Faster Deterministic and Las Vegas Algorithms for Offline Approximate Nearest Neighbors in High Dimensions

Josh Alman* Timothy M. Chan† Ryan Williams‡

Abstract

We present a *deterministic*, truly subquadratic algorithm for offline $(1+\varepsilon)$-approximate nearest or farthest neighbor search (in particular, the closest pair or diameter problem) in Hamming space in any dimension $d \leq n^{\delta}$, for a sufficiently small constant $\delta > 0$. The running time of the algorithm is roughly $n^{2-\varepsilon^{1/2+O(\delta)}}$ for nearest neighbors, or $n^{2-\Omega(\sqrt{\varepsilon}/\log(1/\varepsilon))}$ for farthest. The algorithm follows from a simple combination of expander walks, Chebyshev polynomials, and rectangular matrix multiplication.

We also show how to eliminate errors in the previous Monte Carlo randomized algorithm of Alman, Chan, and Williams [FOCS'16] for offline approximate nearest or farthest neighbors, and obtain a *Las Vegas* randomized algorithm with expected running time $n^{2-\Omega(\varepsilon^{1/3}/\log(1/\varepsilon))}$.

Finally, we note a simplification of Alman, Chan, and Williams' method and obtain a slightly improved *Monte Carlo* randomized algorithm with running time $n^{2-\Omega(\varepsilon^{1/3}/\log^{2/3}(1/\varepsilon))}$.

As one application, we obtain improved deterministic and randomized $(1+\varepsilon)$-approximation algorithms for MAX-SAT.

1 Introduction

We consider the well-known *approximate nearest neighbor search* (ANN) problem in high dimensions: preprocess a set B of n points in d-dimensional space so that given a query point r, a point $b \in B$ can be found that is within a factor $1+\varepsilon$ of the closest distance to r. It is hard to overstate the importance of the problem, which has a wide range of applications, from databases to machine learning. We will concentrate on the case of Hamming space $\{0,1\}^d$, as known embedding techniques can reduce, for example, the ℓ_1 or ℓ_2 metric case to the Hamming case, even deterministically [10, 28].

Deterministic offline ANN. Standard techniques for high-dimensional ANN [10], such as *locality-sensitive hashing* (LSH) [23, 21, 8, 9, 12, 11] and *dimensionality reduction* [30, 23, 33], all rely on Monte Carlo randomization. A fundamental question is whether these techniques can be efficiently derandomized. Finding Las Vegas randomized algorithms with comparable performance is already a nontrivial problem, and has been the subject of several recent papers [38, 2, 43]. Deterministic algorithms seem even more challenging. A deterministic algorithm with subquadratic preprocessing and sublinear query time was given by Indyk [27], but only for computing $(3+\varepsilon)$-approximations.

In this paper, we focus on the *offline* (or *batched*) setting, where a set R of n query ("red") points is given in advance, along with a set B of n data ("blue") points. The offline problem is sufficient for many applications, for example, computing the (monochromatic or bichromatic) closest pair. At the end of his SODA 2000 paper [27], Indyk explicitly raised the question of finding a truly subquadratic deterministic $(1+\varepsilon)$-approximation algorithm for computing closest (and farthest) pairs.

Our main result is a deterministic algorithm for offline $(1+\varepsilon)$-approximate nearest neighbor search in Hamming space, running in $n^{2-\varepsilon^{1/2+O(\delta)}}$ time for any dimension $d \leq n^{\delta}$ for a sufficiently small constant $\delta > 0$. The running time almost matches a previous randomized Monte Carlo algorithm for approximate closest pair or offline ANN, by G. Valiant [42] (although Valiant's result was later superseded by Alman, Chan, and Williams [5]).

Our algorithm consists of two parts:

(i) Solving the "main" case where the closest pair distance is not too small, and

(ii) Reducing the general case to the main case.

In part (i), we solve the main case using Chebyshev polynomials and rectangular matrix multiplication, as in previous Monte Carlo algorithms by Valiant and Alman et al. It has already been observed [5, Remark 3] that such techniques can yield a deterministic algorithm with running time $n^{2-\Omega(\sqrt{\varepsilon}/\log(\frac{d}{\varepsilon \log n}))}$. The fraction $\frac{d}{\varepsilon \log n}$ can be made small by applying dimensionality reduction techniques [33] to bring d down to

*CSAIL and Department of Electrical Engineering and Computer Science, MIT (jalman@mit.edu). Work supported in part by NSF CCF-1651838 and NSF CCF-1741615.

†Department of Computer Science, University of Illinois at Urbana-Champaign (tmc@illinois.edu). Work supported in part by NSF Grant CCF-1814026.

‡CSAIL and Department of Electrical Engineering and Computer Science, MIT (rrw@mit.edu). Work supported in part by NSF Grant CCF-1741615.

$O((1/\varepsilon)^2 \log n)$; however, dimensionality reduction requires randomization! For superlogarithmic dimensions, further ideas are needed.

The main new idea we propose is to use *expander walks*. Random walks in expander graphs are well-studied in theoretical computer science (e.g., see [26, 41]). Our application to derandomizing ANN is simple in hindsight—simple enough (at least in a warm-up version without Chebyshev polynomials) to provide a clean "textbook" application of expander walks. It may not be obvious that the expander walk approach can be combined with Chebyshev polynomials, but a careful reexamination of known analyses of such walks [7] shows that this is indeed possible.

Part (ii), reducing the general case to the main case (i.e., "densification" to increase the distance threshold), was already done implicitly, in Indyk's deterministic $(3+\varepsilon)$-approximation algorithm [27]. He used a pairwise-independent family of hash functions, together with error-correcting codes. However, the dependence on ε gets worse with the reduction (we would lose an entire factor of ε in the exponent). Here we describe an improved reduction using k-wise independence for derandomization (so only a factor of $\varepsilon^{O(\delta)}$ is lost).

It should be noted that Karppa et al. [32] has given a deterministic algorithm for a similar problem they called *outlier correlations*, which can probably be used to solve part (i), but they obtained time bounds of the form $n^{2-\Omega(\varepsilon)}$, which is worse than ours. (Karppa et al. did not explicitly consider finding closest pairs with $1+\varepsilon$ approximation factor for arbitrary point sets, and thus did not address part (ii) at all.) Their method also used expanders, but their description appears more complicated.

Our techniques are also applicable to $(1+\varepsilon)$-approximate offline farthest neighbor search, and in particular, computing the farthest pair, i.e., *diameter*. The deterministic running time is in fact slightly better $(n^{2-\Omega(\sqrt{\varepsilon}/\log(1/\varepsilon))})$ here.

Las Vegas offline ANN. If the goal is to just eliminate errors in the output, better results are possible with randomized Las Vegas algorithms. As mentioned, there were a series of papers on turning LSH into Las Vegas algorithms by Pagh (SODA 2016) [38], Ahle (FOCS 2017) [2], and Wei (SODA 2019) [43], but any (data-oblivious or data-dependent) LSH-based method requires at least $n^{2-\Theta(\varepsilon)}$ time [34, 37] to answer n queries.

For offline approximate nearest (or farthest) neighbor search, we show how to obtain a Las Vegas algorithm with $n^{2-\Omega(\varepsilon^{1/3}/\log(1/\varepsilon))}$ running time, matching our earlier Monte Carlo result [5]. Not only is the time bound better than LSH for ε sufficiently small, but the approach is also less involved than the previous Las-Vegas-ification approaches for LSH [38, 2, 43]. Essentially, we show that the simple idea of using random partitions instead of random samples, as first suggested by Indyk [27] (and also used in part in subsequent methods [38, 2, 43]), is compatible with the polynomial method from [5], after some technical modifications.

Monte Carlo offline ANN. Finally, returning to Monte Carlo algorithms, we reexamine Alman, Chan, and Williams' method and observe a small improvement of the running time to $n^{2-\Omega(\varepsilon^{1/3}/\log^{2/3}(1/\varepsilon))}$, which is currently the best for ε sufficiently small. The improvement may be minor, but the approach simplifies one main part of Alman et al.'s probabilistic polynomial construction, using an idea reminiscent to LSH, interestingly.

An application: MAX-SAT approximation. Our improved polynomial constructions have other applications beyond approximate nearest or farthest neighbors. For example, one application is to MAX-SAT, finding an assignment satisfying a maximum number of clauses in a given CNF formula with n variables and C clauses. We obtain an $(1+\varepsilon)$-approximation algorithm running in $O^*((2-\Omega(\sqrt{\varepsilon}/\log(1/\varepsilon)))^n)$ deterministic time, and $O^*((2-\Omega(\varepsilon^{1/3}/\log^{2/3}(1/\varepsilon)))^n)$ randomized Monte Carlo time, where the O^* notation hides polynomial factors in n and C. Previously, a randomized $(1+\varepsilon)$-approximation algorithm for MAX-k-SAT running in $O^*((2-\Omega(\varepsilon/k))^n)$ time was given by Hirsch [25], which was improved by the deterministic algorithms by Escoffier, Paschos, and Tourniaire [22] running in $O^*((2-\Omega(\varepsilon))^n)$ time, which in turn are improved by our results here when ε is sufficiently small.

Our deterministic algorithm for MAX-SAT shows that the problem of approximating MAX-SAT has a fine-grained reduction (with *no* increase in variables) to approximating MAX-LIN (the problem of optimally satisfying XOR constraints); the latter can be easily solved using a red-blue farthest neighbor algorithm.

2 Preliminaries: ANN via the Polynomial Method

Our algorithms make use of the "polynomial method in algorithm design," a technique used in many recent works on all-pairs shortest paths, the orthogonal vectors problem, exact and approximate nearest neighbor search, and related problems [46, 45, 1, 6, 16, 5, 15, 17, 4]. For two sets $R, B \subseteq \{0,1\}^d$ each of size n, a value t, and $\varepsilon > 0$, consider the decision version of the approximate closest pair problem: find a pair $(r,b) \in R \times B$ with Hamming distance at most $(1+\varepsilon)t$,

or conclude that all pairs have Hamming distance more than t. We design a multivariate nonnegative polynomial $P_{\le t}^{(d,s,\varepsilon)} : \{0,1\}^d \to \mathbb{R}_{\ge 0}$, such that

- if $x_1+\cdots+x_d > (1+\varepsilon)t$, then $P_{\le t}^{(d,s,\varepsilon)}(x_1,\ldots,x_d) \le 1$;
- if $x_1 + \cdots + x_d \le t$, then $P_{\le t}^{(d,s,\varepsilon)}(x_1,\ldots,x_d) > s$.

(In other words, $P_{\le t}^{(d,s,\varepsilon)}$ is a "polynomial threshold function" representation [5] of an *approximate (unweighted) threshold predicate*; for example, for $t = d/2$, it is an approximate majority.) Let $P'(x_1,\ldots,x_d,y_1,\ldots,y_d) = P_{\le t}^{(d,s,\varepsilon)}((x_1-y_1)^2,\ldots,(x_d-y_d)^2)$. Then for any two sets of $\sqrt{s}$ points X and Y in $\{0,1\}^d$, we can solve the approximate closest pair decision problem for X and Y by computing $P''(X,Y) = \sum_{x\in X}\sum_{y\in Y} P'(x,y)$. If all pairs in $X \times Y$ have Hamming distance more than $(1+\varepsilon)t$, then $P''(X,Y) \le s$. If some pair has distance at most t, then $P''(X,Y) > s$.

Suppose $P_{\le t}^{(d,s,\varepsilon)}$ has m monomials and degree q. Then P' has $m' \le 3^q m$ monomials, so we can write P' in the form

$$P'(x,y) = \sum_{\ell=1}^{m'} c_\ell \cdot \left(\prod_{i=1}^{d} x_i^{a_{i,\ell}}\right) \cdot \left(\prod_{j=1}^{d} y_j^{b_{j,\ell}}\right).$$

Defining functions $f, g : \{0,1\}^d \to \mathbb{Z}^{m'}$ by $f(x)[\ell] := c_\ell \cdot \prod_{i=1}^d x_i^{a_{i,\ell}}$ and $g(y)[\ell] := \prod_{j=1}^d y_j^{b_{j,\ell}}$, we see that $P'(x,y) = \langle f(x), g(y)\rangle$. Thus by letting $f(X) := \sum_{x\in X} f(x)$ and $g(Y) := \sum_{y\in Y} g(y)$, we have $P''(X,Y) = \langle f(X), g(Y)\rangle$. Our algorithm thus proceeds by partitioning the input R (resp. B) into $n/\sqrt{s}$ sets $X_1,\ldots,X_{n/\sqrt{s}}$ (resp. $Y_1,\ldots,Y_{n/\sqrt{s}}$) of size $\sqrt{s}$, then computing $\langle f(X_i), g(Y_j)\rangle$ for all $i,j \in [n/\sqrt{s}]$ using fast rectangular matrix multiplication:

LEMMA 2.1. ([20]; SEE ALSO [45]) *For all sufficiently large N, multiplication of an $N \times N^{0.172}$ matrix with an $N^{0.172} \times N$ matrix can be done in $O(N^2 \log^2 N)$ arithmetic operations over any field.*

Setting s so that $3^q m \le (n/\sqrt{s})^{0.172}$, we obtain a final running time of $\widetilde{O}(n^2/s)$[1] (all intermediate numbers will have polylogarithmically many bits).

In most applications of the polynomial method, the number m of monomials is typically bounded using the degree q of the polynomial: Since the inputs are only 0/1, we may assume $P_{\le t}^{(d,s,\varepsilon)}$ is a *multilinear* polynomial (i.e., $a_{i,\ell}, b_{j,\ell} \in \{0,1\}$ for all i,j,ℓ). Hence for $q \le d/2$, the polynomial has $m \le \sum_{i=0}^q \binom{d}{i} \le O(d/q)^q$ monomials.[2]

However, a key message of this paper is that we can sometimes get better algorithms by *optimizing the number m of monomials directly*, instead of optimizing just the degree.

Once the approximate decision problem has been solved, we can solve the approximate closest pair problem by binary search (or more simply, linear search over the logarithmically many powers of $1+\varepsilon$). Offline ANN can be solved in a similar way, for example, by not dividing R into groups, but dividing B into n/s groups of size s (resulting in the multiplication of an $n \times m'$ and $m' \times (n/s)$ matrix, which takes $\widetilde{O}(n^2/s)$ time provided that $3^q m \le (n/s)^{0.172}$). Alternatively, there is a direct reduction from offline ANN to approximate closest pair [6, Theorem 4.4].

When designing randomized algorithms, it suffices to use a probabilistic polynomial that has small error probability ($O(1/s)$) on every fixed input. A *probabilistic polynomial* $P : \{0,1\}^d \to \mathbb{R}$ is a distribution on d-variate polynomials over the integers. We will abuse notation and write P for both the probabilistic polynomial and a polynomial drawn from the distribution. We say P has degree at most q if all polynomials in the support of P have degree at most q, and similarly for the number of monomials. We similarly define a probabilistic *pair* of polynomials as a joint distribution on pairs of polynomials.

When designing Las Vegas randomized algorithms in particular, our idea is to impose extra conditions on the probabilistic polynomial—that if the output value lies in a certain range (e.g., $[0,1]$), correctness of the answer is guaranteed, but if the output value is outside the range (which will occur with low probability), the answer may be erroneous. A similar strategy was used in some probabilistic polynomial constructions over the integers by Beigel et al. [13] and Tarui [40].

3 Deterministic Algorithms

In this section, we present a deterministic algorithm for offline $(1+\varepsilon)$-approximate nearest neighbor search in Hamming space, with running time near $n^{2-\varepsilon^{1/2+O(\delta)}}$ for all dimensions $d \ll n^\delta$ for a sufficiently small $\delta > 0$. As mentioned, it suffices to focus on the approximate decision problem: decide whether the closest pair distance, or each nearest neighbor distance, is approximately smaller than a fixed threshold $t := \alpha_0 d$.

We first solve the problem for the main case when

[1] Throughout the paper, the $\widetilde{O}$ notation hides polylogarithmic factors.

[2] This follows since, by Stirling's approximation, for $k \le n/2$, we have $\binom{n}{k} \le (en/k)^k$.

α_0 is not too small (i.e., $\alpha_0 \gg \varepsilon^{O(\delta)}$). Afterwards, we describe how to reduce the general case to this main case.

3.1 When α_0 is not too small. As explained in Section 2, the key is in the construction of a polynomial for the approximate unweighted threshold predicate. Specifically, we will prove the following theorem:

THEOREM 3.1. *Given d, s, and $\beta_0, \varepsilon \in (0,1)$, we can construct a nonnegative polynomial $P^{(d,s,\varepsilon)}_{\geq \beta_0 d} : \{0,1\}^d \to \mathbb{R}_{\geq 0}$ with $O(\sqrt{1/\varepsilon}\log s)$ degree and $ds^{O(\sqrt{1/\varepsilon}\log(1/\varepsilon\beta_0))}$ monomials, in $\widetilde{O}(ds^{O(\sqrt{1/\varepsilon}\log(1/\varepsilon\beta_0))})$ deterministic time, such that for every $x = (x_1, \ldots, x_d) \in \{0,1\}^d$,*

- *if $x_1 + \cdots + x_d \leq \beta_0 d$, then $P^{(d,s,\varepsilon)}_{\geq \beta_0 d}(x) \leq 1$;*
- *if $x_1 + \cdots + x_d > (1+\varepsilon)\beta_0 d$, then $P^{(d,s,\varepsilon)}_{\geq \beta_0 d}(x) > s$.*

* Warm-up: Derandomization with $1/\varepsilon$ dependency.

To warm up, let us consider proving a weaker version of Theorem 3.1, with $O((1/\varepsilon)\log s)$ degree and $ds^{O(1/\varepsilon)\log(1/\varepsilon\beta_0)}$ monomials.

The simplest polynomial satisfying the above properties is

$$P^{(d,s,\varepsilon)}_{\geq \beta_0 d}(x_1, \ldots, x_d) \;=\; \frac{1}{(\beta_0 d)^q}(x_1 + \cdots + x_d)^q,$$

of degree $q = \log_{1+\varepsilon} s = O((1/\varepsilon)\log s)$. However, the number of monomials is $O(\binom{d}{q}) = O(d/q)^q = s^{O((1/\varepsilon)\log(d/\log s))}$, which is too big when d is super-logarithmic. For our nearest neighbor application, dimensionality reduction can be applied first to bring d down to $O((1/\varepsilon)^2 \log s)$, making the extra $\log(d/\log s)$ factor tolerable, but this requires randomization, which we are trying to avoid.

To reduce the number of monomials, one simple way is to take a random sample of the monomials. By a Chernoff bound, it may be checked that a sample of size about $(1/\beta_0)^{O(q)} = s^{O((1/\varepsilon)\log(1/\beta_0))}$ gives good approximation with high probability, and by the union bound, this holds for *all* $x \in \{0,1\}^d$. This approach can thus prove the existence of a polynomial with a small number of monomials, but an efficient deterministic construction is not obvious. For example, by viewing a sum of monomials of degree q with equal coefficients as a q-uniform hypergraph, the problem is essentially about deterministic constructions of *pseudo-random* or *quasi-random* hypergraphs (in the sense of having bounded "discrepancy"), but known constructions that we can find in the literature [18, 24] appear too weak for our application.

We observe that derandomization actually follows from a simple application of *expander walks*! Specifically, we use the following lemma by Alon, Feige, Wigderson, and Zuckerman [7, Proposition 2.4] (the upper-bound direction was established earlier [3, 31] and can be found in textbooks [35, 41], but we need both directions in our application).

LEMMA 3.1. (Expander Walk Lemma) *Let H be a Δ-regular graph on d vertices, and let λ be the second largest eigenvalue in absolute value of the normalized adjacency matrix. Given a subset B of βd vertices and a number q, let $N(B,q)$ be the number of walks in H of length q that stays inside B. Then for any even q,*

$$|B|\Delta^q(\beta - \lambda(1-\beta))^q \;\leq\; N(B,q) \;\leq\; |B|\Delta^q(\beta + \lambda(1-\beta))^q.$$

Let H be a Δ-regular graph over vertices $\{1, \ldots, d\}$, with $\lambda = \Theta(1/\Delta^{c_0})$ for some constant $c_0 > 0$; the "ideal" value is $c_0 = 1/2$, and known explicit expander constructions can give such an H in $O(d\Delta \log^{O(1)} d)$ time for certain values of c_0 [26] (see also [39, 19]). Choose Δ so that $\lambda = \varepsilon\beta_0/3$ (i.e., $\Delta = \Theta((1/\varepsilon\beta_0)^{1/c_0})$). Let q be an even number, to be set later. For $x = (x_1, \ldots, x_d) \in \{0,1\}^d$, we define our polynomial $P^{(d,s,\varepsilon)}_{\geq \beta_0 d}$ as

$$P^{(d,s,\varepsilon)}_{\geq \beta_0 d}(x) \;=\; \frac{\sum_{\text{length-}q \text{ walk } i_0 \cdots i_q \text{ in } H} x_{i_0} \cdots x_{i_q}}{\beta_0 d \Delta^q (\beta_0 + \lambda)^q}.$$

Analysis. Suppose that $x_1 + \cdots + x_d = \beta d$. Letting $B = \{i : x_i = 1\}$, we see that $\sum_{\text{length-}q \text{ walk } i_0 \cdots i_q \text{ in } H} x_{i_0} \cdots x_{i_q}$ is precisely $N(B,q)$. By Lemma 3.1,

$$\frac{\beta}{\beta_0}\left(\frac{\beta - \lambda}{\beta_0 + \lambda}\right)^q \;\leq\; P^{(d,s,\varepsilon)}_{\geq \beta_0 d}(x) \;\leq\; \frac{\beta}{\beta_0}\left(\frac{\beta + \lambda}{\beta_0 + \lambda}\right)^q.$$

If $\beta \leq \beta_0$, then $P^{(d,s,\varepsilon)}_{\geq \beta_0 d}(x) \leq 1$. On the other hand, if $\beta > (1+\varepsilon)\beta_0$, then $P^{(d,s,\varepsilon)}_{\geq \beta_0 d}(x) \geq (1+\Omega(\varepsilon))^q$, which can be made greater than s by setting $q = \Theta((1/\varepsilon)\log s)$. The polynomial $P^{(d,s,\varepsilon)}_{\geq \beta_0 d}$ has degree $q+1$, and the number of monomials is $O(d\Delta^q) \leq d(1/\varepsilon\beta_0)^{O(q)} = ds^{O((1/\varepsilon)\log(1/\varepsilon\beta_0))}$.

Karppa et al. [32] described a similar result using expanders, but their description and analysis appear more complicated (which makes it difficult to combine with Chebyshev polynomials, as we will do next). They started with the standard expander mixing lemma (instead of expander walks) and used repeated approximate squaring, with more complex calculations.

* Derandomization with $\sqrt{1/\varepsilon}$ dependency.

To improve the degree from $O((1/\varepsilon)\log s)$ to $O(\sqrt{1/\varepsilon}\log s)$, we use Chebyshev polynomials, as in

Valiant [42] and Alman, Chan, and Williams [5]. Let T_q denote the degree-q Chebyshev polynomial of the first kind, which achieves better "gap amplification" than the more naive polynomial x^q. Specifically, the main properties we need are:

- if $|x| \leq 1$, then $|T_q(x)| \leq 1$;
- if $x \geq 1+\varepsilon$, then $T_q(x) \geq \frac{1}{2}e^{q\sqrt{\varepsilon}}$.

Chebyshev polynomials have both positive and negative coefficients; naively applying Lemma 3.1 to each term of the Chebyshev polynomial does not work. We generalize Alon et al.'s proof of Lemma 3.1 as follows:

LEMMA 3.2. (Generalized Expander Walk Lemma) *Let H be a Δ-regular graph on d vertices, and let λ be the second largest eigenvalue in absolute value of the normalized adjacency matrix. Let $Q(y) = \sum_{k=0}^{q} a_k y^k$ be a univariate degree-q polynomial over $\mathbb{R}$, and let $\breve{Q}$ be the convex envelope of Q (i.e., supremum of all convex functions below Q). Given a subset B of βd vertices and a number q, let $N(B,k)$ be the number of walks in H of length k that stay inside B. Then*

$$\begin{aligned}\min_{y \geq \beta - \lambda(1-\beta)} \breve{Q}(y) &\leq \sum_{k=0}^{q} \frac{a_k}{|B|\Delta^k} N(B,k) \\ &\leq \max_{|y| \leq \beta+\lambda(1-\beta)} Q(y).\end{aligned}$$

Proof. By direct modification of Alon et al.'s proof [7]. Let L be $1/\Delta$ times the adjacency matrix of the subgraph of H induced by B. Let $\gamma_1 \geq \gamma_2 \geq \cdots \geq \gamma_{|B|}$ be the eigenvalues of L, and $u_1, \ldots, u_{|B|}$ be the corresponding orthonormal eigenvectors. Let u be the all-1's vector, and write $u = \sum_{i=1}^{|B|} c_i u_i$. Alon et al.'s proof made use of the following observations:[3]

$$N(B,k) = \Delta^k \sum_{i=1}^{|B|} c_i^2 \gamma_i^k \tag{3.1}$$

$$\sum_{i=1}^{|B|} c_i^2 = |B| \tag{3.2}$$

$$\max_{i=1}^{|B|} |\gamma_i| \leq \beta + \lambda(1-\beta) \tag{3.3}$$

$$\frac{1}{|B|}\sum_{i=1}^{|B|} c_i^2 \gamma_i \geq \beta - \lambda(1-\beta). \tag{3.4}$$

[3] (3.1) corresponds to (2.3) in [7], (3.2) is noted immediately after (2.3), (3.3) corresponds to Lemma 2.2, and (3.4) is shown near the final paragraph in the proof of Proposition 2.4.

Let

$$\begin{aligned}Z = \sum_{k=0}^{q} \frac{a_k}{|B|\Delta^k} N(B,k) &= \frac{1}{|B|}\sum_{i=1}^{|B|} c_i^2 \sum_{k=0}^{q} a_k \gamma_i^k \\ &= \frac{1}{|B|}\sum_{i=1}^{|B|} c_i^2 Q(\gamma_i)\end{aligned}$$

(note that these are equalities, and hold regardless of the signs of the a_k's). It follows that

$$\begin{aligned}Z &\leq \frac{1}{|B|}\sum_{i=1}^{|B|} c_i^2 \cdot \max_{|y| \leq \beta+\lambda(1-\beta)} Q(y) \\ &= \max_{|y| \leq \beta+\lambda(1-\beta)} Q(y), \text{ and} \\ Z &\geq \frac{1}{|B|}\sum_{i=1}^{|B|} c_i^2 \breve{Q}(\gamma_i) \\ &\geq \breve{Q}\left(\frac{1}{|B|}\sum_{i=1}^{|B|} c_i^2 \gamma_i\right) \geq \min_{y \geq \beta-\lambda(1-\beta)} \breve{Q}(y),\end{aligned}$$

by Jensen's inequality and the convexity of $\breve{Q}$. ∎

Proof of Theorem 3.1. As before, let H be a Δ-regular graph H over vertices $\{1, \ldots, d\}$, with $\lambda = \Theta(1/\Delta^{c_0})$ for some constant $c_0 > 0$. Choose Δ so that $\lambda = \varepsilon\beta_0/3$ (i.e., $\Delta = \Theta((1/\varepsilon\beta_0)^{1/c_0})$). Let q be an even number, to be set later. Write the rescaled degree-q Chebyshev polynomial $Q(y) = \frac{1}{2}(T_q(\frac{y}{\beta_0+\lambda}) + 1)$ as $\sum_{k=0}^{q} a_k y^k$. For $x = (x_1, \ldots, x_d) \in \{0,1\}^d$, our polynomial $P_{\geq \beta_0 d}^{(d,s,\varepsilon)}$ is defined as

$$P_{\geq \beta_0 d}^{(d,s,\varepsilon)}(x) = \sum_{k=0}^{q} \frac{a_k \sum_{\text{length-}k \text{ walk } i_0 \cdots i_k \text{ in } H} x_{i_0} \cdots x_{i_k}}{\beta_0 d \Delta^k}.$$

Analysis. Suppose that $x_1 + \cdots + x_d = \beta d$. Letting $B = \{i : x_i = 1\}$, we see that $\sum_{\text{length-}k \text{ walk } i_0 \cdots i_k \text{ in } H} x_{i_0} \cdots x_{i_k}$ is precisely $N(B,k)$. By Lemma 3.2,

$$\frac{\beta}{\beta_0} \min_{y \geq \beta - \lambda} \breve{Q}(y) \leq P_{\geq \beta_0 d}^{(d,s,\varepsilon)}(x) \leq \frac{\beta}{\beta_0} \max_{|y| \leq \beta+\lambda} Q(y).$$

If $\beta \leq \beta_0$, then $P_{\geq \beta_0 d}^{(d,s,\varepsilon)}(x) \leq 1$. On the other hand, if $\beta > (1+\varepsilon)\beta_0$, then for any $y \geq \beta - \lambda$,

$$\frac{y}{\beta_0+\lambda} \geq \frac{\beta-\lambda}{\beta_0+\lambda} \geq 1 + \frac{\varepsilon\beta_0 - 2\lambda}{\beta_0+\lambda} \geq 1 + \Omega(\varepsilon),$$

and since the convex envelope $\breve{T}_q$ agrees with T_q over $[1,\infty)$, we have $P_{\geq \beta_0 d}^{(d,s,\varepsilon)}(x) \geq \frac{1}{2}T_q(1+\Omega(\varepsilon)) \geq e^{\Omega(q\sqrt{\varepsilon})}$, which can be made greater than s by setting $q =$

$\Theta(\sqrt{1/\varepsilon}\log s)$. The polynomial $P^{(d,s,\varepsilon)}_{\geq\beta_0 d}$ has degree $q+1$, and the number of monomials is $O(d\Delta^q) \leq d(1/\varepsilon\beta_0)^{O(q)} = ds^{O(\sqrt{1/\varepsilon}\log(1/\varepsilon\beta_0))}$. ∎

We can now solve the ANN problem in the main case via the polynomial method as described in Section 2, by setting $P^{(d,s,\varepsilon)}_{\leq\alpha_0 d}(x_1,\ldots,x_d) := P^{(d,s,\varepsilon\alpha_0)}_{\geq(1-(1+\varepsilon)\alpha_0)d}(1-x_1,\ldots,1-x_d)$, and applying Theorem 3.1 with ε changed to $\varepsilon\alpha_0$. The degree is $q = O(\sqrt{1/\varepsilon\alpha_0}\log s)$, and the number of monomials is $m \leq ds^{O(\sqrt{1/\varepsilon\alpha_0}\log(1/\varepsilon\alpha_0)}$ (the negation of the variables causes an increase of a factor of 2^q, which is absorbed by the bound), and we can set $s = n^{\Theta(\sqrt{\varepsilon\alpha_0}/\log(1/\varepsilon\alpha_0))}$ to ensure that $3^q m \leq (n/s)^{0.172}$.

THEOREM 3.2. *Given $d \leq n^{0.1}$ and $\alpha_0, \varepsilon \in (0,1)$, and given n red and n blue points in $\{0,1\}^d$, the following can be computed in $\widetilde{O}(n^{2-\Omega(\sqrt{\varepsilon\alpha_0}/\log(1/\varepsilon\alpha_0))})$ deterministic time: for every red point q, we can find a blue point of Hamming distance at most $(1+\varepsilon)\alpha_0 d$ from q, or conclude that no blue point has Hamming distance at most $\alpha_0 d$ from q.*

3.2 Densification to increase α_0. The bound in Theorem 3.2 is not good if the parameter α_0 is very small. To fix this issue, we provide a deterministic reduction from the general case to the case when α_0 is not too small. Indyk [27, Section 3] already (implicitly) described such a reduction, which increases α_0 to $\Omega(\varepsilon)$. His reduction consisted of two parts: (1) use pairwise-independent hash functions to map to strings over a larger alphabet, and (2) use error-correcting codes to map back to the binary alphabet. Alternatively, as noted in Andoni et al.'s survey [10], part (1) can be viewed as an unbalanced expander construction (which is quite different from our preceding expander walk approach).

Using k-wise independence instead of pairwise independence, we can improve the first step, increasing α_0 to $\Omega(\varepsilon^{1/(k-1)})$ for an arbitrarily large constant k. Let $d_H(\cdot,\cdot)$ denote the Hamming distance.

LEMMA 3.3. *Given d, an even number k, and $\alpha_0, \varepsilon \in (0,1)$, we can find a number $\alpha_0' = \Omega(\varepsilon^{1/(k-1)})$ and construct a randomized mapping $h : \{0,1\}^d \to \Sigma$ with $|\Sigma| \leq 2^{O(1/\alpha_0)}$, from a sample space of size $O(d)^k$, such that for every fixed $p, q \in \{0,1\}^d$,*

- *if $d_H(p,q) \leq \alpha_0 d$, then $\Pr_h[h(p) \neq h(q)] \leq \alpha_0'$;*
- *if $d_H(p,q) > (1+\varepsilon)\alpha_0 d$, then $\Pr_h[h(p) \neq h(q)] > (1+\Omega(\varepsilon))\alpha_0'$.*

Proof. Choose r random indices $j_1,\ldots,j_r \in [d]$ that are k-wise independent, for a parameter r to be set later. By standard constructions for k-wise independent random variables [29, 35], a sample space of size $O(d)^k$ suffices. For $p = (p_1,\ldots,p_d) \in \{0,1\}^*$, we define

$$h(p) := p_{j_1}\cdots p_{j_r},$$

with $\Sigma = \{0,1\}^r$.

Analysis. We use the following fact: if $E_1,\ldots,E_r$ are k-wise independent events with $\Pr(E_i) = \alpha$, then the probability of the event $E = \bigcup_{i=1}^r E_i$ lies in $1-(1-\alpha)^r \pm O((\alpha r)^k)$, assuming that $\alpha r < 1/2$.

This fact follows from the inclusion-exclusion formula: for even k,

$$\sum_{S\subseteq[r],1\leq|S|\leq k-1} (-1)^{|S|-1}\Pr\left(\bigcap_{i\in S} E_i\right) \leq \Pr(E) \leq \sum_{S\subseteq[r],1\leq|S|\leq k} (-1)^{|S|-1}\Pr\left(\bigcap_{i\in S} E_i\right).$$

By k-wise independence,

$$\sum_{s=1}^{k-1}\binom{r}{s}(-1)^{s-1}\alpha^s \leq \Pr(E) \leq \sum_{s=1}^{k}\binom{r}{s}(-1)^{s-1}\alpha^s.$$

Since

$$\sum_{s=1}^{r}\binom{r}{s}(-1)^{s-1}\alpha^s = -\sum_{s=1}^{r}\binom{r}{s}(-\alpha)^s = -((1-\alpha)^r - 1)$$

by the binomial theorem, and $\sum_{s=k}^r \binom{r}{s}\alpha^s \leq \sum_{s=k}^r (\alpha r)^s \leq O((\alpha r)^k)$ (assuming that $\alpha r < 1/2$), this proves the above fact.

Now, we show that the above function h satisfies the property stated in the lemma. Let E_i be the event that $p_{j_i} \neq q_{j_i}$. These events are k-wise independent, and the event $h(p) \neq h(q)$ is precisely $\bigcup_{i=1}^r E_i$. Assume $(1+\varepsilon)\alpha_0 r < 1/2$ (which will indeed be true).

- Suppose that $d_H(p,q) \leq \alpha_0 d$. Then $\Pr(E_i) \leq \alpha_0$. By the above fact, $\Pr[h(p) \neq h(q)] \leq \Pr\left(\bigcup_{i=1}^r E_i\right) \leq \alpha_0' := 1-(1-\alpha_0)^r + O((\alpha_0 r)^k)$.
- Suppose that $d_H(p,q) \geq (1+\varepsilon)\alpha_0 d$. Then $\Pr(E_i) \geq (1+\varepsilon)\alpha_0$. We can define k-wise independent events E_i'' such that E_i'' is contained in E_i and $\Pr(E_i'') = (1+\varepsilon)\alpha_0$. By the above fact applied to these events E_i'' instead, $\Pr[h(p) \neq h(q)] \geq \Pr\left(\bigcup_{i=1}^r E_i''\right) \geq \alpha_0'' := 1-(1-(1+\varepsilon)\alpha_0)^r - O((\alpha_0 r)^k)$.

Note that $\alpha_0' = \Theta(\alpha_0 r)$. Furthermore, $\alpha_0'' = 1-(1-\alpha_0)^r(1-\Omega(\varepsilon\alpha_0))^r - O((\alpha_0 r)^k) = \alpha_0' + \Omega(\varepsilon\alpha_0 r) - O((\alpha_0 r)^k) = (1+\Omega(\varepsilon))\alpha_0' - O(\alpha_0')^k$, which is $(1+\Omega(\varepsilon))\alpha_0'$ by setting r to be a small constant times $\varepsilon^{1/(k-1)}/\alpha_0$, so that α_0' is a small constant times $\varepsilon^{1/(k-1)}$. ∎

We next use known constructions of ε-balanced error-correcting codes (which follow from known constructions of ε-biased sets) [36]:

LEMMA 3.4. (Error-Correcting Codes) *Given d, an alphabet Σ, and $\delta \in (0,1)$, we can construct a mapping $g : \Sigma \to \{0,1\}^\tau$ for some $\tau = O((1/\delta)^2 \log d)$, such that for every $a, b \in \Sigma$ with $a \neq b$, we have $d_H(g(a), g(b)) \in (1 \pm \delta)\tau/2$. The construction takes $O((1/\delta)^{O(1)} \log^{O(1)} |\Sigma|)$ time.*

LEMMA 3.5. (Improved Densification Lemma) *Given d, k, and $\alpha_0, \varepsilon \in (0,1)$, we can find numbers $d' \leq O(d)^k$ and $\alpha_0' = \Omega(\varepsilon^{1/(k-1)})$ and construct a mapping $f : \{0,1\}^d \to \{0,1\}^{d'}$ which can be evaluated in $O(d)^{k+O(1)}$ deterministic time, such that for every $p, q \in \{0,1\}^d$,*

- *if $d_H(p,q) \leq \alpha_0 d$, then $d_H(f(p), f(q)) \leq \alpha_0' d'$;*
- *if $d_H(p,q) > (1+\varepsilon)\alpha_0 d$, then $d_H(f(p), f(q)) > (1 + \Omega(\varepsilon))\alpha_0' d'$.*

Proof. Define $f(p)$ to be the concatenation of $g(h(p))$ over all mappings h in the sample space of Lemma 3.3, where g is the mapping of Lemma 3.4 with $\delta := c\varepsilon$ for a small enough constant c.

Let D be the number of different hash functions h. If $d_H(p,q) \leq \alpha_0 d$, then $d_H(f(p), f(q)) \leq \alpha_0' D \cdot (1 + \delta)\tau/2$. If $d_H(p,q) > (1+\varepsilon)\alpha_0 d$, then $d_H(f(p), f(q)) > (1+\Omega(\varepsilon))\alpha_0' D \cdot (1-\delta)\tau/2$. We set $d' := D\tau/2$, and reset $\alpha_0' := \alpha_0'(1+\delta)$. ∎

Applying Lemma 3.5 and then Theorem 3.2, we immediately obtain a deterministic algorithm with running time $\widetilde{O}(n^{2-\Omega(\varepsilon^{(1/2)(1+1/(k-1))}/\log(1/\varepsilon))})$, for $d \leq n^{0.1/k}$, for the approximate decision problem for any threshold $\alpha_0 d$, and thus for offline ANN. For $d < 2^{o(\log n/\log(1/\varepsilon))}$, we may even use a nonconstant value of $k = \Theta(\log(1/\varepsilon))$.

THEOREM 3.3. *Given d and $\varepsilon \in (0,1)$, and given n red and n blue points in $\{0,1\}^d$, we can find a $(1+\varepsilon)$-approximate Hamming nearest blue point for every red point, in $\widetilde{O}(n^{2-\varepsilon^{1/2+O(\delta)}})$ deterministic time, if $d \leq n^\delta$ for $\delta \leq 0.05$.*

The running time reduces to $\widetilde{O}(n^{2-\Omega(\sqrt{\varepsilon}/\log(1/\varepsilon))})$ if $d < 2^{o(\log n/\log(1/\varepsilon))}$.

3.3 Other applications. Approximate offline Hamming farthest neighbor search (and approximate diameter) can be solved similarly. We can work directly with the polynomial $P_{\geq t}^{(d,s,\varepsilon)}$ instead of $P_{\leq t}^{(d,s,\varepsilon)}$, and can apply Theorem 3.2 with $\beta_0 := \alpha_0$ and ε unchanged, to obtain a time bound of $\widetilde{O}(n^{2-\Omega(\sqrt{\varepsilon}/\log(1/\varepsilon\alpha_0))})$. It then suffices to apply Lemma 3.5 with $k = 2$ to make $\alpha_0 = \Omega(\varepsilon)$.

THEOREM 3.4. *Given d and $\varepsilon \in (0,1)$, and given n red and n blue points in $\{0,1\}^d$, we can find a $(1+\varepsilon)$-approximate Hamming farthest blue point for every red point, in $\widetilde{O}(n^{2-\Omega(\varepsilon^{1/2}/\log(1/\varepsilon))})$ deterministic time, if $d \leq n^{0.05}$.*

The results can be extended to ℓ_1. For points in $[U]^d$, we can map each point $(x_1, \ldots, x_d)$ to the string $1^{x_1}0^{U-x_1} \cdots 1^{x_d}0^{U-x_d}$ in Hamming space $\{0,1\}^{Ud}$, while preserving distances. This may be inefficient for large U, but the universe size U can be made small. Consider the approximate decision problem of comparing the nearest neighbor distance for a query point q with a fixed value r. It is known [14] that with $O(d)$ shifted uniform grids of side length $O(dr)$, a nearest neighbor can be found in the same cell as q in one of the grids. It suffices to solve the problem inside each grid cell; in each grid cell, coordinates can be rounded to multiples of $\varepsilon r/d$, effectively reducing the universe size to $U = O(\frac{dr}{\varepsilon r/d}) = O(d^2/\varepsilon)$. (For farthest neighbors, we can round coordinates directly without shifted grids.)

The results can also be extended to ℓ_2, using Indyk's deterministic embedding [28] from ℓ_2 to ℓ_1. We therefore have the following result.

THEOREM 3.5. *Given d and $\varepsilon \in (0,1)$, and given n red and n blue points in $\mathbb{R}^d$, we can find a $(1+\varepsilon)$-approximate ℓ_1 or ℓ_2 nearest blue point for every red point, in $\widetilde{O}(n^{2-\varepsilon^{1/2+O(\delta)}})$ deterministic time, if $d \leq n^\delta$ for $\delta \leq 0.05$.*

The running time reduces to $\widetilde{O}(n^{2-\Omega(\sqrt{\varepsilon}/\log(1/\varepsilon))})$ if $d \ll 2^{o(\log n/\log(1/\varepsilon))}$.

Another application is to $(1+\varepsilon)$-approximation algorithms for MAX-SAT. Here we proceed by giving an efficient approximation-preserving reduction from MAX-SAT to MAX-LIN, and arguing that MAX-LIN approximation algorithms can be derived from approximate farthest pair algorithms.

THEOREM 3.6. *Given a CNF formula with n variables and $C \leq 2^{o(n)}$ clauses, and $\varepsilon \in (0,1)$, there is a $(1+\varepsilon)$-approximation algorithm for MAX-SAT that runs $(2 - \Omega(\sqrt{\varepsilon}/\log(1/\varepsilon)))^n \cdot C^{O(1)}$ deterministic time.*

Proof. Recall in the MAX-LIN problem, we are given a set of linear equations over $\mathbb{F}_2$ in n variables, and wish to find an assignment satisfying a maximum number of equations. First, by a known reduction in fine-grained complexity [44], we can obtain a $(2-\Omega(\alpha))^n$-time algorithm for $(1+\varepsilon)$-approximating the MAX-LIN problem directly from an $N^{2-\alpha}$-time algorithm for $(1+\varepsilon)$-approximating red-blue farthest pair on N

red and N blue points.[4] By Theorem 3.4, we can set $\alpha = \varepsilon^{1/2}/\log(1/\varepsilon)$.

Now we show how to obtain a $(1+\varepsilon)$-approximate MAX-SAT algorithm from a $(1+\varepsilon/6)$-approximate MAX-LIN algorithm, with essentially the same running time. The standard reduction from MAX-k-SAT to MAX-k-LIN increases the number of clauses by a factor of $\Omega(2^k)$, which is unacceptable for large k. To avoid this blowup, we use ε-biased sets.

For every clause $c = (\ell_1 \vee \cdots \vee \ell_w)$ of a given MAX-SAT instance instance F over the literals $\ell_1, \ldots, \ell_w$, we do the following. If $w \leq \log(n)$, then we can reduce c to a collection of MAX-w-LIN clauses in the standard way (we include all linear equations over the literals $\ell_1, \ldots, \ell_w$ that are consistent with c). This increases the number of clauses by a $\text{poly}(n)$ factor, and preserves the approximation factor (if c is not satisfied, then all new clauses are unsatisfied; if c is satisfied, then exactly $1/2$ of the new clauses are satisfied). From now on, assume $w > \log(n)$ and n is sufficiently large.

For a parameter $\delta > 0$ to be set later, deterministically construct an δ-biased set $S = \{v_1, \ldots, v_t\} \subset \{0,1\}^w$ of size $t = \text{poly}(w, 1/\delta)$ ([36]), and replace the clause c with the t XOR constraints

$$\sum_{i=1}^{w} \ell_i \cdot v_j[i] = 1 \bmod 2$$

for all $j = 1, \ldots, t$ (Note the $v_j[i]$ are all 0/1 constants, so the XOR constraints are all over the original literals ℓ_j.) Call the obtained MAX-LIN instance F'.

By the properties of small-biased sets, we have: for every variable assignment A, if A satisfies a clause in F then it satisfies between $1/2 - \delta$ and $1/2 + \delta$ of the corresponding XOR constraints in F'. (If A does not satisfy the clause, it satisfies none of the corresponding XOR constraints.) Therefore if A satisfies a ρ-fraction of clauses in F, then the fraction of XOR constraints ρ' satisfied by A in F' is in the interval $[\rho(1/2 - \delta), \rho(1/2 + \delta)]$. Moreover, if A satisfies a ρ'-fraction in F', then it satisfies a ρ-fraction in F, where $\rho \in [\rho'/(1/2+\delta), \rho'/(1/2-\delta)]$. Let $\rho_{\max}$ and $\rho'_{\max}$ be the maximum fraction of constraints satisfiable in F and F', respectively, and note that $\rho_{\max}(1/2 - \delta) \leq \rho'_{\max} \leq \rho_{\max}(1/2 + \delta)$.

[4] Divide the n variables of the MAX-LIN instance into two halves, enumerate all $N = O(2^{n/2})$ partial assignments on both halves, and set up a red-blue farthest pair instance on N red points (from one half) and N blue points (from the other half) such that each red-blue pair has Hamming distance equal to the number of XOR constraints satisfied by the corresponding (full) variable assignment. Then, $(1+\varepsilon)$-approximations to the farthest pair are $(1+\varepsilon)$-approximations to the optimum for the MAX-LIN instance.

Suppose we have an algorithm that $(1+\delta)$-approximates MAX-LIN: given F', it outputs an assignment $A^\star$ satisfying a fraction of constraints $\rho' \geq \rho'_{\max}/(1+\delta)$. Therefore $A^\star$ also satisfies a ρ-fraction of clauses in F, where $\rho \geq \rho'_{\max}/((1/2+\delta)(1+\delta))$. Therefore $\rho \geq \rho_{\max}(1/2-\delta)/((1/2+\delta)(1+\delta))$; that is, ρ satisfies at least a $\rho_{\max}(1-2\delta)/((1+2\delta)(1+\delta))$ fraction of clauses in F. For $\delta = \varepsilon/6$, we have

$$(1-2\delta)/((1+2\delta)(1+\delta)) \geq 1/(1+\varepsilon),$$

for all $\varepsilon \in (0, 3/7)$, and thus obtain a $(1+\varepsilon)$ approximation. (For larger ε, we can just set ε to be a smaller constant, which is absorbed in the big-O.) ∎

4 Las Vegas Algorithms

In this section, we present a Las Vegas algorithm for offline $(1+\varepsilon)$-approximate nearest neighbor search in Hamming space, running in time $\widetilde{O}\left(n^{2-\Omega(\varepsilon^{1/3}/\log(1/\varepsilon))}\right)$ for dimension $d \leq n^\delta$ for a sufficiently small $\delta > 0$. This matches the previous best running time for Monte Carlo algorithms from past work [5].

The Monte Carlo algorithm from prior work (see also Section 5 below) makes use of a probabilistic polynomial threshold representation of an approximate threshold predicate, which consists of two main steps: (1) a probabilistic polynomial for an exact threshold predicate on d inputs with error $1/s$ and degree $O(\sqrt{d \log s})$, and (2) combining it with a Chebyshev polynomial in order to decrease the degree to $O((1/\varepsilon)^{1/3} \log(ds))$ for computing an ε-approximate threshold predicate instead. In this section, we make one key modification to each step so that our resulting probabilistic polynomials never give the wrong answer: they either output the correct answer, or else a large value indicating that an error has occurred.

For the probabilistic polynomial for step (1), we modify the original probabilistic polynomial construction of [6]. The polynomial from the prior work makes use of random samples of entries from the input vector, and notes that the polynomial will output the correct answer as long as certain tail bounds on these random samples hold. In Lemma 4.1 we construct a second polynomial (eTH) for checking whether these tail bounds hold, so that we can tell when the polynomial may be making a mistake. Next, for step (2), we replace a similar random sample with a partitioning of the input first suggested by Indyk [27]. By recursively evaluating our polynomial on each partition, we are guaranteed that at least one part will give the correct answer, and so we can tell whether an error may have occurred based on whether all the recursive calls agreed.

We begin with part (1).

LEMMA 4.1. *Given d, s, t, there is a probabilistic pair of polynomials $(\mathrm{TH}^{(d,s)}_{\le t}, \mathrm{eTH}^{(d,s)}_{\le t})$, where $\mathrm{TH}^{(d,s)}_{\le t}, \mathrm{eTH}^{(d,s)}_{\le t} : \{0,1\}^d \to \mathbb{N}$ both have degree $O(\sqrt{t \log(s)})$, such that for every $x = (x_1, \ldots, x_d) \in \{0,1\}^d$, we have $\Pr[\mathrm{eTH}^{(d,s)}_{\le t}(x) = 0] \ge 1 - 1/s$ and $\mathrm{eTH}^{(d,s)}_{\le t}(x) > 0$ otherwise, and*

- *if $x_1 + \cdots + x_d > t$, then $\mathrm{TH}^{(d,s)}_{\le t}(x) = 0$ or $\mathrm{eTH}^{(d,s)}_{\le t}(x) \neq 0$, and*
- *if $x_1 + \cdots + x_d \le t$, then $\mathrm{TH}^{(d,s)}_{\le t}(x) = 1$ or $\mathrm{eTH}^{(d,s)}_{\le t}(x) \neq 0$.*

Proof. We proceed by strong induction on t. Let $c, k \ge 1$ be two constants to be set later. We may assume that $t \ge 9c^2 \log s$, for otherwise we can naively use a polynomial of degree t. Let $a = c\sqrt{t \log s}$ and define the polynomials:

- $C : \mathbb{Z} \to \mathbb{Z}$ defined as $C(z) = \prod_{r=-a}^{a}(z - r)^2$, so that $C(z) = 0$ when $|r| \le a$, and $C(z) \ge 1$ otherwise, and
- $A : \mathbb{Z}^d \to \mathbb{Z}$ a degree $O(a)$ polynomial such that for $x \in \{0,1\}^d$,
  - if $|x| \in (t, t + 2a]$ then $A(x) = 0$, and
  - if $|x| \in [t - 2a, t]$ then $A(x) = 1$, and
  - otherwise $A(x)$ may take any value.

  Such a polynomial exists by interpolation (see e.g. [6, Lemma 3.1]).
- Two recursively drawn probabilistic pairs of polynomials $(\mathrm{TH}^{(d/k,3s)}_{\le (t-a)/k}, \mathrm{eTH}^{(d/k,3s)}_{\le (t-a)/k})$ and $(\mathrm{TH}^{(d/k,3s)}_{\le (t+a)/k}, \mathrm{eTH}^{(d/k,3s)}_{\le (t+a)/k})$ for d/k-bit inputs.

On input $x \in \{0,1\}^d$, let $\tilde{x} \in \{0,1\}^{d/k}$ be a sample of d/k independent uniformly random entries of x. Define our polynomials as:

$$\mathrm{TH}^{(d,s)}_{\le t}(x) := \mathrm{TH}^{(d/k,3s)}_{\le (t-a)/k}(\tilde{x}) + A(x) \cdot \left(1 - \mathrm{TH}^{(d/k,3s)}_{\le (t-a)/k}(\tilde{x})\right) \cdot \mathrm{TH}^{(d/k,3s)}_{\le (t+a)/k}(\tilde{x}),$$

$$\mathrm{eTH}^{(d,s)}_{\le t}(x) := \mathrm{eTH}^{(d/k,3s)}_{\le (t+a)/k}(\tilde{x}) + \mathrm{eTH}^{(d/k,3s)}_{\le (t-a)/k}(\tilde{x}) + C(|x| - k \cdot |\tilde{x}|).$$

Correctness. A Chernoff bound shows that if c is big enough relative to k, then $\Pr[|x| - k \cdot |\tilde{x}| \notin [-a, a]] \le 1/(3s)$. Hence, by a union bound over the three terms defining $\mathrm{eTH}^{(d,s)}_{\le t}$, we have $\Pr[\mathrm{eTH}^{(d,s)}_{\le t}(x) \neq 0] \le 1/s$. Assuming $\mathrm{eTH}^{(d,s)}_{\le t}(x) = 0$, meaning $\mathrm{TH}^{(d/k,3s)}_{\le (t-a)/k}(\tilde{x})$ and $\mathrm{TH}^{(d/k,3s)}_{\le (t+a)/k}(\tilde{x})$ both give the correct answer, and $|x| - k \cdot |\tilde{x}| \in [-a, a]$, then:

- Case 1: $|x| > t + 2a$. Thus, $|\tilde{x}| > (t + a)/k$ so $\mathrm{TH}^{(d/k,3s)}_{\le (t+a)/k}(\tilde{x}) = \mathrm{TH}^{(d/k,3s)}_{\le (t-a)/k}(\tilde{x}) = 0$ and hence $\mathrm{TH}^{(d,s)}_{\le t}(x) = 0$.
- Case 2: $|x| \in (t, t + 2a]$. Thus, $A(x) = 0$, and $|\tilde{x}| > (t - a)/k$ so $\mathrm{TH}^{(d/k,3s)}_{\le (t-a)/k}(\tilde{x}) = 0$, and hence $\mathrm{TH}^{(d,s)}_{\le t}(x) = 0$.
- Case 3: $|x| \in [t - 2a, t]$. Thus, $A(x) = 1$, and $|\tilde{x}| < (t + a)/k$ so $\mathrm{TH}^{(d/k,3s)}_{\le (t+a)/k}(\tilde{x}) = 1$, and hence $\mathrm{TH}^{(d,s)}_{\le t}(x) = 1$.
- Case 4: $|x| < t - 2a$. Thus, $|\tilde{x}| < (t - a)/k$ so $\mathrm{TH}^{(d/k,3s)}_{\le (t-a)/k}(\tilde{x}) = 1$, and hence $\mathrm{TH}^{(d,s)}_{\le t}(x) = 1$.

Degree. The degree $D(t)$ of $\mathrm{TH}^{(d,s)}_{\le t}$ satisfies the recurrence $D(t) = 2D((t + c\sqrt{t \log s})/k) + O(\sqrt{t \log s})$, which solves to $D(t) = O(\sqrt{t \log s})$ when k is sufficiently large. The degree of $\mathrm{eTH}^{(d,s)}_{\le t}$ is similarly $O(\sqrt{t \log s})$. ∎

LEMMA 4.2. *Given d, s, t, there is a nonnegative probabilistic polynomial $\widehat{\mathrm{TH}}^{(d,s)}_{\le t} : \{0,1\}^d \to \mathbb{N}$ with degree $O(\sqrt{t \log(s)})$, such that for every $x = (x_1, \ldots, x_d) \in \{0,1\}^d$,*

- *if $x_1 + \cdots + x_d > t$, then $\widehat{\mathrm{TH}}^{(d,s)}_{\le t}(x) = 0$ with probability at least $1 - 1/s$;*
- *if $x_1 + \cdots + x_d \le t$, then $\widehat{\mathrm{TH}}^{(d,s)}_{\le t}(x) \ge 1$ with probability 1, and $\widehat{\mathrm{TH}}^{(d,s)}_{\le t}(x) = 1$ with probability at least $1 - 1/s$.*

Proof. Draw $(\mathrm{TH}^{(d,s)}_{\le t}, \mathrm{eTH}^{(d,s)}_{\le t})$ from Lemma 4.1, then pick

$$\widehat{\mathrm{TH}}^{(d,s)}_{\le t}(x) := \left(\mathrm{TH}^{(d,s)}_{\le t}(x)\right)^2 + 2 \cdot \mathrm{eTH}^{(d,s)}_{\le t}(x).$$

We now move on to part (2) mentioned at the beginning of the section.

LEMMA 4.3. *Given d, s, t and $\varepsilon \in (0,1)$, there is a nonnegative probabilistic polynomial $\widehat{\mathrm{TH}}^{(d,s,\varepsilon)}_{\le t} : \{0,1\}^d \to \mathbb{R}_{\ge 0}$ with degree $O((1/\varepsilon)^{1/3} \log(ds))$, such that for every $x = (x_1, \ldots, x_d) \in \{0,1\}^d$,*

- *if* $x_1 + \cdots + x_d > (1+\varepsilon)t$, *then* $\widetilde{\mathrm{TH}}_{\le t}^{(d,s,\varepsilon)}(x) \le 1$ *with probability at least* $1 - 1/s$;
- *if* $x_1 + \cdots + x_d \le t$, *then* $\widetilde{\mathrm{TH}}_{\le t}^{(d,s,\varepsilon)}(x) > s$ *with probability* 1.

Proof. We may assume that $t \ge \log(ds)$, for otherwise we can naively use a polynomial of degree t. Let k be a parameter to be set later. Let $s' = 2ks$.

Take a random partition of $[d]$ into k subsets $R_1, \ldots, R_k$ of size d/k. Let $\Delta = c\sqrt{kt \log s'}$ for a sufficiently large constant c.

Let $Q : \{0,1\}^d \to \mathbb{R}_{\ge 0}$ be a (deterministic) nonnegative polynomial such that for every $x = (x_1, \ldots, x_d) \in \{0,1\}^d$, (i) $Q(x) > s'$ if $x_1 + \cdots + x_d \le t$, and (ii) $Q(x) \le 1$ if $x_1 + \cdots + x_d \in [(1+\varepsilon)t, t+\Delta]$. As in [5], this can be achieved by a shifted, rescaled Chebyshev polynomial $Q(x) = \frac{1}{2}(T_q(\frac{(t+\Delta)-(x_1+\cdots+x_d)}{(t+\Delta)-(1+\varepsilon)t}) + 1)$, with an even degree $q = \Theta(\sqrt{\Delta/(\varepsilon t)} \log(s'))$.

For $x = (x_1, \ldots, x_d) \in \{0,1\}^d$, define

$$H(x) = \sum_{i=1}^{k} \widehat{\mathrm{TH}}_{\le t/k}^{(d/k,s')}(\{x_j : j \in R_i\})$$

$$\widetilde{\mathrm{TH}}_{\le t}^{(d,s,\varepsilon)}(x) = \tfrac{1}{k} Q(x) H(x).$$

Correctness.

- Case 1: $\sum_{j=1}^{d} x_j \le t$. Then $Q(x) \ge s'$. Also, $\sum_{j \in R_i} x_j \le t/k$ for some $i \in [k]$. So, $H(x) \ge 1$ and $\widetilde{\mathrm{TH}}_{\le t}^{(d,s,\varepsilon)}(x) \ge s'/k > s$ with probability 1.
- Case 2: $\sum_{j=1}^{d} x_j \in ((1+\varepsilon)t, t+\Delta]$. Then $Q(x) \le 1$, and $H(x) \le k$ with probability at least $1 - k/s' > 1 - 1/s$. Thus, $\widetilde{\mathrm{TH}}_{\le t}^{(d,s,\varepsilon)}(x) \le 1$ with probability at least $1 - 1/s$.
- Case 3: $\sum_{j=1}^{d} x_j > t + \Delta$. By the Chernoff bound, for each $i \in [k]$, we have $\sum_{j \in R_i} x_j > t/k$ with probability at least $1 - 1/s'$. Thus, $H(x) = 0$ and $\widetilde{\mathrm{TH}}_{\le t}^{(d,s,\varepsilon)}(x) = 0$ with probability at least $1 - 2k/s' = 1 - 1/s$.

The degree of $\widetilde{\mathrm{TH}}_{\le t}^{(d,s,\varepsilon)}$ is

$$O\left(\sqrt{(t/k)\log(s)} + \sqrt{\frac{\sqrt{kt\log(ds)}}{\varepsilon t}} \log(ds)\right).$$

Set $k = \varepsilon^{2/3} t / \log(ds)$. ∎

We next repeat a similar approach of partitioning the input as in Lemma 4.3, with the aim of decreasing the *number of monomials* in the resulting polynomial rather than the degree.

LEMMA 4.4. *Given* d, s, t *and* $\varepsilon \in (0,1)$, *for* $t = \alpha_0 d$, *there is a probabilistic polynomial* $P_{\le t}^{(d,s,\varepsilon)}$ *with degree* $O((1/\varepsilon)^{1/3} \log(ds))$ *and* $(ds)^{O((1/\varepsilon)^{1/3} \log(1/\varepsilon\alpha_0))}$ *monomials, satisfying the same properties as in the previous lemma.*

Proof. Let $k = \lfloor t/((2c/\varepsilon)^2 \log(2ds)) \rfloor$ for a sufficiently large constant c. Let $\varepsilon' = \varepsilon/2$ and $s' = 2ks$.

Take a random partition of $[d]$ into k subsets $R_1, \ldots, R_k$ of size d/k.

For $x = (x_1, \ldots, x_d) \in \{0,1\}^d$, define

$$P_{\le t}^{(d,s,\varepsilon)}(x) = \frac{1}{k} \sum_{i=1}^{k} \widetilde{\mathrm{TH}}_{\le t/k}^{(d/k,s',\varepsilon')}(\{x_j : j \in R_i\}).$$

Correctness.

- Case 1: $\sum_{j=1}^{d} x_j \le t$. Then $\sum_{j \in R_i} x_j \le t/k$ for some $i \in [k]$. Thus, $P_{\le t}^{(d,s,\varepsilon)}(x) \ge s'/k > s$ with probability 1.
- Case 2: $\sum_{j=1}^{d} x_j > (1+\varepsilon)t$. By the Chernoff bound, for each i, we have $\sum_{j \in R_i} x_j > (1+\varepsilon)t/k - c\sqrt{(t/k)\log s'} > (1+\varepsilon')t/k$ with probability at least $1 - 1/s'$. Thus, $P_{\le t}^{(d,s,\varepsilon)}(x) \le k/k = 1$ with probability at least $1 - k/s' - 1/s' > 1 - 1/s$.

The degree of $P_{\le t}^{(d,s,\varepsilon)}$ is $O((1/\varepsilon)^{1/3} \log(ds))$. The number of monomials is at most $k \cdot \binom{d/k}{O((1/\varepsilon)^{1/3}\log(ds))} \le k \cdot \binom{O((1/\alpha_0)(1/\varepsilon)^2 \log s)}{O((1/\varepsilon)^{1/3}\log(ds))} \le d(1/\varepsilon\alpha_0)^{O((1/\varepsilon)^{1/3}\log(ds))} \le (ds)^{O((1/\varepsilon)^{1/3}\log(1/\varepsilon\alpha_0))}$. ∎

We can now apply the polynomial method, as described in Section 2, to obtain a Las Vegas algorithm for the decision version of the offline approximate closest pair problem, using the polynomial in Lemma 4.4. We compute $P''(X,Y)$ for all pairs of groups (X,Y). For each (X,Y) with $P''(X,Y) \le s$, we know with probability 1 that (X,Y) has closest pair distance more than $t = \alpha_0 d$. For each (X,Y) with $P''(X,Y) > s$, we verify that there is a pair of distance at most $(1+\varepsilon)t$ by brute force in $O(\sqrt{s}^2) = O(s)$ time, and terminate the algorithm as soon as the first such pair is found. The algorithm clearly is always correct. For a pair (X,Y) with closest pair distance more than $(1+\varepsilon)t$, the probability that the brute force search is run is at most $O(1/s)$, so the expected cost of the brute force search is $O((1/s) \cdot s) = O(1)$. We set $s = n^{\Theta(\varepsilon^{1/3}/\log(\varepsilon\alpha_0))}$, so that $3^q s \le (n/\sqrt{s})^{0.172}$. Offline ANN can be solved similarly.

Finally, we apply the deterministic Lemma 3.3 with $k = 2$ to make $\alpha_0 = \Omega(\varepsilon)$. We then obtain our main theorem on Las Vegas algorithms.

THEOREM 4.1. *Given d and $\varepsilon \in (0,1)$, and given n red and n blue points in $\{0,1\}^d$, we can find a $(1+\varepsilon)$-approximate Hamming nearest blue point for every red point, in $\widetilde{O}(n^{2-\Omega(\varepsilon^{1/3}/\log(1/\varepsilon))})$ time by a Las Vegas randomized algorithm, if $d \le n^{0.05}$.*

Offline approximate farthest neighbor search is similar, although one has to directly modify most of the above lemmas, to reverse the direction of the inequalities. As in Section 3.3, the results extend to the ℓ_1 or ℓ_2 metric.

Our Las Vegas polynomial construction for threshold predicates have other applications, for example, to obtaining Las Vegas satisfiability algorithms for depth-2 threshold circuits; see [5].

5 Monte Carlo Algorithms

In this section, we give a slight improvement over Alman, Chan, and Williams' Monte Carlo algorithm [5] for offline approximate nearest neighbor search, from $n^{2-\Omega(\varepsilon^{1/3}/\log(1/\varepsilon))}$ running time to $n^{2-\Omega(\varepsilon^{1/3}/\log^{2/3}(1/\varepsilon))}$. The improvement is small, but the approach is interesting in that it simplifies Alman et al.'s polynomial construction, and also brings in some connection to locality-sensitive hashing. The approach is not useful for Las Vegas algorithms, however. Following the same philosophy as our deterministic polynomial construction, the improvement comes not from improving the degree but from reducing the number of monomials.

Alman, Chan, and Williams' polynomial construction for the unweighted threshold predicate is a combination of two parts: (i) a probabilistic polynomial obtained by random sampling (using Chernoff bounds in the analysis), polynomial interpolation, and recursion, and (ii) the Chebyshev polynomial. We replace the first part with the following lemma with a simple direct proof:

LEMMA 5.1. *Given d and $\beta_0, \varepsilon \in (0,1)$ with $\beta_0 = \Theta(1)$, there is a nonnegative probabilistic polynomial $R : \{0,1\}^d \to \mathbb{N}$ with degree $O((1/\delta)\log s)$ and $s^{O(1/\delta)}$ monomials, such that for every fixed $x = (x_1, \ldots, x_d) \in \{0,1\}^d$,*

- *if $x_1 + \cdots + x_d \le (1-\delta)\beta_0 d$, then $R(x) = 0$ with probability $1 - O(1/s)$;*
- *if $x_1 + \cdots + x_d > \beta_0 d$, then $R(x) \ge 1$ with probability $1 - O(1/s)$;*
- *if $x_1 + \cdots + x_d \in ((1-\delta)\beta_0 d, \beta_0 d]$, then $R(x) < s^2$ with probability $1 - O(1/s)$.*

Proof. Define

$$R(x) = \sum_{i=1}^{M} x_{r_{i1}} \cdots x_{r_{iq}},$$

where the r_{ij}'s are independently chosen random indices in $[d]$, and $q := \log_{1/(1-\delta)}(s^2)$, and $M := (1/\beta_0)^q \ln s \le s^{O(1/\delta)}$.

- If $x_1 + \cdots + x_d \le (1-\delta)\beta_0 d$, then $\Pr[R(x) > 0] \le M((1-\delta)\beta_0)^q \le (\ln s)/s^2$.
- If $x_1 + \cdots + x_d > \beta_0 d$, then $\Pr[R(x) = 0] < (1-\beta_0^q)^M < e^{-\beta_0^q M} \le 1/s$.
- If $x_1 + \cdots + x_d \in ((1-\delta)\beta_0 d, \beta_0 d]$, then $\mathbb{E}[R(x)] \le M\beta_0^q \le \ln s$, so by Markov's inequality, $\Pr[R(x) \ge s^2] \le (\ln s)/s^2$. (We could use Chernoff for a better bound, but that would not be necessary.)

This covers all the desired cases. ∎

The above idea is similar to the standard LSH method in Hamming space [23], which uses multiple hash functions each of which is a random projection. Each monomial corresponds essentially to a random projection, and the number of monomials corresponds to the number of hash functions used.

We can now obtain the following theorem by combining with Chebyshev polynomials in the same way as in Alman, Chan, and Williams [5].

THEOREM 5.1. *Given d, s, and $\beta_0, \varepsilon \in (0,1)$ with $\beta_0 = \Theta(1)$, there is a nonnegative probabilistic polynomial $P^{(d,s,\varepsilon)}_{\ge \beta_0 d} : \{0,1\}^d \to \mathbb{R}_{\ge 0}$ with degree $O((1/\varepsilon)^{1/3}\log s \log^{2/3} E)$ and $s^{O((1/\varepsilon)^{1/3}\log^{2/3} E)}$ monomials, where $E := d/\log s$, such that*

- *if $x_1 + \cdots + x_d \le \beta_0 d$, then $P^{(d,s,\varepsilon)}_{\ge \beta_0 d}(x) \le 1$ with probability $1 - O(1/s)$.*
- *if $x_1 + \cdots + x_d > (1+\varepsilon)\beta_0 d$, then $P^{(d,s,\varepsilon)}_{\ge \beta_0 d}(x) > s$ with probability $1 - O(1/s)$.*

Proof. Let δ be a parameter to be chosen later. Let $Q : \{0,1\}^d \to \mathbb{R}$ to be a (deterministic) nonnegative polynomial such that for every $x = (x_1, \ldots, x_d) \in \{0,1\}^d$, (i) $Q(x) > s'$ if $x_1 + \cdots + x_d > (1+\varepsilon)\beta_0 d$, and (ii) $Q(x) \le 1$ if $x_1 + \cdots + x_d \in (\beta_0 d - \Delta, \beta_0 d]$, where $\Delta = \delta d$ and $s' = s^3$. As in [5], this can be achieved by a shifted, rescaled Chebyshev polynomial $Q(x) = \frac{1}{2}(T_q(\frac{(x_1+\cdots+x_d)-(\beta_0 d-\Delta)}{\Delta}) + 1)$, with an even degree $q = \Theta(\sqrt{\Delta/(\varepsilon\beta_0 d)}\log s') = O(\sqrt{\delta/\varepsilon}\log s)$. Let R be the polynomial from the above lemma. Define

$$P^{(d,s,\varepsilon)}_{\ge \beta_0 d}(x) = \tfrac{1}{s^2} R(x) Q(x).$$

Correctness.

- If $x_1 + \cdots + x_d \le (1-\delta)\beta_0 d$, then $R(x) = 0$ and hence $P^{(d,s,\varepsilon)}_{\ge\beta_0 d}(x) = 0$ with probability $1 - O(1/s)$.
- If $x_1 + \cdots + x_d \in ((1-\delta)\beta_0 d, \beta_0 d]$, then $R(x) < s^2$ with probability $1 - O(1/s)$, and $Q(x) \le 1$. So, $P^{(d,s,\varepsilon)}_{\ge\beta_0 d}(x) \le 1$ with probability $1 - O(1/s)$.
- If $x_1 + \cdots + x_d > (1+\varepsilon)\beta_0 d$, then $R(x) \ge 1$ with probability $1 - O(1/s)$, and $Q(x) > s'$. So, $P^{(d,s,\varepsilon)}_{\ge\beta_0 d}(x) > s'/s^2 = s$ with probability $1 - O(1/s)$.

The degree of $P^{(d,s,\varepsilon)}_{\ge\beta_0 d}$ is $O(\sqrt{\delta/\varepsilon}\log s + (1/\delta)\log s)$. The number of monomials in $P^{(d,s,\varepsilon)}_{\ge\beta_0 d}$ is $\binom{d}{O(\sqrt{\delta/\varepsilon}\log s)} \cdot s^{O(1/\delta)} \le s^{O(\sqrt{\delta/\varepsilon}\log E + 1/\delta)}$. The result follows by choosing $\delta = \varepsilon^{1/3}/\log^{2/3} E$. ∎

Like before, we can set $P^{(d,s,\varepsilon)}_{\le\alpha_0 d}(x_1, \ldots, x_d) := P^{(d,s,\varepsilon\alpha_0)}_{\ge(1-(1+\varepsilon)\alpha_0)d}(1-x_1, \ldots, 1-x_d)$, and apply the above theorem with $\beta_0 := 1 - (1+\varepsilon)\alpha_0$ and ε changed to $\varepsilon\alpha_0$. Note that by padding with extra coordinates, we can ensure $\alpha_0 \le 1/2$ and thus $\beta_0 = \Omega(1)$. As in [5, proof of Theorem 1.5], with Monte Carlo randomization, we can apply Kushilevitz, Ostrovsky, and Rabani's dimensionality reduction technique [33], which makes $d = O((1/\varepsilon)^2 \log n)$, implying $E = \mathrm{poly}(1/\varepsilon)$; at the same time, the reduction makes $\alpha_0 = \Theta(1)$.

THEOREM 5.2. *Given d and $\varepsilon \in (0,1)$, and given n red and n blue points in $\{0,1\}^d$, we can find a $(1+\varepsilon)$-approximate Hamming nearest blue point for every red point, in $\widetilde{O}(dn + n^{2-\Omega(\varepsilon^{1/3}/\log^{2/3}(1/\varepsilon))})$ time by a Monte Carlo randomized algorithm.*

Offline approximate farthest neighbor search is similar, and the results extend to the ℓ_1 or ℓ_2 metric. We can obtain a Monte Carlo $(1+\varepsilon)$-approximation algorithm for MAX-SAT with $O^*((2 - \Omega(\varepsilon^{1/3}/\log^{2/3}(1/\varepsilon)))^n)$ running time, in the same manner as in the proof of Theorem 3.6.

A similar approach works for exact offline nearest neighbor search in Hamming space in dimension $d = c\log n$: Alman, Chan, and Williams' time bound of $n^{2-1/O(\sqrt{c}\log^{3/2} c)}$ time to $n^{2-1/O(\sqrt{c}\log c)}$ with Monte Carlo randomization.

References

[1] A. Abboud, R. Williams, and H. Yu, *More applications of the polynomial method to algorithm design*, in Proc. 26th Annual ACM-SIAM Symposium on Discrete Algorithms (SODA), 2015, pp. 218–230.

[2] T. D. Ahle, *Optimal Las Vegas locality sensitive data structures*, in Proc. 58th Annual IEEE Symposium on Foundations of Computer Science (FOCS), 2017, pp. 938–949.

[3] M. Ajtai, J. Komlós, and E. Szemerédi, *Deterministic simulation in LOGSPACE*, in Proc. 19th Annual ACM Symposium on Theory of Computing (STOC), 1987, pp. 132–140.

[4] J. Alman, *An illuminating algorithm for the light bulb problem*, in Proc. 2nd Symposium on Simplicity in Algorithms (SOSA), 2019, pp. 2:1–2:11.

[5] J. Alman, T. M. Chan, and R. Williams, *Polynomial representations of threshold functions and algorithmic applications*, in Proc. 57th Annual IEEE Symposium on Foundations of Computer Science (FOCS), 2016, pp. 467–476.

[6] J. Alman and R. Williams, *Probabilistic polynomials and hamming nearest neighbors*, in Proc. 56th Annual IEEE Symposium on Foundations of Computer Science (FOCS), 2015, pp. 136–150.

[7] N. Alon, U. Feige, A. Wigderson, and D. Zuckerman, *Derandomized graph products*, Computational Complexity, 5 (1995), pp. 60–75.

[8] A. Andoni and P. Indyk, *Near-optimal hashing algorithms for approximate nearest neighbor in high dimensions*, in Proc. 47th Annual IEEE Symposium on Foundations of Computer Science (FOCS), 2006, pp. 459–468.

[9] A. Andoni, P. Indyk, H. L. Nguyen, and I. Razenshteyn, *Beyond locality-sensitive hashing*, in Proc. 25th Annual ACM-SIAM Symposium on Discrete Algorithms (SODA), 2014, pp. 1018–1028.

[10] A. Andoni, P. Indyk, and I. P. Razenshteyn, *Approximate nearest neighbor search in high dimensions*, CoRR, abs/1806.09823 (2018).

[11] A. Andoni, T. Laarhoven, I. P. Razenshteyn, and E. Waingarten, *Optimal hashing-based time-space trade-offs for approximate near neighbors*, in Proc. 28th Annual ACM-SIAM Symposium on Discrete Algorithms (SODA), 2017, pp. 47–66.

[12] A. Andoni and I. Razenshteyn, *Optimal data-dependent hashing for approximate near neighbors*, in Proc. 47th Annual ACM Symposium on Theory of Computing (STOC), 2015, pp. 793–801.

[13] R. Beigel, N. Reingold, and D. A. Spielman, *The perceptron strikes back*, in Proc. 6th IEEE Structure in Complexity Theory Conference, 1991, pp. 286–291.

[14] T. M. Chan, *Approximate nearest neighbor queries revisited*, Discrete & Computational Geometry, 20 (1998), pp. 359–373.

[15] ———, *Applications of Chebyshev polynomials to low-dimensional computational geometry*, Journal of Computational Geometry, 9 (2018), pp. 3–20. Preliminary version in SoCG'17.

[16] T. M. Chan and R. Williams, *Deterministic APSP, orthogonal vectors, and more: Quickly derandomizing Razborov–Smolensky*, in Proc. 27th Annual ACM-SIAM Symposium on Discrete Algorithms (SODA),

2016, pp. 1246–1255.

[17] L. Chen, *On the hardness of approximate and exact (bichromatic) maximum inner product*, CoRR, arXiv:1802.02325 (2018).

[18] F. R. K. Chung and R. L. Graham, *Quasi-random hypergraphs*, Random Struct. Algorithms, 1 (1990), pp. 105–124.

[19] M. B. Cohen, *Ramanujan graphs in polynomial time*, in Proc. 57th Annual IEEE Symposium on Foundations of Computer Science (FOCS), 2016, pp. 276–281.

[20] D. Coppersmith, *Rapid multiplication of rectangular matrices*, SIAM J. Comput., 11 (1982), pp. 467–471.

[21] M. Datar, N. Immorlica, P. Indyk, and V. S. Mirrokni, *Locality-sensitive hashing scheme based on p-stable distributions*, in Proc. 20th Annual ACM Symposium on Computational Geometry (SoCG), 2004, pp. 253–262.

[22] B. Escoffier, V. T. Paschos, and E. Tourniaire, *Approximating MAX SAT by moderately exponential and parameterized algorithms*, Theor. Comput. Sci., 560 (2014), pp. 147–157.

[23] S. Har-Peled, P. Indyk, and R. Motwani, *Approximate nearest neighbor: Towards removing the curse of dimensionality*, Theory of Computing, 8 (2012), pp. 321–350.

[24] J. Haviland and A. Thomason, *Pseudo-random hypergraphs*, Discrete Mathematics, 75 (1989), pp. 255–278.

[25] E. A. Hirsch, *Worst-case study of local search for MAX-k-SAT*, Discrete Applied Mathematics, 130 (2003), pp. 173–184.

[26] S. Hoory, N. Linial, and A. Wigderson, *Expander graphs and their applications*, Bull. Amer. Math. Soc., 43 (2006), pp. 439–561.

[27] P. Indyk, *Dimensionality reduction techniques for proximity problems*, in Proc. 11th Annual ACM-SIAM Symposium on Discrete Algorithms (SODA), 2000, pp. 371–378.

[28] ———, *Uncertainty principles, extractors, and explicit embeddings of L2 into L1*, in Proc. 39th Annual ACM Symposium on Theory of Computing (STOC), 2007, pp. 615–620.

[29] A. Joffe, *On a set of almost deterministic k-wise independent random variables*, Annals Prob., 2 (1974), pp. 161–162.

[30] W. B. Johnson and J. Lindenstrauss, *Extensions of Lipschitz mappings into a Hilbert space*, in Conference in Modern Analysis and Probability, vol. 26 of Contemp. Math., AMS, 1984.

[31] N. Kahale, *On the second eigenvalue and linear expansion of regular graphs*, in Proc. 33rd Annual IEEE Symposium on Foundations of Computer Science (FOCS), 1992, pp. 296–303.

[32] M. Karppa, P. Kaski, J. Kohonen, and P. Ó Catháin, *Explicit correlation amplifiers for finding outlier correlations in deterministic subquadratic time*, in Proc. 24th Annual European Symposium on Algorithms (ESA), 2016, pp. 52:1–52:17.

[33] E. Kushilevitz, R. Ostrovsky, and Y. Rabani, *Efficient search for approximate nearest neighbor in high dimensional spaces*, SIAM J. Computing, 30 (2000), pp. 457–474.

[34] R. Motwani, A. Naor, and R. Panigrahi, *Lower bounds on locality sensitive hashing*, in Proc. 22nd Annual ACM Symposium on Computational Geometry (SoCG), 2006, pp. 154–157.

[35] R. Motwani and P. Raghavan, *Randomized Algorithms*, Cambridge University Press, 1995.

[36] J. Naor and M. Naor, *Small-bias probability spaces: Efficient constructions and applications*, SIAM J. Comput., 22 (1993), pp. 838–856.

[37] R. ODonnell, Y. Wu, and Y. Zhou, *Optimal lower bounds for locality-sensitive hashing (except when q is tiny)*, ACM Transactions on Computation Theory, 6 (2014), p. 5.

[38] R. Pagh, *CoveringLSH: Locality-sensitive hashing without false negatives*, ACM Trans. Algorithms, 14 (2018), pp. 29:1–29:17. Preliminary version in SODA'16.

[39] O. Reingold, S. P. Vadhan, and A. Wigderson, *Entropy waves, the zig-zag graph product, and new constant-degree expanders and extractors*, Ann. of Math., 155(1) (2002).

[40] J. Tarui, *Probabilistic polynomials, AC^0 functions and the polynomial-time hierarchy*, Theor. Comput. Sci., 113 (1993), pp. 167–183.

[41] S. P. Vadhan, *Pseudorandomness*, Now Foundations and Trends, 2011.

[42] G. Valiant, *Finding correlations in subquadratic time, with applications to learning parities and the closest pair problem*, J. ACM, 62 (2015. Preliminary version in FOCS'12), p. 13.

[43] A. Wei, *Optimal Las Vegas approximate near neighbors in ℓ_p*, in Proc. 30th Annual ACM-SIAM Symposium on Discrete Algorithms (SODA), 2019, pp. 1794–1813.

[44] R. Williams, *A new algorithm for optimal 2-constraint satisfaction and its implications*, Theor. Comput. Sci., 348 (2005. See also ICALP'04), pp. 357–365.

[45] ———, *New algorithms and lower bounds for circuits with linear threshold gates*, in Proc. 46th Annual ACM Symposium on Theory of Computing (STOC), 2014, pp. 194–202.

[46] ———, *Faster all-pairs shortest paths via circuit complexity*, SIAM J. Comput., 47 (2018), pp. 1965–1985. Preliminary version in STOC'14.