ACADEMIC PRESS

Journal of Algorithms 44 (2002) 226–245

Journal of Algorithms

www.academicpress.com

On the average performance of orthogonal range search in multidimensional data structures ☆

Amalia Duch [a] and Conrado Martínez [b,*,1]

[a] *Laboratorio Nacional de Informática Avanzada (LANIA), Xalapa, Mexico*
[b] *Departament de Llenguatges i Sistemes Informàtics, Universitat Politècnica de Catalunya, E-08034 Barcelona, Spain*

Received 1 March 2002

Abstract

In this work we present the average-case analysis of orthogonal range search for several multidimensional data structures. We first consider random relaxed K-d trees as a prototypical example. Later we extend these results to many different multidimensional data structures. We show that the performance of range searches is related to the performance of a variant of partial matches using a mixture of geometric and combinatorial arguments. This reduction simplifies the analysis and allows us to give exact upper and lower bounds for the performance of range searches (Theorems 3 and 4) and a useful characterization of the cost of range search as a sum of the costs of partial match-like operations (Theorem 5). Using these results, we can get very precise asymptotic estimates for the expected cost of range searches (Theorem 6).

Keywords: Orthogonal range search; Multidimensional data structures; Multidimensional search; Partial match search; K-d trees; Quadtrees; Average-case analysis

☆ This research was supported by project DGES PB98-0926 (AEDRI) of the Spanish Ministery for Education and Science.

* Corresponding author.
E-mail addresses: amalia@lania.mx (A. Duch), conrado@lsi.upc.es (C. Martínez).

[1] The author was also supported by the Future and Emergent Technologies programme of the EU under contract IST-1999-14186 (ALCOM-FT).

0196-6774/02/$ – see front matter
PII: S0196-6774(02)00213-4

1. Introduction

Orthogonal range search appears frequently in applications of large databases, geographical information systems, multimedia databases and computer graphics, among others [16]. Given a collection of multidimensional data points and a query rectangle, the goal of an orthogonal range search (range search, for short) is to retrieve all the data points in the collection that fall inside the given rectangle. Apart from the applications of range search as such, it is implicitly involved in more complex region queries and other associative queries.

Many data structures have been proposed for the management of multidimensional data and specifically for range search (see for instance [3]). Among these, K-d trees [1], quadtrees [2], K-d tries [15] and multiple variants of these.

However, the mathematical analysis of the performance of range searches has proven a difficult task. The original analysis by Bentley et al. and most subsequent work (e.g., [4,17]) rely on the unrealistic assumption that the considered tree data structure is perfectly balanced, which often yields unduly optimistic results. Only recently, there has been remarkable progresses on this direction with two recent papers [5,7] that provide upper (Ω) and lower bounds (big-Oh) for the average performance of range search in standard K-d trees, squarish K-d trees (a variant of the former introduced in [7]) and other multidimensional data structures.

In this work we analyze the average cost of range queries using the same random model as in [5,7] and obtain sharper results. In particular, we get exact upper and lower bounds (Theorems 3 and 4) and a characterization of the cost of range search as the sum of the cost of partial match-like searches (Theorem 5). Using these results, we can obtain tight asymptotic estimates for the expected cost of range search (Theorem 6). Our proof techniques—a combination of geometric and combinatorial arguments—are rather different from those in [5,7], but they are also easily applicable to many multidimensional data structures. We analyze first the average cost of range search in randomized K-d trees [8] and later discuss how our results generalize to other multidimensional data structures. We begin reviewing random(ized) relaxed K-d trees and orthogonal range queries in Section 2, as well as the random models used in the sequel. In Section 3 we introduce *sliced partial matches* and relate the performance of range search with the performance of sliced partial matches; we use this relationship to provide a tight asymptotic analysis of the average cost of range search. In Section 4 we show that the results for randomized K-d trees can be easily extended to most tree-like multidimensional data structures, namely, standard K-d trees, squarish K-d trees, K-d-t trees, standard and relaxed K-d tries, quadtrees and quadtries.

In the last section, we report the results of a preliminary experimental study that we have conducted in order to validate the analytic results of previous sections. An extended abstract of this works appears in [9].

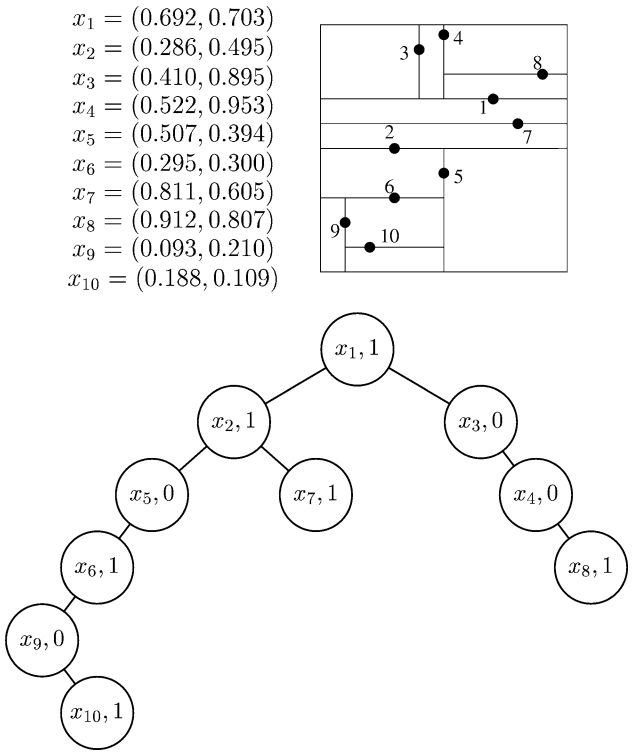

Fig. 1. A relaxed 2-d tree and the corresponding induced partition of $[0, 1]^2$.

2. Basic definitions

Let $F = \{x^{(1)}, x^{(2)}, \ldots, x^{(n)}\}$, $n \geqslant 0$, be the file of K-dimensional data points. We assume that each $x \in F$ is a K-tuple $x = (x_0, \ldots, x_{K-1})$ in $[0, 1]^K$.

A *relaxed K-d tree* [8] for a set F of K-dimensional data points is a binary tree in which:

(a) each node contains a K-dimensional data point and has an associated discriminant $j \in \{0, 1, \ldots, K-1\}$;
(b) for each node x with discriminant j, the following invariant is true: any data point y in the left subtree satisfies $y_j < x_j$ and any data point z in the right subtree satisfies $z_j \geqslant x_j$ (see Fig. 1).

Notice that the sequence of discriminants in a path from the root to any leaf is arbitrary. On the contrary, the definition of standard K-d trees [1] requires the sequence of discriminants along any path to be cyclic, starting with $j = 0$. Thus the root of the tree discriminates with respect to the first coordinate ($j = 0$), its

sons at level 1 discriminate w.r.t. the second coordinate ($j = 1$), and in general, all nodes at level m discriminate w.r.t. coordinate $j = m \bmod K$.

Our average-case analysis of range searches over relaxed K-d trees in Section 3 will assume that trees are *random*. We say that a relaxed K-d tree of size n is *random* if it is built by n insertions where the points are independently drawn from a continuous distribution in $[0,1]^K$ and the discriminants are uniformly and independently drawn from $\{0,\ldots,K-1\}$.

In random relaxed K-d trees, these assumptions about the distribution of the input imply that the $n!^K K^n$ possible configurations of input file and discriminant sequences are equiprobable [8]. For standard K-d trees, since there is a fixed rule for discriminants, the assumption of random insertions (producing random standard K-d trees) implies that all $n!^K$ input sequences are equiprobable [1,13]. In particular, in a random relaxed K-d tree each of the $n \cdot K$ possibilities of (key, discriminant) pairs are equally likely to appear in the root and once the root is fixed, the left and right subtrees are independent random relaxed K-d trees.

A *range query* is a K-dimensional hyperrectangle Q. We shall write $Q = [\ell_0, u_0] \times [\ell_1, u_1] \times \cdots \times [\ell_{K-1}, u_{K-1}]$, with $\ell_i \leqslant u_i$, for $0 \leqslant i < K$. We will use a small variation of the probabilistic model of random range queries introduced in [5,7]. In this model, the edges of a *random range query* have given lengths $\Delta_0, \Delta_1, \ldots, \Delta_{K-1}$, with $0 \leqslant \Delta_i \leqslant 1/2$, for $0 \leqslant i < K$ and the center of the query is an independently drawn point z in

$$\begin{aligned} Z_\Delta &= \prod_{0 \leqslant r < K} \left[-\frac{\Delta_r}{2}, 1 + \frac{\Delta_r}{2} \right] \\ &= \left[-\frac{\Delta_0}{2}, 1 + \frac{\Delta_0}{2} \right] \times \left[-\frac{\Delta_1}{2}, 1 + \frac{\Delta_1}{2} \right] \times \cdots \times \left[-\frac{\Delta_{K-1}}{2}, 1 + \frac{\Delta_{K-1}}{2} \right], \end{aligned}$$

sampled from some continuous distribution. Therefore, $\ell_i = z_i - \Delta_i/2$ and $u_i = z_i + \Delta_i/2$, for $0 \leqslant i < K$. Notice that in this model a range query Q may fall partially outside of $[0,1]^K$, so in general,

$$Q \subset C_\Delta = \prod_{0 \leqslant r < K} [-\Delta_r, 1 + \Delta_r].$$

Range searching in any variant of K-d trees is straightforward. When visiting a node x that discriminates w.r.t. the jth coordinate, we must compare x_j with the jth range $[\ell_j, u_j]$ of the query. If the query range is totally above (or below) that value, we must search only the right subtree (respectively, left) of that node. If, on the contrary, $\ell_j \leqslant x_j \leqslant u_j$ then both subtrees must be searched; additionally, we must check whether x falls or not inside the query hyperrectangle. This procedure continues recursively until empty subtrees are reached.

We will measure the cost of range queries by the number of nodes of the K-d tree visited during the search. If the number of points to be reported by the range search is P then the cost R_n of the range search will be of the form $\Theta(P + W_n)$, where W_n is the *overhead*.

3. Analysis of the cost of range searches

3.1. Bounding rectangles, slices, and sliced partial match

The *bounding rectangle* $B(x) = [l_0(x), u_0(x)] \times \cdots \times [l_{K-1}(x), u_{K-1}(x)]$ of a point $x = (x_0, \ldots, x_{K-1})$ in a K-d tree t is the region of $[0,1]^K$ corresponding to the leaf replaced by x when x was inserted into t. Formally, it is defined as follows:

(1) if x is the root of t then $B(x) = [0,1]^K$;
(2) if $y = (y_0, \ldots, y_{K-1})$ is the father of x in t and y discriminates w.r.t. the jth coordinate then:
   (a) if $x_j < y_j$ then $B(x) = [l_0(y), u_0(y)] \times \cdots \times [l_j(y), y_j] \times \cdots \times [l_{K-1}(y), u_{K-1}(y)]$, and
   (b) if $x_j \geqslant y_j$ then $B(x) = [l_0(y), u_0(y)] \times \cdots \times [y_j, u_j(y)] \times \cdots \times [l_{K-1}(y), u_{K-1}(y)]$.

Lemma 1. *A point x with bounding rectangle $B(x)$ is visited by a range search with query hyperrectangle Q if and only if $B(x)$ intersects Q.*

Proof. See [5,7]. □

In order to relate the performance of range searches with the performance of partial matches, we need to introduce several notions, beginning with that of *slice*. Given a bitstring $w = (w_0, \ldots, w_{K-1})$ of length K, the slice Q_w is the K-dimensional hyperrectangle defined by

$$Q_w = \prod_{0 \leqslant r < K} [\ell'_r, u'_r],$$

where $[\ell'_i, u'_i] = [\max\{0, \ell_i\}, \min\{u_i, 1\}]$ if $w_i = 0$ and $[\ell'_i, u'_i] = [0,1]$ if $w_i = 1$. Notice that $Q_{00\ldots0} = Q \cap [0,1]^K$ and $Q_{11\ldots1} = [0,1]^K$.

Another useful notion is that of *proper slice*. The proper slice $\widehat{Q}_w$ is the hyperregion defined by

$$\widehat{Q}_w = Q_w - \bigcup_{v<w} Q_v,$$

where $v < w$ if and only if $v_i < w_i$ for all $0 \leqslant i < K$. Thus a proper slice $\widehat{Q}_w$ is the region that results when all properly contained slices within Q_w are subtracted from it (see Fig. 2). Alternatively, $\widehat{Q}_w$ is the result of subtracting from Q_w those slices Q_v such that $v < w$ and v differs from w in just one bit. The only proper slice consisting of a simple connected region is $\widehat{Q}_{00\ldots0} = Q_{00\ldots0} = Q \cap [0,1]^K$; in general, $\widehat{Q}_w$ consists of $2^{\text{order}(w)}$ connected subregions, where order(w) is the number of 1's in the bitstring w.

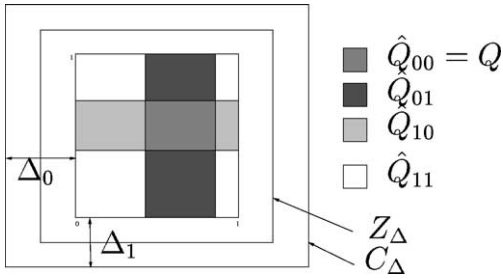

Fig. 2. Example of the proper slices induced by a query Q.

The most important concept in this subsection is that of *sliced partial match*. But let us briefly review first (standard) partial matches. In a partial match search, we are given a query $q = (q_0, q_1, \ldots, q_{K-1})$, with $q_i \in [-\Delta_i, 1 + \Delta_i] \cup \{*\}$ and the goal is to report those points in the file that match the query, that is, the points x such that $x_i = q_i$ if $q_i \neq *$, for all $0 \leqslant i < K$. For a query q, the bitstring $w = (w_0, w_1, \ldots, w_{K-1})$ such that $w_i = 1$ if $q_i \neq *$ and $w_i = 0$ otherwise, is called the *specification pattern* of the query. A query might then be thought as a pair consisting in a point $y \in C_\Delta$ and a bitstring w. Partial matches make sense if at least one coordinate of the query is specified and at least one coordinate is not. Their performance has been extensively studied in several multidimensional data structures (see, for instance, [7,11,12,14]).

Given a query hyperrectangle Q, a bitstring w and a point $y \in C_\Delta$, a sliced partial match acts as a standard partial match with query $q = (q_0, q_1, \ldots, q_{K-1})$ where $q_i = y_i$ if $w_i = 1$ and $q_i = *$ if $w_i = 0$ (hence the specification pattern of the partial match is w), but contrary to a standard partial match it only reports the visited points x in the data structure such that $x \in \widehat{Q}_w$.

To every (sliced) partial match with point y and specification pattern w we associate the hyperplane $H(y, w)$ defined by

$$H(y, w) = \{x \in C_\Delta \mid \forall i\colon\ w_i = 1 \Longrightarrow x_i = y_i\}.$$

Notice that the value of y_i in the definition of $H(y, w)$ is irrelevant if $w_i = 0$.

For instance, if $K = 2$ then $H(y, 00) = C_\Delta$, $H(y, 11) = \{y\}$, $H(y, 01)$ is a segment passing through y parallel to the horizontal axis, and $H(x, 10)$ is a segment passing through x parallel to the vertical axis.

Lemma 2. *A point x with bounding rectangle $B(x)$ is visited and reported by a partial match with query point y and specification pattern w, if and only if the bounding rectangle $B(x)$ intersects the hyperplane $H(y, w)$.*

A point x with bounding rectangle $B(x)$ is visited and reported by a sliced partial match with query hyperrectangle Q, specification pattern w, and query point y, if and only if $x \in \widehat{Q}_w$ and the bounding rectangle $B(x)$ intersects the hyperplane $H(y, w)$.

Proof. The proof is immediate from Lemma 1. Notice that a (sliced) partial match behaves as a range query in which the hyperrectangle query "degenerates" to the hyperplane $H(y, w)$ (when a coordinate is specified in the query the corresponding range $[\ell_i, u_i]$ has identical endpoints; when the coordinate is not specified we have a corresponding full range). In the case of sliced partial matches, only those points that also belong to $\widehat{Q}_w$ are reported. □

3.2. The combinatorial characterizations

In this section we state several relations between the cost $R(t)$ of an orthogonal range search in a K-d tree t and the performance $P_w(t, y)$ of a sliced partial match with specification pattern w and query point y in a K-d tree t. The implicit query hyperrectangle Q is the same for both the range search and the sliced partial match.

Theorem 3. *Given a query Q with corners $v_0, v_1, \ldots, v_{2^K-1}$ and a K-d tree t,*

$$R(t) \leqslant \sum_{0 \leqslant j < 2^K} \sum_{w \in (0+1)^K} P_w(t, v_j).$$

Proof. Consider a point x visited by a range search with query Q. Let w be the index of the proper slice that contains x, i.e., $x \in \widehat{Q}_w$. Recall that since x is visited by the range search we have $B(x) \cap Q \neq \emptyset$ (Lemma 1). Therefore, by Lemma 2, it suffices to show that if $B(x)$ intersects Q then there exists at least one corner v_j of Q such that the hyperplane $H(v_j, w)$ does intersect $B(x)$.

If $B(x)$ contains any of the corners of Q then the statement above is clearly true: since the hyperplane $H(v_j, w)$ contains v_j, it must intersect $B(x)$. If $B(x)$ does not contain any corner of Q, there are two possibilities to consider: either $B(x)$ is entirely within Q, or $B(x)$ intersects one or more faces of Q. If $B(x)$ is totally inside Q then $w = 0 \ldots 0$ and indeed $H(v_j, 0 \ldots 0) = C_\Delta$ intersects $B(x)$ for any corner v_j. On the other hand, if $B(x)$ intersects one or more faces of Q but does not contain a corner nor it is contained inside Q then $w \neq 11 \ldots 1$ and $\widehat{Q}_w$ must "contact" one of the intersected faces, in the sense that the face is a boundary of $\widehat{Q}_w$. Let f be such face. Now, the hyperplane $H(v_j, w)$ contains this face (and hence it intersects $B(x)$), provided that v_j is any corner of the face f (see Fig. 3 for a graphical illustration of this proof when $K = 2$). □

Theorem 4. *Given a query Q with center at z and a K-d tree t,*

$$R(t) \geqslant \sum_{w \in (0+1)^K} P_w(t, z).$$

Proof. The statement of the theorem is immediate, once we show that whenever a point x is reported by a sliced partial match with parameters w and z then it is

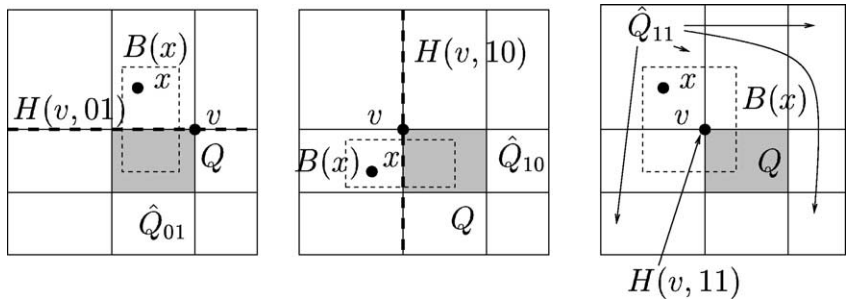

Fig. 3. Graphical illustration of the proof of Theorem 3.

also visited by the range search with query Q. Formally, we have to show that if $x \in \widehat{Q}_w$ and $H(z, w)$ intersects the bounding rectangle $B(x)$ of x then $B(x)$ also intersects Q. Indeed, if $w = 00\ldots0$ then the statement trivially holds since $x \in Q$. On the other hand, if $w \neq 00\ldots0$ then $H(z, w)$ and $\widehat{Q}_w$ are disjoint. Since $B(x)$ intersects $\widehat{Q}_w$ (x is part of both by hypothesis) the only way for $B(x)$ to intersect $H(z, w)$ is to intersect Q too (then, because of Lemma 1, x must be visited by the range search). □

Theorem 5. *Given a query Q with center at z divide C_Δ into 2^K quadrants $C_0, C_1, \ldots, C_{2^K-1}$, with z the contact point of the 2^K quadrants. Let $R^{(i)}(t)$ denote the number of points of the ith quadrant visited by a range search with query Q in the K-d tree t. Similarly, let $P_w^{(i)}(t, y)$ denote the number of points of the ith quadrant reported by sliced partial match with pattern w and point y in the K-d tree t. Let v_i be the unique corner of Q belonging to the ith quadrant. Then*

$$R^{(i)}(t) = \sum_{w \in (0+1)^K} P_w^{(i)}(t, v_i).$$

Proof. For a point x belonging to the ith quadrant and the proper slice $\widehat{Q}_w$, the intersection of $H(v_i, w)$ with $B(x)$ implies the intersection of Q with $B(x)$. On the other hand, if $B(x)$ intersects Q there is at least one corner v such that $H(v, w)$ intersects $B(x)$; it is not difficult to see that one of these corners must be v_i, the unique corner of Q in the ith quadrant (Fig. 3 may also help understanding the proof even though quadrants are not depicted there). □

3.3. The expected cost of range search in relaxed K-d trees

The theorems of Section 3.2 show that the analysis of range search reduces to the analysis of sliced partial matches.

Theorem 6. *Let $\mathbb{E}[R_n]$ be the expected cost of a range search with a random query in a random K-d tree of size n. Let $\mathbb{E}[P_{n,w}]$ be the expected cost of a sliced partial match with pattern w in a random K-d tree of size n, with respect to a uniformly and independently drawn random query point in $[0,1]^K$. Then*

$$\frac{1}{V(Z_\Delta)} \cdot \sum_{w\in(0+1)^K} \mathbb{E}[P_{n,w}] \leqslant \mathbb{E}[R_n] \leqslant V(Z_\Delta) \cdot \sum_{w\in(0+1)^K} \mathbb{E}[P_{n,w}],$$

where $V(Z_\Delta) = \prod_{0\leqslant r<K}(1+\Delta_r)$.

Proof. Clearly $R(t) = \sum_{0\leqslant i<2^K} R^{(i)}(t)$. Hence,

$$R(t) = \sum_{0\leqslant i<2^K} \sum_{w\in(0+1)^K} P_w^{(i)}(t, v_i). \tag{1}$$

Given a corner v of a query Q, if $v \in [0,1]^K$ let $v' = v$, otherwise let v' be the point in the boundary of $[0,1]^K$ closest to v (see Fig. 4). It is pretty clear that if Q falls partially off the $[0,1]^K$ boundary, the cost of the range search is the same as if we had a range query Q' where we had chopped the part of Q that falls outside $[0,1]^K$. And if we shift a query so that a corner v outside $[0,1]^K$ is aligned to v' then the corresponding range search will have a cost which is greater or equal to the cost of a range search where we do not perform such a shift. In other words, for any bitstring w, query Q, K-d tree t and quadrant i,

$$P_w(t, v_i) \leqslant P_w(t, v_i').$$

Hence,

$$R(t) \leqslant \sum_{0\leqslant i<2^K} \sum_{w\in(0+1)^K} P_w^{(i)}(t, v_i').$$

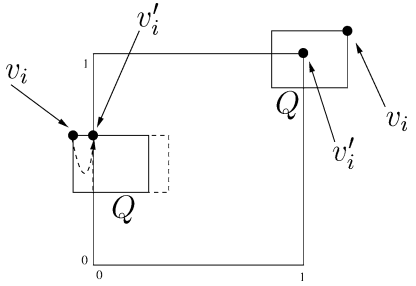

Fig. 4. Shifting queries falling partially off the boundaries.

The next step is to take expectations on both sides of the equation above and observe that, for uniformly distributed centers in Z_Δ, the probability that v_i falls outside $[0,1]^K$ is

$$\left(\prod_{0\leqslant r<K}(1+\Delta_r)\right)-1.$$

Then we have

$$\mathbb{E}[R_n]\leqslant \prod_{0\leqslant r<K}(1+\Delta_r)\cdot \sum_{0\leqslant i<2^K}\sum_{w\in(0+1)^K}\mathbb{E}\big[P_{n,w}^{(i)}\big],$$

where $\mathbb{E}[P_{n,w}^{(i)}]$ is the expected cost of a sliced partial match with respect to the ith quadrant and a random uniformly distributed query point in $[0,1]^K$. Now,

$$\mathbb{E}\big[P_{n,w}^{(i)}\big]=\mathrm{Vol}(C_i)\cdot\mathbb{E}[P_{n,w}],$$

where $\mathrm{Vol}(C_i)$ is the probability that, given a randomly drawn point z in Z_Δ, a random data point in $[0,1]^K$ falls in the ith quadrant defined by z. Since $\sum_{0\leqslant i<2^K}\mathrm{Vol}(C_i)=1$ the upper bound follows.

For the lower bound we use Theorem 4. Taking expectations in both sides of the inequality of Theorem 4 and conditioning on the event "the center of the query is inside $[0,1]^K$" the lower bound given in the statement is immediate, as the probability that the center of a random query falls inside $[0,1]^K$ is $1/\prod_{0\leqslant r<K}(1+\Delta_r)$. □

Although (1) gives a precise relationship between the cost of range search and the cost of sliced partial matches, we cannot use it to get results about the variance of R_n or its probability distribution since the costs of the sliced partial matches (the random variables $P_{n,w}^{(i)}$) are not independent.

Now we need to analyze the expected cost of sliced partial matches in random(ized) relaxed K-d trees. It easily follows from the analysis of the expected cost of standard partial matches in random relaxed K-d trees [14]. Our next theorem gives the expected cost of sliced partial matches in random relaxed K-d trees.

Theorem 7. *If $w\neq 00\ldots0$ and $w\neq 11\ldots1$, the expected cost $\mathbb{E}[P_{n,w}]$ of a sliced partial match in a random relaxed K-d tree of size n w.r.t. a random query point in $[0,1]^K$ and the pattern w is*

$$\mathbb{E}[P_{n,w}]=\mathrm{Vol}\big(\widehat{Q}_w\big)\cdot\beta(\rho)\cdot n^{\alpha(\rho)}+\mathcal{O}(1),$$

where $\rho=\mathrm{order}(w)/K$, $\alpha\equiv\alpha(x)=(\sqrt{9-8x}-1)/2$, $\beta(x)=\Gamma(2\alpha+1)/((1-x)(\alpha+1)\alpha^3\Gamma^3(\alpha))$, and $\mathrm{Vol}(\widehat{Q}_w)$ is the probability that a data point falls inside the proper slice $\widehat{Q}_w$ of a randomly centered query. Furthermore, $\mathbb{E}[P_{n,00\ldots0}]=\mathrm{Vol}(\widehat{Q}_{00\ldots0})\cdot n$ and $\mathbb{E}[P_{n,11\ldots1}]=2\cdot\mathrm{Vol}(\widehat{Q}_{11\ldots1})\cdot(H_{n+1}-1)$, where $H_n=\sum_{1\leqslant j\leqslant n}1/j=\log n+\gamma+\mathcal{O}(1/n)$ denotes the nth harmonic number.

Proof. The recurrence for $\mathbb{E}[P_{n,w}]$ is

$$\mathbb{E}[P_{n,w}] = \mathrm{Vol}(\widehat{Q}_w) + \frac{1}{n}\sum_{0\leqslant k<n}\left[\rho\cdot\left(\frac{k+1}{n+1}\mathbb{E}[P_{k,w}] + \frac{n-k}{n+1}\mathbb{E}[P_{n-k,w}]\right)\right.$$
$$\left. + (1-\rho)\cdot\left(\mathbb{E}[P_{k,w}] + \mathbb{E}[P_{n-k,w}]\right)\right],$$

where ρ and $\mathrm{Vol}(\widehat{Q}_w)$ are defined as in the statement of the theorem.

Let $y_w(x) = \sum_{n\geqslant 0}\mathbb{E}[P_{n,w}]\,x^n$. Then it is easy to show that $y_w(x)$ satisfies

$$x y_w''(x) - 2\frac{2x-1}{1-x}y_w'(x) - 2\frac{2-\rho-x}{(1-x)^2}y_w(x) - 2\frac{\mathrm{Vol}(\widehat{Q}_w)}{(1-x)^3} = 0, \tag{2}$$

and the initial conditions are $y_w(0) = 0$ and $y_w'(0) = \mathrm{Vol}(\widehat{Q}_w)$. The linear differential equation satisfied by the generating function $y(x)$ of the expected cost of standard partial matches is almost the same as (2), except that the independent term there is $-2/(1-x)^3$ and $y'(0) = 1$. It is straightforward then to show that $y_w(x) = \mathrm{Vol}(\widehat{Q}_w)\cdot y(x)$. The statement of the theorem now immediately follows from the known asymptotic estimates for the expected cost of standard partial matches in random relaxed K-d trees [14]. The special cases $w = 00\ldots0$ ($\rho = 0$) and $w = 11\ldots1$ ($\rho = 1$) are similarly handled; they are even easier, as the first behaves like a full traversal of the tree, whereas the second behaves like an exact search in a binary search tree. □

Computing the volumes $\mathrm{Vol}(\widehat{Q}_w)$ of proper slices turns out to be a difficult task, both because our model allows queries to fall partially off the boundaries and the data points do not have to be uniformly distributed. On the other hand, if Δ_is are large then the gap between the lower and upper bounds of Theorem 6 is significant; besides, the random model loses interest as the "frame" around the data region is also too large. But if the Δ_is tend to 0 as $n\to\infty$ (in other words, the number of reported points does not grow linearly with n) and the data points are uniformly distributed, then we can easily establish the following corollary.

Corollary 8. *Given a random relaxed K-d tree storing n uniformly and independently drawn data points in $[0,1]^K$, the expected cost of a random range search of sides $\Delta_0,\ldots,\Delta_{K-1}$ (with $\Delta_i\to 0$ as $n\to\infty$) with center uniformly and independently drawn from Z_Δ, is given by*

$$\mathbb{E}[R_n] \sim \Delta_0\cdots\Delta_{K-1}\cdot n + \sum_{1\leqslant j<K} c_j\cdot n^{\alpha(j/K)}$$
$$+ 2\cdot(1-\Delta_0)\cdots(1-\Delta_{K-1})\cdot\log n + \mathcal{O}(1),$$

where

$$c_j = \beta(j/K)\cdot\sum_{w:\ \mathrm{order}(w)=j}\left(\prod_{i:\ w_i=0}\Delta_i\right)\cdot\left(\prod_{i:\ w_i=1}(1-\Delta_i)\right).$$

Proof. The corollary follows from Theorem 6 and the asymptotic estimates given in Theorem 7. When $\Delta_i \to 0$ as $n \to \infty$, we have $V(Z_\Delta) \to 1$; hence the lower and upper bounds in Theorem 6 match and

$$\mathbb{E}[R_n] \sim \sum_{w \in (0+1)^K} \mathbb{E}[P_{n,w}].$$

Finally, observe that for the uniform distribution, and provided that the Δ_is are small enough, we have

$$\mathrm{Vol}(\widehat{Q}_w) \sim \Bigg(\prod_{i:\ w_i=0} \Delta_i \Bigg) \cdot \Bigg(\prod_{i:\ w_i=1} (1-\Delta_i) \Bigg). \qquad \square$$

Notice that the term $\Delta_0 \cdots \Delta_{K-1} \cdot n$ in $\mathbb{E}[R_n]$ is the expected number of reported points and hence the overhead is $\mathcal{O}(n^{\alpha(1/K)})$.

The result above assumes a random model where the queries may fall partially outside $[0,1]^K$, but it also gives a very good approximation to the average cost of range searches in a random model where the queries must completely fall inside $[0,1]^K$ (again, provided that $\Delta_i \to 0$ for $0 \leqslant i < K$ and the centers and data points are uniformly distributed).

4. Other multidimensional data structures

It is important to stress that no assumptions were made with respect to the way that discriminants are assigned during the construction of the tree, so all theorems of Section 3.2 and Theorem 6 apply to standard K-d trees [1], squarish K-d trees [7], K-d-t trees [6] and other variants. It turns out that the theorems also apply to quadtrees [2] without change.

Furthermore, similar arguments to that of Theorem 7 are also valid, relating the expected cost of sliced partial matches to the expected cost of standard partial matches. In general, when $w \neq 00\ldots0$ and $w \neq 11\ldots1$ for the data structures mentioned above we have

$$\mathbb{E}[P_{n,w}] = \beta_w \cdot \mathrm{Vol}(\widehat{Q}_w) \cdot n^{\alpha(\rho)} + \mathcal{O}(1),$$

where $\rho = \mathrm{order}(w)/K$, $\alpha(x) = 1 - x + \phi(x)$, and β_w is a constant depending on w. The expected cost of range search, when $\Delta_i \to 0$, takes hence the form

$$\begin{aligned} \mathbb{E}[R_n] \;=\; & \Delta_0 \cdots \Delta_{K-1} \cdot n + \sum_{1 \leqslant j < K} c_j \cdot n^{\alpha(j/K)} \\ & + 2 \cdot (1-\Delta_0) \cdots (1-\Delta_{K-1}) \cdot \log n + \mathcal{O}(1), \end{aligned} \tag{3}$$

where $c_j = \sum_{w:\ \mathrm{order}(w)=j} \beta_w \cdot \mathrm{Vol}(\widehat{Q}_w)$.

Different data structures are characterized by different αs and βs. For standard K-d trees the necessary analysis is from [11] where it was shown that $\phi(x) < 0.07$ and that it is the unique real solution of

$$\big(\phi(x)+3-x\big)^{x}\big(\phi(x)+2-x\big)^{1-x}-2=0.$$

No closed expression for the βs is given, but their values can be explicitly computed with some effort; for $K \leqslant 4$ all the numerical values are given in [11].

K-d-t trees are like standard K-d trees (when $t=0$ they coincide) but subject to local rebalancing of subtrees of size $\geqslant 2t+1$ [6]; for this variant $\phi(x)=\phi_t(x)$ is the unique solution of

$$\begin{aligned}&\big[\big(\phi(x)+3+t-x\big)\big(\phi(x)\cdots\big(\phi(x)+3+2t-x\big)\cdots\big)\big]^{x}\\&\quad\cdot\big[\big(\phi(x)+2+t-x\big)\big(\phi(x)\cdots\big(\phi(x)+2+2t-x\big)\cdots\big)\big]^{1-x}\\&\qquad-\frac{2t+2!}{t+1!}=0.\end{aligned}$$

The authors provide the value of β_w for some specific patterns, as well as the expected cost of standard partial matches for several values of t and n, but the supplied data is not sufficient to provide asymptotic estimates of the cost of range searches and either experiments or considerable additional analytic work using the techniques in [6] would be necessary to obtain the values of the βs.

Squarish K-d trees [7] have optimal performance since $\phi(x)=0$; however, the values of the βs are not yet known and the only way for the moment to compute them would be through experimental measurement. For quadtrees the analysis of partial match can be found in [10]; $\alpha(x)$ is the same as for standard K-d trees. But the β_ws depend only on K and the order of w. For $K=2$, we have $\beta_{01}=\beta_{10}=\Gamma(2\alpha+2)/(2\alpha^3\Gamma^3(\alpha))\approx 1.5950991$, but no explicit form is given for higher dimensions.

Last, but not least, the theorems in Section 3 also apply to K-d tries, relaxed K-d tries and quadtries. These multidimensional data structures are space-oriented rather than data-oriented, since they induce partitions of the space that are independent of the data points (except that the recursive subdivision of the space stops when the corresponding region contains only one or no points). In order to apply the theorems of Section 3, we only need to assume that each internal node "contains" the middle point of the hyperplane(s) associated to the internal node, to meaningfully define sliced partial matches in these data structures. The average cost of range searches in these digital data structures satisfies (3), but the β_ws involve a fluctuating periodic term (depending on n) of small amplitude and bounded by a constant. For instance, for relaxed K-d tries [14] we have $\alpha=\alpha(x)=\log_2(2-x)$ and $\beta_w=\beta(\text{order}(w)/K)$ with

$$\beta(x)=\frac{1}{\log 2}\big((\alpha-1)\Gamma(-\alpha)+\delta(\log_2 n)\big),$$

and $\delta(\cdot)$ a periodic function of period 1, mean 0 and small amplitude that also depends on α.

5. Experimental results

We have conducted a series of preliminary experiments to validate the theoretical analysis of the previous sections and to explore its limitations, when the input data does not fulfill some of the hypotheses of the random model.

Each "sample point" in our experiments consisted of a random relaxed and standard K-d tree of size n and dimension K, both built from the same sequence of random insertions. Up to Q range searches with random queries of given edge lengths were requested in both trees. And this was repeated for T pairs of trees of each size and dimension.

The legends of the plots indicate the different parameter values: dimension (K), size (n), number of trees per size (T), number of queries per tree (Q), and edge lengths (Δ). Each plot depicts the empirical expected cost of range search against the theoretical predicted value.

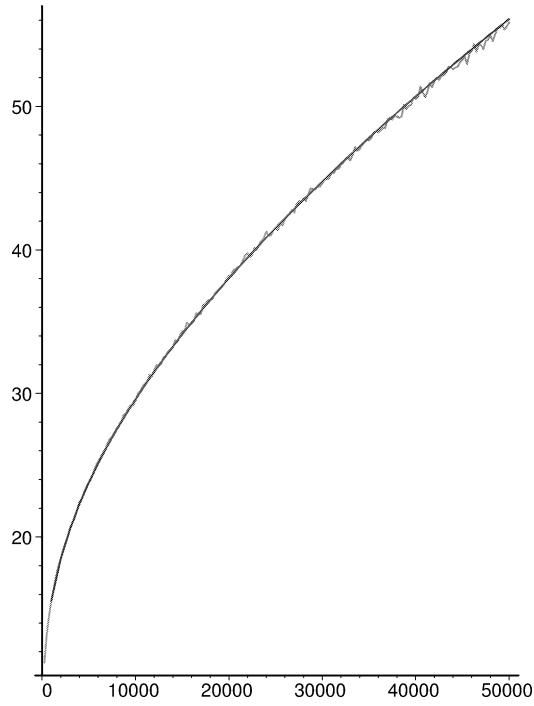

Fig. 5. Relaxed K-d trees ($K = 2$, $n \leqslant 50{,}000$, $T = 300$, $Q = 100$, $\Delta = [0.01, 0.01]$).

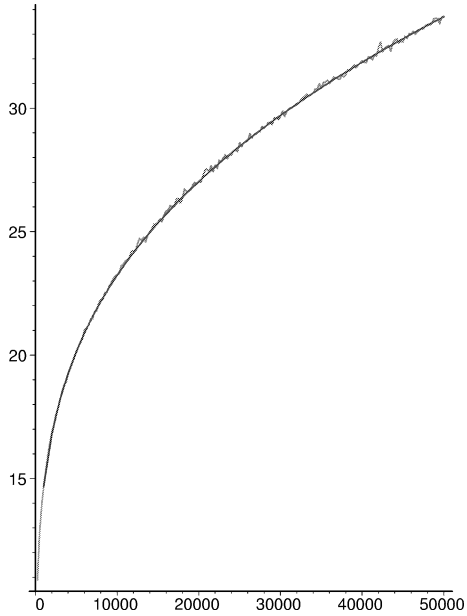

Fig. 6. Relaxed K-d trees ($K = 3$, $n \leqslant 50{,}000$, $T = 300$, $Q = 100$, $\Delta = [0.01, 0.01, 0.01]$).

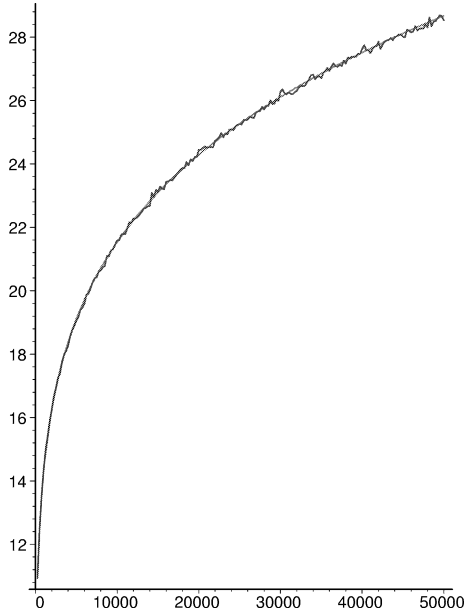

Fig. 7. Relaxed K-d trees ($K = 4$, $n \leqslant 50{,}000$, $T = 300$, $Q = 100$, $\Delta = [0.01, 0.01, 0.01, 0.01]$).

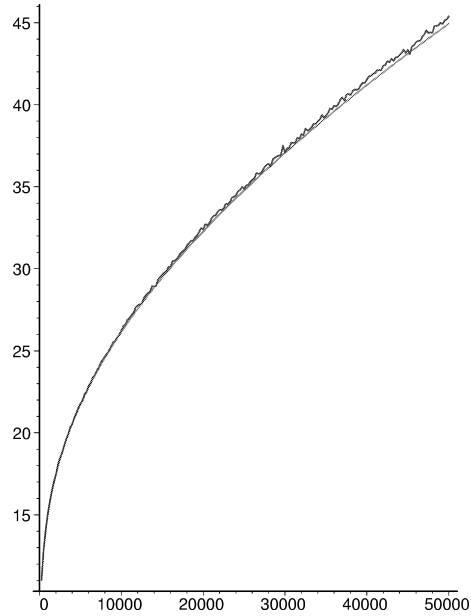

Fig. 8. Standard K-d trees ($K = 2$, $n \leqslant 50{,}000$, $T = 300$, $Q = 100$, $\Delta = [0.01, 0.01]$).

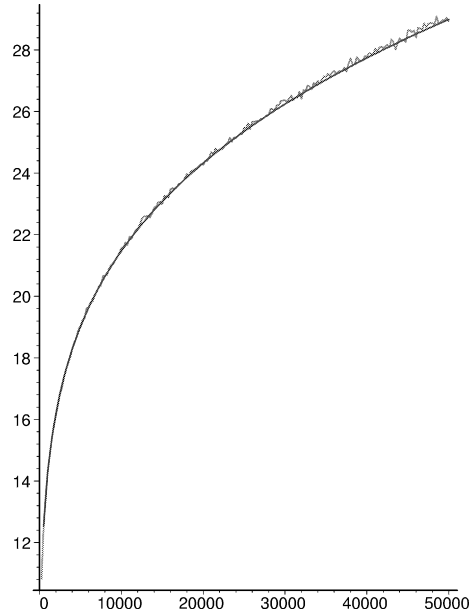

Fig. 9. Standard K-d trees ($K = 3$, $n \leqslant 50{,}000$, $T = 300$, $Q = 100$, $\Delta = [0.01, 0.01, 0.01]$).

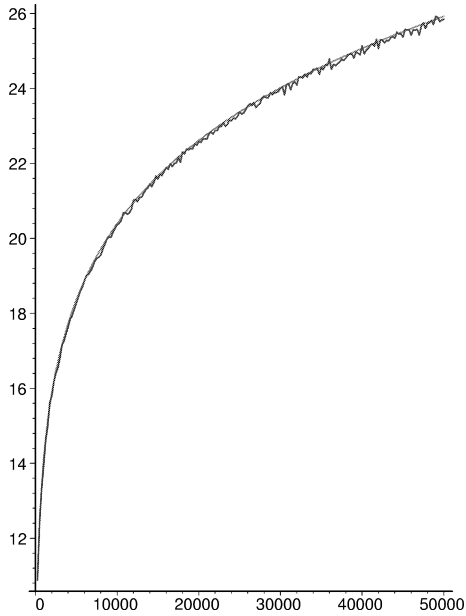

Fig. 10. Standard K-d trees ($K = 4$, $n \leqslant 50{,}000$, $T = 300$, $Q = 100$, $\Delta = [0.01, 0.01, 0.01, 0.01]$).

The experiments suggest that the variance of the cost of range search is high. We conjecture that if $\mathbb{E}[R_n] = a \cdot n + b \cdot n^{\alpha} + o(n^{\alpha})$, the variance of R_n is $\Theta(n^{2\alpha})$.

We have also performed experiments using the same setting as for the first set of experiments, but the data points and query centers were drawn from a clustered distribution (see, for example, Fig. 11).

In order to generate these clustered distributions, a number of "clusters" is fixed in advance and the centers of the clusters uniformly and independently generated in $[0, 1]^K$. To generate a data point, a cluster is selected at random with

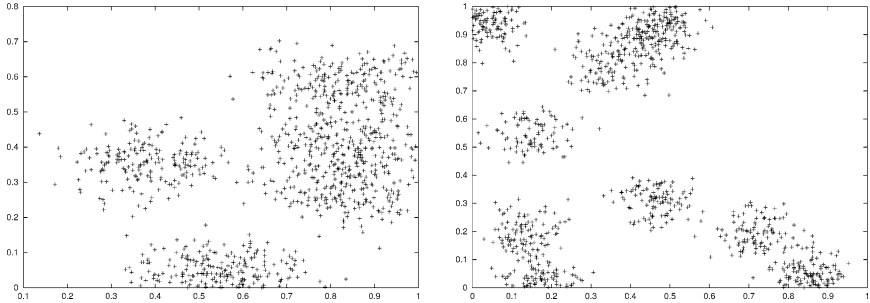

Fig. 11. Two clustered distributions of $n = 1000$ points in $[0, 1]^2$ (left: $c = 5, \sigma = [0.1, 0.05]$; right: $c = 10, \sigma = [0.05, 0.05]$).

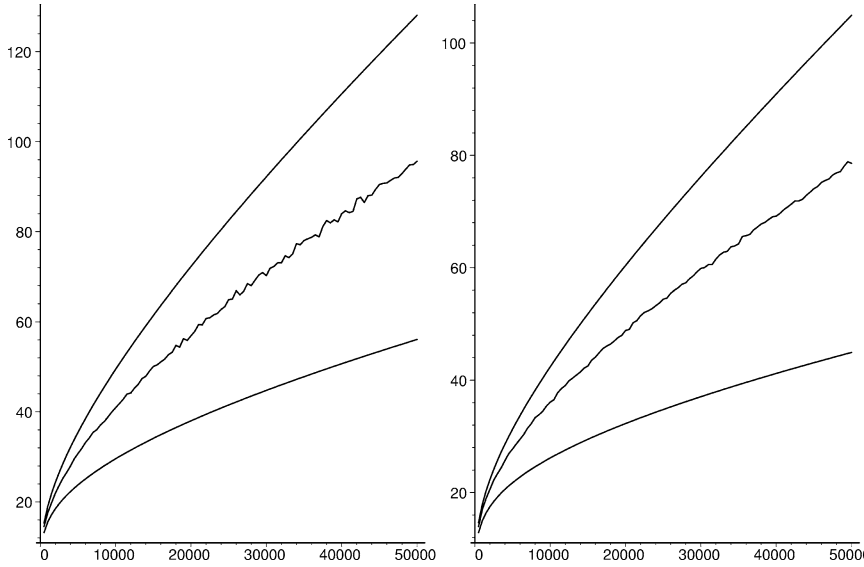

Fig. 12. Clustered distribution, $K = 2$, $n \leqslant 50\,000$, $T = 50$, $Q = 1000$, $\Delta = [0.01, 0.01]$. Left: relaxed K-d trees; right: standard K-d tree. The upper curves in the plots are the theoretical predictions with estimated proper slices' volumes; the middle curves show empirically measured costs; the lower curves are the theoretical predictions with uniform estimates for proper slices' volumes.

identical probability and each coordinate of the point is then generated according to a normal law whose mean is the corresponding coordinate of the center of the cluster. If the generated data point falls outside $[0, 1]^K$ then it is rejected and the procedure repeated.

In our experiments, the number of clusters was $c = 10$ and the standard deviations σ_i of the normal laws were taken all identical to 0.05. The centers of the queries were generated according to the same distribution as the data points.

There are two problems for the application of Corollary 8 to data and range queries generated according to the model above. First, it is very difficult, if not impossible, to analytically compute the volumes of the proper slices. We have prepared a program to compute the approximate values of these volumes for given Δ's, by repeatedly "throwing" queries with the given edge lengths and a large number of data points for each query, keeping frequency counts of the event "point falls in $\widehat{Q}_w$." For instance, when $K = 2$ and $\Delta = [0.01, 0.01]$ we have the estimates $\mathrm{Vol}(\widehat{Q}_{00}) = 0.000868$, $\mathrm{Vol}(\widehat{Q}_{01}) = 0.022335$, $\mathrm{Vol}(\widehat{Q}_{10}) = 0.019432$, and $\mathrm{Vol}(\widehat{Q}_{11}) = 0.957365$, which are significantly different from the volumes for the given values of Δ in the uniform distribution.

The second and most important problem is that the data points and query centers are not independent, so that Corollary 8 cannot actually be applied.

However, our experiments show that it still provides reasonable approximations to the observed data.

The plots for clustered input show the results of the experiments together with:

(a) the estimates given by Corollary 8 as if the points where uniformly distributed;
(b) the estimates obtained by plugging the "experimental" values of the proper slices' volumes (see above) into the formula of Corollary 8.

The programs to conduct the experiments where written in C (using the GNU `gcc-2.8.1` compiler), and run under Solaris 5.7 in a Sun Ultra 5 workstation. AWK scripts where used as a front-end to the C program for easier interaction and for data analysis. The plots were produced with Maple 6 and `gnuplot`. The full suite of programs and data files is available by request from the corresponding author.

References

[1] J.L. Bentley, Multidimensional binary search trees used for associative retrieval, Comm. ACM 18 (9) (1975) 509–517.
[2] J.L. Bentley, R.A. Finkel, Quad trees: A data structure for retrieval on composites keys, Acta Inform. 4 (1974) 1–9.
[3] J.L. Bentley, J.H. Friedman, Data structures for range searching, ACM Comput. Surv. 11 (4) (1979) 397–409.
[4] J.L. Bentley, D.F. Stanat, Analysis of range searches in quad trees, Inform. Process. Lett. 3 (6) (1975) 170–173.
[5] P. Chanzy, L. Devroye, C. Zamora-Cura, Analysis of range search for random K-d trees, Acta Inform. 37 (2001) 355–383.
[6] W. Cunto, G. Lau, Ph. Flajolet, Analysis of kdt-trees: kd-trees improved by local reorganisations, in: F. Dehne, J.-R. Sack, N. Santoro (Eds.), Workshop on Algorithms and Data Structures (WADS'89), Lecture Notes in Comput. Sci., Vol. 382, Springer, 1989, pp. 24–38.
[7] L. Devroye, J. Jabbour, C. Zamora-Cura, Squarish k-d trees, SIAM J. Comput. 30 (2000) 1678–1700.
[8] A. Duch, V. Estivill-Castro, C. Martínez, Randomized k-dimensional binary search trees, in: K.-Y. Chwa, O.H. Ibarra (Eds.), Int. Symposium on Algorithms and Computation (ISAAC'98), Lecture Notes in Comput. Sci., Vol. 1533, Springer, 1998, pp. 199–208.
[9] A. Duch, C. Martínez, On the average performance of orthogonal range search in multidimensional data structures, in: Proc. of the 29th Int. Coloq. on Automata, Languages and Programming (ICALP), Lecture Notes in Comput. Sci., Springer, 2002, accepted.
[10] Ph. Flajolet, G. Gonnet, C. Puech, J.M. Robson, Analytic variations on quadtrees, Algorithmica 10 (1993) 473–500.
[11] Ph. Flajolet, C. Puech, Partial match retrieval of multidimensional data, J. ACM 33 (2) (1986) 371–407.
[12] P. Kirschenhofer, H. Prodinger, Multidimensional digital searching–alternative data structures, Random Structures and Algorithms 5 (1) (1994) 123–134.
[13] H.M. Mahmoud, Evolution of Random Search Trees, Wiley, 1992.

[14] C. Martínez, A. Panholzer, H. Prodinger, Partial match queries in relaxed multidimensional search trees, Algorithmica 29 (12) (2001) 181–204.
[15] R.L. Rivest, Partial-match retrieval algorithms, SIAM J. Comput. 5 (1) (1976) 19–50.
[16] H. Samet, The Design and Analysis of Spatial Data Structures, Addison–Wesley, 1990.
[17] Y.V. Silva-Filho, Average case analysis of region search in balanced k-d trees, Inform. Process. Lett. 8 (5) (1979) 219–223.