Partial Match Queries in Relaxed K-dt trees*

Amalia Duch† Gustavo Lau‡

Abstract

The study of partial match queries on random hierarchical multidimensional data structures dates back to Ph. Flajolet and C. Puech's 1986 seminal paper on partial match retrieval. It was not until recently that fixed (as opposed to random) partial match queries were studied for random relaxed K-d trees, random standard K-d trees, and random 2-dimensional quad trees. Based on those results it seemed natural to classify the general form of the cost of fixed partial match queries into two families: that of either random hierarchical structures or perfectly balanced structures, as conjectured by Duch, Lau and Martínez (On the Cost of Fixed Partial Queries in K-d trees *Algorithmica*, 75(4):684–723, 2016). Here we show that the conjecture just mentioned does not hold by introducing relaxed K-dt trees and providing the average-case analysis for random partial match queries as well as some advances on the average-case analysis for fixed partial match queries on them. In fact this cost –for fixed partial match queries– does not follow the conjectured forms.

1 Introduction.

It is very common nowadays to ask our cell phones for the closest gas station or for all the restaurants in a given bounded region, and so on and so forth. On performing such consultations we are making associative queries. One of the most basic types of those queries is the partial match where we are given some (but not all) of the characteristics of the desired target and we want to know if such an element is present in our available data.

Formally, the study of partial match queries, more specifically of their average cost on multidimensional data structures, goes back to the paper "Partial Match Retrieval of Multidimensional Data" by Ph. Flajolet and C. Puech [16]. In that work the authors focused on the analysis of random partial match queries over different random multidimensional data structures. Subsequently, several papers studied random partial match queries in a wide variety of multidimensional data structures [4, 5, 6, 8, 10, 11, 14, 18, 19, 20]. More recently, the analysis of fixed partial match queries in random K-d trees of any dimension and random quad trees of dimension 2 was introduced in [3, 7, 9, 10]. Although the study of fixed partial match queries remains to be extended to other multidimensional data structures –and dimensions greater than 2 in the case of quad trees– if the conjecture in [10] were true, the expected cost $P_{n,\mathbf{q}}$ of a fixed partial match query $\mathbf{q} = (q_1, \ldots, q_{K-1})$ where each q_i is either specified ($q_i \in (0,1)$) or not specified ($q_i = *$), $0 \le j < K$, on all random trees of size n (not perfectly balanced trees) would be of the form:

(1.1)
$$P_{n,\mathbf{q}} = \nu_{(\mathbf{q})} \cdot \left(\prod_{i:q_i \text{ is specified}} q_i(1-q_i) \right)^{\alpha/2} \cdot n^{\alpha} + \text{l.o.t.}^{1},$$

where $\nu_{(\mathbf{q})}$ is a constant that is dependent on the pattern of specified and unspecified coordinates in $\mathbf{q}$ and α is the same exponent as in random partial match queries and depends on the family of data structures under consideration as well as on the ratio between K and the number of specified coordinates in the query.

In this work, by means of the same techniques used in [10], we show that this conjecture is false. To do so, we introduce relaxed K-dt trees –a variant of random relaxed K-d trees where every subtree of size $2t+1$ is locally rebalanced in such a way that the resulting K-d tree has two subtrees of size t (t is known as the balance factor of the tree)– and we provide advances on the study of random and fixed partial match queries on them. As we already mentioned, relaxed K-dt trees reveal to be a multidimensional data structure whose expected cost of fixed partial match queries neither fit the form given in the conjecture (unexpectedly) nor that of perfectly balanced trees (as expected).

This paper is organized as follows. In Section 2 we give the basic definitions involved in the analysis of relaxed K-dt trees. We introduce K-dt trees in Section 3. In Section 4 we analyze random partial match

*Supported by by funds from the Spanish Ministry for Economy and Competitiveness (MINECO) and the European Union (FEDER funds) under grant COMMAS (ref. TIN2013-46181-C2-1-R) and AGAUR grant SGR 2014:1034 (ALBCOM).

†Technical University of Catalonia. Barcelona Tech. Computer Science Department. Barcelona, Catalonia, Spain.

‡Technical University of Catalonia. Barcelona Tech. Computer Science Department. Barcelona, Catalonia, Spain.

[1] Lower order terms hereinafter.

queries (interesting on their own and a prerequisite for the analysis of fixed partial match queries) and in Section 5 we provide advances in the analysis of fixed partial match queries. By making $t = 1$, in Sections 6 and 7, we are able to provide the exact cost of random partial match queries, and for fixed partial match queries we show that their average cost does not follow the general form given in Equation 1.1, showing therefore that K-dt trees are a counterexample to the conjecture. We finish giving some conclusions and future lines of research in Section 8.

2 Preliminaries.

Let us restate some basic definitions before starting with our analysis.

Let F be a collection of n multidimensional records, each one represented by a K-dimensional key $\mathbf{x} = (x_0, \ldots, x_{K-1})$, with coordinate x_i drawn from a totally ordered domain $\mathcal{D}_i$. For convenience, here we will also assume that, for all $0 \leq i < K$, $\mathcal{D}_i = [0, 1]$. It is generally assumed, without loss of generality, that no two keys in the collection have the same coordinates in any of the dimensions.

A K-d tree T of size n –initially proposed by Bentley [1]– is a binary search tree that is either empty ($n = 0$) or a pointer to a root node that stores a pair $\langle \mathbf{x}, d \rangle$ –where $\mathbf{x}$ is a K-dimensional key and d a discriminant with $0 \leq d < K$– and pointers to two subtrees L and R, both K-d trees, such that:

- for any $\mathbf{x}'$ in L we have that $x'_d < x_d$ and
- for any $\mathbf{x}''$ in R we have that $x''_d > x_d$.

The original or *standard* K-d trees defined by Bentley [1] have, for each node at level ℓ the discriminant $d = \ell \mod K$. A K-d tree T is a *relaxed* K-d tree [8] if, at any node, the discriminant is assigned uniformly at random among the K possibilities.

There are several equivalent definitions of random relaxed K-d trees but for the purposes of this paper it is enough to say that a K-d tree is *random* if its keys are sampled independently at random (coordinate by coordinate) from a continuous distribution.

A *partial match* query $\mathbf{q}$ is given by $\mathbf{q} = (q_0, \ldots, q_{K-1})$ with $q_i \in \mathcal{D}_i \cup \{*\} = [0, 1] \cup \{*\}$. The coordinates $q_i \neq *$ are called *specified*, otherwise they are called *unspecified*. In what follows we assume that the number s of specified coordinates satisfies $0 < s < K$. Specified coordinates can be *extreme* (if $q_i = 0$ or $q_i = 1$), otherwise they are *regular*. This distinction is required for the formal proof of Proposition 5.1, see [10] for more details. Without it all the results presented here become much simpler.

A partial match (PM, hereinafter) search in any kind of K-d tree consists of retrieving all the records in the K-d tree that match the query $\mathbf{q}$, i.e., the records $\mathbf{x}$ such that $x_i = q_i$ whenever $q_i \neq *$.

To perform a PM search with query $\mathbf{q}$, the K-d tree is recursively explored. First, we check whether the root matches $\mathbf{q}$ or not, and report it when it is the case. Then, if the root discriminates with respect to an unspecified coordinate, we make recursive calls in both subtrees. Otherwise –if the root is $\langle \mathbf{x}, d \rangle$– we continue recursively in the appropriate subtree depending on whether $q_d \leq x_d$ or $x_d < q_d$.

If the specified coordinates of $\mathbf{q}$ are sampled independently from the same continuous distribution as the coordinates of the keys in F we say that the partial match query is *random*, otherwise it is considered *fixed*. In fact, the difference between random and fixed PM queries is that the latter is global while the former can be seen as if –at every recursive step of the retrieval algorithm– a new query (with the same pattern of specified/unspecified coordinates) were generated randomly.

Let us observe that, under the random models defined for the trees and the PM queries (if the case), the probability that a key in the tree matches a PM query is zero. Therefore it is natural to wonder whether it is worth analyzing these queries. Indeed, beyond their intrinsic theoretical interest, PM queries are fundamental for the analysis of more complex associative queries such as range queries (see for instance [12]).

A fixed PM query can be seen in terms of the ranks of its coordinates. The rank vector of a PM query $\mathbf{q}$ is the vector $\mathbf{r}(\mathbf{q}) = (r_0, \ldots, r_{K-1})$ where r_i is the number of records $\mathbf{x}$ in the K-dt tree such that $x_i \leq q_i$ if $q_i \neq *$, otherwise $r_i = *$. Any pair of queries $\mathbf{q}$ and $\mathbf{q}'$ with equal rank vectors $\mathbf{r}(\mathbf{q}) = \mathbf{r}(\mathbf{q}')$ have the same partial match cost. Therefore, the expected cost of a PM query as a function of the query $\mathbf{q}$, $P_{n,\mathbf{q}}$, can be deduced from its expected cost as a function of the rank $\mathbf{r}$, $P_{n,\mathbf{r}}$, whenever the coordinates of $\mathbf{q}$ are uniformly and independently generated from $(0, 1)$. It is worth mentioning that uniformity is required only to transform a query $\mathbf{q}$ into rank $\mathbf{r}$ and vice versa but is not required at all to analyze $P_{n,\mathbf{r}}$. Consequently, in our analysis of the expected cost of fixed PM queries in random relaxed K-dt trees in Section 5 we are going to proceed, as done in [10], by analyzing $P_{n,\mathbf{r}}$.

The relation between random and fixed PM queries is as follows. Given a pattern of specified/unspecified coordinates, the expected cost (measured as the number of visited nodes) of a random PM query as a function of the rank $\mathbf{r}$ is the average of the expected costs of all the possible fixed PM queries that fit the same pattern.

3 K-dt trees.

K-dt trees were introduced in [6] where the local rebalance methods applied to the fringe of binary search trees [21, 24] were adapted to K-d trees. That work focused on standard K-dt trees. In this work we introduce relaxed K-dt trees.

A K-dt tree with t a non-negative integer, is a K-d tree where every subtree T of size $2t+1$ is rebalanced in such a way that the size of its two subtrees L and R are both equal to t, i.e., the median with respect to coordinate d (the discriminant at the root, $0 \leq d < K$) of all the keys contained in T is the key in the root. As for K-d trees, K-dt trees are *relaxed* under the same conditions.

Algorithm 1 shows the insertion algorithm of a key $\mathbf{x}$ into a K-dt tree T. As can be seen, every time that a subtree reaches a size of $2t+1$ nodes, it is rebalanced in such a way that its left and right subtrees are both of size t, if that is not the case yet. For brevity we do not show the rebalance function.

Algorithm 1 Insertion algorithm of a key $\mathbf{x}$ with discriminant d into the relaxed K-dt tree T. It returns T after the insertion.

```
function INSERT(x, d, K, t, T)
    ▷ T.size is the number of nodes of T
    ▷ T.key is the key associated to the root of T
    ▷ T.discr is the discriminant of the root of T
    ▷ T.left is the left subtree of T
    ▷ T.right is the right subtree of T
    if T.size = 0 then
        T = new K-dt node(x, d)
    else
        T.size = T.size + 1
        i = T.discr
        if x[i] > T.key[i] then
            T.right = INSERT(x, d, K, t, T.right)
        else
            T.left = INSERT(x, d, K, t, T.left)
    if T.size = 2t + 1 and T.right.size != t then
        T = rebalance(T,t)
    return T
```

Let us illustrate how to build a K-dt tree with an example where we assume that $K = 2$, that $\mathcal{D}_1 = \mathcal{D}_2 = [0, 10]$. Figure 1 shows a relaxed 2-d1 tree based on Table 1, the rebalances performed during the insertions –to guarantee that each subtree of size greater than 2 has at least 1 element in each of its subtrees– and the partition that it induces.

The example illustrates that when a subtree is rebalanced the node that moves to the root will discrim-

Label	X coordinate	Y coordinate
A	5	2
B	3	3
C	2	8
D	1	7
E	9	5
F	4	9
G	8	1
H	6	6

Table 1: Sample data

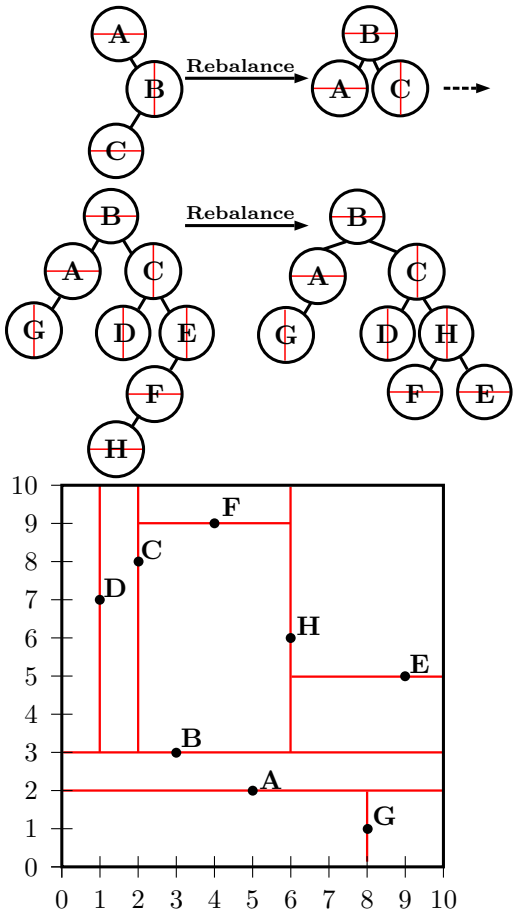

Figure 1: A relaxed 2-d1 tree produced inserting the data in Table 1 in alphabetical order, the rebalances performed during the insertions and the partition that the final 2-d1 tree induces. The nodes that discriminate by coordinate x have a vertical red line and the ones that discriminate by coordinate y have a horizontal red line.

inate by the coordinate that the root used to discriminate before the rebalance and the subtrees will be recreated from scratch, potentially changing the discriminating coordinates of some nodes.

As we already mentioned, a *random* K-dt tree is defined in exactly the same way as a random K-d tree.

Let us define $w_{t,n,j}$ as the probability that in a random K-dt tree of size n the left subtree has size j, $0 \leq j \leq n-1$. Since the n K-dimensional points of the random tree T are a random permutation, when n is less than $2t+1$ the probability that the left subtree has size j is just $1/n$ as in random binary search trees. If on the contrary, n is greater or equal than $2t+1$, the subtree had been rebalanced when it had size exactly $2t+1$, resulting in $\binom{n}{2t+1}$ possibilities. From these $2t+1$ records, one is the root, exactly t are in the left subtree, and the remaining t are in the right one, therefore, the number of possibilities for this situation are $\binom{j}{t}\binom{n-1-j}{t}$, if the left subtree of T is of size j, $0 \leq j \leq n-1$. Then, just as in standard K-dt trees (see [6]), we have:

$$w_{t,n,j} = \begin{cases} \dfrac{1}{n} & \text{if } n < 2t+1 \\ \dfrac{\binom{j}{t}\binom{n-1-j}{t}}{\binom{n}{2t+1}} & \text{if } n \geq 2t+1 \end{cases} \tag{3.2}$$

with the symmetry $w_{t,n,j} = w_{t,n,n-1-j}$. This formula, which is similar to a hypergeometric distribution, also appears in the analysis of the Quicksort variant where the pivot is chosen as the median of a sample of size $2t+1$ elements [23].

Note that the probabilities when there is no rebalancing, the case $t = 0$, coincide with the ones of the case $n < 2t+1$ and these both are $1/n$, which is expected given that the relaxed K-d0 trees are in fact relaxed K-d trees.

4 Random Partial Match.

In this section we analyze the average cost of a random partial match search in a random K-dt tree T.

Let $\overline{P}_n$ be the expected cost measured, as usual, as the number of visited nodes of a random PM search in a random relaxed K-dt tree of size n with a random query $\mathbf{Q}$ in which s coordinates are specified and the remaining $K-s$ coordinates are left unspecified. Of the s specified coordinates exactly s_R coordinates are independently drawn at random from the same continuous distribution(s) from which data coordinates are drawn, and exactly $s_0 = s - s_R$ coordinates are extreme (we can assume, w.l.o.g., that they are 0).

To set a recurrence for $\overline{P}_n$ we have to consider, first, that the base case is clearly $\overline{P}_0 = 0$. Afterwards, if $n > 0$, we consider all the possible sizes j, $0 \leq j < n$, of the left subtree and we use the probability $w_{t,n,j}$ defined above. The right subtree has size $n-1-j$. Then, depending on the discriminant, we have the following three cases:

- The root does not discriminate by a specified coordinate so the search has to continue in both subtrees with probability $(K-s)/K$.
- The root discriminates by a specified coordinate that is extreme then, with probability s_0/K, the search has to continue only on the right subtree.
- The root discriminates by a specified coordinate that is not extreme and the search has to continue in one of the two subtrees with probability $(s-s_0)/K$. In this case, the probability of continuing the search on the left subtree is $(j+1)/(n+1)$ and the probability of continuing on the right one is $(n-j)/(n+1)$.

Therefore when $n > 0$:

$$\overline{P}_n = 1 + \sum_{j=0}^{n-1} w_{t,n,j} \left\{ \frac{K-s}{K}(\overline{P}_j + \overline{P}_{n-1-j}) + \frac{s_0}{K}\overline{P}_{n-1-j} + \frac{s-s_0}{K}\left(\frac{j+1}{n+1}\overline{P}_j + \frac{n-j}{n+1}\overline{P}_{n-1-j}\right)\right\}.$$

Using the symmetry $w_{t,n,j} = w_{t,n,n-1-j}$, rearranging and defining $\rho := s/K$ and $\rho_0 := s_0/K$, we can write the recurrence for $\overline{P}_n$ when $n > 0$ as:

$$\overline{P}_n = 1 + (2 - 2\rho + \rho_0)\sum_{j=0}^{n-1} w_{t,n,j}\overline{P}_j + (2\rho - 2\rho_0)\sum_{j=0}^{n-1}\frac{j+1}{n+1}w_{t,n,j}\overline{P}_j. \tag{4.3}$$

Having the recurrence, we can state the following theorem.

THEOREM 4.1. *Let $\overline{P}_n$ be the expected cost of a random PM search in a random relaxed K-dt tree of size n. Then*

$$\overline{P}_n = \Theta\left(n^{\alpha}\right),$$

where the exponent $\alpha = \alpha(t, \rho, \rho_0)$ is the unique root in $(0,1)$ of the equation:

$$\frac{\rho - \rho_0}{(\alpha+t+2)^{\overline{t+1}}} + \frac{1 - \rho + \rho_0/2}{(\alpha+t+1)^{\overline{t+1}}} = \frac{1}{(t+2)^{\overline{t+1}}}, \tag{4.4}$$

and $x^{\overline{k}} = x(x+1)\ldots(x+k-1)$ is the rising factorial power (see [17]).

Proof. Sketch. Although the exponent α is obtainable in several ways [6, 16, 22] we are going to proceed as in [16] since, we can use the obtained differential equations to give more information on the cost than the one provided by other methods.

The method consists of transforming Recurrence 4.3 into a differential equation for the generating function $P(z) = \sum_{n\geq 0} \overline{P}_n z^n$, which gives

$$(4.5)\quad \begin{aligned}(1-z)^{t+2}P^{(2t+2)}(z) &\\ + 2(t+1)(1-z)^{t+2}P^{(2t+1)}(z) &\\ - (t+1)^{\overline{t+1}}(2-\rho_0)z(1-z)P^{(t+1)}(z) &\\ - (t+1)^{\overline{t+1}}(t+1)(4-2\rho-(2-\rho_0)z)P^{(t)}(z) &\\ = \frac{(2t+2)!}{(1-z)^{t+1}}.&\end{aligned}$$

Taking the homogeneous equation of (4.5), assume that around the singularity $z = 1$, $P(z)$ is of the form $(1-z)^{\varphi}$ where φ is the smallest root of the corresponding indicial equation. Finally, by a Transfer Lemma of Flajolet and Odlyzko [15], with $\alpha = -\varphi - 1$, the theorem follows.

If $s_0 = 0$ then (4.4) simplifies to:

$$(4.6)\quad \frac{\rho}{(\alpha+t+2)^{\overline{t+1}}} + \frac{1-\rho}{(\alpha+t+1)^{\overline{t+1}}} = \frac{1}{(t+2)^{\overline{t+1}}},$$

The equivalent formula for standard K-dt trees [6] can be written as:

$$(4.7)\quad \left(\frac{1}{(\alpha+t+2)^{\overline{t+1}}}\right)^{\rho}\left(\frac{1}{(\alpha+t+1)^{\overline{t+1}}}\right)^{1-\rho} = \frac{1}{(t+2)^{\overline{t+1}}}$$

Note the similarity: (4.6) has a weighted arithmetic average on the left hand side; (4.7) has a weighted geometric average.

The following theorem gives the general form of the generating function that describes the expected cost of random PM queries for general t.

THEOREM 4.2. *Let $P(z) = \sum_{n\geq 0} \overline{P}_n z^n$ be the generating function for the expected cost $\overline{P}_n$ of a random PM search in a random relaxed K-dt tree of size n, then $P(z) = \sum_{j=0}^{2t+1} \beta_j G_j(1-z)(1-z)^{\varphi} + \frac{\beta_{2t+2}}{(\rho-1)(1-z)}$ where $\varphi = -(\alpha+1)$, $0 \leq j \leq 2t+1$, β_j are constants that depend on α and the functions $G_j(1-z)$, constitute a fundamental set of solutions of the generalized hypergeometric differential equation:*

$$(4.8)\quad \begin{aligned}(1-z)z^{2t+1}G^{(2t+2)}(z) &\\ + \sum_{i=1}^{2t+1}\Big((A_i z + B_i)z^{i-1}G^{(i)}(z)\Big) &\\ + A_0 G(z) = 0&\end{aligned}$$

with

$$\begin{aligned}A_i = -\binom{2t+2}{i}\varphi^{\underline{2t+1-i}}(\varphi+1) &\\ - \tau\varphi^{\underline{t-i}}(2-\rho_0)(\varphi+1),&\end{aligned}$$

for $0 \leq i \leq 2t+1$ and

$$\begin{aligned}B_i = \binom{2t+2}{i}\varphi^{\underline{2t+2-i}} &\\ + \tau\varphi^{\underline{t-i}}\big((2-\rho_0)\varphi - (4-2\rho)(t-i) - 2\rho - 2 - \rho_0\big),&\end{aligned}$$

$$\tau = (-1)^t(t+1)^{\overline{t+1}}\binom{t+1}{i},$$

for $1 \leq i \leq 2t+1$, and $x^{\underline{k}} = x(x-1)\ldots(x-k+1)$ is the falling factorial power [17].

Proof. Sketch. To derive the form of $P(z)$ it suffices to observe first that $P(z) = \frac{1}{(\rho-1)(1-z)}$ is a particular solution of the inhomogeneous equation and that a solution of the homogeneous equation is of the form $(1-z)^{\varphi}G(1-z)$, which, after long calculations, gives the generalized hypergeometric differential equation for G.

5 Fixed Partial Match.

We analyze fixed partial match queries for K-dt trees generalizing the method used in [10] for K-d trees, to which we refer the interested reader for more details.

Indeed, following the same steps of [10], replacing $1/n$ by $w_{t,n,j}$, it is first required to prove that the limit

$$f(\mathbf{z}) = f(z_0, \ldots, z_{s_R-1}) = \lim_{n\to\infty} \frac{P_{n,\mathbf{r}}}{n^{\gamma}}$$

exists and is the unique solution of an integral equation; we state that in Proposition 5.1 (a generalization of Proposition 1 of [10]). Then we provide a method to find a differential equation equivalent to that integral equation. In Section 7 we study the differential equation for the case $t = 1$.

PROPOSITION 5.1. *Let $P_{n,\mathbf{r}}$ be the expected cost (measured as the number of visited nodes) of a fixed PM search in a random relaxed K-dt tree of size n with a*

fixed query $\mathbf{q}$ *with rank vector* $\mathbf{r} = (r_0, \ldots, r_{K-1})$ *such that* $z_i = \lim_{n\to\infty} r_i/n \in (0,1)$ *for all* i, $0 \le i < s_R$, $r_i = o(n)$ *or* $r_i = n - o(n)$ *for all* i, $s_R \le i < s$, *and* $r_i = *$ *for all* i, $s \le i < K$ *and* $0 < s_R \le s < K$ [2]. *The* $\lim_{n\to\infty} \frac{P_{n,\mathbf{r}}}{n^\gamma}$ *exists for* $\gamma = \alpha(t,\rho,\rho_0)$ *(as given in Theorem 4.1) and*

$$f(\mathbf{z}) = \lim_{n\to\infty} \frac{P_{n,\mathbf{r}}}{n^\gamma}$$

is the unique solution of the integral equation

(5.9)
$$f(\mathbf{z}) = \lambda_t \sum_{i=0}^{s_R-1} \left\{ z_i^{\gamma+t+1} \int_{z_i}^1 f_{-z_i,z}(\mathbf{z})(z-z_i)^t \frac{dz}{z^{\gamma+2t+2}} + (1-z_i)^{\gamma+t+1} \int_0^{z_i} f_{-z_i,z}(\mathbf{z})(z_i-z)^t \frac{dz}{(1-z)^{\gamma+2t+2}} \right\},$$

where $\lambda_t = \frac{(\alpha+t+2)^{\overline{t+1}}}{t!2s_R}$, $f_{-z_i,z}(\mathbf{z})$ *denotes that the* i*-th component* z_i *of parameter* $\mathbf{z}$ *is replaced by* z, *and the function* f *is subject to the following constraints:*

(a) The function f *is symmetric with respect to any permutation of its arguments.*

(b) For any i, $0 \le i < s_R$, *and* $z_i \in (0,1)$,

$$f(\mathbf{z}) = f_{-z_i,1-z_i}(\mathbf{z}).$$

(c) For any i, $0 \le i < s_R$,

$$\lim_{z_i\to 0} f(\mathbf{z}) = \lim_{z_i\to 1} f(\mathbf{z}) = 0.$$

(d) $\int_0^1 \cdots \int_0^1 f(\mathbf{z})\, dz_0 \cdots dz_{s_R-1} = \beta(t,\rho,\rho_0)$,

where $\beta(t,\rho,\rho_0)$ *is the coefficient of the dominant term of* $\overline{P}_n$.

Proof. Sketch. The main difference between the analysis of K-d trees and K-dt trees is the probability that the left subtree has a particular size. In the former case it is simply $1/n$ where n is the size of the tree. As mentioned before, due to the potential local rebalances, in the latter case we need to replace $1/n$ by $w_{t,n,j}$ and approximate the binomial coefficients by polynomials to get:

$$w_{t,n,j} \sim \frac{1}{2n} \frac{(t+2)^{\overline{t+1}}}{t!} \left(\frac{j}{n}\right)^t \left(\frac{n-1-j}{n}\right)^t.$$

[2] The rank vector corresponds to a partial match query with at least one regular specified coordinate and at least one unspecified coordinate.

In order to solve the integral equation (5.9) given in Proposition 5.1, together with constraints (a)–(d) we follow steps analogous to the ones presented in [10].

To transform the integral equation (5.9) into an equivalent partial differential equation (PDE), we start assuming that the solution to the integral equation is a function in separable variables, that is

$$f(\mathbf{z}) = \phi_0(z_0) \cdot \phi_1(z_1) \cdots \phi_{s_R-1}(z_{s_R-1}).$$

Because of the symmetry of f (constraint (a)), it follows that we can safely assume $\phi_0 = \phi_1 = \cdots = \phi_{s_R-1}$. We simply call these functions ϕ. Furthermore, because of constraint (b), we must have $\phi(z) = \phi(1-z)$ for any $z \in (0,1)$. We must also have $\lim_{z\to 0} \phi(z) = 0$ to satisfy constraint (c).

If, for any function $f : \mathbb{R}^{s_R} \to \mathbb{R}$, we define the operator

(5.10)
$$LR_{t,i}[f](\mathbf{z}) := z_i^{\alpha+t+1} \int_{z_i}^1 f_{-z_i,z}(\mathbf{z})(z-z_i)^t \frac{dz}{z^{\alpha+2t+2}} + (1-z_i)^{\alpha+t+1} \int_0^{z_i} f_{-z_i,z}(\mathbf{z})(z_i-z)^t \frac{dz}{(1-z)^{\alpha+2t+2}}$$

and, for brevity, we denote $\phi_i := \phi(z_i)$ then we can rewrite the integral equation as

(5.11)
$$\phi_0 \cdot \phi_1 \cdots \phi_{s_R-1} = \lambda_t \sum_{i=0}^{s_R-1} \phi_0 \cdots \phi_{i-1} \cdot \phi_{i+1} \cdots \phi_{s_R-1} LR_{t,i}[\phi_i].$$

If for all i, $0 \le i < s_R$, $\phi_i = s_R \lambda_t LR_{t,i}[\phi_i]$ then $f = \phi_0 \cdots \phi_{s_R-1}$ would be a solution of equation (5.9). Given that for all i, $\phi(z_i) = \phi_i$ this last equation can be rewritten as:

(5.12)
$$\phi(z) = s_R \lambda_t \left(z^{\alpha+t+1} \int_z^1 \phi(u)(u-z)^t \frac{du}{u^{\alpha+2t+2}} + (1-z)^{\alpha+t+1} \int_0^z \phi(u)(z-u)^t \frac{du}{(1-u)^{\alpha+2t+2}} \right).$$

To convert the integral equation (5.12) into a differential equation we start by defining for any integers i, j and any real function $g(z)$ the operators:

$$L_{i,j}[g](z) := z^{\alpha+i} \int_z^1 g(u) \frac{du}{u^{\alpha+j}}$$

and

$$\Lambda_{i,j}[g](z) := L_{i,j}[g](z) + L_{i,j}[g](1-z).$$

Expanding we can rewrite the integral equation (5.12) as:

(5.13)
$$\phi(z) = s_R \lambda_t \sum_{i=0}^{t} \binom{t}{i} (-1)^i \Lambda_{t+i+1,t+i+2}[\phi](z).$$

PROPOSITION 5.2. *Defining for any integers i, j and function $g(z)$ the operator*

$$\Psi_{i,j}[g](z) := z(1-z)g''(z) + (\alpha+i)(2z-1)g'(z) - (\alpha+j)(\alpha+2i-j+1)g(z), \tag{5.14}$$

then the composite operator

$$\Psi_{t,0} \circ \Psi_{t,1} \circ \cdots \circ \Psi_{t,t-1} \circ \Psi_{t,t}$$

converts the integral equation (5.13) *into a linear differential equation of order* $2t+2$.

Proof. For fixed t, the operators $\Psi_{t,i}$ and $\Psi_{t,j}$ commute, so we can apply the operators in any order to each term. By mathematical induction we prove that, for any $0 \le i \le t$,

$$\Psi_{t,t} \circ \Psi_{t,t-1} \circ \cdots \circ \Psi_{t,t-i+1} \circ \Psi_{t,t-i} \circ \Lambda_{t+i+1,t+i+2}[\phi](z)$$

has no integrals and is a linear function of $\phi(z)$ and its derivatives.

6 Random Partial Match, $t = 1$.

When $t = 1$ the random case gives rise to a fourth order differential equation and a fourth degree indicial equation, and therefore seems to be the unique case (besides $t = 0$) where it is possible to provide a general complete characterization.

THEOREM 6.1. *The expected cost, $\overline{P}_n$ of a PM search in a random relaxed K-d1 tree of size n is:*

$$\overline{P}_n \sim \beta(1,\rho,\rho_0) \cdot n^{\alpha(1,\rho,\rho_0)},$$

where $\beta(1,\rho,\rho_0)$ is a positive constant and $\alpha(1,\rho,\rho_0)$ is the unique root in $(0,1)$ of the equation:

$$\alpha^3 + 9\alpha^2 + 2(3\rho_0+7)\alpha + 24(\rho-1) = 0. \tag{6.15}$$

Proof. Sketch. The procedure is to specialize the generating function of Theorem 4.2 to the case $t = 1$. In order to solve the resulting generalized hypergeometric differential equation it is required to find (developing its canonical form) the parameters:

$$\begin{aligned}
a_1 &= \varphi, \\
a_2 &= \varphi+1, \\
a_3 &= \varphi+2, \\
a_4 &= \varphi-5, \\
b_1 &= \varphi+1, \\
b_2 &= \frac{1}{2}\left(3\varphi + \sqrt{40+12\varphi-3\varphi^2} - 4\right), \\
b_3 &= \frac{1}{2}\left(3\varphi - \sqrt{40+12\varphi-3\varphi^2} - 4\right).
\end{aligned}$$

Finally, with c constant and F the hypergeometric function,

$$\beta(1,\rho,\rho_0) = \frac{c}{{}_3F_2\left(\begin{smallmatrix} a_1,a_3,a_4 \\ b_2,b_3 \end{smallmatrix} \middle| 1\right)}.$$

7 Fixed Partial Match, $t = 1$.

In this section we present our main result: we show that in the case $t = 1$, there is no solution of the form $\nu(z(1-z))^\lambda$ for the differential equation equivalent to Equation (5.9). This is a counterexample to the conjecture presented in [10] for the case of relaxed K-dt trees. In the case $t = 1$ Equation (5.12) becomes:

$$\begin{aligned}
\phi(z) = s_R\lambda_1 \Bigg(& z^{\alpha+2} \int_z^1 \phi(u)(u-z)\frac{du}{u^{\alpha+4}} \\
& + (1-z)^{\alpha+2} \int_0^z \phi(u)(z-u)\frac{du}{(1-u)^{\alpha+4}} \Bigg).
\end{aligned} \tag{7.16}$$

PROPOSITION 7.1. *The equation* (7.16) *has no solution of the form $\nu(z(1-z))^\lambda$ where ν and λ are constants.*

Proof. Sketch. Applying $\Psi_{1,0}$ and $\Psi_{1,1}$ to (7.16), makes all the integrals disappear. After expanding the left hand side, replacing $s_R\lambda_1$ by $(\alpha+3)(\alpha+4)/2$ and simplifying we get the following differential equation:

$$\begin{aligned}
& 2z^2(1-z)^2\phi^{(4)}(z) \\
& \quad + 4\alpha z(1-z)(2z-1)\phi^{(3)}(z) \\
& + \left(\left(\alpha^2-5\alpha-12\right) - 2\left(5\alpha^2-\alpha-12\right)z(1-z)\right)\phi''(z) \\
& \quad - 2\alpha\left(\alpha^2-\alpha-8\right)(2z-1)\phi'(z) \\
& \quad\quad - 4(\alpha+3)(\alpha+1)\alpha\phi(z) = 0
\end{aligned} \tag{7.17}$$

Replacing $\phi(z)$ by $(z(1-z))^\lambda$ and dividing the resulting equation by $(z(1-z))^{\lambda-2}$ we obtain a polynomial equation on z of degree 2. There is no value of λ that makes the three coefficients of this equation equal to zero. Therefore, given that the differential equation is linear, it can not have a solution of the form $\nu(z(1-z))^\lambda$.

8 Conclusions and Future Work.

Throughout this work we provide the average-case analysis of random and fixed PM queries in random relaxed K-dt trees. In particular, we show that the expected cost of fixed PM queries in relaxed K-dt trees does not fulfill –unexpectedly– the conjecture in [10], i.e., it is not of the form that corresponds to randomly balanced hierarchical multidimensional data structures (such as K-d trees and quad trees, see Equation (1.1)). Moreover, it is neither of the form:

$$P_{n,\mathbf{q}} = \nu \cdot \left(\prod_{i:q_i \text{ is specified}} q_i(1-q_i) \right)^\lambda \cdot n^\alpha + \text{l.o.t.}$$

with λ and ν constants (including the case $\lambda = 0$).

Intuitively, the family of relaxed K-dt trees have an expected cost of fixed PM queries somewhere in between that of unbalanced hierarchical multidimensional data structures and that of perfectly balanced ones. An interesting open question is to provide an understanding of the differences between these costs and the way to go from one to the other. It seems that the balancing process of relaxed K-dt trees influences the exponent α as well as the coefficient β. Some preliminary work indicates that when $t \to \infty$ the expected cost of fixed partial match queries in relaxed K-dt trees has the form of the cost of perfectly balanced structures.

Ongoing (and open) work includes the solution of the differential Equation (5.12) for fixed partial match queries when $t \geq 1$.

Future work includes extensions of the analysis in this work to fixed partial match queries in standard K-dt trees trees [6], quad trees of dimension higher than 2 [13] and Quad-K-d trees [2].

References

[1] J. L. Bentley. Multidimensional binary search trees used for associative searching. *Comm. ACM*, 18(9):509–517, 1975.

[2] N. Bereczky, A. Duch, K. Németh, and S. Roura. Quad-k-d trees: A general framework for k-d trees and quad trees. *Theoret. Comput. Sci.*, 616:126–140, 2016.

[3] N. Broutin, R. Neininger, and H. Sulzbach. A limit process for partial match queries in random quadtrees and 2-d trees. *Ann. Appl. Probab.*, 23(6):2560–2603, 2013.

[4] H. H. Chern and H. K. Hwang. Partial match queries in random quadtrees. *SIAM J. Comput.*, 32:904–915, 2003.

[5] H. H. Chern and H. K. Hwang. Partial match queries in random k-d trees. *SIAM J. Comput.*, 35(6):1440–1466, 2006.

[6] W. Cunto, G. Lau, and Ph. Flajolet. Analysis of kdt-trees: kd-trees improved by local reorganisations. In F. Dehne, J. R. Sack, and N. Santoro, editors, *Workshop on Algorithms and Data Structures (WADS'89)*, volume 382 of *Lecture Notes in Computer Science*, pages 24–38. Springer-Verlag, 1989.

[7] N. Curien and A. Joseph. Partial match queries in two-dimensional quadtrees: A probabilistic approach. *Adv. in Appl. Probab.*, 43:178–194, 2011.

[8] A. Duch, V. Estivill-Castro, and C. Martínez. Randomized k-dimensional binary search trees. In K. Y. Chwa and O. H. Ibarra, editors, *Proc. of the 9$^{\text{th}}$ Int. Symp. on Algorithms and Computation (ISAAC)*, volume 1533 of *Lecture Notes in Computer Science*, pages 199–208. Springer-Verlag, 1998.

[9] A. Duch, R. M. Jiménez, and C. Martínez. Selection by rank in k-dimensional binary search trees. *Random Structures & Algorithms*, 45(1):14–37, 2014.

[10] A. Duch, G. Lau, and C. Martínez. On the cost of fixed partial match queries in k-d trees. *Algorithmica*, 75(4):684–723, 2016.

[11] A. Duch, G. Lau, and C. Martínez. Random partial match in quad-k-d trees. In *LATIN 2016: Theoretical Informatics - 12th Latin American Symposium, Ensenada, Mexico, April 11-15, 2016, Proceedings*, pages 376–389, 2016.

[12] A. Duch and C. Martínez. On the average performance of orthogonal range search in multidimensional data structures. *J. Algorithms*, 44(1):226–245, 2002.

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

[14] Ph. Flajolet, G. Gonnet, C. Puech, and J. M. Robson. Analytic variations on quad trees. *Algorithmica*, 10:473–500, 1993.

[15] Ph. Flajolet and A. Odlyzko. Singularity analysis of generating functions. *SIAM J. on Discrete Mathematics*, 3(2):216–240, 1990.

[16] Ph. Flajolet and C. Puech. Partial match retrieval of multidimensional data. *J. Assoc. Comput. Mach.*, 33(2):371–407, 1986.

[17] R. L. Graham, D. E. Knuth, and O. Patashnik. *Concrete Mathematics*. Addison-Wesley, Reading, Mass., 2nd edition, 1994.

[18] P. Kirschenhofer and H. Prodinger. Multidimensional digital searching–alternative data structures. *Random Structures & Algorithms*, 5(1):123–134, 1994.

[19] C. Martínez, A. Panholzer, and H. Prodinger. Partial match queries in relaxed multidimensional search trees. *Algorithmica*, 29(1–2):181–204, 2001.

[20] R. Neininger. Asymptotic distributions for partial match queries in k-d trees. *Random Structures & Algorithms*, 17:403–427, 2000.

[21] P. V. Poblete and J. I. Munro. The analysis of a fringe heuristic for binary search trees. *J. Algorithms*, 6:336–350, 1985.

[22] S. Roura. Improved master theorems for divide-and-conquer recurrences. *J. Assoc. Comput. Mach.*, 48(2):170–205, 2001.

[23] R. Sedgewick. The analysis of quicksort programs. *Acta Informatica*, 7:327–355, 1977.

[24] A. Walker and D. Wood. Locally balanced binary trees. *The Computer Journal*, 19(4):322–325, 1976.