Partial Match Queries in Quad-K-d Trees

Amalia Duch
Universitat Politècnica de Catalunya, Barcelona, Spain

Conrado Martínez
Universitat Politècnica de Catalunya, Barcelona, Spain

Abstract

Quad-K-d trees [Bereckzy et al., 2014] are a generalization of several well-known hierarchical K-dimensional data structures. They were introduced to provide a unified framework for the analysis of associative queries and to investigate the trade-offs between the cost of different operations and the memory needs (each node of a quad-K-d tree has arity 2^m for some m, $1 \leq m \leq K$). Indeed, we consider here partial match – one of the fundamental associative queries – for several families of quad-K-d trees including, among others, relaxed K-d trees and quadtrees. In particular, we prove that the expected cost of a random partial match $\hat{P}_n$ that has s out of K specified coordinates in a random quad-K-d tree of size n is $\hat{P}_n \sim \beta \cdot n^\alpha$ where α and β are constants given in terms of K and s as well as additional parameters that characterize the specific family of quad-K-d trees under consideration. Additionally, we derive a precise asymptotic estimate for the main order term of $P_{n,\mathbf{q}}$ – the expected cost of a fixed partial match in a random quad-K-d tree of size n. The techniques and procedures used to derive the mentioned costs extend those already successfully applied to derive analogous results in quadtrees and relaxed K-d trees; our results show that the previous results are just particular cases, and states the validity of the conjecture made in [Duch et al., 2016] to a wider variety of multidimensional data structures.

2012 ACM Subject Classification Theory of computation → Data structures design and analysis; Theory of computation → Design and analysis of algorithms

Keywords and phrases Quadtree, Partial match queries, Associative queries, Multidimensional search, Analysis of algorithms

Digital Object Identifier 10.4230/LIPIcs.AofA.2022.8

Funding This work has been supported by funds from the MOTION Project (Project PID2020-112581GB-C21) of the Spanish Ministery of Science and Innovation MCIN/AEI/10.13039/501100011033.

1 Introduction

Considered as fundamental associative queries, partial match queries have been widely studied in the literature, see for instance [18, 23]. More specifically, given a collection (or file) $\mathcal{F}$ of n records, in which each *record* in $\mathcal{F}$ is an ordered K-tuple of values (the record's attributes or coordinates), a *query* in $\mathcal{F}$ retrieves the records satisfying certain given conditions. If the imposed conditions deal with more than one attribute, then the query is considered *associative*.

In particular, *partial match* (PM hereinafter) queries consist of retrieving from $\mathcal{F}$ all the records with attributes matching some specified attributes of the given query record. Indeed, the analysis of PM queries (either random or fixed) has been carried out in a wide variety of hierarchical multidimensional data structures, see [6, 7, 8, 11, 14, 15, 21] and references therein.

From the point of view of their analysis, it would be of great interest to unify all these results in a comprehensive way. The general framework of quad-K-d trees (introduced in [4]) is an attempt in that direction. A quad-K-d tree is a multidimensional tree in which each node discriminates with respect to some number m, $1 \leq m \leq K$, of coordinates (and thus it

has 2^m subtrees). The number m and the subset of discriminating coordinates (the *type*) is potentially distinct for each node. When all the K coordinates are used to discriminate at each node, the tree is a quadtree [2], in contrast, when exactly one of the K coordinates is used to discriminate at each node, the tree is a K-d tree [1].

In this work we follow the approach of Chern and Hwang [8] to prove (extending the preliminary results in [11]) that the expected cost $\hat{P}_n$ of a random PM query in a random quad-K-d tree of size n is $\hat{P}_n = \beta \cdot n^\alpha +$ lower order terms (l.o.t.), with α and β given in Theorem 3.

We also give (following the steps in [14] and [12]) a precise asymptotic estimate[1] for the main order term of of the expected cost $P_{n,\mathbf{q}} = \nu \cdot f(\mathbf{q}) \cdot n^\alpha +$ l.o.t. of a fixed partial match with query $\mathbf{q}$ in a random quad-K-d tree of size n, where α is the same as for random PM queries. This is formalized in Theorem 6 where we give the explicit form of the constant ν and of $f(\mathbf{q})$.

Our results apply to any family of quad-K-d trees whose nodes have *types* that are independent from one another. This includes, indeed, random relaxed K-d trees and quadtrees.

The paper is organized as follows. In Section 2 we give some preliminaries of quad-K-d trees (Subsection 2.1) and partial matches (Subsection 2.2). We then derive, in Section 3, the expected cost of a random and fixed PM queries in a random quad-K-d tree (Subsections 3.1 and 3.2, respectively). We finish in Section 4 with conclusions and a brief discussion around further work in this research topic.

2 Preliminaries

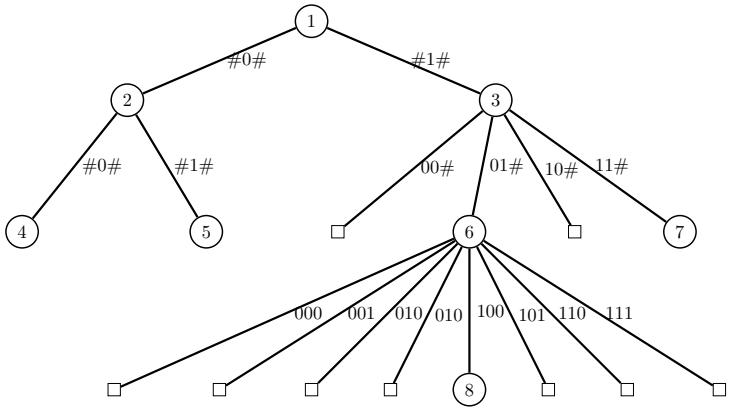
2.1 Quad-K-d Trees

Quad-K-d trees generalize K-dimensional trees [1] and quadtrees [2]. In K-d trees each node $\mathbf{x}$ has a *discriminating* coordinate i, $0 \leq i < K$: all keys $\mathbf{y}$ in the left subtree have their i-th coordinate y_i smaller than x_i. Likewise all keys $\mathbf{z}$ in the the right subtree have their i-th coordinate z_i larger than x_i. Each node thus is associated with a region (*bounding box*) of the domain from which the keys are drawn, and divides that region into two, according to the i-th coordinate being smaller or larger. The choice of which coordinate discriminantes at each node leads to several variants of K-d trees. For quadtrees, each node $\mathbf{x}$ induces a partition of its associated region into 2^K quadrants, and it will have thus 2^K subtrees, one for each quadrant. If we label subtrees with a bitstring $\mathbf{w}$ of length K, the i-th bit of $\mathbf{w}$ indicates if all keys $\mathbf{y}$ in the subtree have $y_i < x_i$ (when $w_i = 0$) or $y_i > x_i$ (when $w_i = 1$).

Quad-K-d trees, being a generalization of both K-dimensional trees and quadtrees, store for each node a subset of m discriminating coordinates, with m ranging from 1 to K, potentially different for each node. We say that such a node is of *t*ype m. Nodes in a K-d tree are all of type 1, nodes in a quadtree are all of type K, and a general quad-K-d tree may contain a mixture of nodes of all possible different types.

The subset of discriminating coordinates of a node is represented by its characteristic function or *coordinate split vector* $\boldsymbol{\delta}$: if $\delta_i = 0$ then the corresponding node does not discriminate with respect to the i-th coordinate, otherwise $\delta_i = 1$ and the node will discriminate

[1] Our result applies under the reasonable but technically hard to establish assumption that the limit of $P_{n,\mathbf{q}}/n^\alpha$ when $n \to \infty$ exists.



Node	Coordinates	Split vector
1	(0.35,0.5,0.17)	010
2	(0.31, 0.45, 0.47)	010
3	(0.29, 0.6, 0.9)	110
4	(0.3, 0.3, 0.3)	001
5	(0.4, 0.47, 0.3)	110
6	(0.2, 0.7, 0.5)	111
7	(0.38, 0.8, 0.89)	011
8	(0.25, 0.68, 0.4)	100

Figure 1 An example of a 3-dimensional quad-K-d tree, omitting some empty subtrees as well as the partition induced by the tree in $[0,1]^3$.

with respect to that coordinate. Thus a node of type m has 2^m subtrees and partitions its associated region of the domain of the keys into 2^m subregions, each subregion associated to a subtree.

Figure 1 shows an example of a 3-dimensional quad-K-d tree. Inside each node appears its label and in the table therein is the 3-dimensional key associated to the node together with its split vector. Next to every edge appears the label (the string $\mathbf{w}$) of the subtree where it points to. When $w_i = 0$ it means that i is a discriminating coordinate ($\delta_i = 1$) and all keys $\mathbf{y}$ in the subtree have $y_i < x_i$. If $w_i = 1$ then i is a discriminating coordinate ($\delta_i = 1$) but now all keys $\mathbf{y}$ in the subtree have $y_i > x_i$. When i is not a discriminating coordinate ($\delta_i = 0$) we indicate so in the subtree/subregion label $\mathbf{w}$ by setting $w_i = \#$.

More formally, a K-dimensional record (or key) is a K-tuple of values $\mathbf{x} = (x_0, \ldots, x_{K-1})$ where each x_i is drawn from a totally ordered domain D_i. The domain $D = D_0 \times \cdots \times D_{K-1}$ is known as the search space and without loss of generality it is usually assumed to be $D = [0,1]^K$. Then, the formal definition of quad-K-d trees is as follows.

▶ **Definition 1** (Bereckzy et al. [3]). *A quad-K-d search tree T of size $n \geq 0$ stores a set of n K-dimensional records, each holding a key $\mathbf{x} = (x_0, \ldots, x_{K-1})$ and a* coordinate split *bitvector $\boldsymbol{\delta} = (\delta_0, \ldots, \delta_{K-1}) \in \{0,1\}^K$. The quad-$K$-d tree T is such that*

- *either it is empty if $n = 0$, or*
- *its root r stores a record with key $\mathbf{x}$, a coordinate split vector $\boldsymbol{\delta}$ that contains exactly m ones (i.e., the root node is of order m), with $1 \leq m \leq K$, and pointers to its 2^m subtrees that store the $n-1$ remaining records as follows: each subtree, let us call it $T_{\mathbf{w}}$, is itself a quad-K-d tree and its associated string $\mathbf{w} = w_0 w_1 \ldots w_{K-1} \in \{0,1,\#\}^K$, is such that for all j, $0 \leq j < K$, $\delta_j = 0 \implies w_j = \#$, and for any key $\mathbf{y} \in T_{\mathbf{w}}$,*
  - *if $\delta_j = 1$ and $w_j = 0$, then $y_j < x_j$*
  - *if $\delta_j = 1$ and $w_j = 1$, then $y_j > x_j$.*

It is worth noting that with this definition, as is the case of binary search trees and other search trees, we do not consider cases in which two or more keys have some identical coordinates; however the definition above can be adapted (and hence the algorithms supported by the data structure) to cover these cases. However, because of our assumptions in the analysis, we can safely disregard this situation since the probability that two records share a coordinate value is 0.

As we have already mentioned, both K-d trees and quadtrees are special cases of quad-K-d trees. In fact, for any quad-K-d tree T of size n, if the split vector $\boldsymbol{\delta}$ associated with every node of T contains all the K coordinates ($\delta_j = 1$ for all j) then T is a quadtree, and if it contains exactly one for every node then T is a relaxed K-d tree [10]. We use here the term *relaxed* K-d trees to stress that the discriminating coordinate of each node is arbitrary, in contrast with standard K-d trees [1], squarish K-d trees [9], and other families of K-d trees where the discriminating coordinate of each node is determined by some fixed rule. Thus relaxed K-d trees are the most general family of K-d trees, of which all other are particular instances. We can have a similar situation with quad-K-d trees: we can define families of quad-K-d trees in which some fixed rule prescribes the types of the nodes and which are the discriminating coordinates of each node. We therefore will use the term *relaxed quad-K-d trees* to emphasise that we consider the most general family of quad-K-d trees, as implied by Definition 1. All over this work, unless otherwise stated, we will refer to relaxed quad-K-d trees as simply quad-K-d trees.

A node holding key $\mathbf{x}$ and split vector $\boldsymbol{\delta}$ in a quad-K-d tree is *of type m* if and only if $\boldsymbol{\delta}$ is of order m, $1 \leq m \leq K$. Every node of type m has 2^m children. A *m-regular* (or m-ary) quad-K-d tree is a quad-K-d tree that has all its internal nodes of type m. Indeed, quadtrees of dimension K are K-regular quad-K-d trees and all variants of K-d trees are 1-regular quad-K-d trees.

The probabilistic analysis of PM queries in quad-K-d trees works under the assumption that the trees under consideration are *random*.

▶ **Definition 2.** *A random relaxed quad-K-d tree of size n is a quad-K-d tree built by n random insertions*[2] *into an initially empty tree, and it additionally satisfies the following two conditions:*

1. *The types of its n nodes are given by n i.i.d. random variables in the set $\{1, \ldots, K\}$. We denote as τ_m the probability that an arbitrary node is of type m.*
2. *For a node of type m, any subset of m coordinates out of K is equally likely to be the set of discriminating coordinates (that is, those for which $\delta_j = 1$). Thus the probability that the discriminating coordinates of a node of type m are $0 \leq i_0 < i_1 < \cdots < i_{m-1} < K$ is $1/\binom{K}{m}$, for any subset $\{i_0, \ldots, i_{m-1}\} \subseteq \{0, \ldots, K-1\}$.*

The previous definition of random quad-K-d trees is equivalent to the conventional random models found in the literature for the particular cases of random quadtrees and random relaxed K-d trees. Several other instances of random quad-K-d trees have been proposed in previous works [4, 11], we refer the reader to these works for a detailed analysis of the space needs and the expected cost of exact searches (IPL) in the different families, relating them to parameters such as the average arity $\overline{d} = \sum_{1 \leq m \leq K} 2^m \tau_m$ and the average order $\overline{m} = \sum_{1 \leq m \leq K} m \tau_m$.

2.2 Partial Match

A PM query is a pair $\langle \mathbf{q}, \mathbf{u} \rangle$, where $\mathbf{q} = (q_0, \ldots, q_{K-1})$ is a K-dimensional key and $\mathbf{u} = (u_0, \ldots, u_{K-1})$ is the *pattern* of the query; each $u_i = S$ (the i-th attributed of the query is *specified*) or $u_i = *$ (the i-th attributed is *unspecified*). Alternatively, we can define a PM

[2] There are several ways to characterize random insertions. The one we will consider here is that every coordinate of the data point to be inserted is independently drawn from some continuous distribution in $[0, 1]$.

query as K-tuple $\mathbf{q} = (q_0, \ldots, q_{K-1})$ where each q_i is a value in the i-th domain D_i or $q_i = *$ to indicate that it is unspecified; we will use most often this second form to represent PM queries.

The PM query $\langle \mathbf{q}, \mathbf{u} \rangle$ is *random* if $\mathbf{q} = (q_0, \ldots, q_{K-1})$ is independently drawn from the same continuous distribution as the data points, otherwise we say that the PM is *fixed*.

The goal of the PM search is to report all data points $\mathbf{x} = (x_0, \ldots, x_{K-1})$ in the tree such that $x_i = q_i$ whenever $q_i \neq *$. The number of specified coordinates will be denoted by s; the interesting cases are when $0 < s < K$. Therefore, to perform a PM search with query $\mathbf{q}$, a quad-K-d tree with root of type m is recursively explored as follows. First, we check whether the root with associated $\mathbf{x}$ matches $\mathbf{q}$ or not, to report it in the former case. Since the node discriminates by m coordinates, let us consider that i of them correspond to specified coordinates in the query ($0 \leq i \leq \min\{s, m\}$). Then, we make recursive calls in all the 2^{m-i} subtrees $T_{\mathbf{w}}$ of the root such that $w_j = \#$ or $q_j = *$, or $w_j = 0$ and $q_j \leq x_j$, or $w_j = 1$ and $q_j > x_j$.

For example, imagine that $K = 5$ and the query is $\mathbf{q} = (0.1, 0.82, *, 0.76, *)$. Moreover, suppose that the root node is of type $m = 3$ with coordinate split vector $\boldsymbol{\delta} = (1, 0, 1, 1, 0)$ and that it contains the key $\mathbf{x} = (0.54, 0.46, 0.39, 0.03, 0.62)$. The 8 subtrees of the root node are labelled as $(0, \#, 0, 0, \#), (0, \#, 0, 1, \#), \ldots, (1, \#, 1, 1, \#)$. It turns out that only $i = 2$ of the discriminating coordinates are specified in the query, the third one is not. Since $0.1 < 0.54$ and $0.76 > 0.03$ we will have to explore the two subtrees $(0, \#, 0, 1, \#)$ and $(0, \#, 1, 1, \#)$, since, even though the third coordinate is a discriminating coordinate of the node, it is not specified in the query: that is, we can't discard the keys of any of these two subtrees.

3 Analysis

The cost of the PM search is measured –as usual in the literature– as the number of nodes visited by the algorithm in the corresponding tree. According to this way of measuring the cost, it is relevant towards our analysis to observe that, except for eventual matches, only matter the relative order of the coordinates of the stored keys.

Indeed, let us call the *rank vector* of a query $\mathbf{q}$ the vector $\mathbf{r}(\mathbf{q}) = (r_0, \ldots, r_{K-1})$ such that $r_i = *$, if $q_i = *$, and r_i is the number of records $\mathbf{x}$ in the collection F such that $x_i \leq q_i$ ($0 \leq r_i \leq n$), if $q_i \neq *$. Then for any two given queries $\mathbf{q}$ and $\mathbf{q}'$ with equal rank vectors $\mathbf{r}(\mathbf{q}) = \mathbf{r}(\mathbf{q}')$ the PM procedure described above will visit exactly the same set of nodes of the tree. In our analysis, we will use rank vectors instead of the queries themselves (as done in [11] and [12]) and consider the cost $\mathcal{P}_{n,\mathbf{r}}$ of a PM query with given rank vector $\mathbf{r}$ in a random quad-K-d tree of size n. We will use $\mathcal{P}_{n,\mathbf{q}}$ for the cost of a PM query with a query $\mathbf{q}$ in a random quad-K-d tree of size n, where each $q_i \in (0, 1)$ or $q_i = *$. From our discussion above $\mathcal{P}_{n,\mathbf{q}} = \mathcal{P}_{n,\mathbf{r}(\mathbf{q})}$. Moreover, because of the symmetries in the model of random quad-K-d trees we can safely assume that the queries are of the form $\mathbf{q} = (q_0, \ldots, q_{s-1}, *, \ldots, *)$ with $q_i \in (0, 1)$ for all i, $0 \leq i < s$, we will abuse of the notation and simply write $\mathbf{q} = (q_0, \ldots, q_{s-1})$ and $\mathbf{r} = (r_0, \ldots, r_{s-1})$, omitting the non-specified coordinates.

3.1 Analysis of Random Partial Match

Let $\hat{P}_n$ denote the expected cost of a random PM query in a random relaxed quad-K-d tree of size n, this corresponds to a query $\langle \mathbf{q}, \mathbf{u} \rangle$ where $\mathbf{q} = (q_0, \ldots, q_{s-1})$ is a random K-dimensional point drawn independently from the same distribution of the data points in the quad-K-d tree.

Here we are going to proceed –as is usual in the literature– obtaining the expected cost P'_n of an "idealised" PM search procedure in which, every time that we recursively invoke it on a subtree, a new random query is generated (but inside the region of the space associated to the particular subtree). The random variable for the cost of this idealised PM search is obviously different from the one that gives the cost of a PM search with a random query; but their respective expected costs coincide: $P'_n = \hat{P}_n$. So, in an abuse of notation, in what follows we will use $\hat{P}_n$ instead of P'_n.

In [11], Roura's continuous master theorem [22] was used to show that the expected cost $\hat{P}_n$ of a random partial match in a random relaxed quad-K-d tree of size n is

$$\hat{P}_n = \Theta(n^\alpha),$$

where α is the unique real solution in the interval $(0,1)$ of the equation

$$\sum_{m=1}^{K} \tau_m \sum_{0\le i\le s} \frac{\binom{m}{i}\binom{K-m}{s-i}}{\binom{K}{s}} \frac{2^m}{(\alpha+1)^{m-i}(\alpha+2)^i} = 1. \tag{1}$$

Here we carry out a more delicate and precise analysis, following the same steps as Chern and Hwang [8] in their analysis of PM in random quadtrees to obtain also the coefficient of the leading term of the cost. We can thus establish the following theorem.

▶ **Theorem 3.** *For any variant of random relaxed quad-K-d trees of size $n \ge 2$, the expected cost $\hat{P}_n$ of a random PM query with s $(0 < s < K)$ specified coordinates, satisfies*

$$\hat{P}_n = \beta_{s,K} n^\alpha + o(n^\alpha),$$

where α is the unique solution in $(0,1)$ of

$$\sum_{m} \tau_m \sum_{i\ge 0} \frac{\binom{K-m}{s-i}\binom{m}{i}}{\binom{K}{s}} \frac{2^m}{(\alpha+1)^{m-i}(\alpha+2)^i} = 1,$$

and

$$\beta_{s,K} = \frac{2^s}{(\alpha_1-1)(1-\alpha_2)\dots(1-\alpha_k)} \frac{1}{\Gamma(\alpha)^{K-s}\Gamma(\alpha+1)^s} \prod_{2\le j\le K} \frac{\Gamma(\alpha-\alpha_j)}{\Gamma(1-\alpha_j)},$$

where $\alpha_1, \alpha_2, \dots \alpha_K$ with $\Re(\alpha_1) \ge \Re(\alpha_2) \ge \dots \ge \Re(\alpha_K)$ are the roots of:

$$\Phi(z) = (z+1)^s z^{K-s} - \sum_{m=1}^{K} 2^m \tau_m \sum_i \frac{\binom{m}{i}\binom{K-m}{s-i}}{\binom{K}{s}} (z+1)^{s-i} z^{K-m-(s-i)},$$

and $\alpha = \alpha_1 - 1$.

Proof. To prove the theorem, we start conditioning on the type m of the root of the random quad-K-d tree. Let $\hat{P}_n^{(m)}$ denote the expected cost of a random PM query, conditional to the root of the quad-K-d tree being of type m; similarly, let $\hat{P}_n^{(i,m)}$ denote the same expected cost now conditional on the root being of type m and that exactly i, $0 \le i \le \min(m,s)$ of the discriminating coordinates of the root are specified in the PM query. Then, since the probability that the root of a random quad-K-d tree is of type m is τ_m we have, for $n > 0$,

$$\begin{aligned}
\hat{P}_n &= \sum_{m=1}^{K} \tau_m \hat{P}_n^{(m)}, \\
\hat{P}_n^{(m)} &= \sum_{0\le i\le m} \mu_{i,m} \hat{P}_n^{(i,m)}, \\
\hat{P}_n^{(i,m)} &= 1 + \sum_{0\le k<n} \pi_{n,k}^{(i,m)} \hat{P}_k,
\end{aligned}$$

where $\mu_{i,m}$ is the probability that exactly i of the m discriminating coordinates are specified (these quantities also depend on s and K), and $\pi_{n,k}^{(i,m)}$ is the average number of recursive calls on root's subtrees of size k when the tree is of size n, its root is of type m and exactly i out of its m discriminating coordinates are specified. It is not difficult to prove that $\mu_{i,m} = \frac{\binom{m}{i}\binom{K-m}{s-i}}{\binom{K}{s}}$ and therefore

$$\hat{P}_n = 1 + \sum_{0\le k<n} \pi_{n,k}\hat{P}_k, \qquad n > 0 \tag{2}$$

where $\pi_{n,k} = \sum_{m=1}^{K} \tau_m \sum_{0\le i\le m} \mu_{i,m}\pi_{n,k}^{(i,m)}$, and $\hat{P}_0 = 0$.

Our next step is to apply the (symmetric) Binomial Transform (see for instance [5, 24] or [20, 5.2.2, p. 136]) to the sequence $\{\hat{P}_n\}_{n\ge0}$, that is,

$$\hat{P}_n^* := \sum_{k=1}^{n} \binom{n}{k} (-1)^k \hat{P}_k, \tag{3}$$

with $\hat{P}_0 = 0$. We can easily prove then the following proposition.

▶ **Proposition 4.** *For $n \ge 2$, the sequence $\hat{P}_n^*$ satisfies the first order recurrence*

$$\hat{P}_n^* - \hat{P}_{n-1}^* = -\mu(n)\hat{P}_{n-1}^*,$$

where $\hat{P}_1^* = -1$ and $\mu(n) := \mu(n,K,s,\tau) = \sum_{m=1}^{K} \tau_m 2^m \sum_{0\le i\le m} \frac{\binom{m}{i}\binom{K-m}{s-i}}{\binom{K}{s}} n^{-m+i}(n+1)^{-i}$.

Proof. As $n \ge 2$, we can replace $\hat{P}_n^*$ and $\hat{P}_{n-1}^*$ by their values as defined in (3); then a few mathematical manipulations yield

$$\begin{aligned}\hat{P}_n^* - \hat{P}_{n-1}^* &= \sum_{k=1}^{n} \binom{n-1}{k-1}(-1)^k \sum_{0\le j<k} \pi_{k,j}\hat{P}_j \\ &= -\sum_{j=0}^{n-1} \hat{P}_j \sum_{k=j}^{n-1} \binom{n-1}{k}(-1)^k \pi_{k+1,j}.\end{aligned}$$

Now, the inner sum of previous equation can be expanded as follows

$$\begin{aligned}\sum_{k=j}^{n-1} \binom{n-1}{k}(-1)^k \pi_{k+1,j} &= \sum_{m-1}^{K} \tau_m 2^m \sum_{0\le i\le m} \frac{\binom{m}{i}\binom{K-m}{s-i}}{\binom{K}{s}} \sum_{k=j}^{n-1} \binom{n-1}{k}(-1)^k \pi_{k+1,j}^{(i,m)} \\ &= \sum_{m=1}^{K} \tau_m 2^m \sum_{0\le i\le m} \frac{\binom{m}{i}\binom{K-m}{s-i}}{\binom{K}{s}} \binom{n-1}{j}(-1)^j n^{-m+i}(n+1)^{-i},\end{aligned}$$

because

$$\sum_{k=j}^{n-1} \binom{n-1}{k}(-1)^k \pi_{k+1,j}^{(i,m)} = \binom{n-1}{j}(-1)^j (n+1)^{-i} n^{-m+i}. \tag{4}$$

To prove this we follow the analogous derivation in [8]; we give it in detail in Appendix A.

Finally, using the definition of $\mu(n)$ given in the statement of the proposition, it follows that for $n \ge 2$,

$$\hat{P}_n^* - \hat{P}_{n-1}^* = -\sum_{j=0}^{n-1} \hat{P}_j \binom{n-1}{j}(-1)^j \mu(n) = -\mu(n)\hat{P}_{n-1}^*. \qquad ◀$$

▶ **Proposition 5.** *For $n \geq 1$, $\hat{P}_n$ satisfies*

$$\hat{P}_n = \sum_{j=1}^{n} \binom{n}{j} (-1)^{j+1} \frac{2^s}{j!^{K-s}(j+1)!^s} (2-\alpha_1)^{\overline{j-1}} (2-\alpha_2)^{\overline{j-1}} \cdots (2-\alpha_K)^{\overline{j-1}},$$

where $x^{\overline{r}} = x \cdot (x+1) \cdot (x+r-1)$ denotes the r-th raising factorial of x [19].

Proof. From Proposition 4, it follows that, for $n \geq 2$,

$$\hat{P}^*_n = (1-\mu(n))\hat{P}^*_{n-1},$$

and iterating,

$$\hat{P}^*_n = (-1)^n \prod_{j=2}^{n} (1-\mu(j))\hat{P}_1 = -\prod_{j=2}^{n} \frac{\Phi(j)}{j^{K-s}(j+1)^s},$$

where $\Phi(z)$ is given in the statement of the theorem and since $\Phi(n)(n+1)^{-s}n^{-(K-s)} = 1-\mu(n)$. Let $\alpha_1, \ldots, \alpha_K$ be the K roots of $\Phi(z)$, then we can write

$$\Phi(z) = (z-\alpha_1)\cdots(z-\alpha_K)$$

and

$$\begin{aligned}\hat{P}^*_n &= -\prod_{j=2}^{n} \frac{(j-\alpha_1)\cdots(j-\alpha_K)}{j^{K-s}(j+1)^s} = -\left(\prod_{j=2}^{n} \frac{j-\alpha_1}{j^{K-s}(j+1)^s}\right)\cdots\left(\prod_{j=2}^{n} \frac{j-\alpha_K}{j^{K-s}(j+1)^s}\right) \\ &= -\frac{2^s}{n!^{K-s}(n+1)!^s}(2-\alpha_1)^{\overline{n-1}}(2-\alpha_2)^{\overline{n-1}}\cdots(2-\alpha_K)^{\overline{n-1}},\end{aligned}$$

Since the binomial transform is an involution, we have

$$\hat{P}_n = \sum_{j=1}^{n} \binom{n}{j} (-1)^j \hat{P}^*_j,$$

and the statement of the proposition follows. ◀

In order to complete the proof of the theorem we obtain the asymptotic behaviour of $\hat{P}_n$ from Proposition 5 using Nørlund-Rice's integrals (see for instance [16, 17]), from where we get the asymptotic estimate given in the theorem. In particular, $\hat{P}_n \sim \beta \cdot n^\alpha$, with the values of α and β given in the statement of the theorem. ◀

Notice that setting $\tau_K = 1$ and $\tau_m = 0$ for $m < K$ in Equation (1) we obtain the indicial equation $\Phi(z) = z^{K-s}(z+1)^s - 2^K = 0$ of quadtrees [8]. Likewise, with $\tau_1 = 1$ and $\tau_m = 0$ for $m > 1$ we obtain

$$\Phi(z) = z^{K-s-1}(z+1)^{s-1}\left(z(z+1) - 2\left[\frac{K-s}{K}(z+1) + \frac{s}{K}z\right]\right),$$

whose roots are $z = 0$, $z = 1$ and the roots of $z(z+1) - 2\frac{K-s}{K}(z+1) + \frac{s}{K}z = 0$, the indicial equation giving the exponent α for relaxed K-d trees (see [10, 21]).

3.2 Analysis of Fixed Partial Match

In this subsection we consider the expected cost of a PM search with rank vector $\mathbf{r}$, or equivalently with a query $\mathbf{q}$, if every coordinate of each data point is independent and uniformly distributed in $(0,1)$, see [11] for a discussion about the two "models" and how results for one translate into results for the other. We shall start stating the main theorem of this section, also one of the major contributions of this extended abstract, and devote the rest of the subsection to schematise its proof.

▶ **Theorem 6.** *Let $\mathbf{r} = (r_0, r_1, \ldots, r_{K-1})$ be a rank vector, where $r_i \in [0..n] \cup \{*\}$, $0 \le i < K$, and such that exactly s of the ranks are specified, that is, we have $r_i \neq *$, for s ranks, $0 < s < K$. Let $z_i = \lim_{n\to\infty} r_i/n$ if $r_i \neq *$ and suppose $z_i \in (0,1)$ for all i such that $r_i \neq *$, $0 \le i < K$.*

Then, for any variant of random quad-K-d trees the expected cost $P_{n,\mathbf{r}}$ of a fixed PM query with rank vector $\mathbf{r}$ in a random quad-K-d tree of size n, is

$$P_{n,\mathbf{r}} = \nu_{s,K} \left(\prod_{r_i \neq *} z_i(1-z_i) \right)^{\alpha/2} n^{\alpha} + l.o.t.,$$

where

$$\nu_{s,K} = \beta_{s,K} \frac{\Gamma^s(\alpha+2)}{\Gamma^{2s}(\alpha/2+1)},$$

and α and $\beta_{s,K}$ are the same as in Theorem 3, provided that

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

exists.

Proof. Let $\mathcal{P}_{n,\mathbf{r}}$ be the cost (number of visited nodes) of a partial match search in a random quad-K-d tree of size n where the query has fixed rank vector $\mathbf{r}$. Our goal is to find the main order asymptotics of $P_{n,\mathbf{r}} = \mathbb{E}\{\mathcal{P}_{n,\mathbf{r}}\}$; more specifically, our goal is to find an explicit function $f(z_0, \ldots, z_{s-1})$ and to show that for $\mathbf{r} = (r_0, \ldots, r_{s-1}, *, \ldots, *)$, if $\lim_{n\to\infty} r_i/n = z_i$, with $z_i \in (0,1)$ for all i, $0 \le i < s$ and

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

exists, with α the exponent of n in the expected cost of a random partial match search (see previous subsection), then the function f is as stated in the theorem.

The first step in our analysis is to setup a recurrence for $P_{n,\mathbf{r}}$. As in our analysis of random PMs, we condition first on the type m of the root node, so we can write

$$P_{n,\mathbf{r}} = \sum_{m=1}^{K} \tau_m P_{n,\mathbf{r}}^{(m)},$$

where $P_{n,\mathbf{r}}^{(m)} = \mathbb{E}\{\mathcal{P}_{n,\mathbf{r}} \,|\, \text{root of type } m\}$. Now, we need to condition on which coordinates are specified by the query and used to discriminate at the root node. Then, we have

$$P_{n,\mathbf{r}}^{(m)} = \sum_{i=0}^{m} \frac{\binom{K-s}{m-i}}{\binom{K}{m}} \left(1 + \sum_{\substack{\mathbf{w}\in(0+1+\#)^K \\ |\mathbf{w}|_{0,1}=i}} Q_{n,\mathbf{r},\mathbf{w}} \right)$$

where $Q_{n,\mathbf{r},\mathbf{w}}$ is the contribution to the total fixed PM cost of the recursive call on subtree $T_{\mathbf{w}}$ when the root of the quad-K-d tree of size n is of type $m = |\mathbf{w}|$ and it discriminates with respect to i of the coordinates that are specified in $\mathbf{r}$ (namely, the root discriminates with respect to those coordinates for which the bitstring $\mathbf{w}$ indexing $T_{\mathbf{w}}$ is 0 or 1; thus $|\mathbf{w}|_{0,1} = i$ and the remaining bits are '#'). The factor $\binom{K-s}{m-i}$ in the formula above takes care of the $m-i$ unspecified coordinates, as it does not matter which $m-i$ coordinates of the root are discriminating coordinates if the query has them unspecified. Moreover, the first summation actually runs from $i = 0$ to $i = \min(s, m)$ since $\binom{K-s}{m-i} = 0$ whenever $i > m$.

The next step is to express $Q_{n,\mathbf{r},\mathbf{w}}$ in terms of $P_{n',\mathbf{r}'}$, where n' is the size of $T_{\mathbf{w}}$ and it will depend on n, $\mathbf{w}$ and the rank vector $\mathbf{j}$ of the root of the quad-K-d tree. Likewise, the rank vector $\mathbf{r}'$ of the recursive call to PM inside $T_{\mathbf{w}}$ will depend on n, $\mathbf{w}$, $\mathbf{r}$ and $\mathbf{j}$. We might thus write

$$\frac{1}{(n+1)^K} \sum_{\mathbf{j}} \sum_{n'} \sum_{\mathbf{r}'} \pi(n, n', \mathbf{r}, \mathbf{r}') P_{n',\mathbf{r}'},$$

where $\pi(n, n', \mathbf{r}, \mathbf{r}')$ is the probability that the rank vector of the PM search in $T_{\mathbf{w}}$ is $\mathbf{r}'$ and the size of $T_{\mathbf{w}}$ is n'.

The full recurrence is quite involved, but the analysis follows closely that of quadtrees [12] with regards to the discriminating coordinates ($w_k \in \{0, 1\}$) and that of relaxed K-d trees [14] when the coordinates are not discriminating ($w_k = \#$). This is because the PM behaves as if the query were unspecified for that particular coordinate (as if $r_k = *$, despite r_k is actually a value in $[0, n]$). When we pass to the limit both sides of the recurrence, and under the assumption that

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

exists, with $z_i = \lim_{n\to\infty} r_i/n$ and $0 < z_i < 1$, the probabilities $\pi(n, n', \mathbf{r}, \mathbf{r}')$ become highly concentrated around the expected values of $\mathbf{r}'$ and thus it becomes considerably simplified. Summations transform into integrals and the recurrence leads to an integral equation for $f(z_0, \ldots, z_{s-1})$.

As we have mentioned before, when the root node of the quad-K-d tree is of type m the behaviour is like that of a quadtree of dimension m; but the choice of discriminating coordinates (equivalently, the coordinate split vector $\boldsymbol{\delta}$) of the node will determine which coordinates will be taken into consideration, and instead of s specified coordinates we shall only have i specified coordinates, with $0 \le i \le \min(s, m)$. Moreover, the identity of the chosen discriminating coordinates matters, as now we are dealing with a fixed partial match – in contrast to random PM where only the distinction between specified or non-specified coordinates is relevant and so they can be handled equally. This is an important aspect that introduces a new degree of difficulty in the analysis of fixed PM in quad-K-d trees. The other difficulty lies in deducing how the rank vector $\mathbf{r}$ changes when the PM recursively continues in a subtree $T_{\mathbf{w}}$, that is, what is the probability that the rank vector is $\mathbf{r}'$ in the recursive call in $T_{\mathbf{w}}$ given that the rank vector was $\mathbf{r}$. In a quadtree all coordinates are discriminating and affect $\mathbf{r}$ in the same way, only depending on the bitvector $\mathbf{w}$ and the key $\mathbf{x}$ at the root. Nevertheless, because of the symmetry of the problem, it is safe to assume that the specified coordinates of the query are the first s coordinates, making the analysis a bit easier.

The situation that we face here is more similar to the case of K-d trees, but in that case there is only a single discriminating coordinate and all the others can be thought of as unspecified at that level, therefore we can condition on the event that i, $0 \le i < K$, is the only specified coordinate. In quad-K-d trees we have to deal with m discriminating

coordinates, and hence we must condition on the event that coordinate split vector is $\boldsymbol{\delta}$, for all $\binom{K}{m}$ possible $\boldsymbol{\delta}$. As the coordinate split vector is implicitly encoded in the labels $\mathbf{w}$ of the subtrees, we can avoid an explicit sum over all posible $\boldsymbol{\delta}$, and just sum over all possible $\mathbf{w}$.

The behaviour of the PM with respect to the $K-s$ unspecified coordinates (the last ones in the query, by assumption) is always the same, whether the root node discriminates or not with respect to any of those coordinates. Thus we can pull out a common factor contributed by the $m-i$ discriminating coordinates which are not specified in the PM query and condition on $\mathbf{w} \in \{0,1,\#\}^s$ with exactly i symbols that are 0 or 1, so we also sum over all possible values of i, $0 \le i \le \min(s,m)$. This is simpler than conditioning over all possible $\mathbf{w} \in \{0,1,\#\}^K$ of which m symbols are 0 or 1, and then condition on i of the first s coordinates being 0 or 1.

Let us begin first with the (identical) contribution of all 2^{m-i} subtrees with a particular choice $\mathbf{k} = (k_0, \ldots, k_{m-i})$ of non-specified discriminating coordinates:

$$\int_0^1 \cdots \int_0^1 f(z_0,\ldots,z_{s-1}) \cdot (u_{k_0}^\alpha + (1-u_{k_0})^\alpha) \cdots (u_{k_{m-i}}^\alpha + (1-u_{k_{m-i}})^\alpha)\, du_{k_0} \cdots du_{k_{m-i}} = \left(\frac{2}{\alpha+1}\right)^{m-i} f(z_0,\ldots,z_{s-1}).$$

We have therefore the following integral equation

$$f(z_0,\ldots,z_{s-1}) = \sum_{m=1}^{K} \tau_m \sum_{i\ge 0} \frac{\binom{K-s}{m-i}}{\binom{K}{m}} \left(\frac{2}{\alpha+1}\right)^{m-i} \times \sum_{\substack{\mathbf{w}\in(0+1+\#)^s \\ |\mathbf{w}|_{0,1}=i}} \Bigg\{ \int_{I_{w_0}(z_0)} \cdots \int_{I_{w_{s-1}}(z_{s-1})} f\Big(\rho_{w_0}(z_0,u_0),\ldots,\rho_{w_{s-1}}(z_{s-1},u_{s-1})\Big) \cdot \Big(\theta_{w_0}(u_0)\cdots\theta_{w_{s-1}}(u_{s-1})\Big)^\alpha du_{s-1}\cdots du_0 \Bigg\}, \quad (5)$$

where $\rho_1(z_i,u_i) = \frac{z_i}{u_i}$, $\rho_0(z_i,u_i) = \frac{1-z_i}{1-u_i}$, $\rho_\#(z_i,u_i) = z_i$, $\theta_1(u) = u$, $\theta_0(u) = 1-u$, $\theta_\#(u) = 1$, $I_1(z) = [z,1]$, $I_0(z) = [0,z]$ and $I_\#(z) = [0,1]$.

Recall that only $i \le m$ coordinates will be specified and discriminating at the root of the quad-K-d tree; the remaining coordinates are either unspecified or non-discriminating and the PM does the same from the point of view of those coordinates; we have thus to sum over all possible choices involving the s specified coordinates: the root does not discriminate for that coordinate ($w_j = \#$), the root discriminates for that coordinate and the rank specified in the PM is less (or equal to) than the rank of the key of the root node ($w_j = 0$) or the root discriminates for that coordinate and the rank specified in the PM is greater than the rank of the key of the root node ($w_j = 1$).

In order to find a solution for the integral equation above, we will use that $f(z_0,\ldots,z_{s-1})$ – if it exists – satisfies the following conditions:

1. $f(z_0,\ldots,z_{s-1})$ is symmetric on all variables, that is, for any i and j,

$$f(z_0,\ldots,z_i,\ldots,z_j,\ldots,z_{s-1}) = f(z_0,\ldots,z_j,\ldots,z_i,\ldots,z_{s-1});$$

and,

2. averaging $P_{n,\mathbf{r}}$ over all possible $\mathbf{r}$ should give us $\hat{P}_n$, hence

$$\int_0^1 \int_0^1 \cdots \int_0^1 f(z_0,\ldots,z_{s-1})\, dz_0 \cdots dz_{s-1} = \beta_{s,K},$$

where $\beta_{s,K}$ is the constant factor of the main order term of $\hat{P}_n$ (see Theorem 3).

We will begin assuming that the integral equation (5) admits a solution on separate variables, that is, $f(z_0, z_1, \ldots, z_{s-1}) = \phi_0(z_0) \cdot \phi_1(z_1) \cdots \phi_{s-1}(z_{s-1})$. And because of the symmetries that f satisifes, namely Condition #1 above, we must have $\phi = \phi_0 = \cdots = \phi_{s-1}$.

Then

$$\begin{aligned}
\phi(z_0)\ldots\phi(z_{s-1}) &= \sum_{m=1}^{K} \tau_m \sum_{i=0}^{m} \frac{\binom{K-s}{m-i}}{\binom{K}{m}} \left(\frac{2}{\alpha+1}\right)^{m-i} \times \sum_{\substack{\mathbf{w}\in(0+1+\#)^s \\ |\mathbf{w}|_{0,1}=i}} \Bigg\{ \\
&\int_{I_{w_0}(z_0)} \cdots \int_{I_{w_{s-1}}(z_{s-1})} \phi(\rho_{w_0}(z_0,u_0))\ldots\phi(\rho_{w_{s-1}}(z_{s-1},u_{s-1}))\times \\
&\qquad \Big(\theta_{w_0}(u_0)\cdots\theta_{w_{s-1}}(u_{s-1})\Big)^{\alpha} du_{s-1}\cdots du_0 \Bigg\} \\
&= \sum_{m=1}^{K} \tau_m \sum_{i=0}^{m} \frac{\binom{K-s}{m-i}}{\binom{K}{m}} \left(\frac{2}{\alpha+1}\right)^{m-i} \sum_{\substack{\mathbf{w}\in(0+1+\#)^s \\ |\mathbf{w}|_{0,1}=i}} \Bigg\{ \\
&\qquad\qquad \prod_{k=0}^{s-1} \int_{I_{w_k}[z_k]} \phi(\rho_{w_k}(z_k,y))(\theta_{w_k}(z_k,y))^{\alpha} dy \Bigg\}.
\end{aligned}$$

Since we are assuming that $f(z_0, \ldots, z_{s-1}) = \phi(z_0)\cdots\phi(z_{s-1})$ the summation on $\mathbf{w}$ on the right hand side can be written

$$\begin{aligned}
&\sum_{\substack{\mathbf{w}\in(0+1+\#)^s \\ |\mathbf{w}|_{0,1}=i}} \Bigg\{ \int_{I_{w_0}(z_0)} \cdots \int_{I_{w_{s-1}}(z_{s-1})} f\Big(\rho_{w_0}(z_0,u_0),\ldots,\rho_{w_{s-1}}(z_{s-1},u_{s-1})\Big) \\
&\qquad \cdot \Big(\theta_{w_0}(u_0)\cdots\theta_{w_{s-1}}(u_{s-1})\Big)^{\alpha} du_{s-1}\cdots du_0 \Bigg\} \\
&= \sum_{\substack{\mathbf{w}\in(0+1+\#)^s \\ |\mathbf{w}|_{0,1}=i}} \Bigg\{ \prod_{k=0}^{s} \int_{I_{w_k}(z_k)} \phi(\rho_{w_k}(z_k,u_k)) \left(\theta_{w_k}(u_k)\right)^{\alpha} du_k \Bigg\} \\
&= \sum_{\mathbf{k}=(k_0,\ldots,k_{i-1})\subset\{0,\ldots,s-1\}} \Bigg\{ \prod_{\ell=0}^{i-1} \Bigg(\int_0^{z_{k_\ell}} \phi(z_{k_\ell}/u) u^{\alpha} du \\
&\qquad\qquad + \int_{z_{k_\ell}}^{1} \phi((1-z_{k_\ell})/(1-u))\,(1-u)^{\alpha} du \Bigg) \cdot \prod_{0\le j<s : j\notin\mathbf{k}} \int_0^1 \phi(z_j) du \Bigg\}.
\end{aligned}$$

Making the change of variables in the integrals ($y := z_i/u_i$ or $y := (1-z_i)/(1-u_i)$, as needed), we arrive at

$$\begin{aligned}
\phi(z_0)\cdot\phi(z_1)\cdots\phi(z_{s-1}) &= \sum_{m=1}^{K} \tau_m \sum_{i=0}^{m} \frac{\binom{K-s}{m-i}}{\binom{K}{m}} \left(\frac{2}{\alpha+1}\right)^{m-i} \sum_{\mathbf{k}=(k_0,\ldots,k_{i-1})\subset\{0,\ldots,s-1\}} \Bigg\{ \\
&\prod_{\ell=0}^{i-1} \left(z_{k_\ell}^{\alpha} \int_0^{z_{k_\ell}} \phi(u) \frac{du}{u^{\alpha+2}} + (1-z_{k_\ell})^{\alpha} \int_{z_{k_\ell}}^{1} \phi(u) \frac{du}{(1-u)^{\alpha+2}} \right) \cdot \prod_{0\le j<s : j\notin\mathbf{k}} \phi(z_j) \Bigg\}. \quad (6)
\end{aligned}$$

Let $\phi(z) = \nu(z(1-z))^{\varphi-1}$, with $\varphi = \alpha/2+1$, for an arbitrary constant ν. If we define

$$L[\phi(z)] = z^\alpha \int_z^1 \phi(y)\frac{dy}{y^{\alpha+2}}, \qquad \text{and}$$
$$R[\phi(z)] = (1-z)^\alpha \int_0^z \phi(y)\frac{dy}{(1-y)^{\alpha+2}},$$

then it is easy to show (see [14] for more details) that

$$\phi(z) = \varphi \cdot (L[\phi(z)] + R[\phi(z)])$$

Let $\mathbf{k} = \{k_0, k_1, \ldots, k_{i-1}\} \subset \{0,\ldots,s-1\}$ and $\hat{\mathbf{k}} = \{\hat{k}_0, \hat{k}_1, \ldots, \hat{k}_{s-1-i}\} = \{0,\ldots,s-1\}\backslash\mathbf{k}$, then, for any $\mathbf{k}$ we have

$$\begin{aligned}
&\phi(z_{\hat{k}_0})\cdots\phi(z_{\hat{k}_{s-1-i}}) \cdot \prod_{\ell=0}^{i-1}(L[\phi(z_{k_\ell})] + R[\phi(z_{k_\ell})]) \\
&\quad = \phi(z_{\hat{k}_0})\cdots\phi(z_{\hat{k}_{s-1-i}}) \cdot \left(\frac{1}{\varphi}\right)^i \cdot \phi(z_{k_0})\cdots\phi(z_{k_{i-1}}) \\
&\quad = \phi(z_0)\cdots\phi(z_{s-1}) \cdot \left(\frac{1}{\varphi}\right)^i.
\end{aligned}$$

Therefore the summation in (6) reads

$$\begin{aligned}
&\sum_{\mathbf{k}:\mathbf{k}\subset\{0,\ldots,s-1\}} \phi(z_{\hat{k}_0})\cdots\phi(z_{\hat{k}_{s-1-i}}) \cdot \prod_{\ell=0}^{i-1}(L[\phi(z_{k_\ell})] + R[\phi(z_{k_\ell})]) \\
&\qquad = \binom{s}{i}\left(\frac{1}{\varphi}\right)^i \phi(z_0)\cdots\phi(z_{s-1}).
\end{aligned}$$

Thus we can express the right hand side of (5) as

$$\begin{aligned}
&\sum_{m=1}^K \tau_m \sum_{i=0}^m \frac{\binom{K-s}{m-i}}{\binom{K}{m}}\left(\frac{2}{\alpha+1}\right)^{m-i}\Bigg\{ \\
&\qquad \sum_{\substack{\mathbf{k}:\mathbf{k}\subset\{0,\ldots,s-1\} \\ \hat{\mathbf{k}}=\{0,\ldots,s-1\}\backslash\mathbf{k}}} \phi(z_{\hat{k}_0})\cdots\phi(z_{\hat{k}_{s-1-i}}) \prod_{\ell=0}^{i-1}(L[\phi(z_{k_\ell})] + R[\phi(z_{k_\ell})])\Bigg\} \\
&= \sum_{m=1}^K \tau_m \sum_{i=0}^m \frac{\binom{K-s}{m-i}}{\binom{K}{m}}\left(\frac{2}{\alpha+1}\right)^{m-i}\binom{s}{i}\left(\frac{1}{\varphi}\right)^i \phi(z_0)\cdots\phi(z_{s-1}) \\
&= \phi(z_0)\cdots\phi(z_{s-1}) \cdot \left\{\sum_{m=1}^K \tau_m \sum_{i=0}^m \frac{\binom{K-s}{m-i}\binom{s}{i}}{\binom{K}{m}} \frac{2^m}{(\alpha+1)^{m-i}(\alpha+2)^i}\right\} \\
&\qquad = \phi(z_0)\cdots\phi(z_{s-1}) = f(z_0,\ldots,z_{s-1}),
\end{aligned}$$

where we have used the fact that the summation enclosed in curly braces of the second-to-last line is, by definition, equal to 1 (see (1)).

To conclude we need to find the value of μ. Using Condition #2, the integral of f in the hypercube $[0,1]^s$ must be equal to $\beta_{s,K}$. That is, we must have

$$\left(\int_0^1 \nu\,(z(1-z))^{\alpha/2}\,dz\right)^s = \beta_{s,K},$$

and thus the value of $\nu_{s,K}$ given in Theorem 6 must be

$$\beta_{s,K}\left(\frac{1}{\int_0^1 (z(1-z))^{\alpha/2}\,dz}\right)^s = \beta_{s,K}\frac{\Gamma^s(\alpha+2)}{\Gamma^{2s}(\alpha/2+1)}.$$ ◀

4 Conclusions and Further Work

We have derived in this paper the expected performance of partial match queries in random relaxed quad-K-d trees, for random as well as for fixed queries. Our results show that the behaviour of random relaxed quad-K-d trees is identical to that of quadtrees and relaxed K-d trees (which are two important particular cases): we always have $\hat{P}_n \sim \beta n^\alpha$ for random partial match and $P_{n,\mathbf{q}} \sim \nu \cdot f(\mathbf{q}) \cdot n^\alpha$ for fixed partial matches, with the exponent α and the constant factors β and ν depending only of the dimension K, the number of specified coordinates s in the partial match and the specific family of quad-K-d trees – which is "encoded" in the probabilities $\{\tau_m\}_{1\leq m\leq K}$.

Our unifying analysis confirms, for this very general family of multidimensional trees –the random relaxed quad-K-d trees–, the conjecture made in [14] on the expected performance of fixed partial matches.

We also see these results as a fundamental first step to establish trade-offs between the amount of space needed to store the data structure and the expected performance of exact searches and of partial matches – the last, in turn, is fundamental to derive the performance of other associative queries, such as orthogonal range queries and nearest neighbour queries (see for instance [9, 13]).

Indeed, although it is not possible to find closed forms for α, β and ν (as they involve the roots of a polynomial of degree K), we can still investigate numerically how these parameters behave on a particular family of relaxed quad-K-d trees, characterised by the probabilities τ. In fact, with the general definition of relaxed quad-K-d trees we have introduce "control knobs" like the probability p in binomial split, pseudo-binomial split and geometric split quad-K-d trees (see [4, 11]), which allow us to smoothly transition from random relaxed K-d trees when $p = 0$ (best memory usage, largest IPL and partial match costs) to random quadtrees when $p = 1$ (worst memory usage, smallest IPL and partial match costs). Another example would be to consider m-regular quad-K-d trees (where all nodes are of type m) and study how different performance parameters evolve with m as we go from $m = 1$ to $m = K$.

In practical settings, such a study is relevant as it helps to select the family of quad-K-d trees (that is, fix the probabilities τ_m) which better fit the requirements of a given application.

Another possible line of research is to extend these results to families of quad-K-d trees which are not *relaxed*, like the standard K-d trees [1] or squarish K-d trees [9] or their generalisations to m-ary trees: these are examples of important families of quad-K-d trees where the types of the nodes are not independently chosen and/or the discriminating coordinates within a given node are not randomly assigned.

References

1 J. L. Bentley. Multidimensional Binary Search Trees Used for Associative Searching. *Communications of the ACM*, 18(9):509–517, 1975.

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

3 Nikolett Bereczky, Amalia Duch, Krisztián Németh, and Salvador Roura. Quad-k-d trees. In *LATIN 2014: Theoretical Informatics - 11th Latin American Symposium, Montevideo, Uruguay, March 31 - April 4, 2014. Proceedings*, pages 743–754, 2014.

4 Nikolett Bereczky, Amalia Duch, Krisztián Németh, and Salvador Roura. Quad-kd trees: A general framework for kd trees and quad trees. *Theor. Comput. Sci.*, 616:126–140, 2016.

5 Khristo N. Boyadzhiev. *Notes on the Binomial Transform.* World Scientific, 2018.

6 N. Broutin, R. Neininger, and H. Sulzbach. A limit process for partial match queries in random quadtrees and 2-d trees. *Annals of Applied Probability*, 23(6):2560–2603, 2013.

7 H.-H. Chern and H.-K. Hwang. Partial match queries in random k-d trees. *SIAM Journal on Computing*, 35(6):1440–1466, 2006.

8 H.-H. Chern and H.-K. Hwang. Partial match queries in random quadtrees. *SIAM Journal on Computing*, 32(4):904–915, 2012.

9 L. Devroye, J. Jabbour, and C. Zamora-Cura. Squarish k-d trees. *SIAM Journal on Computing*, 30:1678–1700, 2000.

10 A. Duch, V. Estivill-Castro, and C. Martínez. Randomized K-dimensional binary search trees. In K. Y. Chwa and O. H. Ibarra, editors, *9th Annual International Symposium on Algorithms and Computation (ISAAC 98)*, volume 1533 of *Lecture Notes in Computer Science*, pages 199–208, 1998.

11 A. Duch, G. Lau, and C. Martínez. Random partial match in quad-k-d trees. In E. Kranakis, G. Navarro, and E. Chávez, editors, *LATIN 2016: Theoretical Informatics (Proceedings of the 12$^{\text{th}}$ Latin American Theoretical Informatics Conference (LATIN))*, volume 9644 of *Lecture Notes in Computer Science*, pages 376–389. Springer-Verlag, 2016.

12 A. Duch, G. Lau, and C. Martínez. Fixed partial match queries in quadtrees. In J.A. Fill amd M.D. Ward, editor, *Proceedings of the 29$^{\text{th}}$ International Meeting on Probabilistic, Combinatorial and Asymptotic Methods for the Analysis of Algorithms (AofA)*, volume 110 of *Leibniz International Proceedings in Informatics (LIPIcs)*, pages 20:1–20:18, 2018.

13 A. Duch and C. Martínez. On the average performance of orthogonal range search in multidimensional data structures. *Journal of Algorithms*, 44(1):226–245, 2002. doi:{10.1016/S0196-6774(02)00213-4}.

14 Amalia Duch, Gustavo Lau, and Conrado Martínez. On the cost of fixed partial match queries in k-d trees. *Algorithmica*, 75(4):684–723, 2016.

15 Ph. Flajolet and C. Puech. Partial match retrieval of multidimensional data. *Journal of the ACM*, 33(2):371–407, 1986.

16 Philippe Flajolet and Robert Sedgewick. Mellin transforms and asymptotics: Finite differences and rice's integrals. *Theoretical Computer Science*, 144(1&2):101–124, 1995. doi:10.1016/0304-3975(94)00281-M.

17 Philippe Flajolet and Robert Sedgewick. *Analytic Combinatorics.* Cambridge University Press, 2009. URL: http://www.cambridge.org/uk/catalogue/catalogue.asp?isbn=9780521898065.

18 V. Gaede and O. Günther. Multidimensional access methods. *ACM Computing Surveys*, 30(2):170–231, 1998.

19 Ronald L. Graham, Donald E. Knuth, and Oren Patashnik. *Concrete Mathematics: A Foundation for Computer Science, 2nd Ed.* Addison-Wesley, 1994.

20 D. E. Knuth. *The Art of Computer Programming: Sorting and Searching*, volume 3. Addison–Wesley, 2nd edition, 1998.

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

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

23 H. Samet. *The Design and Analysis of Spatial Data Structures.* Addison-Wesley, 1990.

24 Neil J.A. Sloane and Simon Plouffe. *The Encyclopedia of Integer Sequences.* Academic Press, 1995.

A Proof of Equation (4)

Our goal here is to prove (4):

$$\sum_{k=j}^{n-1}\binom{n-1}{k}(-1)^k\pi_{k+1,j}^{(i,m)} = \binom{n-1}{j}(-1)^j(n+1)^{-i}n^{-m+i},$$

an intermediate result in the proof of Propositon 4. Recall that $\pi_{k+1,j}^{(i,m)}$ denotes the average number of top-level recursive calls on subtrees of size j when the quad-K-d tree is of size $k+1$, its root is of type m and exactly i discriminating coordinates out of m are specified. But this number is exactly the same as the average number of top-level recursive calls on subtrees of size j on a m-dimensional quadtree of size $k+1$ for a random partial match in which i of the m dimensions are specified; and this probability is given by [8, Eq. (2.1)]

$$\pi_{k+1,j}^{(i,m)} = \binom{k}{j}\int_{[0,1]^m}\mathbf{dx}\; x_0\cdots x_{i-1}(x_0\cdots x_{m-1})^j(1-x_0\cdots x_{m-1})^{k-j},$$

where $\mathbf{dx} := dx_0\cdots dx_{m-1}$.

We can then write the LHS of (4), for a fixed j,

$$\begin{aligned}
&\sum_{k=j}^{n-1}\binom{n-1}{k}(-1)^k\pi_{k+1,j}^{(i,m)}\\
&\quad=\sum_{k=j}^{n-1}\binom{n-1}{k}(-1)^k\binom{k}{j}\int_{[0,1]^m}\mathbf{dx}\; x_0\cdots x_{i-1}(x_0\cdots x_{m-1})^j(1-x_0\cdots x_{m-1})^{k-j}\\
&\quad=\binom{n-1}{j}\sum_{k=j}^{n-1}\binom{n-1-j}{k-j}(-1)^k\times\Bigg\{\\
&\qquad\int_{[0,1]^m}\mathbf{dx}\; x_0\cdots x_{i-1}(x_0\cdots x_{m-1})^j(1-x_0\cdots x_{m-1})^{k-j}\Bigg\}\\
&\quad=\binom{n-1}{j}(-1)^j\Bigg\{\sum_{\ell=0}^{n-1-j}\binom{n-1-j}{\ell}(-1)^\ell\\
&\qquad\times\int_{[0,1]^m}\mathbf{dx}\; x_0\cdots x_{i-1}(x_0\cdots x_{m-1})^j(1-x_0\cdots x_{m-1})^{\ell}\Bigg\}\\
&\quad=\binom{n-1}{j}(-1)^j\int_{[0,1]^m}\mathbf{dx}\; x_0\cdots x_{i-1}(x_0\cdots x_{m-1})^j\Bigg\{\sum_{\ell=0}^{n-1-j}\binom{n-1-j}{\ell}\times\\
&\qquad(-1)^\ell(1-x_0\cdots x_{m-1})^{\ell}\Bigg\}\\
&\quad=\binom{n-1}{j}(-1)^j\int_{[0,1]^m}\mathbf{dx}\; x_0\cdots x_{i-1}(x_0\cdots x_{m-1})^j(x_0\cdots x_{m-1})^{n-1-j}\\
&\quad=\binom{n-1}{j}(-1)^j\int_{[0,1]^m}\mathbf{dx}\; x_0\cdots x_{i-1}(x_0\cdots x_{m-1})^{n-1}\\
&\quad=\binom{n-1}{j}(-1)^j\left(\int_0^1 x^n dx\right)^i\cdot\left(\int_0^1 x^{n-1}dx\right)^{m-i}=\binom{n-1}{j}(-1)^j(n+1)^{-i}n^{-m+i}.
\end{aligned}$$