Range Search and Nearest Neighbor Search in K-d Trees

Philippe Chanzy

Computer Science
McGill University, Montreal

November 1993

A thesis submitted to the Faculty of Graduate Studies and Research in partial fulfillment of the requirements of the degree of Master of Science.

Abstract

This thesis presents an analysis of the expected complexity of range searching and nearest neighbor searching in random 2-d trees. We show that range searching in a random rectangle $\Delta_x \times \Delta_y$ can be done in $O\left[\Delta x \Delta y\, n + (\Delta x + \Delta y)\, n^{\alpha} + \ln n\right]$ expected time. A matching lower bound is also obtained. We also show that nearest neighbor searching in random 2-d trees by any algorithm must take time bounded by $\Omega\left[n^{\alpha - 1/2}/(\ln n)^{\alpha}\right]$ where $\alpha = (\sqrt{17} - 3)/2$. This disproves a conjecture by Bentley that nearest neighbor search in random 2-d trees can be done in $O(1)$ expected time.

Résumé

Cette thèse présente une analyse de la complexité moyenne du *range searching* et du *nearest neighbor searching* dans un arbre bidimensionnel (*2-d tree*) quelconque. D'une part, nous démontrons que le *range searching* dans un rectangle quelconque de taille $\Delta_x \times \Delta_y$ peut être exécuté en un temps moyen en $O\left[\Delta x \Delta y\ n + (\Delta x + \Delta y)\ n^{\alpha} + \ln n\right]$. Nous calculons aussi la borne inférieure correspondante. D'autres parts, nous montrons qu' indépendamment de l'algorithm utilisé, le *nearest neighbor searching* dans un 2-d tree quelconque prend un temps en $\Omega\left[n^{\alpha-1/2}/(\ln n)^{\alpha}\right]$ pour $\alpha = (\sqrt{17} - 3)/2$. Ceci nous permet de rejeter une conjecture de Bentley affirmant que le *nearest neighbor search* dans un *2-d tree* peut être éxecuté en un temps moyen de $O(1)$.

Acknowledgments

I would like to acknowledge the School of Computer Science, the office and technical support staff for their assistance in all aspects of my research.

I would like to thank my supervisor, Luc Devroye, for assisting me with my thesis, for his comments and suggestions and for giving me the opportunity to visit Stanford.

I would also like to thank my parents and family for their support and encouragement as well as Mr., Mrs. and Etienne Laurent for their warm hospitality.

Finally, I wish to express my gratefulness to all my friends who have stood by me, listened to my useless remarks and took coffee breaks with me. In particular, I thank Mary for correcting my English day after day, Sandro for his perfect knowledge of *Latex* and Charles, Alexis, François and Claudio for their encouraging comments.

Contents

1 Introduction

The multidimensional binary search tree or **k-d tree** was introduced by Bentley in 1975 [1], it is a data structure used for storing multikey records and for sorting and searching multidimensional data. The k-d tree is a generalization of the one-dimensional binary search tree; every node is used to split the remaining elements into those that are larger and those that are smaller according to a particular key or coordinate. This coordinate is determined by the level of the node in the tree in a rotational fashion. Level 0 is assigned the first coordinate and level i is assigned the $(1 + i \bmod k)^{th}$ coordinate.

A range search consists of finding all points in a hyper-rectangle. With respect to range searching, the k-d tree seems to be as efficient as the **quadtree**, which is another multidimensional data structure introduced by Bentley in 1974 [5]. In this thesis, we study range searching and its implications for random 2-d trees, obtained on the basis of n independent identically distributed random variables in the unit square. Bentley and Lee claimed that the upper bound for k-dimensional range searching for both quadtrees [11] [5] and k-d trees [11] [2] with n nodes is $O(n^{1-1/k} + F)$ where F is the number of nodes lying within the search region. Based on empirical tests Bentley also claimed that nearest neighbor search in random k-d trees could be performed in logarithmic expected time [1]. Furthermore, he proved that if a splitting discriminator is chosen at every level of the k-d tree, we can build an optimized k-d tree in which the nearest neighbor problem can be solved in $O(\ln n)$ expected time [9]. Finally he also claimed that range searching in k-d trees could probably be done in $O(\ln n + F)$ expected time where F is the number of nodes found in the search region [3] [2].

However, more recently Flajolet and Puech proved that the partial match retrieval problem in which s of the k fields are specified, could be solved in $\Theta(n^{1-s/k+\theta(s/k)})$ expected number of comparisons for random quadtrees with n nodes [8] and $O(n^{1-s/k+\theta(s/k)})$ for random k-d trees with n nodes [7] [6] where $\theta(u)$ is a strictly positive function. This disproves Bentley's upper bound $O(n^{1-s/k})$ [1] that appears to hold only for very particular cases of k-d trees. They also proved that in k-d tries, the partial match problem takes $O(n^{1-s/d})$ expected time and concluded that 2-d tries are thus more efficient than both quadtrees and 2-d trees which respectively take $\Theta(n^{(\sqrt{17}-3)/2})$ and $O(n^{(\sqrt{17}-3)/2})$ comparisons [13].

Section 2 gives a brief description of definitions and notations used in this thesis. Section 3 presents an algorithm and section 4 contains an analysis of range search. In section 7, we give one algorithm for searching for the nearest neighbor of a point. Then, in section 8, we give a lower bound for nearest neighbor searching of a randomly chosen node in a 2-d tree that contradicts Bentley's conjecture that the expected cost of a random node nearest neighbor search is $O(1)$.

2 Definitions and notations

For a 2-d tree, points in the plane are split alternately according to x and y coordinates. Hence, if we consider a sequence of n independent random points identically and uniformly distributed on the unit square $[0,1]^2$, $U_1, \ldots, U_n$, we can build a 2-d tree by successively inserting the corresponding nodes $u_1, \ldots, u_n$. Starting with the unit square as initial rectangle R_0, each new node u_t is inserted in the smallest rectangle, R_{u_t}, containing it and splits R_{u_t} horizontally (y-split) or vertically (x-split) into 2 smaller and disjoint rectangles R'_{u_t} and R''_{u_t} (Figure 1), which are called $u_t - rectangles$.

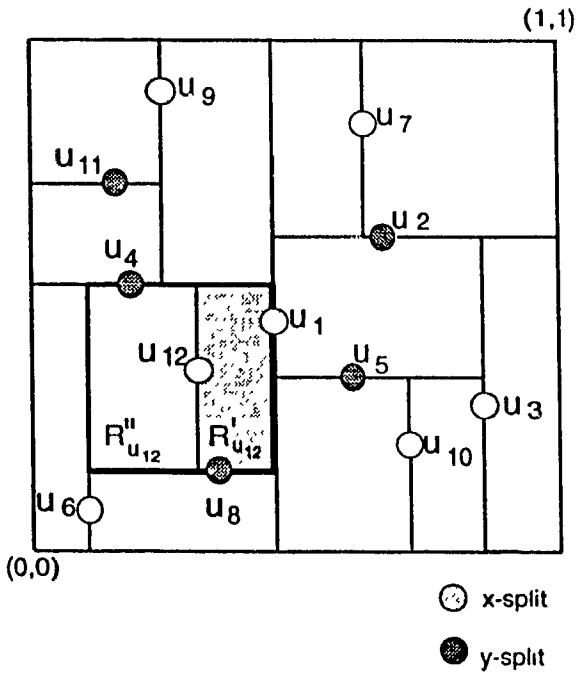

Figure 1: Insertion of u_{12} in T: a vertical split.

We observe that after $U_1, \ldots, U_n$ are inserted, the square is partitioned into $n+1$ rectangles. However, the total number of rectangles, including overlapping ones, is equal to $2n+1$. Each rectangle in the partition thus created is closed. We say that a rectangle R "meets" a rectangle R' if and only if the interiors of R and R' intersect. Let $\mathcal{B}_i$ be the set of all the rectangles created by the first i insertions. Starting with a set $\mathcal{B}_0$ containing the unit square R_0 as a unique element, $\mathcal{B}_i$ is deduced from $\mathcal{B}_{i-1}$ by adding the two new $u_i - rectangles$ R'_{u_i} and R''_{u_i} created by the ith insertion,

$$\mathcal{B}_i = \mathcal{B}_{i-1} \cup \{u_i - rectangles\} = \mathcal{B}_{i-1} \cup \{R'_{u_i}, R''_{u_i}\}.$$

Observe that $|\mathcal{B}_i| = 2i+1$. Finally each k-d tree T may be associated with a binary search tree T_b(Figure 2) By induction we can show that the probability of obtaining a random binary search tree T_b by inserting random nodes in a k-d tree is the same as that of obtaining T_b by inserting random nodes in a standard binary search tree (Bentley, 1975) [1]. Therefore, binary search trees corresponding to random 2-d trees have the same structural properties as random binary search trees. We write l_{u_i}, r_{u_i} for the left and right children of u_i, and denote the coordinates of node u_i by (x_{u_i}, y_{u_i}). Note that u_1 is the root of T.

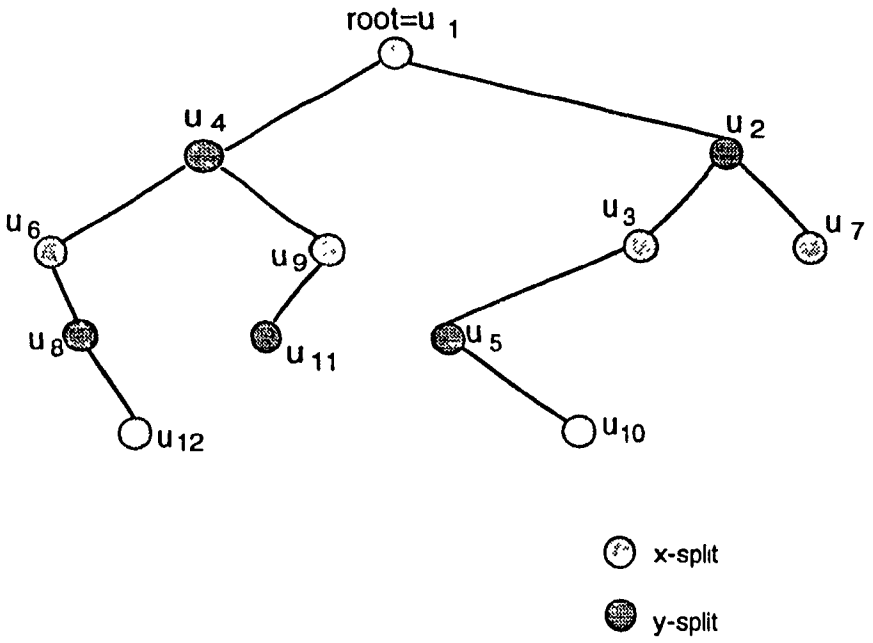

Figure 2: Binary search tree T_b for figure 1.

3 Range search algorithm

Given a 2-d tree T with n nodes $u_1, \ldots, u_n$, the range search problem is that of finding all the points lying within a rectangle Q centered at Z, where Z is a random point uniformly distributed on $[0,1]^2$, and has dimensions

$$Q = [x_{min}, x_{max}] \times [y_{min}, y_{max}].$$

$Q-rectangles$ are rectangles having the same dimensions as Q (Figure 3). In this thesis, we describe an algorithm for range search and analyze its expected time performance. Define $\Delta_x = x_{max} - x_{min}, \Delta_y = y_{max} - y_{min}$.

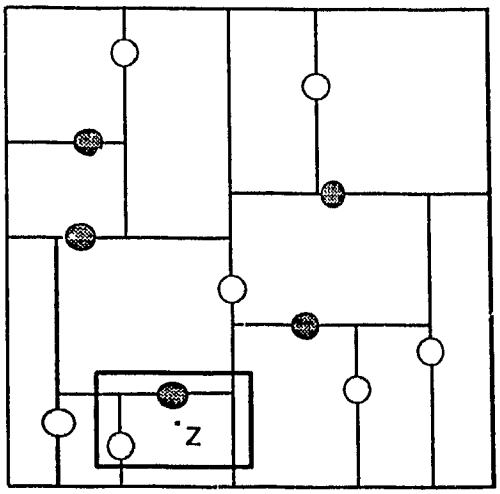

Figure 3: A search rectangle Q.

The following recursive algorithm was suggested by Bentley (1975) [1] (see also Gonnet and Baeza-Yates [10] or Wood [14]). It returns Γ, the set of all the nodes in T lying within our search rectangle Q.

RANGE_SEARCH(T, Q)

Note: we denote $Q = [x_{min}, x_{max}] \times [y_{min}, y_{max}]$ and let Z be the center of Q. This algorithm returns the set Γ of points in T that are covered by Q. We also denote T_u the subtree rooted at u.

$u \leftarrow$ root$[T]$, $\Gamma \leftarrow \emptyset$
if $|T| = 0$
  return Γ
if $u \in Q$ then
  $\Gamma \leftarrow \{u\}$
if $|T| = 1$
  return $\Gamma \leftarrow \Gamma \cup$ RANGE_SEARCH(empty_tree, Q)
if u splits T according to its x-coordinate
  case
    $x_u \leq x_{min}$: $\Gamma \leftarrow \Gamma \cup$ RANGE_SEARCH(T_{r_u}, Q) (a)
    $x_u \geq x_{max}$: $\Gamma \leftarrow \Gamma \cup$ RANGE_SEARCH(T_{l_u}, Q) (b)
    $x_{min} \leq x_u \leq x_{max}$: $\Gamma \leftarrow \Gamma \cup$ RANGE_SEARCH(T_{r_u}, Q) (c)
      $\Gamma \leftarrow \Gamma \cup$ RANGE_SEARCH(T_{l_u}, Q)
else {u splits T according to its y-coordinate}
  case
    $y_u \leq y_{min}$: $\Gamma \leftarrow \Gamma \cup$ RANGE_SEARCH(T_{r_u}, Q) (a)
    $y_u \geq y_{max}$: $\Gamma \leftarrow \Gamma \cup$ RANGE_SEARCH(T_{l_u}, Q) (b)
    $y_{min} \leq y_u \leq y_{max}$: $\Gamma \leftarrow \Gamma \cup$ RANGE_SEARCH(T_{r_u}, Q) (c)
      $\Gamma \leftarrow \Gamma \cup$ RANGE_SEARCH(T_{l_u}, Q)
return Γ

end {RANGE_SEARCH}

4 Analysis of range search

Bentley's algorithm traverses all the paths starting from u_1 which are likely to contain a node lying within Q. A search on a path ends either when a leaf is reached or when the rectangle R_{u_i} of the unique child u_i of the node u_j passed by the algorithm does not meet Q. Hence, none of the non-visited nodes lies within Q; furthermore we may conclude that the set Γ returned by the algorithm is exactly the set of all the nodes in T lying within Q.

Lemma 1. A node u_i is visited by the algorithm if and only if at least one u_i – *rectangle* meets Q. Then, for each node u_i visited, the number of u_i – *rectangles* meeting our search region Q is equal to the number of descendants of u_i passed by the algorithm.

PROOF: In cases (a) and (b) of the algorithm, every node u_i visited by the algorithm splits a rectangle R_{u_i} meeting Q into two $u_i - rectangles$ rectangles (R'_{u_i} and R''_{u_i}) such that R'_{u_i} meets Q and R''_{u_i} does not (Figure 4).

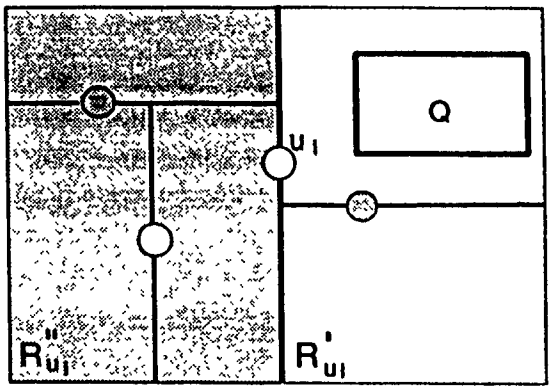

Figure 4: in cases (a) and (b), R'_{u_i} meets Q and R''_{u_i} does not.

In case (c) of Bentley's algorithm, the node u_i passed divides a rectangle R_{u_i} meeting Q into two rectangles R'_{u_i} and R''_{u_i}, both of which meet Q (Figure 5).

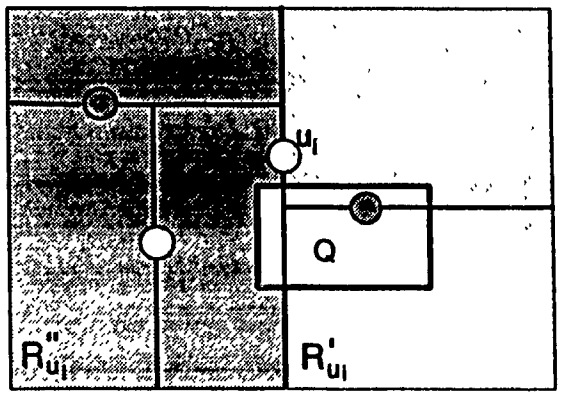

Figure 5: in case (c), R'_{u_i} and R''_{u_i} meet Q.

Conversely, let us now assume that there exists a node u_i which has not been visited but defines two $u_i - rectangles$ (R'_{u_i}, R''_{u_i}) such that at least one meets Q. This means that on the path from u_1 to u_i, there is an ancestor of u_i, u_j, such that either u_j splits T according to x_{u_j} where $x_{u_j} \leq x_{min}$ and u_i is in the subtree rooted at l_{u_i}, or $x_{u_j} \geq x_{max}$ and u_i is in the subtree rooted at r_{u_i}, or u_j splits T according to y_{u_j} where $y_{u_j} \leq y_{min}$ and u_i is in the

subtree rooted at l_{u_t}, or $y_{u_j} \geq y_{max}$ and u_t is in the subtree rooted at r_{u_t}. These four cases are symmetrical. We assume without loss of generality that we are in the first situation. Then, $x_{u_t} < x_{u_j} \leq x_{min}$ and u_t lies within one $u_j - rectangle$, denoted by R'_{u_j}, that does not meet Q. Hence, the two $u_t - rectangles$ being included in R'_{u_j} do not meet Q. This contradicts the hypothesis made on u_t and we can conclude that for any node u_t in T, if one $u_t - rectangle$ meets Q then this implies that u_t must have been visited by the algorithm.■

Lemma 2. The number of nodes visited by the algorithm, N, is equal to the number of rectangles in $\mathcal{B}_n$ meeting Q.

PROOF: Let Υ be the set of nodes visited by the algorithm.

$$\begin{aligned} N &= |\Upsilon| \\ &= 1 + \sum_{u_t \in \Upsilon} \text{number of children of } u_t \text{ visited by a RANGE\_SEARCH} \\ &= 1 + \sum_{u_t \in \Upsilon} \text{number of } u_t - rectangles \text{ meeting } Q. \end{aligned}$$

According to Lemma 1, if a node u_t is not in Υ then no $u_t - rectangle$ meets Q, hence since Υ is a subset of $\{u_1, u_2, \ldots, u_n\}$,

$$\begin{aligned} N &= 1 + \sum_{u_t \in \{u_1, \ldots, u_n\}} \text{number of } u_i - rectangles \text{ meeting } Q \\ &= \text{number of rectangles in } \mathcal{B}_n \text{ meeting } Q. \end{aligned}$$

■

Theorem 1. The expected number of rectangles in $\mathcal{B}_n$ meeting our search rectangle Q centered at Z and of size $\Delta x \times \Delta y$ is such that

$$\mathbf{E}\{N\} \leq \gamma \left(\Delta x \Delta y \; n + (\Delta x + \Delta y) \; n^{\alpha} + \ln n \right),$$

where γ is a positive constant and $\alpha = \frac{\sqrt{17}-3}{2} \approx 0.56$.

PROOF: Let $\mathcal{B}_n = \{R_t : t \in \{0, 1, \ldots, 2n\}\}$ and let R_i be a rectangle of size $a_t \times b_i$ (a_t=x-length, b_t=y-length). We partition the unit square $[0,1]^2$ into a grid with N_Q tiny $Q - rectangles$ of size $\Delta x \times \Delta y$ (Figure 6). Observe that

$$N_Q = \left\lceil \frac{1}{\Delta x} \right\rceil \times \left\lceil \frac{1}{\Delta y} \right\rceil.$$

Since R_t is a random rectangle, $\mathbf{P}$[one edge of R_i is on the grid] = 0. Therefore, R_t meets n_t $Q - rectangles$ (Figure 7), where

$$n_t = \left(2 + \left\lfloor \frac{a_t}{\Delta x} \right\rfloor\right) \left(2 + \left\lfloor \frac{b_i}{\Delta y} \right\rfloor\right)$$

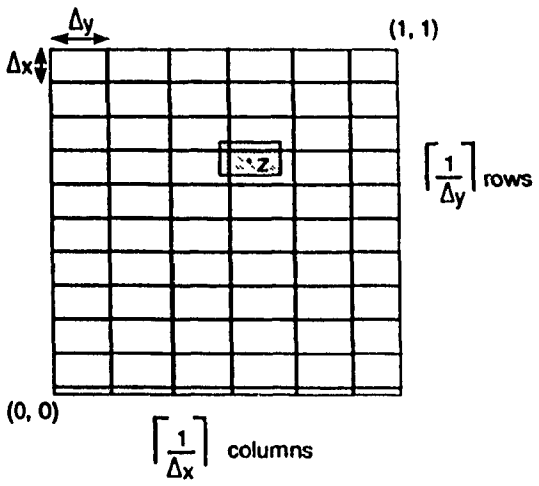

Figure 6: Q meets four adjacent $Q-rectangles$.

$$\leq \left(2+\frac{a_i}{\Delta x}\right)\left(2+\frac{b_i}{\Delta y}\right)$$
$$\leq 4+2(a_i+b_i)\left(\frac{1}{\Delta x}+\frac{1}{\Delta y}\right)+\frac{a_i b_i}{\Delta x \Delta y}.$$

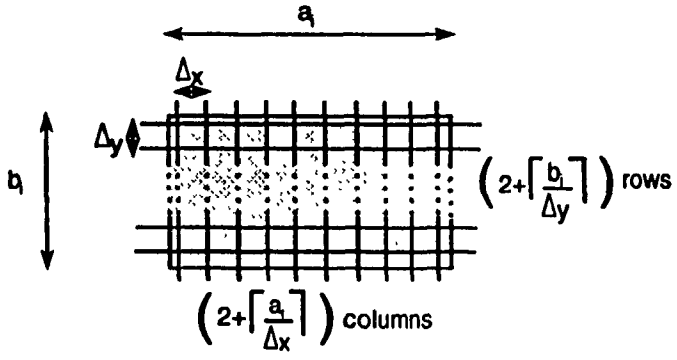

Figure 7: R_i meets n_i $Q-rectangles$.

Also,

$$n_i \geq \left(1+\frac{a_i}{\Delta x}\right)\left(1+\frac{b_i}{\Delta y}\right).$$

Then,

$$\sum_{i=0}^{2n} n_i \leq 8n + 4 + 2\left(\frac{1}{\Delta x} + \frac{1}{\Delta y}\right)\sum_{i=0}^{2n}(a_i + b_i) + \frac{1}{\Delta x \Delta y}\sum_{i=0}^{2n} a_i b_i.$$

$$\mathbf{E}\left\{\sum_{i=0}^{2n} n_i\right\} \leq 12n + 2\left(\frac{1}{\Delta x} + \frac{1}{\Delta y}\right)\mathbf{E}\left\{\sum_{i=0}^{2n}(a_i + b_i)\right\} + \frac{1}{\Delta x \Delta y}\mathbf{E}\left\{\sum_{i=0}^{2n} a_i b_i\right\}.$$

Thus, $\mathbf{E}\left\{\sum_{i=1}^{2n} n_i\right\}$ may be considered as the expected number of pairs $(R_i, Q_j), (R_i \in B_n, Q_j \in \mathcal{Q} = \{Q_1, \ldots, Q_{N_Q}\})$ such that R_i meets Q_j and the expected number of rectangles R_i in $\mathcal{B}_n$ meeting Q is given by

$$\begin{aligned}
&\mathbf{E}\left\{\sum_{i=0}^{2n} I_{[R_i \text{ meets } Q]}\right\} \\
&\leq \mathbf{E}\left\{\sum_{i=0}^{2n}\sum_{j=1}^{N_Q} I_{[R_i \text{ meets } Q_j]} I_{[Q_j \text{ meets } Q]}\right\} \\
&\leq \mathbf{E}\left\{\sum_{k=1}^{N_Q} I_{[Z \in Q_k]} \sum_{i=0}^{2n}\sum_{j=1}^{N_Q} I_{[Q_j \text{ is } Q_k \text{ or one of its neighbors}]} I_{[R_i \text{ meets } Q_j]}\right\} \\
&\leq \mathbf{E}\left\{\sum_{i=0}^{2n}\sum_{j=1}^{N_Q} I_{[R_i \text{ meets } Q_j]} \sum_{k=1}^{N_Q} I_{[Z \in Q_k]} I_{[Q_k \text{ is } Q_j \text{ or one of its neighbors}]}\right\} \\
&\leq \mathbf{E}\left\{\sum_{i=0}^{2n}\sum_{j=1}^{N_Q} I_{[R_i \text{ meets } Q_j]}\right\} \mathbf{E}\left\{\sum_{k=1}^{N_Q} I_{[Z \in Q_k]} I_{[Q_k \text{ is } Q_j \text{ or one of its neighbors}]}\right\}
\end{aligned}$$

since the latter two variables are independent. The area of any Q — *rectangle* in Q is not greater than $\Delta x \Delta y$; therefore the probability that Z lies in Q_k is less than $\Delta x \Delta y$. Furthermore Q_k has at most eight neighbors, hence given a rectangle R_i and a Q — *rectangle* Q_i,

$$\begin{aligned}
&\mathbf{E}\left\{\sum_{k=1}^{N_Q} I_{[Z \in Q_k]} I_{[Q_k \text{ is } Q_j \text{ or one of its neighbors}]}\right\} \\
&= \sum_{k=1}^{N_Q} \mathbf{P}\left[Z \in Q_k\right] I_{[Q_k \text{ is } Q_j \text{ or one of its neighbors}]} \\
&\leq 9\Delta x \Delta y.
\end{aligned}$$

The expected number of rectangles R_i in $\mathcal{B}_n$ meeting Q is not greater than

$$9\Delta x \Delta y \mathbf{E}\left\{\sum_{i=0}^{2n}\sum_{j=1}^{N_Q} I_{[R_i \text{ meets } Q_j]}\right\} = 9\Delta x \Delta y \mathbf{E}\left\{\sum_{i=0}^{2n} n_i\right\}$$

and finally,

$$\mathbf{E}\{N\} \leq 108\Delta x \Delta y\, n + 18(\Delta x + \Delta y)\mathbf{E}\left\{\sum_{i=0}^{2n}(a_i + b_i)\right\} + 9\mathbf{E}\left\{\sum_{i=0}^{2n} a_i b_i\right\}.$$

We will now see in the following Lemmas how to evaluate the two last terms of our upper bound.

Lemma 3. Let $a_i \times b_i$ be the size of rectangle R_i in $\mathcal{B}_n$, $0 \leq i \leq 2n$. Then

$$\mathbf{E}\left\{\sum_{i=0}^{2n}(a_i + b_i)\right\} = O(n^{\frac{\sqrt{17}-3}{2}}) \approx O(n^{0.56}).$$

PROOF: We may represent the successive insertions in T by a binary search tree T_b' with $2n+1$ nodes, where each node represents a rectangle R_i of $\mathcal{B}_n$ having either no descendant (if R_i is no more divided by any insertion), or two descendants (if a new point is inserted in R_i). Let $(V_1, \ldots, V_{k+1}, W_1, \ldots, W_{k+1})$ be independent random variables uniformly and identically distributed on $[0,1]$ and suppose the tree begins with an x-split. Then, starting at level 0 with $R_0 = (a_0 = 1, b_0 = 1)$, on level 1 of the tree we find two rectangles whose horizontal and vertical lengths form couples of random variables distributed as $(V_1, 1)$ and $(1 - V_1, 1)$. The sizes of both rectangles on level 1 are therefore distributed as $(V_1 \times 1)$. Similarly, on level two, for each existing rectangle its size is distributed as $(V_1 \times W_1)$. Then, by induction, each node at level $2k$ $(k \geq 2)$ represents a rectangle whose size is distributed as $\left(\prod_{i=1}^{k} V_i \times \prod_{i=1}^{k} W_i\right)$ and at level $2k+1$ $(k \geq 1)$ each node corresponds to a rectangle whose size is distributed as $\left(\prod_{i=1}^{k+1} V_i \times \prod_{i=1}^{k} W_i\right)$. We notice that a_i and b_i do not have the same distribution since the choice of the coordinate for the first splitting affects the shape of the tree. Let us find an upper bound for

$$\sum_{i=0}^{2n}(\mathbf{E}\{a_i\} + \mathbf{E}\{b_i\}) = \sum_{i=0}^{2n}\mathbf{E}\{a_i\} + \sum_{i=0}^{2n}\mathbf{E}\{b_i\}.$$

For any rectangle R represented by a node u at level $2k-1$ in T_b', the expected number S of points lying within R in T is the expected size of the subtree rooted at N in T_b'

$$\mathbf{E}\{S\} = (\lfloor \ldots \lfloor \lfloor nV_1 \rfloor \times W_1 \rfloor \ldots \times V_m \rfloor \times W_m \rfloor \times \ldots \times V_{k-1} \rfloor \times W_{k-1} \rfloor \times V_k \rfloor).$$

For a new node to fall within R and create rectangles on level $2k$, we must have $\mathbf{E}\{S\} \geq 1$. Thus, on level $2k$, the existence of each node is given by

$$I_{\{\lfloor \ldots \lfloor \lfloor nV_1 \rfloor \times W_1 \rfloor \ldots \times V_m \rfloor \times W_m \rfloor \times \ldots \times V_{k-1} \rfloor \times W_{k-1} \rfloor \times V_k \rfloor \geq 1\}},$$

which can be bounded from above by

$$I_{\{n(V_1 \times \ldots \times V_k \times W_1 \times \ldots \times W_{k-1}) \geq 1\}}$$

and bounded from below by

$$I_{\{n(V_1 \times \ldots \times V_k \times W_1 \times \ldots \times W_{k-1}) \geq 2k\}}.$$

We can then write for the 2^{2k} nodes on level $2k$ ($k \geq 2$),

$$\begin{aligned}\mathbf{E}\{a_{2k}\} &\leq \mathbf{E}\left\{V_1 \times \ldots \times V_k \; I_{\{nV_1 \times \ldots \times V_k \times W_1 \times \ldots \times W_{k-1} \geq 1\}}\right\} \\ &\leq \mathbf{E}\left\{V_1 \times \ldots \times V_{k-1} \; I_{\{nV_1 \times \ldots \times V_{k-1} \times W_1 \times \ldots \times W_{k-1} \geq 1\}}\right\}\end{aligned}$$

and

$$\begin{aligned}\mathbf{E}\{b_{2k}\} &\leq \mathbf{E}\left\{W_1 \times \ldots \times W_k \; I_{\{nV_1 \times \ldots \times V_k \times W_1 \times \ldots \times W_{k-1} \geq 1\}}\right\} \\ &\leq \mathbf{E}\left\{W_1 \times \ldots \times W_{k-1} \; I_{\{nV_1 \times \ldots \times V_{k-1} \times W_1 \times \ldots \times W_{k-1} \geq 1\}}\right\}.\end{aligned}$$

Similarly, for the 2^{2k+1} nodes on level $2k+1$ ($k \geq 1$),

$$\begin{aligned}\mathbf{E}\{a_{2k+1}\} &\leq \mathbf{E}\left\{V_1 \times \ldots \times V_{k+1} \; I_{\{nV_1 \times \ldots \times V_k \times W_1 \times \ldots \times W_k \geq 1\}}\right\} \\ &\leq \mathbf{E}\left\{V_1 \times \ldots \times V_k \; I_{\{nV_1 \times \ldots \times V_k \times W_1 \times \ldots \times W_k \geq 1\}}\right\}\end{aligned}$$

and

$$\mathbf{E}\{b_{2k+1}\} \leq \mathbf{E}\left\{W_1 \times \ldots \; W_k \times I_{\{nV_1 \times \ldots \times V_k \times W_1 \times \ldots \times W_k \geq 1\}}\right\}.$$

Therefore,

$$\sum_{i=0}^{2n} \mathbf{E}\{a_i + b_i\} \leq C_0 \times \sum_{k=1}^{+\infty} 2^{2k} \mathbf{E}\left\{V_1 \times \ldots \times V_k \; I_{\{nV_1 \times \ldots \times V_k \times W_1 \times \ldots \times W_k \geq 1\}}\right\}.$$

Define

$$\begin{aligned}V &= -\ln(V_1 \times \ldots \times V_k), \\ W &= -\ln(W_1 \times \ldots \times W_k).\end{aligned}$$

We note that V and W are independent gamma(k) random variables, and the density of V is

$$f(v) = \frac{v^{k-1}e^{-v}}{\Gamma(k)} \text{ for } v \geq 0.$$

Then

$$\begin{aligned}&\sum_{k=1}^{+\infty} 2^{2k} \mathbf{E}\left\{V_1 \times \ldots \times V_k \times I_{\{nV_1 \times \ldots \times V_k \times W_1 \times \ldots \times W_k \geq 1\}}\right\} \\ &= \sum_{k=1}^{+\infty} 2^{2k} \mathbf{E}\left\{e^{-V} \times I_{[ne^{-V}e^{-W} \geq 1]}\right\} \\ &= \sum_{k=1}^{+\infty} 2^{2k} \mathbf{E}\left\{e^{-V} \times I_{[V+W \leq \ln n]}\right\} \\ &= \sum_{k=1}^{+\infty} 2^{2k} \int_{v \geq 0} \int_{w \geq 0} f(v) f(w) e^{-v} \times I_{[v+w \leq \ln n]} dw dv \\ &= \int_{v \geq 0} \int_{w \geq 0} \underbrace{\left(\sum_{k=1}^{+\infty} 2^{2k} \frac{v^k w^k}{\Gamma(k)^2}\right)}_{\sigma} \frac{e^{-2v}e^{-w}}{vw} \times I_{[v+w \leq \ln n]} dw dv.\end{aligned}$$

Let first give an upper bound for σ. Define $\beta = 2\sqrt{vw}$.
Since

$$\Gamma(v) \geq \left(\frac{v}{e}\right)^v \sqrt{\frac{2\pi}{v}},$$

we may write

$$\sigma \leq \sum_{k=1}^{+\infty} \frac{\beta^{2k}}{\left(\left(\frac{k}{e}\right)^k \sqrt{\frac{2\pi}{k}}\right)^2} \leq \sum_{k=1}^{+\infty} \frac{k\beta^{2k}}{2\pi(k/e)^{2k}},$$

and, using $k! \leq \left(\frac{k}{e}\right)^k \sqrt{2\pi k} e^{1/12k}$,

$$\begin{aligned}
\sigma &\leq \sum_{k=1}^{+\infty} \frac{e^{1/12}\sqrt{2k}\beta(2\beta)^{2k-1}}{\sqrt{2\pi}(2k-1)!} \\
&\leq \frac{\beta e^{2\beta} e^{1/12}}{\sqrt{2\pi}} \sum_{k=0}^{+\infty} \frac{(2\beta)^k e^{-2\beta}}{k!} \sqrt{k+1} \\
&\leq \frac{\beta e^{2\beta} e^{1/12}}{\sqrt{2\pi}} E\left[\sqrt{P+1}\right],
\end{aligned}$$

where P is a random variable with a Poisson distribution of parameter (2β). Then, Jensen's inequality allows us to write

$$\begin{aligned}
\sigma &\leq \frac{\beta e^{2\beta} e^{1/12}}{\sqrt{2\pi}} \sqrt{2\beta+1} \leq \frac{\beta e^{2\beta} e^{1/12}}{\sqrt{2\pi}} (\sqrt{2\beta}+1) \\
&\leq \frac{e^{1/12}}{\sqrt{2\pi}} (\beta + \sqrt{2}\beta^{3/2}) e^{2\beta}.
\end{aligned}$$

We can now give a new upper bound of our sum

$$\begin{aligned}
&\sum_{k=1}^{+\infty} 2^{2k} \mathbf{E}\left\{V_1 \times \ldots \times V_k \times I_{\{nV_1 \times \ldots \times V_k \times W_1 \times \ldots \times W_k \geq 1\}}\right\} \\
&\leq \int_{v,w\geq 0} \int_{v+w \leq \ln n} \frac{2e^{1/12}}{\sqrt{2\pi}} \left(\frac{1+2(vw)^{1/4}}{\sqrt{vw}}\right) e^{4\sqrt{vw}-2v-w} dw dv.
\end{aligned}$$

If we transform the coordinates by setting

$$\omega = v + w, v = \theta\omega, w = (1-\theta)\omega, 0 \leq \theta \leq 1, \omega > 0,$$

the Jacobian is ω and the above integral reduces to

$$\frac{2e^{1/12}}{\sqrt{2\pi}} \int_{0\leq\theta\leq 1} \int_{0\leq\omega\leq\ln n} \left(\frac{1}{\sqrt{\theta(1-\theta)}} + \frac{2\sqrt{\omega}}{(\theta(1-\theta))^{1/4}}\right) e^{\overbrace{\omega(4\sqrt{\theta(1-\theta)}-\theta-1)}^{\phi(\theta)}} d\omega d\theta.$$

We note that the function $\phi(\theta)$ reaches a maximum at $\theta_0 = \frac{1-1/\sqrt{17}}{2}$ where the maximal value is $\alpha = \phi(\theta_0) = \frac{\sqrt{17}-3}{2}$. Furthermore, ϕ is concave and there exists a constant $\nu > 0$ such that, for $\theta \in [0,1]$

$$\phi(\theta) \leq \phi(\theta_0) - \nu(\theta - \theta_0)^2.$$

If we denote $C_1 = \frac{2e^{1/12}}{\sqrt{2\pi}}$, for all $\epsilon \in (0, \min(\theta_0, 1-\theta_0))$,

$$\sum_{k=1}^{+\infty} 2^{2k} \mathbf{E}\left\{V_1 \times \ldots \times V_k \times I_{\{nV_1 \times \ldots \times V_k \times W_1 \times \ldots \times W_k \geq 1\}}\right\} \leq$$

$$\underbrace{C_1 \int_{|\theta-\theta_0|\leq\epsilon} \int_{0\leq\omega\leq \ln n} \left(\frac{1}{\sqrt{\theta(1-\theta)}} + \frac{2\sqrt{\omega}}{(\theta(1-\theta))^{1/4}}\right) e^{\omega\phi(\theta)} d\omega d\theta}_{I_1}$$

$$+ \underbrace{C_1 \int_{|\theta-\theta_0|\geq\epsilon} \int_{0\leq\omega\leq \ln n} \left(\frac{1}{\sqrt{\theta(1-\theta)}} + \frac{2\sqrt{\omega}}{(\theta(1-\theta))^{1/4}}\right) e^{\omega\phi(\theta)} d\omega d\theta}_{I_2}.$$

INTEGRAL I_1. If we write $\alpha^* = \sup_{|\theta-\theta_0|\leq\epsilon} \left(\frac{1}{\theta(1-\theta)}\right)$, then

$$\begin{aligned}
I_1 &\leq \underbrace{C_1(\alpha^*)^{1/2}}_{C_2} \int_{|\theta-\theta_0|\leq\epsilon} \int_{0\leq\omega\leq\ln n} \left(1 + 2\sqrt{\omega}\right) e^{\omega\phi(\theta)} d\omega d\theta \\
&\leq C_2 \left[\int_{0\leq\theta\leq1} \int_{0\leq\omega\leq1} 3e^{\omega\phi(\theta)} d\omega d\theta + \int_{0\leq\theta\leq1} \int_{1\leq\omega\leq\ln n} 3\sqrt{\omega} e^{\omega\phi(\theta)} d\omega d\theta\right] \\
&\leq C_2 \left[3e^{\alpha} + \int_{\theta\in\Re} \int_{1\leq\omega\leq\ln n} 3\sqrt{\omega} e^{\omega(\alpha-\nu(\theta-\theta_0)^2)} d\omega d\theta\right] \\
&\leq C_2 \left[3e^{\alpha} + \int_{1\leq\omega\leq\ln n} e^{\omega\alpha} \left(\int_{x\in\Re} \frac{3}{\sqrt{\nu}} e^{-x^2} dx\right) d\omega\right] \\
&\leq C_2 \left[3e^{\alpha} + 3\sqrt{\frac{\pi}{\nu}} \int_{1\leq\omega\leq\ln n} e^{\omega\alpha} d\omega\right] \\
&\leq C_2 \left[3e^{\alpha} + \frac{3}{\alpha}\sqrt{\frac{\pi}{\nu}} n^{\alpha}\right] \\
&= O(n^{\alpha}).
\end{aligned}$$

INTEGRAL I_2. If we write $\alpha^{**} = \sup_{|\theta-\theta_0|\geq\epsilon} \phi(\theta)$, then $\alpha^{**} \leq \alpha$ and

$$\begin{aligned}
I_2 &\leq C_1 \int_{0\leq\omega\leq\ln n} \int_{|\theta-\theta_0|\geq\epsilon} \left(\frac{1}{\sqrt{\theta(1-\theta)}} + \frac{1}{(\theta(1-\theta))^{1/4}}\right) 2\sqrt{\omega} e^{\alpha^{**}\omega} d\theta d\omega \\
&\leq 2C_1 e^{\alpha^{**}\ln n} (\ln n)^{3/2} \int_{0\leq\theta\leq1} \left(\frac{1}{\sqrt{\theta(1-\theta)}} + \frac{1}{(\theta(1-\theta))^{1/4}}\right) d\theta.
\end{aligned}$$

This last integral converges so that

$$I_2 \leq C_3 n^{\alpha^{**}} (\ln n)^{3/2} = O(n^{\alpha}).$$

We conclude that

$$\sum_{k=1}^{+\infty} 2^{2k} \mathbf{E}\left\{V_1 \times \ldots \times V_k \, I_{\{nV_1 \times \ldots \times V_k \times W_1 \times \ldots \times W_k \geq 1\}}\right\} = O(n^\alpha),$$

and therefore

$$\sum_{i=0}^{2n} \mathbf{E}\left\{a_i + b_i\right\} = O(n^\alpha) = O(n^{\frac{\sqrt{17}-3}{2}}) \approx O(n^{0.56}).$$

■

Lemma 4. If a_i and b_i are as in Lemma 3,

$$\mathbf{E}\left\{\sum_{i=0}^{2n} a_i b_i\right\} = 2H_{n+1} - 1.$$

PROOF: Define

$$A_k = \sum_{R \in \mathcal{B}_k} \text{area}(R),$$

where R and $\mathcal{B}_k$ are as defined above. The first area A_0 is equal to 1 and by induction, according to the definition of sets $\mathcal{B}_k$ and $\mathcal{B}_{k+1}$, A_{k+1} is defined by the sum of A_k and the areas of the two new u_{k+1} − *rectangles*. These two rectangles form a partition of $R_{u_{k+1}}$ so that A_{k+1} is equal to the sum of A_k and the area of $R_{u_{k+1}}$. Thus, the last iteration returns

$$A_n = \sum_{R \in \mathcal{B}_n} \text{area}(R) = \sum_{i=0}^{2n} a_i b_i.$$

We deduce that

$$\mathbf{E}\left\{\sum_{i=0}^{2n} a_i b_i\right\} = \mathbf{E}\left\{\sum_{R \in \mathcal{B}_n} \text{area}(R)\right\} = \mathbf{E}\left\{A_n\right\},$$

that is,

$$\mathbf{E}\left\{\sum_{i=0}^{2n} a_i b_i\right\} = A_1 + \sum_{i=2}^{n} \mathbf{E}\left\{\text{area}(R_{u_i})\right\}.$$

To evaluate this expression we need the following Lemma that gives a precise calculation of the second term of the above sum. ■

Lemma 5. For $i \geq 2$,

$$\mathbf{E}\left\{\text{area}(R_{u_i})\right\} = \frac{2}{1+i}.$$

PROOF: By definition, R_{u_i} is the smallest rectangle accepting u_i. It is well-known that the i rectangles defined by $u_1, \ldots, u_{i-1}$ are distributed as uniform spacings. That is they are distributed as $(V_{j+1} - V_j)$ where $0 \leq j \leq i-1$ and $0 = V_{(0)} < V_{(1)} < \ldots < V_{(i-1)} < V_{(i)} = 1$,

and $V_{(1)}, \ldots, V_{(i-1)}$ is a permutation of i.i.d uniform [0, 1] random variables $V_1, \ldots, V_{i-1}$. Thus,

$$\begin{aligned}
\mathbf{E}\{\text{area}(R_{u,})\} &= \mathbf{E}\left\{\sum_{j=0}^{i-1}(V_{(j+1)} - V_{(j)})^2\right\} \\
&= i\mathbf{E}\left\{(V_{(1)})^2\right\} \qquad \text{(symmetry)} \\
&= i\int_0^1 x^2 f(x)dx,
\end{aligned}$$

where f is the density function of $V_{(1)}$.
From

$$\begin{aligned}
\mathbf{P}[V_{(1)} \geq x] &= \mathbf{P}[V_1 \geq x] \times \ldots \times \mathbf{P}[V_{i-1} \geq x] \\
&= (1-x)^{i-1} \\
&= \int_x^1 f(y)dy,
\end{aligned}$$

we see that $f(x) = (i-1)(1-x)^{i-2}$ and

$$\mathbf{E}\left\{V_{(1)}^2\right\} = (i-1) \times \int_0^1 x^2(1-x)^{i-2}dx = \frac{2}{i(i+1)}.$$

Thus,

$$\mathbf{E}\{\text{area}(R_{u,})\} = \frac{2}{i+1}.$$

■

CONTINUATION OF PROOF OF Lemma 5.
We deduce that

$$\begin{aligned}
\mathbf{E}\{A_n\} &= 2 + \sum_2^n \frac{2}{i+1} \\
&= 2\sum_1^{n+1} \frac{1}{i} - 1 \\
&= 2H_{n+1} - 1.
\end{aligned}$$

■

CONTINUATION OF PROOF OF Theorem 1.
From the properties of harmonic numbers,

$$\ln(n+2) \leq H_{n+1} \leq 1 + \ln(n+1),$$

and therefore,

$$\mathbf{E}\left\{\sum_{i=0}^{2n} a_i b_i\right\} \sim 2\ln n.$$

Collecting all this yields the bound

$$\begin{aligned} \mathbf{E}\{N\} &\leq 108\Delta x\Delta y\, n + C(\Delta x + \Delta y)n^{\alpha} + 18H_{n+1} - 9 \\ &\leq \gamma\left(\Delta x\Delta y\, n + (\Delta x + \Delta y)\, n^{\alpha} + \ln n\right), \end{aligned}$$

where γ is a positive constant. ■

In section 6, we will show that the bound of Theorem 1 is tight. The above inequality is a sum whose terms reflect the main computations of the algorithm. The first term of the upper bound ($\Delta x\Delta y\ n$) represents the number of points of the 2-d tree lying within the search rectangle and returned by the algorithm. The second term ($(\Delta x + \Delta y)\ n^{\alpha}$) is an upper bound of the expected number of nodes visited outside the search rectangle. These nodes cannot be eliminated since one coordinate lies within the bounds while the other is unknown. Finally, the third term ($\ln n$) corresponds to the expected length of the path from the root to the leaves found in the search rectangle Q.

The expected cost of the algorithm has an order of growth dominated by one of the three terms depending on the given Δx and Δy values. Without loss of generality, we may assume that Δx is larger than Δy. Hence, if $\Delta y = \Omega(1/n^{k_1})$ such that $k_1 \leq 1 - \alpha$ then $\mathbf{E}\{N\} = O(\Delta x\Delta y\ n)$. However, if $\Delta y = \Omega(1/n^{k_1})$ and $O(1/n^{k_2})$ such that $k_1 < \alpha$ and $k_2 > 1 - \alpha$ then the expected cost is $O\left((\Delta x + \Delta y)\ n^{\alpha}\right)$. Furthermore, the second term still dominates if $\Delta y = O(1/n^{k_2})$ when $k_2 \geq \alpha$ and $\Delta x = \Omega(1/n^{k_3})$ when $k_3 < \alpha$. Finally, the expected cost is logarithmic in n when $\Delta y = O(1/n^{k_2})$ and $\Delta x = O(1/n^{k_3})$ if $k_2 \geq \alpha$ and $k_3 \geq \alpha$. We then have $\mathbf{E}\{N\} = O(\ln n)$.

According to the above remarks, in partial match query (i.e. when $(\Delta x, \Delta y) = (1,0)$ (or $(0,1)$)) the expected cost of the algorithm is bounded from above by the second term of our sum, $\mathbf{E}\{N\} = O(n^{\alpha})$. This yields the result proved by Flajolet and Puech [7] for partial match retrieval in k-d trees of size n for $k = 2$.

However, we notice that the upper bound does not confirm Bentley's claim [1] that the expected cost for a partial match query in a k-d tree of size n where s fields are specified is $O(n^{1-s/k})$. This result appears to be valid in the very special case of perfectly balanced k-d trees in which the splitting coordinate at each level of the tree is determined by the data values. Moreover, Bentley's conjecture [2] that the expected cost of range searching is $O(\ln n + F)$ is not verified by our upper bound: in the simple case of $\Delta x = \Delta y = 1/\sqrt{n}$, the expected cost of our algorithm is $O(n^{\alpha - 1/2}) \approx O(n^{0.06})$.

5 The shape of the rectangles

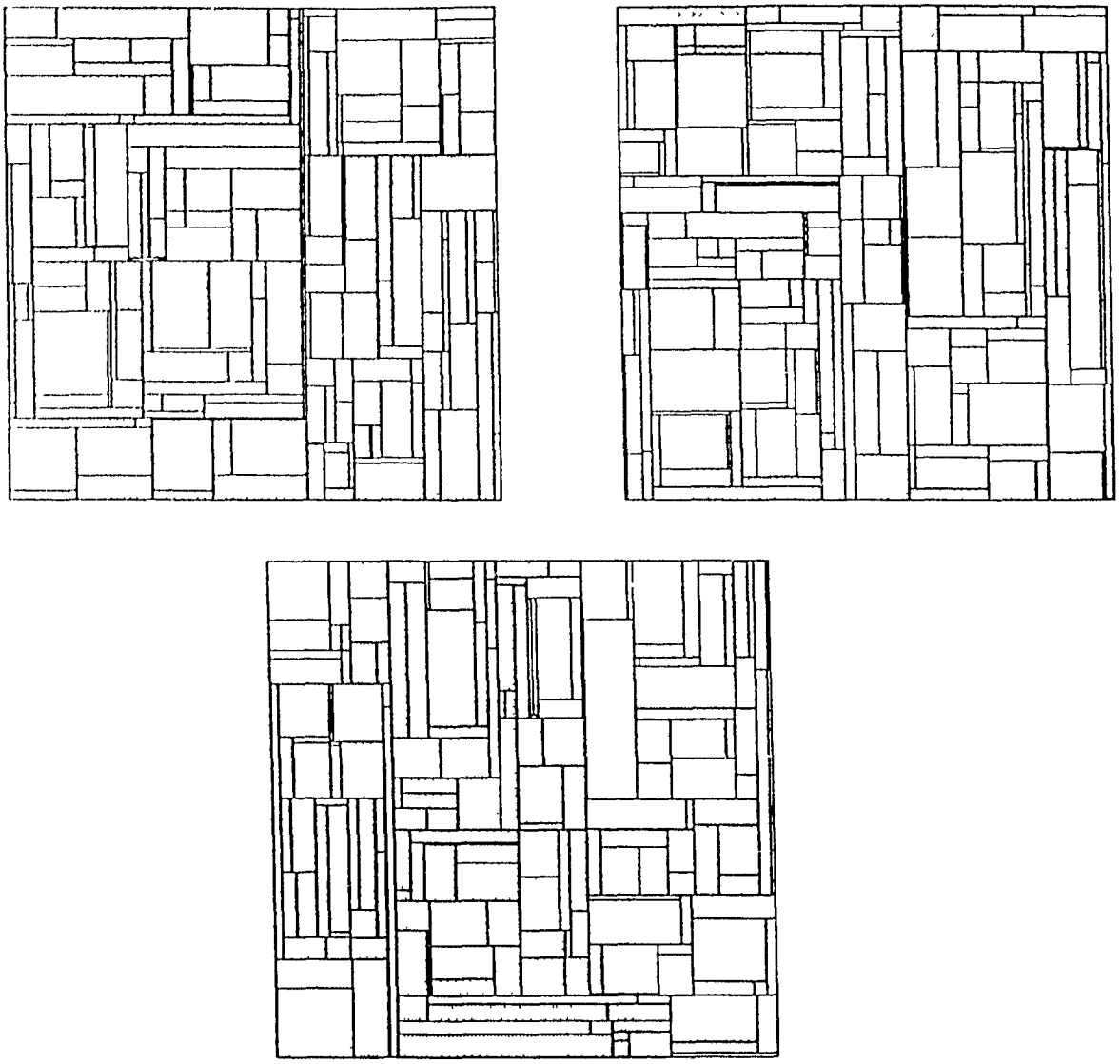

Figure 8: Random 2-d trees. Note the elongated nature of most rectangles.

Let denote X_k and Y_k the horizontal and vertical lengths of a rectangle taken at level k. A rectangle drawn randomly from a node at level $2k$ has edge ratio X_{2k}/Y_{2k} where

$$\frac{X_{2k}}{Y_{2k}} \stackrel{\mathcal{L}}{=} \frac{V_1,\ldots,V_k}{W_1,\ldots,W_k} I_{\{\lfloor\ldots\lfloor\lfloor nV_1\rfloor\times W_1\rfloor\ldots\times V_m\rfloor\times W_m\rfloor\times\ldots\times V_{k-1}\rfloor\times W_{k-1}\rfloor\times V_k\rfloor\geq 1\}}.$$

Therefore, using the same definition as in the previous section,

$$\begin{aligned} V &= -\ln(V_1\times\ldots\times V_k), \\ W &= -\ln(W_1\times\ldots\times W_k), \end{aligned}$$

at level $2k$,

$$\begin{aligned}\frac{X_{2k}}{Y_{2k}} &= e^{W-V} I_{\{\lfloor\ldots\lfloor\lfloor nV_1\rfloor\times W_1\rfloor\ldots\times V_m\rfloor\times W_m\rfloor\times\ .\times V_{k-1}\rfloor\times W_{k-1}\rfloor\times V_k\rfloor\geq 1\}} \\ &\geq e^{W-V} I_{\{nV_1\times\ .\ \times V_k\times W_1\times\ldots\times W_{k-1}\geq 2k\}} \\ &\geq e^{W-V} I_{\{nV_1\times\ldots\times V_k\times W_1\times\ .\times W_k\geq 2k+1\}}.\end{aligned}$$

In the same way, at level $2k+1$, we have

$$\begin{aligned}\frac{X_{2k+1}}{Y_{2k+1}} &\geq \frac{V_1,\ldots,V_k}{W_1,\ldots,W_{k+1}} I_{\{nV_1\times\ldots\times V_k\times W_1\times\ .\ \times W_k\geq 2k+1\}} \\ &\geq \frac{V_1,\ \ldots,V_k}{W_1,\ldots,W_k} I_{\{nV_1\times.\ \times V_k\times W_1\times\ \ldots\times W_k\geq 2k+1\}} \\ &\geq e^{W-V} I_{\{nV_1\times\ldots\times V_k\times W_1\times.\ \times W_k\geq 2k+1\}}.\end{aligned}$$

Thus, we have the following inequalities

$$\begin{aligned}\min\left\{\frac{X_{2k}}{Y_{2k}}, \frac{X_{2k+1}}{Y_{2k+1}}\right\} &\geq \frac{V_1,\ldots,V_k}{W_1,\ldots,W_k} I_{\{nV_1\times\ldots\times V_k\times W_1\times\ldots\times W_k\geq 2k+1\}} \\ &\geq e^{W-V}\times I_{[V+W\leq\ln(\frac{n}{2k+1})]}.\end{aligned}$$

We note that V and W are both gamma(k) distributed and that according to Central Limit Theorem,

$$\frac{W-V}{\sqrt{2k}} \xrightarrow{\mathcal{L}} \mathcal{N}(0,1),$$

where $\xrightarrow{\mathcal{L}}$ denotes convergence in distribution, and $\mathcal{N}(0,1)$ is the standard normal law. This means that the above ratios have an order of growth of approximately $e^{\sqrt{2k}}$. For $k=\Theta(\ln n)$, very high values of X_{2k}/Y_{2k} and X_{2k+1}/Y_{2k+1} reflect the elongated shape of the rectangles.

Therefore, it appears that most of the rectangles in our 2-d tree are very long (Figure 8). This is responsible for the $O(n^\alpha)$ upper bound calculated for the expected sum of the perimeters of the rectangles in $\mathcal{B}_n$ as the perimeter (a_i+b_i) of R_i decreases very slowly with i. In other words, this disproportion explains such a high number of $Q-$ *rectangles* meeting R_i. It also explains why the k-d tree is virtually useless for nearest neighbor search.

6 A lower bound for range searching

The upper bound for the expected cost of the range search given in Theorem 1 was composed of three terms reflecting the main computations of the algorithm. Here, we show that Theorem 1 is tight.

Theorem 2. The expected number of nodes visited by a range search in a rectangle Q centered at Z and of size $\Delta x \times \Delta y$ is such that

$$\mathbf{E}\{N\} = \Omega\left[\Delta x \Delta y\, n + (\Delta x + \Delta y)\left(\frac{n}{\ln n}\right)^{\alpha} + \ln n\right],$$

where $\alpha = \frac{\sqrt{17}-3}{2} \approx 0.56$.

PROOF: Three terms clearly lower bound the expected cost of the range search.
PART 1: By definition the range search returns all the points lying in a rectangle of size $\Delta x \times \Delta y$, hence, since the expected number of points in this region is $(\Delta x \Delta y\, n)$, the algorithm takes at least $\Omega(\Delta x \Delta y\, n)$ expected time.
PART 2: The search is done by visiting a random binary search tree and cannot be completed unless a leaf is reached. Therefore, the expected minimal distance from the root to a leaf in a random binary search tree with n nodes being $\Theta(\ln n)$ [4], the expected cost of the range search can be bounded from below by $\Omega(\ln n)$.
PART 3: Now, we evaluate that the number n_i of $Q-$*rectangles* of size $\Delta x \times \Delta y$ meeting a random rectangle R_i of the 2-d tree T,

$$n_i \geq \left(1 + \frac{a_i}{\Delta x}\right)\left(1 + \frac{b_i}{\Delta y}\right).$$

Thus, we can write

$$\mathbf{E}\left\{\sum_{i=0}^{2n} n_i\right\} \geq 2n + \frac{1}{\Delta x}\mathbf{E}\left\{\sum_{i=0}^{2n} a_i\right\} + \frac{1}{\Delta y}\mathbf{E}\left\{\sum_{i=0}^{2n} b_i\right\} + \frac{1}{\Delta x \Delta y}\mathbf{E}\left\{\sum_{i=0}^{2n} a_i b_i\right\}.$$

Since at least one fourth of our random search rectangle centered at M meets the unit square (Figure 6), we deduce that any random search rectangle meets at least 1/16th of a $Q-$*rectangle* of the grid. Thus, the expected number of nodes visited by the range search is

$$\begin{aligned}\mathbf{E}\{N\} &\geq \frac{\Delta x \Delta y}{16}\mathbf{E}\left\{\sum_{i=0}^{2n} n_i\right\} \\ &\geq \frac{1}{16}\left(\Delta x \Delta y\, n + \Delta y \mathbf{E}\left\{\sum_{i=0}^{2n} a_i\right\} + \Delta x \mathbf{E}\left\{\sum_{i=0}^{2n} b_i\right\} + \mathbf{E}\left\{\sum_{i=0}^{2n} a_i b_i\right\}\right).\end{aligned}$$

A third lower bound for the expected cost of the range search algorithm results then from

$$\sum_{i=0}^{2n} \mathbf{E}\{a_i\} = \Omega\left[\left(\frac{n}{\ln n}\right)^{\alpha}\right],$$

$$\sum_{i=0}^{2n} \mathbf{E}\{b_i\} = \Omega\left[\left(\frac{n}{\ln n}\right)^{\alpha}\right].$$

We now show these bounds. We have

$$\begin{aligned}\mathbf{E}\{a_{2k}\} &\geq \mathbf{E}\left\{V_1 \times \ldots \times V_k \; I_{\{nV_1 \times \ldots \times V_k \times W_1 \times \ldots \times W_{k-1} \geq 2k\}}\right\} \\ &\geq \mathbf{E}\left\{V_1 \times \ldots \times V_k \; I_{\{nV_1 \times \ldots \times V_k \times W_1 \times \ldots \times W_k \geq 2k\}}\right\},\end{aligned}$$

and

$$\begin{aligned}\mathbf{E}\{b_{2k}\} &\geq \mathbf{E}\left\{W_1 \times \ldots \times W_k \; I_{\{nV_1 \times \ldots \times V_k \times W_1 \times \ldots \times W_{k-1} \geq 2k\}}\right\} \\ &\geq \mathbf{E}\left\{W_1 \times \ldots \times W_k \; I_{\{nV_1 \times \ldots \times V_k \times W_1 \times \ldots \times W_k \geq 2k\}}\right\}.\end{aligned}$$

For all $k \geq 1$ we also have

$$\mathbf{E}\{a_{2k-1}\} \geq \mathbf{E}\{a_{2k}\},$$

and

$$\mathbf{E}\{b_{2k-1}\} \geq \mathbf{E}\{b_{2k}\}.$$

Thus, there exists a constant $C_0 > 0$ such that

$$\sum_{i=0}^{2n} \mathbf{E}\{a_i\} \geq C_0 \sum_{k=1}^{\infty} 2^{2k} \mathbf{E}\left\{V_1 \times \ldots \times V_k \; I_{\{nV_1 \times \ldots \times V_k \times W_1 \times \ldots \times W_k \geq 2k\}}\right\}.$$

With the same notations as in the proof of the upper bound (Lemma 1), we obtain

$$\begin{aligned}\sum_{i=0}^{2n} \mathbf{E}\{a_i\} &\geq C_0 \int_{v \geq 0} \int_{w \geq 0} \left(\sum_{k=1}^{\infty} 2^{2k} \frac{v^k w^k}{\Gamma(k)^2}\right) \frac{e^{-2v} e^{-w}}{vw} \, I_{[v+w \leq \ln \frac{n}{2k}]} \, dw dv \\ &\geq C_0 \int_{v \geq 0} \int_{w \geq 0} \left(\underbrace{\sum_{k=\delta \ln n}^{2 \ln n} 2^{2k} \frac{v^k w^k}{\Gamma(k)^2}}_{\sigma}\right) \frac{e^{-2v} e^{-w}}{vw} \, I_{[v+w \leq \ln \frac{n}{4 \ln n}]} \, dw dv.\end{aligned}$$

where δ is an arbitrary constant in (0,2). By Stirling's approximation, as in the proof of the upper bound, we note that

$$\sigma \geq C_1 \beta e^{2\beta} \mathbf{E}\{\sqrt{1+P} \times I_{[2\delta \ln n \leq P \leq 4 \ln n]}\}$$

where $\beta = 2\sqrt{vw}$, P is a random with a Poisson distribution of parameter (2β) and C_1 is a positive constant. This is bounded from below by

$$\begin{cases} C_2 \beta e^{2\beta} \sqrt{\ln n} & \text{if } \delta \ln n \leq \beta \leq 2 \ln n, \\ 0 & \text{otherwise,} \end{cases}$$

where C_2 is another positive constant. But $\sqrt{\ln n} \geq (vw)^{1/4}$, so that we have the lower bound

$$\sigma \geq C_3 \beta^{3/2} e^{2\beta} I_{[\delta \ln n \leq \beta \leq 2 \ln n]}.$$

This leads to a lower bound equal to

$$C_4 \int_{v,w\geq 0} \int_{v+w\leq \ln\left(\frac{n}{2k}\right), 2\sqrt{vw}\in[\delta \ln n, 2\ln n]} e^{4\sqrt{vw}-2v-w}(vw)^{1/4} dv dw.$$

With the transformation used in the upper bound,that is

$$\omega = v + w, v = \theta\omega, w = (1-\theta)\omega, 0 \leq \theta \leq 1, \omega > 0,$$

with a Jacobian equal to ω, the latter expression reduces to

$$C_4 \int_{0\leq\theta\leq 1, 0\leq\omega\leq \ln\left(\frac{n}{2k}\right)} \int_{\omega\sqrt{\theta(1-\theta)}\in\left[\frac{\delta}{2}\ln n, \ln n\right]} (\theta(1-\theta))^{1/4} \omega^{3/2} e^{\omega\phi(\theta)} d\omega d\theta.$$

We know that ϕ reaches its maximum at $\theta_0 = \frac{1-1/\sqrt{17}}{2}$ and takes the value $\alpha = \phi(\theta_0) = \frac{\sqrt{17}-3}{2}$. We also note that $\phi(\theta) \geq \alpha - \nu(\theta-\theta_0)^2$ for some $\nu > 0$ and give a new lower bound

$$C_4 \int_{\frac{1}{4}\leq\theta\leq\frac{1}{2}} \int_{\frac{\delta}{2}\ln n\leq\omega\leq\ln\left(\frac{n}{4\ln n}\right)} \left(\frac{3}{16}\right)^{1/4} \sqrt{\omega} e^{\omega(\alpha-\nu(\theta-\theta_0)^2)} d\omega d\theta.$$

By the normal integral we have the following inequality

$$\int_{\frac{1}{4}}^{\frac{1}{2}} e^{\omega\nu(\theta-\theta_0)^2} d\theta \geq \frac{C_5}{\sqrt{\omega\nu}}.$$

for some constant C_5. Thus we obtain as lower bound

$$\left(\frac{3}{16}\right)^{1/4} \frac{C}{\sqrt{\nu}} \int_{\frac{\delta}{2}\ln n}^{\ln\left(\frac{n}{4\ln n}\right)} \omega e^{\omega\alpha} d\omega \geq \left(\frac{3}{16}\right)^{1/4} \frac{C}{\sqrt{\nu}} \int_{\frac{\delta}{2}\ln n}^{\ln\left(\frac{n}{4\ln n}\right)} e^{\omega\alpha} d\omega = \Theta\left[\left(\frac{n}{\ln n}\right)^{\alpha}\right]$$

if $\delta < 2$. We have then proved that

$$\sum_{i=0}^{2n} \mathbf{E}\{a_i\} = \Omega\left[\left(\frac{n}{\ln n}\right)^{\alpha}\right].$$

Similarly we may prove that

$$\sum_{i=0}^{2n} \mathbf{E}\{b_i\} = \Omega\left[\left(\frac{n}{\ln n}\right)^{\alpha}\right].$$

Collecting all this yields a general lower bound for the expected cost of the range search;

$$\mathbf{E}\{N\} = \Omega\left[\Delta x \Delta y\, n + (\Delta x + \Delta y)\left(\frac{n}{\ln n}\right)^{\alpha} + \ln n\right].$$

■

REMARK: More careful but tedious work in the truncation allows us to get rid of the $1/(\ln n)^\alpha$ factor in the above lower bound. We used the inequality

$$\lfloor \ldots \lfloor \lfloor nV_1 \rfloor \times V_2 \rfloor \ldots \rfloor \times V_k \rfloor \geq nV_1 \times \ldots \times V_k - \underbrace{\lceil 1 + V_k + V_kV_{k-1} + \ldots + V_kV_{k-1} \ldots V_2 \rceil}_{Z_k}$$

and bounded Z_k by k, which led to the extra $1/(\ln n)^\alpha$ factor. However, as k goes to infinity, Z_k approaches Z_∞,

$$Z_k \xrightarrow{\mathcal{L}} Z_\infty \text{ where } \mathbf{E}\{Z_\infty\} = 2, \mathrm{Var}\{Z_\infty\} = 1/2.$$

As Z_k is basically behaving as a constant random variable, we could then make the logarithmic factor disappear. The calculations are not included here.

7 Nearest neighbor algorithm

The search for the nearest neighbor of a node u_i in T may be divided into two independent searches: either the nearest neighbor is in the subtree T_{u_i} rooted at u_i or it is in the tree T minus T_{u_i}. In a random 2-d tree T, the subtree rooted at any node u_i can be considered as a random 2-d tree in which the search for the nearest neighbor of u_i has the same properties as the search of the nearest neighbor of the root of T. This section presents an application of the range search when searching for the nearest neighbor of a randomly chosen node u and we will then see that the nearest neighbor search in the subtree rooted at u may be done in $O(1)$ expected time.

Lemma 6. Suppose that we can execute an algorithm A on a 2-d tree T with n nodes in $O(n^\beta)$ expected time where β is a positive constant strictly smaller than 1. Then A, applied to the subtree of T rooted at a randomly chosen node u, T_u, takes expected time O(1).

PROOF: After u_i is inserted, T_b, the binary search tree corresponding to our 2-d tree, consists of $i+1$ external nodes. Each of these nodes is equally likely to be an ancestor of any of the $n-i$ remaining nodes. Thus, the expected size of the subtree rooted at u_i is

$$\mathbf{E}\{S_i\} = \frac{n-i}{i+1}.$$

According to our hypothesis, if we denote C_i the cost of the algorithm A on the subtree rooted at u_i then

$$\mathbf{E}\{C_i|S_i\} \leq c\ S_i^\beta$$

where $0 \leq \beta < 1$ and $0 < c < \infty$. Due to the concavity of x^β, we can write

$$\mathbf{E}\{C_i\} = \mathbf{E}\{\mathbf{E}\{C_i|S_i\}\}$$

$$\leq c\,\mathbf{E}\{S_i\}^{\beta}$$
$$\leq c\left(\frac{n-i}{i+1}\right)^{\beta}$$
$$\leq c\left(\frac{n}{i}\right)^{\beta}.$$

Given that each node u_i is chosen with equal probability $1/n$, if we denote by C the cost for computing A on T_u, the subtree rooted at a randomly chosen point u, then

$$\begin{aligned}\mathbf{E}\{C\} &= \frac{1}{n}\sum_{i=1}^{n}\mathbf{E}\{C_i\}\\ &\leq \frac{c}{n}\sum_{i=1}^{n}\left(\frac{n}{i}\right)^{\beta}\\ &\leq c\,n^{\beta-1}\left(1+\int_1^n\frac{1}{x^{\beta}}dx\right)\\ &= O(1).\end{aligned}$$

■

We will now present an algorithm that finds the nearest neighbor of a randomly chosen node u_i of a 2-d tree and verifies the above hypothesis. It is not adaptive: it requires knowledge of the size of the rectangle to which the data points belong. Given a 2-d tree T with $n+1$ nodes, this algorithm, NN(T,u_i), finds the nearest neighbor of u_i by computing the range search returning the set Γ_c of all the points in the 2-d tree T lying within a region Q_c defined as the square centered at u_i and of size $\Delta x = \Delta y = \frac{c}{\sqrt{n}}$.

NN(T, u_i)

Note: this algorithm returns the nearest neighbor in the tree T with $n+1$ nodes of a node u_i. Q_c denotes the square centered at u_i and of size $\Delta x = \Delta y = \frac{c}{\sqrt{n}}$.

```
Delete u_i from T                    {we have a new 2-d tree T' with n nodes}
Set c ← 0
Repeat
    c ← c + 1
    Γ_c ← RANGE_SEARCH (T', Q_c)
until Γ_c ≠ ∅                        {Note that c ≤ √n at this point}
Find the nearest neighbor of u_i in Γ_c by brute force
{end NN}
```

Theorem 3. This algorithm finds the nearest neighbor of a random node u in a 2-d tree T with $n+1$ nodes in

$$O(n^{\alpha-1/2}) \approx O(n^{\sqrt{17}/2-2}) \approx O(n^{0.06})$$

expected time.

PROOF: Let us first evaluate the expected cost of deleting the root of a 2-d tree T. If we suppose that the root splits the tree according to its x-coordinate, deleting the root may be done recursively by finding and deleting the x-minimum node in the right subtree of T (as suggested by Bentley in [1]). This node can be found with very high probability with a range search in a rectangle Q such that

$$Q = [x, x + 1/n] \times [0, 1],$$

where x is the x-coordinate of the root of T. If τ_0 denotes the time taken by this procedure, by Theorem 1, we may write

$$\mathbf{E}\{\tau_0\} = O(n^{\alpha}).$$

Moreover, by Lemma 6, we note that the expected cost of the deletion of a random node is

$$\mathbf{E}\{\tau_0\} = O(1).$$

After the deletion, the range search is applied to a square Q_c totally independent of T'. Let N be the first c for which $|\Gamma_c| > 0$ in the algorithm. Let τ_c be the time taken by RANGE_SEARCH(T', Q_c). Then, the total time is

$$\tau = \sum_{i=1}^{N} \tau_c,$$

and,

$$\begin{aligned} \mathbf{E}\{\tau\} &= \sum_{i=1}^{\infty} \mathbf{E}\left\{\tau_c \, I_{[N \geq c]}\right\} \\ &\leq \sum_{i=1}^{\infty} \mathbf{E}\{\tau_c\} \mathbf{P}\{N \geq c\}. \end{aligned}$$

Here we employ an association inequality stating that τ_c and N are negatively dependent. By Theorem 1,

$$\mathbf{E}\{\tau(c)\} \leq \gamma\left(c^2 + n^{\alpha-\frac{1}{2}} \times c + \ln n\right),$$

for some positive constants γ and α. Also,

$$\begin{aligned} \mathbf{P}\{N \geq c\} &\leq \mathbf{P}\{\Gamma_{c-1} = \emptyset\} \\ &\leq \left(1 - \frac{1}{4}\frac{(c-1)^2}{n}\right)^n \\ &\leq e^{-(c-1)^2/4}. \end{aligned}$$

as at least one fourth of Q_c intersects the unit square (Figure 9). Hence, the expected time $\mathbf{E}\{\tau\}$ satisfies

$$\mathbf{E}\{\tau\} \leq \gamma\left(\Lambda_0 + \Lambda_1\, n^{\alpha-\frac{1}{2}} + \Lambda_2 \ln n\right),$$

where

$$\begin{aligned}
\Lambda_0 &= \sum_{c=1}^{\infty} c^2 \times e^{-(c-1)^2/4}, \\
\Lambda_1 &= \sum_{c=1}^{\infty} c \times e^{-(c-1)^2/4}, \\
\Lambda_2 &= \sum_{c=1}^{\infty} e^{-(c-1)^2/4}.
\end{aligned}$$

We notice that $\Lambda_2 \leq \Lambda_1 \leq \Lambda_0$ and that $\sum_{c=1}^{\infty} c^2\, e^{-(c-1)^2/4} \leq \infty$. Thus $\mathbf{E}\{\tau\} = O(n^{\alpha-1/2})$.

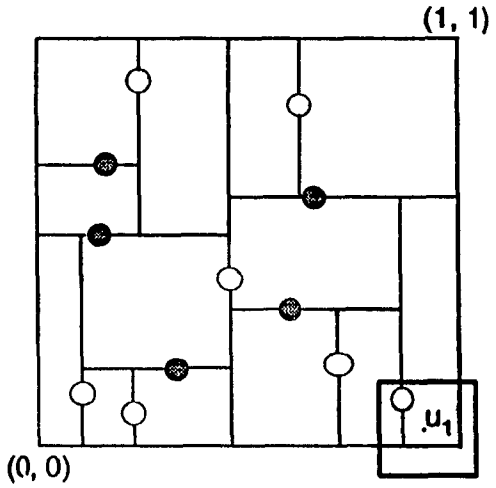

Figure 9: a square Q_c.

The complexity of the last step – finding the nearest neighbor of the root u_1 in Γ_c – is not greater than τ. Therefore the expected complexity of the algorithm is $O(n^{\alpha-1/2})$ and Lemma 6 allows us to conclude the proof of Theorem 3. ∎

In the next section we will see that the expected time for finding the nearest neighbor of a random node in a 2-d tree may be bounded from below for any algorithm that is given a random 2-d tree by

$$\mathbf{E}\{\tau\} = \Omega\left[\frac{n^{\alpha-1/2}}{(\ln n)^{\alpha}}\right].$$

This lower bound is very close to the upper bound proved for our algorithm. We also proved that the nearest neighbor of the root in T can be found in $O(n^{\beta})$ expected time where

$0 < \beta < 1$. Therefore, according to Lemma 6, the nearest neighbor of a randomly chosen node u in T_u, the subtree rooted at u, can be found in $O(1)$ expected time.

The above algorithm may be made adaptive. Furthermore we may increase c geometrically ($c \leftarrow 2c$) for faster execution times.

8 A lower bound for finding the nearest neighbor

In section 7 we described an algorithm for finding the nearest neighbor of a random node in a random 2-d tree whose complexity may be bounded from above by n^β where β is a strictly positive constant. We are very far from Bentley's conjecture that claims the nearest neighbor of a random node, in a random k-d tree with n nodes, could be found in $O(1)$ expected time. In this section we give some evidence that this conjecture may be disproved.

Theorem 4. If we consider a sequence of n independent random points identically and uniformly distributed on the unit square $[0,1]^2$, $U_1, \ldots, U_n$, we can build a 2-d tree by successively inserting the corresponding nodes $u_1, \ldots, u_n$. Let Z be a random point uniformly distributed on the unit square. Then the expected time $\mathbf{E}\{\tau\}$ for finding the nearest neighbor M of Z among $u_{n/2+1}, \ldots, u_n$ may be bounded from below by

$$\mathbf{E}\{\tau\} = \Omega\left[\frac{n^{\alpha-1/2}}{(\ln n)^\alpha}\right].$$

PROOF: Without loss of generality we assume n is odd. Let R be the square centered at Z and whose vertices touch the circle C centered at Z with radius Δ equal to the distance between Z and M. Let Q be the square centered at Z of sides $2/\sqrt{n}$. If we define the "splitting line" of a node u_i as the line segment according to which u_i splits the 2-d tree, then in the smaller 2-d tree T_1 built from $u_1, \ldots, u_{n/2}$, any node whose splitting line intersects R may be closer to Z than M. This means that we cannot conclude M is the nearest neighbor of Z until we have visited all the nodes whose splitting line cuts R.

If Q is included in R, the expected time $\mathbf{E}\{\tau\}$ taken by the nearest neighbor search may be bounded from below by

$$\begin{aligned}\mathbf{E}\{\tau\} &\geq \mathbf{E}\left\{N\, I_{[Q \subset R]}\right\} \\ &\geq \mathbf{E}\left\{N\, I_{[\Delta \geq \sqrt{2/n}]}\right\}\end{aligned}$$

where N is the number of splitting lines of $u_1, \ldots, u_{n/2}$ in T_1 intersecting Q. In this smaller tree T_1, if we denote "final rectangles" the $n/2+1$ smallest rectangles partitioning the unit square (no node lies within a final rectangle) then each of these lines may be seen as bording two final rectangles meeting Q. Thus, N is the number of final rectangles meeting Q

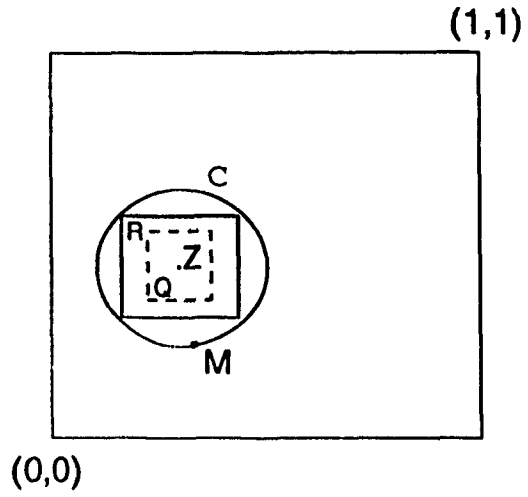

Figure 10: Z and its nearest neighbor, M.

minus 1. In the proof of Theorem 2, we can modify PART 3 and write in the case of our square Q,

$$\mathbf{E}\{N\} \geq \frac{2}{\sqrt{n}}\,\mathbf{E}\left\{\sum_{i=0}^{\frac{n}{2}+1} a_i\right\} - 1.$$

Now, we have

$$\mathbf{E}\{a_{2k}\} \geq \mathbf{E}\left\{V_1 \times \ldots \times V_k \; I_{\{nV_1 \times \ldots \times V_k \times W_1 \times \ldots \times W_{k-1} = 2k\}}\right\},$$

and the calculations are easily adapted and return the lower bound

$$\mathbf{E}\{N\} \geq \Omega\left[\frac{2}{\sqrt{n}}\left(\frac{n}{\ln n}\right)^{\alpha}\right].$$

Since the nearest neighbor of Z is sought among a data set independent of the lines cutting Q, we have

$$\mathbf{E}\{\tau\} \geq \mathbf{E}\{N\}\;\; \mathbf{P}\left\{\Delta \geq \sqrt{2/n}\right\}.$$

From Theorem 2 we recall that

$$\mathbf{E}\{N\} \geq \frac{C}{\sqrt{n}}\left(\frac{n^{\alpha-1/2}}{(\ln n)^{\alpha}}\right),$$

where C is a positive constant. We also have

$$\mathbf{P}\left\{\Delta \geq \sqrt{2/n}\right\} \geq \left(1 - \frac{\pi}{4}\frac{2}{n}\right)^{n-2} \overset{n\to\infty}{\to} e^{-\pi/2}.$$

Hence,

$$\mathbf{E}\{\tau\} = \Omega\left[\frac{n^{\alpha-1/2}}{(\ln n)^{\alpha}}\right].$$

■

This result contradicts Bentley's conjecture that the nearest neighbor of a random node in a random k-d tree could be found in constant expected time.

9 Conclusion

Using a geometrical and probabilistic approach, we have analyzed the complexity of range searching and nearest neighbor searching in random 2-d trees. The general formula of Theorem 1 applies to many kinds of operations besides ordinary range search. In particular, in the case of partial match query, our theorem returns the result Flajolet and Puech [7] proved with the help of generating functions. It has also been shown in the text how the formula applies to deletions of nodes and to nearest neighbor searching. We would like to extend these results to non-uniform distributions.

References

[1] J. L. Bentley. *Multidimensional binary search trees used for associative searching* Communications of the ACM, Vol. 18, No. 9, 1975, pp. 509–517.

[2] J. L. Bentley. *Multidimensional binary search trees in database applications.* IEEE Transactions On Software Engineering, Vol. SE–5, No. 4, 1979, pp. 333–340

[3] J. L. Bentley and J. H. Friedman. *Algorithms and data structures for range queries.* Proceedings of Computer Science and Statistics: 11th Annual Symposium on the Interface, 1978, pp. 297–307.

[4] L. Devroye. *A note on the height of binary search trees.* Journal of the ACM, Vol. 33, 1986, pp. 489–498.

[5] R. A. Finkel and J. L. Bentley. *Quad trees a data structure for retrieval on composite keys.* Acta Informatica, Vol. 4, 1974, pp. 1–9.

[6] P. Flajolet and C. Puech. *Tree structures for partial match retrieval.* Proceedings of the 24th Annual IEEE Symposium on the Foundations of Computer Science. IEEE Press, New York, 1983, pp. 282–288.

[7] P. Flajolet and C. Puech. *Partial match retrieval of multidimensional data.* Journal of the ACM, Vol. 33, No. 2, 1986, pp. 371–407.

[8] P. Flajolet, G. H. Gonnet, C. Puech and M. Robson. *The analysis of multidimensional searching in quad-trees.* Proceedings of the Second Annual ACM-SIAM Symposium on Discrete Algorithms. ACM, New York and SIAM, Philadelphia, 1990, pp. 100–109.

[9] J. H. Friedman, J. L. Bentley and R. A. Finkel. *An algorithm for finding best matches in logarithmic time.* ACM Transactions Mathematical Software, Vol. 3, No. 3, 1977, pp. 209–226.

[10] G. H. Gonnet, R. Baeza-Yates. *Handbook of Algorithms and Data Structures: in Pascal and C. /2nd ed.* Addison-Wesley, Workingham, England, 1991.

[11] D. T. Lee and C. K. Wong. *Worst-case analysis for region and partial region searches in multidimensional binary search trees and balanced quad trees.* Acta Informatica, Vol 9, 1977, pp. 23–29.

[12] M. H. Overmars. *The Design of Dynamic Data Structures.* Berlin, New York, Springer-Verlag, 1983.

[13] J. S. Vitter and P. Flajolet. *Average-case analysis of algorithms and data structures.* Handbook of Theoretical Computer Science, Vol A: Algorithms and Complexity, J. van Leeuwen, MIT Press, Amsterdam, 1990, pp. 431–524.

[14] D. Wood. *Data Structures, Algorithms, and Performance.* Addison-Wesley, Reading, MA, 1993.