

# A Strong Lower Bound for Approximate Nearest Neighbor Searching

Ding Liu<sup>a</sup>

<sup>a</sup>*Department of Computer Science, Princeton University,  
Princeton, NJ, 08544 USA  
dingliu@cs.princeton.edu*

---

## Abstract

We prove a lower bound of  $d^{1-o(1)}$  on the query time for any deterministic algorithms that solve approximate nearest neighbor searching in Yao's cell probe model. Our result greatly improves the best previous lower bound for this problem, which is  $\Omega(\frac{\log \log d}{\log \log \log d})$  [8]. Our proof is also much simpler than the proof of [8].

*Key words:* Computational Geometry; Approximate Nearest Neighbor Searching; Lower Bound

---

## 1 Introduction

Nearest neighbor searching (henceforth, NNS) is a fundamental problem in computational geometry with applications to a number of areas [14,15,13,19,5,18]. A typical setting for this problem is: given a set of  $n$  points (called a database) in a vector space of dimension  $d$  (for example,  $d$ -dimensional Euclidean space), we want to pre-process the database so that given any query point  $q$ , we can quickly find the closest point to  $q$  in the database. In low dimensions the problem is well solved [17]. On the other hand, many applications involve database points represented as vectors in high dimensional spaces. Unlike the low dimensional case, the problem is considerably more difficult in high dimensions [16,27,11,1]. To our knowledge, there is currently no algorithm with storage  $poly(n, d)$  and query time  $poly(\log n, d)$ . This is often referred to as the “curse of dimensionality”.

It is then natural to consider the *approximate* nearest neighbor searching problem (henceforth, ANNS): instead of insisting on a true closest point to the query  $q$ , we only seek a  $\beta$ -approximate nearest neighbor (henceforth, an  $\beta$ -ANN) of  $q$ , for some approximation factor  $\beta > 1$ . An  $\beta$ -ANN of  $q$  is any

point  $p$  such that  $\text{dist}(p, q) \leq \beta \cdot \text{dist}(p', q)$  for any  $p'$  in the database. There has been considerable research on ANNS [2,3,12,9,22], culminating in a couple of recent papers [21,23] that give randomized algorithms with the following guarantee: approximation factor  $\beta = 1 + \varepsilon$  for any  $\varepsilon > 0$ , space  $d^{O(1)}n^{O(1/\varepsilon^2)}$  and query time  $((d \log n)/\varepsilon)^{O(1)}$ . Thus ANNS does not suffer from the curse of dimensionality, at least from the point of view of randomized algorithms and a fixed  $\varepsilon > 0$ .

**The cell-probe model.** Several lower bound results for NNS and ANNS are known in Yao's cell-probe model [26]. The cell-probe model is a general data structure model for measuring the number of memory accesses required by a search algorithm. Given a database of  $n$  points, a table  $T$  is built in preprocessing: it consists of a set of cells with possibly different sizes. To answer a query, the algorithm makes several probes to  $T$ . Each probe is made by computing an index  $k$  and looking up the entry  $T[k]$ . The running time is defined as the maximum number of probes needed (in the worst case) to correctly answer any possible query. Because of its generality, a lower bound proved in the cell-probe model can be applied to any sequential algorithms. In the cell-probe model there are three parameters  $s$ ,  $b$  and  $t$ :  $s$  denotes the number of cells in the table;  $b$  denotes the maximum cell size in terms of the number of bits<sup>1</sup>; and  $t$  denotes the number of probes made on the table.

**Previous lower bound results.** All previous lower bounds for NNS and ANNS are proved in the Hamming cube [6,8,4]. In this setting, the database points and the query are from the Hamming cube  $C_d$  (the  $d$ -dimensional binary cube  $\{0, 1\}^d$ ), and the distance function is the Hamming distance. All previous lower bounds [6,8,4] are established in the following scenario:  $s = (nd)^{O(1)}$  and  $b = (d \log n)^{O(1)}$ , and lower bounds are given for  $t$ . They also make the assumption that  $d = \omega(\log n)$ . This assumption is necessary because for  $d = O(\log n)$ , the trivial solution of storing all possible answers uses space  $n^{O(1)}$  and makes a single probe for each query. Here we make the same assumption.

Depending on which problem is considered (NNS or ANNS), and whether the algorithm is deterministic or randomized, there are four cases.

- *NNS, deterministic.* A lower bound of  $t = \Omega(d/\log n)$  is proved by Borodin et al. [6] for any deterministic algorithm that solves NNS, when  $s = \text{poly}(n, d)$  and  $b = \text{poly}(\log n, d)$ .
- *NNS, randomized.* A lower bound of  $t = \Omega(d/\log n)$  is proved by Barkol and Rabani [4] for any randomized algorithm that solves NNS, when  $s = \text{poly}(n, d)$  and  $b = \text{poly}(\log n, d)$ .
- *ANNS, deterministic.* The only previous lower bound in this case is proved by Chakrabarti et al. [8] (and presented in much more detail in [10,7]).

---

<sup>1</sup> The cells may have different sizes. This assumption makes the model even more general.

Specifically, it is shown that  $t \geq \delta \log \log d / (2 \log \log \log d)$  when the approximation factor is at most  $2^{\lfloor (\log d)^{1-\delta} \rfloor}$ , for any constant  $\delta > 0$ .

- *ANNS, randomized.* Currently there is no lower bound known in this case.

**Our result.** We prove a new lower bound for ANNS in the deterministic case in the Hamming cube  $C_d$ . Thus the problem considered in this paper is the same as in [8], and our lower bound is significantly stronger than the previous one. Here is our result<sup>2</sup>.

**Theorem 1** *For the  $\beta$ -approximate nearest neighbor searching problem with  $n$  points in  $C_d = \{0, 1\}^d$ , any deterministic algorithm in the cell-probe model that uses  $(nd)^{O(1)}$  cells with maximum cell size  $d^{O(1)}$ , must make  $\Omega(d/(\beta^2 \log n))$  probes in the worst case. This holds for any  $\beta \leq \sqrt{d}/20$ . Specifically, for any fixed  $\varepsilon > 0$  and any  $\beta$  up to  $2^{\lfloor (\log d)^{1-\varepsilon} \rfloor}$ , the lower bound is  $d^{1-o(1)}$ .*

Our result is a significant improvement over the previous lower bound in [8], which says that, under exactly the same conditions as stated in Theorem 1, for any fixed  $\varepsilon > 0$  and any approximation factor up to  $2^{\lfloor (\log d)^{1-\varepsilon} \rfloor}$ , the query time is  $\Omega(\frac{\log \log d}{\log \log \log d})$ . Here our result implies that for any approximation factor up to  $2^{\lfloor (\log d)^{1-\varepsilon} \rfloor}$ , the query time lower bound is  $d^{1-o(1)}$ . Thus our lower bound is a strict improvement over the previous result, for any approximation factor.

Our proof is much simpler than the proof in [8], which is based on a complicated hierarchical structure. On the other hand, our proof uses the standard richness technique [25] and follows essentially the same line as the proof in [6]. Note that the approximation factor in Theorem 1 is quite generous: in fact, it is trivial to achieve  $d$ -approximation by returning an arbitrary database point.

We comment that ANNS can be solved by a *randomized* algorithm that makes only  $O(\log \log d)$  probes to a table with parameters  $s = (nd)^{O(1)}$  and  $b = (d \log n)^{O(1)}$  [21,23]. Thus our lower bound gives a sharp contrast of  $d^{1-o(1)}$  versus  $O(\log \log d)$  between the search time complexity of deterministic versus randomized algorithms for ANNS. This shows that randomization is essential in the algorithms of [21,23].

In section 2.1, we briefly explain the main idea in our proof and its relation to previous work. In section 2.2, we review the various tools and techniques used in the proof. We define the  $(\lambda, \beta)$ -approximate neighbor problem in section 2.3 and show a reduction from this problem to  $\beta$ -ANNS. Finally we prove a strong lower bound for the  $(\lambda, \beta)$ -approximate neighbor problem in section 2.4.

---

<sup>2</sup> Throughout this paper  $\log$  refers to logarithm to the base 2.

## 2 A Strong Lower Bound for Approximate Nearest Neighbor Searching

### 2.1 The main idea

Our proof is very similar to the lower bound proof for NNS by Borodin et al. [6]. The basic idea there is to consider the decision version of NNS which asks, for some given  $\lambda > 0$ , whether or not there exists a database point within distance  $\lambda$  from the query. The advantage of considering the decision version of NNS is that it has a binary communication matrix<sup>3</sup>, so that we can apply the richness technique from asymmetric communication complexity to derive a communication lower bound for the decision version. Such a communication lower bound immediately implies the desired cell probe lower bound due to a reduction in [25].

In the attempt to apply the same proof technique to ANNS, we need to define its decision version first. However, the meaning of the decision version of ANNS is not as clear as that of NNS. This is because a correct answer to an ANNS query could be any database point in the allowed approximation region. The main contribution of this paper is to exploit a certain gap regarding the distance relations that can hold among all possible queries and databases. This idea leads to the following definition of the decision version of ANNS: given  $\lambda > 0$  and an approximation factor  $\beta > 1$ , if all database points are at distance more than  $\lambda$  from the query then output 1; if there exists a database point within distance  $\lambda/\beta$  from the query then output 0; otherwise output 0 or 1 arbitrarily. Note that unlike the NNS case, there are many possible decision versions of ANNS. We prove a lower bound that holds for all of them using the framework of [6]. This lower bound then holds for ANNS. In the following sections we present the technical details.

### 2.2 Tools and techniques

This section is devoted to the tools and techniques used in the proof. Most of them have appeared in [6], and we state them here for completeness. We begin by reviewing a few useful facts. For  $0 < p < 1$ , the entropy function is  $H(p) = -p \log p - (1 - p) \log (1 - p)$ . The following fact is an estimation on the entropy function when  $p$  is close to  $1/2$ .

---

<sup>3</sup> This is a matrix of all possible problem instances of the decision version of NNS, with each row being a possible query and each column a possible database, and the entry indexed by that row and column being the answer for that particular instance.

**Fact 2** For  $x > 0$  small enough,  $H(1/2 - x) = 1 - (2 \log e)x^2 + O(x^4)$ .

A proof of this fact follows by approximating  $\log(1/2 - x)$  and  $\log(1/2 + x)$  by their respective Taylor's series expansions for small  $x > 0$ . Let  $B_d(\lambda)$  denote the Hamming ball of radius  $\lambda$  centered at an arbitrary point in  $C_d$ . We need to bound  $|B_d(\lambda)|$ , the number of points inside this Hamming ball. The following fact is probably folklore.

**Fact 3** For any  $0 < p < 1/2$ ,  $|B_d(pd)| \leq 2^{dH(p)}$ .

**Proof:** Since  $p < 1/2$ , for any  $i \leq pd$ , we have  $p^{pd}(1-p)^{(1-p)d} \leq p^i(1-p)^{d-i}$ . Thus,

$$\begin{aligned} |B_d(pd)|2^{-dH(p)} &= |B_d(pd)|p^{pd}(1-p)^{(1-p)d} \\ &= \sum_{i=0}^{pd} \binom{d}{i} p^{pd}(1-p)^{(1-p)d} \\ &\leq \sum_{i=0}^d \binom{d}{i} p^i(1-p)^{d-i} = 1. \end{aligned}$$

□

We also need the following standard Chernoff bound.

**Fact 4 (Chernoff bound [10])** For every  $a > 0$ ,  $|B_d(d/2 - a)| \leq e^{-a^2/(2d)} \cdot 2^d$ .

Another useful result is Harper's isoperimetric inequality [20,6].

**Fact 5 (Harper's isoperimetric inequality [20])** For  $A \subset C_d$ , let  $r > 0$  be such that  $A \geq |B_d(r)|$ . Then for every  $\lambda > 0$ ,  $|B_d(A, \lambda)| \geq |B_d(r + \lambda)|$ , where  $B_d(A, \lambda)$  denotes the set of cube points at distance at most  $\lambda$  from a point in  $A$ .

Next we state a relationship between *asymmetric communication complexity* and cell-probe complexity. Let a function  $f: \mathcal{X} \times \mathcal{Y} \rightarrow U$ . In the asymmetric communication complexity model there are two players, Alice and Bob. Alice gets an input  $x \in \mathcal{X}$  and Bob gets an input  $y \in \mathcal{Y}$ , and the goal is to compute  $f(x, y) \in U$  (that is, at least one player should know  $f(x, y)$  at the end of this communication game). The complexity measure is the total number of bits communicated by each side, and an  $[a, b]$ -protocol is one in which Alice sends  $a$  bits and Bob sends  $b$  bits. The following observation [24] shows that lower bounds in asymmetric communication complexity lead to lower bounds in the cell-probe model.

**Lemma 6 (Miltersen [24])** *For any function  $f$  if there is a deterministic (resp. randomized) solution in the cell probe model with parameters  $s$ ,  $b$ , and  $t$ , then there is a deterministic (resp. randomized)  $[t\lceil\log s\rceil, tb]$ -protocol for the corresponding communication problem.*

It thus suffices to show a strong lower bound for ANNS in the asymmetric communication complexity model. A general technique for this purpose is the richness technique [25]. We now briefly review it. Let  $f: \mathcal{X} \times \mathcal{Y} \rightarrow \{0, 1\}$ . We associate a communication matrix  $M_f$  with the function  $f$ . The rows of  $M_f$  are indexed by the possible inputs to Alice and the columns by the possible inputs to Bob, and the  $(x, y)$  entry of  $M_f$  is  $f(x, y)$ . Following the notations in [6], we say that a communication problem  $f$  is  $[u, v]$ -rich if  $M_f$  has at least  $v$  columns each containing at least  $u$  ones. The richness technique is given by the following lemma [25].

**Lemma 7 (Miltersen et al. [25])** *Let  $f$  be  $[u, v]$ -rich. If  $f$  has a deterministic  $[a, b]$ -protocol then  $M_f$  contains a submatrix of dimension at least  $u/2^{a+2} \times v/2^{a+b+2}$  containing only 1-entries.*

To use this lemma to prove lower bounds for  $f$  in the asymmetric communication complexity model, we need to show: (1)  $M_f$  is rich enough; (2)  $M_f$  does not contain any large 1-monochromatic submatrix.

### 2.3 The $(\lambda, \beta)$ -approximate neighbor problem

We define the  $(\lambda, \beta)$ -approximate neighbor problem by defining its communication function  $f$ . Suppose Alice has a query  $x \in C_d$  and Bob has a database  $D \in C_d^n$  (we allow repetitions in the database). Let  $h(x, D)$  be the distance between  $x$  and its nearest neighbor in  $D$ . A function  $f: x \times D \rightarrow \{0, 1\}$  is called a *valid* communication function for the  $(\lambda, \beta)$ -approximate neighbor problem if:

$$f(x, D) = \begin{cases} 1 & \text{if } h(x, D) > \lambda; \\ 0 & \text{if } h(x, D) \leq \lambda/\beta; \\ \text{arbitrary (0 or 1)} & \text{otherwise.} \end{cases}$$

Note that this definition gives a family  $\mathcal{F}$  of functions. We will prove a communication complexity lower bound that holds for *every*  $f \in \mathcal{F}$ . Before this we first show a reduction from the  $(\lambda, \beta)$ -approximate neighbor problem to  $\beta$ -ANNS.

**Fact 8** *Suppose there is a  $[a, b]$ -protocol for  $\beta$ -ANNS in the asymmetric com-*

communication complexity model, then there is a  $[a, b + d]$ -protocol for some valid communication function  $f$  of the  $(\lambda, \beta)$ -approximate neighbor problem. This is true for any  $\lambda > 0$ .

To see why, suppose we have a  $[a, b]$ -protocol for  $\beta$ -ANNS. We run it on input  $(x, D)$  and get a point  $p \in D$  on at least one side (Alice or Bob), such that  $p$  is a  $\beta$ -approximate nearest neighbor of  $x$ . In case Bob is the only one who knows  $p$ , he sends it to Alice using  $d$  bits. Thus both sides know  $p$  after a  $[a, b + d]$ -protocol. Alice compares  $H(x, p)$  with  $\lambda$  and outputs  $f(x, D) = 1$  if and only if  $H(x, p) > \lambda$ . The correctness of this protocol is justified as follows: Let  $p^*$  be a true nearest neighbor of  $x$ . If  $H(x, p^*) > \lambda$  then the protocol is supposed to output 1, it does so because  $H(x, p) \geq H(x, p^*) > \lambda$ . On the other hand if  $H(x, p^*) \leq \lambda/\beta$  then the protocol is supposed to output 0, it does so because  $H(x, p) \leq \beta H(x, p^*) \leq \lambda$ .

In view of the discussions above, a good communication lower bound for  $\beta$ -ANNS follows immediately from a good communication lower bound that applies to *every* valid communication function for the  $(\lambda, \beta)$ -approximate neighbor problem.

#### 2.4 The lower bound

Let  $f$  be any valid communication function for the  $(\lambda, \beta)$ -approximate neighbor problem. We prove the following lower bound for any asymmetric communication protocol that computes  $f$ .

**Lemma 9** *Let  $\beta \leq \sqrt{d}/20$ . There exists a  $\lambda$  such that in any communication protocol that computes  $f$ , either Alice sends  $\Omega(d/\beta^2)$  bits or Bob sends  $\Omega(nd/\beta^2)$  bits. In particular, for any fixed  $\varepsilon > 0$  and any  $\beta$  up to  $2^{\lfloor (\log d)^{1-\varepsilon} \rfloor}$ , either Alice sends at least  $d^{1-o(1)}$  bits or Bob sends at least  $nd^{1-o(1)}$  bits.*

Note that the lower bound in Lemma 9 is nearly optimal (resp. asymptotically optimal) when  $\beta = d^{o(1)}$  (resp. when  $\beta = O(1)$ ). In fact, any function of the query and the database can be computed after Alice sends her input using  $d$  bits or Bob sends his input using  $nd$  bits. Below is the proof of Lemma 9.

**Proof:** We set the relevant parameters as follows:

- $n$  and  $d$  are large enough integers such that  $d = \omega(\log n)$  and  $d$  is even.
- $\beta \leq \sqrt{d}/20$ ; it is also larger than any constant appearing in the proof.
- $\lambda = d/2 - \sqrt{2d \ln(2n)}$ ;  $\lambda_0 = \lambda/\beta$ .

We look at the communication matrix  $M_f$  that has  $2^d$  rows and  $2^{nd}$  columns.

**Claim 10**  $M_f$  is  $[2^{d-1}, 2^{nd}]$ -rich.

To see this, we fix a column (a database). For each point  $p$  in the database, the number of queries within distance at most  $\lambda$  from  $p$  is  $|B_d(\lambda)| \leq 2^{d-1}/n$  by Fact 4, thus the number of queries within distance at most  $\lambda$  from at least one point in the database is at most  $2^{d-1}$ . This means that there are at least  $2^{d-1}$  queries such that for each such a query its nearest neighbor in the chosen database is at distance more than  $\lambda$  from it. By the definition of  $f$ , all the entries indexed by these  $2^{d-1}$  queries in that fixed column are 1. Claim 10 follows. The next claim is that  $M_f$  does not contain any large 1-monochromatic submatrix.

**Claim 11** Every  $2^{d-d/(169\beta^2)} \times 2^{nd-nd/(32\beta^2)}$  submatrix of  $M_f$  contains a zero entry.

**Proof:** Consider any fixed set  $Q$  of queries of cardinality  $2^{d-d/(169\beta^2)}$ . Let  $\lambda_Q$  be the largest integer such that  $|B_d(\lambda_Q)| \leq |Q|$ . It must be the case that  $\lambda_Q < d/2 - 1$ . Otherwise we have  $|Q| = 2^{d-d/(169\beta^2)} \geq |B_d(d/2 - 1)| > 2^{d-1} - \binom{d}{d/2} \geq 2^{d-2}$ , a contradiction with  $\beta < \sqrt{d}/20$ .

Let  $p = (\lambda_Q + 1)/d$ , then  $0 < p < 1/2$ . By Fact 3,  $|Q| < |B_d(\lambda_Q + 1)| \leq 2^{dH(p)}$ . We then have  $1 - 1/(169\beta^2) < H(p)$ . This shows that  $H(p)$  is close to 1 for  $\beta$  large enough, and hence  $p$  is close to  $1/2$ . Applying Fact 2 we have  $1 - (1/2 - p)^2 > H(p) > 1 - 1/(169\beta^2)$ , and so

$$1/2 - p < 1/(13\beta). \tag{1}$$

We now consider the set  $B_d(Q, \lambda_0)$ . Recall that  $B_d(Q, \lambda_0) = \cup_{q \in Q} B_d(q, \lambda_0)$ . We say that a point  $p \in C_d$  is *good* for  $Q$  if  $p \notin B_d(Q, \lambda_0)$ . We say that a database  $D$  is *good* for  $Q$  if every point in  $D$  is good for  $Q$ . Note that, from the above definitions, if a database  $D$  is *not good* for  $Q$ , then there must exist  $p \in D$  and  $q \in Q$  such that  $H(p, q) \leq \lambda_0$ . Since  $f$  is a valid communication function, then by its definition (also recall  $\lambda_0 = \lambda/\beta$ ),  $f(q, D) = 0$ . This shows that once we pick a column indexed by a “not good” database for  $Q$ , there exists a zero entry in the  $|Q| \times 1$  submatrix formed by  $Q$  and that column, and this is true for any  $Q$ . In the following we show that for any  $Q$  of cardinality  $2^{d-d/(169\beta^2)}$ , the number of good databases for  $Q$  is less than  $2^{nd-nd/(32\beta^2)}$ . It then follows immediately that every  $2^{d-d/(169\beta^2)} \times 2^{nd-nd/(32\beta^2)}$  submatrix contains at least one zero entry.

We first bound the number of points in  $C_d$  that are “not good” for  $Q$ . Clearly this number is  $|B_d(Q, \lambda_0)|$  which is at least  $|B_d(\lambda_Q + \lambda_0)|$  by Fact 5. We have:

$$|B_d(\lambda_Q + \lambda_0)| \geq |B_d(pd - 1 + d/(3\beta))|$$

$$\begin{aligned}
&= |B_d(d/2 + d/(3\beta) - (1/2 - p)d - 1)| \\
&\geq |B_d(d/2 + d/(3\beta) - d/(13\beta) - 1)| \\
&\geq |B_d(d/2 + d/(4\beta))|.
\end{aligned}$$

The first inequality above follows from the fact that  $\lambda_0 = \lambda/\beta > d/(3\beta)$ , and the second follows from inequality 1. So the number of good points for  $Q$  is at most:

$$\begin{aligned}
2^d - |B_d(d/2 + d/(4\beta))| &\leq |B_d(d/2 - d/(4\beta))| \\
&\leq e^{-d/(32\beta^2)} 2^d \\
&< 2^{d-d/(32\beta^2)}.
\end{aligned}$$

The second inequality above follows from Fact 4. Since each good database has  $n$  good points, it is immediate that the number of good databases for  $Q$  is less than  $2^{nd-nd/(32\beta^2)}$ . This completes the proof of Claim 11.  $\square$

Combining Claim 10, Claim 11, Lemma 7 we then prove Lemma 9.  $\square$

In view of Fact 8, we have the same lower bound for the asymmetric communication complexity of  $\beta$ -ANNS. Finally by applying Lemma 6 we then establish Theorem 1.

### 3 Acknowledgments

The author would like to thank Paul Beame for pointing out the current form of Fact 3; thank Bernard Chazelle and Amit Chakrabarti for helpful discussions; and thank several anonymous referees for helpful comments and suggestions.

### References

- [1] P.K. Agarwal, J. Matoušek. Ray shooting and parametric search. In *Proc. 24th Annual ACM Symposium on the Theory of Computing*, pages 517-526, 1992.
- [2] S. Arya, and D. Mount. Approximate nearest neighbor searching. In *Proc. 4th Annual ACM-SIAM Symposium on Discrete Algorithms*, pages 271-280, 1993.
- [3] S. Arya, D. Mount, N. Netanyahu, R. Silverman, and A. Wu. An optimal algorithm for approximate nearest neighbor searching in fixed dimensions. In

- Proc. 5th Annual ACM-SIAM Symposium on Discrete Algorithms*, pages 573-582, 1994.
- [4] O. Barkol, and Y. Rabani. Tighter lower bounds for nearest neighbor search and related problems in the cell probe model. In *Proc. 32nd Annual ACM Symposium on the Theory of Computing*, pages 388-396, 2000.
  - [5] J.S. Beis, and D.G. Lowe. Shape indexing using approximate nearest-neighbor search in high-dimensional spaces. In *Proc. IEEE Conf. Comp. Vision Patt. Recog.*, pages 1000-1006, 1997.
  - [6] A. Borodin, R. Ostrovsky, and Y. Rabani. Lower bounds for high dimensional nearest neighbor search and related problems. In *Proc. 31st Annual ACM Symposium on the Theory of Computing*, pages 312-321, 1999.
  - [7] A. Chakrabarti. *Limitations of Non-Uniform Computational Models*. Ph.D. Thesis, Computer Science Department, Princeton University, 2002.
  - [8] A. Chakrabarti, B. Chazelle, B. Gum, and A. Lvov. A lower bound on the complexity of approximate nearest-neighbor searching on the Hamming cube. In *Proc. 31st Annual ACM Symposium on the Theory of Computing*, pages 305-311, 1999.
  - [9] T.M. Chan. Approximate nearest neighbor queries revisited. In *Proc. 13th Annual ACM Symposium on Computational Geometry*, pages 352-358, 1997.
  - [10] B. Chazelle. *The Discrepancy Method: Randomness and Complexity*. Cambridge University Press, Cambridge, 2000.
  - [11] K. Clarkson. A randomized algorithm for closest-point queries. *SIAM J. on Computing*, 17:830-847, 1988.
  - [12] K. Clarkson. An algorithm for approximate closest-point queries. In *Proc. 10th Annual ACM Symposium on Computational Geometry*, pages 160-164, 1994.
  - [13] S. Cost, and S. Salzberg. A weighted nearest neighbor algorithm for learning with symbolic features. *Machine Learning*, 10:57-67, 1993.
  - [14] T.M. Cover, and P.E. Hart. Nearest neighbor pattern classification. *IEEE-IT*, 13:21-27, 1967.
  - [15] L. Devroye, and T.J. Wagner. Nearest neighbor methods in discrimination. In *Handbook of Statistics*, Vol. 2, P.R. Krishnaiah and L.N. Kanal eds. North Holland, 1982.
  - [16] D. Dobkin, and R. Lipton. Multidimensional search problems. *SIAM J. on Computing*, 5:181-186, 1976.
  - [17] H. Edelsbrunner. *Algorithms in Combinatorial Geometry*. Springer-Verlag, 1987.
  - [18] R. Fagin. Fuzzy queries in multimedia database systems. In *Proc. 17th ACM SIGACT-SIGMOD-SIGART Symposium on Principles of Database Systems*, pages 1-10, 1998.

- [19] M. Flickner, H. Sawhney, W. Niblack, J. Ashley, Q. Huang, B. Dorn, M. Gorkani, J. Hafner, D. Lee, D. Petkovic, D. Steele, and P. Yanker. Query by image and video content: the QBIC system. *IEEE Computer*, 28:23-32, 1995.
- [20] L. Harper. Optimal numberings and isoperimetric problems on graphs. *Journal of Combinatorial Theory*, 1:385-394, 1966.
- [21] P. Indyk, R. Motwani. Approximate nearest neighbors: Towards removing the curse of dimensionality. In *Proc. 30th Annual ACM Symposium on the Theory of Computing*, pages 604-613, 1998.
- [22] J.M. Kleinberg. Two algorithms for nearest-neighbor search in high dimensions. In *Proc. 29th Annual ACM Symposium on the Theory of Computing*, pages 599-608, 1997.
- [23] E. Kushilevitz, R. Ostrovsky, and Y. Rabani. Efficient search for approximate nearest neighbor in high dimensional spaces. *SIAM J. on Computing*, 30(2):457-474, 2000.
- [24] P.B. Miltersen. Lower bounds for union-split-find related problems on random access machines. In *Proc. 26th Annual ACM Symposium on the Theory of Computing*, pages 625-634, 1994.
- [25] P.B. Miltersen, N. Nisan, S. Safra, and A. Wigderson. On data structures and asymmetric communication complexity. *J. of Computer and System Sciences*, 57(1):37-49, 1998.
- [26] A.C. Yao. Should tables be sorted. *J. of the ACM*, 28(3):615-628, 1981.
- [27] A.C. Yao, and F.F. Yao. A general approach to  $d$ -dimension geometric queries. In *Proc. 17th Annual ACM Symposium on the Theory of Computing*, pages 163-168, 1985.