the leaves of $\mathcal{H}$ constitute the set P_1^*. For $r > 1$, we define P_r^* as the intersection of P_{r-1}^* with the balls at the leaves of $\mathcal{H}_r$. We define $P_1 = P_1^*$. For $r > 1$, Alice chooses the set P_r to be a certain subset of P_r^* according to a strategy to be specified in §4.4. For $r > 1$, we keep the set P_r of admissible queries from being too small by requiring the following:

- QUERY INVARIANT: The fraction of the leaves in $\mathcal{H}_r$ whose associated balls intersect P_r is at least $1/d$.

Note that the size of the initial collection P_1 of admissible queries is not quite as large as 2^d, although it is still a fractional power of it. Indeed,

$$|P_1| \quad = \quad (2^d)^{\frac{1-1/\beta^{h^{t-1}}}{\beta-1}}.$$

By our assumption on table size, the index $f_1(x)$ that Alice gives Bob during the first round can take on at most $(dn)^c$ distinct values. This subdivides P_1 into as many equivalence classes. The same is true at any around $r < t$, and so P_r is partitioned into the classes $P_{r,1}, \ldots, P_{r,(dn)^c}$. An internal node v of $\mathcal{H}_r$ is called *dense for* $P_{r,i}$ if the fraction of its children whose associated balls intersect $P_{r,i}$ is at least $1/d$. The node v is said to be dense if it is dense for at least one $P_{r,i}$.

Lemma 4.1 *The union of the balls associated with the dense non-root nodes of* $\mathcal{H}_r$ *contains at least a fraction* $1/2d$ *of the balls at the leaves.*

Proof: Consider one of the partitions $P_{r,i}$. Color the nodes of $\mathcal{H}_r$ whose associated balls intersect $P_{r,i}$. Further, mark every colored non-root node that is dense for $P_{r,i}$. Finally, mark every descendant in $\mathcal{H}_r$ of a marked node. For $1 \leq k \leq h$, let L_k be the number of leaves of $\mathcal{H}_r$ whose depth-k ancestor in $\mathcal{H}_r$ is colored and unmarked. (We include v as one of v's ancestors.) Let L be the number of leaves of $\mathcal{H}_r$. Clearly $L_1 \leq L$. For $k > 1$, an unmarked colored depth-k node is the child of a colored depth-$(k-1)$ node that is *not* dense for $P_{r,i}$. It follows that $L_k < L_{k-1}/d$ and so, for any $k \geq 1$, $L_k \leq L/d^{k-1}$.

Repeating this argument for all the $P_{r,i}$'s in the partition, we find that all the unmarked, colored nodes, at a fixed depth $k \geq 1$, are ancestors of at most $(dn)^c L/d^{k-1}$ leaves. In particular, the number of unmarked, colored leaves is at most

$$(dn)^c L/d^{h-1} < L/2d. \tag{6}$$

This last inequality follows from (1) and (4). Incidentally, the quantity h is defined the way it is precisely to make this inequality hold.

The query invariant ensures at least L/d colored leaves, so there are at least $L/2d$ colored, marked leaves. Moving up the tree $\mathcal{H}_r$, we find that the marked nodes whose parents are unmarked are ancestors of at least $L/2d$ leaves. All such nodes are dense, which completes the proof. ∎

4.2 Admissible Key-Sets

The collections K_r of admissible key-sets need not be specified explicitly. Instead, we define a probability distribution $\mathcal{D}_r$ over the set of all $\binom{2^d}{n}$ key-sets of size n and indicate a lower bound on the probability that a random key-set drawn from $\mathcal{D}_r$ is admissible, i.e., belongs to K_r. Beginning with the case $r = 1$, we define a random key-set S_1 recursively in terms of a random variable S_2, which itself depends on $S_3, \ldots, S_t$. To treat all these cases at once, we define S_r, for $1 \leq r \leq t$:

- For $r < t$, we define a *random S_r within $\mathcal{H}_r$* in two stages:

  [1] For each $k = 1, 2, \ldots, h-1$, choose d^5 nodes of $\mathcal{H}_r$ of depth k at random, uniformly without replacement among the nodes of depth k that are not descendants of chosen nodes of smaller depth. The $(h-1)d^5$ nodes chosen in this way are said to be *picked by S_r*.

  [2] For each node v picked by S_r, recursively choose a random S_{r+1} within the corresponding tree $\mathcal{H}_{r+1}$ (i.e., defined with respect to node v). Such a S_{r+1} is called the *canonical projection* of S_r on v. The union of these $(h-1)d^5$ projections S_{r+1} defines a *random S_r within $\mathcal{H}_r$*.

- For $r = t$, a random S_t within (some) $\mathcal{H}_t$ is obtained by selecting d^5 nodes at random, uniformly without replacement, among the leaves of the depth-one tree $\mathcal{H}_t$: S_t consists of the d^5 centers of the balls associated with these leaves.

Note that a random S_r consists of exactly $(h-1)^{t-r}d^{5(t-r+1)}$ points, thus satisfying the definition of n in (4) for the case of S_1. A random S_1 is admissible with probability one (since no information has been exchanged yet), and so the set of all S_1's constitutes K_1. Obviously, this cannot be true for $r > 1$, since for one thing S_r does not even have the right size, i.e., n.

Suppose we have defined the distribution $\mathcal{D}_{r-1}$, for some $r > 1$. As we shall see from Bob's strategy, this implies the choice of a specific $\mathcal{H}_{r-1}$. To define $\mathcal{D}_r$, we choose some node v in $\mathcal{H}_{r-1}$ (which immediately implies the choice of $\mathcal{H}_r$ for the next round). Any key-set S_1 whose construction involves choosing an S_r within the tree $\mathcal{H}_r$ associated with node v is called *v-based* and its subset formed by the corresponding S_r is called its *v-projection.*

By abuse of terminology, we say that S_r is admissible if it is a v-projection of at least one key-set of K_{r-1}: for each admissible S_r, choose one such key-set arbitrarily and call it the *v-extension* of S_r; for any other S_r, choose as its (unique) v-extension any v-based key-set whose v-projection is S_r (such a key-set is non-admissible). To define the distribution $\mathcal{D}_r$, we assign probability zero to any key-set S_1 that is not a v-extension; if it is, we assign it the probability of its v-projection with respect to the distribution of a random S_r. During round $r-1$, Bob gets to choose K_r among the key-sets with nonzero probability in $\mathcal{D}_r$.

We set a lower bound on the number of admissible key-sets by requiring Bob's strategy to enforce the following

- KEY-SET INVARIANT: A random S_r is admissible with probability at least 2^{-d^2}.

The underlying distribution is the one derived from the construction of S_r, which is also equivalent to $\mathcal{D}_r$.

In what follows, we shall need a tail estimate for the hypergeometric distribution. The next lemma provides it:

Lemma 4.2 *Consider a set of N of objects, a fraction $1/T$ of which are "good". Pick a random subset of size $m \leq N$ of these objects, all subsets of size m being equally likely, and let the random variable X denote the fraction of elements of this subset that are good. Then for any real $t > 0$ we have* $\text{Prob}[\frac{1}{T} - X \geq t] \leq e^{-2mt^2}$.

Proof: This follows directly from Theorems 1 and 4 in [16].

Lemma 4.3 *Fix an arbitrary $\mathcal{H}_r$ $(r < t)$. There exists some k_0 $(1 \leq k_0 < h)$ such that, with probability at least 2^{-d^2-1}, a random S_r within $\mathcal{H}_r$ is admissible and picks at least d^3 dense nodes of $\mathcal{H}_r$ of depth k_0.*

Proof: By Lemma 4.1, the dense non-root nodes of $\mathcal{H}_r$ are ancestors of at least a fraction $1/2d$ of the leaves. By the pigeonhole principle, for some k_0 with $1 \leq k_0 < h$, at least a fraction $1/2dh$ of the nodes of depth k_0 are dense. Of course, not all these nodes can be picked by S_r: only those that do not have ancestors that have been picked further up the tree are candidates. But this rules out fewer than hd^5 nodes, which by Lemma 3.4, represents a fraction at most $hd^5 2^{-\sqrt{d}}$ of all the nodes of depth k_0. This means that from among the set of depth-k_0 nodes that can be picked by S_r, the fraction $1/T_0$ that is dense satisfies

$$\frac{1}{T_0} \;\geq\; \frac{\frac{1}{2dh} - \frac{hd^5}{2^{\sqrt{d}}}}{1 - \frac{hd^5}{2^{\sqrt{d}}}} \;>\; \frac{1}{3dh}\,.$$

Among the d^5 nodes we pick at depth k_0, we expect at least $d^5/3dh$ of them to be dense, and thus we should exceed the lemma's target of d^3 with overwhelming probability, say, $1 - 2^{-d^2-1}$. Using Lemma 4.2 we see that this is indeed the case: choose the set of objects in the lemma to be the set of depth-k_0 nodes that are available for picking by S_r and let the "good" objects among these nodes be the dense nodes. Choose $m = d^5$, $T = T_0$ and $t = 1/T_0 - 1/d^2 > 0$. The lemma now says that the number R of dense nodes we pick satisfies

$$\text{Prob}[R \leq d^3] \;=\; \text{Prob}[R/d^5 \leq 1/d^2] \;\leq\; e^{-2d^5(\frac{1}{T_0} - \frac{1}{d^2})^2}$$

But, as observed above, $T_0 \leq 3dh$ and so, after some routine algebra, we obtain $\text{Prob}[R \leq d^3] \leq 2^{-d^2-1}$.

The key-set invariant completes the proof. ∎

4.3 Probability Amplification

In the r^{th} round, the table entry $T[f_r(x, T[f_1(x)], \ldots)]$ that Bob returns to Alice can take on at most 2^d distinct values, and so the collection of admissible key-sets is partitioned into equivalence classes $K_{r,1}, \ldots, K_{r,2^d}$. Bob has to choose one of these classes to form the new collection K_{r+1} of admissible key-sets. Unfortunately, such a large number of classes is likely to cause a violation of the key-set invariant. To amplify the probability that a random key-set is admissible back to 2^{-d^2}, we exploit the fact that the distribution is defined over a product space, and borrowing an idea from Ajtai [1], we project the distribution on its "highest-density" coordinate.

Lemma 4.4 *For $r < t$, there exists a dense node v of $\mathcal{H}_r$ such that the conditional probability that the canonical projection on v of a random S_r is admissible, given that it picks v, is at least $1/2$.*

Proof: Let D be a subset of dense nodes of depth k_0 (referred to in Lemma 4.3). We define $\mathcal{E}_D$ to be the event that the set of dense nodes of depth k_0 picked by S_r is exactly D. Let p_D be the probability that S_r is admissible and that $\mathcal{E}_D$ occurs, and let c_D be the conditional probability that S_r is admissible, given $\mathcal{E}_D$. By Lemma 4.3, summing over all subsets D of dense depth-k_0 nodes of size at least d^3, we find that

$$\sum_D c_D \cdot \text{Prob}[\mathcal{E}_D] = \sum_D p_D \geq 2^{-d^2-1},$$

and therefore $c_{D_0} \geq 2^{-d^2-1}$, for some D_0 of size at least d^3.

Now we derive a key fact from the product construction of the probability spaces for key-sets. Consider the $|D_0|$-dimensional space, where each $v \in D_0$ defines a coordinate. Each point in this space represents an S_r and is characterized by a vector $\langle n_1, \ldots, n_{|D_0|} \rangle$, where n_i is the canonical projection of S_r onto the i^{th} node of D_0. By the definition of admissibility for S_r's, if $\langle n_1, \ldots, n_{|D_0|} \rangle$ is an admissible S_r, then all the n_i's in its vector representation are admissible S_{r+1}'s. Let A_{n_i} be the set of *all* admissible S_{r+1}'s within the $\mathcal{H}_{r+1}$ corresponding to the i^{th} node of D_0. Clearly the admissible S_r's that belong to the $|D_0|$-dimensional space are all contained in

$$A_{n_1} \times \cdots \times A_{n_{|D_0|}}$$

the size of which is a fraction $\prod_{v \in D_0} c_v$ of the S_r's for which D_0 is exactly the set of dense nodes of depth k_0 picked by S_r, where c_v is the probability that a random S_{r+1} within the $\mathcal{H}_{r+1}$ corresponding to v is admissible. Because within S_r the random construction of any S_{r+1} is independent of $S_r \setminus S_{r+1}$, c_v is also the conditional probability that the canonical projection on v of a random S_r is admissible, given that it picks v. Thus we see that

$$c_{D_0} \leq \prod_{v \in D_0} c_v.$$

Since $|D_0| \geq d^3$, it follows that

$$c_v \geq \left(2^{-d^2-1}\right)^{1/|D_0|} \geq \frac{1}{2},$$

for some $v \in D_0$. ∎

4.4 Maintaining the Invariants

We summarize the strategies of Alice and Bob and discuss the enforcement of the two invariants. Skipping the trivial case $r = 1$, we show that if the invariants hold at the beginning of round $r < t$, they also hold at the beginning of round $r + 1$. Prior to round r, consider the node v from $\mathcal{H}_r$ described in Lemma 4.4. Since v is dense there is some $P_{r,i}$ such that the fraction of v's children whose associated balls intersect $P_{r,i}$ is at least $1/d$. Alice chooses such a $P_{r,i}$ and defines $P_{r,i} \cap P^*_{r+1}$ to be P_{r+1}, the set of admissible queries prior to round $r+1$. The tree $\mathcal{H}_{r+1}$ is then rooted at v, and its leaves coincide with the children of v in $\mathcal{H}_r$. Thus, the fraction of the leaves of $\mathcal{H}_{r+1}$ whose associated balls intersect P_{r+1} is at least $1/d$, and the query invariant holds.

Turning now to the key-set invariant, recall that during round r, Bob is presented with a table entry, which holds one of 2^d distinct values. By the choice of v in Lemma 4.4, the probability that a random S_{r+1} at v is admissible is at least a half. A key observation is that this is the same probability that a random key-set from $\mathcal{D}_{r+1}$ is in K_r. By the pigeonhole principle, there is a value of the table entry for which, with probability at least $(1/2)2^{-d}$, a random key-set from $\mathcal{D}_{r+1}$ is in K_r and produces a table with that specific entry value. Since $2^{-d-1} > 2^{-d^2}$, the key-set invariant holds after round r.

4.5 Forcing t Rounds

To complete the proof of Theorem 1.1, we must show that the invariants on query-sets and key-sets are strong enough to guarantee that $P_t \times K_t$ is nontrivial, i.e. that after $t - 1$ rounds, we still have at least two admissible problem instances which produce different answers. We shall soon prove that there exists at least one key-set $S \in K_t$ which picks two distinct leaves v_1 and v_2 of the tree $\mathcal{H}_t$ whose associated balls contain queries q_1 and q_2, respectively, in P_t. Notice that by construction, the family of balls associated with the leaves of $\mathcal{H}_t$ is a $\beta/16$-separated family. Since any key must lie within some ball in this family, no key can be a $\beta/16$-*ANN* for both q_1 and q_2. But (2) says that $\beta/16 = 2^{\lfloor (\log d)^{1-\varepsilon} \rfloor}$ which concludes the argument.

We prove the existence of such an S by contradiction. For any S_t let $\nu(S_t)$ denote the number of queries in P_t that it picks (which is shorthand for "the number of nodes it picks each of whose balls contains at least one query in

P_t"). Suppose that no admissible S_t picks more than one query. Then the probability p that a random S_t is admissible satisfies

$$p \leq \text{Prob}[\nu(S_t) = 0] + \text{Prob}[\nu(S_t) = 1].$$

To form a random S_t we pick d^5 leaves of $\mathcal{H}_t$ at random, uniformly. By the query invariant, at least $1/d$ of them belong to P_t. So,

$$p < \Big(1 - \frac{1}{d}\Big)^{d^5} + 2^d \Big(1 - \frac{1}{d}\Big)^{d^5 - 1} < e^{-d^4} + 2^{d+1} e^{-d^4} < e^{-d^3}.$$

By the key-set invariant, we must have $p > 2^{-d^2}$, hence a contradiction. This concludes the proof of Theorem 1.1. ∎

References

[1] M. Ajtai. A lower bound for finding predecessors in Yao's cell probe model. *Combinatorica*, 8:235–247, 1988.

[2] S. Arya and D. M. Mount. Approximate nearest neighbor searching. In *Proc. 4th Annu. ACM-SIAM Symp. Disc. Alg.*, pages 271–280, 1993.

[3] S. Arya, D. M. Mount, and O. Narayan. Accounting for boundary effects in nearest-neighbor searching. *Disc. Comput. Geom.*, 16(2):155–176, 1996.

[4] S. Arya, D. M. Mount, N. S. Netanyahu, R. Silverman, and A. Wu. An optimal algorithm for approximate nearest neighbor searching. *J. ACM*, 45(6):891–923, 1998. Preliminary version in *Proc. 5th Annu. ACM-SIAM Symp. Disc. Alg.*, pages 573–582, 1994.

[5] O. Barkol and Y. Rabani. Tighter lower bounds for nearest neighbor search and related problems in the cell probe model. In *Proc. 32nd Annu. ACM Symp. Theory Comput.*, pages 388–396, 2000.

[6] P. Beame and F. Fich. Optimal bounds for the predecessor problem. In *Proc. 31st Annu. ACM Symp. Theory Comput.*, pages 295–304, 1999.

[7] M. Bern. Approximate closest-point queries in high dimensions. *Inform. Process. Lett.*, 45(2):95–99, 1993.

[8] A. Borodin, R. Ostrovsky, and Y. Rabani. Lower bounds for high dimensional nearest neighbor search and related problems. In *Proc. 31st Annu. ACM Symp. Theory Comput.*, pages 312–321, 1999.

[9] F. Cazals. Effective nearest neighbors searching on the hyper-cube, with applications to molecular clustering. In *Proc. 14th Annu. ACM Symp. Comput. Geom.*, pages 222–230, 1998.

[10] T. Chan. Approximate nearest neighbor queries revisited. *Disc. Comput. Geom.*, 20(3):359–373, 1998. Preliminary version in *Proc. 13th Annu. ACM Symp. Comput. Geom.*, pages 352–358, 1997.

[11] B. Chazelle. *The Discrepancy Method: Randomness and Complexity.* Cambridge University Press, Cambridge, 2000.

[12] K. L. Clarkson. A probabilistic algorithm for the post office problem. In *Proc. 17th Annu. ACM Symp. Theory Comput.*, pages 175–184, 1985.

[13] K. L. Clarkson. A randomized algorithm for closest-point queries. *SIAM J. Comput.*, 17(4):830–847, 1988.

[14] K. L. Clarkson. An algorithm for approximate closest-point queries. In *Proc. 10th Annu. ACM Symp. Comput. Geom.*, pages 160–164, 1994.

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

[16] W. Hoeffding. Probability inequalities for sums of bounded random variables. *J. Amer. Stat. Assoc.*, 58(301):13–30, 1963.

[17] P. Indyk and R. Motwani. Approximate nearest neighbors: towards removing the curse of dimensionality. In *Proc. 30th ACM Symp. Theory Comput.*, pages 604–613, 1998.

[18] M. Karchmer and A. Wigderson. Monotone circuits for connectivity require super-logarithmic depth. *SIAM J. Disc. Math.*, 3(2):255–265, 1990.

[19] J. M. Kleinberg. Two algorithms for nearest neighbor search in high dimensions. In *Proc. 29th Annu. ACM Symp. Theory Comput.*, pages 599–608, 1997.

[20] E. Kushilevitz and N. Nisan. *Communication Complexity.* Cambridge University Press, Cambridge, 1997.

[21] E. Kushilevitz, R. Ostrovsky, and Y. Rabani. Efficient search for approximate nearest neighbor in high-dimensional spaces. *SIAM J. Comput.*, 30(2):457–474, 2000. Preliminary version in *Proc. 30th Annu. ACM Symp. Theory Comput.*, pages 614–623, 1998.

[22] N. Linial, E. London, and Y. Rabinovich. The geometry of graphs and some of its algorithmic applications. *Combinatorica*, 15(2):215–245, 1995. Preliminary version in *Proc. 35th Annu. IEEE Symp. Found. Comput. Sci.*, pages 577–591, 1994.

[23] P. B. Miltersen. Lower bounds for union-split-find related problems on random access machines. In *Proc. 26th Annu. ACM Symp. Theory Comput.*, pages 625–634, 1994.

[24] P. B. Miltersen, N. Nisan, S. Safra, and A. Wigderson. On data structures and asymmetric communication complexity. *J. Comput. Syst. Sci.*, 57(1):37–49, 1998. Preliminary version in *Proc. 27th Annu. ACM Symp. Theory Comput.*, pages 103–111, 1995.

[25] B. Xiao. *New Bounds in Cell Probe Model.* PhD thesis, UC San Diego, 1992.

[26] A. C. Yao. Should tables be sorted? *J. ACM*, 28(3):615–628, 1981.

[27] P. N. Yianilos. Data structures and algorithms for nearest neighbor search in general metric spaces. In *Proc. 4th Annu. ACM-SIAM Symp. Disc. Alg.*, pages 311–321, 1993.

About Authors

Amit Chakrabarti is at the Department of Computer Science, Princeton University, Princeton, NJ 08544, USA; *amitc@cs.princeton.edu.*

Bernard Chazelle is at the Department of Computer Science, Princeton University, Princeton, NJ 08544, USA; *chazelle@cs.princeton.edu*.

Benjamin Gum is at the Department of Mathematics and Computer Science, Grinnell College, Grinnell, IA 50112, USA; *gum@cs.grinnell.edu*.

Alexey Lvov is at the IBM T. J. Watson Research Center, Yorktown Heights, NY 10598, USA; *lvov@us.ibm.com*.

Acknowledgments

This work was supported in part by NSF Grant CCR-93-01254, NSF Grant CCR-96-23768, ARO Grant DAAH04-96-1-0181, and NEC Research Institute. Amit Chakrabarti's work was supported in part by a DIMACS Summer Fellowship. Benjamin Gum's work was supported in part by a National Science Foundation Graduate Fellowship.

The authors wish to thank Satish B. Rao and Warren D. Smith for interesting discussions on *ANN* searching, which inspired them to look at lower bounds for this problem. They also thank Piotr Indyk, Allan Borodin, Eyal Kushilevitz, Rafail Ostrovsky and Yuval Rabani for interesting comments, suggestions and clarifications.