Computer Science Review 61 (2026) 100904

Contents lists available at ScienceDirect

Computer Science Review

journal homepage: www.elsevier.com/locate/cosrev

The analysis of partial match queries in random multidimensional trees: A selected survey

Amalia Duch [a,1], Hsien-Kuei Hwang [b,1], Conrado Martínez [a,1,*], Ralph Neininger [c,1]

[a] Department of Computer Science, Universitat Politècnica de Catalunya, Jordi Girona, 1–3, Barcelona, 08034, Spain
[b] Institute of Statistical Science, Academia Sinica, 128 Academia Road Sec.2, Nankang, Taipei, 115, Taiwan
[c] Institute of Mathematics, Goethe University Frankfurt, Robert-Mayer-Straße 10, 60325, Frankfurt a. M., Germany

ARTICLE INFO

Keywords:
Multidimensional data structures
k-dimensional trees
Quadtrees
Partial match queries
Associative queries

ABSTRACT

We analyze the probabilistic performance of partial match queries in random multidimensional search trees, using *k-d* trees (and their variants) and quadtrees, as primary examples. A partial match query aims to retrieve all points from a tree that match a given query point on a predetermined subset of coordinates. As one of the most basic associative searches, its analysis is foundational for evaluating more complex queries. This selection of data structures allows us to demonstrate common analytical techniques and offer a historical perspective on the field.

1. Introduction

To a computer scientist, Piet Mondrian's non-representational, grid-like abstractions of objects with primary colors are reminiscent of the bounding-box decomposition of a two-dimensional space by a hierarchical data structure like a k-d tree (see Fig. 1). Such spatial partitions are often encountered, though usually implicitly, in our digital era. For example, when navigating with GPS, exploring virtual environments, searching the web, or playing video games, k-d trees or similar structures are often at work behind the scenes to retrieve information efficiently.

More concretely, text may be seen as points in a high-dimensional space for many clustering problems; a GPS map is often displayed as two-dimensional data; augmented reality can be modelled with three-dimensional data; and game graphics work well with three- and higher-dimensional data. Physicists even consider the whole Universe to be eleven-dimensional.

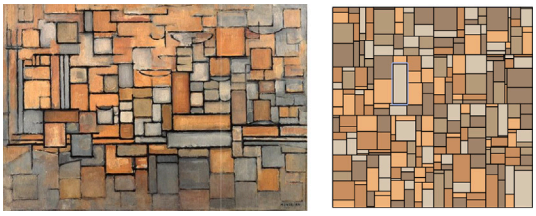

Fig. 1. Left: Piet Mondrian's 1913 painting Gemälde No. 1; right: a decomposition of the unit square by a simulation of a random 2-d tree generated byinserting 256 random points with randomly assigned colors.

The volume of multidimensional data that is generated and has to be stored often grows exponentially with the dimensionality. Consequently, *multidimensional data structures* (also known as *multidimensional access methods*, *spatial access methods* or *spatial index structures*, see [48] for a review) are necessary to efficiently store and manipulate data; k-d trees represent one of the simplest and most fundamental models.

More formally, a multidimensional data structure maintains a finite collection $\mathcal{F}$ of items (also called *records*). Each item contains a distinct k-dimensional key $\mathbf{x} = (x_0, \ldots, x_{k-1})$, where x_i is a value from a totally ordered domain D_i; thus $\mathcal{F} \subset D = D_0 \times \cdots \times D_{k-1}$.

For convenience of analysis, it is usually assumed, w.l.o.g., that D_i is the real interval $[0, 1]$, and $D = [0, 1]^k$. We also make this assumption when describing data structures and algorithms; the more general case where the keys $\mathbf{x} \in D$ for some domain $D = D_0 \times \cdots \times D_{k-1}$ presents no special difficulty.

Retrieving (or searching) data in multidimensional data structures is usually known as *associative data retrieval*. In addition to insertions, deletions, and exact search, a multidimensional data structure often supports queries such as *orthogonal range search* (identifying records in the data

* Corresponding author.
Email address: conrado@cs.upc.edu (C. Martínez).

[1] The authors contributed equally to this work.

https://doi.org/10.1016/j.cosrev.2026.100904
Received 26 June 2024; Received in revised form 30 November 2025; Accepted 12 January 2026

structure that are contained in a given hyper-rectangle Q), and *nearest neighbor search* (finding the closest record to a given point q according to some distance or similarity metric); see [48,76]. At the heart of associative retrieval lies the concept of *partial match*, whose analysis in several data structures is the main subject of this work. Given a query point $\mathbf{q}$ in which some coordinate values are given and others are left unspecified, a partial match algorithm aims to identify and retrieve items whose coordinates match the specified coordinates of $\mathbf{q}$, regardless of the unspecified ones.

Goal of the survey. The goal of this survey is to present the techniques and results underlying the probabilistic analysis of partial match searches in random multidimensional search trees, with a particular focus on k-d trees, quadtrees, and variants. We briefly discuss the reasons for these choices below.

First, understanding partial match searches is crucial for the probabilistic analysis of more complex associative queries, such as orthogonal range and nearest-neighbor queries (see, for instance, [13,30]). Partial match search is hence not only of independent interest, but also a key step toward the analysis of other multidimensional queries.

Another reason is that the probabilistic analysis of multidimensional data structures and associative queries is notoriously difficult and challenging. Although many powerful mathematical techniques have been developed in the forty years since Flajolet and Puech's seminal paper [44]—and it is our purpose to present and discuss several of them in some detail—little is known about the probabilistic behavior of many multidimensional data structures for point data, and even less for spatial data in general, especially for data structures that were designed to provide worst-case guarantees. The search for probabilistic models that are both relevant and mathematically tractable further increases the difficulty: in many real-life applications, the underlying distributions are simply too complicated for our current analytical toolbox. It is therefore not surprising that the performance of multidimensional data structures is often investigated empirically, as a theoretical complexity analysis is unavailable, or it requires making a substantial number of simplifying assumptions and introducing several parameters —which can be determined only empirically— to derive a theoretical model that yields good predictions on the observed performance. It is important to note that for multidimensional data structures and associative queries worst-case upper bounds are often trivial and mostly uninformative.

The theoretical results discussed in this survey do apply to important and interesting situations, even if their scope is somewhat limited. At present, it seems infeasible to extend them to other well-known multidimensional data structures, such as range trees, R-trees and their variants (e.g., X-trees and TV-trees), among others [7,9,49,54,57]. However, the models and techniques presented here may still provide useful approximate descriptions in these more sophisticated settings, or suggest plausible hypotheses about their performance that can be tested empirically. This is a particularly valuable avenue, as results derived from more idealized models (for instance those assuming coordinate independence) are often observed to hold in more general situations, even when complex dependence among coordinates is present.

A third reason for our selection of topics is that the mathematical techniques that we will explore in the next sections go beyond their specific application to analyze multidimensional searches; they have proven very useful in many other applications, in particular in the probabilistic or average-case analysis of fundamental algorithms like quicksort or quickselect, and of data structures like linear probing hash tables, tries, binary search trees, and many more (see, for instance [46,81]). Conversely, the early efforts in the analysis of partial match in k-d trees and other multidimensional data structures, most notably Flajolet and Puech's seminal paper [44] were largely influential in the development of the *singularity analysis techniques* (see section 6 in Part II, and helped shape the area that later became known as *Analytic Combinatorics* [46].

Last but not least, k-d trees and quadtrees were historically among the first multidimensional data structures proposed to provide efficient support for associative queries. Although many other multidimensional data structures have been developed since then, k-d trees, quadtrees, and their variants still have numerous applications, especially for low-dimensional data, for example, in volume rendering [47,52], ray tracing [51,86], collision detection in 2D and 3D [22,83,84], among many others.

Structure of the survey. Part I contains the introductory material. Section 2 is devoted to the presentation of the main data structures introduced for multidimensional search, in particular k-d trees and quadtrees, which are the central data structure objects of this survey. The formal definition of associative queries, and in particular that of partial match queries, is given in Section 3, together with the algorithms to handle these queries in multidimensional trees such as k-d trees and quadtrees. To analyze partial match queries probabilistically, it is necessary to specify a probabilistic model for the input data, that is, a probability distribution on the data, which in turn leads to what we call random k-d trees and quadtrees; this is the subject of Section 4.

Part II contains the core of this survey. In it, we review some representative results about the performance of partial match searches in different multidimensional trees, illustrating the main techniques and tools that were used to obtain those results. The first section offers a broad historical review, briefly surveying most of the known results and their development over time. The subsequent sections are organized around the fundamental techniques involved and provide details and intuition about selected results and how they were obtained. Section 6 covers some of the most celebrated and well-established mathematical tools based on complex analysis, central to the area of Analytic Combinatorics [46], such as singularity analysis and Mellin transforms. After that, Section 7 reviews some of the strongest known results that establish convergence to limiting distributions and characterize those limits for the quantities of interest ("cost of partial match search in such and such data structure"), with the powerful contraction method as the main tool behind most of them. Finally, Section 8 is devoted to master theorems and related methods, in particular we review the continuous master theorem (CMT) and its application to the analysis of partial match queries in k-d trees. Thereafter, we consider extensions of the CMT and the techniques behind its proof, to cope with the analysis of random partial match searches in some variants of k-d trees such as the standard or hybrid median, where the ordinary CMT does not work, and to cope with the analysis of the expected cost of partial matches given a fixed query $\mathbf{q}$, for several different multidimensional trees and arbitrary dimensions. These last methods lack the rigor and reach of the results in Section 7, but they provide useful results about the expected performance of fixed query partial matches for which the rigorous probabilistic analysis is still lacking.

Part I: Data Structures, Associative Queries and Data Models

2. Data structures for multidimensional search problems

Multidimensional data structures are often classified in the following ways:

- static or dynamic, depending on whether online insertion and deletion of items are allowed;
- space-driven or data-driven, according to the way they organize the stored items;
- hierarchical or non-hierarchical, depending on their shape being tree-like or not; and
- general purpose or ad-hoc according to their design purpose and functionality.

As we previously mentioned, we focus in this paper on general purpose data-driven hierarchical multidimensional data structures, although for completeness we also briefly mention in this section some

representative data structures of each group. For more information and insights on a huge variety of multidimensional data structures, including, but not limited to, those mentioned here, we refer the interested reader to [34,48,76,77,82].

For simplicity, we assume that the k-dimensional records (or keys) to be stored are all *compatible*.

Definition 1 (Compatibility). We say that two k-dimensional keys $\mathbf{x}$ and $\mathbf{y}$ are *compatible* if $x_i \neq y_i$ for every $i \in \{0, 1, \ldots, k-1\}$.

To analyze the typical behavior of data structures and algorithms, we assume the following random model.

Definition 2 (Random model). A *random sample* (*random data set*, *randomly generated sample*), is a collection of records (keys, points) that are independently and identically distributed in $[0,1]^k$, all drawn from some continuous probability distribution in $[0,1]^k$.

In most cases, the analysis in this paper applies to random data sets whose coordinates have arbitrary continuous marginal distributions [56]. However, in some situations, it is more convenient to assume that the data structures are built by successive insertions of independent and uniformly distributed points from $[0,1]^k$. Note that any two k-dimensional keys generated in this way are compatible (see Definition 1). Sometimes we shall also make the stronger assumption that every attribute of the keys in $\mathcal{F}$ is drawn *independently and uniformly* at random from $[0,1]$.

2.1. Basic multidimensional data structures

The simplest data structures dealing with multidimensional data consist of a set of k *linked lists*, which all require $\Theta(kn)$ (n being the length of the lists) for the storage, construction time, and associative queries in the worst case.

Lists can be improved by means of the *projection* technique (also called *inverted files*) in which each coordinate of the list is sorted. Geometrically, this corresponds to project, on each coordinate, the set of stored points. Such a pre-sorting stage of the inverted file takes $\Theta(kn \log n)$ time in the worst case for a sample of n records in $[0,1]^k$ (more generally, records in $\mathcal{D}$). The storage use is of order $\Theta(kn)$. An exact match query can be answered by performing binary search in any of the k sorted lists in $\Theta(\log n)$ worst-case time. However, the worst-case cost for any kind of associative queries remains $\Theta(kn)$.

Despite their higher linear time, these multidimensional data structures are easily implemented, and competitive when the file is small or of moderate size, or when the match rate of associative queries is high (significant portions of the file satisfy the query).

2.2. Data driven multidimensional trees

This subsection is devoted to the description of the multidimensional trees on which we will focus our analysis: k-dimensional binary search trees and quadtrees, two generalizations of binary search trees. Quad k-d trees [2,3] further generalize k-d trees and quadtrees, but we do not present them here. Keep in mind that quadtrees and k-d trees are multidimensional hierarchical trees whose shape and construction depend not only on the relative positions of the records in the space, but also on the order in which these records are inserted into the data structure.

Recall that throughout these descriptions we assume that the records stored in these trees are mutually compatible, since the definitions and algorithms, as presented, are only correct under this assumption. If equality does arise in applications, then the data structures and the algorithms should be slightly modified [21,34,75,76]. Moreover, for the purposes of our analysis, we assume that these trees are random.

Definition 3 (Random tree). We say that a multidimensional tree (a k-d tree, k-d tree or quad k-d tree) is *random* if it is generated by successive insertions of records from a randomly generated sample, as in Definition 2.

2.2.1. k-dimensional binary trees

Bentley [4] introduced standard k-dimensional binary search trees (standard k-d trees), which are generalized binary search trees that divide the space ($[0,1]^k$) into hyper-rectangles (as stated in Definition 4).

Definition 4 (k-d tree [4]). A standard k-d tree for a set $\mathcal{F} = \{\mathbf{x}^{(1)}, \mathbf{x}^{(2)}, \ldots, \mathbf{x}^{(n)}\}$ of compatible k-dimensional records is a binary search tree such that:

1. Each node contains a k-dimensional record and has an associated discriminant $j \in \{0, 1, \ldots, k-1\}$.
2. For every node with key $\mathbf{x}$ and discriminant j, the following invariant holds: any record in the left subtree with key $\mathbf{y}$ satisfies $y_j < x_j$, and any record in the right subtree with key $\mathbf{y}$ satisfies $y_j > x_j$.
3. The root node has depth 0 and discriminant 0. All nodes at depth d have discriminant $(d \bmod k)$.

The implementation of standard k-d trees does not require explicitly storing the discriminant at each node, as it can be implicitly determined from the third condition above.

However, other variants do require storing discriminants, and in that case we will speak more generally of a node $\langle \mathbf{x}, j \rangle$, meaning that the node contains a record with key $\mathbf{x}$ and an associated discriminant j. An example of a standard k-d tree, together with the induced partition of the space, is given in Fig. 2.

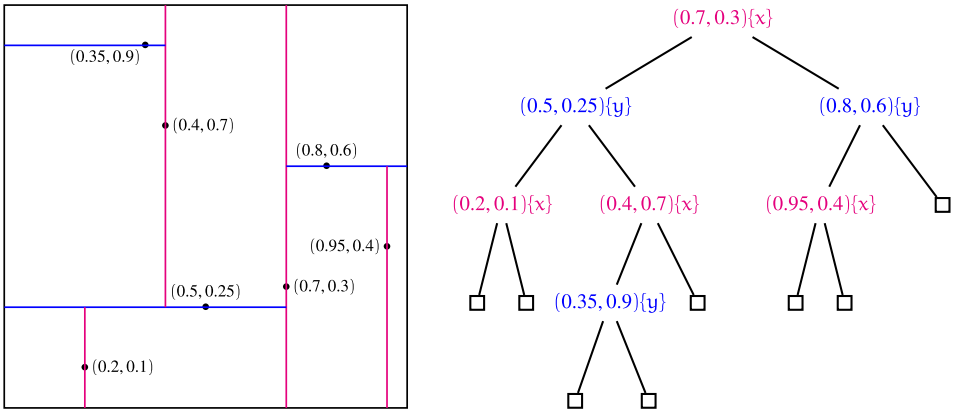

Fig. 2. Example of a k-d tree built from 2-dimensional points.

A k-d tree for a file $\mathcal{F}$ can be incrementally built by successive insertions into an initially empty k-d tree as follows. The insertion of $\mathbf{x}$ in an empty tree T results in a k-d tree with root node $\langle \mathbf{x}, 0\rangle$ and empty subtrees. An insertion of $\mathbf{y}$ at the root $\langle \mathbf{x}, i\rangle$ of some subtree, compares the new key's i-th coordinate y_i with the i-th coordinate x_i of the key at the root: if $y_i < x_i$, the new key is recursively inserted into the left subtree; otherwise, if $y_i > x_i$ then $\mathbf{y}$ is recursively inserted in the right subtree.

When the insertion algorithm reaches an empty subtree (leaf) at depth d, a new node $\langle \mathbf{y}, d \bmod k\rangle$ is created and it replaces the leaf.

We now introduce the concept of *bounding box* or *bounding hyper-rectangle* of a node.

Definition 5 (Bounding box). Given a key $\mathbf{x}$ in a k-d tree T, its *bounding hyper-rectangle* $B(\mathbf{x}) = [\ell_0(\mathbf{x}), u_0(\mathbf{x})]\times\ldots\times[\ell_{k-1}(\mathbf{x}), u_{k-1}(\mathbf{x})]$ is the region of $[0,1]^k$ defined as follows:

1. If $\mathbf{x}$ is the root of T then $B(\mathbf{x}) = [0,1]^k$;
2. If $\langle \mathbf{y}, j\rangle$ is the parent of $\mathbf{x}$ in T, and its bounding box is $B(\mathbf{y}) = [\ell_0, u_0]\times\cdots\times[\ell_{k-1}, u_{k-1}]$, then
   - if $x_j < y_j$ then $B(\mathbf{x}) = [\ell_0, u_0]\times\cdots\times[\ell_j, y_j]\times\cdots\times[\ell_{k-1}, u_{k-1}]$;
   - if $x_j \geqslant y_j$ then $B(\mathbf{x}) = [\ell_0, u_0]\times\cdots\times[y_j, u_j]\times\cdots\times[\ell_{k-1}, u_{k-1}]$.

Observe that $B(\mathbf{x})$ is the region of $[0,1]^k$ corresponding to the leaf that $\mathbf{x}$ replaced when $\mathbf{x}$ was inserted into the k-d tree. We thus extend the notion of bounding boxes to leaves: the bounding box $B(v)$ of a leaf v is the region of $[0,1]^k$ associated with v; if we insert a new key $\mathbf{x}$ belonging to $B(v)$ into the tree, then v will be replaced by a node containing $\mathbf{x}$. Collectively, the bounding boxes of the leaves partition the entire search space $[0,1]^k$.

In general, the more balanced the tree (or the more regular the induced partition), the more efficient the exact searches and the associative queries. If the n keys to store in the tree are known in advance, one can choose the order of insertion and/or the discriminants to produce well-balanced trees, and a partition of the space as regular as possible; see [48,77] for several options available for the *offline* construction of k-d trees. An interesting *online* alternative to construct more balanced trees is standard k-d-t trees, introduced by Cunto, Lau and Flajolet [17]. In these trees, local rebalancing of small subtrees is applied to improve the global balance of the tree. Discriminants for standard k-d-t trees are assigned exactly the same way as in standard k-d trees (or k-d-t trees with $t = 0$).

In this paper, we focus mostly on random standard k-d trees, a typical instance of which is comparatively balanced and performs efficiently, as we show later. In addition to this standard model, we also study some other variants that differ only in the way the discriminants are assigned in each node of the tree; see Table 1. In particular, we address relaxed k-d trees [23,31], and two other variants that rely on the assumption of the uniform distribution of keys, namely, the squarish k-d trees proposed by Devroye, Jabbour and Zamora-Cura [DevJabZam00], and the median and hybrid median k-d trees introduced in [33]. These trees are built using heuristic procedures to achieve more balanced trees or more regular partitions of the search space.

Table 1
Rules to assign the discriminants of each node in several variants of k-d trees.

Types of k-d trees	Rule for discriminants
standard	cyclically $(0, 1, \ldots, k-1, 0, \ldots)$
relaxed	uniformly at random in $\{0, 1, \ldots, k-1\}$
squarish	dimension with larger side of the node's bounding box
median	dimension with the most centered value in the node's bounding box
hybrid median	combination of median and standard: dimension with the most centered value in the node's bounding box with no repeated discriminant within each sequence of k consecutive splits
Local balancing of small subtrees can be applied to each of the above variants	

Although the assumption of uniformity of keys may not be met in many applications, we can avoid dealing with insertions that lead to very poorly balanced trees (for example, 2-dimensional data inserted in increasing order of their x coordinate). The randomization techniques in [62] for binary search trees can be generalized to k-d trees (and also to quadtrees, see next subsection) [21,23]. Thanks to randomization, the trees will behave as if they were produced by i.i.d. random insertions, even in cases where the insertions are not in random order. Therefore, theoretical results will hold, as long as they do not require uniformly distributed data.

2.2.2. Quadtrees

Quadtrees[2] were introduced by Finkel and Bentley [36,76]: like k-d trees, they are another generalization of binary search trees, but unlike k-d trees, they are no longer binary but 2^k-ary; see Definition 7, which requires the following dominance relation $\prec_w$ between two k-dimensional keys.

Definition 6 (Partial dominance). Given a bit string $w = w_0 w_1 \cdots w_{k-1}$ of length k and two compatible k-dimensional keys $\mathbf{x}$ and $\mathbf{y}$, we say that $\mathbf{x}$ *is related to* $\mathbf{y}$ *under the relation* $\prec_w$, and write $\mathbf{x} \prec_w \mathbf{y}$, if and only if, for all $i \in \{0, 1, \ldots, k-1\}$, we have $x_i < y_i$ whenever $w_i = 0$ and $x_i \geqslant y_i$ whenever $w_i = 1$. Otherwise, we say that $\mathbf{x} \not\prec_w \mathbf{y}$.

The relation $\prec_w$ is antisymmetric and transitive, and since the keys $\mathbf{x}$ and $\mathbf{y}$ are compatible, if $\mathbf{x} \prec_w \mathbf{y}$, then $\mathbf{y} \prec_{\overline{w}} \mathbf{x}$, for every $w \in \{0,1\}^k$, where $\overline{w}$ is the complementary bit string of w: $\overline{w}_i = 1 - w_i$, for all i, $0 \leqslant i < k$. Furthermore, $\mathbf{x} \prec_w \mathbf{y}$ for exactly one bit string $w \in \{0,1\}^k$ and $\mathbf{x} \not\prec_{w'} \mathbf{y}$ for all other w'.

Definition 7 (Quadtree). A *quadtree* T for a set of k-dimensional keys is either an empty tree, or a 2^k-ary tree in which

1. each node contains a k-dimensional key and pointers to 2^k subtrees T_w for all $w \in \{0,1\}^k$, and
2. any record with key $\mathbf{x}$ in the w-th subtree of node $\mathbf{y}$ satisfies $\mathbf{x} \prec_w \mathbf{y}$.

With a slight abuse of language, the w-th subtree or w-th quadrant is referred to as the subtree T_w or the corresponding hyper-quadrant w. An example of a quadtree, together with its induced partition of the space, is given in Fig. 3.

The insertion of a new element in an existing quadtree is then straightforward from Definition 7: a comparison of the new element with the root determines the subtree where the insertion algorithm must proceed recursively.

In general, similar to k-d trees, the more balanced the tree (or the more regular the induced partition), the more efficient the query-response. There are also several variants to keep quadtrees well-balanced; see [48,77]. Here we focus uniquely on randomly built quadtrees, whose typical shape tends to be balanced. Randomization techniques have also been developed for quadtrees [35], and they can be used if needed.

2.3. Space-driven data structures & other

In addition to the general-purpose, hierarchical, data-driven multidimensional data structures (tree-like) presented in previous subsubsection, a wide variety of other data structures are non-hierarchical, space-driven, specially tailored for associative query purposes, and/or external (files too large to be stored in the main memory). Here we briefly mention some selected examples.

[2] The term *quad trees* has also been used often in the literature, for instance, in the original paper of Finkel and Bentley.

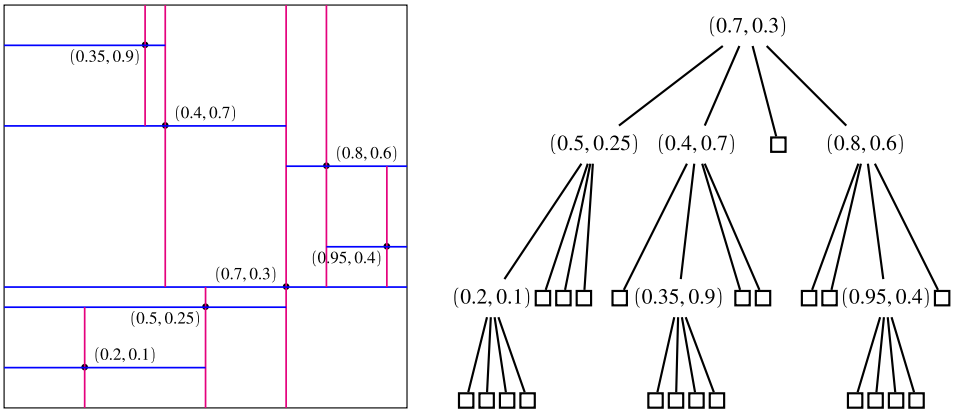

Fig. 3. Example of a quadtree built from2-dimensional points.

2.3.1. Grid files

The *cell* or *fixed grid* method, first introduced by Nievergelt, Hinterberger and Sevcik [65], is a good example of space driven, non-hierarchical data structure. It is based on extendible hashing [72], which divides the space into equal-sized hypercells or buckets. The data structure is a k-dimensional array with an entry per cell that is implemented as a linked list containing the points in it. The space required for this data structure may be super-linear in the number n of records, even if the data is uniformly distributed. In particular, Regnier [71] showed that the average size of the grid file of n k-dimensional records is $\Theta\left(n^{1+(k-1)/(kb+1)}\right)$, where b is the bucket size, and that the average occupancy of the data buckets is approximately 69%.

Most other variants of the non-hierarchical multidimensional data structures are based, inspired or closely related in one form or another to grid files; see [34] for more examples and properties.

2.3.2. Digital search trees

Digital search trees [12], also known as tries, are an example of hierarchical, space-driven data structures. Like hierarchical data-driven data structures, they generate a partition of the search space $[0,1]^k$, but in this case the partition is regular since it depends on subdivisions of the space that are independent of the stored data set. The recursive partition of a region of the search space continues until the region contains one (or none) data points.

The name *digital* for this type of tree refers to the fact that when inserting elements into the tree, the digit representation of the tree records is used. For analysis purposes, the random model may be either binomial or Poisson depending on context.

A typical example of digital trees is the standard k-d tries. Given a randomly generated data set, a k-d trie can be constructed as follows. At level 0, the first digit of the first key is used. If it is a zero, then the insertion process continues in the left sub-trie, and in the right sub-trie otherwise. The first bit of the second key is used at level 1, and so on, until level $k-1$. Then, at level k, we use the second bit of the first key and so on. In general, at level j, we use the $(\lfloor j/k \rfloor + 1)$-st bit of the $(j \bmod k)$-th attribute of the given key. Similar procedure can be carried out to generate relaxed k-d tries and quad tries; see [12,59].

2.3.3. Range trees

Range trees, introduced by Bentley [5], represent a good example of a multidimensional data structure optimized for a specific query. They achieve the best worst-case search time for range search ($\Theta(\log^k n) + R$, for files of n k-dimensional records, where R of them are reported by the range search) among all the structures described so far. Unfortunately, their large preprocessing and storage cost ($\Theta(n \log^{k-1} n)$) makes them infeasible for most applications. Nevertheless, they are still interesting from a theoretical point of view.

3. Partial match & other associative queries

Our use of the term *associative query* encompasses various search operations with constraints, with the exception of *exact search* (where we check for the presence or absence of a record with key $\mathbf{x}$ in the collection and retrieve the associated value if any), which will not be discussed here.

3.1. Types of query

Partial match query. Here the query $\mathbf{q}$ consists of a k-tuple, where each coordinate q_i is a value in D_i or a *wildcard* $q_i = * \notin D_i$. In a partial match, we are interested in retrieving all records with a key $\mathbf{x}$ such that $x_i = q_i$ whenever $q_i \neq *$, and at least one of the coordinates in the query must be *specified* ($q_i \neq *$); otherwise, every element of the collection trivially matches the query, and at least one coordinate must be *unspecified* ($q_i = *$), if not, we are dealing with an exact search. The number of specified coordinates in $\mathbf{q}$ will be denoted by s, $0 < s < k$. In some cases, the performance of a partial match search will depend not only on k, s and the specified values, but also on the particular *pattern* of specified and unspecified coordinates.

For a query $\mathbf{q}$, the *query pattern* is a binary string $\mathbf{u} = \mathbf{u}(\mathbf{q})$ such that $u_i = 1$ if q_i, and $u_i = 0$ if $q_i = *$. For example, for the query $\mathbf{q} = (0.3, *, 0.12, *)$, we have $s = 2$, $k = 4$ and $\mathbf{u} = 1010$.

Although our probabilistic models are not symmetric in dimension, the cost of partial match queries in some data structures such as quadtrees is invariant under permutations of the specified coordinates, provided the query pattern remains the same; for example, the query $\mathbf{q} = (0.3, *, 0.12, *)$ and its permuted version $\mathbf{q} = (0.12, *, 0.3, *)$ lead to the same search cost.

For other data structures, such as standard k-d trees, the cost of the partial match search with query $\mathbf{q} = (0.3, *, 0.12, *)$ ($\mathbf{u} = 1010$) differs from that with query $\mathbf{q} = (0.3, 0.12, *, *)$ ($\mathbf{u} = 1100$).

Finally, there are also data structures (such as quadtrees or relaxed k-d trees) for which the specific query pattern $\mathbf{u}$ is irrelevant; see Section 2.2.

Orthogonal range queries. Given a k-dimensional hyperrectangle with "sides" parallel to the orthogonal axes, the goal is to report the keys inside such hyperrectangle. The query can be easily specified by a collection of k intervals $Q = [\ell_0, u_0] \times [\ell_1, u_1] \times \cdots \times [\ell_{k-1}, u_{k-1}]$, and we are to report all records in the collection with key $\mathbf{x}$ such that $\ell_i \leqslant x_i \leqslant u_i$, for all i, $0 \leqslant i < k$. Sometimes, one or more of the interval ends are not bounded, and thus the query region is not bounded[3] along some dimensions, e.g., we may be interested in all 2-dimensional points such that

[3] Except when the domains D_i are bounded themselves.

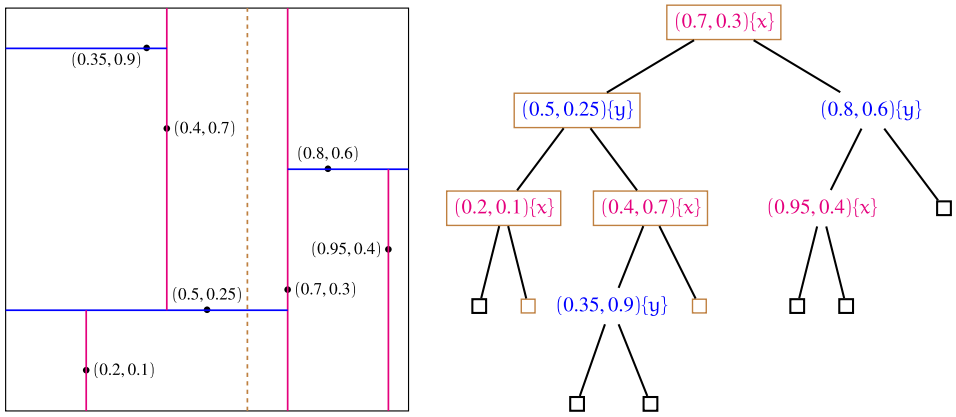

Fig. 4. Example of a partial match in a 2-dimensional tree; the query is $\mathbf{q} = (0.6, *)$ and represented by the brown dashed line.

$a \leqslant x \leqslant b$ and $y \geqslant c$, for given a, b and c; our query would be $[a, b]\times[c, \infty)$ (or $[a, b] \times [c, 1]$ if all keys are confined to the square $[0, 1]^2$).

Region query. Given a region $\mathcal{R} \subset \mathcal{D}$, the task is to report all the elements in $\mathcal{F} \cap \mathcal{R}$. An important special case is when $\mathcal{R}$ is the k-dimensional ball of radius r centered at $\mathbf{q}$, and the purpose of the query is to retrieve all elements in $\mathcal{F}$ whose keys lie inside $\mathcal{R}$; equivalently, all elements in $\mathcal{F}$ at (Euclidean) distance at most r from $\mathbf{q}$.

Nearest neighbor search. The K-nearest neighbor (K-NN, for short) query identifies the K elements in $\mathcal{F}$ that are closest to a given query $\mathbf{q}$. Such a query, as well as *similarity* or *proximity search*, is the subject of many publications due to its usefulness in many applications, ranging from machine learning to geographic information systems. We refer the interested reader to [19,77,87]. Note that the data structures that we discuss in this review (mostly k-dimensional trees, quadtrees and their variants) may not be suitable for high dimensional data, a context often arising in some applications of K-NN queries (e.g., finding images similar to a given one). In addition, a K-NN search might be seen as a region search in which we update and "shrink" the region as the searching algorithm explores the data structure: at any given moment, the current region $\mathcal{R}$ is the smallest ball centered on the query point $\mathbf{q}$ and enclosing the K nearest neighbors found so far.

3.2. Associative queries and bounding boxes

In this review we focus on hierarchical multidimensional data structures, more specifically k-dimensional search trees, k-dimensional quadtrees and several variants of those. In all these cases, we have a well-defined notion of *bounding box* (Definition 5, page 10), the region of the domain $\mathcal{D}$ associated with each internal node and leaf in the tree. For every internal node $\mathbf{x}$, the bounding boxes of its children form a partition of the bounding box of $\mathbf{x}$; applying this recursively, the bounding boxes of all the descendant leaves of $\mathbf{x}$ form a partition of the bounding box of $\mathbf{x}$.

For all the region queries discussed above (with ball queries, orthogonal range queries and partial match queries as particular cases), we can state a very general principle, a fundamental relation between bounding boxes and the performance of these associative queries. The algorithm performing the associative search visits the nodes in the data structure such that their bounding box intersects the region query $\mathcal{R}$, and no others. Fig. 4 shows an example of partial match query in a 2-dimensional search tree. The query, depicted as a brown dashed line, is $\mathbf{q} = (0.6, *)$. None of the points in the collection actually match this query; however, the figure highlights (with brown rectangular frames) the nodes of the 2-d tree that are visited, namely, those whose bounding boxes intersect the query. The region associated with $\mathbf{q}$ is, in this simple example, the vertical line passing through $x = 0.6$. In general, a partial match query

Algorithm 1 Partial match search in k-d trees.

```
▷ Returns nodes in T matching the query q ◁
function PARTIAL_MATCH(T: tree, q: query)
    if T ≠ □ then
        x:=T.key
        if x matches q then Report x
        i:=T.discr
        if q[i] =* then
            PARTIAL_MATCH(T.left, q)
            PARTIAL_MATCH(T.right, q)
        else
            if q[i] ⩽ x[i] then
                PARTIAL_MATCH(T.left, q)
            else
                PARTIAL_MATCH(T.right, q)
```

$\mathbf{q}$ with s specified coordinates out of k, describes a $(k - s)$-dimensional hyperplane embedded in the k-dimensional domain $\mathcal{D}$.

Indeed, imagine that we are conducting a region search for $\mathcal{R}$ and suppose that we are at the root node $\mathbf{x}$ of the tree T. We will check if $\mathbf{x}$ is one of the elements that we will report or not. Then for every subtree T_1, T_2, …of T, we compute whether their respective bounding boxes B_1, B_2, …intersect or not $\mathcal{R}$. If T_i's bounding box intersects $\mathcal{R}$, then T_i might contain elements that belong to $\mathcal{R}$ and thus we will need to recursively explore T_i. This is very general, different associative queries and multidimensional data structures will only differ in the way we traverse the subtrees of T, when we check if the root of T fulfills the query (is inside $\mathcal{R}$ or not), or how we efficiently check whether a given subtree T_i intersects $\mathcal{R}$ or not.

As an example, Algorithm 1 shows the algorithm for partial match search in k-d trees. We assume that every node $\langle \mathbf{x}, i \rangle$ in the given k-d tree T explicitly stores the discriminant; if T points to the root $\langle \mathbf{x}, i \rangle$ then T.key accesses $\mathbf{x}$, T.discr $= i$ and T.left and T.right point to the roots of the left and right subtrees of T, respectively. If the root of T discriminates w.r.t. i then if $q_i = *$ both subtrees can contain data points matching $\mathbf{q}$; on the other hand, if $q_i \neq *$ then we compare q_i against x_i to determine in which subtree we should recursively continue the search.

4. Probabilistic models for the analysis of associative queries

The next section covers in considerable depth the main results and techniques for the analysis of the performance of partial match searches and other associative queries; here, we will briefly discuss the probabilistic models used to carry out the analysis of the performance of the

various associative queries.[4] The focus will be on partial match searches, as the results for other associative queries can be expressed in terms of partial matches [13,30].

As we hinted above, the performance of associative queries is usually measured by the number of visited nodes, or closely related quantities such as the total number of keys $\mathbf{x}$ inspected, or the number of distance evaluations performed. That will be the case in all sections of Part II, where we will measure the cost $\mathcal{P}_{n,\mathbf{q}}$ of a partial match with query $\mathbf{q}$ in a random multidimensional tree that stores n elements, by the number of nodes of the tree visited during the search. This, in turn, coincides with the number of bounding boxes intersected by $\mathbf{q}$.

It is important to mention here that in some cases, instead of the partial match queries $\mathbf{q}$, we will consider their *rank vectors* $\mathbf{r}$. The first explicit use of rank vectors, in this context, that we are aware of, is [27], but using "interchangeably" both relative ranks and i.i.d. random data in $[0,1]$ has been used quite often in the analysis of algorithms.

Definition 8 (Rank vector). Given a k-dimensional key $\mathbf{z} \in \mathcal{D}$ and a collection $\mathcal{F} \subset \mathcal{D}$ of n keys (equivalently, a multidimensional tree T that stores the collection $\mathcal{F}$) the *rank vector* of $\mathbf{z}$, denoted $\mathbf{r} = \mathbf{r}(\mathbf{z}, \mathcal{F})$, is a k-dimensional integer vector $\mathbf{r} = (r_0, r_1, \dots, r_{k-1})$ such that

$$r_i = |\{\mathbf{x} \in \mathcal{F} \mid x_i \leqslant z_i\}|, \qquad 0 \leqslant i < k.$$

Since $|\mathcal{F}| = n$, we have $0 \leqslant r_i \leqslant n$ for all i.

We will often omit $\mathcal{F}$ or T when referring to rank vectors, and leave it implicit. In fact, that implicit collection will be a set of n i.i.d. random data points drawn from $\mathcal{D}$ when we consider the cost as a random variable. If, instead of a key $\mathbf{x}$, we have a partial match query $\mathbf{q} = (q_0, \dots, q_{k-1})$ we will define $\mathbf{r}(\mathbf{q})$ with r_i defined like above (just using q_i instead of z_i) if $q_i \neq *$, and $r_i = *$ if $q_i = *$.

As mentioned, we will sometimes analyze $\mathcal{P}_{n,\mathbf{r}}$, the cost of a partial match search in a random multidimensional tree of size n when the rank vector of the query is $\mathbf{r}$, instead of $\mathcal{P}_{n,\mathbf{q}}$. Partial match searches with identical rank vectors will visit exactly the same nodes in the tree; moreover, we can easily translate results from $\mathsf{P}_{n,\mathbf{q}}$ to $\mathsf{P}_{n,\mathbf{r}}$ and vice versa:

$$\mathsf{P}_{n,\mathbf{q}} = \sum_{\mathbf{r}} \mathbb{P}\left[\mathbf{r}(\mathbf{q}) = \mathbf{r}\right] \cdot \mathsf{P}_{n,\mathbf{r}}.$$

When the data is uniformly distributed, this entails

$$\mathsf{P}_{n,\mathbf{q}} = \mathsf{P}_{n,\bar{\mathbf{r}}} + \text{l.o.t.},$$

$$\mathsf{P}_{n,\mathbf{r}} = \mathsf{P}_{n,\bar{\mathbf{q}}} + \text{l.o.t.},$$

where $\bar{r}_i = nq_i$ if $q_i \neq *$, and $\bar{r}_i = *$ if $q_i = *$, and similarly, $\bar{q}_i = r_i/n$. It is also important to mention here that further simplifications can be made in the analysis of some multidimensional trees, thanks to additional symmetries of the data structures and the probability model generating the data. For example, for quadtrees and several variants of k-d trees (but not standard k-d trees!) built from random uniform data, we can assume w.l.o.g. that the query has the pattern $\mathbf{u}(\mathbf{q}) = 1^s \cdot 0^{k-s}$. A few observations are in order:

1. If two partial match queries $\mathbf{q}$ and $\mathbf{q}'$ have the same rank vector, $\mathbf{r}(\mathbf{q}) = \mathbf{r}(\mathbf{q}')$, then $\mathcal{P}_{n,\mathbf{q}} = \mathcal{P}_{n,\mathbf{q}'}$. This is because the partial match search does not depend on the actual values of the coordinates of the keys, but on the *relative ranks*. For example, if the root of k-d tree is $\langle \mathbf{x}, i \rangle$ and q_i is specified, then we will continue recursively in the left subtree if and only if $q_i \leqslant x_i$, that is, if and only if the rank of q_i is smaller than or equal to the rank of x_i. Therefore, since we assume $r_i = r'_i$, the partial match search algorithm will continue visiting the same subtree for both queries, and hence visit the same number of nodes.
2. Since we assume that the n data points of a random multidimensional tree are generated by drawing each of its k coordinates i.i.d. from some continuous distribution F_i in $\mathcal{D}_i = [0,1]$ (Definition 2), the rank vector of a query $\mathbf{q}$ will be $r_i = *$ whenever $q_i = *$ and r_i will be a uniform integer in $\{0, \dots, n\}$ whenever $q_i \neq *$.
3. Results for $\mathsf{P}_{n,\mathbf{q}} = \mathbb{E}[\mathcal{P}_{n,\mathbf{q}}]$ can be derived from the results for $\mathsf{P}_{n,\mathbf{r}} = \mathbb{E}\left[\mathcal{P}_{n,\mathbf{r}}\right]$, by conditioning

$$\mathsf{P}_{n,\mathbf{q}} = \sum_{\mathbf{r}} \mathbb{P}\left[\mathbf{r}(\mathbf{q}) = \mathbf{r}\right] \cdot \mathsf{P}_{n,\mathbf{r}}.$$

   Moreover, under our stronger assumption that the data points in $\mathcal{F}$ are generated by producing each x_i independently and uniformly at random in $[0,1]$, it follows that r_i is given by a binomial $\mathrm{Bin}(n, q_i)$ random variable when $q_i \neq *$, and in view of the smooth behavior of $\mathsf{P}_{n,\mathbf{r}} = \mathbb{E}\left[\mathcal{P}_{n,\mathbf{r}}\right]$, we have

$$\mathsf{P}_{n,\mathbf{q}} = \mathsf{P}_{n,\bar{\mathbf{r}}} + \text{l.o.t.},$$

   where $\bar{r}_i = nq_i$ if $q_i \neq *$. Similarly, for a fixed rank vector $\mathbf{r}$, we will have

$$\mathsf{P}_{n,\mathbf{r}} = \mathsf{P}_{n,\bar{\mathbf{q}}} + \text{l.o.t.}$$

   with $\bar{q}_i = r_i/n$ if $r_i \neq *$.
4. If the probabilistic model for the data points does not introduce any spatial asymmetry (as is the case for the usual model in Definition 2), and given that our multidimensional data structures do not exhibit such asymmetries either, $\mathcal{P}_{n,\mathbf{r}} = \mathcal{P}_{n,\mathbf{q}'}$ for any two partial match queries which can be obtained by permutation of the specified coordinates. For example, if $k = 5$ the cost of a partial match with $\mathbf{q} = (*, 0.1, 0.72, *, 0.6)$ will have exactly the same distribution as the cost of a partial match search with $\mathbf{q}' = (*, 0.72, 0.6, *, 0.1)$ (actually, the same can be said for the $s! = 6$ possible queries that we obtain by permutation of the $s = 3$ specified coordinates in $\mathbf{q}$). Indeed, the property is true if all coordinates of all data points are generated independently and uniformly at random in $[0,1]$, but it is also true if every coordinate of every data point is independently generated from some continuous distribution F in $[0,1]$ (the same distribution F for all k coordinates).

   On the other hand, some of our data structures impose some order across the k dimensions of the space when inducing a partition of $\mathcal{D} = [0,1]^k$; that is the case for standard k-d trees, but it is not the case with other variants of k-d trees like the relaxed k-d trees that pick a random coordinate to discriminate at each node, or quadtrees that partition the current bounding box w.r.t. all coordinates at each node (see Section 2). In the former case, the performance of partial matches will depend not only on the actual values of the specified coordinates, but also on the particular pattern of specified and unspecified coordinates. Thus, for example, the cost $\mathcal{P}_{n,\mathbf{q}}$ of a partial match with the query $\mathbf{q} = (*, 0.1, 0.1)$ does not have the same distribution as $\mathcal{P}_{n,\mathbf{q}'}$ with the query $\mathbf{q}' =$ (0.1,*,0.1); however, the random variable will have the same distribution for both queries if we consider the partial match in a random relaxed k-d tree or a quadtree.

In addition to partial matches with a fixed query $\mathbf{q}$ (or rank vector $\mathbf{r}$) we are also interested in partial match searches with a random query $\mathbf{Q}$. Historically, this was the first kind of partial match search for which precise average-case analysis was conducted; a more detailed account of this historical development is given in Section 5. In random partial matches, we fix a pattern of specified and unspecified coordinates $\mathbf{u} = u_0 \cdots u_{k-1}$

[4] In many cases, either because too many points satisfies the query, or because our data structures do not enforce any kind of strong balancing, the worst-case of an associative query will be linear, requiring (almost) full exploration of the collection of data points—just as a linear scan of the collection, checking for every data point whether it satisfies the query or not.

with $u_i = 1$ (meaning that coordinate i is specified) or $u_i = 0$ (meaning it is unspecified). The specified coordinates are generated according to the same distribution as the coordinates of the data points (if we assume they are independently generated), or we draw a data point independently of the elements in $\mathcal{F}$ according to the same distribution as the elements in $\mathcal{F}$ and then we "mask" the unspecified coordinates, replacing them with wildcards $*$.

For random partial matches we are thus dealing with a new random variable $\hat{P}_{n,\mathbf{u}}$ where we have a random multidimensional tree with n data points and a random query with fixed pattern $\mathbf{u}$.

Because of our discussion above, when the probabilistic model for data (and query) generation is symmetric, for example, when each specified coordinate is i.i.d. uniform on $[0, 1]$ and no dimension is privileged when creating the partition of the space, then the actual pattern of specified and unspecified coordinates may become irrelevant. In this case, the cost of a partial match search depends only on n, k and s, *the number of specified coordinates* in the query, as well as on the specific data structure that organizes the collection of data points. Accordingly, we will use $\hat{P}_{n,s}$ (or just $\hat{P}_n$) to emphasize that the actual pattern of specified and unspecified coordinates does not matter, and keep $\hat{P}_{n,\mathbf{u}}$ or $\hat{P}_{n,\mathbf{u}}$ for the situations where the specific pattern is relevant, like in standard k-d trees (see, for example, Theorem 10 in Section 6).

Last but not the least, the analysis of orthogonal range searches and more general region queries follows steps analogous to those for partial match searches. As before, we measure the cost in terms of the number of visited nodes in the multidimensional tree storing the collection, and we distinguish between queries with a fixed query object (hyperrectangle, ball, …) and random query objects. In the latter case, we are mostly interested in queries of fixed "size" and random location. For example, in a random ball query we fix the radius α and choose the center at random according to the same distribution as the data points. In a random orthogonal range query we fix the lengths $\Delta_0, \ldots, \Delta_{k-1}$ of the intervals (the sides of the hyperrectangle), and choose at random (again with the same distribution as for the data points) either the center of the hyperrectangle or one of its corners.

Although in principle the analysis applies to random queries of arbitrary size, the most interesting cases are those where the query size is "small", in particular, when the number of data points satisfying the query is sublinear. For instance, if the data points are generated uniformly at random and the "volume" of the query region is $\mathcal{O}(f(n)/n) = o(1)$, then we expect to report $\mathcal{O}(f(n)) = o(n)$ points, and the expected cost of the associative query can be decomposed as $R+W$, where $R = \mathcal{O}(f(n))$ is the expected number of reported points and W is the *overhead* of the operation. The cost R is unavoidable, since we must report the points that satisfy the query, but in principle the overhead W could be avoided, and we would like it to be as small as possible.

It is also important to note that in random associative queries the query region may lie partially outside $\mathcal{D}$. Trying to exclude these situations would introduce various undesirable mathematical complications and make the analysis less elegant. Moreover, when we restrict attention to "small" queries, there is no compelling reason to impose that the query regions lie entirely within $\mathcal{D}$: the probability that a random query region intersects the boundary of D then becomes negligible.

Part II: The Analysis of Partial Match Queries

5. A historical account

The introduction of new, efficient data structures often demands a more precise analysis of their performance. This analysis, in turn, leads to methodological challenges and advances, which motivate the evolution of new data structures or algorithms, particularly when these advances prove applicable to other structures. The historical developments in the analysis of multidimensional data structures, which are briefly reviewed in this section, are testimony to such a harmonious interplay between the design and analysis of algorithms.

Flajolet and Puech, largely motivated by the significant progress made in the theory and application of computational geometry and particularly in multidimensional data structures in the 1970s, were the first to analyze in great detail in [44] the asymptotic efficiency of k-d trees, k-d tries and grid-files. As the authors state in their abstract:

> The methods used include a detailed study of a differential system around a regular singular point in conjunction with suitable contour integration techniques for the analysis of k-d trees, and properties of the Mellin integral transform for k-d tries and extendible cell algorithms.

The various technical terms mentioned here already indicate the complexity of the underlying analytic problems. Using these complex analytic tools, the authors were the first to derive the average cost of random partial match queries with a specified pattern u, which satisfies asymptotically.

random k-d trees	random k-d tries
$\beta_u n^{1-s/k+\theta(s/k)}$	$\beta_u \left(\log n^{1/k}\right) n^{1-s/k}$
$\beta_u, \theta(t) > 0$	$\beta_u(t)$: periodic

Similar to the local balancing schemes used for improving the performance of quicksort or binary search trees, the median-of-$(2t + 1)$ variant of k-d trees, called k-d-t trees, was introduced by Cunto, Lau and Flajolet [17], as already mentioned above. These trees are locally reorganized to maintain balance: $t = 0$ implies no reorganization (which corresponds to standard k-d trees), and any subtree of size $\geqslant 2t$ has at least t elements in each of its two subtrees. In addition to reducing the occurrence of skinny bounding boxes (as shown in Fig. 1), these trees also offer the possibility of increasing the sample size $2t+1$ to reach the asymptotic optimality (as exhibited by a completely balanced tree) for the expected cost. In particular, the expected cost for a random partial match query in random k-d-t trees of n elements satisfies asymptotically the same pattern as that for k-d trees:

$$\beta n^{\alpha}, \tag{1}$$

with $\alpha := 1 - s/k + \theta(s, k, t) > 1 - s/k$.

Such a pattern arises in a wide family of multidimensional trees such as relaxed k-d trees and quadtrees among others, each with a different θ. The leading constant β in the cases of quadtrees, k-d trees, and k-d-t trees was later fully characterized by Chern and Hwang [14,15] more than a decade after α had been derived. These constants are calculated using more advanced analytic combinatorial techniques that rely on asymptotics for differential equations with polynomial coefficients, Mellin transforms, binomial transforms, and singularity analysis; see Section 6 for more details.

Although the characterization of the leading constant β is a highly non-trivial problem in the case of quadtrees and k-d trees, a simplified variant called *relaxed k-d trees*, proposed in [23], greatly simplifies the analysis, and both α and β for the expected cost of the partial match queries can be fully computed by solving second-order ordinary differential equations of second order; see [60].

Table 2 provides a timeline of the advancements concerning the expected cost of random partial match queries in hierarchical multidimensional data structures, particularly the calculations of α and β and Table 3 summarizes them all.

Several papers (e.g., [60,64]) reported results on the variance of the cost and limit law of a random partial match query, but these actually correspond to an idealized, simplified model where randomness and independence of the query are preserved at each stage. For further discussion, see the end of Section 7.8; see also [55, Section 2.3]. This problem has finally been resolved for the 2-dimensional case; see Corollary 17.

Table 2
Timeline for the analysis of random partial match queries.

Year	Results	Data structures
1986	α	k-d trees and k-d tries (Flajolet and Puech) [44]
1989	α	k-d-t trees (Cunto et al.) [17]
1993	α	quad trees (Flajolet et al.) [40]
1995	extreme points	quad trees (Flajolet et al.) [42]
	variance	k-d tries (Schachinger) [78]
1998	α, β	relaxed k-d trees (Duch et al.) [23]
2000	α	squarish k-d trees (Devroye et al.) [13]
	limit law	k-d tries, k = 2 (Schachinger) [79]
2001	mean & var.	relaxed k-d trees and k-d tries (Martínez et al.) [60]
2003	β	quad trees (Chern and Hwang) [14]
2004	limit law	asymmetric k-d tries (Schachinger) [80]
2006	β	k-d trees (Chern and Hwang) [15]
2013	var. & limit law	quad trees, k-d trees, k = 2 (Broutin et al.) [11]
2016	α	quad k-d trees (Duch et al.) [28]
2017	α	relaxed k-d-t trees (Duch and Lau) [26]
2022	α	median & hybrid median k-d trees (Duch et al.) [33]
2023	β	quad k-d trees (Duch, Martínez) [32]

Table 3
State of the art analysis of the average cost of random partial match queries.

Data structure	α		β	
standard k-d trees	✓	[44]	✓	[15]
standard k-d-t trees	✓	[17]	✓	[15]
relaxed k-d trees	✓	[23]	✓	[23]
squarish k-d trees	✓	[13]	×	
relaxed k-d-t trees	✓	[26]	×	
median k-d trees	✓	[33]	×	
hybrid median k-d trees	✓	[33]	×	
quadtrees	✓	[40]	✓	[14]
quad k-d trees	✓	[28]	✓	[32]

Also noteworthy of mention are squarish k-d trees [25], which significantly reduce elongated rectangles and achieve asymptotically the optimal $\Theta(n^{1-s/k})$ expected cost for partial match queries. The approach is probabilistic and the leading constant remains an open problem (see Section 7).

Along a different line, Roura's Continuous Master Theorem (explained in Section 8.1) provides another useful real-analytic technique that is often very effective in obtaining the dominant order n^{α} for a wide variety of multidimensional data structures. Moreover, for some classes of data structures (e.g., the median and hybrid median k-d trees [33]), it has been, until now, the only feasible method to analyze partial match queries. The subtlety that the randomness between the random query pattern and the random tree to conduct partial match search is not preserved when the search is performed recursively in the subtrees was first reported in [16] (see also [11]). This opens another chapter on the analysis of *fixed* partial match queries; see Table 4 for the developments in a timeline rendering. In addition to the expected cost, there are also rigorous results that establish the convergence in distribution in the two-dimensional case; see [11,16]. Roughly, for the cost of such fixed patterns, the asymptotic expected cost may differ from the pattern βn^{α} that we have observed up to now. See Section 7 for more details on the results and techniques involved.

In what follows, we illustrate the main mathematical techniques that have been put into service, through a few examples.

6. Analytic methods

Critical to most precise analysis of the cost of partial match queries (and other cost measures) in the given structures is the availability of *recursive decomposition* of the data structures or the underlying algorithmic procedures, which often generate recurrence relations whose natures range from linear to nonlinear, from holonomic to full-history, from multidimensional to multivariate, etc. Indeed, recursion is omnipresent in computer algorithms. Diverse resolution methods for recurrence relations have been developed, varying according to the underlying techniques employed: algebraic, combinatorial, probabilistic, real-analytic, complex-analytic, etc. For simplicity of presentation, we will group them into *complex-analytic methods* (relying on complex analysis) and *real-analytic methods* (without resorting to complex analysis).

While the adoption of real-analytic methods is often more natural, readers unfamiliar with complex analysis may wonder why such an analysis is used in solving problems whose nature and formulation have nothing to do with it? The simplest insight comes from the fact that manipulation of the imaginary powers e^{it} is in almost all cases much easier than the corresponding real trigonometric arithmetic. Further insight comes from the Fast Fourier Transform (transforming between time and frequency domains, selected to be among the top ten algorithms of the last century with the greatest influence in science and engineering; see [18]), which is indispensable in many engineering and daily-life applications, notably those dealing with signals and images. A famous quote by Jacques Hadamard also adds an implicit philosophical flavor [50, p. 123]:

> …the shortest and best way between two truths of the real domain often passes through the imaginary one.
>
> (…la voie la plus courte et la meilleure entre deux vérités du domaine réel passe souvent par le domaine imaginaire.)

We discuss complex-analytic methods in this section, with other methods being described later in Sections 7 and 8.

In general, except for the rare lucky cases where the problem can be solved by direct enumeration, the *generating function* of the cost measure is the key to all analytic methods, whether real-analytic or complex-analytic. Quoted from the preface of Flajolet and Sedgewick's authoritative book [46]:

> Analytic combinatorics aims to enable precise quantitative predictions of the properties of large combinatorial structures. …With a careful combination of symbolic enumeration methods and complex analysis, drawing heavily on generating functions, results of sweeping generality emerge that can be applied in particular to fundamental structures such as permutations, sequences, strings, walks, paths, trees, graphs and maps.

We focus on tree structures in this paper as they represent the most widely used ones that are mathematically tractable, and many general tools have been developed in the literature to characterize their algorithmic efficiency, including particularly partial match queries. Note that

Table 4
Timeline for the analysis of fixed partial match queries.

Year	Results	Data structures
2011	convergence and h($\mathbf{q}$)	2-dimensional quad trees (Curien and Joseph) [16]
2012	convergence and h($\mathbf{q}$) when s = 1	k-d trees (Duch et al.) [24]
2013	variance, limit law	quad trees, k-d trees in 2 dim. (Broutin et al.) [11]
2016	convergence and h($\mathbf{q}$)	k-d trees (Duch et al.) [27]
2017	implicit characterization of h($\mathbf{q}$)	relaxed k-d-t trees (Duch and Lau) [26]
2024	h($\mathbf{q}$)	quad k-d trees (Duch and Martínez) [32]

the asymptotic performance of most of the partial match queries is at most of polynomial growth in tree size (queries at a higher cost being less useful in practice).

Depending on the way data are compared and directed to each subtree, we distinguish between *digital tree models* where input data are compared on a bitwise basis (space-driven structures), and *key comparison (or non-digital) tree models* where data are directly compared without referring to their digital structures (data-driven structures).

Analytic tools and methodologies developed for these models differ, although they often share common analytic features.

- Digital tree models: generating functions lead mostly to algebraic or functional equations whose resolution (exact or asymptotic) often relies on tools based on finite differences and Mellin transforms (see [37,45]):

  Recurrence → Poisson gen. fun. → Algebraic eq. → Mellin transf.

- Non-digital tree models: differential equations or systems (linear or non-linear, ordinary or partial) abound in such models, and most asymptotic analysis relies heavily on complex analysis, notably singularity analysis (see [46,56]):

  Recurrence → Ordinary gen. fun. → Differential eq. → Singularity ana.

6.1. Asymptotics of the expected cost

Proposed then, and widely used since the 1970s (see [4,36,77]), multidimensional search trees gradually received more attention in the 1980s as far as their major cost measures are concerned, which include depth, height, range search, partial match queries, etc., and more asymptotic and probabilistic tools were introduced and developed. Significant advances were made by Devroye, whose approach is mostly *probabilistic* (including branching processes), and Flajolet, who followed a completely different line and initiated the *analytic-combinatorial* study of multidimensional search trees. In particular, the expected cost $\hat{\mathsf{P}}_n = \mathbb{E}\left[\hat{P}_n\right]$ was shown to be of the form (see [40,44])

$$\hat{\mathsf{P}}_n \sim \beta n^{\alpha},$$

for some constant $\beta > 0$, where $\alpha \in (0,1)$ solves the equation:

$$(z+1)^{1-\rho}(z+2)^{\rho} = 2, \tag{2}$$

with $\rho := \frac{s}{k}$. This in contrast to previous belief that $\hat{\mathsf{P}}_n \approx n^{1-\rho}$, as being argued under fully-balanced trees; see [4,40]. Note that, by Lagrange inversion formula, we can express α in terms of the powers of ρ:

$$\alpha = 1 - \sum_{j\geqslant 1} d_j \rho^j, \quad \text{with} \quad d_j = \frac{1}{j}[y^{j-1}]\left(\frac{\log\frac{2-y}{3-y}}{\frac{1}{y}\log(1-\frac{y}{2})}\right)^j > 0,$$

for $j = 1, 2, \ldots$. Thus $\alpha > 1 - \rho$.

The approach used in [40,41,44] begins by first constructing a linear system (of differential equations) for the generating function of the cost, which is then solved using singularity perturbation techniques; see also [46] for singularity analysis of generating functions.

Such an analytic approach is very powerful and general, yet as is typical in the asymptotic resolution of linear systems, characterization of the leading constant C remains undetermined in both cases (k-d trees and quadtrees). The same results were later obtained by Devroye, Jabbour and Zamora-Cura [25], again with an unknown β.

A full characterization of the leading constant β (as well as a finer asymptotic expansion if desired) was later given in [14,15] with completely different approaches. We now describe the expressions in both cases and then sketch the methods of proof.

6.2. Partial match queries in random quadtrees

Theorem 9. *([14]) The expected cost $\hat{\mathsf{P}}_n = \mathbb{E}\left[\hat{P}_n\right]$ of a random partial match query with s specified coordinates in a random quadtree of n nodes satisfies*

$$\hat{\mathsf{P}}_n \sim \beta_{k,s} n^{\alpha}, \tag{3}$$

where

$$\beta_{k,s} := \frac{1}{(2^{k-s}-1)\Gamma(\alpha+1)^{k-s}\Gamma(\alpha+2)^{s}} \prod_{2\leqslant j\leqslant k} \frac{\Gamma(\alpha-\alpha_j)}{\Gamma(-\alpha_j)},$$

for $1 \leqslant s < k$ and $k \geqslant 2$, Γ is the Gamma function and the α_j's are the zeros of the polynomial $(z+1)^{k-s}(z+2)^{s} = 2^k$, and

$$\alpha = \alpha_1 > \Re(\alpha_2) \geqslant \cdots \geqslant \Re(\alpha_k).$$

The approaches used in [40] and in [14] both rely on the recurrence satisfied by $\hat{\mathsf{P}}_n$:

$$\hat{\mathsf{P}}_n = 1 + 2^k \sum_{1\leqslant j<n} \pi_{n,j}[j] \qquad (n \geqslant 0), \tag{4}$$

with $\hat{\mathsf{P}}_0 = 0$, where

$$\pi_{n,j} = \binom{n-1}{j} \int_{(0,1)^k} x_0\cdots x_{s-1}(x_0\cdots x_{k-1})^j(1-x_0\cdots x_{k-1})^{n-1-j}\mathrm{d}\mathbf{x}. \tag{5}$$

Note that the leading constant β is independent of the query pattern but depends only on the number of specified coordinates s.

Here is a quick heuristic reasoning to identify the growth order n^{α}. Assume the Ansatz: $\hat{\mathsf{P}}_n \sim \beta n^{\delta}$ for some $\beta, \delta > 0$. Then the RHS of the recurrence (4) is asymptotic to

$$\beta 2^k \int_{(0,1)^k} x_0\cdots x_{s-1} \underbrace{\sum_{1\leqslant j<n}\binom{n-1}{j} j^{\delta}(x_0\cdots x_{k-1})^j(1-x_0\cdots x_{k-1})^{n-1-j}}_{=\mathbb{E}\left[\mathrm{Binomial}(n-1;x_0\cdots x_{k-1})^{\delta}\right]\sim(x_0\cdots x_{k-1})^{\delta}n^{\delta}} \mathrm{d}\mathbf{x}$$
$$\approx \beta 2^k n^{\delta}\int_{(0,1)^k} x_0\cdots x_{s-1}(x_0\cdots x_{k-1})^{\delta}\mathrm{d}\mathbf{x} = \frac{\beta 2^k n^{\delta}}{(\delta+1)^{k-s}(\delta+2)^{s}}.$$

Equating both sides of the recurrence (4), we see that δ solves the Eq. (2). This back-of-the-envelope calculation leaves β undetermined.

For a rigorous proof, a crucial step then is to show that the binomial transform of $\hat{\mathsf{P}}_n$:

$$\hat{\mathsf{P}}^*_n := \sum_{1\leqslant j\leqslant n}\binom{n}{j}(-1)^j\hat{\mathsf{P}}^*_j \qquad (n \geqslant 1)$$

satisfies the surprisingly simple recurrence (see [14])

$$\hat{\mathsf{P}}^*_n = \left(1 - \frac{2^k}{n^{k-s}(n+1)^{s}}\right)\hat{\mathsf{P}}^*_{n-1} \qquad (n \geqslant 2),$$

with $\hat{\mathsf{P}}^*_1 = -1$. This then leads, by inverting the binomial transform, to the *exact solution* of the recurrence (4):

$$\hat{\mathsf{P}}_n = \sum_{1\leqslant j\leqslant n}\binom{n}{j}(-1)^{j+1}\prod_{2\leqslant i\leqslant j}\left(1 - \frac{2^k}{i^{k-s}(i+1)^{s}}\right).$$

Expressing the inner product in terms of the zeros α_j, one then obtains the asymptotic cost (3) using Stirling's formula.

Two other approaches with some generality based on generating functions and integral transforms are also presented in [14].

6.3. Partial match queries in random k-d trees

The situation of k-d trees is more involved, and one is naturally led to differential equations with polynomial coefficients satisfied by the generating function of the expected cost; such equations or the underlying cost sequences are referred to as holonomic or D-finite in the combinatorial literature. The approach developed in [15] is very general and applies in its full generality to almost all holonomic sequences.

Let $\mathbf{u} = (u_0, \dots, u_{k-1}) \in [0,1]^k$ be the pattern of the random partial match query (recall that $u_j = 1$ means that the jth coordinate is specified and $u_j = 0$ otherwise). For $u \in \{0,1\}$, define $\pi_{n,j}(u) := \frac{1+uj}{n(1+un)}$. Then the expected cost $\hat{\mathsf{P}}_n$ of a partial match query in a random standard k-d tree of n nodes satisfies the recurrence:

$$\begin{aligned}\hat{\mathsf{P}}_n = 1 &+ \sum_{1\leqslant i<k} \sum_{1\leqslant j_i<\cdots<j_1<n} \pi_{n,j_1}(q_0)\cdots\pi_{j_{i-1},j_i}(q_i) \\ &+ 2^k \sum_{1\leqslant j_k<\cdots<j_1<n} \pi_{n,j_1}(q_0)\cdots\pi_{j_{k-1},j_k}(u_{k-1})\hat{\mathsf{P}}_{j_k},\end{aligned} \tag{6}$$

for $n \geqslant 1$ with $\hat{\mathsf{P}}_0 = 0$. Assume the Ansatz $\hat{\mathsf{P}}_n \sim \beta n^\delta$ for some $\delta, \beta > 0$. Note that $\pi_{n,j}(0) = \frac{1}{n}$, and $\pi_{n,j}(1) = \frac{j}{n(n+1)}$. Thus,

$$\beta \sum_{1\leqslant j<n} \pi_{n,j}(u) j^\delta \sim \beta n^\delta \times \begin{cases} \frac{\beta}{\delta+1}, & \text{if } u = 0; \\ \frac{\beta}{\delta+2}, & \text{if } u = 1. \end{cases}$$

Consequently, we see that the RHS of the recurrence (6) is asymptotic to

$$2^k \sum_{1\leqslant j_k<\cdots<j_1<n} \pi_{n,j_1}(u_0)\cdots\pi_{j_{k-1},j_k}(u_{k-1})\hat{\mathsf{P}}_{j_k} \approx \frac{\beta 2^k}{(\delta+1)^{k-s}(\delta+2)^s} n^\delta.$$

Again $\delta = \alpha$ is chosen to be the zero of the Eq.(2). However, such a heuristic calculation does not provide further insight for β.

The approach developed in [15] translates the recurrence (6) into the following (holonomic) differential equation with polynomial coefficients for the corresponding generating function f (with $\vartheta := z(\mathrm{d}/\mathrm{d}z)$):

$$\sum_{0\leqslant j\leqslant d} (1-z)^j P_j(\vartheta) f = g,$$

for some d depending on the pattern $\mathbf{u}$, where in particular $P_0(x) = x^{k-s}(x+1)^s - 2^k$ and $g = c/(1-z) + p(z)$ with p a polynomial. Define

$$G(s) = \frac{c}{s-1} + \sum_{0\leqslant \ell\leqslant k} \frac{(-1)^\ell p^{(\ell)}(1)}{\ell!(s+\ell)}.$$

Theorem 10. *([15]) The expected cost $\hat{\mathsf{P}}_n$ of a partial match query in a random standard k-d tree tree of n nodes satisfies*

$$\hat{\mathsf{P}}_n \sim \beta_{\mathbf{u}} n^\alpha,$$

with again $\alpha > 0$ solving the Eq. (2). Here

$$\beta_{\mathbf{u}} := \sum_{j\geqslant 1} G(j+\alpha) B_j, \quad \text{with} \quad B_j = \begin{cases} 0, & \text{if } j \leqslant 0; \\ 1, & \text{if } j = 1; \\ -\sum_{1\leqslant i\leqslant \mu} \frac{P_i(j+\alpha)}{P_0(j+\alpha)} B_{j-i}, & \text{if } j \geqslant 2, \end{cases}$$

the series being absolutely convergent.

While all expressions are rather complicated, they are effectively computable; also in special cases they lead to simpler solutions. For asymptotic expressions for the variance and a limit law of $\hat{\mathsf{P}}_n$ in the 2-dimensional case, see Corollary 17.

The Ph.D. Thesis of Lau [55] contains a more up-to-date account of recent developments in the analysis of partial match queries in multidimensional search trees; see also the Ph.D. Thesis of Duch [21] for an earlier account.

7. Probabilistic methods

We now review probabilistic tools and analysis for the complexity of partial match queries for the case of 2-dimensional quadtrees. Advances have been made by various probabilistic tools:

- to extend the analysis of $\hat{\mathcal{P}}_n$ of Section 6 beyond its expectation in Theorem 9 toward an understanding of the variance and asymptotic distribution of $\hat{\mathcal{P}}_n$, or
- to study the complexity for a fixed query or the worst case complexity over all possible queries.

We denote the complexity (i.e., the number of nodes visited) of a fixed partial match query in the random 2-dimensional quadtree built from n i.i.d. data uniformly distributed over $[0,1]^2$ by $\hat{\mathcal{P}}_n(q) := \hat{\mathcal{P}}_{n,\mathbf{q}}$ with $\mathbf{q} = (q, *)$ where $q \in [0,1]$. The basis of a refined probabilistic analysis is to consider the $\hat{\mathcal{P}}_n(q)$ jointly for all $q \in [0,1]$, i.e., as a stochastic process $(\hat{\mathcal{P}}_n(q))_{q\in[0,1]}$ and to use a distributional recurrence relation for this process. To this end we first sketch the setting of the function space and weak convergence in Sections 7.1 and 7.2 before we then start to derive and work with distributional recurrence relations.

Results similar to those presented in this section for standard 2-dimensional trees can be found in [10,11].

7.1. The partial match complexities as random càdlàg functions

We consider the $\hat{\mathcal{P}}_n(q)$ simultaneously for all $q \in [0,1]$ as a stochastic process $(\hat{\mathcal{P}}_n(q))_{q\in[0,1]}$, which has piecewise constant paths. An appropriate space for its paths is the space $D[0,1]$ consisting of all functions $x : [0,1] \to \mathbb{R}$ having left limits and being right-continuous, i.e., such that

$$\lim_{s\uparrow q} x(s) \text{ exists for all } q \in [0,1) \quad \text{and} \quad \lim_{s\downarrow q} x(s) = x(q) \text{ for all } q \in [0,1).$$

Such functions are called *càdlàg* (continue à droite, limites à gauche). Measuring closeness of functions $f, g \in D[0,1]$ can be done within the uniform norm $\|f - g\|$, which is however not suitable in the context of partial match queries due to measurability issues. It is more flexible to use the Skorokhod metric

$$d_{\mathrm{SK}}(f,g) = \inf_\lambda \max\{\|f\circ\lambda - g\|, \|\lambda - \mathrm{id}\|\},$$

where the infimum is taken over all increasing bijections $\lambda : [0,1] \to [0,1]$ and id denotes identity. We consider $(\hat{\mathcal{P}}_n(q))_{q\in[0,1]}$ as a stochastic process in the complete metric space $(D[0,1], d_{\mathrm{SK}})$.

7.2. Weak convergence on $(D[0,1], d_{SK})$

For a fine distributional analysis of the complexities $\hat{\mathcal{P}}_n(q)$, we first look for a normalization $\mathcal{Z}_n(q)$ such that the normalized process $\mathcal{Z}_n = (\mathcal{Z}_n(q))_{q\in[0,1]}$ converges in distribution toward a non-degenerate limit process $\mathcal{Z} = (\mathcal{Z}(q))_{q\in[0,1]}$; see Theorem 14 below. Such a result $\mathcal{Z}_n \to \mathcal{Z}$ in distribution within $(D[0,1], d_{\mathrm{SK}})$, called a *functional limit theorem*, has far reaching implications due to the (continuous) mapping theorem [8, Theorem 2.7]:

> For any metric space (M, ϱ) and any continuous function $\phi : D[0,1] \to M$, we obtain the convergence $\phi(\mathcal{Z}_n) \to \phi(\mathcal{Z})$ in distribution.

This even holds for discontinuous (measurable) functions ϕ if the set D_ϕ of discontinuities of ϕ satisfies $\mathbb{P}(\mathcal{Z} \in D_\phi) = 0$. This will be exploited in the example of partial match below for various appropriate such functions ϕ.

The classical theory of weak convergence of probability distributions on metric spaces in general and for $(D[0,1], d_{\mathrm{SK}})$ in particular is presented in [8]. In the classical theory weak convergence of distributions on $(D[0,1], d_{\mathrm{SK}})$ is obtained by convergence of the finite dimensional marginals and tightness, which is usually shown by studying the modulus of continuity and using the Arzelà–Ascoli theorem; see,

e.g., Billingsley [8, Theorem 13.2]. However, in the present context of partial match the classical approach has not yet been successfully carried out. Instead, the following recursive distributional decomposition of $(\mathcal{Z}_n(q))_{q\in[0,1]}$ is the basis of the asymptotic distributional analysis.

7.3. Recursive decomposition of the partial match process

We denote the numbers of data points (for our 2-dimensional quadtree) that fall into each of the four rectangles generated by the first point $(U_1, V_1) =: (U, V)$ by $I^{(n)} = (I_1^{(n)}, I_2^{(n)}, I_3^{(n)}, I_4^{(n)})$, where n is the total number of data. Conditional on (U, V), the $I^{(n)}$ is multinomially $M(n-1; UV, U(1-V), (1-U)V, (1-U)(1-V))$ distributed, using a numbering of the four quadrants. Conditional on (U, V) and $I^{(n)}$ each point set within a rectangle is a set of i.i.d. points with the uniform distribution on the respective rectangle and these four point sets are independent. Thus, for processes $(\hat{\mathcal{P}}_j^{(r)}(q))_{q\in[0,1]}$ being independent and independent of $(U, V, I^{(n)})$, and $(\hat{\mathcal{P}}_j^{(r)}(q))_{q\in[0,1]}$ distributed as $(\hat{\mathcal{P}}_j(q))_{q\in[0,1]}$ for $r = 1, \dots, 4$ and $j \in \mathbb{N}_0$, we have the distributional recurrence

$$\begin{aligned}\left(\hat{\mathcal{P}}_n(q)\right)_{q\in[0,1]} \overset{d}{=} \Bigg(1 + \mathbf{1}_{\{q<U\}}\left[\hat{\mathcal{P}}^{(1)}_{I_1^{(n)}}\left(\frac{q}{U}\right) + \hat{\mathcal{P}}^{(2)}_{I_2^{(n)}}\left(\frac{q}{U}\right)\right] \\ + \mathbf{1}_{\{q\geqslant U\}}\left[\hat{\mathcal{P}}^{(3)}_{I_3^{(n)}}\left(\frac{q-U}{1-U}\right) + \hat{\mathcal{P}}^{(4)}_{I_4^{(n)}}\left(\frac{q-U}{1-U}\right)\right]\Bigg)_{q\in[0,1]}. \end{aligned} \tag{7}$$

The arguments q/U and $(q-U)/(1-U)$ ensure that a vertical line $x_0 = q$ within the square $[0,1]^2$, after normalization, corresponds to the line $x_0 = q/U$ in the left rectangles (if $q < U$) and to the line $x_1 = (q-U)/(1-U)$ in the right rectangles (if $q \geqslant U$). The asymptotic analysis of such distributional recurrences has been carried out for many recurrences, mainly for random variables in $\mathbb{R}$, by the contraction method which is briefly explained now.

7.4. Framework of the contraction method

In this subsection an outline of the contraction method is given in a form tailored to the analysis of partial match. Generally, given a sequence of random variables $(Y_n)_{n\geqslant 0}$ with values in a Banach space $\mathbb{B}$ that satisfies a recurrence in distribution of the form

$$Y_n \overset{d}{=} \sum_{1\leqslant r\leqslant K} A_r(n) Y^{(r)}_{I_r^{(n)}} + b(n), \qquad n \geqslant n_0. \tag{8}$$

Here $\overset{d}{=}$ denotes equality in distribution, and $(Y_j^{(r)})_{j\geqslant 0}$ have the same distribution as $(Y_n)_{n\geqslant 0}$ for all $r = 1, \dots, K$, where $K \geqslant 1$ and $n_0 \geqslant 0$ are fixed integers. Moreover, $I^{(n)} = (I_1^{(n)}, \dots, I_K^{(n)})$ is a vector of random integers in $\{0, \dots, n\}$. The $A_r(n)$ are random linear operators on the space and $b(n)$ is a random vector in the Banach space $\mathbb{B}$. It is also crucially assumed that $(Y_j^{(1)})_{j\geqslant 0}, \dots, (Y_j^{(K)})_{j\geqslant 0}$ and $(A_1(n), \dots, A_K(n), b(n), I^{(n)})$ are independent. However, dependencies between the coefficients $A_r(n)$, $b(n)$ and the integers $I_r^{(n)}$ are allowed.

In many applications from the probabilistic analysis of algorithms, we have $\mathbb{B} = \mathbb{R}$ and the $A_r(n)$ are simply random variables in $\mathbb{R}$; see [67].

In an application to partial match, we cast $Y_n = (\hat{\mathcal{P}}_n(q))_{q\in[0,1]}$ and note that Eq. (7) has exactly the form (8) with $K = 4$, $b(n) = 1$ and random linear maps $A_r(n)$ for $r = 1, \dots, 4$, which contain the indicators such as $\mathbf{1}_{\{q<U\}}$ and the time transformation such as $q \mapsto q/U$.

The process $(\hat{\mathcal{P}}_n(q))_{q\in[0,1]}$ considered as a random variable in $(D[0,1], d_{SK})$ does not map to a Banach space. However, Eq. (8) with its conditions can still be formulated for random variables in $(D[0,1], d_{SK})$ and this is the setting in which we now briefly outline the contraction method.

The contraction method starts manipulating Eq. (8) with normalization. Generally, $(Y_n)_{n\geqslant 0}$ is normalized by centering (if necessary) and dividing by the order of the standard deviation. Subsequently, we assume that the scaling has already been done and we denote the scaled process by $(X_n)_{n\geqslant 0}$. (For our case $Y_n = (\hat{\mathcal{P}}_n(q))_{q\in[0,1]}$ this will be done explicitly below, cf. (13).) Note that affine normalizations of the Y_n yield a sequence $(X_n)_{n\geqslant 0}$ that also satisfies a recurrence of type (8) with new coefficients:

$$X_n \overset{d}{=} \sum_{1\leqslant r\leqslant K} A_r^{(n)} X^{(r)}_{I_r^{(n)}} + b^{(n)}, \qquad n \geqslant n_0, \tag{9}$$

with conditions on identical distributions and independence similar to recurrence (8). The coefficients $A_r^{(n)}$ and $b^{(n)}$ in the modified recurrence (9) are computable from the original coefficients $A_r(n)$, $b(n)$ and the normalization used; see, for example, [67, equation (4)] for the case of random vectors in $\mathbb{R}^d$.

The next step of the contraction method is to identify the limits of the new coefficients $A_r^{(n)}$, $b^{(n)}$,

$$A_r^{(n)} \to A_r, \qquad b^{(n)} \to b \qquad (n \to \infty) \tag{10}$$

appropriately. If with $n \to \infty$ also the $I_r^{(n)}$ grow in probability then one may conjecture that the quantities X_n converge to a random variable X. Letting formally $n \to \infty$, Eq. (9) turns into

$$X \overset{d}{=} \sum_{1\leqslant r\leqslant K} A_r X^{(r)} + b \tag{11}$$

with $X^{(1)}, \dots, X^{(K)}$ distributed as X and $X^{(1)}, \dots, X^{(K)}, (A_1, \dots, A_k, b)$ independent. The next step is to characterize the limit distribution $\mathcal{L}(X)$ by use of the recursive distributional Eq. (11). The idea of [73] to formalize such an approach and to obtain weak convergence $X_n \to X$ consists of firstly using the right-hand side of (11) to define a map as follows: Denote by $\mathcal{M}(\mathbb{B})$ the space of all probability measures on $\mathbb{B}$ and

$$\begin{aligned} &T : \mathcal{M}(\mathbb{B}) \to \mathcal{M}(\mathbb{B}), \\ &T(\mu) = \mathcal{L}\left(\sum_{1\leqslant r\leqslant K} A_r Z^{(r)} + b\right), \end{aligned} \tag{12}$$

where $(A_1, \dots, A_K, b), Z^{(1)}, \dots, Z^{(K)}$ are independent and $Z^{(1)}, \dots, Z^{(K)}$ have the distribution μ. Hence, a random variable X solves (11) if and only if its distribution $\mathcal{L}(X)$ is a fixed-point of T. Secondly, to find fixed-points of T appropriate subspaces of $\mathcal{M}(\mathbb{B})$ are endowed with a complete metric, such that the restriction of the map T turns into a contraction. Then Banach's fixed-point theorem yields (in the subspace) a unique fixed-point of T and one may as well use the metric to also derive convergence of $\mathcal{L}(X_n)$ to $\mathcal{L}(X)$ in this metric. The metric used should be strong enough to imply weak convergence, so that one obtains the desired limit law $X_n \to X$. In cases where the metric used is not complete (or not known to be complete) as in our case below the existence of a fixed-point may be obtained by additional arguments; however, the general approach remains the same.

Metrics that have proved suitable in this context are the minimal L_p-metrics (also called Wasserstein-metrics) and the family of Zolotarev metrics. The Zolotarev metric ζ_2 is a suitable metric for the application of the contraction method to the partial match process. We refer the reader for details on these metrics and their use in the context of the contraction method to [1,85] for the minimal L_p-metrics and for the family of Zolotarev metrics to [53,68]. For the development of the contraction method on function spaces, we refer to the review given on pages 1779 bottom and on page 1780 of [68].

7.5. Expectation of the partial match process

For the application of the contraction method by use of the Zolotarev metric ζ_2 it is sufficient to normalize the components individually. Since it turns out that expectation and standard deviation are asymptotically of the same order, one may just normalize with the order of the expectations. We have:

Theorem 11. *For the random 2-dimensional quadtree with n points there exists an $\varepsilon > 0$ such that, as $n \to \infty$,*

$$\sup_{q\in[0,1]} \left| n^{-\alpha}\, \mathbb{E}[\hat{P}_n(q)] - K_0 (q(1-q))^{\alpha/2} \right| = O\big(n^{-\varepsilon}\big),$$

where

$$\alpha = \frac{\sqrt{17}-3}{2}, \quad K_0 = \frac{\Gamma(2\alpha+2)\Gamma(\alpha+2)}{2\Gamma^3(\alpha+1)\Gamma^2(\alpha/2+1)}.$$

The asymptotic expectation for fixed q was identified using fragmentation theory, see [6] for a general introduction, and a coupling argument of Markov chains by [16]. A strengthening to a uniform version, together with a bound on the error as stated in Theorem 11, was provided in [11] by refining the arguments of [16] and employing a Poissonization of the input size with de-Poissonization based on concentration.

Note that for $q \in \{0,1\}$ the latter asymptotic result only yields an order of $o(n^\alpha)$. At the border of the square, a more precise result is given in [40,42], we have

$$\mathbb{E}[\hat{P}_n(0)] = \mathbb{E}[\hat{P}_n(1)] = \Theta\left(n^{\sqrt{2}-1}\right).$$

Regarding process convergence of $(\hat{P}_n(q))_{q\in[0,1]}$ we normalize

$$\mathcal{Z}_n(q) := \frac{\hat{P}_n(q)}{K_0 n^\alpha}, \quad q \in [0,1]. \tag{13}$$

7.6. Normalization and limit process

Starting from recurrence (7), by calculating the recurrence (9), which is omitted here, we obtain for (11) the recursive distributional equation

$$\begin{aligned}(\mathcal{Z}(q))_{q\in[0,1]} \overset{d}{=} \Bigg(& \mathbf{1}_{\{q<U\}} \left[(UV)^\alpha \mathcal{Z}^{(1)}\left(\frac{q}{U}\right) + \big(U(1-V)\big)^\alpha \mathcal{Z}^{(2)}\left(\frac{q}{U}\right)\right] \\ & + \mathbf{1}_{\{q\geqslant U\}} \left[\big((1-U)V\big)^\alpha \mathcal{Z}^{(3)}\left(\frac{q-U}{1-U}\right) \right. \\ & \left. + \big((1-U)(1-V)\big)^\alpha \mathcal{Z}^{(4)}\left(\frac{q-U}{1-U}\right)\right] \Bigg)_{q\in[0,1]},\end{aligned} \tag{14}$$

where U and V are independent $[0,1]$-uniform random variables and $\mathcal{Z}^{(i)}$, $i = 1,\dots,4$ are independent copies of the process $\mathcal{Z}$, which are also independent of U and V.

The existence to a solution of Eq. (14) is the subject of Section 4 of [11]. A solution $(\mathcal{Z}(q))_{q\in[0,1]}$ is obtained as a point-wise limit of martingales:

Theorem 12. *The map T in (12) associated with the recursive distributional equation in (14) has a solution $\mathcal{L}((\mathcal{Z}(q))_{q\in[0,1]})$, which is unique in the subspace of probability measures $\mathcal{L}((\Upsilon(q))_{q\in[0,1]})$ on $(D[0,1], d_{SK})$ with $\mathbb{E}[\|\Upsilon(q)\|_\infty^2] < \infty$ and $\mathbb{E}[\Upsilon(q)] = (q(1-q))^{\alpha/2}$. Moreover, $(\mathcal{Z}(q))_{q\in[0,1]}$ has continuous paths, i.e, is a random element of $C[0,1]$.*

An approximation of a realization of this solution $(\mathcal{Z}(q))_{q\in[0,1]}$ of Eq. (14) is shown in Fig. 5. Besides having continuous paths, another property of the limit process is that its one-dimensional marginal distributions are identical up to a deterministic multiplicative constant; see [11, Theorem 5]:

Theorem 13. *There exists a random variable $\Psi \geqslant 0$ such that for all $q \in [0,1]$,*

$$\mathcal{Z}(q) \overset{d}{=} (q(1-q))^{\alpha/2}\Psi. \tag{15}$$

The distribution of Ψ is the unique solution of the recursive distributional equation

$$\Psi \overset{d}{=} U^{\alpha/2}V^\alpha\Psi + U^{\alpha/2}(1-V)^\alpha\Psi' \tag{16}$$

with $\mathbb{E}[\Psi] = 1$ and $\mathbb{E}[\Psi^2] < \infty$ and $\Psi' \overset{d}{=} \Psi$ such that (Ψ,Ψ') is independent of (U,V).

7.7. Functional limit theorem for the partial match process

The program of the contraction method outlined in Section 7.4 has been carried out in [11] for the partial match process based on the Zolotarev metric ζ_2. The resulting functional limit theorem, using the notations of Section 7.6, states as follows:

Theorem 14. *For the normalized process $(\mathcal{Z}_n(q))_{q\in[0,1]}$ we have*

$$(\mathcal{Z}_n(q))_{q\in[0,1]} \xrightarrow{d} (\mathcal{Z}(q))_{q\in[0,1]}$$

in $(D[0,1], d_{SK})$, where the limit process $\mathcal{Z} = (\mathcal{Z}(q))_{q\in[0,1]}$ is characterized in Theorem 12.

While the convergence in Theorem 14 is in distribution, the question of almost sure convergence has been addressed in [20]. Using fragmentation theory and building upon results of [11] in [20] the following was obtained:

Theorem 15. *For all $q \in [0,1]$ we have almost surely as $n \to \infty$ that*

$$\mathcal{Z}_n(q) \to K_0 \mathcal{M}_\infty(q),$$

where $(\mathcal{M}_\infty(q))_{q\in[0,1]}$ is the limit of a continuous time martingale. Moreover, in probability

$$\sup_{q\in[0,1]} |\mathcal{Z}_n(q) - K_0\mathcal{M}_\infty(q)| \xrightarrow{\mathbb{P}} 0.$$

7.8. Implications of the functional limit theorem

The projections $\pi_{q_1,\dots,q_d} : D[0,1] \to \mathbb{R}^d, f \mapsto (f(q_1),\dots,f(q_d))$ are generally not continuous as maps within the Skorokhod metric d_{SK}. Nevertheless, we may apply the continuous mapping theorem since the limit process $\mathcal{Z}$ has continuous paths almost surely. Hence, the functional limit theorem (Theorem 14) implies convergence of the finite dimensional marginals by the continuous mapping theorem, cf. Section 7.2. In particular, for all $q \in [0,1]$, we have

$$\frac{\hat{P}_n(q)}{K_0 n^\alpha} \xrightarrow{d} \mathcal{Z}(q) \overset{d}{=} (q(1-q))^{\alpha/2}\Psi, \tag{17}$$

with Ψ given in Theorem 13. The convergence in (17) also holds for all moments.

Furthermore, the map $\sup : D[0,1] \to \mathbb{R}, f \mapsto \sup_{q\in[0,1]} f(q)$ is continuous. Hence, the functional limit theorem (Theorem 14) implies asymptotics for the worst case complexity of partial match over all possible queries:

Corollary 16. *([11]) For the worst case complexity $\mathcal{W}_n := \sup_{q\in[0,1]} \hat{P}_n(q)$ we have*

$$\frac{\mathcal{W}_n}{K_0 n^\alpha} \xrightarrow{d} \mathcal{W} := \sup_{q\in[0,1]} \mathcal{Z}(q),$$

together with convergence of all moments. In particular, $\mathbb{E}[\mathcal{W}] < \infty$, $0 < \mathbb{V}(\mathcal{W}) < \infty$ and

$$\mathbb{E}\left[\sup_{q\in[0,1]} \hat{P}_n(q)\right] \sim K_0\, \mathbb{E}[\mathcal{W}] n^\alpha, \quad \mathbb{V}\left(\sup_{q\in[0,1]} \hat{P}_n(q)\right) \sim K_0^2 \mathbb{V}(\mathcal{W}) n^{2\alpha}.$$

Since convergence in the functional limit theorem (Theorem 14) is shown within the Zolotarev metric ζ_2, the convergence also holds for the covariance functions. In particular, this provides access to the second moments as follows. With the notation $\mu_2(q) := \mathbb{E}[\mathcal{Z}(q)^2]$ we obtain from (14) by conditioning on U and a simplification that

$$\begin{aligned}\mu_2(q) = & \frac{2}{2\alpha+1}\left\{ \int_q^1 x^{2\alpha}\mu_2\left(\frac{q}{x}\right)dx + \int_0^q (1-x)^{2\alpha}\mu_2\left(\frac{1-q}{1-x}\right)dx \right\} \\ & + 2\mathrm{B}(\alpha+1,\alpha+1)\cdot\frac{(q(1-q))^\alpha}{\alpha+1}.\end{aligned}$$

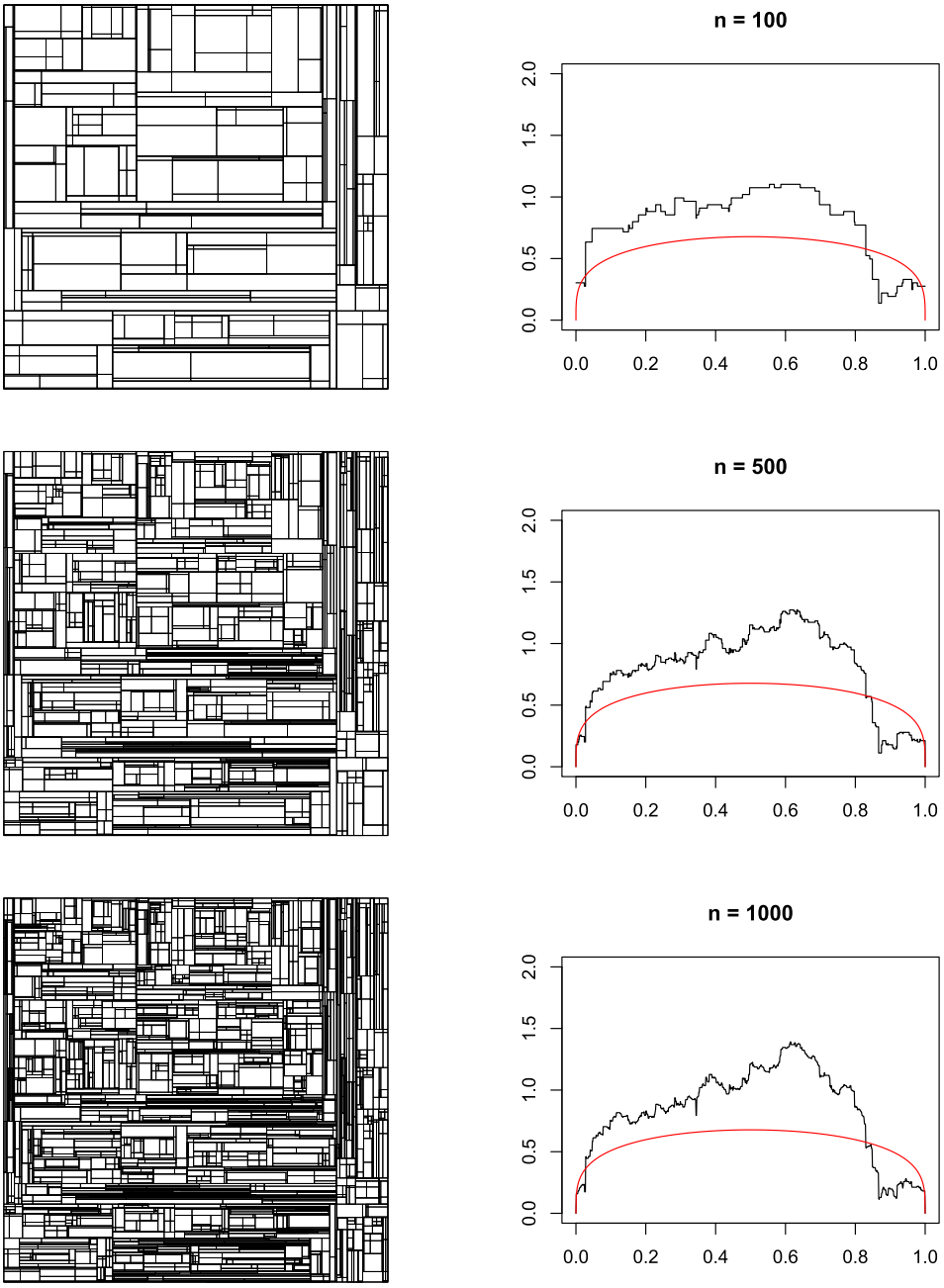

Fig. 5. Realization of a decomposition of the unit square induced by the quadtree for an increasing number n of data ($n = 100, 500, 1000$) in the uniform probabilistic model. On the right the corresponding normalized partial match processes $q \mapsto \mathcal{Z}_n(q)$ together with its mean function $q \mapsto (q(1-q))^{\alpha/2}$ (in red). This indicates also an approximation of the limit process $(\mathcal{Z}(q))_{q\in[0,1]}$ and the almost sure limit behavior of the partial match process.

By plugging in it is verified that a solution of the latter integral equation is given by

$$\mathbb{E}[\mathcal{Z}(q)^2] = \mu_2(q) = 2\mathrm{B}(\alpha+1,\alpha+1)\frac{2\alpha+1}{3(1-\alpha)}(q(1-q))^{\alpha}.$$

The uniqueness of the solution is demonstrated by a contraction argument on an appropriate function space. Together with the mean of $\mathcal{Z}(q)$, see Theorem 12, one obtains

$$\mathbb{V}\big(\mathcal{Z}(q)\big) \sim \left[2\mathrm{B}(\alpha+1,\alpha+1)\frac{2\alpha+1}{3(1-\alpha)} - 1\right](q(1-q))^{\alpha}. \tag{18}$$

We return to the model with a uniform query being independent of the data. Note that for ξ uniformly distributed on $[0,1]$ and independent of the data we have $\hat{P}_n(\xi) = \hat{P}_n$ with $\hat{P}_n$ as in Section 6 and Theorem 9 for dimension 2. With the functional limit theorem (Theorem 14) we obtain a limit law for $\hat{P}_n$ as a corollary as well as the asymptotic variance on which we further comment below.

Corollary 17. *[[11]] For ξ uniformly distributed on $[0,1]$ and independent of the data we have*

$$\mathbb{V}(\hat{P}_n) = \mathbb{V}(\hat{P}_n(\xi)) \sim K_v n^{2\alpha},$$

where

$$K_v = K_0^2\left(\frac{2(2\alpha+1)}{3(1-\alpha)}B^2(\alpha+1,\alpha+1) - B^2(\alpha/2+1,\alpha/2+1)\right)$$
$$= 0.447363034\ldots$$

and

$$\frac{\hat{P}_n}{K_0 n^\alpha} = \frac{\hat{P}_n(\xi)}{K_0 n^\alpha} \xrightarrow{d} \mathcal{Z}(\xi),$$

where ξ is independent of the limiting process $\mathcal{Z}$ in Theorem 14.

The derivation of the asymptotic variance of $\hat{P}_n(\xi)$ in Corollary 17 has a twisted history: Before drawing back to the functional limit theorem (Theorem 14) and its limit process $(\mathcal{Z}(q))_{q\in[0,1]}$ it had been attempted by both analytic and probabilistic methods, to build upon the following univariate recurrence derived from (7),

$$\hat{P}_n(\xi) \overset{d}{=} 1 + \mathbf{1}_{\{\xi<U\}}\left[\hat{P}^{(1)}_{I_1^{(n)}}\left(\frac{\xi}{U}\right) + \hat{P}^{(2)}_{I_2^{(n)}}\left(\frac{\xi}{U}\right)\right] + \mathbf{1}_{\{\xi\geq U\}}\left[\hat{P}^{(3)}_{I_3^{(n)}}\left(\frac{\xi-U}{1-U}\right) + \hat{P}^{(4)}_{I_4^{(n)}}\left(\frac{\xi-U}{1-U}\right)\right], \tag{19}$$

where $(\hat{P}^{(r)}_j(q))_{q\in[0,1]}$ are independent and independent of $(U, V, \xi, I^{(n)})$, and $(\hat{P}^{(r)}_j(q))_{q\in[0,1]}$ are distributed as $(\hat{P}_j(q))_{q\in[0,1]}$ for $r = 1, \dots, 4$ and $j \in \mathbb{N}_0$.

Now, conditional on $\xi < U$ we have that ξ/U is uniformly distributed over $[0, 1]$. Similarly, conditional on $\xi \geqslant U$ the random variable $(\xi - U)/(1 - U)$ is uniformly distributed over $[0, 1]$. Hence, the four contributions

$$\hat{P}^{(1)}_{I_1^{(n)}}\left(\frac{\xi}{U}\right), \quad \hat{P}^{(2)}_{I_2^{(n)}}\left(\frac{\xi}{U}\right), \quad \hat{P}^{(3)}_{I_3^{(n)}}\left(\frac{\xi-U}{1-U}\right), \quad \hat{P}^{(4)}_{I_4^{(n)}}\left(\frac{\xi-U}{1-U}\right)$$

are conditional on $I^{(n)} = (k, \ell, m, n)$ distributed as the quantities $\hat{P}_k(\xi), \hat{P}_\ell(\xi), \hat{P}_m(\xi), \hat{P}_n(\xi)$ respectively. Thus, conditioning on $I^{(n)}$ and taking expectations on the left- and right-hand sides of (19) give a recurrence for the analysis of $\mathbb{E}[\hat{P}_n(\xi)]$. This is recurrence (4) (for dimension 2) on which the analysis of [14,40] is based, leading to Theorem 9.

However, for starting a similar analysis of second moments (or the variance) there appears the basic problem that although $(\hat{P}^{(r)}_j(q))_{q\in[0,1]}$ are independent for all $r = 1, \dots, 4$ and $j \in \mathbb{N}_0$ the contributions

$$\hat{P}^{(1)}_k\left(\frac{\xi}{U}\right), \quad \hat{P}^{(2)}_\ell\left(\frac{\xi}{U}\right), \quad \hat{P}^{(3)}_m\left(\frac{\xi-U}{1-U}\right), \quad \hat{P}^{(4)}_n\left(\frac{\xi-U}{1-U}\right)$$

become dependent due to the joint appearance of ξ. Ignoring this dependence and instead assuming independence in such an analysis leads to the correct order $n^{2\alpha}$ for $\mathbb{V}(\hat{P}_n(\xi))$ but does not yield the correct constant K_v; see [17,60,64,66].

8. Master theorems and beyond

8.1. The continuous master theorem

In addition to the powerful and mathematically sophisticated techniques that we explored in previous sections, there exist a few other real-analytic or elementary methods that often prove helpful for the analysis of multidimensional data structures. One of the most useful such methods is the *Continuous Master Theorem* [74] (CMT, for short) of Roura, which can be applied to a wide variety of divide-and-conquer recurrences of the form:

$$F_n = t_n + \sum_{0\leqslant j<n} \omega_{n,j} F_j, \qquad n \geqslant n_0,$$

with a *toll function* $t_n = \Theta(n^a \log^b n)$ of at most polynomial growth and a sequence of *weights* $\{\omega_{n,j}\}$ that satisfy the following two conditions:

- $W_n = \sum_{0\leqslant j<n} \omega_{n,j} \leqslant 1$ for $n \geqslant n_0$, and
- $Z_n = \frac{1}{nW_n} \sum_{0\leqslant j<n} j\omega_{n,j} \leqslant Z < 1$ for $n \geqslant n_0$.

In words, these conditions state that there is at least one recursive call (on average) and that the average size of the inputs to recursive calls is a proper fraction of the original size n—the latter justifying the divide-and-conquer nature of these recurrences. Briefly, the theorem builds upon the intuitive "ansatz method" (or matched asymptotics; see page 30 and successive pages of Section 6 for concrete examples), and puts it on solid grounds.

Roura's CMT has been used for the analysis of several algorithms; see for instance [63] and [69]. In the context of multidimensional data structures and associative queries, the CMT has been successfully applied to analyze the expected cost of exact searches and the expected cost of random partial match searches of several multidimensional trees.

For the application of the theorem, one needs to find a *shape function*, a smooth continuation or an approximation of the discrete weights. Let $\omega : [0, 1] \to \mathbb{R}$ be a shape function. The idea is that $\omega(j/n)$ serves as a good approximation for $\omega_{n,j}$ for large n. Technically, $\omega(z)$ must fulfill two conditions stated in the theorem below. In general, $\omega(z) = \lim_{n\to\infty} n\omega_{n,\lfloor z\cdot n\rfloor}$ is said to be a shape function if the limit exists.

Once the shape function is identified, the theorem can be effectively applied to find the asymptotic behavior of the main order term of F_n, by computing a quantity $\mathcal{H}$ (known as *const-entropy*), and eventually another quantity $\tilde{\mathcal{H}}$ (called *log-entropy*) or solving the equation

$$\int_0^1 z^x \omega(z)\, dz - 1 = 0,$$

depending on the sign of $\mathcal{H}$.

Theorem 18. *([74]) Let F_n be recursively defined by*

$$F_n = \begin{cases} b_n, & \text{for} \quad 0 \leqslant n < n_0, \\ t_n + \sum_{0\leqslant j<n} \omega_{n,j} F_j, & \text{for} \quad n \geqslant n_0, \end{cases}$$

where the toll function satisfies $t_n \sim \kappa n^a \ln^b(n)$ as $n \to \infty$ for constants $\kappa \neq 0$, $a \geqslant 0$ and $b > -1$. Assume there exists a function $\omega : [0, 1] \to \mathbb{R}$, such that $\int_0^1 w(z)dz$ exists and is at least 1, and such that

$$\sum_{0\leqslant j<n} \left|\omega_{n,j} - \int_{j/n}^{(j+1)/n} \omega(z)\, dz\right| = \mathcal{O}(n^{-c}),$$

for some constant $c > 0$. With $\mathcal{H} = 1 - \int_0^1 z^a \omega(z)\, dz$, we have the following cases:

1. *If $\mathcal{H} > 0$, then $F_n \sim \frac{t_n}{\mathcal{H}}$.*
2. *If $\mathcal{H} = 0$, then $F_n \sim \frac{t_n \ln(n)}{\tilde{\mathcal{H}}}$, with $\tilde{\mathcal{H}} = (b + 1) \int_0^1 z^a \ln(1/z)\, w(z)\, dz$.*
3. *If $\mathcal{H} < 0$, then $F_n \sim \Theta(n^\alpha)$ for the unique $\alpha \in \mathbb{R}^+$ with $\int_0^1 z^\alpha \omega(z)\, dz = 1$.*

Some remarks on the use of CMT are in order.

- The CMT is easy to apply and more accessible to non-experts, even in situations with complicated weights such as the median and hybrid median k-d trees [33] or quad k-d trees [2,3,28]. It quickly gives the growth order of the recurrence in all cases, and in the lucky cases when $\mathcal{H} \geqslant 0$ the constants in the leading order are also available. When the asymptotic behavior is known only to within a bounded factor $\Theta(1)$ as in the third case ($\mathcal{H} < 0$), other approaches such as the analytic ones discussed in Section 6 may be applied; see, for example, for quad k-d trees [32].
- A typical situation of the second case of Theorem 18 is the internal path length (IPL) of a random search tree (the sum of all distances from each node to the root), leading to $n \log n$ asymptotic behavior. The IPL is related to the cost of the construction of a random tree by n insertions of random elements into an initially empty tree. Also, by dividing the expected IPL by the size, we get the expected cost of a random successful exact search.

In what follows, we first sketch a back-of-the-envelope derivation of Theorem 18, providing more intuition for the asymptotic approximations. Then we give a few representative examples of the application

of the CMT, and after that we briefly discuss its applications in the literature, as well as how it has been adapted to cope with systems of divide-and-conquer recurrences.

8.1.1. A heuristic derivation of Theorem 18

To see how the expressions in Theorem 18 are derived, we assume the simpler situation when $t_n \sim \kappa n^a$ ($b = 0$). We begin by assuming the Ansatz $F_n \sim Cn^\alpha$. Then $\alpha \geqslant a$. Let

$$J(s) := \int_0^1 \omega(t) t^s dt.$$

Then, according to the sign of $\mathcal{H} = 1 - J(a)$, we distinguish into three cases.

- If $J(a) < 1$ or $\mathcal{H} > 0$, then by the shape function approximation, we obtain

$$\sum_{0\leqslant j<n} \omega_{n,j} F_j \approx C \sum_{0\leqslant j<n} \omega_{n,j} j^\alpha \approx Cn^\alpha \int_0^1 \omega(t) t^\alpha dt = CJ(\alpha) n^\alpha, \tag{20}$$

  which then yields

$$t_n = F_n - \sum_{0\leqslant j<n} \omega_{n,j} F_j \approx Cn^\alpha - Cn^\alpha \int_0^1 \omega(t) t^\alpha dt = C\mathcal{H} n^\alpha,$$

  implying that $\alpha = a$ and $C\mathcal{H} = \tau$, which yields the result in the first case of Theorem 18.
- If $J(a) = 1$ or $\mathcal{H} = 0$, then the Ansatz that $F_n \sim Cn^\alpha$ fails. Assume the new Ansatz that $F_n \sim Cn^\alpha \log n$. Then we have

$$\begin{aligned} F_n - \sum_{0\leqslant j<n} \omega_{n,j} F_j &\approx Cn^\alpha \log n - C \sum_{1\leqslant j<n} \omega_{n,j} j^\alpha \log j \\ &\approx Cn^\alpha \log n - Cn^\alpha \int_0^1 \omega(t) t^\alpha \log(tn) dt \\ &= Cn^\alpha \int_0^1 \omega(t) t^\alpha \log(1/t) dt, \end{aligned}$$

  giving $\alpha = a$ and the expression in the second case of Theorem 18 when $b = 0$. The case where $b > 0$ is similar.
- If $\mathcal{H} < 0$ or $J(a) > 1$, then the above calculations fail to identify the value of α. Write formally

$$F_n = \frac{1}{2\pi i} \int \phi(s) n^s ds, \quad \text{and} \quad t_n = \frac{1}{2\pi i} \int T(s) n^s ds,$$

  for some $\phi(s), T(s)$ and integration path. Then the Ansatz $F_n \sim Cn^\alpha$ gives, by (20),

$$t_n = F_n - \sum_{0\leqslant j<n} \omega_{n,j} F_j \approx \frac{1}{2\pi i} \int \phi(s)\big(1 - J(s)\big) n^s ds,$$

  so that $\phi(s) = T(s)/(1 - J(s))$. Now following a similar reasoning for Mellin inversion (see [38]), we expect roughly that the real part of the largest root α of the equation $1 - J(s) = 0$ gives the dominant polynomial order n^α for F_n. More precisely, if α is the sole simple zero of $J(s) = 1$ on the vertical line $\Re(s) = \alpha$, then we also expect the finer asymptotic approximation $F_n \sim Cn^\alpha$, with $C = -T(\alpha)/J'(\alpha)$ (the function $T(s)$ is related to the Dirichlet series $\sum_{n\geqslant 1} t_n n^{-s}$ of t_n; see [39]). Situations with multiple zeros or higher order zeros can be dealt with similarly; see [38] for more information.

8.2. Applications of the continuous master theorem

8.2.1. Example #1: relaxed k-d trees

The analysis of random partial matches in relaxed k-d trees (see Section 2) was originally carried out in [23] and later refined in [60], in both cases using generating functions, singularity analysis and other tools from the analytic combinatorics machinery. Alternatively, they also provide a very simple and natural example of the application of CMT.

Consider the expected cost $\hat{\mathsf{P}}_n = \mathbb{E}\left[\hat{\mathcal{P}}_n\right]$ of a random partial match in a relaxed k-d tree of size $n > 0$. In view of the symmetries of the probabilistic model assumed to generate the data points and the query, and the fact that the discriminants of the nodes are uniformly and independently chosen from $\{0, \dots, k-1\}$, it follows that the pattern of specified/unspecified coordinates of the query does not matter, and that the root discriminates w.r.t. a specified coordinate with probability $\frac{s}{k}$ (and w.r.t. an unspecified coordinate with complementary probability $1 - \frac{s}{k}$). Then, the algorithm recursively proceeds in one or both subtrees, and we face smaller instances of exactly the same problem: a random query with s out of k specified coordinates in a random relaxed k-d tree of size $j < n$.

Now, the probability that the left subtree is of size j, $0 \leqslant j < n$, is $1/n$, because of our assumption that each coordinate of each data point is generated i.i.d. from some continuous distribution in $[0, 1]$. If the corresponding coordinate of the query is not specified, we will proceed in both subtrees, and thus we will have a term $\hat{\mathsf{P}}_j + \hat{\mathsf{P}}_{n-1-j}$ in the recurrence. On the other hand, if the coordinate is specified, then the probability that we must continue in the left subtree is $(j+1)/(n+1)$, and the probability that we continue in the right subtree is $(n-j)/(n+1)$. This easily follows if we consider the rank vectors of $\mathbf{r}' = \mathbf{r}(\mathbf{q}) = (r'_0, \dots, r'_{k-1})$ and of $\mathbf{r} = \mathbf{r}(\mathbf{x}) = (r_0, \dots, r_{k-1})$, with $\langle \mathbf{x}, i\rangle$ being the root of the random tree. Then, since $r_i = j + 1$ (when conditioning on the size of the left subtree being j), and r'_i is uniform in $\{0, \dots, n+1\}$, the stated probabilities $(j+1)/(n+1)$ and $(n-j)/(n+1)$ immediately follow.

Putting all pieces together yields the final recurrence

$$\begin{aligned} \hat{\mathsf{P}}_n &= 1 + \frac{1}{n}\left(1 - \frac{s}{k}\right) \sum_{0\leqslant j<n} \hat{\mathsf{P}}_j + \hat{\mathsf{P}}_{n-1-j} + \frac{1}{n}\frac{s}{k} \sum_{0\leqslant j<n} \frac{j+1}{n+1}\hat{\mathsf{P}}_j + \frac{n-j}{n+1}\hat{\mathsf{P}}_{n-1-j} \\ &= 1 + \frac{2}{n}\left(1 - \frac{s}{k}\right) \sum_{0\leqslant j<n} \hat{\mathsf{P}}_j + \frac{2}{n(n+1)}\frac{s}{k} \sum_{0\leqslant j<n} (j+1)\hat{\mathsf{P}}_j. \end{aligned} \tag{21}$$

Introducing $\rho := s/k$, the weights in the recurrence above are $\omega_{n,j} = \frac{2}{n}(1-\rho) + \frac{2}{n}\frac{j+1}{n+1}\rho$, and the shape function $\omega(z) = \lim_{n\to\infty} n \cdot \omega_{n,z\cdot n} = 2(1-\rho) + 2\rho z$. Since $t_n = 1$, we have

$$\mathcal{H} = 1 - \int_0^1 \omega(z)\, z^0\, dz = 1 - (2(1-\rho) + \rho) = -1 + \rho,$$

which is always negative (because we assume $s < k$, that is, $\rho < 1$). We are thus in the third case of the CMT, and we need to find the solution to the equation

$$\int_0^1 \omega(z)\, z^\alpha\, dz - 1 = 0,$$

that is, we are looking for the real positive value x such that

$$2(1-\rho)\frac{1}{\alpha+1} + 2\rho\frac{1}{\alpha+2} = 1,$$

namely, $\alpha = (\sqrt{9 - 8\rho} - 1)/2$. The CMT entails that $\hat{\mathsf{P}}_n = \Theta(n^\alpha)$. Compare this to the results originally obtained by [23] (also [60]) using analytic methods and singularity analysis; the CMT gives us the correct exponent α, but the more powerful techniques of Section 6 can give us the exact form of $\hat{\mathsf{P}}_n$ and the asymptotic estimate $\hat{\mathsf{P}}_n = \beta \cdot n^\alpha + \mathcal{O}(1)$, with the explicit form for β in terms of s and k (actually, in terms of $\rho = s/k$), which the CMT cannot give—the CMT can't even establish the existence of such constant factor β!

8.2.2. Example #2: median k-d trees

In median k-d trees [33] (mentioned in Section 2) discriminants are chosen to achieve a more balanced partition of the space $[0, 1]^k$, namely,

selecting for each node the coordinate closest to the corresponding coordinate of the rescaled center of the node's associated bounding box. This ultimately translates to a more complicated expression for the probability that the left subtree L of a random median k-d tree T of size n is of size j. In particular, if $j \leqslant \lfloor n/2 \rfloor$ then

$$\pi_{n,j} = \mathbb{P}[|L| = j \,|\, |T| = n] = \frac{1}{n^k} \cdot \left[(2j+2)^k - (2j+1)^k\right],$$

whereas if $j > \lfloor n/2 \rfloor$ then

$$\pi_{n,j} = \mathbb{P}[|L| = j \,|\, |T| = n] = \frac{1}{n^k} \cdot \left[(2(n-j)-1)^k - (2(n-j)-2)^k\right].$$

Those probabilities are a fundamental part of the recurrences describing the IPL or the cost of partial matches. However, due to their more complex form, the application of more powerful techniques such as the probabilistic methods of Section 7 or those from the analytic combinatorics camp described in Section 6 becomes more cumbersome or even unfeasible.

Take, for example, the expected IPL I_n of a random median k-d tree of size n. We compute the expected IPLs of the left and right subtrees, then we add 1 for every node in those subtrees (for a total of $n-1$, as each path to the root of the tree is one unit larger. This gives the recurrence for computing I_n:

$$I_n = n - 1 + \sum_{0 \leqslant j < n} \pi_{n,j} \left(I_j + I_{n-1-j}\right). \tag{22}$$

Notice that the recurrence above is fairly general; one only needs to replace $\pi_{n,j}$ with the appropriate quantity for the particular multidimensional tree under consideration.

The first step in the application of the CMT is to derive the shape function corresponding to the weights $\omega_{n,j} = \pi_{n,j} + \pi_{n,n-1-j}$, which can readily be shown to be

$$\omega(z) = \lim_{n\to\infty} n \cdot \omega_{n,z\cdot n} = \begin{cases} k2^k z^{k-1}, & \text{if } z \leqslant 1/2, \\ k2^k (1-z)^{k-1}, & \text{if } z \geqslant 1/2. \end{cases}$$

Then computing $\mathcal{H} = 1 - \int_0^1 z\omega(z)\,dz$ gives $\mathcal{H} = 0$, so we compute instead

$$\tilde{\mathcal{H}} = -\int_0^1 z \ln z\, \omega(z)\, dz.$$

According to the CMT, we then have $I_n = c_k n \ln n + o(n \log n)$, with $c_k = 1/\tilde{\mathcal{H}}_k$. There is no easy closed form for $\tilde{\mathcal{H}}_k$ (we have added the subscript k to stress the dependence on k), but we can evaluate it for any particular value of k, for example, $c_2 = 6/(5 - 2\ln 2) \approx 1.66\ldots$, $c_3 = 3/(4 - 3\ln 2) \approx 1.562\ldots$, etc. We can also prove that $\tilde{\mathcal{H}}_k \to \ln 2$ as $k \to \infty$; that is, $I_n \sim n \log_2 n + o(n \log n)$ as $k \to \infty$. Other properties, such as the monotonicity of c_k, are not difficult to establish.

The CMT is also useful for the analysis of the expected cost of random partial matches. The recurrence is very similar to (21) but with the probabilities $1/n$ replaced by $\pi_{n,j}$. As $\mathcal{H} = -1 + \rho < 0$ for this new recurrence, we also need to find the real positive solution α of

$$2^{-\alpha}\left(\frac{k-s}{k+\alpha} + \frac{s}{2(k+\alpha+1)}\right) + 2^k\left\{s\,B(1/2; k+1, \alpha+1) + (k-s)\,B(1/2; k, \alpha+1)\right\} = 1, \tag{23}$$

with $B(z; a, b) = \int_0^z t^{a-1}(1-t)^{b-1}\,dt$ denoting the incomplete Beta function [70]. Numerical approximations to α with arbitrary precision can easily be computed. An interesting fact is that α depends on both s and k, not only on the ratio $\rho = s/k$, unlike other variants of k-d trees, including standard and relaxed k-d trees. On the other hand, if we keep the ratio $\rho = s/k$ constant and let $k \to \infty$, it is not difficult to prove that the lower limit for α is $\log_2(2-\rho)$. Indeed, if we call $\alpha_k(\rho) = \alpha(s,k)$ the solution of (23), numerical computations indicate that $\alpha_k(\rho)$ is monotonically decreasing in k:

$$\alpha_2(\rho) \geqslant \alpha_3(\rho) \geqslant \cdots \alpha_\infty(\rho) = \log_2(2-\rho),$$

for any $\rho \in (0,1)$. Moreover, $\alpha^{[\mathrm{rlx}]}(\rho) \geqslant \alpha_k(\rho) \geqslant \log_2(2-\rho) > \alpha^{[\mathrm{std}]}(\rho)$ for any ρ and any k, with $\alpha^{[\mathrm{rlx}]}$ and $\alpha^{[\mathrm{std}]}$ denoting the exponents in the expected cost of random partial matches in relaxed and standard k-d trees, respectively.

8.3. Extensions of the CMT: systems of divide & conquer recurrences

In [33] some results were obtained that allow the application of CMT when one has to cope with a system of divide-and-conquer recurrences instead of a single one. Those systems can be expressed in terms of a sequence of D-dimensional vectors $\{\mathbf{F}_n\}$ of costs, another sequence of D-dimensional vectors $\{\mathbf{t}\}_n$ of toll functions and a sequence of $D \times D$ matrices $\{\Omega_{n,j}\}_{n,j}$, with $\Omega_{n,j} = \left(\omega_{n,j}^{(i,\ell)}\right)$ and $\omega_{n,j}^{(i,\ell)}$ being the weight of algorithm i with inputs of size n making a recursive call to algorithm ℓ with inputs of size j. Then

$$\mathbf{F}_n = \mathbf{t}_n + \sum_{0 \leqslant j < n} \Omega_{n,j} \cdot \mathbf{F}_j.$$

The results do not apply to arbitrary systems of equations, only when the underlying call graph associated with $\Omega_{n,j}$ satisfies a certain cycle structure. Fortunately, this is the case with the analysis of IPL or random partial matches; such a simple structure can be exploited to use the CMT and derive the expected IPL or the exponent α in the expected cost of random partial matches—the latter is the solution to $\det(I - \Phi(\alpha)) = 0$, where $\Phi(x)$ is a certain matrix that is defined from the $\Omega_{n,j}$'s, and it is the $D \times D$ analogue of $\int_0^1 z^x \omega(z)\,dz$. Since the equation is, in general, a polynomial of higher degree, we cannot hope for a closed form for α, but numerical approximations can be computed to any desired precision. These extensions of the CMT have been successfully applied to analyze hybrid median k-d trees, and they could also be used to re-derive known results for standard k-d trees or standard k-d-t trees, in general to variants of k-d trees where the discriminants are assigned in some sort of "cyclic" fashion top-down— so that the discriminants along any subsequence of k nodes in a path from the root to a leaf do not repeat.

8.4. Asymptotic analysis of fixed partial match searches

The brief heuristic arguments for deriving (3) and (6) in Section 6 (which also provide the intuition behind the CMT, Section 8.1) have likewise been used to study the expected cost of partial matches with a fixed query $\mathbf{q}$.

There are a few rigorous results, e.g., [11,16], about $P_{n,\mathbf{q}}$ and its expectation $\mathsf{P}_{n,\mathbf{q}}$ that have been established using the probabilistic methods in Section 7, but they are limited to quadtrees and the standard k-d trees in the case $k = 2$, and do not extend directly to higher dimensions (actually, to $s > 1$). However, assuming that $\mathbb{E}\left[P_{n,\mathbf{q}}\right]/n^\alpha$ has a limit $f(\mathbf{q})$ as $n \to \infty$, we can derive the conditions that $f(\mathbf{q})$ must satisfy and give a closed form for it in the case of several different multidimensional trees. In particular, for a large family of multidimensional trees, we have

$$f(\mathbf{q}) = \nu_{\mathbf{u}(\mathbf{q})} \prod_{i:\, q_i \neq *} h(q_i) = \nu_{\mathbf{u}(\mathbf{q})} \left(\prod_{i:\, q_i \neq *} q_i(1-q_i)\right)^{\alpha/2}, \tag{24}$$

with α the exponent in the expected cost $\Theta(n^\alpha)$ of random partial matches in the corresponding data structure, and

$$\nu_{\mathbf{u}(\mathbf{q})} = \beta_{\mathbf{u}(\mathbf{q})} \frac{\Gamma^s(\alpha+2)}{\Gamma^{2s}(\alpha/2+1)}.$$

where s is the number of specified coordinates in $\mathbf{q}$, as usual, and $\beta_{\mathbf{u}(\mathbf{q})}$ is the constant factor in the leading term $\beta_{\mathbf{u}(\mathbf{q})} n^\alpha$ of the expected cost

of random partial matches[5] (cf. Theorem 11 in Section 7). The result requires the assumption that $f(\mathbf{q}) = \lim_{n\to\infty} \mathbb{E}\left[\hat{P}_{n,\mathbf{q}}\right]/n^{\alpha}$ exists. Under that assumption, however, it holds for any dimension k, any query $\mathbf{q} \in (0,1)^k$, and any random relaxed quad k-d tree, a family that includes relaxed k-d trees and quadtrees as particular instances. The result also holds for standard k-d trees (of any dimension).

The pervasiveness of the "core" $\left(\prod_i q_i(1-q_i)\right)^{\alpha/2}$ led the authors of [27] (two of us among them) to conjecture that (24) would hold for all data structures such that $\alpha > 1-s/k$; for example, squarish k-d trees [25] or standard k-d tries [44], which have $\alpha = 1 - s/k$, were explicitly excluded from the conjecture. Nevertheless, there is ample empirical evidence to the contrary.

However, in [26] it was shown that (24) does not hold for standard k-d-t trees if $t > 0$, although no closed form for $f(\mathbf{q})$ could be obtained for these trees.

It is also worth mentioning that, despite a quite different context, these heuristic techniques discussed here, complemented with probabilistic arguments to prove the convergence in distribution of the random variable of interest (a suitably normalized version of it) to a limit distribution, have allowed the rigorous analysis of the expected cost of several non-trivial variants of quickselect [58,61].

The use of heuristic techniques for the analysis of the expected cost of partial matches with a fixed query originated with [24], which considered relaxed and standard k-d trees of arbitrary dimension $k \geqslant 2$ but queries with exactly $s = 1$ specified coordinates.

Around the same time, Curien and Joseph [16], Broutin et al. [10,11], and, building upon these papers, Curien [20] studied, with rigorous probabilistic methods, quadtrees and standard k-d trees for $k = 2$ (and hence $s = 1$), as we have already mentioned. The more general cases with $s > 1$ were later tackled using the heuristic methods discussed here, in a series of papers by two of the authors and other coauthors: relaxed and standard k-d trees [27], relaxed k-d-t trees [26], quadtrees [29] and quad k-d trees [32], the last paper covering several previous results as particular cases. A common feature of all these papers, which sets them apart from other works on fixed partial match queries, is that they consider $\mathsf{P}_{n,\mathbf{r}} = \mathbb{E}\left[P_{n,\mathbf{r}}\right]$. Under the assumption that data points are uniformly distributed in $[0,1]^k$, we have $\mathcal{P}_{n,\mathbf{r}} \sim \mathcal{P}_{n,\mathbf{q}}$ as long as $\mathbf{r}/n \to \mathbf{q} \in (0,1)^k$ when $n \to \infty$ (see our discussion on page 18 in Section 3).

In the remainder of this section, we will illustrate these techniques for the case of relaxed k-d trees, which is arguably the easiest. This case is comparatively easier than others (e.g., standard k-d trees or quadtrees), but it captures at the same time the essence of the technique and the main difficulties involved. In particular, showing that $f(\mathbf{q})$ given in (24) is the unique solution to the partial differential equation with given constraints that must be satisfied.

For relaxed k-d trees, the symmetries the data structure, of the probabilistic model that generates the data and the uniform distribution of the discriminants entail that we can assume w.l.o.g. that $\mathbf{r} = (r_0, r_1, \dots, r_{s-1}, *, *, \dots, *)$. For the analysis, we will further assume that $r_i = q_i \cdot n + o(n)$ for $0 < q_i < 1$ and $0 \leqslant i < s$, although it is possible to study the expected cost $\hat{\mathsf{P}}_{n,\mathbf{q}}$ when a few, say $s_0 \leqslant s$, coordinates are either $q_i = 0$ or $q_i = 1$ (analogously, when $r_i = o(n)$ or $r_i = n - o(n)$); in that case, the exponent α changes, as it will depend on s, k and s_0, but most of the analysis proceeds in the same way.

The first step is to write a recurrence for $\hat{\mathsf{P}}_{n,\mathbf{r}}$. To that end, one must separate the cases in which the root $\langle \mathbf{x}, i\rangle$ discriminates according to a specified coordinate, that is, when $0 \leqslant i < s$, from the cases in which i is not specified, $s \leqslant i < K$. The recurrence is, ultimately, of the form

$$\hat{\mathsf{P}}_{n,\mathbf{r}} = 1 + \frac{1}{n}\sum_{0\leqslant j<n}\sum_{i=0}^{k-1}\sum_{\mathbf{r},\mathbf{r}'} \pi(i,n,j,\mathbf{r},\mathbf{r}')\cdot \hat{\mathsf{P}}_{\mathrm{j},\mathbf{r}'},$$

where $\pi(i,n,j,\mathbf{r},\mathbf{r}')$ is the average number of recursive calls made by the algorithm on a subtree of size j with rank vector $\mathbf{r}'$, given that the input k-d tree is of size n, the query has (rescaled) rank vector $\mathbf{r}$, and the root discriminates with respect to i. It is reminiscent of the recurrences that we have seen for random partial match queries, but the weights $\pi(i,n,\mathbf{r},\mathbf{r}')$ are obviously more intricate. The final recurrence in full detail is indeed quite complicated; but its structure is not difficult to understand. The same can be said when we study the corresponding recurrences for quadtrees, quad k-d trees, etc., albeit the recurrences for those cases are even more daunting.

The key to solving these recurrences is to prove (or to assume) that $\mathsf{P}_{n,\mathbf{r}}/\mathrm{n}^{\alpha}$ tends to the limit $f(\mathbf{z}) \neq 0$ when $n \to \infty$ and $\mathbf{r}/n \to \mathbf{z}$. Using the classical tools of asymptotic analysis, as in the proof of the CMT, we can then translate the recurrence for $\mathsf{P}_{n,\mathbf{r}}$ to an integral equation for $f(\mathbf{z})$:

$$f(z_0,\dots,z_{s-1}) = \lambda\sum_{i=0}^{s-1}\left\{ z_i^{\alpha+1}\int_{z_i}^{1} f(z_0,\dots,z_{i-1},z,z_{i+1},\dots)\,\frac{dz}{z^{\alpha+2}} + (1-z_i)^{\alpha+1}\int_0^{z_i} f(z_0,\dots,z,z_{i+1},\dots)\,\frac{dz}{(1-z)^{\alpha+2}}\right\}, \quad (25)$$

where $\lambda := (\alpha+2)/2s$, and f is subject to the following "boundary" conditions: (1) $f(\mathbf{z})$ is symmetric, that is, $f(z_0,\dots,z_{s-1}) = f(z_{\sigma(0)},\dots,z_{\sigma(s-1)})$ for any permutation σ;(2) $f(z_0,\dots,z_i,\dots,z_{s-1}) = f(z_0,\dots,1-z_i,\dots,z_{s-1})$ for any $z_i \in (0,1)$; (3) $\lim_{z_i\to 0^+} f(z_0,\dots,z_{s-1}) = \lim_{z_i\to 1^-} f(z_0,\dots,z_{s-1}) = 0$ for any z_i, and 4) $\int_0^1\cdots\int_0^1 f(\mathbf{y})\,d\mathbf{y} = \beta$, with β the constant factor that depends only on the pattern $\mathbf{u}(\mathbf{q})$ (or just s and k) in the expected cost $\sim \beta n^{\alpha}$ of a **random** partial match. All the "boundary" conditions must hold because of the symmetries of the problem, and by the definition; for example, if we average $\mathsf{P}_{n,\mathbf{r}}$ for all possible n^s $\mathbf{r}$'s we must get the expected cost of a partial match with a random query $\hat{\mathsf{P}}_n$, hence the value of definite integral in our item (4) must be β.

If there is only one specified coordinate, we will have an integral equation in a single variable, which can easily be transformed into a differential equation for $f(z)$; the differential equation is relatively simple to solve (it is a second-order linear ODE) and its uniqueness is immediate. It turns out to be

$$f(z) = \beta\cdot\frac{\Gamma(\alpha+2)}{\Gamma^2(\alpha/2+1)}\cdot h(z) = \beta\cdot\frac{\Gamma(\alpha+2)}{\Gamma^2(\alpha/2+1)}\cdot(z\cdot(1-z))^{\alpha/2}.$$

But if $s > 1$, we do not have an ODE. Instead, we get a PDE, the solution of which looks much harder, not to speak of establishing its uniqueness. The symmetries of the problem (of the data structure and of the probabilistic model) come to rescue, since we can assume that

$$f(z_0,\dots,z_{s-1}) = \phi(z_0)\cdot\cdots\cdot\phi(z_{s-1}),$$

with $\phi(z) = \phi(1-z)$, $\lim_{z\to 0^+}\phi(z) = \lim_{z\to 1^-}\phi(z) = 1$ and

$$\phi(z) = \frac{\alpha+2}{2}\left(z^{\alpha+1}\int_z^1 \phi(u)\frac{du}{u^{\alpha+2}} + (1-z)^{\alpha+1}\int_0^z \phi(u)\frac{du}{(1-u)^{\alpha+2}}\right),$$

leading to the ODE

$$z(1-z)\phi'' + (\alpha-\lambda)(2z-1)\phi' + \alpha\phi = 0,$$

and the initial condition $\phi(0) = 0$, which gives $\phi(z) = C\cdot h(z) = C\cdot(z(1-z))^{\alpha/2}$, for a constant C to be determined.

Constraint (4) allows us to fix the value of C, since $\beta = C^s\left(\int_0^1 h(u)\,du\right)^s$. To prove the uniqueness of the solution to the integral Eq. (25) plus the four constraints on f, we use the fact that the linear homogeneous PDE satisfied by the function f has all real-analytic coefficients in the domain $(0,1)^s$ and, in addition, that the highest derivative in the PDE is $\partial^{2s} f/\partial z_0^2\cdots\partial z_{s-1}^2$ and its coefficient $\prod_{0\leqslant i<s} z_i(1-z_i)$ is clearly always positive in $(0,1)^s$. Hence, the PDE is elliptic, and by Holmgren's theorem, any solution is real-analytic. Applying Cauchy-Kovalevskaya's theorem, it follows that it must be unique, the theorem

[5] Eventually, $\nu_{\mathbf{u}(\mathbf{q})} = \nu_{s,k}$ if the pattern of the query $\mathbf{q}$ does not matter, as is the case for quadtrees and several other multidimensional trees.

guarantees that there is a unique real-analytic solution (see, for instance, [43,88]). Altogether, these results justify the uniqueness of the solution and the representation of f in separable variables.

9. Conclusions and perspectives

We have reviewed, through some selected examples, the major developments in the analysis of multidimensional data structures, in particular, the analysis of partial match searches in k-d trees and quadtrees, especially focusing on a few general mathematical tools that have proved successful in such contexts. The usefulness of such tools, from discrete sequences to continuous functions, from classical analysis to modern stochastic processes, and from big-Oh estimates to finer approximations, goes far beyond multidimensional data structures, as already witnessed by the applications described in the cited references. We believe that these tools will also be valuable in other problems of a similar nature: for example, the analysis of other types of associative queries in hierarchical multidimensional data structures (not limited to trees).

Despite four decades of progress, many questions remain open, and new analytical approaches are needed. We highlight several directions that we believe merit further investigation.

- *Finer asymptotic approximations*: How can we improve the existing $\Theta(n^{\alpha})$ asymptotic estimates to asymptotically equivalent approximations? Is the $\Theta(1)$-term asymptotic to a constant or to a periodic function in nature?
- *Deeper stochastic properties*: Very few results are known beyond the expected cost. In particular, how can we characterize the asymptotic variance and the limiting distribution of the cost of partial match queries in tree structures for data in dimensions higher than two?
- *Broader classes of multidimensional data structures*: How can we extend the average-case analysis to other classes of multidimensional data structures?
- *More general associative queries*: Associative queries such as orthogonal range queries find more applications than partial match queries in practice. The prototypical analysis carried out for the latter may serve as fundamental building blocks for the former; see [13,30]. For more complex region queries, the analysis can often be reduced to that of orthogonal range queries with respect to the smallest hyperrectangle containing the query region.

Dedication

This work has been written in honor of the distinguished career of Josep Díz as editor-in-chief of this journal. During his professional career, Josep has been a key figure in the development of the Theoretical Computer Science group at the Polytechnic University of Catalonia (home to two of the authors). We have greatly benefited from his being a fundamental driving force for TCS in Europe since his early years. He has made numerous contributions to the development of TCS in Europe, including, among others, his many valuable services to EATCS and the scientific events organized by the association, his service in the European ACM chapter, and his role as one of the leaders of the extremely successful series of ALCOM research projects funded by the European Union in the 1990s and early 2000s.

Many thanks Josep! Desitjem que t'agradi aquest *review*. Salut!

Funding

The work of Amalia Duch and Conrado Martínez has been supported by funds from the MOTION Project (Project PID2020-112581GB-C21) of the Spanish Ministry of Science and Innovation MCIN/AEI/10.13039/501100011033.

Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Data availability

No data was used for the research described in the article.

References

[1] L. Ambrosio, N. Gigli, G. Savaré, Gradient flows in metric spaces and in the space of probability measures, in: Lectures in Mathematics ETH Zürich, second ed, Birkhäuser Verlag, Basel, 2008.
[2] N. Bereczky, A. Duch, K. Németh, S. Roura, Quad-k-d trees, in: Alberto Pardo, Alfredo Viola (Eds.), Proceedings of the 11th Latin American Theoretical Informatics Conference (LATIN), vol. 8392 of Lecture Notes in Computer Science, p. Springer-Verlag, 2014, pp. 743–754.
[3] N. Bereczky, A. Duch, K. Németh, S. Roura, Quad-kd trees: a general framework for kd trees and quad trees, Theor. Comput. Sci. 616 (2016) 126–140.
[4] J.L. Bentley, Multidimensional binary search trees used for associative searching, Commun. ACM 18 (9) (1975) 509–517.
[5] J.L. Bentley, Decomposable searching problems, Inf. Process. Lett. 8 (5) (1979) 133–136.
[6] J. Bertoin, Random fragmentation and coagulation processes, in: Cambridge Studies in Advanced Mathematics, vol. 102 of Cambridge University Press, Cambridge, 2006.
[7] J.L. Bentley, J.H. Friedman, Data structures for range searching, ACM Comput. Surv. 11 (4) (1979) 397–409.
[8] P. Billingsley, Convergence of probability measures, in: Wiley Series in Probability and Statistics: Probability and Statistics, second ed, John Wiley & Sons, Inc., New York, 1999. A Wiley-Interscience Publication.
[9] S. Berchtold, D.A. Keim, H.-P. Kriegel, The X-tree: an index structure for high-dimensional data, in: T.M. Vijayaraman, A.P. Buchmann, C. Mohan, N.L. Sarda (Eds.), VLDB'96, Proceedings of 22th International Conference on Very Large Data Bases, September 3-6, 1996, Mumbai (Bombay), India, Morgan Kaufmann, 1996, pp. 28–39.
[10] N. Broutin, R. Neininger, H. Sulzbach, Partial match queries in random quadtrees, in: Y. Rabani (Ed.), Proceedings of the 23Rd Annual ACM-SIAM Symposium on Discrete Algorithms (SODA), 2012, pp. 1056–1065.
[11] N. Broutin, R. Neininger, H. Sulzbach, A limit process for partial match queries in random quadtrees and 2-d trees, Ann. Appl. Probab. 23 (6) (2013) 2560–2603.
[12] W.A. Burkhard, Hashing and trie algorithms for partial match retrieval, ACM Trans. Database Syst. 1 (2) (1976) 175–187.
[13] P. Chanzy, L. Devroye, C. Zamora-Cura, Analysis of range search for random k-d trees, Acta Inform. 37 (2001) 355–383.
[14] H.-H. Chern, H.-K. Hwang, Partial match queries in random quadtrees, SIAM J. Comput. 32 (2003) 904–915.
[15] H.-H. Chern, H.-K. Hwang, Partial match queries in random k-d trees, SIAM J. Comput. 35 (6) (2006) 1440–1466.
[16] N. Curien, A. Joseph, Partial match queries in two-dimensional quadtrees: a probabilistic approach, Adv. Appl. Probab. 43 (2011) 178–194.
[17] W. Cunto, G. Lau, P. Flajolet, Analysis of Kdt-trees: Kd-trees improved by local reorganisations, in: F. Dehne, J.R. Sack, N. Santoro (Eds.), Workshop on Algorithms and Data Structures (WADS'89). Lecture Notes in Computer Science, vol. 382, Springer-Verlag, 1989, pp. 24–38.
[18] T.H. Cormen, C.E. Leiserson, R.L. Rivest, C. Stein, Introduction to Algorithms, MIT Press, 2022.
[19] E. Chávez, G. Navarro, R. Baeza-Yates, J.L. Marroquín, Searching in metric spaces, ACM Comput. Surv. 33 (3) (2001) 273–321.
[20] N. Curien, Strong convergence of partial match queries in random quadtrees, Comb. Probab. Comput. 21 (5) (2012) 683–694.
[21] A. Duch Brown, Design and Analysis of Multidimensional Data Structures. (Ph.D. thesis), Universitat Politècnica de Catalunya, 2004.
[22] M. de Berg, M. Roeloffzen, B. Speckmann, Kinetic compressed quadtrees in the black-box model with applications to collision detection for low-density scenes, in: L. Epstein, P. Ferragina (Eds.), Algorithms – ESA 2012, Berlin, Heidelberg. Springer Berlin Heidelberg, 2012, pp. 383–394.
[23] A. Duch, V. Estivill-Castro, C. Martínez, Randomized k-dimensional binary search trees, in: K.Y. Chwa, O.H. Ibarra (Eds.), 9th International Symposium on Algorithms and Computation (ISAAC), vol. 1533 of Lecture Notes in Computer Science, p. Springer-Verlag, 1998, pp. 199–208, Heidelberg.
[24] A. Duch, R.M. Jiménez, C. Martínez, Selection by rank in k-dimensional binary search trees, Random Struct. & Alg. 45 (1) (2014) 14–37.
[25] L. Devroye, J. Jabbour, C. Zamora-Cura, Squarish k-d trees, SIAM J. Comput. 30 (2000) 1678–1700.
[26] A. Duch, G. Lau, Partial match queries in relaxed k-dt trees, in: C. Martínez, M.D. Ward (Eds.), Proceedings of the 14th ACM-SIAM Workshop on Analytic Algorithmics and Combinatorics (ANALCO), SIAM, 2017, pp. 131–138.
[27] A. Duch, G. Lau, C. Martínez, On the cost of fixed partial match queries in k-d trees, Algorithmica 75 (4) (2016) 684–723.
[28] A. Duch, G. Lau, C. Martínez, Random partial match in quad-k-d trees, in: E. Kranakis, G. Navarro, E. Chávez (Eds.), Proceedings of the 12th Latin American Theoretical Informatics Conference (LATIN) of Lecture notes in computer science, vol. 9644 Springer-Verlag, 2016, pp. 376–389.
[29] A. Duch, G. Lau, C. Martínez, Fixed partial match queries in quadtrees, in: J.A. Fill, M.D. Ward (Eds.), Proceedings of the 29Th International Meeting on Probabilistic, Combinatorial and Asymptotic Methods for the Analysis of Algorithms (AofA) Leibniz international proceedings in informatics (LIPIcs), vol. 110 of 2018, pp. :20:1–:20:18.

[30] A. Duch, C. Martínez, On the average performance of orthogonal range search in multidimensional data structures, J. Algorithms 44 (1) (2002) 226–245.
[31] A. Duch, C. Martínez, Updating relaxed k-d trees, ACM Trans. Algorithms 6 (1) (2009) :4:1–:4:24.
[32] A. Duch, C. Martínez, On the expected cost of partial match queries in random quad-k-d trees, La Matematica. Off. J. Assoc. Women Math. 3 (2024) 385–416.
[33] A. Duch, C. Martínez, M. Pons, S. Roura, Median and hybrid median k-dimensional trees, in: A. Castañeda, F. Rodríguez-Henríquez (Eds.), Proceedings of the 15th Latin American Theoretical Informatics Conference (LATIN) Lecture notes in computer science, vol. 13568 of Heidelberg. Springer-Verlag, 2022, pp. 38–53.
[34] eds, D.P. Mehta, S. Sahni, Handbook of Data Structures and Applications, Chapman and Hall/CRC, 2017.
[35] A. Duch, Randomized insertion and deletion in point quad trees, in: R. Fleischer, G. Trippen (Eds.), Proceedings of the 15Th International Symposium on Algorithms and Computation (ISAAC) Lecture notes in computer science, vol. 3341 of Springer-Verlag, 2004, pp. 415–426.
[36] R.A. Finkel, J.L. Bentley, quad trees: a data structure for retrieval on composite key, Acta Inform. 4 (1) (1974) 1–9.
[37] P. Flajolet, X. Gourdon, P. Dumas, Mellin transforms and asymptotics: harmonic sums, Theor. Comput. Sci. 144 (1–2) (1995) 3–58.
[38] P. Flajolet, X. Gourdon, P. Dumas, Mellin transforms and asymptotics: harmonic sums, Theor. Comput. Sci. 144 (1–2) (1995) 3–58.
[39] P. Flajolet, P. Grabner, P. Kirschenhofer, H. Prodinger, R.F. Tichy, Mellin transforms and asymptotics: digital sums, Theor. Comput. Sci. 123 (2) (1994) 291–314.
[40] P. Flajolet, G. Gonnet, C. Puech, J.M. Robson, Analytic variations on quad trees, Algorithmica 10 (1993) 473–500.
[41] P. Flajolet, T. Lafforgue, Search costs in quadtrees and singularity perturbation asymptotics, Discrete Comput. Geom. 12 (1994) 151–175.
[42] P. Flajolet, G. Labelle, L. Laforest, B. Salvy, Hypergeometrics and the cost structure of quadtrees, Random Struct. & Alg. 7 (2) (1995) 117–144.
[43] G.B. Folland, Introduction to Partial Differential Equations, second ed, Princeton University Press, 1995.
[44] P. Flajolet, C. Puech, Partial match retrieval of multidimensional data, J. ACM 33 (2) (1986) 371–407.
[45] P. Flajolet, R. Sedgewick, Mellin transforms and asymptotics: finite differences and Rice's integrals, Theor. Comput. Sci. 144 (1–2) (1995) 101–124.
[46] P. Flajolet, R. Sedgewick, Analytic Combinatorics, Cambridge University Press, 2009.
[47] P. Goswami, F. Erol, R. Mukhi, R. Pajarola, E. Gobbetti, An efficient multi-resolution framework for high quality interactive rendering of massive point clouds using multi-way kd-trees, The Visual Computer 29 (1) (2013) 69–83.
[48] V. Gaede, O. Günther, Multidimensional access methods, ACM Comput. Surv. 30 (2) (1998) 170–231.
[49] A. Guttman, R-trees: a dynamic index structure for spatial searching, ACM SIGMOD Record 14 (2) (1984) 47–57.
[50] J. Hadamard, The Mathematician's Mind: the Psychology of Invention in the Mathematical Field, Princeton University Press, 1965.
[51] V. Havran, Heuristic ray shooting algorithms, in: Doctoral Dissertation, Czech Technical University, 2000.
[52] S. Hussain, H. Grahn, Fast kd-tree construction for 3d-rendering algorithms like ray tracing, in: G. Bebis, R. Boyle, B. Parvin, D. Koracin, N. Paragios, SYEDA.-M. Tanveer, T. Ju, Z. Liu, S. Coquillart, C. Cruz-Neira, T. Müller, T. Malzbender (Eds.), Advances in Visual Computing, Berlin, Heidelberg. Springer Berlin Heidelberg, 2007, pp. 681–690.
[53] S. Janson, S. Kaijser, Higher moments of banach space valued random variables, Mem. Amer. Math. Soc. 238 (1127):vii + 110, 2015).
[54] K.I. Lin, H. Jagadish, C. Faloutsos, The TV-tree: an index structure for high-dimensional data, The VLDB J. 3 (4) (1994) 517–543.
[55] G.S. Lau Laynes-Lozada, Analysis of Partial Match Queries in Multidimensional Search Trees. (Ph.D. thesis), Universitat Politècnica de Catalunya, 2019.
[56] H.M. Mahmoud, Evolution of Random Search Trees, Wiley-Interscience Series in Discrete Mathematics and Optimization, 1991.
[57] Y. Manolopoulos, A. Nanopoulos, A.N. Papadopoulos, Y. Theodoridis, R-Trees: Theory and Applications: Theory and Applications, Springer Science & Business Media, 2006.
[58] C. Martínez, M. Nebel, S. Wild, Sesquickselect: one and a half pivots for cache-efficient selection, in: M. Mishna, J.I. Munro (Eds.), Proceedings of the 16th ACM-SIAM Workshop on Analytic Algorithmics and Combinatorics (ANALCO), SIAM, 2019, pp. 54–66.
[59] C. Martínez, A. Panholzer, H. Prodinger, Partial match queries in relaxed multidimensional search trees, Algorithmica 29 (1–2) (2001) 181–204.
[60] C. Martínez, A. Panholzer, H. Prodinger, Partial match queries in relaxed multidimensional search trees, Algorithmica 29 (1–2) (2001) 181–204.
[61] C. Martínez, D. Panario, A. Viola, Adaptive sampling strategies for quickselect, ACM Trans. Algor. 6 (3):53:1–53:32 + appendices (14 pages), 2010).
[62] C. Martínez, S. Roura, Randomized binary search trees, J. ACM 45 (2) (1998) 288–323.
[63] C. Martínez, S. Roura, Optimal sampling strategies in quicksort and quickselect, SIAM J. Comput. 31 (3) (2001) 683–705.
[64] R. Neininger, Asymptotic distributions for partial match queries in k-d trees, Random Struct. Algorithms 17 (3–4) (2000) 403–427.
[65] J. Nievergelt, H. Hinterberger, K.C. Sevcik, The grid file: an adaptable symmetric multikey file structure, ACM Trans. Database Syst. 1 (9) (1984) 38–71.
[66] R. Neininger, L. Rüschendorf, Limit laws for partial match queries in quadtrees, Ann. Appl. Probab. 11 (2) (2001) 452–469.
[67] R. Neininger, L. Rüschendorf, A general limit theorem for recursive algorithms and combinatorial structures, Ann. Appl. Probab. 14 (1) (2004) 378–418.
[68] R. Neininger, H. Sulzbach, On a functional contraction method, Ann. Probab. 43 (4) (2015) 1777–1822.
[69] M. Nebel, S. Wild, C. Martínez, Analysis of pivot sampling in dual-pivot quicksort: a holistic analysis of yaroslavskiy's partition scheme, Algorithmica 75 (4) (2016) 632–683.
[70] R.B. Paris, Incomplete Gamma and related functions, In F.W.J. Olver, D.W. Lozier, R.F. Boisvert, C.W. Clark (Eds.), NIST Handbook of Mathematical Functions, Chapter 8, Cambridge University Press, 2010.
[71] M. Regnier, Analysis of the grid file algorithms, BIT 25 (2) (1985) 335–357.
[72] F. Ronald, J. Nievergelt, N. Pippenger, R. Strong, Extendible hashing—a fast access method for dynamic files, ACM Trans. Database Syst. 4 (3) (1979) 315–344.
[73] U. Rösler, A limit theorem for "quicksort", RAIRO Inform. Théor. Appl. 25 (1) (1991) 85–100.
[74] S. Roura, Improved master theorems for divide-and-conquer recurrences, J. ACM 48 (2) (2001) 170–205.
[75] H. Samet, Applications of Spatial Data Structures, Addison-Wesley, 1990.
[76] H. Samet, The Design and Analysis of Spatial Data Structures, Addison-Wesley, 1990.
[77] H. Samet, Foundations of Multidimensional and Metric Data Structures, Morgan Kaufmann, San Francisco, CA, 2006.
[78] W. Schachinger, The variance of a partial match retrieval in a multidimensional symmetric trie, Random Struct. & Alg. 7/1 (1995) 81–95.
[79] W. Schachinger, Limiting distributions for the costs of partial match retrievals in multidimensional tries, Random Struct. & Alg. 17 (2000) 428–459.
[80] W. Schachinger, Distributional results for costs of partial match queries in asymmetric k-dimensional tries, SIAM J. Comput. 33 (4) (2004) 952–983.
[81] R. Sedgewick, P. Flajolet, An Introduction to the Analysis of Algorithms, second ed, Addison-Wesley Professional, Boston, Massachusetts, 2013.
[82] B. Seeger, H.P. Kriegel, Techniques for design and implementation of spatial access methods, in: Proceedings of the Fourteenth International Conference on Very Large Databases, 1988, pp. 360–371.
[83] J. Schauer, A. Nüchter, Efficient point cloud collision detection and analysis in a tunnel environment using kinematic laser scanning and k-d tree search, Int. Arch. Photogramm. Remote Sens. Spat. Inf. Sci. XL-3 (2014) 289–295.
[84] J. Schauer, A. Nüchter, Collision detection between point clouds using an efficient kd tree implementation, Adv. Eng. Inform. 29 (3) (2015) 440–458.
[85] C. Villani, Optimal transport, in: Grundlehren Der Mathematischen Wissenschaften [Fundamental Principles of Mathematical Sciences], vol. 338 of Springer-Verlag, Berlin, 2009. Old and new.
[86] I. Wald, S. Boulos, P. Shirley, Ray tracing deformable scenes using dynamic bounding volume hierarchies, ACM Trans. on Graph. (TOG) 25 (1) (2006) 1–18.
[87] P. Zezula, G. Amato, V. Dohnal, M. Batko, Similarity Search, Springer-Verlag, 2006.
[88] D. Zwillinger, Handbook of Differential Equations, third ed, Academic Press, 1997.