Chapter 2

Nearest-Neighbor Search without preprocessing

In this chapter we present two methods for solving the L_∞-nearest-neighbor problem. They need no preprocessing and require storage only for the point set P. Their average running time assuming that the set P is drawn randomly from the unit cube $[0,1]^d$ under uniform distribution is $\Theta\left(\frac{nd}{\ln n} + n\right)$, thereby improving the brute-force method by a factor of $\Theta\left(\frac{1}{\ln n}\right)$ for high dimensions $d \geq \ln n$. The methods have the advantage that they solve the problem exactly, and, furthermore, are very simple and easy to implement.

At first glance, the improving factor of $\Theta(1/\ln n)$ with respect to the brute-force method may not look like a significant improvement. However, the constant in the Θ-term is close to one, and for many applications n is in the range of tens or hundreds of thousands, so the speed-up factor comes close to 10. This consideration is confirmed by the experimental comparison with the brute-force method in Section 4.5.

2.1 The CUBE METHOD

The CUBE METHOD is a query algorithm for L_∞-nearest-neighbor search that needs no preprocessing. It has been introduced and analyzed by Alt and Hoffmann [5, 40].

In this section we present a slightly modified expected runtime analysis of the CUBE METHOD in order to make the extensions of this method, which we develop in this thesis, more easily understandable.

2.1.1 The idea and the core algorithm

The idea of the query algorithm is to consider for a query point $q = (q_1, \ldots, q_d) \in [0,1]^d$ the cube $C_{q,\alpha} = \left[q_1 - \frac{\alpha}{2}, q_1 + \frac{\alpha}{2}\right] \times \cdots \times \left[q_d - \frac{\alpha}{2}, q_d + \frac{\alpha}{2}\right]$ of side length α around q, which is expected to contain a low number $\varphi(n,d)$ of the points of P. Suitable values of α and φ will be specified later. An orthogonal range searching procedure determines the set P_α of data points contained in the cube $C_{q,\alpha}$. We present throughout this thesis four searching strategies to determine the set P_α. If $P_\alpha \neq \emptyset$ we determine the nearest neighbor of q by the brute-force method for the points of P_α. There is a nonzero probability that the cube $C_{q,\alpha}$ does not contain any points of P. In this case,

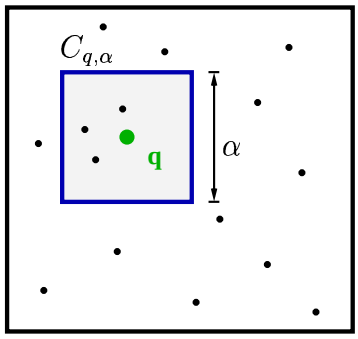
the brute-force method will be called for all points of P. The expected number φ should have the property that the probability of this event, $P_\alpha = \emptyset$, is so small that it does not effect the total asymptotic running time. This query algorithm is called the CUBE METHOD.

The input of the method is the query point q and the data structure which stores the points of P. The data structure depends on the considered orthogonal searching procedure. We symbolically denote this data structure by P in the following schematic description of the CUBE METHOD.

CUBE_METHOD(q, P)

α := SIDE_LENGTH(φ^*, q);
P_α := SEARCH_CUBE(α, q, P);
if ($P_\alpha \neq \emptyset$)
    $\hat{p}$:= BRUTE_FORCE(P_α);
else
    $\hat{p}$:= BRUTE_FORCE(P); (BRUTE)
return $\hat{p}$;

Procedure SEARCH_CUBE determines the set $P_\alpha = P \cap C_{q,\alpha}$. In the following we specify parameter φ^*. Because of uniform distribution, the side length α should be determined such that the volume of the box $C_{q,\alpha} \cap [0,1]^d$ equals $\frac{\varphi}{n}$. To compute α *exactly* would involve determining the root of a piecewise polynomial function of degree at most d, which is difficult and time consuming. Procedure SIDE_LENGTH, which we describe in Section 2.1.3, computes for a parameter $\varphi^* \in [1, n-1]$ the side length α of a cube $C_{q,\alpha}$ with center q such that the expected number of points in $P \cap C_{q,\alpha}$ is $\varphi \in [\varphi^*, \varphi^* + 1)$. The running time of SIDE_LENGTH is $O(d \log(d\varphi^*))$.



$$E\,[\,|C_{q,\alpha} \cap P|\,] \;=\; \varphi \in [\varphi^*, \varphi^* + 1)$$

The expected runtime E[T_{cube}] of the CUBE METHOD is given by:

$$\begin{aligned} \mathrm{E}[\,T_{cube}\,] \;=\; & \mathrm{E}[\,T_{side}\,] \;+\; \mathrm{E}[\,T_{search}\,] \;+\; \Pr[\text{ cube } C_{q,\alpha} \text{ empty }] \cdot T_{brute}(n) \\ & + \Pr[\text{ cube } C_{q,\alpha} \text{ not empty }] \cdot \mathrm{E}[\,T_{brute}(|P_\alpha|)\,] \end{aligned} \tag{2.1}$$

where T_{search} is the runtime of the SEARCH_CUBE procedure, T_{side} is the runtime of the SIDE_LENGTH procedure and $T_{brute}(m)$ is the runtime of brute-force for m points.

A suitable parameter φ^* is to be determined such that the expected runtime is minimized asymptotically.

Remark 2.1. *The* CUBE METHOD *can be improved as follows: if $C_{q,\alpha}$ is empty, the alternative to determining the nearest neighbor by brute-force is to consider a larger cube with center q and search it for points. The size of the current cube should be increased as long as it contains no points of P. Nevertheless, this variant called* GROWING-CUBE*, which we analyze in Section 4.1, has the same expected asymptotic complexity as the* CUBE METHOD*. The advantage of* GROWING-CUBE *is that it can be extended to compute the k nearest neighbors of the query point q from the point set P as we show in Section 4.2.*

The running time of the CUBE METHOD is essentially determined by the running time of the SEARCH_CUBE procedure. In the following section we present an orthogonal range searching algorithm which needs no preprocessing. It determines the points of P contained in the cube $C_{q,\alpha}$ with an expected runtime of $O(\frac{nd}{\ln(n/\varphi)} + n)$, where φ is the expected number of points in $C_{q,\alpha}$.

2.1.2 Searching the cube by scanning

The searching algorithm presented in this section works as follows. For each point p^i its coordinates p^i_j are tested whether they are contained in the appropriate interval $[q_j - \frac{\alpha}{2}, q_j + \frac{\alpha}{2}]$. As soon as a point $p^i \in P$ turns out to be not contained in $C_{q,\alpha}$, the algorithm eliminates it from further consideration. If no coordinate of p^i did fail the test, that is $p^i_j \in [q_j - \frac{\alpha}{2}, q_j + \frac{\alpha}{2}],\ \forall j \in \{1, \ldots, d\}$, then point p^i is added to the actual list of points contained in $C_{q,\alpha}$. The following gives a schematic description of this procedure called SCAN.

SCAN$(\alpha, q, P, \mathcal{D})$

$P_\alpha := \emptyset$;
forall $p^i \in P$
  for $j = \mathcal{D}(1), \ldots, \mathcal{D}(d)$
    if $|p^i_j - q_j| > \frac{\alpha}{2}$ then break for-loop; (TESTS)
  if (no coordinate failed) then $P_\alpha := P_\alpha \cup \{p^i\}$;
return P_α ;

SCAN has as input the side length α, the query point q, the point set P and a list $\mathcal{D} = (\mathcal{D}(1), \ldots, \mathcal{D}(d))$, which is a permutation of the dimensions $\{1, 2, \ldots, d\}$. $\mathcal{D}$ gives the order in which the coordinates of the points p^i should be tested. This order can also be computed by procedure SIDE_LENGTH. The set P_α containing the points of cube $C_{q,\alpha}$ is the result of procedure SCAN.

The CUBE-METHOD with the SCAN searching procedure requires no preprocessing and $\Theta(nd)$ storage. It is the aim of this section to show that it has an expected asymptotic runtime of $O(\frac{nd}{\ln n} + n)$.

Expected runtime analysis of the SCAN procedure

We measure the running time T_{scan} of the SCAN procedure by the number of comparisons of coordinates performed in step (TESTS). Let S_i be the discrete random variable for the number of comparisons performed for the point p^i, $1 \le i \le n$. Obviously, $T_{scan} = \sum_{i=1}^{n} S_i$.

Let $C^j_{q,\alpha} = \mathcal{I}^j_1 \times \ldots \times \mathcal{I}^j_d$, $1 \le j \le d$, be defined as follows:

$$\mathcal{I}^j_l \;=\; \begin{cases} [q_l - \frac{\alpha}{2}, q_l + \frac{\alpha}{2}] & \text{if } l \in \{\mathcal{D}(1), \ldots, \mathcal{D}(j)\} \\ [0,1] & \text{otherwise} \end{cases}$$

The box $C^j_{q,\alpha}$ is obtained by relaxing the side lengths of the cube to unit intervals with respect to those dimensions that are different from the first j ones in the list $\mathcal{D}$. $C^d_{q,\alpha}$ is the cube $C_{q,\alpha}$.

The expected number $\mathrm{E}[S_i]$ of tests with respect to the point $p^i \in P$ is given by:

$$\mathrm{E}[S_i] \;=\; \sum_{j=1}^{d} j \cdot \Pr[S_i = j] \;=\; \sum_{j=1}^{d} \Pr[S_i \ge j] \;=\; 1 + \sum_{j=1}^{d-1} \Pr\left[\, p^i \in C^j_{q,\alpha} \,\right], \tag{2.2}$$

where $\Pr[S_i \ge j]$ is the probability that the while-loop in (TESTS) is carried out at least j times ($1 \le j \le d$). Clearly, $\Pr[S_i \ge 1] = 1$ and $\Pr[S_i \ge j] = \Pr[\, p^i \in C^{j-1}_{q,\alpha}\,]$ ($1 < j \le d$). Because of uniform distribution $\Pr[\, p^i \in C^j_{q,\alpha}\,]$ equals the volume $\boldsymbol{V}(\, C^j_{q,\alpha} \cap [0,1]^d\,)$ of the box $C^j_{q,\alpha} \cap [0,1]^d$.

The expected number φ of data points in the cube $C_{q,\alpha}$ is related to the volume of the box $C_{q,\alpha} \cap [0,1]^d$, which equals $\frac{\varphi}{n}$. We prove bounds on $\mathrm{E}[T_{scan}]$, which are functions on n, d and φ. This main result of this section is summarized in the following lemma.

Lemma 2.1. *There exists a suitable order $\mathcal{D}$ of the dimensions, such that* SCAN *computes $P \cap C_{q,\alpha}$ with an expected number of $O(\frac{nd}{\ln(n/\varphi)} + n)$ comparisons, if the points of P are drawn independently at random from $[0,1]^d$ under uniform distribution.*

Remark 2.2. *If $C_{q,\alpha} \subseteq [0,1]^d$ then the expected number φ equals $n \cdot \boldsymbol{V}(\, C_{q,\alpha}\,) = n\alpha^d$, thus $\alpha = \sqrt[d]{\frac{\varphi}{n}}$. We also have $Pr[\, p^i \in C^j_{q,\alpha}\,] = \boldsymbol{V}(\, C^j_{q,\alpha}\,) = \alpha^j$ and equation (2.2) implies:*

$$E[T_{scan}] \;=\; n(1 + \alpha + \alpha^2 + \ldots + \alpha^{d-1}) \;=\; \frac{n - \varphi}{1 - \sqrt[d]{\varphi/n}}\,.$$

We will prove the bound $\mathrm{E}[T_{scan}] \le \frac{n-\varphi}{1-\sqrt[d]{\varphi/n}}$ in the general case. What remains then is to estimate $\frac{1}{1-\sqrt[d]{\varphi/n}}$, which can be done by applying the following result:

Lemma 2.2. *Let $0 < \lambda < 1$, then the following inequalities hold:*

$$\frac{1}{2} + \frac{1}{\ln(\frac{1}{\lambda})} \le \frac{1}{1-\lambda} \le 1 + \frac{1}{\ln(\frac{1}{\lambda})}$$

Proof. a) The left inequality is equivalent to

$$(1-\lambda)\ln(\tfrac{1}{\lambda}) + 2(1-\lambda) \le 2\ln(\tfrac{1}{\lambda}) \quad\Longleftrightarrow\quad 2(1-\lambda) \le (1+\lambda)\ln(\tfrac{1}{\lambda}),$$

where $\lambda \in (0,1)$. Consider $h\colon (0,1) \to \mathbb{R}$ with $h(\lambda) = (1+\lambda)\ln(\frac{1}{\lambda}) - 2(1-\lambda)$. It is easy to see that $\lim_{\lambda\to 0} h(\lambda) = \infty$, $\lim_{\lambda\to 1} h(\lambda) = 0$ and

$$h'(\lambda) \;=\; \ln(\tfrac{1}{\lambda}) - (1+\lambda)\cdot\tfrac{1}{\lambda} + 2 \;=\; \ln(\tfrac{1}{\lambda}) - \tfrac{1}{\lambda} + 1 \;<\; 0$$

for $\lambda \in (0,1)$. Thus, $h(\lambda) \ge 0$ for $\lambda \in (0,1)$ which implies the inequality to be proven.

b) The right inequality is equivalent to

$$\ln(\tfrac{1}{\lambda}) \leq (1-\lambda)\ln(\tfrac{1}{\lambda}) + (1-\lambda) \quad \Longleftrightarrow \quad 0 \leq 1 - \lambda - \lambda\ln(\tfrac{1}{\lambda}),$$

where $\lambda \in (0,1)$. Consider $h\colon (0,1) \to \mathbb{R}$ with $h(\lambda) = 1 - \lambda - \lambda\ln(\frac{1}{\lambda})$. It is easy to see that $\lim_{\lambda\to 0} h(\lambda) = 1$, $\lim_{\lambda\to 1} h(\lambda) = 0$ and $h'(\lambda) = -\ln\left(\frac{1}{\lambda}\right) < 0$ for $\lambda \in (0,1)$. Thus, $h(\lambda) \geq 0$ for $\lambda \in (0,1)$ which implies the inequality to be proven.

□

Lemma 2.3. *The inequality* $1 - \lambda \leq \ln(\frac{1}{\lambda})$ *holds for* $\lambda \in (0,1)$.

Proof. By Lemma 2.2, we obtain $\frac{1}{\ln(\frac{1}{\lambda})} \leq \frac{1}{1-\lambda}$ for $\lambda \in (0,1)$. □

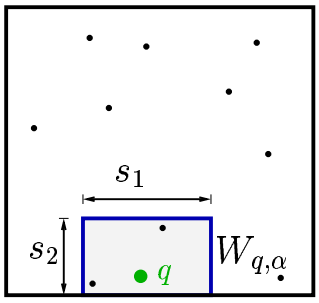
Proof of Lemma 2.1: The volume of the box $W_{q,\alpha} = C_{q,\alpha} \cap [0,1]^d$ is the product of its side lengths. Let s_j be the side length of $W_{q,\alpha}$ for dimension j, $1 \leq j \leq d$:

$$s_j(\alpha) = \min(q_j + \alpha/2, 1) - \max(q_j - \alpha/2, 0)$$

We have

$$s_j(\alpha) \;=\; \begin{cases} \alpha & \text{if } 0 \leq \alpha \leq 2m(q_j) \\ m(q_j) + \frac{\alpha}{2} & \text{if } 2m(q_j) < \alpha \leq 2 - 2m(q_j) \\ 1 & \text{otherwise} \end{cases} \tag{2.3}$$

where $m(q_j) = \min(q_j, 1-q_j)$. Because of the choice of α the volume $\boldsymbol{V}(W_{q,\alpha}) = \prod_{j=1}^{d} s_j(\alpha)$ equals $\frac{\varphi}{n}$.



$$s_2 < s_1 \;\Rightarrow\; \mathcal{D} = (2,1)$$

We obtain by (2.2) the expected number of comparisons for point p^i:

$$\mathrm{E}[S_i] = 1 + s_{\mathcal{D}(1)}(\alpha) + s_{\mathcal{D}(1)}(\alpha) \cdot s_{\mathcal{D}(1)}(\alpha) + \ldots + \prod_{l=1}^{d-1} s_{\mathcal{D}(l)}(\alpha)\,, \tag{2.4}$$

where $\mathcal{D} : \{1,\ldots,d\} \to \{1,\ldots,d\}$ specifies the order in which the coordinates will be checked in step (TESTS). Observe that some of the side lengths of $W_{q,\alpha}$ may be close to 1. To reduce the

expected number of tests it is better to check those coordinates first where the box $W_{q,\alpha}$ has a small side length. We define $\mathcal{D}$ such that it corresponds to the increasing order of the side lengths:

$$s_{\mathcal{D}(1)}(\alpha) \leq s_{\mathcal{D}(2)}(\alpha) \leq \cdots \leq s_{\mathcal{D}(d)}(\alpha) \ .$$

Note that in this case formula (2.4) is minimal over all permutations $\mathcal{D}$. By formula (2.3) the following holds for the side lengths $s_j = s_j(\alpha)$:

$$s_j(\alpha) \leq s_l(\alpha) \iff \min(q_j, 1-q_j) \leq \min(q_l, 1-q_l) \ .$$

Thus, the increasing order of the side lengths depends only on the query point q, and the corresponding order $\mathcal{D}$ can be computed in time $O(d \log d)$, once at the beginning in an additional, preparation step of procedure SIDE_LENGTH.

We bound the partial products $\prod_{l=1}^{j} s_{\mathcal{D}(l)}(\alpha)$ as follows. Let λ be the geometric mean of the side lengths $s_j(\alpha)$, $j = 1, \ldots, d$. We claim that

$$\prod_{l=1}^{j} s_{\mathcal{D}(l)}(\alpha) \leq \lambda^j. \tag{2.5}$$

Consider $\pi_k = \prod_{j=1}^{k} \beta_j$, where $\beta_j = \frac{s_{\mathcal{D}(l)}(\alpha)}{\lambda}$ for $j = 1, \ldots, d$. We get for some $l \leq d$ that

$$0 \leq \beta_1 \leq \ldots \beta_l \leq 1 \leq \beta_{l+1} \leq \cdots \leq \beta_d$$

consequently, $1 \geq \pi_1 \geq \cdots \geq \pi_l \leq \pi_{l+1} \leq \cdots \leq \pi_d = 1$ and therefore $\pi_k \leq 1$ for all $1 \leq k \leq d$ which is equivalent with inequality (2.5).

Since $\lambda = \sqrt[d]{\boldsymbol{V}(W_{q,\alpha})} < 1$, we obtain by (2.5) the following:

$$\mathrm{E}[T_{scan}] \leq n(1 + \lambda + \lambda^2 + \cdots + \lambda^{d-1}) = \frac{n-\varphi}{1-\lambda} \tag{2.6}$$

We estimate the value of $\frac{1}{1-\lambda}$ where $\lambda = \sqrt[d]{\boldsymbol{V}(W_{q,\alpha})} = \sqrt[d]{\frac{\varphi}{n}} \in (0,1)$ by Lemma 2.2. This together with inequality (2.6) provides an upper bound on the expected number of comparisons required by SCAN:

$$\mathrm{E}[T_{scan}] \ \leq \ (n-\varphi) + \frac{(n-\varphi)d}{\ln(n/\varphi)}. \tag{2.7}$$

Note that if special case $C_{q,\alpha} \subseteq [0,1]^d$ occurs, then $\lambda = \alpha$ and additionally, we obtain by Lemma 2.2 the lower bound

$$\mathrm{E}[T_{scan}] = \frac{n-\varphi}{1-\lambda} \geq \frac{n-\varphi}{2} + \frac{(n-\varphi)d}{\ln(n/\varphi)} \tag{2.8}$$

□

2.1.3 Computation of the side length α

For a query point q the volume of the box $W_{q,\alpha} = C_{q,\alpha} \cap [0,1]^d$ is given by $\boldsymbol{V}(W_{q,\alpha}) = \prod_{j=1}^{d} s_j(\alpha)$. Side lengths s_j are defined in (2.3).

We do not compute α *exactly* such that $\boldsymbol{V}(W_{q,\alpha}) = \varphi^*/n$, since that would involve determining the root of a piecewise polynomial function of degree at most d, which is difficult and time consuming. Instead, since the volume $\boldsymbol{V}(W_{q,\alpha})$ is a monotone increasing function in α, the procedure SIDE_LENGTH(φ^*, q) computes some value α such that

$$\frac{\varphi^*}{n} \leq \boldsymbol{V}(W_{q,\alpha}) = \prod_{j=1}^{d} s_j(\alpha) < \frac{\varphi^*+1}{n} \leq 1 \,, \tag{2.9}$$

where φ^* is a real parameter in $[1, n-1]$.

Hoffmann [40] shows that such a value α can be determined in time $O(d\log(nd))$, by searching the interval $[0,2]$. We slightly modify his approach and show that a value α that fulfills (2.9) can be determined by searching the interval $[\sqrt[d]{\varphi^*/n}, 2\sqrt[d]{\varphi^*/n}\,]$ in time $O(d\log(\varphi^* d))$.

Given query point q, we define the function $V_q : \mathbb{R}_+ \to [0,1]$ such that:

$$V_q(\alpha) = \prod_{l=1}^{d} s_j(\alpha) \,, \tag{2.10}$$

where the functions $s_j : \mathbb{R}_+ \to [0,1]$, $1 \leq j \leq d$, are defined in (2.3). Since functions s_j are continuous and monotone increasing, the volume function V_q is also continuous and monotone increasing with $V_q(0) = 0$ and $V_q(2) = 1$. Therefore, we can determine some value α with $\frac{\varphi^*}{n} \leq V_q(\alpha) < \frac{\varphi^*+1}{n}$ by binary search.

Let point $q_c = (\frac{1}{2}, \ldots, \frac{1}{2})$ be the center of the unit cube $[0,1]^d$. Then the volume function $V_{q^c} : \mathbb{R}_+ \to [0,1]$ is given by:

$$V_{q^c}(\alpha) \;=\; \begin{cases} \alpha^d & \text{if } 0 \leq \alpha \leq 1 \\ 1 & \text{otherwise.} \end{cases}$$

Obviously, $V_{q^c}\left(\sqrt[d]{\frac{\varphi^*}{n}}\right) = \frac{\varphi^*}{n}$.

Lemma 2.4. Given are $n, d \in \mathbb{N}$ with $n \geq 2$ and $d \geq 2$, and $\gamma \in \mathbb{R}$ with $1 \leq \gamma \leq n-1$. For all $\epsilon \geq 0$ fulfilling

$$\epsilon \leq \sqrt[d]{\gamma/n} \cdot \frac{1}{d(\gamma+1)}$$

the following holds:

$$V_{q^c}\left(\sqrt[d]{\gamma/n} + \epsilon\right) \;<\; \frac{\gamma+1}{n}$$

Proof. For $n \in \mathbb{N}$, $n \geq 2$ and a real $x \in [-1, \infty)$, $x \neq 0$ we have $(1+\frac{x}{n})^n < e^x$. This implies:

$$\left(1 - \frac{1}{\gamma+1}\right)^{\gamma+1} < e^{-1} \quad \Rightarrow \quad e < \left(\frac{\gamma+1}{\gamma}\right)^{\gamma+1} \quad \Rightarrow \quad e^{1/(\gamma+1)} < 1 + \frac{1}{\gamma} \,,$$

therefore,

$$\left(\frac{1/(\gamma+1)}{d}+1\right)^d \le e^{1/(\gamma+1)} < 1+\frac{1}{\gamma} \quad \Rightarrow \quad \frac{1}{d(\gamma+1)} < \left(1+\frac{1}{\gamma}\right)^{1/d} - 1\,,$$

and we obtain:

$$\begin{aligned}\frac{\gamma+1}{n} &= \left(\left(\frac{\gamma}{n}\right)^{1/d} + \left(\frac{\gamma+1}{n}\right)^{1/d} - \left(\frac{\gamma}{n}\right)^{1/d}\right)^d = \left[\left(\frac{\gamma}{n}\right)^{1/d} + \left(\frac{\gamma}{n}\right)^{1/d}\left(\left(1+\frac{1}{\gamma}\right)^{1/d} - 1\right)\right]^d \\ &> \left(\left(\frac{\gamma}{n}\right)^{1/d} + \left(\frac{\gamma}{n}\right)^{1/d}\frac{1}{d(\gamma+1)}\right)^d \ge \left(\left(\frac{\gamma}{n}\right)^{1/d} + \epsilon\right)^d = V_{q^c}\left(\sqrt[d]{\gamma/n} + \epsilon\right)\end{aligned}$$

□

Lemma 2.5. Let the point $q_c = (\frac{1}{2}, \ldots, \frac{1}{2})$ be the center of the unit cube $[0,1]^d$. Given a query point $q \in [0,1]^d$ we have:

$$V_q(\alpha) \le V_{q^c}(\alpha) \le V_q(2\alpha)\,, \quad \text{for all } \alpha \in [0,\infty)\,.$$

Proof. By the definition of V_q and V_{q^c} we have $V_q(\alpha) \le V_{q^c}(\alpha)$. We claim $V_{q^c}(\alpha) \le V_q(2\alpha)$. If $\alpha \le 1$ then the side lengths of the box $C_{q,2\alpha} \cap [0,1]^d$ fulfill $s_j(2\alpha) \ge \min\{\, 2\alpha, \alpha + \min(q_j, 1-q_j), 1\,\} \ge \alpha$. This implies $V_{q^c}(\alpha) = (\alpha)^d \le V_q(2\alpha)$. Since any cube of side length 2 around a query point $q \in [0,1]^d$ will contain the unit cube, we obtain $V_q(2\alpha) = 1 = V_{q^c}(\alpha)$ for $\alpha > 1$. □

Lemma 2.6. Given a query point $q \in [0,1]^d$, consider the function $V_q : \mathbb{R}_+ \to [0,1]$ as defined in (2.10). It is a piecewise polynomial function of degree at most d. For $\gamma \in [1, n-1]$, some value $\alpha \in [0,1]$ with

$$\gamma \le n \cdot V_q(\alpha) < \gamma + 1$$

can be computed in $O(d \log(\gamma d))$ time.

Proof. By Lemma 2.5 we obtain $V_q\left(\sqrt[d]{\frac{\gamma}{n}}\right) \le V_{q^c}\left(\sqrt[d]{\frac{\gamma}{n}}\right) \le V_q\left(2\sqrt[d]{\frac{\gamma}{n}}\right)$. Thus, α can be determined by binary search in the interval $\left[\, \sqrt[d]{\frac{\gamma}{n}}, 2\sqrt[d]{\frac{\gamma}{n}}\,\right]$ as shown in the following schematic description of procedure SIDE_LENGTH.

SIDE_LENGTH(γ, q)

left $:= \sqrt[d]{\frac{\gamma}{n}}$; right $:= 2\sqrt[d]{\frac{\gamma}{n}}$;
α := right;
vol $:= V_q(\alpha)$;
while (vol $< \frac{\gamma}{n}$ or vol $> \frac{\gamma+1}{n}$)
  if (vol $> \frac{\gamma}{n}$) then right $:= \alpha$;
  else left $:= \alpha$;
  $\alpha := \frac{\text{left}+\text{right}}{2}$; vol $:= V_q(\alpha)$;
return α;

After the k-th iteration the length of the interval [left, right] is $l_k = \sqrt[d]{\frac{\gamma}{n}} \cdot (\frac{1}{2})^{k-1}$.

We denote by $\hat{\alpha}$ the value that fulfills $V_q(\hat{\alpha}) = \frac{\gamma}{n}$. This value exists, since V_q is continuous and monotone increasing. Throughout the searching process $\hat{\alpha}$ lies in the interval $[\text{left}, \text{right}]$. For all $\epsilon \geq 0$ we have $V_q(\hat{\alpha} + \epsilon) \leq V_{q^c}(\sqrt[d]{\gamma/n} + \epsilon)$ [40].

By Lemma 2.4, for $\alpha \in [\hat{\alpha}, \hat{\alpha} + \epsilon]$, where $\epsilon = \sqrt[d]{\frac{\gamma}{n}} \cdot \frac{1}{d(\gamma+1)}$ the following holds:

$$\frac{\gamma}{n} = V_q(\hat{\alpha}) \leq V_q(\alpha) \leq V_q(\hat{\alpha} + \epsilon) \leq V_{q^c}\left(\sqrt[d]{\gamma/n} + \epsilon\right) < \frac{\gamma + 1}{n}.$$

This implies that if $l_k \leq 2\epsilon$ then either a value $\alpha \in [\hat{\alpha}, \hat{\alpha} + \epsilon]$ has been already found or it will be found in the next iteration. The minimal k with the property $l_k \leq 2\epsilon$ can be determined as follows:

$$\sqrt[d]{\gamma/n} \cdot \left(\frac{1}{2}\right)^{k-1} \leq \frac{2\sqrt[d]{\gamma/n}}{d(\gamma+1)} \quad \Leftrightarrow \quad k \log(1/2) \leq \log\left(\frac{1}{d(\gamma+1)}\right) \quad \Leftrightarrow \quad k \geq \log(d(\gamma+1))$$

Consequently, the computation of α takes at most $\lceil \log(d(\gamma+1)) \rceil + 1$ steps. Each step involves an evaluation of V_q which takes $O(d)$ time. So we need $O\left(d \log(\gamma d)\right)$ time to determine α.

□

Theorem 2.1. *Procedure* SIDE_LENGTH(φ^*, q) *determines in time* $O(d \log(d\varphi^*))$ *the side length* α *of the cube* $C_{q,\alpha}$ *such that the expected number* φ *of points in* $P \cap C_{q,\alpha}$ *lies in the interval* $[\varphi^*, \varphi^* + 1)$.

2.1.4 The total expected runtime

There is a nonzero probability that the cube $C_{q,\alpha}$ does not contain any points of P. In this case, the brute-force method is called having a runtime of $\Theta(nd)$. In the following we determine the parameter φ^* such that the probability of this event, $C_{q,\alpha} \cap P = \emptyset$, is so small that it does not effect the total asymptotic runtime.

The side length α of the cube $C_{q,\alpha}$ is computed such that the expected number of points in $P \cap C_{q,\alpha}$ is $\varphi \in [\varphi^*, \varphi^* + 1)$. The probability that $C_{q,\alpha}$ contains no points of P is $(1 - \frac{\varphi}{n})^n$. With probability $1 - (1 - \frac{\varphi}{n})^n$ there is at least a point in $C_{q,\alpha}$ and in this case brute-force is called with the set $P_\alpha = C_{q,\alpha} \cap P$, having the expected runtime $\Theta(\varphi d)$.

The expected runtime $\mathrm{E}[T_{cube}]$ of the CUBE METHOD is proportional to the number of performed comparisons and arithmetic operations. Thus, by (2.1), the expected runtime of the CUBE-METHOD is given by:

$$\begin{aligned} \mathrm{E}[T_{cube}] &= \Theta(\mathrm{E}[T_{scan}]) + \left(1 - \frac{\varphi}{n}\right)^n \Theta(nd) + \left(1 - \left(1 - \frac{\varphi}{n}\right)^n\right) \cdot \Theta(\varphi d) + \Theta(d \ln(d\varphi)) \\ &= O\left(\frac{nd}{\ln(n/\varphi)} + n + e^{-\varphi} nd + \varphi d + d \ln(d\varphi)\right) \end{aligned} \tag{2.11}$$

We used the fact that the side length α of the cube $C_{q,\alpha}$ and the increasing order of the side lengths of the box $C_{q,\alpha} \cap [0,1]^d$ are computed in $\Theta(d \ln(d\varphi))$ time.

To obtain a total asymptotic runtime of $O\left(\frac{nd}{\ln(n/\varphi)} + n + d \ln d\right)$, we guarantee $\varphi \leq \frac{n}{\ln n}$ and the following:

$$e^{-\varphi} \cdot nd \leq \frac{nd}{\ln(n/\varphi)} \tag{2.12}$$

(2.12) is equivalent to $\varphi \geq \ln(\ln(n/\varphi))$, which is satisfied for $\varphi \geq \ln\ln n$.

Thus, in order to satisfy (2.12) and in the same time to keep a small upper bound on $\mathrm{E}[T_{scan}]$, it is sufficient to choose parameter $\varphi^* = \ln\ln n$, since φ lies in the interval $[\varphi^*, \varphi^* + 1)$. For this setting of parameter φ^* we obtain $\mathrm{E}[T_{cube}] = O\left(\frac{nd}{\ln n} + n + d\ln d\right)$, with a small constant (close to 1) in the O-term.

Remark 2.3. The upper bound on $\mathrm{E}[T_{cube}]$ in (2.11) is $\Omega\left(\frac{nd}{\ln n} + n + d\ln d\right)$ for any value of φ in $[1, n)$. For the special case $C_{q,\alpha} \subseteq [0,1]^d$ we proved in (2.8) also a lower bound on $E[T_{scan}]$. Therefore, if $C_{q,\alpha} \subseteq [0,1]^d$ we obtain :

$$\mathrm{E}[T_{cube}] \;=\; \Theta\left(\frac{(n-\varphi)d}{\ln(n/\varphi)} + n + \varphi d + d\ln d\right) \;=\; \Omega\left(\frac{nd}{\ln n} + n + d\ln d\right) .$$

Thus, there is no value of φ^* that can improve asymptotically the total expected runtime, which we obtain for $\varphi^* = \ln\ln n$.

Summarizing, we obtain:

Theorem 2.2. *Let P be a set of n points. The* CUBE METHOD *with the* SCAN *searching procedure finds the nearest neighbor from P to some query point with an expected asymptotic runtime of $O\left(\frac{nd}{\ln n} + n + d\ln d\right)$, if the points of P are drawn independently at random under uniform distribution.*

2.2 The ADAPTIVE METHOD

The SCAN procedure can be improved as follows: keep decreasing α, whenever some point p^m lying closer to q is found. More precisely, if p^m turns out to be in the actual cube $C_{q,\alpha}$ then the actual nearest neighbor $\hat{p}$ is set to be p^m and the new side length α is set to be $2 \cdot \|p^m - q\|_\infty$. After the procedure has scanned all points, $\hat{p}$ is the nearest neighbor of q if the scanned cube is not empty. This procedure is called ADAPTIVE_SCAN.

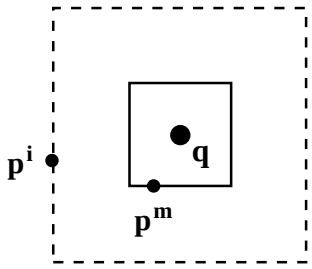


Figure 2.1: The ADAPTIVE SCAN

ADAPTIVE_SCAN$(\alpha, q, P, \mathcal{D})$

$P_\alpha := \emptyset$;
forall $p^i \in P$
  $dist := 0$;
  for $j \in \mathcal{D}$
    if $|p^i_j - q_j| > \frac{\alpha}{2}$ then break for-loop; (TESTS)
    else $dist := \max\{dist, |p^i_j - q_j|\}$;
  if (no coordinate failed)
    then $\alpha := 2 \cdot dist$; $nn := p^i$;
return nn ;

The order $\mathcal{D}$ of the dimensions $\{1, \ldots, d\}$ corresponds to the increasing order of the side lengths of the box $C_{q,\alpha} \cap P$.

Procedure ADAPTIVE_SCAN can be called with $\alpha = 2$, which means that we start with the whole unit cube and proceed the scan in the presented, adaptive way. This method is called the ADAPTIVE METHOD.

This adaptive approach was proposed by Hoffmann [40]. Alt [4] shows that the expected runtime of the ADAPTIVE METHOD is $O(\frac{nd}{\ln n} + n)$ if the query point is the center of the unit cube.

We present here an alternative derivation of the expected runtime analysis. It is based on results that we independently obtained for the analysis of the data structure that we present in Section 5.

Let A_i be the discrete random variable for the number of comparisons performed in step (TESTS) of ADAPTIVE_CUBE $(2, q, P, \mathcal{D})$ for the $(i+1)$th point p^{i+1}, which is considered after i points have been processed. We denote by α_i the random variable for the actual side length of the cube after i points have been processed. The tests for the point p^{i+1} are done with respect to the cube C_{q,α_i}.

Lemma 2.7. *The expected number $E[A_i]$ of comparisons performed for the $(i+1)$st tested point is bounded by:*

$$E[A_i] \leq \sum_{j=0}^{d-1} \int_0^1 y^{j/d} \cdot i \cdot (1-y)^{i-1}\, dy$$

Proof. The variable A_i depends on the side length variable α_i. The conditional expectation $\mathrm{E}[A_i \mid \alpha_i]$ of A_i given α_i is a random variable. By the theorem on conditional expectation (see [34]), we have:

$$\mathrm{E}[A_i] = \mathrm{E}[\ \mathrm{E}[A_i \mid \alpha_i]\ ] = \int_{-\infty}^{\infty} \mathrm{E}[A_i \mid \alpha_i = x\ ] \cdot f_{\alpha_i}(x)\, dx \tag{2.13}$$

where f_{α_i} is the density function of the side length α_i. The distribution function $F_{\alpha_i}(x)$ of α_i is given by:

$$F_{\alpha_i}(x) = \mathrm{Pr}[\ \alpha_i \leq x\ ] = 1 - \mathrm{Pr}[\ \alpha_i > x\ ] \tag{2.14}$$

A) Case $q = (\frac{1}{2}, \ldots, \frac{1}{2})$: The box $C_{q,\alpha_i} \cap [0,1]^d$ has equal side lengths. The event "$\alpha_i > x$" is equivalent to the event "$d(p^l, q) > x$ for all $l = 1, \ldots, i$", which means that none of the points $p^1, \ldots, p^i$ are contained in the cube $C_{q,x}$. Thus, by (2.14), we get

$$F_{\alpha_i}(x) = 1 - \mathrm{Pr}[\ \alpha_i > x\ ] = 1 - (1-x^d)^i\ .$$

Since $f_{\alpha_i}(x) = F'_{\alpha_i}(x) = i(1-x^d)^{i-1}(x^d)'$ and

$$\mathrm{E}[A_i \mid \alpha_i = x] = 1 + x + x^2 + \ldots + x^{d-1}$$

we obtain

$$\begin{aligned} \mathrm{E}[A_i] &= \int_0^1 (1 + x + x^2 + \ldots + x^{d-1}) \cdot i \cdot (1-x^d)^{i-1} \cdot (x^d)'\, dx \\ &= \sum_{j=0}^{d-1} \int_0^1 x^j \cdot i \cdot (1-x^d)^{i-1} \cdot (x^d)'\, dx\ . \end{aligned}$$

By substitution of variables $y = x^d$, we get

$$\mathrm{E}[A_i] = \sum_{i=0}^{d-1} \int_0^1 y^{j/d} \cdot i \cdot (1-y)^{i-1}\, dy\ . \tag{2.15}$$

B) General case: Let $C_{q,x}$ be the cube around q of side length x and let $s_j(x)$, $j = 1, \ldots, d$ be the side lengths of the box $C_{q,x} \cap [0,1]^d$ in increasing order. We denote by $\lambda(x)$ their geometric mean.

In analogy to arguments presented in case A), the distribution function $F_{\alpha_i}(x)$ of the random variable α_i is given by:

$$F_{\alpha_i}(x) = 1 - \mathrm{Pr}[\ \alpha_i > x\ ] = 1 - \left(\ 1 - \prod_{j=1}^{d} s_j(x)\ \right)^i = 1 - \left(\ 1 - \lambda^d(x)\ \right)^i$$

We have $f_{\alpha_i}(x) = F'_{\alpha_i}(x) = i \cdot (\, 1 - \lambda^d(x)\,)^{i-1} \cdot (\, \lambda^d(x)\,)'$. By (2.5), we have:

$$\begin{aligned} \mathrm{E}[A_i \mid \alpha_i = x] &= 1 + s_1(x) + s_1(x) \cdot s_2(x) + \ldots + \prod_{j=1}^{d-1} s_j(x) \\ &\leq 1 + \lambda(x) + \lambda^2(x) + \ldots + (\lambda(x))^{d-1} \ . \end{aligned}$$

Therefore,

$$\mathrm{E}[A_i] \leq \sum_{j=0}^{d-1} \int_0^1 \lambda^j(x) \cdot i \cdot \left(\, 1 - \lambda^d(x)\,\right)^{i-1} \cdot \left(\, \lambda^d(x)\,\right)' \, dx$$

and by substitution of variables $y = \lambda^d(x)$ we obtain:

$$\mathrm{E}[A_i] \leq \sum_{j=0}^{d-1} \int_0^1 y^{j/d} \cdot i \cdot (1-y)^{i-1} \, dy \ .$$

□

Now we use a result that we independently obtained for the analysis of the data structure that we present in Section 5. By Lemma 5.3 we have:

$$\int_0^1 y^{j/d} \cdot i \cdot (1-y)^{i-1} \, dy = \binom{i + j/d}{i}^{-1}$$

which can be estimated as follows.

Lemma 2.8. *For $i \geq 1$ and $1 \leq j \leq d$ we have:*

$$\binom{i + j/d}{i}^{-1} \leq \frac{1}{(\sqrt[d]{i}\,)^j}$$

Proof. The stated inequality

$$\binom{i + j/d}{i}^{-1} = \prod_{r=1}^{i} \frac{r}{r + j/d} \ \leq \ \frac{1}{(\sqrt[d]{i}\,)^j} \tag{2.16}$$

is equivalent to

$$\sum_{r=1}^{i} \ln(r + j/d) \ - \ \sum_{r=1}^{i} \ln r \ \geq \ \frac{j}{d} \cdot \ln(i) \qquad \text{for all } j = 1, \ldots, d \ . \tag{2.17}$$

Let $h\colon [0,1] \to \mathbb{R}$ be defined by $h(x) = \sum_{r=1}^{i} \ln(r+x) \ - \ \sum_{r=1}^{i} \ln r \ - \ x \cdot \ln(i)$. We have $h'(x) = \sum_{r=1}^{i} \frac{1}{r+x} - \ln(i)$ and $h'(0) = \sum_{r=1}^{i} \frac{1}{r} - \ln(i) > 0$ (see [33], page 263). Since $h''(x) = \sum_{r=1}^{i} -\frac{1}{(r+x)^2} \ < \ 0$, the function $h(x)$ is either increasing on $[0,1]$ or increasing on $[0,t]$ and decreasing on $[t,1]$ for some $t \in [0,1]$. This implies together with $h(0) = 0$ and $h(1) = \ln(i+1) - \ln i > 0$ the fact $h(x) \geq 0$ for $x \in [0,1]$, which proves (2.17).

□

Corollary 2.1. *For $i \geq 2$ we have*

$$E[A_i] \leq \sum_{j=0}^{d-1} \frac{1}{(\sqrt[d]{i}\,)^j} \leq \frac{1-1/i}{1-\sqrt[d]{1/i}}$$

Theorem 2.3. *The* ADAPTIVE METHOD *finds the nearest neighbor from P to some query point $q \in [0,1]^d$ with an expected asymptotic runtime of $O\left(\frac{nd}{\ln n} + n + d\ln d\right)$ if the points of P are drawn independently at random from $[0,1]^d$ under uniform distribution.*

Proof. As already mentioned the increasing order of the side lengths of some box $C_{q,x} \cap [0,1]^d$ depends only on the query point q and does not depend on the side length x. It can be computed once at the beginning in $O(d \ln d)$ time.

The running time of the ADAPTIVE METHOD is proportional to the number of performed comparisons of coordinates. The expected number of coordinate comparisons of the procedure ADAPTIVE_CUBE $(2, q, P, \mathcal{D})$ is given by $\mathrm{E}[A] = \sum_{i=0}^{n-1} \mathrm{E}[A_i]$ where $A_i \leq d$ and $A = \sum_{i=0}^{n-1} A_i$. Let $\lambda = \sqrt[d]{1/i}$ in Lemma 2.2. We obtain:

$$\frac{1}{2} + \frac{d}{\ln i} \leq \frac{1}{1-\sqrt[d]{1/i}} \leq 1 + \frac{d}{\ln i} \qquad \text{for } i = 2, \ldots, n-1$$

which together with Corollary 2.1 provides the upper bound $\mathrm{E}[A_i] \leq 1 + \frac{d}{\ln i}$. Since $\frac{1}{\ln x}$ is a convex function for $x \geq 2$, we get for $i \geq 3$:

$$\mathrm{E}[A] \;\leq\; 3d + n + \sum_{i=3}^{n-1} \frac{d}{\ln i} \;\leq\; n + 3d + d \cdot \int_{e^2}^{n-1} \frac{1}{\ln x}\, dx + d \cdot \sum_{i=3}^{8} \frac{1}{\ln i}$$

To estimate $\int_{e^2}^{n-1} \frac{1}{\ln x}\, dx$ we use the following:

$$\left[\frac{x}{\ln x}\right]_{e^2}^{n-1} \;=\; \int_{e^2}^{n-1} \frac{1}{\ln x} dx \;-\; \int_{e^2}^{n-1} \frac{1}{(\ln x)^2} dx \;=\; \int_{e^2}^{n-1} \frac{1}{\ln x}\left(1 - \frac{1}{\ln x}\right) dx \qquad (2.18)$$

Since $\frac{1}{\ln x}\left(1 - \frac{1}{\ln x}\right) \geq \frac{1}{2} \cdot \frac{1}{\ln x} > 0$ for $x \geq e^2$ we get by (2.18):

$$\left[\frac{x}{\ln x}\right]_{e^2}^{n-1} \;\leq\; \int_{e^2}^{n-1} \frac{1}{\ln x} dx \;\leq\; 2 \cdot \left[\frac{x}{\ln x}\right]_{e^2}^{n-1} \;\leq\; 2\frac{n}{\ln n} - e^2$$

Thus:

$$\mathrm{E}[A] \;<\; n + \frac{2nd}{\ln n} + c \cdot d \qquad (2.19)$$

where $c = 3 - e^2 + \sum_{j=3}^{8} \frac{1}{\ln j} < 4$. □