



UNIVERSITAT POLITÈCNICA
DE CATALUNYA
BARCELONATECH

Analysis of partial match queries in multidimensional search trees

by

Gustavo Salvador Lau Laynes-Lozada

ADVERTIMENT La consulta d'aquesta tesi queda condicionada a l'acceptació de les següents condicions d'ús: La difusió d'aquesta tesi per mitjà del repositori institucional UPCCommons (<http://upcommons.upc.edu/tesis>) i el repositori cooperatiu TDX (<http://www.tdx.cat/>) ha estat autoritzada pels titulars dels drets de propietat intel·lectual **únicament per a usos privats** emmarcats en activitats d'investigació i docència. No s'autoritza la seva reproducció amb finalitats de lucre ni la seva difusió i posada a disposició des d'un lloc aliè al servei UPCCommons o TDX. No s'autoritza la presentació del seu contingut en una finestra o marc aliè a UPCCommons (*framing*). Aquesta reserva de drets afecta tant al resum de presentació de la tesi com als seus continguts. En la utilització o cita de parts de la tesi és obligat indicar el nom de la persona autora.

ADVERTENCIA La consulta de esta tesis queda condicionada a la aceptación de las siguientes condiciones de uso: La difusión de esta tesis por medio del repositorio institucional UPCCommons (<http://upcommons.upc.edu/tesis>) y el repositorio cooperativo TDR (<http://www.tdx.cat/?locale-attribute=es>) ha sido autorizada por los titulares de los derechos de propiedad intelectual **únicamente para usos privados enmarcados** en actividades de investigación y docencia. No se autoriza su reproducción con finalidades de lucro ni su difusión y puesta a disposición desde un sitio ajeno al servicio UPCCommons. No se autoriza la presentación de su contenido en una ventana o marco ajeno a UPCCommons (*framing*). Esta reserva de derechos afecta tanto al resumen de presentación de la tesis como a sus contenidos. En la utilización o cita de partes de la tesis es obligado indicar el nombre de la persona autora.

WARNING On having consulted this thesis you're accepting the following use conditions: Spreading this thesis by the institutional repository UPCCommons (<http://upcommons.upc.edu/tesis>) and the cooperative repository TDX (<http://www.tdx.cat/?locale-attribute=en>) has been authorized by the titular of the intellectual property rights **only for private uses** placed in investigation and teaching activities. Reproduction with lucrative aims is not authorized neither its spreading nor availability from a site foreign to the UPCCommons service. Introducing its content in a window or frame foreign to the UPCCommons service is not authorized (*framing*). These rights affect to the presentation summary of the thesis as well as to its contents. In the using or citation of parts of the thesis it's obliged to indicate the name of the author.



UNIVERSITAT POLITÈCNICA
DE CATALUNYA
BARCELONATECH

Analysis of Partial Match Queries in Multidimensional Search Trees

Article-based thesis submitted to the
Department of Computer Science
Universitat Politècnica de Catalunya

in partial fulfillment of the
requirements for the degree of
Ph.D. in Computer Science

by

Gustavo Salvador Lau Laynes-Lozada

under the supervision of
Amalia Duch Brown and
Conrado Martínez Parra

Barcelona, October 2019

*Since I am an imperfect scholar and,
even more certainly, a fallible human being,
I will inevitably be making factual errors,
drawing some unjustifiable conclusions,
and perhaps passing along my opinions as facts.*

— Neil Postman, *The End of Education*

Abstract

A variety of applications need to manage collections of multidimensional data, where each object is identified by a point in some real or abstract space, a prime example being Geographical Information Systems. These applications often require multidimensional data structures that allow associative queries—those that specify conditions for more than one coordinate—in addition to the traditional operations of insert, update, delete and exact search. One of the main types of associative queries is partial match (PM), where only some coordinates are specified and the goal is to determine which objects match them. Partial match queries are particularly important because their analysis forms the basis of the analysis of other types of associative queries, such as orthogonal range queries (which points fall inside a given (hyper)rectangular area?), region queries (e.g. which points are within a given distance of some other given point?) or nearest neighbor queries (find the k closest neighbors to a given point). In this thesis we analyze in depth the average performance of partial match queries in several representative multidimensional search trees, a significant subclass of multidimensional data structures.

Multidimensional search trees, in particular quad trees and K -d trees, were introduced in the mid 1970s as a generalization of binary search trees. Partial match queries on them are answered by performing a recursive traversal of some subtrees. For several decades their analysis in multidimensional search trees was done with the important, and frequently implicit, assumption that in each recursive call new coordinates of the partial match query were generated randomly. The reason for this simplifying assumption was that, for expected costs, such analysis is equivalent to analyzing the performance of the partial match algorithm when the input is a random PM query. Early in this decade, a few teams started the average-case analysis of partial match queries without such assumption: the specified coordinates of the query remain fixed throughout all the recursive calls. These queries are called fixed PM queries. The goal of this relatively recent approach is to analyze the performance of the partial match algorithm, but with the quantities of interest depending on the particular query \mathbf{q} given as input. The analysis of fixed PM queries, together with that of random ones—which turns out to play an important role in the analysis of the former—give us a very detailed and precise description of the performance of the partial match algorithm that could be extended to other relevant associative queries.

The main contribution of this thesis is to deepen and generalize previous work done in the average-case analysis of partial match queries in several types of multidimensional search trees. In particular, our focus has been the analysis of fixed PM queries. Our results about them generalize previous results which covered the case where only one coordinate is specified in the PM query—and for any dimension—or the case of 2-dimensional data structures. Using a combinatorial approach, different to the probabilistic approaches used by other researchers, we obtain asymptotic formulas for the expected cost of fixed PM queries in relaxed and standard K -d trees. We establish that, in both cases, the expected cost satisfies a common pattern in the relationship with the expected cost of random PM queries. Moreover, the same pattern appeared in the analysis, previously done by other researchers, of the expected cost of fixed partial match in 2-dimensional quad trees. Those results led us to conjecture that such formula would be pervasive to describe the expected cost of partial match queries in many different multidimensional trees, assuming some additional technical conditions about the family of multidimensional search trees under consideration. Indeed, we prove this to be the case also for K -dimensional quad trees.

However, we disprove that conjecture for a new variant of K -d trees with local balancing that we define: relaxed K -dt trees. We analyze the expected cost of random PM queries and fixed PM queries in them and, while we do not find a closed-form expression for the expected cost of fixed PM queries, we prove that it cannot be of the same form that we had conjectured.

For random PM queries in both relaxed and standard K -dt trees, we obtain two very general results that unify several specific results that appear scattered across the literature. Finally, we also analyze random PM queries in quad- K -d trees—a generalization of both quad trees and K -d trees—and obtain a very general result that includes as particular cases previous results in relaxed K -d trees and quad trees.

Resumen

Son muchas las aplicaciones en las que se necesita administrar colecciones de datos multidimensionales, en las que cada objeto es identificado por un punto en un espacio real o abstracto; un ejemplo paradigmático son los sistemas de información geográfica. Estas aplicaciones a menudo requieren estructuras de datos multidimensionales que permitan consultas asociativas—aquellas que especifican condiciones para más de una coordenada—además de las operaciones tradicionales de inserción, actualización, eliminación y búsqueda exacta. Uno de los principales tipos de consultas asociativas es la búsqueda parcial, donde solamente se especifican algunas coordenadas y el objetivo es determinar qué objetos coinciden con ellas. Las búsquedas parciales son particularmente importantes porque su análisis forma la base del análisis de otros tipos de consultas asociativas, tales como las consultas por rangos ortogonales (¿qué puntos están dentro de un área (hiper)rectangular dada?), las consultas por región (por ejemplo, dados un punto y una distancia ¿cuáles puntos están a dicha distancia o menos de ese punto?) o las consultas del vecino más cercano (encontrar los k puntos más cercanos a un punto dado). En esta tesis analizamos en profundidad el rendimiento promedio de las búsquedas parciales en varios árboles multidimensionales de búsqueda representativos, los cuales constituyen una subclase significativa de las estructuras de datos multidimensionales.

Los árboles multidimensionales de búsqueda, en particular *quad*trees y árboles K -d, fueron definidos a mediados de la década de los años 1970 como una generalización de los árboles binarios de búsqueda. Las búsquedas parciales en ellos se responden realizando un recorrido recursivo de algunos de sus subárboles. Durante varias décadas su análisis en árboles multidimensionales de búsqueda se hizo con la suposición importante, y con frecuencia implícita, de que en cada llamada recursiva se generaban al azar nuevas coordenadas de la consulta de búsqueda parcial. La razón de esta suposición simplificadora fue que, para los costes esperados, dicho análisis es equivalente a analizar el rendimiento del algoritmo de búsqueda parcial cuando la entrada es una búsqueda parcial aleatoria. A principios de esta década, algunos equipos comenzaron a analizar el caso promedio de búsquedas parciales sin dicha suposición: las coordenadas especificadas de la consulta permanecen fijas a lo largo de todas las llamadas recursivas. Estas consultas se llaman búsquedas parciales fijas. El objetivo de este enfoque reciente es analizar el rendimiento del algoritmo de búsqueda parcial, pero ahora las cantidades de interés dependen de la consulta particular \mathbf{q}

dada como entrada. El análisis de búsquedas parciales fijas, junto con el de las aleatorias—que resulta desempeñar un papel importante en el análisis de las primeras—nos da una descripción muy detallada y precisa del rendimiento del algoritmo de búsqueda parcial que podría ser extendida a otras consultas asociativas relevantes.

La principal contribución de esta tesis es profundizar y generalizar resultados anteriores referentes al análisis en caso medio de búsquedas parciales en varios tipos de árboles multidimensionales de búsqueda. En particular nos enfocamos en el análisis de búsquedas parciales fijas. Nuestros resultados sobre ellas generalizan resultados previos que cubren el caso donde solamente una coordenada es especificada en la búsqueda parcial—y para cualquier dimensión—o el caso de estructuras de datos de dos dimensiones. Usando un enfoque combinatorio, diferente a los enfoques probabilísticos utilizados por otros investigadores, obtenemos fórmulas asintóticas para el costo esperado de búsquedas parciales fijas en árboles K -d relajados y estándares. Establecemos que, en ambos casos, el costo esperado satisface un patrón común en la relación con el costo esperado de búsquedas parciales aleatorias. Además, el mismo patrón apareció en el análisis, previamente hecho por otros investigadores, del costo esperado de búsquedas parciales fijas en quadtrees de dos dimensiones. Esos resultados nos llevaron a conjeturar que tal fórmula sería generalizada para describir el costo esperado de consultas de búsqueda parcial en muchos árboles multidimensionales diferentes, asumiendo algunas condiciones técnicas adicionales sobre la familia de árboles multidimensionales de búsqueda bajo consideración. De hecho, demostramos que este también es el caso en quadtrees de K dimensiones.

Sin embargo, definimos una nueva variante de árboles K -d con reorganización local que cumplen tales condiciones, los árboles K -dt relajados, analizamos el costo esperado de búsquedas parciales aleatorias y fijas en ellos y, aunque no encontramos una expresión cerrada para el costo esperado de las búsquedas parciales fijas, demostramos que no puede ser de la misma forma que habíamos conjeturado.

También obtenemos dos resultados muy generales para búsquedas parciales aleatorias en árboles K -dt relajados y estándares que unifican varios resultados específicos que aparecen dispersos en la literatura. Finalmente, analizamos búsquedas parciales aleatorias en una generalización de quadtrees y árboles K -d, llamada árboles quad- K -d, y obtenemos un resultado general que incluye como casos particulares resultados previos en árboles K -d relajados y quadtrees.

Resum

Són moltes les aplicacions en què es requereix administrar col·leccions de dades multidimensionals, en les quals cada objecte és identificat per un punt en un espai real o abstracte; un exemple paradigmàtic són els sistemes d'informació geogràfica. Aquestes aplicacions fan servir sovint estructures de dades multidimensionals que permetin consultes associatives—aquelles on s'especifiquen condicions per a més d'una coordenada—a més de les operacions tradicionals d'inserció, actualització, eliminació i cerca exacta. Un dels principals tipus de consultes associatives és la cerca parcial, on només s'especifiquen algunes coordenades i l'objectiu és determinar quins objectes coincideixen amb elles. Les consultes de cerca parcial són particularment importants perquè la seva anàlisi forma la base de l'anàlisi d'altres tipus de consultes associatives, com ara les cerques per rangs ortogonals (quins punts estan dins d'una àrea (hiper)rectangular donada?), les consultes per regió (per exemple, donats un punt i una distància, quins punts estan a aquesta distància o menys d'aquest punt?) o les consultes del veí més proper (on cal trobar els k punts més propers a un punt donat). En aquesta tesi analitzem en profunditat el rendiment mitjà de les cerques parcials en arbres multidimensionals de cerca representatius, els quals constitueixen una subclasse significativa de les estructures de dades multidimensionals.

Els arbres multidimensionals de cerca, en particular els quadrees i els arbres K -d, van ser definits a mitjans de la dècada dels anys 1970 com una generalització dels arbres binaris de cerca. Les consultes de cerca parcial s'hi responen realitzant un recorregut recursiu d'alguns subarbres. Durant molts anys l'anàlisi en arbres multidimensionals de cerca es va fer amb la suposició important, i sovint implícita, que en cada crida recursiva es generen a l'atzar noves coordenades de la consulta de cerca parcial. La raó d'aquesta suposició simplificadora va ser que, per als costos mitjans, aquesta anàlisi és equivalent a analitzar el rendiment de l'algorisme de cerca parcial quan l'entrada és una consulta de cerca parcial aleatòria. A principis d'aquesta dècada, alguns equips van començar a analitzar el cas mitjà de cerques parcials sense aquesta suposició: les coordenades especificades de la consulta romanen fixes durant totes les crides recursives. Aquestes consultes s'anomenen cerques parcials fixes. L'objectiu d'aquest enfocament recent és analitzar el rendiment de l'algorisme de cerca parcial, però ara les quantitats d'interès depenen de la consulta particular \mathbf{q} donada com a entrada. L'anàlisi de cerques parcials fixes, juntament amb el de les aleatòries —que té un paper important per a l'anàlisi de les primeres—ens dóna una descripció molt

detallada i precisa del rendiment de l'algorisme de cerca parcial que podria ser estesa a altres consultes associatives rellevants.

La principal contribució d'aquesta tesi és aprofundir i generalitzar resultats previs referents a l'anàlisi en cas mitjà de les cerques parcials en diversos tipus d'arbres multidimensionals de cerca. En particular ens enfocuem en l'anàlisi de les cerques parcials fixes. Els nostres resultats en generalitzen resultats previs els quals cobreixen el cas on només una coordenada està especificada a la cerca parcial—i per a qualsevol dimensió—o el cas d'estructures de dades de dues dimensions. Usant un enfocament combinatori, diferent als enfocaments probabilístics utilitzats per altres investigadors, obtenim fórmules asimptòtiques per al cost esperat de cerques parcials fixes en arbres K -d relaxats i estàndards. Establim que, en tots dos casos, el cost esperat satisfà un patró comú en la relació amb el cost esperat de cerques parcials aleatòries. A més, el mateix patró va aparèixer en l'anàlisi, prèviament fet per altres investigadors, del cost esperat de cerques parcials fixes en quadrees de dues dimensions. Aquests resultats ens van portar a conjecturar que tal fórmula seria general per descriure el cost esperat de consultes de cerca parcial en molts arbres multidimensionals diferents, assumint algunes condicions tècniques addicionals sobre la família d'arbres multidimensionals de cerca sota consideració. De fet, demostrem que aquest és també el cas pels quadrees de K dimensions.

Tanmateix, definim una nova variant de arbres K -d amb equilibri local que compleixen aquestes condicions, els arbres K -dt relaxats, n'analitzem el cost esperat de cerques parcials aleatòries i fixes i, tot i no trobar una expressió tancada per al cost esperat de les cerques parcials fixes, demostrem que no pot ser de la mateixa forma que havíem conjecturat.

També obtenim dos resultats molt generals per a les cerques parcials aleatòries en arbres K -dt relaxats i estàndards, els quals unifiquen diversos resultats específics que apareixen dispersos a la literatura. Finalment, analitzem cerques parcials aleatòries en una generalització de quadrees i arbres K -d, anomenada arbres quad- K -d, i obtenim un resultat general que inclou com a casos particulars resultats previs en arbres K -d relaxats i quadrees.

Acknowledgements

First of all, I would like to express my deepest appreciation to my supervisors, Amalia Duch and Conrado Martínez. I will always be grateful to them for suggesting that I register in the PhD program, welcoming me into their team, teaching me so many things and for all their guidance, help and support. Without them this work would not have been possible at all.

I am deeply indebted to Walter Cunto for introducing me to the analysis of algorithms and giving me the great opportunity of working with him and Philippe Flajolet on my master's thesis back in 1987 at the Simón Bolívar University in Caracas. This PhD thesis can be seen as the continuation of that work.

I am extremely grateful to Robert Sedgewick for his Analytic Combinatorics Coursera course and to my friend Enzo Chiariotti (a.k.a. Enzo Alda) for recommending that I watch it in 2013. My initial reaction was “No way, I have no time for a Coursera course” and I ended up doing a PhD!

I would like to thank the organizers of a special conference in memory of Philippe Flajolet held in 2011. On its website in a presentation about Search Trees given by Conrado, on the slide about my work with Walter and Philippe, there is a footnote stating: “results about variance must be taken with a grain of salt”. That piqued my curiosity and led me to try to meet Conrado. Without that conference I would not have rejoined the analysis of algorithms community after 25 years.

I am also grateful to my friend Edelmira Pasarella for introducing me to Conrado and Amalia and all her logistical support all these years.

Thanks also to my partners and colleagues at Episteme Capital for their support and the flexibility that allowed me to pursue this PhD while working full time with them. Special thanks to Jameel Kassam for his help proofreading my English.

I would like to dedicate this work to my entire family; my life has meaning thanks to them. My dear parents, Rosa and Salvador, my sisters, Elvira and Ivonne, my beloved wife, Xiomara, my sons, Sergi and Roberto, and my nephews, Bernardo, Ramón and Carlos Eduardo as well as all the rest of the family: my utmost thanks to you all.

Gustavo Lau
gustavolau.com

Contents

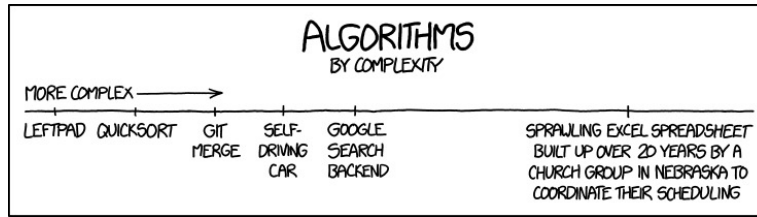
Abstract	i
Resumen	iii
Resum	v
Acknowledgements	vii
I Introduction	1
II Preliminaries	9
1 Associative Queries and Search Trees	11
1.1 Associative Queries	11
1.1.1 Nearest neighbor query	12
1.1.2 Orthogonal Range Query	12
1.1.3 Partial Match Query	13
1.2 Trees and Point Search Trees	14
1.3 Multidimensional Search Trees	18
1.3.1 A Taxonomy	18
1.3.2 Quad trees	20
1.3.3 K -d trees	22
1.3.4 K -dt trees	26
1.3.5 Quad- K -d trees	29
1.4 Search Algorithms	31
2 Probabilistic Model	33
2.1 Random Search Tree Model	33
2.2 Queries and Ranks	37
2.3 Random Variables for Partial Match Queries	38
2.3.1 Random Partial Match Queries	38
2.3.2 Randomized Partial Match Algorithm	38
2.3.3 Fixed Partial Match Queries	40

2.3.4	Random PM Queries vs Randomized PM Algorithm . . .	40
2.3.5	Random PM Queries vs Fixed PM Queries	40
2.3.6	Fixed PM Queries vs Fixed PM Ranks	41
3	Methods	43
3.1	Mathematical Tools	43
3.1.1	Generating functions	43
3.1.2	Random Partial Match Queries	45
3.1.3	Fixed Partial Match Queries	49
3.2	Experiments	50
4	State of the Art	55
4.1	Random Partial Match Queries	55
4.1.1	Standard K -d trees	55
4.1.2	Standard K -dt trees	56
4.1.3	Quad trees	57
4.1.4	Relaxed K -d trees	57
4.1.5	Squarish K -d trees	58
4.2	Fixed Partial Match Queries	58
4.2.1	Quad trees	58
4.2.2	Relaxed K -d trees	59
4.2.3	Standard K -d trees	60
III	Results	61
5	K-d trees	63
5.1	Article: Fixed PM Queries in Relaxed K -d Trees	64
5.2	Article: Fixed PM Queries in K -d Trees	77
6	K-dt trees	121
6.1	Article: PM Queries in Relaxed K -dt trees	123
7	Quad trees	133
7.1	Article: Fixed PM Queries in Quadtrees	133
8	Quad-K-d trees	153
8.1	Article: Random PM in Quad- K -d Trees	153
IV	Conclusions and Open Problems	169
V	Appendices	177
A	Theorems used in proofs	179
A.1	Roura's Continuous Master Theorem	179

A.2 Flajolet and Odlyzko's Transfer Lemma	181
B Proofs	183
B.1 Random PM queries with extreme coordinates in K -dt trees . . .	183
B.2 Lemma for alternative proofs	187
 Bibliography	 191

Part I

Introduction



— Randall Munroe, xkcd.com

Analysis of algorithms is the area of computer science devoted to the study of the amount of resources, or computational complexity, needed for the execution of an algorithm. In this area we assume that an algorithm is correct and we analyze its efficiency. The resources considered are typically storage space and time but we will focus on the latter complexity. Therefore, we want to find a function that expresses how much running time a particular algorithm will take as a function of the input size. We call this the cost function, or simply the cost or time, of the algorithm. To make the analysis independent of the machine, the output of the cost function is neither expressed in seconds, nor in any other standard time unit, but as the number of some specific operations that the algorithm would have to perform. In a large number of cases it is not feasible to find precise formulas and so we find asymptotic ones. Usually this is sufficient as it is of more interest to analyze the algorithms' behavior for large input data sets.

In analysis of algorithms one of the most common approaches to measure the performance of an algorithm is by its cost in the worst case of its possible inputs of a given size. However, in many occasions the worst case is unlikely and the analysis of the average case has more practical value¹. In order to perform the latter we need to know, at least to a very reasonable degree, the probability distribution of the input data parameterized by its size.

This thesis is about the average-case analysis of particular algorithms, partial match (PM) queries, in a specific class of multidimensional data structures, multidimensional search trees. The prime example of multidimensional data is Geographical Information Systems, for instance, the NASA Center for Climate Simulation website: <https://maps.nccs.nasa.gov>. Samet, in his reference book [Sam06], mentions several other applications that require multidimensional data structures such as the following, with references to examples that use multidimensional search trees for indexing the data:

- Pattern recognition, e.g. [LVV03].
- Image processing, e.g. [Ada+09].
- Game programming, e.g. Chapter 16 of [Tre04].

¹If the probability distribution is unknown, changes or if it varies for different applications, hybrid models between worst-case and average-case analysis—like smoothed analysis—can be more useful, see [Rou19] for a review of different approaches to analysis of algorithms.

- Finite-element analysis, e.g. in computational physics [KH07].

Multidimensional data can be point data, specific locations in space, or spatial data, that is lines or regions of two or more dimensions. In this thesis we are going to focus on the former, where each record is identified by a key composed of many fields, that is its coordinates.

For all examples of multidimensional data in this section and the next, we assume that the number of dimensions, K , is 2 and we use the data in Table 1 assuming that we insert the points in alphabetical order into an initially empty data structure.

Table 1: Sample data for examples

Label	X coordinate	Y coordinate
A	0.7	0.5
B	0.1	0.4
C	0.3	0.2
D	0.8	0.7
E	0.5	0.9
F	0.9	0.8

Any data structure used to represent multidimensional point data must allow not only exact queries, to check if a specific point is part of a multidimensional dataset, but also associative queries, in which there are conditions for more than one coordinate but not all coordinates of the key are specified. Among the most important associative queries we have:

- Nearest neighbor queries: Given a point, which is the closest point in the dataset? An example could be: which is the closest pharmacy to our current location?
- Orthogonal range queries: Which members of the dataset are in a given rectangular area? For example, which Peruvian restaurants are in a rectangular area of Barcelona’s Eixample?
- Partial match queries: Given some specified coordinates, which points of the dataset match those coordinates? For instance, which YouTube political videos were created by Russian IP addresses?

Besides their intrinsic importance, partial match queries are of particular interest because analysis of nearest neighbor and orthogonal range queries can be done using the analysis of PM queries [CDZ01; DM02; DJM14]. In Figure 1 we present an example of a partial match query in 2 dimensions.

Many multidimensional data structures have been defined and analyzed since the 1960s; see Section 6.5 of Volume 3 of Knuth’s *The Art of Computer Programming* [Knu98] for a review. Some of those data structures partition the space independently of the data (tries) while others do it based on the data (trees).

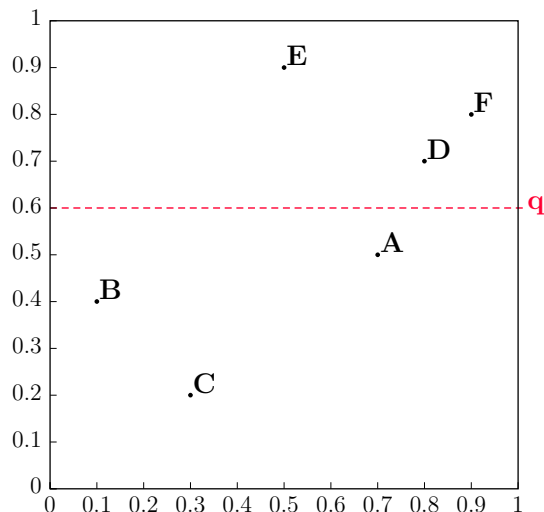


Figure 1: [Table 1](#) data and a partial match query $\mathbf{q} = (*, 0.6)$.

As mentioned before, this thesis is dedicated to the average-case analysis of PM queries in several types of multidimensional search trees.

A generalization of binary search trees for multidimensional keys, quad trees, were introduced in 1974 by Finkel and Bentley [FB74]. In quad trees all the nodes split the space in each of the coordinates of the key. In the case of two dimensions each node splits the space into four quadrants, hence the name quad trees. In computer graphics, 3-D quad trees are called octrees [Mea80].

In 1975 Bentley introduced K -d trees [Ben75], another generalization of binary search trees where each node splits the space by only one of the coordinates of the key: the root of the tree classifies by the first coordinate, its children by the second, and so on in a cyclical manner: nodes at level i classify by coordinate $i \bmod K$. [Figure 2](#) shows the standard 2-d tree produced from [Table 1](#) and the partition that it induces in the square $[0, 1]^2$.

For randomly generated K -d trees, Bentley claimed that partial match queries with s specified coordinates could be performed, measured in the number of nodes visited, in expected time $O(n^{1-s/k})$ but in 1986 Flajolet and Puech [FP86] disproved that claim by proving that the expected time is $O(n^{1-s/k+\theta(s/k)})$ where $\theta(u)$ is a strictly positive function of u in the interval $[0, 1]$. This paper was extremely influential as it was the first precise analysis of partial match queries. Additionally, it was one of the first to use complex analysis techniques that were later generalized in a seminal paper by Flajolet and Odlyzko [FO90].

In 1993 Flajolet and others [Fla+93] analyzed partial match queries in quad trees. They found that the partial match expected time has the same order as in K -d trees, $O(n^{1-s/k+\theta(s/k)})$, but with different multiplicative constants.

Those analyses of partial match queries were done with the implicit assumption that in each recursive call a new partial match query was generated

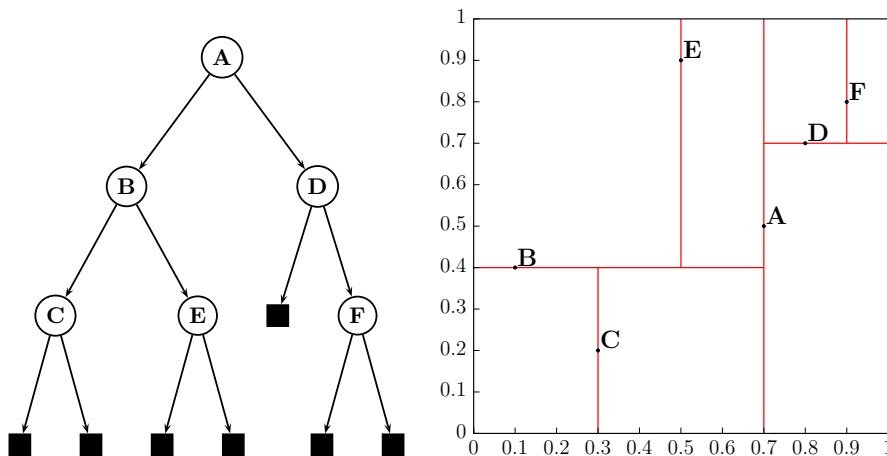


Figure 2: A standard 2-d tree produced from [Table 1](#) and the partition that it induces in $[0, 1]^2$.

randomly. Early in this decade several teams started the analysis of PM queries where the parameters of the query remain fixed throughout all the recursive calls. To distinguish them, the first type of PM queries are called random and the latter fixed.

In 1998, Duch, Estivill-Castro and Martínez [[DEM98](#)] introduced a variant of K -d trees, relaxed K -d trees, where each time a node is created its discriminant coordinate is chosen at random instead of being predefined in a cyclical way. To distinguish them from relaxed K -d trees, the original K -d trees defined by Bentley [[Ben75](#)] are called standard K -d trees.

Structure of the thesis

This thesis is organized as follows. In [Part II](#), in [Chapter 1](#) we give formal definitions of the associative queries, the data structures and the search algorithms. We describe the probabilistic model in [Chapter 2](#), including a more detailed distinction than the one in the articles of the different random variables used in the analysis of partial match queries. In [Chapter 3](#), we explain the general mathematical tools used for our theoretical analyses; there, we also explain the methodology used for the experimental studies included in many of our articles. We finish [Part II](#) with [Chapter 4](#), where we give the state of the art.

We present the articles that have been published as part of this research in [Part III](#). [Chapter 5](#) includes the first two articles, where we analyze fixed PM queries in K -d trees. In the first article [[DLM14](#)], we generalize the formula for relaxed K -d trees with $s = 1$, which was obtained in 2012 by Duch, Jiménez and Martínez [[DJM14](#)], to any s specified coordinates.

In the second article [[DLM16a](#)], the journal version of the first, we addition-

ally find a formula for fixed PM queries in standard K -d trees. We prove that in both cases the expected cost $P_{n,\mathbf{q}}$ of a fixed partial match query $\mathbf{q} \in [0, 1]^n$ in a tree that contains n data points satisfies:

$$P_{n,\mathbf{q}} = \Theta \left(\left(\prod_{i:q_i \text{ is specified}} q_i(1 - q_i) \right)^{\alpha/2} \cdot n^\alpha \right) \quad (1)$$

where α is the exponent, depending only on s and K , of the expected cost of random PM queries (each family of multidimensional trees has a characteristic α). We conjectured there that a similar relationship between the cost of random PM queries and fixed PM queries holds for quadtrees and other variants of K -d trees and we conjectured that for another variant, the squarish K -d trees, $P_{n,\mathbf{q}}$ satisfies:

$$P_{n,\mathbf{q}} = \Theta(n^{1-s/K}) \quad (2)$$

with the hidden constant factor of the main term independent of \mathbf{q} .

In [Chapter 6](#), in the article [\[DL17\]](#), we introduce a new K -d tree variant: relaxed K -dt trees, where every subtree of size $\leq 2t + 1$ is locally rebalanced and the discriminant coordinate at each node is randomly chosen. Standard K -dt trees were defined by Cunto, Lau and Flajolet in the late 1980s [\[CLF89\]](#). We analyze the average cost of random PM queries in this new variant and prove that formula (1) is not valid for fixed PM queries in K -dt trees, even though we could not find a closed-form formula for their expected cost.

In [Chapter 7](#), in the article [\[DLM18\]](#), we analyze fixed PM queries in quad trees and, under the assumptions of the existence of the limit

$$\lim_{n \rightarrow \infty} \frac{P_{n,\mathbf{q}}}{n^\alpha}$$

and uniformity of the coordinates of the data points, we prove that formula (1) holds, and thus that K -dimensional quad trees satisfy our conjecture in [\[DLM16a\]](#). Under those assumptions, we generalize Curien and Joseph's results [\[CJ11\]](#) as they had analyzed fixed PM queries in quad trees with $K = 2$.

In [Chapter 8](#), in the last article [\[DLM16b\]](#), we analyze random PM queries in quad- K -d trees, which are a generalization of both relaxed K -d trees and quad trees that were defined by Bereczky, Duch, Németh and Roura in 2014 [\[Ber+16\]](#).

We finalize the thesis by presenting the conclusions and open problems of this research in [Part IV](#).

Part II

Preliminaries

Chapter 1

Associative Queries and Search Trees

In this chapter we describe the historical motivation for associative queries and give formal descriptions of both partial match queries and the multidimensional search trees that we study in this thesis.

1.1 Associative Queries

In 1973, in the first edition of Volume 3 of *The Art of Computer Programming* [Knu73], in the *Searching* chapter, Knuth studied searching for primary keys and also retrieval on secondary keys. The first problem consists of, given a set of records identified by keys—that belong to a set with a total order—and a value for that key, finding which record has its key equal to that value. The second problem requires searching records given values of fields other than the primary key. The first problem is also known as searching for a single or *unidimensional key*, while the second is also called searching for a composite or *multidimensional key*. We use the latter terminology as well as the term *coordinates* instead of fields. More formally, we define a file F as a finite subset of a domain $\mathcal{D} = \mathcal{D}_0 \times \cdots \times \mathcal{D}_{K-1}$, $K \geq 1$, where each \mathcal{D}_i is a totally ordered set. We refer to the elements of \mathcal{D} as keys. We assume, without loss of generality, that $\mathcal{D}_i = [0, 1]$ for all i , that is $\mathcal{D} = [0, 1]^K$.

For retrieval on secondary keys, Knuth described three types of queries:

- *Simple queries*, one value is specified for one coordinate.
- *Range queries*, a range of values is specified for one coordinate.
- *Boolean queries*, the combination of the previous types using the operations **AND**, **OR** and **NOT**.

These type of queries are part of what are known as associative queries. We will focus on the partial match ones but before, for completeness, we describe in detail nearest neighbor and orthogonal range queries.

1.1.1 Nearest neighbor query

This type of query is best illustrated with a real life example. If, using your smartphone, you want to find the nearest taxi to your current location then your application is doing a nearest neighbor query.

Definition 1. *Given a file F of K -dimensional keys and a query point \mathbf{q} , a nearest neighbor query consists of finding the key in the file closest to \mathbf{q} according to some predefined distance measure d . This is,*

$$\{x \in F \mid d(\mathbf{q}, x) \leq d(\mathbf{q}, y), \forall y \in F\}.$$

Figure 3 has an example of a nearest neighbor query.

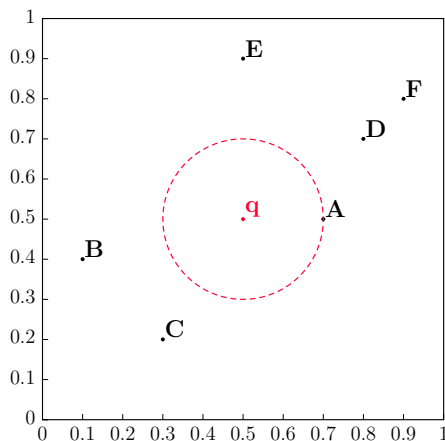


Figure 3: Given $\mathbf{q} = (0.5, 0.5)$ a nearest neighbor query under Euclidean distance returns **A**.

1.1.2 Orthogonal Range Query

In an orthogonal range query a contiguous range of values is specified for each coordinate. In two dimensions this specifies a rectangle and in three dimensions a rectangular cuboid.

Definition 2. *Given a file F of K -dimensional keys and a hyperrectangle*

$$\mathbf{q} = [a_0, b_0] \times [a_1, b_1] \times \cdots \times [a_{K-1}, b_{K-1}],$$

an orthogonal range query returns the subset of keys in F which belong to \mathbf{q} . That is, the set

$$\{\mathbf{x} = (x_0, \dots, x_{K-1}) \in F \mid a_i \leq x_i \leq b_i, \forall i \in \{0, \dots, K-1\}\}.$$

In general, combinations of Knuth’s range queries using the operation **AND** are orthogonal range queries. Figure 4 shows an orthogonal range query.

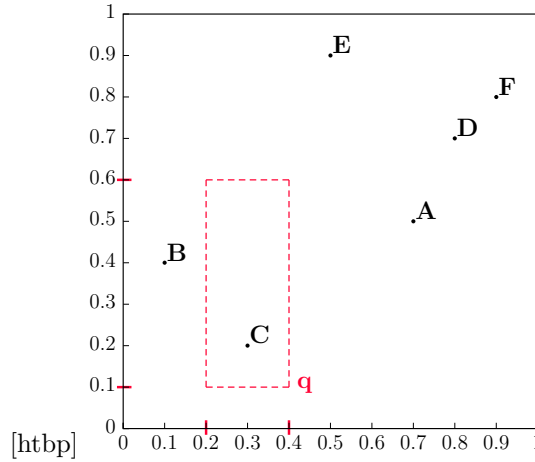


Figure 4: An orthogonal range query $q = [0.2, 0.4] \times [0.1, 0.6]$.

1.1.3 Partial Match Query

In a partial match query, only some of the coordinates of the requested points are specified. For instance, in two dimensions this would mean specifying a line parallel to the x or y axis, as the example in Figure 5 shows. In three dimensions it would be a line parallel to one of the axes or a plane parallel to one of the planes that contains two of the axes.

Definition 3. Given a query $\mathbf{q} = (q_0, q_1, \dots, q_{K-1})$ where each q_i is either an element of $[0, 1]$ (it is specified) or $*$ (it is unspecified), a partial match (PM) query returns the subset of keys x in F whose coordinates coincide with the specified coordinates of \mathbf{q} . Defining the PM hyperplane associated to the PM query \mathbf{q} as the set

$$\mathcal{H}(\mathbf{q}) = \{\mathbf{x} = (x_0, \dots, x_{K-1}) \in \mathcal{D} \mid x_i = q_i, \forall q_i \in [0, 1]\}.$$

the query returns $\mathcal{H}(\mathbf{q}) \cap F$.

For simplicity, in figures we use the label \mathbf{q} for the hyperplane $\mathcal{H}(\mathbf{q})$.

Knuth’s simple queries are partial match queries with just one coordinate specified. Combinations of Knuth’s simple queries using the operation **AND** are partial match queries.

Sometimes it will be necessary to distinguish between *extreme specified coordinates*, the ones that have $q_i = 0$ or $q_i = 1$, and the rest of them, the *regular specified coordinates*. We assume that the number s of specified coordinates satisfies $0 < s < K$ and the number s_0 of extreme specified coordinates satisfies

$0 \leq s_0 \leq s$. The ratios $\rho := s/K$ and $\rho_0 := s_0/K$ will be of particular interest in several cases. The *query pattern* is a string $\mathbf{u}(\mathbf{q}) = u_0 u_1 \cdots u_{K-1}$, such that $u_i = S$ if $q_i \neq *$ and $u_i = *$ if $q_i = *$. To distinguish between extreme and regular specified coordinates we will use $u_i = E$ for the extreme coordinates.

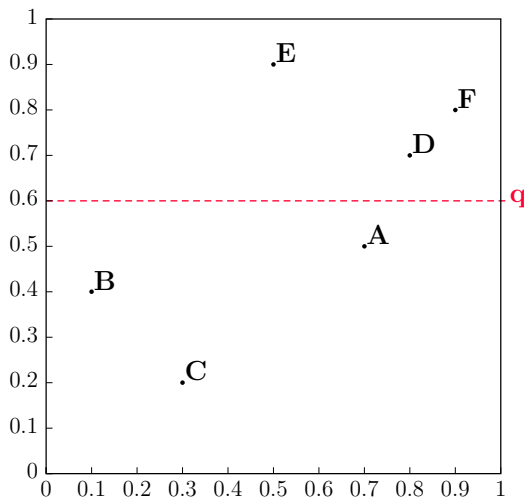


Figure 5: The partial match query $\mathbf{q} = (*, 0.6)$ with query pattern $\mathbf{u} = *S$.

1.2 Trees and Point Search Trees

Going back to *The Art of Computer Programming* [Knu73], among the data structures used for the problem of searching for primary keys, Knuth described and analyzed binary search trees, B-trees, tries and hashing while for the retrieval on secondary keys he described inverted files and combinatorial hashing among others. For the latter problem in the second edition [Knu98] he included grid files, quad trees and K -d trees. We show that the search trees used in both problems can be seen in a unified way. For that purpose, we start by defining trees and point search trees in a generic way that includes unidimensional structures, like binary search trees, B-trees and tries, as well as multidimensional search trees, like quad trees and K -d trees, and we explain concepts that apply to both cases.

For all the data structures that we mention in this work we refer the reader to the handbooks [Sam06; MS18] and references therein for more information about them.

Definition 4. A rooted ordered tree, or simply a tree, is either

- a leaf, or
- a node, called the root, attached to a sequence of trees.

We use the term trees for rooted ordered trees because we are not going to work with more general kinds of trees; see [FS13] for a detailed description of the tree nomenclature found in the literature. The trees in the sequence attached to a node are called the first, second, and m -th, subtrees of that tree, or, abusing language, of that node. The subtrees of its subtrees are also subtrees of a tree. We will always show the root of a tree drawing it at the top of each figure. The *children* of a node are the nodes/leaves directly below it, its *descendants* are the ones farther down. The *parent* of a node/leaf is the node directly above it, its *ascendants* or *ancestors* are the nodes farther up in the path from the root to the node/leaf. The root has no parent. In the literature leaves and nodes are sometimes called external and internal nodes respectively. The number of children at each node can vary and is called the *branching factor* or *degree*. The *size* of a tree is its number of nodes. An important particular case of trees are binary trees:

Definition 5. A binary tree is a tree where each node has degree 2.

The first subtree of a node in a binary tree is called the left subtree and the second one is called the right subtree. In the figures we represent leaves by \blacksquare and nodes by \circ . Figure 6 shows an example of a binary tree.

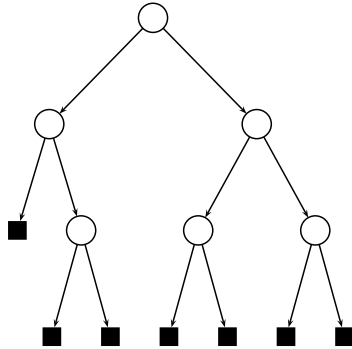


Figure 6: A binary tree. The root is at the top of the figure.

Intuitively, a point search tree T is a tree with one or more keys associated to each node in such a way that the keys in the root determine a partition of the set of keys, with each part associated to a child of the root. More precisely:

Definition 6. A point search tree T that stores a file $F \subset [0, 1]^K$ of n keys, and has a bounding box $BB(T) = [0, 1]^K$, is a tree with the following properties:

- When $n = 0$, T is empty (T consists of a leaf).
- When $n > 0$, T has as children point search trees T_1, \dots, T_m that store respectively $F_1, \dots, F_m \subset F$ and have respectively bounding boxes $BB(T_1), \dots, BB(T_m)$ such that:

- the root of T stores a subset $\mathcal{R} \subset F$ of r keys,
- for each $\mathbf{x} \in \mathcal{R}$, the root of T has associated total order relations, the discriminant rules, $\{\prec_{\mathbf{x},k}\}_{1 \leq k \leq m}$ such that for $\mathbf{y} \in F \setminus \mathcal{R}$:

$$\mathbf{y} \in F_k \iff \mathbf{y} \prec_{\mathbf{x},k} \mathbf{x}.$$

- $BB(T_1), \dots, BB(T_m)$ is a partition of $BB(T)$. We say that \mathcal{R} and the set of discriminant rules induce such partition.
- F_1, \dots, F_m is a partition of $F \setminus \mathcal{R}$ such that for all k , $1 \leq k \leq m$, $F_k \subset BB(T_k)$.

Intuitively, the set $\{\prec_{\mathbf{x},k}\}$ allows us, for any $\mathbf{y} \in [0, 1]^K$, to determine which T_k could store \mathbf{y} by performing comparisons between some (eventually all) of the coordinates of \mathbf{y} and the coordinates of the elements of \mathcal{R} . For example, in binary search trees, see [Definition 7](#) below, the relations $\prec_{\mathbf{x},k}$ are $\prec_{\mathbf{x},1} \equiv <$ and $\prec_{\mathbf{x},2} \equiv \geq$. Note that the set $\{\prec_{\mathbf{x},k}\}$ could vary with each node depending on the type of point search tree.

Even though there are families of search trees not covered by [Definition 6](#), like segment and metric trees, frequently we will, abusing language, refer to the point search trees simply as *search trees*. [Definition 6](#) covers a wide range of data structures, including unidimensional search trees ($K = 1$) as well as the multidimensional search trees ($K > 1$) studied in this thesis.

The number of keys stored at each node can vary. We will assume that T has n nodes, that is that it has size n . We will call the tree obtained by dropping the keys from the nodes of T the *underlying tree* of T . Given two point search trees T_1 and T_2 , we will write $T_1 \sim T_2$ when T_1 and T_2 have the same underlying tree. It is clear that \sim is an equivalence relation. All the point search trees with the same underlying tree form an equivalence class by that relation.

Given a point search tree T_v with root node v , sometimes, abusing the term, we will call the bounding box of T_v simply the bounding box of v . Each node splits its bounding box into a partition. The structure of the search tree provides a hierarchy of more and more refined partitions of the initial bounding box $BB(T)$, which simply coincides with the full domain \mathcal{D} ($\mathcal{D} = [0, 1]^K$ with our assumption), starting with $\{BB(T)\}$ and finishing with the partition formed by the bounding boxes of its leaves, which we will call the *partition induced* by the search tree.

Historically, the first point search trees that were studied were the binary search trees. They are binary trees with a single key associated to each node, that is their domain is unidimensional:

Definition 7. *A binary search tree is a point search tree with a totally ordered domain \mathcal{D} , where each node has degree 2 and stores one key with the property that all the keys in its left subtree are smaller than such key and all the keys in its right subtree are greater than or equal to such key.*

[Figure 7](#) shows a binary search tree with domain $\mathcal{D} = [0, 1]$ and the bounding boxes of its nodes and leaves. There the root splits $[0, 1]$ into $\{[0, 0.6], [0.6, 1]\}$,

node 2 splits $[0, 0.6)$ into $\{[0, 0.2), [0.2, 0.6)\}$ and so on. The tree in [Figure 6](#) is the underlying tree of the binary search tree in [Figure 7](#).

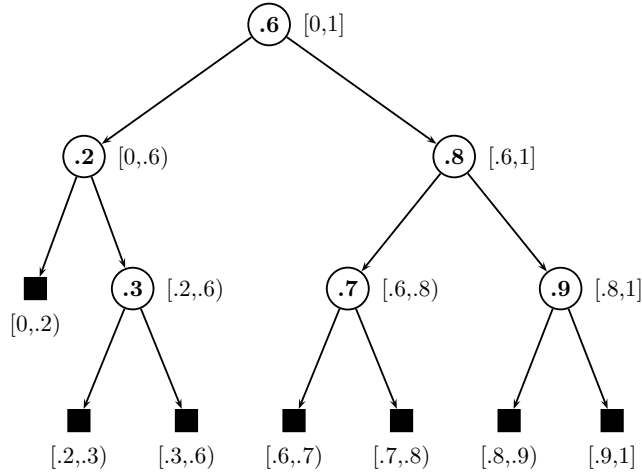


Figure 7: A binary search tree with domain $[0, 1]$ annotated with the bounding boxes of its nodes and leaves. Its underlying tree is the tree in [Figure 6](#).

[Table 2](#) shows a taxonomy of unidimensional search trees.

Partition type	Keys per node	Locally balanced?	Type of unidimensional point search tree
Space driven	0 or 1		Tries
Data driven	1	No	Binary search trees (BSTs)
		Yes	BSTs with fringe heuristic
	≥ 1		B-trees

The first criterion to classify them is how the partition of the data is performed in each node: independently of the data (space-driven) or driven by it. We will focus on the case of data-driven search trees, that is, those where the induced partition is governed by the data points stored in the search tree, as opposed to space-driven search trees, such as tries or digital search trees for $K = 1$. The second criterion to classify point search trees is the number of keys stored in each node. We will not explore any further search trees with many keys per node ($r > 1$), such as (unbalanced) m -ary search trees or B-trees for $K = 1$.

We will explain the binary search trees with fringe heuristic in [Section 1.3.4](#), when we introduce their multidimensional counterpart.

Typically a search tree is built adding one node at a time to an initially empty search tree. If the branching factor m is constant, then the number of leaves is $(m - 1)n + 1$ because, starting with one leaf, every time we add one node we replace one leaf by m leaves.

1.3 Multidimensional Search Trees

1.3.1 A Taxonomy

There are several multidimensional point search trees, depending on how the partition of the space is decided, by how many and by which dimensions the space is split and if there are local rebalances to obtain a more balanced structure. [Table 3](#) shows a taxonomy of multidimensional point search trees. The first two criteria to classify them are the same as in the unidimensional case: the partition type and the number of keys per node. We will study the case of data-driven search trees and we will not enter into more details about multidimensional space-driven search trees, such as K -d tries [[Ore82](#); [FP86](#)], quad tries and quad- K -d tries. Samet, in [[Sam90](#); [Sam06](#)], calls the K -d tries and quad tries PR k -d trees and PR quadtrees respectively (P for point and R for Region). Quad- K -d tries are the space driven variant of quad- K -d trees and can be defined analogously to K -d tries and quad tries. Regarding the number of keys per node, in all the trees that we study the nodes hold a single key, therefore the size of such tree is equal to the number of keys stored in it.

The next criterion to classify multidimensional point search trees, the number of discriminating dimensions for each node, has to do with the relations $\{\prec_{\mathbf{x},k}\}_{1 \leq k \leq m}$. In the families of trees that we are going to study these relations will be derived from choosing for each node d discriminant dimensions, i_1, i_2, \dots, i_d , then each of the relations $\{\prec_{\mathbf{x},k}\}_{1 \leq k \leq m}$ corresponds to one of the possible combinations of comparisons $y_{i_l} < x_{i_l}$ or $y_{i_l} \geq x_{i_l}$, $l = 1, \dots, d$. Therefore in these families of search trees the branching factor is always a power of 2: $m = 2^d$.

We defer the explanation of the next criterion, if there are local rebalances or not, to [Section 1.3.4](#). The last criterion, rule for discrimination, refers to how it is decided which are the discriminating dimensions in each node.

Table 3: A taxonomy of multidimensional point search trees

Partition type	Keys per node	# of discriminating dimensions (per node)	Locally balanced?	Rule for discrimination		
				All (Quad)/ Cyclically (K -d)	Random	Longest edge
Space driven	0 or 1	K		Quad tries		
		1		K -d tries		
		$d, 1 \leq d \leq K$		Quad- K -d tries		
Data driven	1	K		Quad trees		
		1	No	Standard K -d trees	Relaxed K -d trees	Squarish K -d trees
			Yes	Standard K -dt trees	Relaxed K -dt trees	
		$d, 1 \leq d \leq K$		Quad- K -d trees		

1.3.2 Quad trees

In 1974 Finkel and Bentley [FB74] introduced a simple generalization of binary search trees for composite keys, where each node stores a key and uses the values of its coordinates to discriminate by all of them, that is the number of discriminating dimensions d is equal to K . They described in detail the two-dimensional case, in which each node partitions its bounding box into four quadrants, analogous to NE, NW, SW and SE on a map, hence the name quad trees. They explained that this data structure could easily be generalized to any number of dimensions.

Definition 8 (Finkel and Bentley [FB74]). *A quad tree T that stores a file $F \subset [0, 1]^K$ of n keys is either*

- *empty when $n = 0$, in which case we call it a leaf, or*
- *its root stores a record with key $\mathbf{x} = (x_0, \dots, x_{K-1})$ and pointers to its 2^K children that store the $n-1$ remaining records as follows: each child, let us call it T_w , is associated with a string $w = w_0w_1 \dots w_{K-1} \in \{0, 1\}^K$, such that $\forall w \in \{0, 1\}^K$, T_w is a quad tree and, for any key $\mathbf{y} = (y_0, \dots, y_{K-1}) \in T_w$ and $0 \leq i < K$, it holds that*

- *if $w_i = 0$ then $y_i < x_i$*
- *if $w_i = 1$ then $y_i \geq x_i$.*

Given that in quad trees all the nodes have 2^K children, any quad tree of size n induces a partition of the domain $[0, 1]^K$ into $(2^K - 1)n + 1$ regions, each corresponding to a leaf in the quad tree. Figure 8 shows an example of a quad tree and the partition that it induces in $[0, 1]^2$.

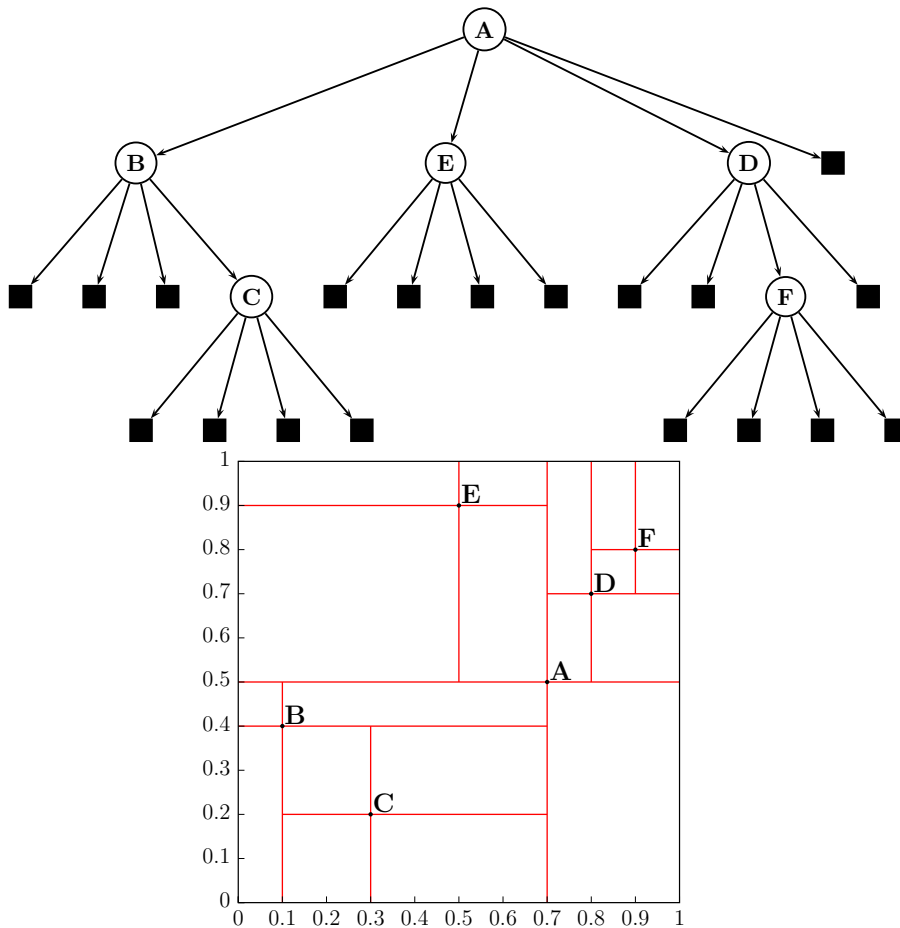


Figure 8: A quad tree produced from [Table 1](#) and the partition that it induces in $[0, 1]^2$.

1.3.3 K -d trees

In 1975 Bentley [Ben75] introduced K -d trees, another generalization of binary search trees, where each node stores a multidimensional key and uses the values of only one of its coordinates to discriminate or partition its bounding box. That is, in this case the number of discriminating dimensions d is equal to 1.

Definition 9 (Bentley [Ben75]). *A K -dimensional tree (K -d tree) T that stores a file $F \subset [0, 1]^K$ of n keys is either*

- *empty when $n = 0$, in which case we call it a leaf, or*
- *its root stores a record with key $\mathbf{x} = (x_0, \dots, x_{K-1})$, a discriminant i , $0 \leq i < K$, and pointers to two subtrees, the left subtree T_L and the right one T_R , that store the $n - 1$ remaining records with the following constraints:*
 - *if $\mathbf{y} = (y_0, \dots, y_{K-1}) \in T_L$ then $y_i < x_i$*
 - *if $\mathbf{y} = (y_0, \dots, y_{K-1}) \in T_R$ then $y_i \geq x_i$.*

Since all its nodes have two children, any K -d tree of size n induces a partition of its domain into $n + 1$ regions, each corresponding to one of its leaves.

We denote by $\langle \mathbf{x}, i \rangle$ a node that contains a key \mathbf{x} with discriminant i . In a K -d tree, if a node $\langle \mathbf{x}, i \rangle$ has bounding box $[x_{0,0}, x_{0,1}] \times \dots \times [x_{i,0}, x_{i,1}] \times \dots \times [x_{K,0}, x_{K,1}]$ then the bounding box of its left subtree is $[x_{0,0}, x_{0,1}] \times \dots \times [x_{i,0}, x_i] \times \dots \times [x_{K,0}, x_{K,1}]$ and the bounding box of its right subtree is $[x_{0,0}, x_{0,1}] \times \dots \times [x_i, x_{i,1}] \times \dots \times [x_{K,0}, x_{K,1}]$.

In the original variant of K -d trees, which we call standard, the discriminant coordinate rotates cyclically.

Definition 10 (Bentley [Ben75]). *A standard K -d tree is a K -d tree where nodes at level i of it discriminate by coordinate $i \bmod K$. The root of the K -d tree is considered to be at level 0, therefore it discriminates by coordinate 0.*

Figure 9 shows a standard K -d tree and the partition induced by it.

In 1998 Duch, Estivill-Castro and Martínez [DEM98] proposed another variant of K -d trees: relaxed K -d trees. For these, when each node is created, the discriminant coordinate is randomly chosen. Analysis of the PM expected cost in relaxed K -d trees is simpler as it leads to a single equation, while for standard K -d trees a system of equations is needed.

Definition 11 (Duch, Estivill-Castro and Martínez [DEM98]). *A relaxed K -d tree is a K -d tree where each node is assigned, and stores, a random discriminant uniformly and independently drawn from $\{0, \dots, K - 1\}$.*

Figure 10 shows a relaxed K -d tree and the partition induced by it.

In order to improve the performance of searches, in 2000 Devroye, Jabour and Zamora-Cura [DJZ00] introduced another variant, squarish K -d trees, where the discriminant is chosen in a way that provides a more balanced partition of the space.

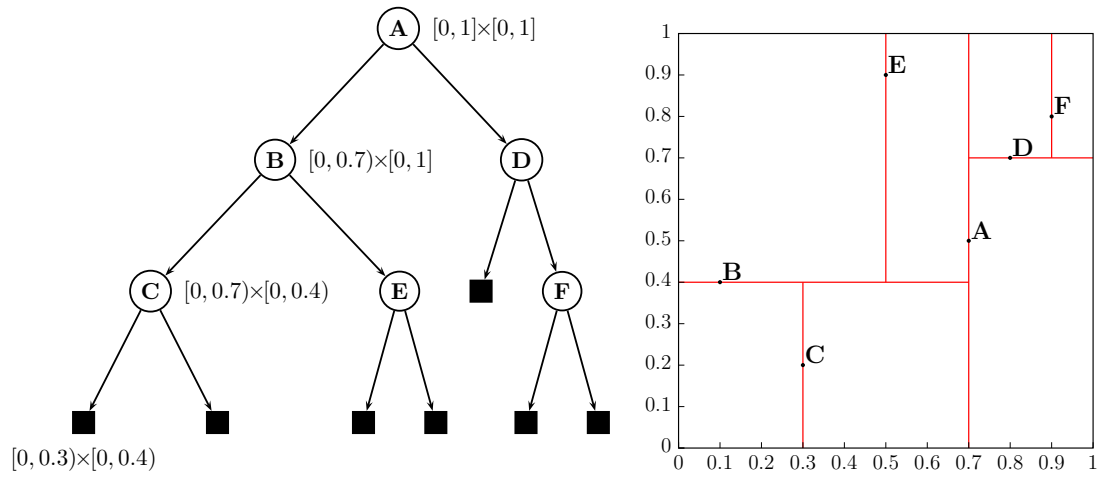


Figure 9: A standard 2-d tree produced from Table 1, annotated with some bounding boxes, and the partition that it induces in $[0, 1]^2$.

Definition 12 (Devroye, Jabbour and Zamora-Cura [DJZ00]). A squarish K -d tree is a K -d tree where each node is assigned a discriminant along the coordinate for which the bounding box of the node is most elongated.

Figure 11 shows a squarish K -d tree and the partition induced by it.

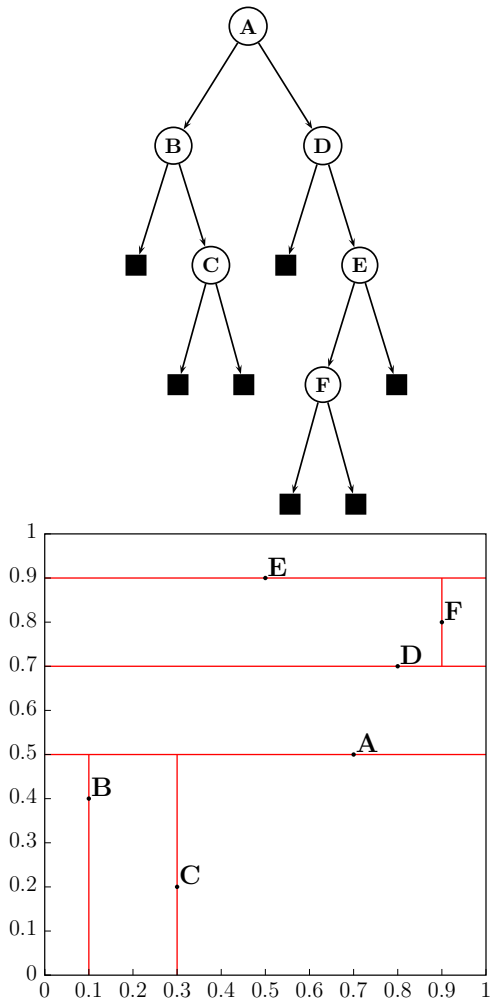


Figure 10: A relaxed 2-d tree produced from [Table 1](#) and the partition that it induces in $[0, 1]^2$.

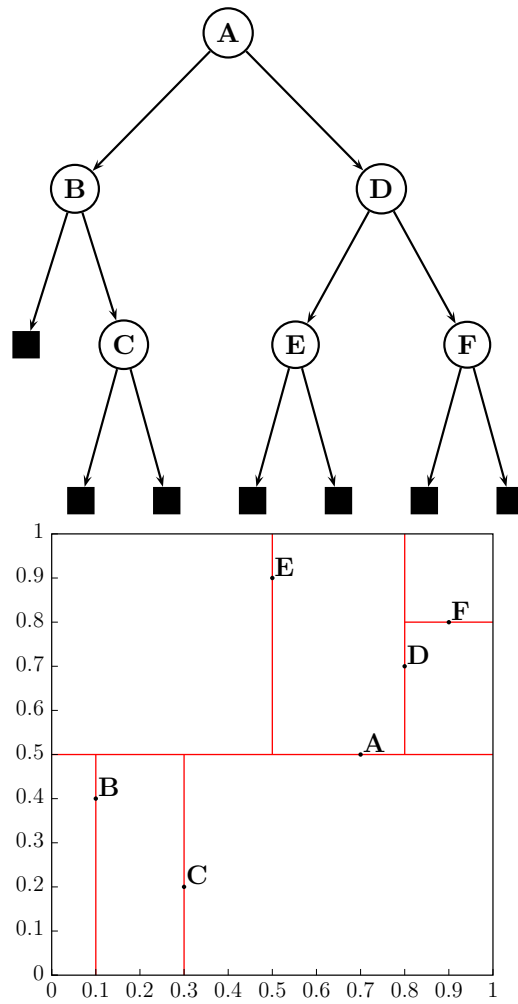


Figure 11: A squarish 2-d tree produced from [Table 1](#) and the partition that it induces in $[0, 1]^2$.

1.3.4 K -dt trees

To introduce the concept of locally balanced search trees, also known as search trees with fringe heuristic or fringe-balanced search trees, it is worth explaining the analogy between binary search trees and the quicksort algorithm [Hoa62]. Given a set of keys in a totally ordered domain, to sort them, quicksort starts by choosing one of them as a pivot. Using the divide-and-conquer principle quicksort splits the keys into two subsets: the ones that are smaller and the ones that are larger than the pivot. These two subsets are recursively sorted. The partition of the set of keys is analogous to a binary search tree with the pivot stored in the root.

As a way to improve quicksort, particularly the worst case, when the keys are already sorted, Singleton [Sin69] proposed to choose the pivot as the median of a sample of size three. The analogue in binary search trees is that every time we insert a node that would be the only child of an only child, we rotate the last three nodes in that branch to have the median of the three keys in the root of that subtree, see Figure 12. No other rebalance is performed in the search tree, so they are always done at the fringe of the search tree, hence the name fringe heuristic.

Definition 13. Given an integer $t > 0$, the t -fringe of a binary tree is the set of subtrees of size $\leq 2t$ such that if they are subtrees of another tree, the size of such subtree is $> 2t$.

The 0-fringe of a binary tree is the set of its leaves.

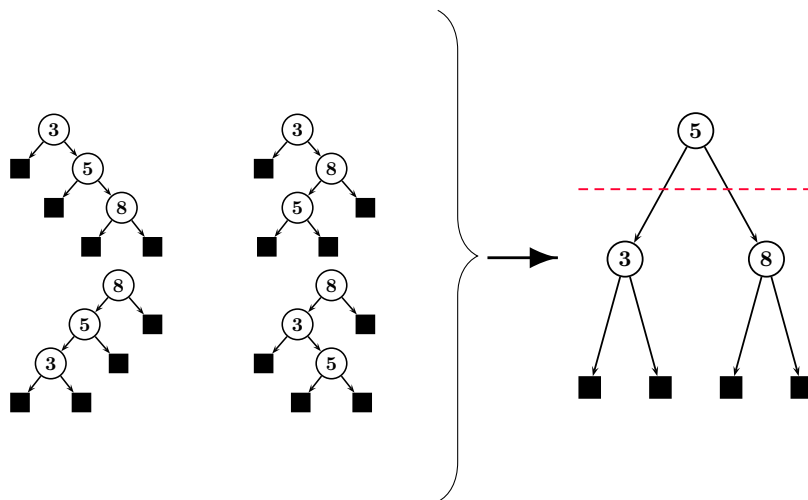


Figure 12: Local rebalance examples, with $t = 1$, in a binary search tree. After the rebalance the root is outside the 1-fringe, that is above the red dashed line, and, assuming only insertions, it will not be subject to any more rebalances.

The median-of-three strategy to select the pivot in quicksort can be gener-

alized to the median of $2t + 1$ for any integer $t > 0$. Analogously, Bell [Bel65] and Walker and Wood [WW76] defined the *locally balanced binary search trees* or *fringe-balanced binary search trees*, where the insert, update and delete operations are such that any subtree of size greater than $2t$ has at least t elements in each of its left and right subtrees. Devroye [Dev93] analyzed the expected height of fringe-balanced binary search trees and Poblete and Munro [PM85] computed the expected number of comparisons, and its variance, in fringe-balanced binary search trees. Cunto, Lau and Flajolet [Lau88; CLF89] adapted the local rebalance approach to K -d trees:

Definition 14 (Cunto, Lau and Flajolet [CLF89]). *A K -dt tree is a K -d tree in which each subtree of size greater than $2t$ has at least t elements in each of its left and right subtrees. Equivalently, a K -dt tree is a K -d tree in which any node just above the t -fringe has at least t elements in each of its left and right subtrees.*

Note that when $t = 0$ a K -dt tree is a K -d tree. This definition is independent of the way in which discriminants are assigned to nodes. Figure 13 shows an example of a standard K -dt tree with $K = 2$ and $t = 1$, the local rebalances performed when inserting the keys in Table 1 alphabetically, and the partition that it induces in $[0, 1]^2$. Note that this partition is very different from the one induced by standard K -d trees (Figure 9). When there is a rebalance, Figure 13 shows the search tree in an intermediate step: after the insertion and before the rebalance is performed. In the search trees in such intermediate steps a node just above the t -fringe has less than t elements in one of its subtrees and therefore it needs a local rebalance. In future inserts, only the nodes in the fringe are subject to rebalances.

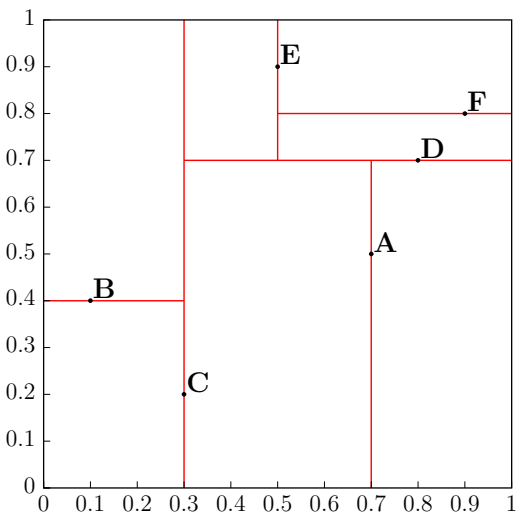
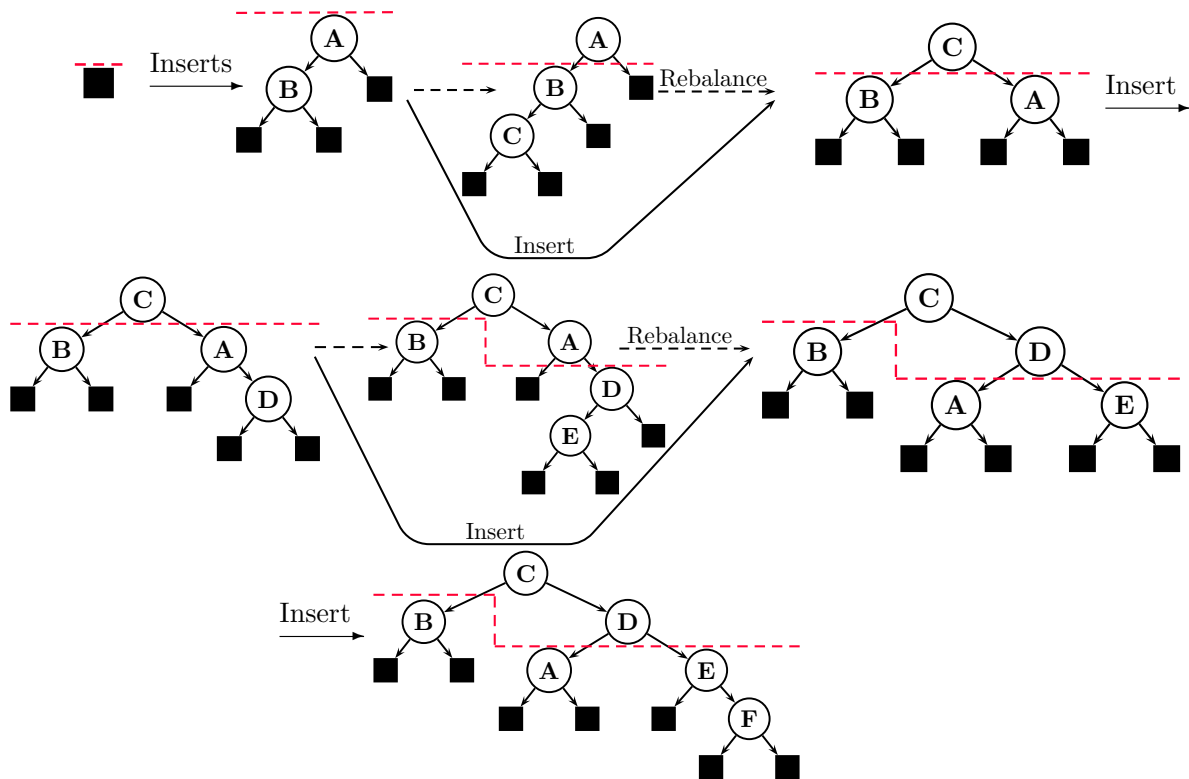


Figure 13: A standard K -dt tree ($K = 2$, $t = 1$), the local rebalances performed when inserting the keys alphabetically, and the partition that it induces in $[0, 1]^2$. The subtrees just below the red lines are the fringe. In the intermediate search trees, just before a rebalance, a node just above the t -fringe has less than t elements in one of its subtrees. In future inserts, only the nodes in the fringe are subject to rebalances.

1.3.5 Quad- K - d trees

In 2014 Bereczky, Duch, Németh and Roura [Ber+14; Ber+16] introduced quad- K - d trees a generalization of both K - d trees and quad trees. In quad- K - d trees each node discriminates by a variable number of coordinates.

Definition 15 (Bereczky, Duch, Németh and Roura [Ber+16]). *A quad- K - d search tree (quad- K - d tree) T that stores a file $F \subset [0, 1]^K$ of n keys is either*

- *empty when $n = 0$, in which case we call it a leaf, or*
- *its root stores a record with key $\mathbf{x} = (x_0, \dots, x_{K-1})$, a coordinate split bit-vector $\delta = (\delta_0, \dots, \delta_{K-1})$ that contains exactly d ones (i.e., it is of order d), with $1 \leq d \leq K$, and pointers to 2^d subtrees that store the $n - 1$ remaining records as follows: each of those subtrees, let us call it T_w , is associated with a string $w = w_0 \dots w_{K-1} \in \{0, 1, \#\}^K$, such that $\forall w \in \{0, 1, \#\}^K$, T_w is a quad- K - d tree and, for any key $\mathbf{y} = (y_0, \dots, y_{K-1}) \in T_w$ and $0 \leq i < K$, it holds that*
 - *if $\delta_i = 0$ then $w_i = \#$*
 - *if $\delta_i = 1$ and $w_i = 0$, then $y_i < x_i$*
 - *if $\delta_i = 1$ and $w_i = 1$, then $y_i \geq x_i$.*

If for all the nodes of a quad- K - d tree T the associated split vector δ contains all the K coordinates ($\delta_i = 1$ for all i) then T is a quad tree, and if, again for all the nodes, it contains exactly one coordinate then T is a relaxed K - d tree. Figure 14 shows an example of a quad- K - d tree with domain $[0, 1]^2$ and the partition that it induces.

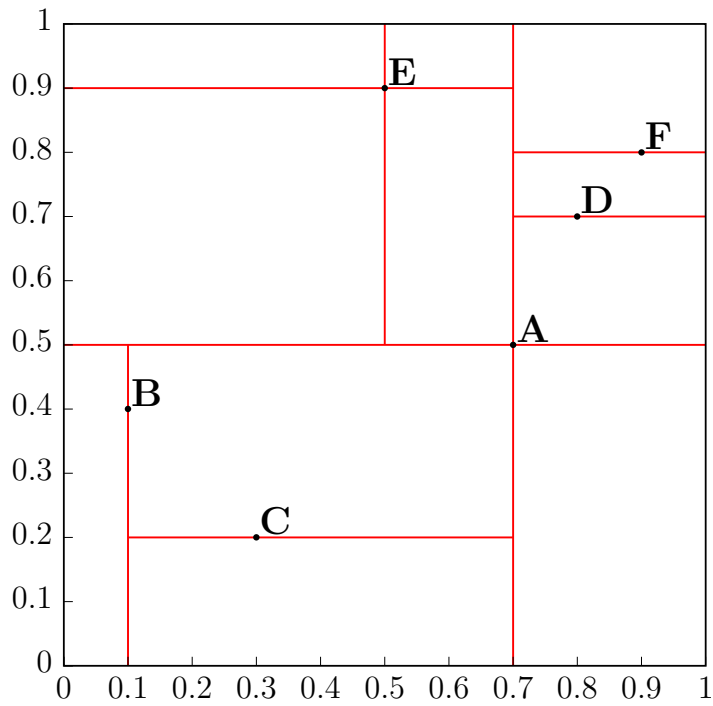
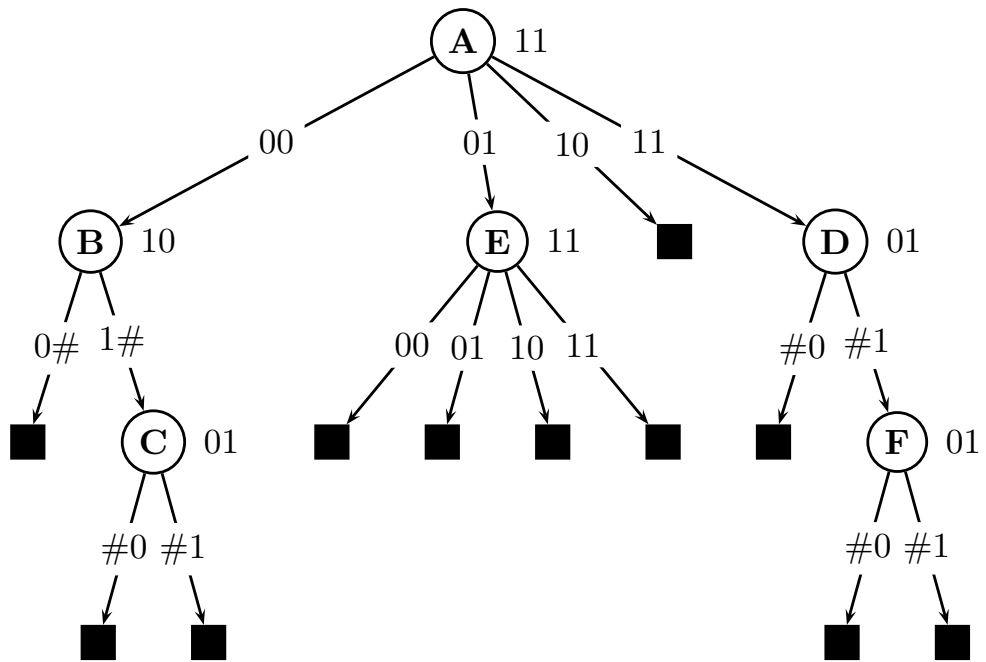


Figure 14: A quad- K - d tree produced from [Table 1](#), annotated with the split vector of each node and the associated string of each subtree, and the partition that it induces in $[0, 1]^2$.

1.4 Search Algorithms

There are several types of associative searches where we search for a subset of the domain, that is a region within it. In an exact search the specified region is a single point, in a partial match we look for the keys that belong to the query hyperplane, and in an orthogonal range search we look for the keys falling inside a given hyperrectangle. For these three types of queries we are going to define a generic region search algorithm. Note that there are other types of associative searches like nearest neighbor not included in this algorithm. Given a subset \mathcal{Q} of $[0, 1]^K$ and a search tree T , [Algorithm 1](#) is a generic search algorithm that retrieves all the records in the given region \mathcal{Q} .

Algorithm 1 Generic region search in a point search tree

```
procedure REGIONSEARCH( $\mathcal{Q}, T$ )  
  if  $T$  is a leaf then return  
   $\mathbf{x} \leftarrow$  key of the root of  $T$   
  if  $\mathbf{x} \in \mathcal{Q}$  then  
    Report  $\mathbf{x}$   
   $m \leftarrow$  number of children of  $T$   
  for  $k = 1$  to  $m$  do  
    if  $\mathcal{Q}$  intersects  $BB(T_k)$  then  
      REGIONSEARCH( $\mathcal{Q}, T_k$ )
```

In the case of partial match queries instead of the region \mathcal{Q} the parameter is a PM query \mathbf{q} because in that case the region is the PM hyperplane $\mathcal{H}(\mathbf{q})$. As an example, we show in [Algorithm 2](#) the partial match algorithm for the particular case of K -d trees. The algorithms for other variants of multidimensional point search trees are analogous.

For all the search trees the exact, or full match, search is simply a Region-Search where the region specified is a single point.

The insertion algorithm is an exact search. Upon conclusion of an unsuccessful search, the leaf in which the search ends is replaced by a new node with the new key that is being inserted. In the case of K -dt trees, if needed, a rebalance is then performed.

Algorithm 2 Partial match algorithm in a K -d tree

```
procedure KDTREESPARTIALMATCH( $\mathbf{q}$ ,  $T$ )  
  if  $T$  is a leaf then return  
   $\mathbf{x} \leftarrow$  key of the root of  $T$   
  if  $x_i = q_i, \forall q_i \in [0, 1]$  then  
    Report  $\mathbf{x}$   
   $i \leftarrow$  discriminant of the root of  $T$   
   $\triangleright T_L$  is the left subtree of  $T$   
   $\triangleright T_R$  is the right subtree of  $T$   
  if  $q_i = *$  then  
    KDTREESPARTIALMATCH( $\mathbf{q}$ ,  $T_L$ )  
    KDTREESPARTIALMATCH( $\mathbf{q}$ ,  $T_R$ )  
  else  
    if  $q_i < x_i$  then  
      KDTREESPARTIALMATCH( $\mathbf{q}$ ,  $T_L$ )  
    else  
      KDTREESPARTIALMATCH( $\mathbf{q}$ ,  $T_R$ )
```

Chapter 2

Probabilistic Model

In this chapter we start by explaining the probabilistic model that we use in the average-case analysis of partial match queries. We then describe the different random variables used in a more precise way than what is usual in the literature. We finalize by giving some relationships between the random variables that are useful in our analysis.

2.1 Random Search Tree Model

Throughout this thesis, we shall measure, as it is usual in the literature, the cost of the partial match search algorithm by the number of visited nodes in the search tree. Abusing language, we will refer to the cost of the partial match search algorithm simply as the cost of the PM query. Taking into account that a search tree is built adding one node at a time to an initially empty search tree, the order in which the data is inserted is a crucial factor in the cost of any search. For that reason, given a type of search tree, instead of considering all the possible search trees as equally likely (the uniform model), the standard model in the field is the random permutation or *random search tree model*. In this model all the coordinates of all the keys are independent and identically distributed (i.i.d.) random variables with a continuous distribution. Given our assumption that $\mathcal{D}_i = [0, 1]$ for all i , this continuous distribution is on the interval $[0, 1]$.

Given an input sequence of keys $\mathbf{x}_1, \dots, \mathbf{x}_n$ where $\mathbf{x}_j = (x_j^{(0)}, \dots, x_j^{(K-1)})$, the underlying tree of the point search tree built by inserting the elements of the sequence in that order only depends on how the coordinates are sorted. Let $r_j^{(i)}$ be the rank of $x_j^{(i)}$ in $\{x_1^{(i)}, \dots, x_n^{(i)}\}$, that is, $r_j^{(i)}$ is the number of coordinates $x_{j'}^{(i)}$, $1 \leq j' \leq n$ such that $x_{j'}^{(i)} < x_j^{(i)}$ and let $\mathbf{r}_j = (r_j^{(0)}, \dots, r_j^{(K-1)})$ be the *rank vector* associated to the data point \mathbf{x}_j . Then the underlying tree of the point search tree depends only on the rank vectors $\mathbf{r}_1, \dots, \mathbf{r}_n$, or equivalently, it depends only on the K permutations of order n $(r_1^{(i)}, \dots, r_n^{(i)})$, $i = 0, \dots, K-1$.

To assume that all coordinates of all data points are independently drawn from a continuous distribution ¹ on the interval $[0, 1]$ is then equivalent to assuming that the K permutations $(r_1^{(i)}, \dots, r_n^{(i)})$ are random permutations of order n , independently drawn from one another. To simplify, unless we state the opposite, we assume that the continuous distribution used to generate all the coordinates is $\text{Uniform}(0,1)$.

In other words, in the random search tree model the $n!^K$ possible input sequences of rank vectors are equiprobable; see Section 2.3 of [Mah92] for a formal proof of this fact for the case $K = 1$. Note that two different input sequences of rank vectors could generate the same point search tree. As an example, for the case of binary search trees ($K = 1$) of size $n = 3$, Table 4 shows all $3!$ permutations $(r_1^{(0)}, r_2^{(0)}, r_3^{(0)})$ of the set $\{1, 2, 3\}$, the binary search trees built with them as inputs, their underlying trees and their probabilities. The two permutations $(2, 1, 3)$ and $(2, 3, 1)$ generate the same binary search tree, therefore neither the binary search trees nor their underlying trees are equiprobable.

Table 5 illustrates the case of standard 2-d trees with three keys. There are two permutations to consider, one for each coordinate. The table shows some examples of the standard K -d trees generated by the permutations, all the underlying trees and their probabilities. In the underlying trees we emphasize with red lines the discriminating coordinates of each node: a vertical line for nodes that discriminate by the first coordinate and a horizontal one for the ones that discriminate by the second coordinate.

In the case of K -d trees the probabilities of the underlying trees are the same as in the case of binary search trees, see [Mah92]. In this case the random tree model can also be described as one in which the j -th inserted data point \mathbf{x}_j is equally likely to fall in any of the j leaves. Finally, it can also be characterized as one where all the potential sizes of the left (right) subtree are equally likely, that is, the probability that the left (right) subtree has size j is $1/n$ for all $j = 0, \dots, n - 1$ (if the root is a node then the subtrees are at most of size $n - 1$). These characterizations do not apply to quad trees and quad- K -d trees as their underlying trees are not binary ones.

In the cases where there is randomness in the discriminant, namely relaxed K -d trees and quad- K -d trees we should extend the concept of underlying tree to include the discriminant coordinate of each node because that is relevant in the partial match cost.

In the case of quad- K -d trees, the number of discriminating coordinates for each node, d , is chosen uniformly at random from $\{0, \dots, K - 1\}$ and any subset of d coordinates out of K is equally likely to be the set of discriminating coordinates, see [DLM16b] for more details about this case.

Finally, note that our assumption that the coordinates are independently drawn from a continuous distribution on the interval $[0, 1]$ implies that, for any two distinct keys x_j and $x_{j'}$, the probability that $x_j^{(i)} = x_{j'}^{(i)}$ is zero. Therefore

¹Actually, we can assume that the i -th coordinates of all data points are independently drawn from some continuous distribution F_i on the interval $[0, 1]$, $i = 0, \dots, K - 1$.

to simplify the analysis we assume, without loss of generality, that no two keys in the file have the same value in any of the coordinates.

Table 4: Permutations, BSTs generated and underlying tree probabilities

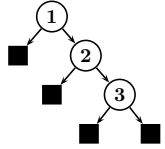
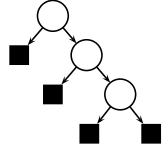
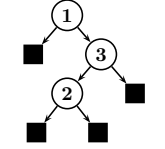
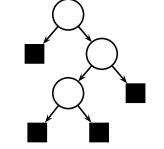
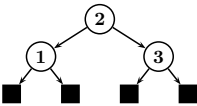
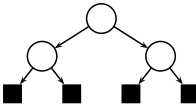
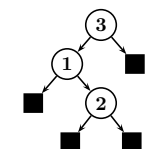
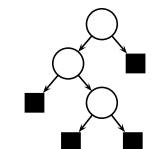
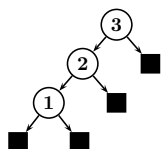
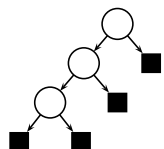
x coordinate permutation: $r_1^{(0)}, r_2^{(0)}, r_3^{(0)}$	Binary search tree (BST)	Underlying tree	Underlying tree probability
1, 2, 3			$\frac{1}{6}$
1, 3, 2			$\frac{1}{6}$
2, 1, 3 2, 3, 1			$\frac{1}{3}$
3, 1, 2			$\frac{1}{6}$
3, 2, 1			$\frac{1}{6}$

Table 5: Permutations, examples of 2-d trees generated, underlying trees and their probabilities. The sequences in bold are the ones that generate the examples. The red lines in the nodes of the underlying trees indicate by which coordinate each node discriminates

x coordinate permutation: $r_1^{(0)}, r_2^{(0)}, r_3^{(0)}$	y coordinate permutation: $r_1^{(1)}, r_2^{(1)}, r_3^{(1)}$	Standard 2-d tree example	Underlying tree	Underlying tree probability
1, 2, 3 1, 2, 3 1, 2, 3 1, 3, 2 1, 3, 2 1, 3, 2	1, 2, 3 2, 1, 3 3, 1, 2 1, 2, 3 2, 1, 3 3, 1, 2			$\frac{1}{6}$
1, 2, 3 1, 2, 3 1, 2, 3 1, 3, 2 1, 3, 2 1, 3, 2	1, 3, 2 2, 3, 1 3, 2, 1 1, 3, 2 2, 3, 1 3, 2, 1			$\frac{1}{6}$
2, 1, 3 2, 3, 1	Any Any			$\frac{1}{3}$
3, 1, 2 3, 1, 2 3, 1, 2 3, 2, 1 3, 2, 1 3, 2, 1	1, 2, 3 2, 1, 3 3, 1, 2 1, 2, 3 2, 1, 3 3, 1, 2			$\frac{1}{6}$
3, 1, 2 3, 1, 2 3, 1, 2 3, 2, 1 3, 2, 1 3, 2, 1	1, 3, 2 2, 3, 1 3, 2, 1 1, 3, 2 2, 3, 1 3, 2, 1			$\frac{1}{6}$

2.2 Queries and Ranks

Given a point search tree T and a partial match query \mathbf{q} , the *rank vector* of \mathbf{q} with respect to T is the vector $\mathbf{r}(\mathbf{q}, T) = (r_0, \dots, r_{K-1})$ defined as follows: if $q_i = *$ then $r_i = *$; if $q_i \neq *$ then r_i is the number of keys \mathbf{x} in the file stored in T such that $x_i \leq q_i$. Note that, given we are focused on search trees where each node holds a single key, each specified r_i ranges between 0 and n , the size of T . Most of the time it will be clear to which tree T we are referring and we will just write $\mathbf{r}(\mathbf{q})$ instead of $\mathbf{r}(\mathbf{q}, T)$.

Analogously to the query pattern, the *rank vector pattern* is defined as follows. Given a rank vector \mathbf{r} , its pattern is a string $\mathbf{u}(\mathbf{r}) = u_0 u_1 \cdots u_{K-1}$, such that $u_i = S$ if $r_i \neq *$ and $u_i = *$ if $r_i = *$.

If two partial match queries \mathbf{q} and \mathbf{q}' are such that $\mathbf{r}(\mathbf{q}) = \mathbf{r}(\mathbf{q}')$ then, for any multidimensional point search tree of any type, the comparisons to search for \mathbf{q} and \mathbf{q}' will take exactly the same path in the search tree and therefore the cost ² of \mathbf{q} will be the same as the cost of \mathbf{q}' , which is illustrated in Figure 15. This is similar to the fact that the underlying tree of a point search tree depends only on the rank vectors of the input.

In the next section, where we define the random variables that we study, we take advantage of the fact that the cost of a partial match query \mathbf{q} only depends on the rank vector $\mathbf{r}(\mathbf{q})$.

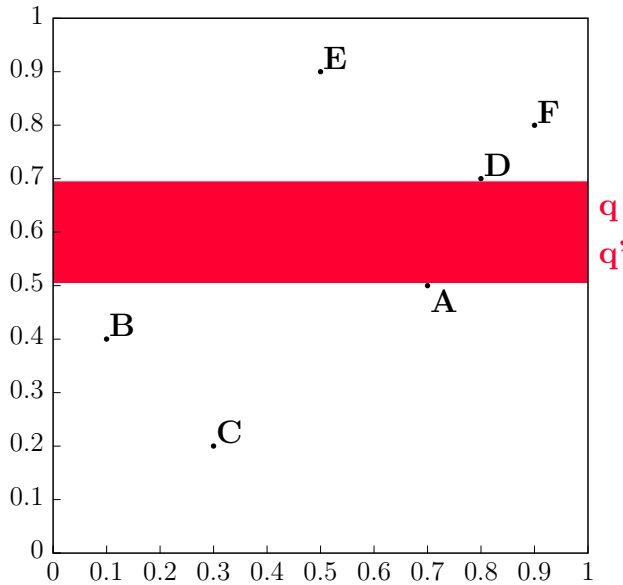


Figure 15: Any two partial match queries \mathbf{q} and \mathbf{q}' in the red region have the same rank vector and the same cost.

²Sometimes, to abbreviate, instead of the cost of the partial match search algorithm with input \mathbf{q} we will say the cost of the partial match query \mathbf{q} or simply the cost of \mathbf{q} .

2.3 Random Variables for Partial Match Queries

We will start by defining the different random variables used in the analysis of partial match queries and then give the relationships between them. We are going to systematically use calligraphic letters, e.g. \mathcal{P} , and variants with sub- and superscripts, to denote random variables, and the corresponding uppercase letters, e.g. P , to denote expectations, thus $P = \mathbb{E}[\mathcal{P}]$. We will use V , eventually with sub- and superscripts, to denote the variance of the random variable of interest.

2.3.1 Random Partial Match Queries

For several decades since the introduction of multidimensional search trees in the 1970s, the analysis of partial match queries in these data structures focused on random queries. Given a random search tree and a query pattern \mathbf{u} , $\mathbf{Q} = (Q_0, \dots, Q_{K-1})$ is a *random PM query* with pattern \mathbf{u} if for all i , $0 \leq i \leq K-1$:

$$Q_i = \begin{cases} * & \text{if } u_i = * \\ \text{random variable i.i.d. as the coordinates} & \text{if } u_i = S \\ \text{of the keys in the random search tree} & \end{cases}$$

$\hat{\mathcal{P}}_{n,\mathbf{Q}}$ is the random variable representing the cost of the partial match algorithm with a random PM query \mathbf{Q} and a random search tree of size n as input.

Sometimes it suffices to describe the random PM query \mathbf{Q} by some parameters, in which case we will denote by $\hat{\mathcal{P}}_{n,\mathbf{a}}$ the random variable of the cost of the partial match algorithm with a random query that has the characteristics described by \mathbf{a} , which could be one or more additional parameters. For example, \mathbf{a} could be the query pattern $\mathbf{u} = \mathbf{u}(\mathbf{Q})$, or the ratio ρ of the number of specified coordinates s to the dimension K . For instance, $\hat{\mathcal{P}}_{n,\mathbf{u}}$ is the random variable of the cost of the partial match algorithm with a random query \mathbf{Q} with pattern \mathbf{u} and a random search tree of size n as input.

2.3.2 Randomized Partial Match Algorithm

Analysis of the expected cost of random PM queries is usually done based on recurrences which imply that in each recursive call a new set of values is used for the query specified. We call such theoretical algorithm *randomized partial match*. In order to more clearly see the difference between this algorithm and the partial match algorithms presented in [Section 1.4](#) we show in [Table 6](#), as an example, both types of algorithms for the particular case of K -d trees. These differences for other types of point search trees are totally analogous.

$\mathcal{P}'_{n,\mathbf{u}}$ is the random variable representing the cost of the randomized partial match algorithm with the pattern \mathbf{u} and a random search tree of size n as input.

Table 6: Partial match algorithms for K -d trees

<i>K</i> -d tree Randomized PM algorithm	<i>K</i> -d tree PM algorithm
Input: A query pattern $\mathbf{u} = u_0 u_1 \cdots u_{K-1}$ where each $u_i \in \{S, *\}$ and a K -d tree T .	Input: A query $\mathbf{q} = (q_0, q_1, \dots, q_{K-1})$ where each $q_i \in [0, 1] \cup \{*\}$ and a K -d tree T .
<pre> procedure RANDOMIZEDPARTIALMATCH(\mathbf{u}, T) if T is a leaf then return $\mathbf{x} \leftarrow$ key of the root of T $i \leftarrow$ discriminant of the root of T $\triangleright T_L$ is the left subtree of T $\triangleright T_R$ is the right subtree of T if $u_i = *$ then RANDOMIZEDPARTIALMATCH(\mathbf{u}, T_L) RANDOMIZEDPARTIALMATCH(\mathbf{u}, T_R) else $\triangleright \text{BB}(\mathbf{x}) = \text{Bounding box}(\mathbf{x})$ $\triangleright \text{BB}(\mathbf{x}) = [a_0, b_0] \times \cdots \times [a_{K-1}, b_{K-1}]$ Generate a value $q_i \sim \text{Uniform}(a_i, b_i)$ if $q_i < x_i$ then RANDOMIZEDPARTIALMATCH(\mathbf{u}, T_L) else RANDOMIZEDPARTIALMATCH(\mathbf{u}, T_R) </pre>	<pre> procedure PARTIALMATCH(\mathbf{q}, T) if T is a leaf then return $\mathbf{x} \leftarrow$ key of the root of T if $x_i = q_i, \forall q_i \in [0, 1]$ then Report \mathbf{x} $i \leftarrow$ discriminant of the root of T $\triangleright T_L$ is the left subtree of T $\triangleright T_R$ is the right subtree of T if $q_i = *$ then PARTIALMATCH(\mathbf{q}, T_L) PARTIALMATCH(\mathbf{q}, T_R) else if $q_i < x_i$ then PARTIALMATCH(\mathbf{q}, T_L) else PARTIALMATCH(\mathbf{q}, T_R) </pre>

2.3.3 Fixed Partial Match Queries

In the early 2010s several research groups started to tackle the analysis of partial match algorithms taking into account that the coordinates of the query specified remain fixed throughout all the recursive calls. To emphasize this characteristic, and to distinguish them from the analysis of random PM queries, these types of queries are called *fixed PM queries*.

$\mathcal{P}_{n,\mathbf{q}}$ is the random variable of the cost of the partial match algorithm with the query \mathbf{q} and a random search tree of size n as input.

$\mathcal{P}_{n,\mathbf{r}}$ is the random variable of the cost of the partial match algorithm with a query \mathbf{q} with rank vector \mathbf{r} and a random search tree of size n as input.

2.3.4 Random PM Queries vs Randomized PM Algorithm

Even though the randomized partial match algorithm is not used in practice, its implicit use in the analysis of random PM queries is justified because their expected cost is the same. More precisely:

Proposition 2.1. *Let \mathbf{Q} be a random PM query,*

$$\mathbb{E}[\mathcal{P}'_{n,\mathbf{u}}] = \mathbb{E}[\hat{\mathcal{P}}_{n,\mathbf{Q}} | \mathbf{Q} \text{ is a random query with pattern } \mathbf{u}]$$

or simply $P'_{n,\mathbf{u}} = \hat{P}_{n,\mathbf{u}}$.

This proposition has been widely used, many times implicitly, in the analysis of random PM queries.

Note that, given that the second moment is not linear, for their variances we have

$$\mathbb{V}[\mathcal{P}'_{n,\mathbf{u}}] \neq \mathbb{V}[\hat{\mathcal{P}}_{n,\mathbf{Q}} | \mathbf{Q} \text{ is a random query with pattern } \mathbf{u}],$$

or simply $V'_{n,\mathbf{u}} \neq \hat{V}_{n,\mathbf{u}}$.

2.3.5 Random PM Queries vs Fixed PM Queries

There is an alternative way to define the random variable $\hat{\mathcal{P}}_{n,\mathbf{u}}$ based on $\mathcal{P}_{n,\mathbf{r}}$. Remember that in the random search tree model the $n!^K$ possible input sequences of rank vectors are equiprobable. If we were to insert a new key, the $(n+1)!^K$ new permutations would be equiprobable as well. We have an extra factor $(n+1)^K$ due to the new key because in each coordinate there are $n+1$ equally likely ranks for that coordinate of the new key. We can think of a random PM query \mathbf{Q} as a potential “new” key where only s coordinates have been specified and therefore there are $(n+1)^s$ possibilities for its rank vector. More formally, given a pattern \mathbf{u} , we define the set of possible rank vectors with pattern \mathbf{u} as:

$$R_{n,\mathbf{u}} = \left\{ \mathbf{r} = (r_0, \dots, r_{K-1}) \mid r_i = * \text{ if } u_i = * \text{ and } 0 \leq r_i \leq n \text{ if } u_i = S \right\}$$

then $\hat{\mathcal{P}}_{n,\mathbf{u}}$ is $\mathcal{P}_{n,\mathcal{R}}$, where \mathcal{R} is taken uniformly at random among the $(n+1)^s$ elements of $R_{n,\mathbf{u}}$.

The definition above implies an important relationship between the expected costs of random and fixed PM queries that we use to compute the constants in the expressions that we obtain for the latter.

Proposition 2.2. *Given a pattern \mathbf{u} of specified/unspecified coordinates, the expected cost, $\hat{P}_{n,\mathbf{u}}$, of a random PM query with pattern \mathbf{u} is the average of the expected costs of all the possible fixed PM queries $P_{n,\mathbf{r}}$, that have the same pattern:*

$$\hat{P}_{n,\mathbf{u}} = \frac{\sum_{\mathbf{r} \in R_{n,\mathbf{u}}} P_{n,\mathbf{r}}}{(n+1)^s}.$$

Higher order moments of $\hat{\mathcal{P}}_{n,\mathbf{u}}$ can be obtained from the higher order moments of $\mathcal{P}_{n,\mathbf{r}}$.

2.3.6 Fixed PM Queries vs Fixed PM Ranks

On [Section 3.1](#), Queries and ranks, of [\[DLM16a\]](#) we explain in detail the close relationship between $P_{n,\mathbf{q}}$ and $P_{n,\mathbf{r}}$.

Proposition 2.3. *Assume that the coordinates of the random search tree are uniformly distributed on the interval $[0, 1]$. Given a fixed PM query $\mathbf{q} = (q_0, \dots, q_{K-1})$, if we define $\bar{\mathbf{r}} = n\mathbf{q}$ as $\bar{r}_i = nq_i$ if $q_i \neq *$, else otherwise $\bar{r}_i = *$, then*

$$P_{n,\mathbf{q}} = \Theta(P_{n,\bar{\mathbf{r}}}).$$

Proposition 2.4. *Assume that the coordinates of the random search tree are uniformly distributed on the interval $[0, 1]$. Given a rank vector $\mathbf{r} = (r_0, \dots, r_{K-1})$, if we define $\bar{\mathbf{q}} = \mathbf{r}/n$ as $\bar{q}_i = r_i/n$ if $0 \leq r_i \leq n$, else otherwise $\bar{q}_i = *$, then*

$$P_{n,\mathbf{r}} = \Theta(P_{n,\bar{\mathbf{q}}}).$$

Chapter 3

Methods

In this chapter we present a few of the general mathematical techniques used to carry out the theoretical analysis, as well as the experimental methods that we applied to validate the theoretical models and to estimate their range of applicability. For example, several of our results apply in the asymptotic regime, that is, when “ n is large enough”. Careful experiments allow us to conclude that with $n \geq 5000$ —and even less—the asymptotic estimates that we obtain from the theoretical analysis give very good approximations and elicit meaningful predictions about the practical performance.

3.1 Mathematical Tools

3.1.1 Generating functions

The use of generating functions to solve linear recurrences (also called difference equations) has a long history, having been introduced by Abraham de Moivre early in the 18th century [Knu97]. We illustrate how they can be used with one of the first sequences analyzed this way, the Fibonacci sequence:

$$\begin{aligned} F_0 &= 0, & F_1 &= 1, \\ F_n &= F_{n-1} + F_{n-2} & \text{for } n &\geq 2. \end{aligned} \tag{3.1}$$

This can be seen as a linear difference equation that is homogeneous and has constant coefficients. Its generating function is the formal power series¹:

$$F(z) = \sum_{n \geq 0} F_n z^n. \tag{3.2}$$

¹A formal power series is a polynomial with an infinite number of terms. Unlike a power series its variable cannot take a numerical value so convergence is not an issue.

If we multiply (3.2) by z and by z^2 we obtain respectively:

$$zF(z) = \sum_{n \geq 0} F_n z^{n+1} = \sum_{n \geq 1} F_{n-1} z^n \quad (3.3)$$

$$z^2 F(z) = \sum_{n \geq 0} F_n z^{n+2} = \sum_{n \geq 2} F_{n-2} z^n \quad (3.4)$$

Then we compute (3.2) - (3.3) - (3.4):

$$(1 - z - z^2)F(z) = F_0 + (F_1 - F_0)z + \sum_{n \geq 2} (F_n - F_{n-1} - F_{n-2})z^n$$

and using the definition (3.1)—including the initial conditions—the right side simplifies to z . Therefore:

$$F(z) = \frac{z}{1 - z - z^2}.$$

Factorizing the denominator and doing some algebra we get:

$$F(z) = \frac{z}{(1 - \varphi z)(1 - \hat{\varphi} z)}. \quad (3.5)$$

where φ is the golden ratio $(1 + \sqrt{5})/2$ and $\hat{\varphi} = (1 - \sqrt{5})/2$. Using partial fraction decomposition:

$$F(z) = \frac{1}{\sqrt{5}} \left(\frac{1}{1 - \varphi z} - \frac{1}{1 - \hat{\varphi} z} \right).$$

Using the geometric series identity $\sum_{n \geq 0} x^n = 1/(1 - x)$:

$$F(z) = \frac{1}{\sqrt{5}} \left(\sum_{n \geq 0} \varphi^n z^n - \sum_{n \geq 0} \hat{\varphi}^n z^n \right) = \sum_{n \geq 0} \frac{\varphi^n - \hat{\varphi}^n}{\sqrt{5}} z^n.$$

Given that two formal power series are equal if and only if all their coefficients are equal, we have a solution to the difference equation (3.1):

$$F_n = \frac{\varphi^n - \hat{\varphi}^n}{\sqrt{5}}.$$

We can compute the asymptotic behavior of the Fibonacci sequence by elementary considerations. Given that $\varphi \approx 1.618 > 1$ and $|\hat{\varphi}| \approx 0.618 < 1$, φ^n increases with n and $|\hat{\varphi}^n|$ decreases with n . Therefore for large values of n we have:

$$F_n \approx \frac{\varphi^n}{\sqrt{5}}. \quad (3.6)$$

We can use the expression of $F(z)$ as a rational function (3.5) to give a very basic example of singularity analysis. This is a very powerful technique developed by Flajolet and Odlyzko [FO90] that considers the generating functions

not only as formal power series but as functions in a circle of convergence in the complex plane (as suggested by the use of z for the variable name).

In the case of a rational generating function that has a unique pole² $1/\beta$ of smallest absolute value, the dominant term of the sequence is $C\beta^n n^{\nu-1}$ where ν is the multiplicity of $1/\beta$ and C is a constant for which there is an explicit formula; see Theorem 4.1 (Asymptotics of linear recurrences) in [FS13].

For the Fibonacci sequence, given (3.5), it is clear that the singularities of $F(z)$ are $1/\varphi$ and $1/\hat{\varphi}$. The dominant singularity is the one with smaller absolute value, so in this case it is $1/\varphi$, and its multiplicity is 1. The explicit formula for C gives the value $1/\sqrt{5}$, therefore the dominant term of the sequence is $\varphi^n/\sqrt{5}$ as stated in (3.6).

In a similar way, in singularity analysis there are a variety of transfer lemmas that allow direct deduction of the asymptotic behavior of a sequence from the asymptotic expansion of its generating function near its dominant singularities. It is a standard tool of analytic combinatorics [FS09]. The use of complex analysis to obtain asymptotic estimates in analytic combinatorics is similar to its use in analytic number theory.

Wilf’s *generatingfunctionology* [Wil90] is a textbook on generating functions while Flajolet and Sedgewick’s *An Introduction to the Analysis of Algorithms* [FS13] introduces them, as well as analytic combinatorics, with applications to the analysis of algorithms. The application of generating functions to solve linear difference equations is totally analogous to the application of the Laplace transform to solve linear differential equations. Mackey [Mac80] explains that the use of generating functions in probability theory³ is one of three independent origins of harmonic analysis, the other two being analytic number theory and mathematical physics.

3.1.2 Random Partial Match Queries

We analyze random PM queries in [DLM14; DLM16a; DL17; DLM16b]. In all those cases, either explicitly or implicitly, we start by finding a linear recurrence, or system of linear recurrences, for the expected cost(s) $\hat{P}_{n,\mathbf{a}}$ of the random PM queries, where \mathbf{a} represents one or more additional parameters. We then define the generating function(s) $\hat{P}_{\mathbf{a}}(z)$ for the sequence(s) $\hat{P}_{n,\mathbf{a}}$:

$$\hat{P}_{\mathbf{a}}(z) = \sum_{n \geq 0} \hat{P}_{n,\mathbf{a}} z^n.$$

We then translate the linear recurrence(s) into an integral equation (system of integral equations) for the generating function(s) $\hat{P}_{\mathbf{a}}(z)$. From these equations, taking derivatives or applying some convenient differential operators, we

²A pole of a rational function in reduced form is a value that makes the denominator zero.

³One of the first applications of generating functions was to compute probabilities. In 1718 Abraham de Moivre introduced, at least implicitly, probability generating functions in *The Doctrine of Chances*, the first textbook on probability theory. Pierre-Simon Laplace in *Théorie analytique des probabilités*, published in 1812, coined the term “generating functions” and extended the theory of difference equations to equations in several variables [Sea49; Mac80].

deduce a linear differential equation (system of linear differential equations) with variable coefficients for the generating function(s) $\hat{P}_{\mathbf{a}}(z)$.

Only on [Theorem 2](#) of [\[DLM14\]](#) are we able to solve the differential equation that we derive, but that is not a requisite to be able to obtain the asymptotic estimate for $\hat{P}_{n,\mathbf{a}}$. In [\[DLM16b\]](#) we continue the analysis using Roura's Continuous Master Theorem (CMT) [\[Rou01\]](#). This is a powerful result to derive the asymptotic behavior of the solution of divide-and-conquer recurrences, such as those arising in the analysis of random PM queries [\[Duc04; DEM98\]](#). Given its importance we include it in [Appendix A.1](#).

As an example of how to apply Roura's CMT, we now apply it to the recurrence that appears in the proof of [Theorem 5](#) of [\[DLM16a\]](#) (also [Theorem 2](#) of [\[DLM14\]](#)). For the expected cost \hat{P}_{n,ρ,ρ_0} of random PM queries with s specified coordinates, $0 < s < K$, and s_0 extreme specified coordinates, $0 \leq s_0 \leq s$, in a random relaxed K -d tree of size n we have there the following recurrence (in this section to abbreviate we just call it \hat{P}_n):

$$\begin{aligned} \hat{P}_0 &= 0, \\ \hat{P}_n &= 1 + \frac{2(s-s_0)}{K} \frac{1}{n} \sum_{j=0}^{n-1} \frac{j+1}{n+1} \hat{P}_j + \frac{2(K-s)+s_0}{K} \frac{1}{n} \sum_{j=0}^{n-1} \hat{P}_j \quad \text{for } n \geq 1. \end{aligned} \quad (3.7)$$

Using $\rho = s/K$ and $\rho_0 = s_0/K$, we can rewrite the recurrence as

$$\hat{P}_n = 1 + \sum_{j=0}^{n-1} \left(\frac{2(\rho - \rho_0)}{n} \frac{j+1}{n+1} + \frac{2-2\rho+\rho_0}{n} \right) \hat{P}_j.$$

Using the terminology of Roura's CMT ([Definition 16](#)):

$$t_n = 1, \quad \omega_{n,j} = \frac{2(\rho - \rho_0)}{n} \frac{j+1}{n+1} + \frac{2-2\rho+\rho_0}{n}.$$

In order to apply [Lemma A.1](#), we rewrite the weights of the recurrence in the following way:

$$\omega_{n,j} = \frac{(2-2\rho+\rho_0)n + 2(\rho - \rho_0)j + 2 - \rho_0}{n(n+1)}$$

and we define

$$\omega(z) := (2-2\rho+\rho_0) + 2(\rho - \rho_0)z.$$

Then we compute

$$\int_0^1 \omega(z) dz = \int_0^1 \left((2-2\rho+\rho_0) + 2(\rho - \rho_0)z \right) dz = 2 - \rho$$

which, given that $\rho < 1$, is greater than 1. Therefore the conditions of [Lemma A.1](#) are satisfied, so the recurrence for \hat{P}_n is continuous and $\omega(z)$ is its shape function.

Now in order to apply [Theorem A.2](#) we note that the toll function is of the form $t_n \sim Kn^a \ln^b(n)$ with $K = 1$, $a = 0$ and $b = 0$ and we compute:

$$H = 1 - \int_0^1 z^a \omega(z) dz = 1 - (2 - \rho) = \rho - 1,$$

which, given that $\rho < 1$, is lower than 0. Therefore we can apply the third case of [Theorem A.2](#) and, given that $\hat{P}_n \geq 0$ for every $n \geq 0$, get

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

where α is the unique non-negative real solution to the equation

$$\int_0^1 z^\alpha \omega(z) dz = 1.$$

Computing the integral this equation becomes:

$$(\rho - \rho_0) \frac{2}{\alpha + 2} + (1 - \rho + \rho_0/2) \frac{2}{\alpha + 1} = 1. \quad (3.8)$$

It is easy to verify that, given $0 < \rho < 1$ and $0 \leq \rho_0 \leq \rho$, when $\alpha = 0$ the left-hand side of this equation is greater than 1 and when $\alpha = 1$ it is lower than 1, therefore it has a solution in $(0, 1)$.

We know from singularity analysis, see below, that

$$\hat{P}_n = \beta n^\alpha + o(n^\alpha)$$

for a constant β . However the CMT can only give us here that $\hat{P}_n = \Theta(n^\alpha)$ and cannot be used to establish the existence of β , let alone get its value. On the other hand, the CMT gives the precise value of α and is much easier to understand and be applied by non-experts than other more sophisticated mathematical techniques. We recommend the reader to see [\[Rou97\]](#) for a detailed description of the application of the CMT to the analysis of random PM queries in standard K -d trees.

The solution in $(0, 1)$ to equation [\(3.8\)](#) is the one given on [Theorem 2](#) of [\[DLM14\]](#):

$$\alpha = \frac{1}{2} \left(\sqrt{(3 - \rho_0)^2 - 8(\rho - \rho_0)} - 1 - \rho_0 \right). \quad (3.9)$$

In [\[DLM14; DLM16a; DLM18\]](#) instead of using Roura's CMT we decided to use singularity analysis. Given that we use it so frequently we include the relevant Transfer Lemma in [Appendix A.2](#). To contrast both methods we are going to analyze the same example, that is recurrence [\(3.7\)](#), using singularity analysis. The first step is to define the generating function

$$\hat{P}(z) = \sum_{n \geq 0} \hat{P}_n z^n.$$

Then we multiply the recurrence (3.7) by z^n and add for all $n \geq 0$. Using operations on generating functions like the following [FS13], given $A(z) = \sum_{n \geq 0} a_n z^n$:

$$\begin{aligned} zA(z) &= \sum_{n \geq 1} a_{n-1} z^n, \\ A'(z) &= \sum_{n \geq 0} (n+1) a_{n+1} z^n, \\ \frac{A(z)}{1-z} &= \sum_{n \geq 0} \left(\sum_{0 \leq k \leq n} a_k \right) z^n, \end{aligned}$$

we obtain the differential equation:

$$\hat{P}''(z) + \hat{P}'(z) \frac{2 - (4 - \rho_0)z}{z(1-z)} - \hat{P}(z) \frac{4 - 2\rho - (2 - \rho_0)z}{z(1-z)^2} = \frac{2}{z(1-z)^3}.$$

That is a second order linear differential equation with initial conditions $\hat{P}(0) = 0$ and $\hat{P}'(0) = 1$ (because $\hat{P}_0 = 0$ and $\hat{P}_1 = 1$). With similar techniques to [MPP01] we are able to find the solution—mentioned in [DLM16a]—of this differential equation:

$$\hat{P}(z) = \frac{1}{1-\rho} \left(\frac{{}_2F_1\left(\begin{matrix} 1-\alpha-\rho_0, -\alpha \\ 2 \end{matrix} \middle| z\right)}{(1-z)^{\alpha+1}} - \frac{1}{1-z} \right),$$

where ${}_2F_1\left(\begin{matrix} a, b \\ c \end{matrix} \middle| z\right)$ denotes the hypergeometric function [DLMF, Chap. 15] and α is as in (3.9).

To get the asymptotic estimate for \hat{P}_n we only need to study the asymptotic behavior of $\hat{P}(z)$ near its dominant singularity at $z = 1$. The second term of $\hat{P}(z)$ is negligible and the hypergeometric function is analytic at $z = 1$, therefore:

$$\hat{P}(z) \sim \beta'(1-z)^{-\alpha-1} + O((1-z)^{-1})$$

with $\beta' = \frac{1}{1-\rho} {}_2F_1\left(\begin{matrix} 1-\alpha-\rho_0, -\alpha \\ 2 \end{matrix} \middle| 1\right)$. We can evaluate this using the identity [DLMF, Subsec. 15.4(ii)]

$${}_2F_1\left(\begin{matrix} a, b \\ c \end{matrix} \middle| 1\right) = \frac{\Gamma(c)\Gamma(c-a-b)}{\Gamma(c-a)\Gamma(c-b)}.$$

Using the transfer lemma Lemma A.3 of Flajolet and Odlyzko [FO90; FS09] we have

$$\hat{P}_n \sim \frac{\beta'}{\Gamma(\alpha+1)} n^\alpha.$$

Therefore:

$$\beta = \frac{1}{1-\rho} \frac{\Gamma(2\alpha+1+\rho_0)}{\Gamma(\alpha+1+\rho_0)(\alpha+1)\alpha^2\Gamma^2(\alpha)}.$$

Note that—when we are able to solve the differential equation for the generating function—this method allows us to find a closed analytic formula not only for α but also for β in terms of K , s and s_0 .

Another method to analyze random PM queries that we used was the one developed in [Lau88; CLF89] based on classic asymptotic analysis [Bru58], see Appendix B.1 for details.

On a side note, in the literature other methods have been applied over the years to the analysis of random PM queries. An elementary probabilistic method is used in [Zam00; CDZ01]. The contraction method introduced by Rösler [Rös91] to analyze Quicksort is applied for instance in [Nei99; Nei00]. This powerful technique has been used, among other results, to analyze the variance and the limiting distribution of both random and fixed PM queries. For the mathematically inclined reader, in this method recursive sequences of distributions of random variables are seen as functional equations in metric spaces. Then the limits of those sequences of distributions are characterized as fixed points of contraction maps.

3.1.3 Fixed Partial Match Queries

We analyze fixed PM queries in [DLM14; DLM16a; DL17; DLM18]. In all those articles we take advantage of the equivalences explained on Section 2.2 and we analyze the expected cost $P_{n,r} = \mathbb{E}[\mathcal{P}_{n,r}]$ instead of the expected cost $P_{n,q} = \mathbb{E}[\mathcal{P}_{n,q}]$. To focus on the ranks of the query’s coordinates instead of their values allows us to use a combinatorial approach.

We then find recurrence(s) for $P_{n,r}$. These are considerably more complex than the ones obtained for random PM queries because we need to take into account several cases; for example, if the discriminating coordinate of the root is specified or not and in which subtrees the partial match search should continue. Again, we define the generating functions $P_{\mathbf{r}}(z)$ for the sequences that we study:

$$P_{\mathbf{r}}(z) = \sum_{n \geq 0} P_{n,\mathbf{r}} z^n$$

With considerably more work than in the case of random PM queries, we translate the linear recurrence(s) into an integral equation (system of integral equations) for the generating function(s) $P_{\mathbf{r}}(z)$.

From these equations, taking derivatives or applying some convenient differential operators, we deduce a linear differential equation (system of linear differential equations) with variable coefficients for the generating function(s) $P_{\mathbf{r}}(z)$.

To solve differential equations, to bound errors between sums and integral approximations and to prove that the solutions found are indeed solutions of the integral equations we use a variety of results from classical analysis including:

- Solutions to the hypergeometric differential equation [DLMF, Sec. 15.10].
- Laplace’s method [FS13] for estimating values of sums and integrals.

- Euler-Maclaurin summation formula again for estimating the value of sums and integrals [DLMF, Subsec. 2.10(i)].
- De Moivre-Laplace limit theorem [Fel68] to approximate sums with the normal distribution.
- Binomial approximation to the hypergeometric distribution [JKK92].
- Holmgren’s theorem and Cauchy-Kovalevskaya theorem [Fol95; Zwi97]) for the existence and uniqueness of solutions to analytic partial differential equations.

Finally, during the work done to analyze fixed PM queries in [DLM16a] we discovered a lemma that significantly simplifies such analysis as it leads to a proof that does not require knowledge of the expected cost of random PM queries with extreme specified coordinates, that is [Theorem 5](#) of [DLM16a] (also [Theorem 2](#) of [DLM14]). This discovery was made too late to be included in that article but we consider it worth mentioning here. Due to its detailed and technical nature we leave the statement and the proof of such lemma to [Appendix B.2](#).

3.2 Experiments

As Flajolet and Sedgewick state in *An Introduction to the Analysis of Algorithms* 2nd ed. [FS13], p. 5: “This approach is *scientific*: we build mathematical models to describe the performance of real-world algorithm implementations, then use these models to develop hypotheses that we validate through experimentation.” In order to test our results we build actual implementations of the point search trees and execute on them the partial match algorithms that we are studying. We count the number of nodes visited in each partial match search, that being the measure of the cost we use. Based on statistics of those costs we compute empirical estimates of the parameters that describe the expected costs— α , β and ν —and compare them with the theoretical values that we predict. This way we can have reasonable estimates of the errors introduced by some approximations and can also get an idea of the magnitude of n for which the asymptotic regime is reached.

There are other cases where we are able to perform experiments before having theoretical results at all, when we only have conjectures. That occurred during the work of [DLM16a] where our experiments helped us to refine the conjecture proposed there for squarish K -d trees.

Each run of our experiments for fixed PM queries can be described by the following parameters:

- \mathcal{T} , the type of point search tree
- \mathbf{q} , the query
- n , the size of the trees in the sample

- M , the size of the sample

For each run we generate M random point search trees of type \mathcal{T} and size n . In each tree we perform a PM search with query \mathbf{q} , counting the total number of visited nodes and taking the corresponding sample mean of those realized costs.

For fixed PM queries we performed a variety of ad-hoc experiments described in [DLM14; DLM16a]. Here we present some experimental results obtained for fixed PM queries in quad trees after the article [DLM18].

Given the symmetry of quad trees we assume, without loss of generality, that the s specified coordinates of \mathbf{q} are the first s coordinates, $0 < s < K$, and therefore that $\mathbf{q} = (q_0, \dots, q_{s-1}, *, \dots, *)$. We write $\mathbf{q} = (q_0, \dots, q_{s-1})$ with the convention that the implicit $K - s$ remaining components are all $*$'s.

In all the graphs the theoretical curves or surfaces are the smooth ones. For the case $K = 2$, $s = 1$, Figure 16 shows the theoretical and empirical estimates of $P_{n,\mathbf{q}}$ as a function of q_0 with a sample size $M = 500$. Figure 17 shows for the case $K = 6$, $s = 3$ the estimates as a function of q_2 for some fixed values of q_0 and q_1 with a sample size $M = 100$. Finally, Figure 18 shows for the same case the estimates as a function of q_1 and q_2 for some fixed values of q_0 again with a sample size $M = 100$. In all cases there is a good match between the theoretical estimates and the empirical ones, similar to the ones obtained for K -d trees in [DLM14; DLM16a].

All programs used in the experiments for quad trees were written in the C++ programming language and compiled with the GNU gcc compiler version v4.2.1. The experiments were run on an Intel Core i5, 3.1 GHz, 4 cores processor.

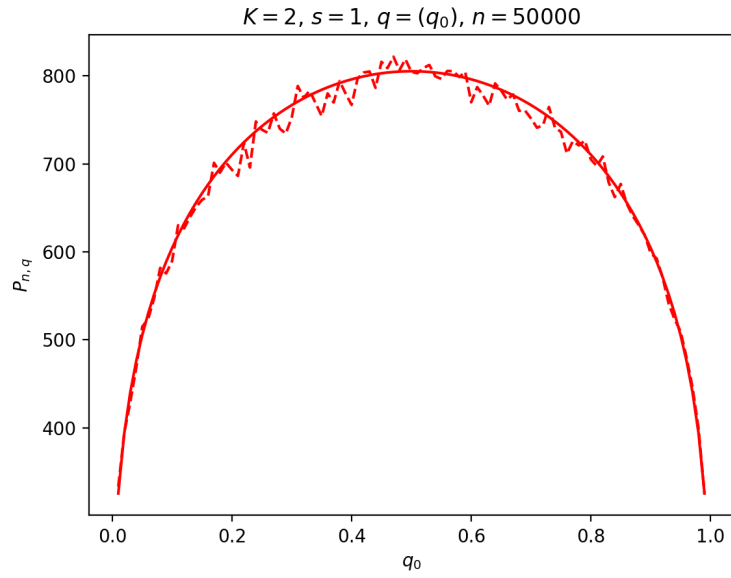


Figure 16: Quadrees Experiments vs Theory 1

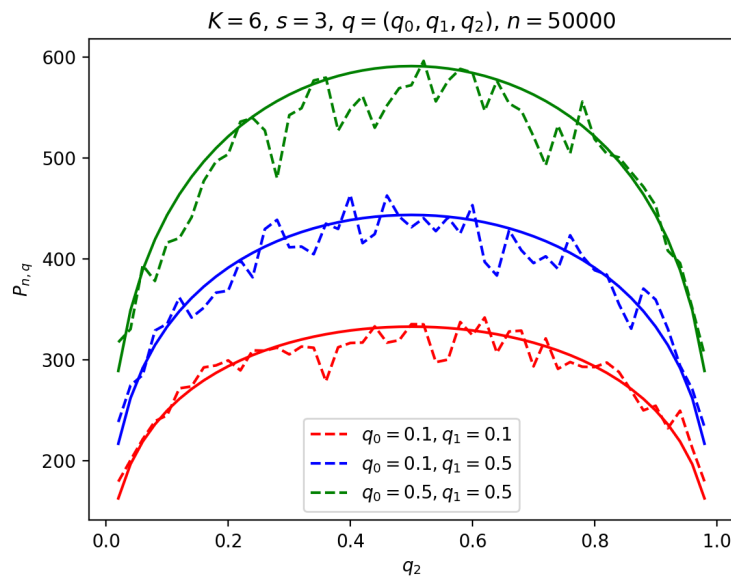
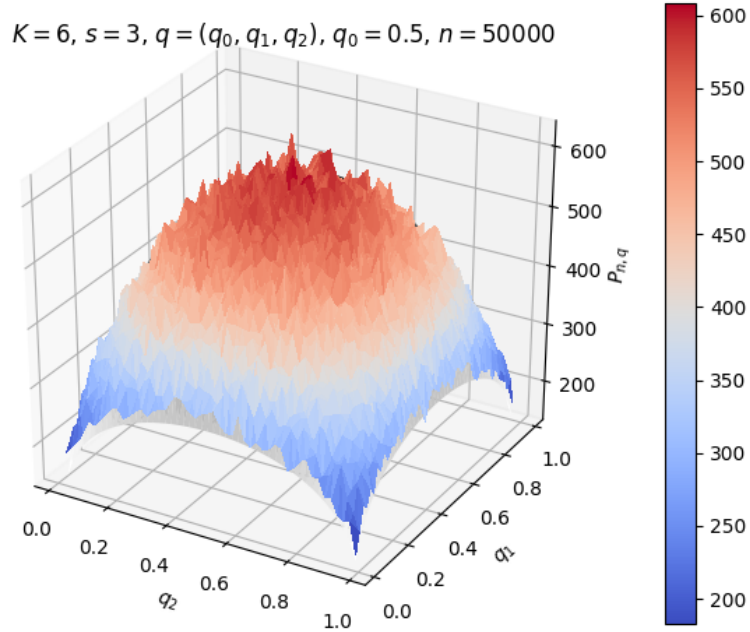
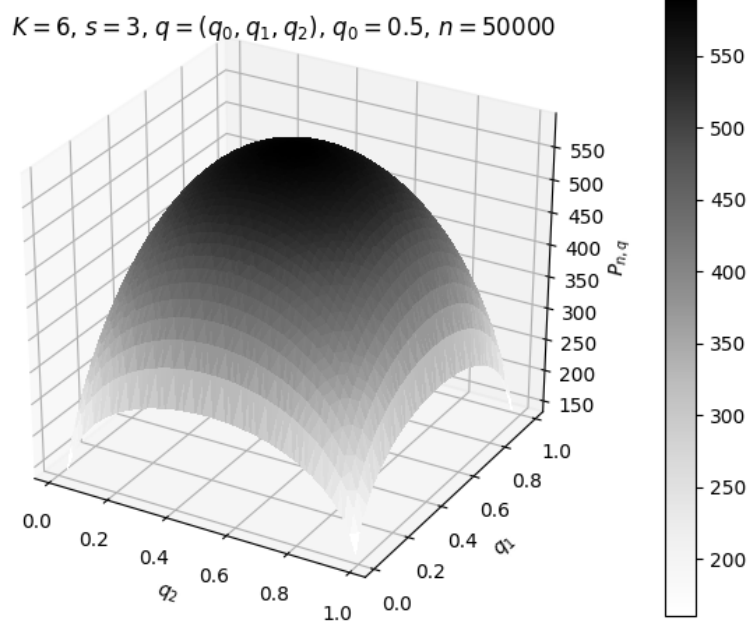


Figure 17: Quadrees Experiments vs Theory 2



Experiment



Theory

Figure 18: Quadrees Experiments vs Theory 3

Chapter 4

State of the Art

In this chapter we present the results for both random and fixed PM queries that were known prior to this work.

4.1 Random Partial Match Queries

4.1.1 Standard K -d trees

In 1986 Flajolet and Puech, in their influential paper [FP86], proved the below theorems for random PM queries in random K -d trees and K -d tries. They analyzed standard K -d trees. One reason why this paper is so important is because prior to it the analysis of PMs assumed, wrongly, that the average performance of the K -d tree would be as if it were perfectly balanced. Another reason is that it was one of the first examples of the use of singularity analysis to obtain asymptotic estimates, something which Flajolet and Odlyzko would systematize in [FO90].

Theorem 4.1 (Flajolet and Puech [FP86]). *The expected cost $\hat{P}_{n,\mathbf{u}}$ of a random PM query with pattern \mathbf{u} —where exactly s , $0 < s < K$, coordinates are specified—in a random standard K -d tree of size n satisfies*

$$\hat{P}_{n,\mathbf{u}} = \beta_{\mathbf{u}} n^{\alpha_{\rho}} + o(n^{\alpha_{\rho}}), \quad (4.1)$$

where $\beta_{\mathbf{u}}$ depends on the query pattern \mathbf{u} and α_{ρ} depends on the ratio $\rho = s/K$ due to it being the unique solution in the interval $(0, 1)$ of the equation:

$$\left(\frac{2}{\alpha+2}\right)^{\rho} \left(\frac{2}{\alpha+1}\right)^{1-\rho} = 1. \quad (4.2)$$

This paper—[FP86]—only contains some numeric values for $\beta_{\mathbf{u}}$. At that time finding an explicit formula for $\beta_{\mathbf{u}}$ was considered a difficult open problem. In 2006 Chern and Hwang [CH06] published explicit, although quite intricate,

expressions for the $\beta_{\mathbf{u}}$ of [Theorem 4.1](#). We refer the reader to that article for the details.

As we shall see, similar results to the ones obtained by Flajolet and Puech for random PM queries were then proved for other variants of K -d trees and other multidimensional data structures.

4.1.2 Standard K -dt trees

In 1988 Cunto, Lau and Flajolet [[Lau88](#); [CLF89](#)] generalized [Theorem 4.1](#) for standard K -dt trees. We denote by $x^{\overline{k}}$ the rising factorial power $x(x+1)\dots(x+k-1)$ [[GKP94](#)].

Theorem 4.2 (Cunto, Lau and Flajolet [[Lau88](#); [CLF89](#)]). *The expected cost $\hat{P}_{n,t,\mathbf{u}}$ of a random PM query with pattern \mathbf{u} —where exactly s , $0 < s < K$, coordinates are specified—in a random standard K -dt tree of size n satisfies*

$$\hat{P}_{n,t,\mathbf{u}} = \beta_{t,\mathbf{u}} n^{\alpha_{t,\rho}} + o(n^{\alpha_{t,\rho}}),$$

where $\beta_{t,\mathbf{u}}$ depends on t and the query pattern \mathbf{u} and $\alpha_{t,\rho}$ depends on t and the ratio $\rho = s/K$ due to it being the unique solution in the interval $(0,1)$ of the equation:

$$\left(\frac{(t+2)^{\overline{t+1}}}{(\alpha+t+2)^{\overline{t+1}}} \right)^{\rho} \left(\frac{(t+2)^{\overline{t+1}}}{(\alpha+t+1)^{\overline{t+1}}} \right)^{1-\rho} = 1. \quad (4.3)$$

[Theorem 4.1](#) is the particular case $t = 0$ of [Theorem 4.2](#).

Cunto, Lau and Flajolet also proposed, for the first time, a formula for the variance of the cost of random PM queries for K -d trees as well as K -dt trees. In that work the distinction between random PM queries and the randomized partial match algorithm—that is between $\mathcal{P}'_{n,t,\mathbf{u}}$ and $\hat{\mathcal{P}}_{n,t,\mathbf{u}}$ —was not made explicitly.

Theorem 4.3 (Cunto, Lau and Flajolet [[Lau88](#); [CLF89](#)]). *The variance $V'_{n,t,\mathbf{u}}$ of the randomized partial match algorithm—where exactly s , $0 < s < K$, coordinates are specified—in a random standard K -dt tree of size n satisfies*

$$V'_{n,t,\mathbf{u}} = \beta'_{t,\mathbf{u}} n^{2\alpha_{t,\rho}} + o(n^{2\alpha_{t,\rho}}),$$

where $\beta'_{t,\mathbf{u}}$ depends on t and the pattern \mathbf{u} and $\alpha_{t,\rho}$ is the same as in [Theorem 4.2](#).

In 2006 Chern and Hwang [[CH06](#)] published, in the same article where they provided expressions for the $\beta_{\mathbf{u}}$ of [Theorem 4.1](#), even more intricate expressions for the $\beta_{t,\mathbf{u}}$ of [Theorem 4.2](#), that is the leading constant coefficients in the asymptotic approximations of the expected cost of PMs in K -dt trees. We again refer the reader to that article for details.

4.1.3 Quad trees

In the early 1990s Flajolet, Gonnet, Puech, and Robson published the analysis of random PM queries in quad trees; the final version was published in 1993 [Fla+93].

Theorem 4.4 (Flajolet, Gonnet, Puech, and Robson [Fla+93]). *The expected cost $\hat{P}_{n,s,K}$ of a random PM query—where exactly s , $0 < s < K$, coordinates are specified—in a random quad tree of size n satisfies*

$$\hat{P}_{n,s,K} = \beta_{s,K} n^{\alpha_\rho} + o(n^{\alpha_\rho}),$$

where $\beta_{s,K}$ depends on s and K and α_ρ is the same as in [Theorem 4.1](#).

In the early 2000s Chern and Hwang [CH03] used a new approach for the analysis of random PM queries in quadtrees that allowed them to obtain an explicit expression for the constant $\beta_{s,K}$ of [Theorem 4.4](#).

Theorem 4.5 (Chern and Hwang [CH03]). *The leading constant coefficient in the asymptotic approximation of the expected cost of a random PM query—where exactly s , $0 < s < K$, coordinates are specified—in a random standard quad tree of size n , that is the $\beta_{s,K}$ of [Theorem 4.4](#), is*

$$\beta_{s,K} = \frac{1}{(2^{K-s} - 1)\Gamma(\alpha_\rho)^{K-s}\Gamma(\alpha_\rho + 1)^s} \prod_{2 \leq j \leq K} \frac{\Gamma(\alpha_\rho - \alpha_j)}{\Gamma(1 - \alpha_j)}, \quad (4.4)$$

where Γ is the Gamma function and the α_j 's are the solutions of the indicial equation (4.2):

$$\alpha_\rho = \alpha_1 > \Re(\alpha_2) \geq \dots \geq \Re(\alpha_K).$$

4.1.4 Relaxed K -d trees

Some initial analysis results for random PM queries in relaxed K -d trees were obtained in 1998 by Duch, Estivill-Castro and Martínez [DEM98]. The analysis was completed by Martínez, Panholzer, and Prodinger in [MPP01].

Theorem 4.6 (Duch *et al.* [DEM98], Martínez *et al.* [MPP01]). *The expected cost $\hat{P}_{n,\rho}$ of a random PM query—where exactly s , $0 < s < K$, coordinates are specified—in a random relaxed K -d tree of size n satisfies*

$$\hat{P}_{n,\rho} = \beta_\rho n^{\alpha_\rho} + o(n^{\alpha_\rho}),$$

where both β_ρ and α_ρ depend on the ratio $\rho = s/K$:

$$\alpha_\rho = \frac{1}{2} \left(\sqrt{9 - 8\rho} - 1 \right)$$

and

$$\beta_\rho = \frac{\Gamma(2\alpha + 1)}{(1 - \rho)(\alpha + 1)\alpha^3\Gamma^3(\alpha)}.$$

Note that α_ρ is the unique solution in the interval $(0, 1)$ of the equation:

$$\rho \frac{2}{\alpha + 2} + (1 - \rho) \frac{2}{\alpha + 1} = 1. \quad (4.5)$$

Even though the exponent α_ρ of relaxed K -d trees is higher than the one for standard ones, the analysis of the relaxed case is simpler given that their definition leads to a single recurrence equation instead of a system of recurrence equations. This allows an easier way to obtain a formula for β_ρ .

Martínez, Panholzer, and Prodinger [MPP01] also gave a formula for the variance of the cost of random PM queries in relaxed K -d trees. Again, in that work the distinction between random PM queries and the randomized partial match algorithm—that is between $\mathcal{P}'_{n,\rho}$ and $\hat{\mathcal{P}}_{n,\rho}$ —was not made explicitly.

Theorem 4.7 (Martínez, Panholzer, and Prodinger [MPP01]). *The variance $V'_{n,\rho}$ of the randomized partial match algorithm—where exactly s , $0 < s < K$, coordinates are specified—in a random relaxed K -d tree of size n satisfies*

$$V'_{n,\rho} = \beta'_\rho n^{2\alpha_\rho} + o(n^{2\alpha_\rho}),$$

where α_ρ is the same as in [Theorem 4.6](#) and

$$\beta'_\rho = \frac{8\Gamma(2\alpha + 2)}{(\alpha + 1)^2 \alpha^2 (2\alpha + 1)(3\alpha + 1)\Gamma^4(\alpha + 1)} - \frac{4\Gamma^2(2\alpha + 2)}{(\alpha + 1)^4 \alpha^2 (2\alpha + 1)^2 \Gamma^6(\alpha + 1)}.$$

4.1.5 Squarish K -d trees

In 2000 Devroye, Jabour and Zamora-Cura [DJZ00] introduced the squarish K -d trees and analyzed the expected performance of random PM queries in them.

Theorem 4.8 (Devroye, Jabour and Zamora-Cura [DJZ00]). *The expected cost $\hat{P}_{n,s,K}$ of a random PM query—where exactly s , $0 < s < K$, coordinates are specified—in a random squarish K -d tree of size n satisfies*

$$\hat{P}_{n,s,K} = \Theta(n^{\alpha_\rho}),$$

where α_ρ depends on the ratio $\rho = s/K$: $\alpha_\rho = 1 - \rho$.

The constant coefficient hidden in the Θ notation is unknown. Asymptotically the cost of random PM queries in squarish K -d trees is optimal.

4.2 Fixed Partial Match Queries

4.2.1 Quad trees

Only recently have a handful of papers studied the performance of PM search with given fixed queries. In [CJ11], the authors give the expectation of a fixed PM search in 2-dimensional quad trees.

Theorem 4.9 (Curien and Joseph [CJ11]). *For a 2-dimensional random quad tree of size n the expected cost $P_{n,q}$ of a PM search with fixed query $q \in (0, 1)$ where exactly 1 out of the 2 coordinates of the query is specified satisfies*

$$P_{n,q} \sim \nu \cdot (q \cdot (1 - q))^{\alpha/2} \cdot n^\alpha, \quad (4.6)$$

where $\alpha = \alpha_{1/2}$ is the same exponent as in the expected cost of random PM queries (Theorem 4.4), that is α is the unique solution in the interval $(0, 1)$ of equation (4.2) with $\rho = 1/2$:

$$\alpha = \frac{\sqrt{17} - 3}{2}$$

and ν is the following constant:

$$\nu = \frac{\Gamma(2\alpha + 2)\Gamma(\alpha + 2)}{2\Gamma^3(\alpha + 1)\Gamma^2(\frac{\alpha}{2} + 1)}.$$

In 2012, using the contraction method, Broutin, Neininger and Sulzbach [Sul12; BNS12; BNS13] obtained the limit distribution and variance of fixed PM queries in 2-dimensional quad trees.

Theorem 4.10 (Broutin, Neininger and Sulzbach [BNS13]). *For a 2-dimensional random quad tree of size n the variance of the cost $V_{n,q} = \mathbb{V}[P_{n,q}]$ of a PM search with fixed query $q \in (0, 1)$ where exactly 1 out of the 2 coordinates of the query is specified satisfies*

$$V_{n,q} \sim \left(2B(\alpha + 1, \alpha + 1) \frac{2\alpha + 1}{3(1 - \alpha)} - 1 \right) (q \cdot (1 - q))^\alpha \cdot n^{2\alpha}, \quad (4.7)$$

where $B(a, b) := \int_0^1 x^{a-1}(1-x)^{b-1} dx$ denotes the Eulerian beta integral ($a, b > 0$) and α is the same as in Theorem 4.9.

4.2.2 Relaxed K -d trees

In 2012 Duch, Jiménez and Martínez [DJM14] analyzed the expected performance of fixed PM queries in relaxed K -d trees when $s = 1$, that is when exactly one coordinate is specified. They wrote the theorem in terms of ranks instead of queries but, as we have explained on Section 2.3.6, they are equivalent.

Theorem 4.11 (Duch, Jiménez and Martínez [DJM14]). *For a query \mathbf{q} with rank vector $\mathbf{r} = (r_0, \dots, r_{K-1})$ such that $r_0 = z_0 n + o(n)$, $0 < z_0 < 1$ and $r_i = *$ for all i , $1 \leq i < K$, the expected cost of the partial match in a random relaxed K -d tree of size n is*

$$P_{n,\mathbf{r}} = \nu \cdot (z_0(1 - z_0))^{\alpha/2} \cdot n^\alpha + o(n^\alpha),$$

where $\alpha = \alpha_{1/K} = (\sqrt{(9 - 8/K)} - 1)/2$ is the same exponent as in the expected cost of random PM queries, Theorem 4.6, and, with $\beta_{1/K}$ as given in that theorem, ν is:

$$\nu = \beta_{1/K} \frac{\Gamma(\alpha + 2)}{\Gamma^2(\alpha/2 + 1)}.$$

4.2.3 Standard K -d trees

Duch, Jiménez and Martínez [DJM14] also analyzed the expected performance of fixed PM queries in standard K -d trees. Again they did it for the case of only one specified coordinate ($s = 1$) and in terms of ranks instead of queries.

Theorem 4.12 (Duch, Jiménez and Martínez [DJM14]). *For a query \mathbf{q} with rank vector $\mathbf{r} = (r_0, \dots, r_{K-1})$ such that $r_j = z_j n + o(n)$, $0 < z_j < 1$ for some $0 \leq j < K$ and $r_i = *$ for all i , $0 \leq i < K, i \neq j$, the expected cost of the partial match in a random standard K -d tree of size n is*

$$P_{n,\mathbf{r}} = \nu \cdot (z_0(1 - z_0))^{\alpha/2} \cdot n^\alpha + o(n^\alpha),$$

where $\alpha = \alpha_{1/K}$ is the same exponent as in the expected cost of random PM queries, [Theorem 4.1](#), that is it is the unique solution in the interval $(0, 1)$ of equation [\(4.2\)](#) with $\rho = 1/K$:

$$\left(\frac{2}{\alpha + 2}\right)^{1/K} \left(\frac{2}{\alpha + 1}\right)^{1-1/K} = 1,$$

and ν is:

$$\nu = \frac{\Gamma(\alpha + 2)}{\Gamma^2(\frac{1}{2}\alpha + 1)} \beta_j = \frac{\Gamma(\alpha + 2)}{\Gamma^2(\frac{1}{2}\alpha + 1)} \frac{2^j}{(\alpha + 1)^j} \beta_0,$$

where β_j is the constant factor in the cost of random PM queries in a standard K -d tree where only the j -th coordinate is specified.

From the work of Chern and Hwang published in 2006 [CH06], an explicit formula for β_0 can be obtained. We do not include it due to its intricacy.

Part III
Results

Chapter 5

K -d trees

This chapter contains the articles, [DLM14] and [DLM16a], in which we analyze fixed PM queries in K -d trees. Notice that the second article is the journal version of the first so they overlap substantially.

A by-product of the analysis of the algorithms introduced in [DJM14] was the average-case analysis of fixed PM queries with just one specified coordinate in random K -d trees, both relaxed and standard. We continued that analysis first by working on relaxed K -d trees and then on standard ones.

The main result of the first article [DLM14] is an asymptotic formula for the expected cost of fixed PM queries in random relaxed K -d trees for any number s of specified coordinates. That is a generalization of Theorem 4.11 (Theorem 3 of [DJM14]), where $s = 1$ is assumed. See Theorem 1 in page 3 of the first article (page 67 of this thesis).

The main additional result of the second article [DLM16a] is an asymptotic formula for the expected cost of fixed PM queries in random standard K -d trees for any number s of specified coordinates. This result generalizes Theorem 4.12 (Theorem 4 of [DJM14]), where $s = 1$ is assumed, to any s specified coordinates. See Theorem 7 in page 22 of the second article (page 100 of this document).

A by-product of the analysis of fixed PM queries in [DLM14] is the average-case analysis of random PM queries with extreme specified coordinates in relaxed K -d trees. That is a generalization of Theorem 4.6. Remember that given a PM query $\mathbf{q} = (q_0, q_1, \dots, q_{K-1})$, extreme specified coordinates are ones that have $q_i = 0$ or $q_i = 1$. See Theorem 2 in page 4 of the first article (page 68 of this thesis).

Analogously to the case of relaxed K -d trees, a by-product of the analysis of fixed PM queries in [DLM16a] is the average-case analysis of random PM queries with extreme specified coordinates in standard K -d trees. This is a generalization of Theorem 4.1. See Theorem 6 in page 21 of the second article (page 99 of this document).

Note that, given the equivalence between the expected cost of a given rank and a given query (Section 2.3.6) in both cases, relaxed and standard K -d trees, we have that the expected cost $P_{n,\mathbf{q}}$ of a fixed partial match query $\mathbf{q} \in [0, 1]^n$

of n nodes satisfies:

$$P_{n,\mathbf{q}} = \Theta \left(\prod_{i: q_i \text{ is specified}} q_i (1 - q_i)^{\alpha/2} \right) \cdot n^\alpha$$

where α is the exponent of the respective expected cost of random PM queries. In these articles we conjectured that a similar relationship between the cost of random PM queries and the fixed PM queries holds for quadtrees and other variants of K -d trees. We also, based on experiments as mentioned on [Section 3.2](#), conjectured in [\[DLM16a\]](#) that for another variant, the squarish K -d trees, $P_{n,\mathbf{q}}$ satisfies:

$$P_{n,\mathbf{q}} = \Theta(n^{1-s/K}).$$

Note that in these articles we use \tilde{P} or \bar{P} , with subscripts, for the expected cost of random PM queries instead of \hat{P} . In [\[DLM14\]](#) we use t as a subindex while in [\[DLM16a\]](#) we use t for s_0 , the number of specified coordinates that are not extreme. In later articles we avoided those uses of t because in [\[DL17\]](#) we use t for the local rebalance parameter.

5.1 Article: Fixed PM Queries in Relaxed K -d Trees

This article is [\[DLM14\]](#):

Amalia Duch, Gustavo Lau, and Conrado Martínez. “On the Average Performance of Fixed Partial Match Queries in Random Relaxed K -d Trees”. In: *25th International Conference on Probabilistic, Combinatorial and Asymptotic Methods for the Analysis of Algorithms (AofA 2014)*. Ed. by M. Bousquet-Mélou and M. Soria. Discrete Mathematics & Theoretical Computer Science (Proceedings). 2014, pp. 97–108. url: <https://hal.inria.fr/hal-01077251v2/document#page=106>

Attention;

Pages 65 to 76 of the thesis are available at the editor’s web
<https://hal.inria.fr/hal-01077251v2>

5.2 Article: Fixed PM Queries in K -d Trees

This article is [DLM16a]:

Amalia Duch, Gustavo Lau, and Conrado Martínez. “On the Cost of Fixed Partial Match Queries in K -d Trees”. In: *Algorithmica* 75.4 (2016), pp. 684–723. doi: [10.1007/s00453-015-0097-4](https://doi.org/10.1007/s00453-015-0097-4)

Attention!

Pages 78 to 120 of the thesis are available at the editor’s web
<https://link.springer.com/article/10.1007%2Fs00453-015-0097-4>

Chapter 6

K -dt trees

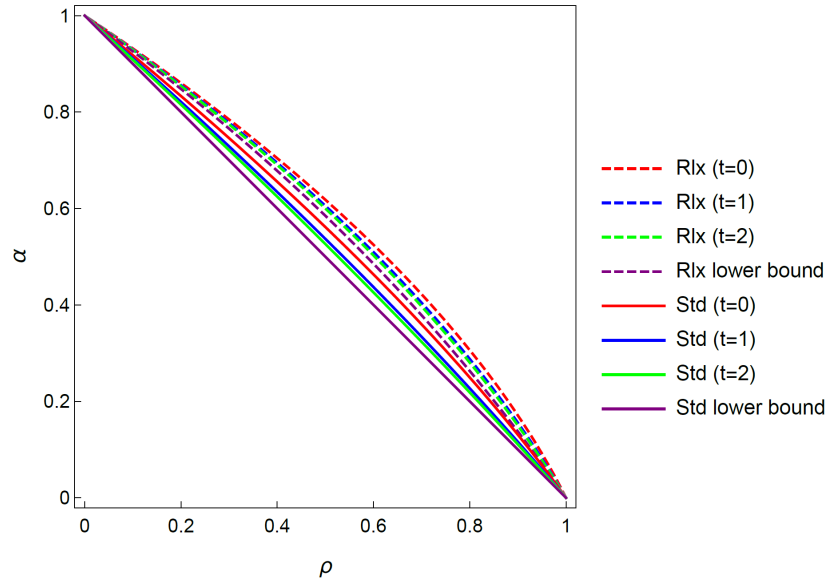
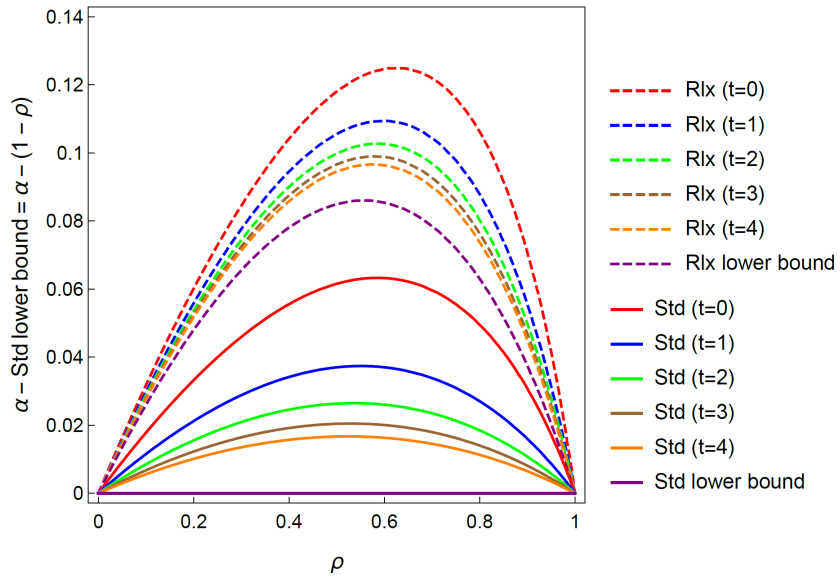
In the article in this chapter [DL17] we introduce a new K -d tree variant, relaxed K -dt trees, adding local rebalances to relaxed K -d trees, and we analyze both random PM queries and fixed PM queries in them. On Theorem 4.1 of this article (page 128 of this document) we give the expected cost of random PM queries in relaxed K -dt trees. This theorem generalizes Theorem 4.6 in two different ways. First, by doing the analysis for any value of t instead of the case of no rebalance ($t = 0$) and second, by analyzing the PM query with s_0 extreme specified coordinates, Theorem 4.6 is the particular case $s_0 = 0$.

In Figure 19 and Figure 20 we show a couple of graphs that we could not include in the article due to lack of space. Figure 19 shows the exponents α of the expected costs of random PM queries for both relaxed and standard K -dt trees for $t = 0$, $t = 1$ and $t = 2$ as a function of $\rho = s/K$. We also show the lower bounds of both variants, relaxed and standard, the latter being $1 - \rho$, which is the diagonal in the graph. For more clarity, in Figure 20 we show the same data, plus the cases $t = 3$ and $t = 4$, but subtracting $1 - \rho$ from α . This graph illustrates the fact that, for both relaxed and standard K -dt trees, the biggest reduction in the exponent α is when going from $t = 0$ to $t = 1$, and that as t increases the improvements become less and less (this was already known for the standard case).

The other main result of [DL17] is to prove that the expected cost of fixed PM queries in random relaxed K -d trees is not of the form conjectured in [DLM16a]:

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

See Proposition 7.1 (page 131 of this thesis).

Figure 19: α vs ρ .Figure 20: $\alpha - (1 - \rho)$ vs ρ .

Regarding random PM queries, we report here some further, unpublished, advances that we have achieved after [DL17] for both relaxed and standard K -dt trees. We denote by $x^{\overline{k}}$ the rising factorial power $x(x+1)\cdots(x+k-1)$ [GKP94].

Theorem 6.1. *The expected cost $\hat{P}_{n,t,\rho,\rho_0}$ of a random PM query—where exactly s , $0 < s < K$, coordinates are specified and exactly s_0 , $0 \leq s_0 \leq s$, of the specified coordinates are extreme—in a random relaxed K -dt tree of size n satisfies*

$$\hat{P}_{n,t,\rho,\rho_0} = \beta_{t,\rho,\rho_0} n^{\alpha_{t,\rho,\rho_0}} + o(n^{\alpha_{t,\rho,\rho_0}}),$$

where both β_{t,ρ,ρ_0} and α_{t,ρ,ρ_0} depend on t and the ratios $\rho = s/K$ and $\rho_0 = s_0/K$. The exponent α_{t,ρ,ρ_0} is the unique solution in the interval $(0, 1)$ of the equation:

$$\rho_0 \frac{(t+2)^{\overline{t+1}}}{2(a+t+1)^{\overline{t+1}}} + (\rho - \rho_0) \frac{(t+2)^{\overline{t+1}}}{(a+t+2)^{\overline{t+1}}} + (1-\rho) \frac{(t+2)^{\overline{t+1}}}{(a+t+1)^{\overline{t+1}}} = 1. \quad (6.1)$$

Theorem 6.1 is a more precise version of Theorem 4.1 of [DL17] because in that theorem we only stated that $\hat{P}_{n,t,\rho,\rho_0} = \Theta(n^\alpha)$. Theorem 5 of [DLM16a] is the particular case $t = 0$ of Theorem 6.1.

Theorem 6.2. *The expected cost $\hat{P}_{n,t,\mathbf{u}}$ of a random PM query with pattern \mathbf{u} —where exactly s , $0 < s < K$, coordinates are specified and exactly s_0 , $0 \leq s_0 \leq s$, of the specified coordinates are extreme—in a random standard K -dt tree of size n satisfies*

$$\hat{P}_{n,t,\mathbf{u}} = \beta_{t,\mathbf{u}} n^{\alpha_{t,\rho,\rho_0}} + o(n^{\alpha_{t,\rho,\rho_0}}),$$

where $\beta_{t,\mathbf{u}}$ depends on t and the query pattern \mathbf{u} and α_{t,ρ,ρ_0} depends on t and the ratios $\rho = s/K$ and $\rho_0 = s_0/K$ due to it being the unique solution in the interval $(0, 1)$ of the equation:

$$\frac{(t+2)^{\overline{t+1}}}{2(a+t+1)^{\overline{t+1}}} \rho^{\rho_0} + \frac{(t+2)^{\overline{t+1}}}{(a+t+2)^{\overline{t+1}}} \rho^{\rho-\rho_0} + \frac{(t+2)^{\overline{t+1}}}{(a+t+1)^{\overline{t+1}}} \rho^{1-\rho} = 1. \quad (6.2)$$

Theorem 4.2 is the particular case $\rho_0 = 0$ of Theorem 6.2 and Theorem 6 of [DLM16a] is the particular case $t = 0$ of Theorem 6.2.

We have written the equations (6.1) and (6.2) in a way that emphasizes their similarity. Both have on the left-hand side weighted means of same three terms, in the case of relaxed K -dt trees is an arithmetic mean and in the case of standard ones is a geometric one.

We provide the proofs of Theorems 6.1 and 6.2 in Appendix B.1.

6.1 Article: PM Queries in Relaxed K -dt trees

This article is [DL17]:

Amalia Duch and Gustavo Lau. “Partial Match Queries in Relaxed K -dt trees”. In: *Proc. of the Fourteenth ACM-SIAM Workshop on Analytic Algorithmics and Combinatorics (ANALCO)*. 2017, pp. 131–138. doi: [10.1137/1.9781611974775.13](https://doi.org/10.1137/1.9781611974775.13)

Attention;

Pages 124 to 132 of the thesis are available at the editor’s web
<https://epubs.siam.org/doi/10.1137/1.9781611974775.13>

Chapter 7

Quad trees

The main result of the article in this chapter [DLM18] is the average-case analysis of the cost of fixed PM queries in quad trees, see [Theorem 3](#) in page 11 of the article in this chapter (page 145 of this thesis). Under the assumption that the limit of $P_{n,r}/n^\alpha$ when $n \rightarrow \infty$ exists, this result generalizes, to any values of K and s , [Theorem 4.9](#) where $K = 2$ and $s = 1$ is assumed.

7.1 Article: Fixed PM Queries in Quadtrees

This article is [DLM18]:


Amalia Duch, Gustavo Lau, and Conrado Martínez. “Fixed Partial Match Queries in Quadtrees”. In: *29th International Conference on Probabilistic, Combinatorial and Asymptotic Methods for the Analysis of Algorithms (AofA 2018)*. Ed. by J. A. Fill and M. D. Ward. Vol. 110. Leibniz International Proceedings in Informatics (LIPIcs). Dagstuhl, Germany: Schloss Dagstuhl–Leibniz-Zentrum fuer Informatik, 2018, 20:1–20:18. ISBN: 978-3-95977-078-1. DOI: [10.4230/LIPIcs.AofA.2018.20](https://doi.org/10.4230/LIPIcs.AofA.2018.20)

Fixed Partial Match Queries in Quadrees

Amalia Duch

Universitat Politècnica de Catalunya


duch@cs.upc.edu

 <https://orcid.org/0000-0003-4371-1286>

Gustavo Lau

Universitat Politècnica de Catalunya


glau@cs.upc.edu

 <https://orcid.org/0000-0002-3460-9186>

Conrado Martínez

Universitat Politècnica de Catalunya

conrado@cs.upc.edu

 <https://orcid.org/0000-0003-1302-9067>

Abstract

Several recent papers in the literature have addressed the analysis of the cost $\mathcal{P}_{n,\mathbf{q}}$ of partial match search for a given fixed query \mathbf{q} —that has s out of K specified coordinates— in different multidimensional data structures. Indeed, detailed asymptotic estimates for the main term in the expected cost $P_{n,\mathbf{q}} = \mathbb{E}\{\mathcal{P}_{n,\mathbf{q}}\}$ in standard and relaxed K -d trees are known (for any dimension K and any number s of specified coordinates), as well as stronger distributional results on $\mathcal{P}_{n,\mathbf{q}}$ for standard 2-d trees and 2-dimensional quadrees. In this work we derive a precise asymptotic estimate for the main order term of $P_{n,\mathbf{q}}$ in quadrees, for any values of K and s , $0 < s < K$, under the assumption that the limit of $P_{n,\mathbf{q}}/n^\alpha$ when $n \rightarrow \infty$ exists, where α is the exponent of n in the expected cost of a *random* partial match query with s specified coordinates in a random K -dimensional quadree.

2012 ACM Subject Classification Theory of computation \rightarrow Data structures design and analysis, Theory of computation \rightarrow Design and analysis of algorithms

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

Digital Object Identifier 10.4230/LIPIcs.AofA.2018.20

Funding This work has been partially supported by funds from the Spanish Ministry of Economy, Industry and Competitiveness (MINECO) and the European Union (FEDER) under grant GRAMM (TIN2017-86727-C2-1-R), and by funds from the Catalan Government (AGAUR) under grant 2017SGR 786.

Acknowledgements We are thankful to the anonymous reviewers of the preliminary version of this paper. Their comments and advice have been very helpful to improve it in many ways, particularly in Subsection 3.3.

1 Introduction

One of the fundamental features of any hierarchical multidimensional data structure such as quadrees is to efficiently support partial match (PM) queries. These queries are as follows. Given a collection F of K -dimensional ($K \geq 2$) tuples of the form $\mathbf{x} = (x_0, \dots, x_{K-1})$, with each x_i ($0 \leq i < K$) belonging to a totally ordered domain \mathcal{D}_i , and a query $\mathbf{q} = (q_0, \dots, q_{K-1})$



© Amalia Duch, Gustavo Lau, and Conrado Martínez;

licensed under Creative Commons License CC-BY

29th International Conference on Probabilistic, Combinatorial and Asymptotic Methods for the Analysis of Algorithms (AofA 2018).

Editors: James Allen Fill and Mark Daniel Ward; Article No. 20; pp. 20:1–20:18



Leibniz International Proceedings in Informatics

LIPICs Schloss Dagstuhl – Leibniz-Zentrum für Informatik, Dagstuhl Publishing, Germany

such that $q_i \in D_i \cup \{*\}$ ($0 \leq i < K$), the goal of a PM query is to find all those tuples in F such that x_i matches q_i whenever $q_i \neq *$. Coordinates such that $q_i \neq *$ are called *specified*, otherwise they are called *unspecified*; we assume that the number s of specified coordinates satisfies $0 < s < K$.

The average-case analysis of PM queries in random quadrees and other multidimensional data structures has a long history. In the case of quadrees, a fundamental milestone was the paper by Flajolet, Gonnet, Puech, and Robson [7] where the authors proved that the expected cost of random PM queries with s specified coordinates in random K -dimensional quadrees of n nodes is $\beta_{s,K} n^{\alpha(s/K)} + l.o.t.$ for some constant $\beta_{s,K}$; and $\alpha = \alpha(s/K)$ the unique real solution in $[0, 1]$ of the indicial equation

$$(\alpha + 2)^s (\alpha + 1)^{K-s} = 2^K. \quad (1)$$

The exponent α turns out to be exactly the same as in the expected cost of random PM queries in standard K -d trees. It was not until 2003 that Chern and Hwang [2] obtained an explicit expression for $\beta_{s,K}$, for general s and K , this is:

$$\beta_{s,K} = \frac{1}{(2^{K-s} - 1)\Gamma(\alpha + 1)^{K-s}\Gamma(\alpha + 2)^s} \prod_{2 \leq j \leq K} \frac{\Gamma(\alpha - \alpha_j)}{\Gamma(-\alpha_j)}, \quad (2)$$

for $0 < s < K$ and $K \geq 2$ and where Γ is the Gamma function and the α_j 's are the roots of equation (1) and $\alpha = \alpha_1 > \Re(\alpha_2) \geq \dots \geq \Re(\alpha_K)$. Note that Chern and Hwang [2] used the indicial equation for $\alpha + 1$ so they gave a formula for $\beta_{s,K}$ as a function of $\alpha'_j = \alpha_j + 1$, $j = 1, \dots, K - 1$.

In 2011 fixed PM queries were studied for the first time in 2-dimensional quadrees by Curien and Joseph [3] where the authors computed the expected cost $\mathbb{E}\{\mathcal{P}_{n,\mathbf{q}}\}$ of a fixed PM query in 2-dimensional quadrees. In particular, they showed that if $\mathbf{q} = (q, *)$, then $P_{n,\mathbf{q}} = \mathbb{E}\{\mathcal{P}_{n,\mathbf{q}}\} \sim \nu_{1,2} \cdot (q \cdot (1 - q))^{\alpha/2} \cdot n^\alpha$, where $\alpha = \alpha(1/2) = (\sqrt{17} - 3)/2$ is the same exponent as in the expected cost for random PM queries [7], and $\nu_{1,2} = \frac{\Gamma(2\alpha+2)\Gamma(\alpha+2)}{2\Gamma^3(\alpha+1)\Gamma^2(\frac{\alpha}{2}+1)}$. The asymptotic distribution was obtained for this particular case by Broutin, Neininger and Sulzbach in 2012 [1].

In this work, we extend the results of [3] to give a precise asymptotic estimate of the expected cost of a fixed PM query in random K -dimensional quadrees, for general K and s . In particular, we show that this cost is of the form

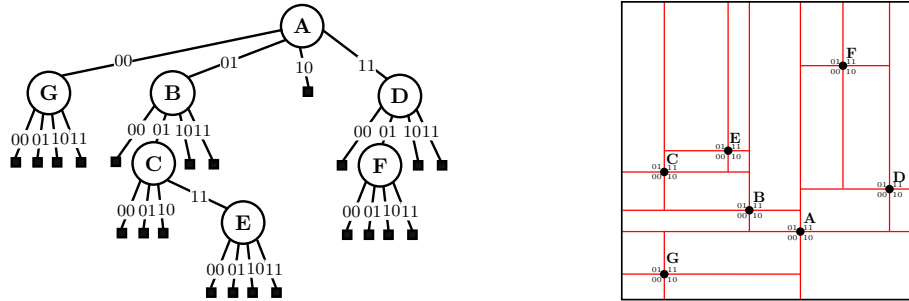
$$\nu_{s,K} \cdot \left(\prod_{i:q_i \neq *} q_i(1 - q_i) \right)^{\alpha/2} \cdot n^\alpha + l.o.t.,$$

where $\nu_{s,K}$ is a constant that depends on s , K and the particular query \mathbf{q} and $\alpha = \alpha(s/K)$ is the same as for random PM queries (see above).

The paper is organised as follows. In Section 2 we give some preliminaries. We explain our methodology in Section 3 through the simplest case $K = 2$ (Subsection 3.1). We continue with the general case of arbitrary s and K (Subsection 3.2). To complete the analysis one needs to solve an integral equation; that is the subject of Subsection 3.3. Section 4 contains some final remarks as well as some future lines of work.

2 Preliminaries

Let F be a collection of n multidimensional records, each one endowed with a K -dimensional key $\mathbf{x} = (x_0, \dots, x_{K-1})$, with coordinate x_j drawn from a totally ordered domain \mathcal{D}_j . For convenience, here we will assume that, for all $0 \leq j < K$, $\mathcal{D}_j = [0, 1]$.



■ **Figure 1** A 2-dimensional quadtree of file $F = \{A, B, C, D, E, F, G\}$ and the partition that it induces of the space. In this example $F_{00} = \{G\}$, $F_{01} = \{B, C, E\}$ and $F_{0*} = \{B, C, E, G\}$.

► **Definition 1.** A quadtree T of size n is a 2^K -ary tree storing a collection F of n K -dimensional records. T is either empty (when $n = 0$) or each one of its n nodes holds a key from F , such that the root node of T stores a record with key \mathbf{x} and pointers to 2^K subtrees, that hold the remaining $n - 1$ records of F . Every subtree of T , let say $T_{\mathbf{w}}$, is associated to a bitstring $\mathbf{w} = w_0w_1 \dots w_{K-1} \in \{0, 1\}^K$, in such a way that $T_{\mathbf{w}}$ is a quadtree, and for any key $y \in T_{\mathbf{w}}$, it holds that $y_j \leq x_j$ if $w_j = 0$ and $y_j > x_j$ if $w_j = 1$, for all $0 \leq j < K$.

Any quadtree of size n induces a partition of the domain into $(2^K - 1)n + 1$ regions, each corresponding to a leaf (or equivalently empty subtree) in the quadtree. An example of a quadtree and the partition of the space that it induces is shown in Figure 1. To build a quadtree starting from an empty tree, each insertion of a new record with key \mathbf{x} follows a path from the root to a leaf; at each step, we compare \mathbf{x} and the key at the current node to determine in which of the 2^K subtrees the insertion should continue recursively, and the process ends when a leaf is reached and it is replaced by a new node containing \mathbf{x} and 2^K empty subtrees. The region associated to the substituted leaf is called the *bounding box* of the subtree rooted at \mathbf{x} . Following the same convention used for the names of the subtrees, we will denote by $B_{\mathbf{w}}$ the bounding boxes of subtrees $T_{\mathbf{w}}$ associated to the tree rooted at \mathbf{x} and by $F_{\mathbf{w}}$ the subset of data points of F that fall inside $B_{\mathbf{w}}$.

Consider a string \mathbf{v} over the alphabet $\Sigma = \{0, 1, *\}$. We define as $\mathcal{L}(\mathbf{v})$ the set of binary strings matching \mathbf{v} ; that is, where each occurrence of the symbol $*$ stands for a 0 or a 1. For instance, $\mathcal{L}(001) = \{001\}$, $\mathcal{L}(0*1) = \{001, 011\}$ and $\mathcal{L}(1**00) = \{10000, 10100, 11000, 11100\}$. With this notation let us define the following extension of the notion of bounding box $B_{\mathbf{v}} = \bigcup_{\mathbf{w} \in \mathcal{L}(\mathbf{v})} B_{\mathbf{w}}$.

Likewise $F_{\mathbf{v}}$ is the union of the (disjoint) $F_{\mathbf{w}}$'s with \mathbf{w} matching \mathbf{v} . For example, in two dimensions $B_{**} = [0, 1]^2$ is the bounding box of the root of the quadtree, F_{0*} is the subset of all those keys with first coordinate smaller than the first coordinate of the root, that is, the ones stored in T_{00} and T_{01} (see Figure 1).

To perform a PM search with query \mathbf{q} , the quadtree is recursively explored as follows. First, we check whether the root \mathbf{x} matches \mathbf{q} or not, to report it in the former case. Then, we make recursive calls in all the 2^{K-s} subtrees $T_{\mathbf{w}}$ such that the first s bits of \mathbf{w} are such that $w_i = 0$ whenever $q_i \neq *$ and $q_i \leq x_i$, and $w_i = 1$ whenever $q_i \neq *$ and $q_i > x_i$, $0 \leq i < s$, and the remaining $K - s$ bits can be either 0 or 1.

One key observation about the PM search in quadtrees (or similar data structures) is that, except for eventual matches, only the relative ranks of the coordinates matter. Let

we call the *rank vector* of a query \mathbf{q} the vector $\mathbf{r}(\mathbf{q}) = (r_0, \dots, r_{K-1})$ such that $r_i = *$, if $q_i = *$, and r_i is the number of records \mathbf{x} in the collection F such that $x_i \leq q_i$ ($0 \leq r_i \leq n$), if $q_i \neq *$. Then for any two given queries \mathbf{q} and \mathbf{q}' with equal rank vectors $\mathbf{r}(\mathbf{q}) = \mathbf{r}(\mathbf{q}')$ the PM procedure described above will visit exactly the same set of nodes of the tree. In our analysis, we shall be using rank vectors instead of the queries themselves (as done in [6]) and consider, for instance, the cost $\mathcal{P}_{n,\mathbf{r}}$ of a PM query with given rank vector \mathbf{r} in a random quadtree of size n . The probability model for random quadtrees that we will use throughout this work is that the tree is built by inserting in any order n keys drawn independently at random (coordinate by coordinate) from a continuous distribution. For the sake of simplicity, we can safely assume that the distribution is Uniform(0, 1). Because of the symmetry of the model we can also assume that the s specified coordinates of \mathbf{q} are the first s coordinates, $0 < s < K$, and therefore that $\mathbf{q} = (q_0, \dots, q_{s-1}, *, \dots, *)$ and $\mathbf{r} = (r_0, \dots, r_{s-1}, *, \dots, *)$. We shall write hence $\mathbf{q} = (q_0, \dots, q_{s-1})$ and $\mathbf{r} = (r_0, r_1, \dots, r_{s-1})$ with the convention that the implicit $K - s$ remaining components are all $*$'s.

3 Analysis

Our goal in this section is to find the expected cost $P_{n,\mathbf{r}} = \mathbb{E}\{\mathcal{P}_{n,\mathbf{r}}\}$, measured as the number of visited nodes, of a PM query with a fixed rank vector \mathbf{r} in a random quadtree of n nodes.

In order to show our methodology and to give some intuition on the problem we are going to start our analysis with the easiest case $K = 2$ in Subsection 3.1. Afterwards, in Subsection 3.2, we analyze the general case.

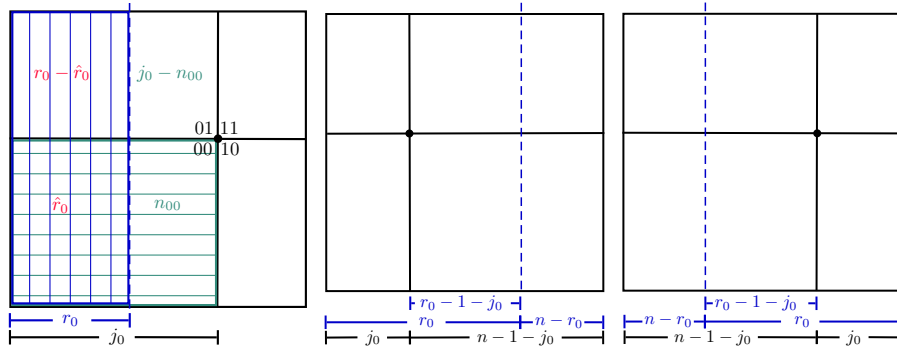
In both subsections we are going to obtain a recurrence for $P_{n,\mathbf{r}}$. Then, in order to solve the general recurrence, we translate it into an integral equation whose solution will give us the leading term in the asymptotic estimate for $P_{n,\mathbf{r}}$. The solution of the integral equation is given in Subsection 3.3.

3.1 The case $K = 2$

Given a 2-dimensional quadtree T , its root splits the space into four rectangles: B_{00} (south-west of the root), B_{01} (north-west of the root), B_{10} (south-east of the root) and B_{11} (north-east of the root). These four rectangles are the corresponding *bounding boxes* of the four subtrees T_{00} , T_{01} , T_{10} and T_{11} from Definition 1. Recall also that $B_{0*} = B_{00} \cup B_{01}$ and $B_{*0} = B_{00} \cup B_{10}$ are, respectively, the rectangles west and south of the root. For any string $\mathbf{u} \in \{0, 1, *\}^2$, the number of data points in $B_{\mathbf{u}}$ (equivalently, the cardinality of $F_{\mathbf{u}}$) will be denoted $\mathcal{N}_{\mathbf{u}}$. For a random quadtree the $\mathcal{N}_{\mathbf{u}}$'s are random variables.

Let us now address the recurrence for $P_{n,\mathbf{r}}$, and to simplify let us write P_{n,r_0} , as $\mathbf{r} = (r_0, *)$. The basis of recursion is trivially $P_{0,r_0} = 0$. If $n > 0$, let $\mathbf{j} = (j_0, j_1)$ be the rank vector of the root. Since \mathbf{q} contains only one specified coordinate, the relation between j_0 and r_0 determines whether the query intersects either B_{0*} or B_{1*} . If $r_0 \leq j_0$, then the query intersects B_{0*} ; otherwise it intersects B_{1*} . In our recurrence for P_{n,r_0} the value $j_0 = \mathcal{N}_{0*} = |F_{0*}|$ run from r_0 to $n - 1$, leading to a non-empty intersection of B_{0*} and the query, or from 0 to $r_0 - 1$, leading to a non-empty intersection of B_{1*} and the query. Because of the randomness assumptions, each possible value of \mathcal{N}_{0*} has probability $1/n$ and hence this factor will weight the expected cost of the PM query conditioned to $\mathcal{N}_{0*} = j_0$.

The number of data points in B_{0*} is j_0 by definition, and the number of data points in B_{1*} is $n - 1 - j_0$. If the query intersects B_{0*} then the rank of the query with respect to B_{0*} is still r_0 , but if it intersects B_{1*} then its rank with respect to B_{1*} is $r_0 - 1 - j_0$. So the contribution to P_{n,r_0} coming from the recursive traversal of B_{0*} involves a set of j_0



■ **Figure 2** A partial match in a two-dimensional quadtree. The first diagram shows the case $r_0 \leq j_0$, the second one the case $j_0 < r_0$ and the third one how the east-west symmetry converts the second case into the first one.

points and the rank of the query is r_0 while the contribution coming from B_{1*} involves a set $n - 1 - j_0$ points and, because of the symmetry $P_{n,r_0} = P_{n,n-r_0}$, the rank of the query is $n - r_0$. Hence, we can reduce the case $j_0 < r_0$ to the case $r_0 \leq j_0$, see Figure 2.

In the general case we would have to consider 2^s regions $B_{\mathbf{w}}$ described by bitstrings $\mathbf{w} = w_0 \cdots w_{s-1} * \cdots *$, where each w_i is 0 or 1 depending on whether $r_i \leq j_i$ or not; as we consider all possible \mathbf{j} , the query will intersect these 2^s different regions, and we will be able to use these “east-west” symmetry considerations to reduce their analysis to the analysis of one of them, say, $B_{00\dots 0* \dots *}$.

Let us come back to $K = 2$. The region B_{0*} is the union of the two bounding boxes B_{00} and B_{01} (in general we will consider regions $B_{\mathbf{w}}$ that contain 2^{K-s} bounding boxes) and our goal is to use further symmetries to reduce the analysis of the cost of traversing both bounding boxes to the analysis of just traversing one of them, say, B_{00} .

Let Q_{j_0,r_0} be the contribution to the expected cost of a PM query due to the recursive call in T_{00} , when the query has rank r_0 in the first coordinate and given that there are $j_0 \geq r_0$ nodes to the west of the root.

Suppose that $\mathcal{N}_{00} = n_{00}$. The rank vector of the query in the recursive call to T_{00} will be $(\hat{r}_0, *)$, and the contribution to the expected cost will then be P_{n_{00},\hat{r}_0} . So it only remains to determine: a) the probability that $\mathcal{N}_{00} = n_{00}$, given the rank vector of the root \mathbf{j} and, b) the probability that the rank vector of the query with respect to B_{00} is $(\hat{r}_0, *)$. Let us define the subsets of data points $F'_{\mathbf{v}}$ and the corresponding bounding boxes $B'_{\mathbf{v}}$ like $F_{\mathbf{v}}$ and $B_{\mathbf{v}}$, but with respect to the given query, instead of the root. The value \hat{r}_0 is the number of data points in the intersection between B_{00} and B'_{0*} , see Figure 2. We will use $\mathcal{R}_{\langle \mathbf{0} \rangle} := |F_{00} \cap F'_{0*}|$.

In general, $\langle \mathbf{i} \rangle := *^i 0 *^{K-1-i}$, so using this convention, we can also write $\mathcal{N}_{\langle \mathbf{0} \rangle} = j_0$ and $|F'_{\langle \mathbf{0} \rangle}| = r_0$. Conditioned on the sizes of F_{00} , $F_{\langle \mathbf{0} \rangle}$ and $F'_{\langle \mathbf{0} \rangle}$, the random variable $\mathcal{R}_{\langle \mathbf{0} \rangle}$ obeys a hypergeometric distribution:

$$\Pr \left\{ \mathcal{R}_{\langle \mathbf{0} \rangle} = \hat{r}_0 \mid \mathcal{N}_{00} = n_{00}, \mathcal{N}_{\langle \mathbf{0} \rangle} = j_0, |F'_{\langle \mathbf{0} \rangle}| = r_0 \right\} = \frac{\binom{n_{00}}{\hat{r}_0} \binom{j_0 - n_{00}}{r_0 - \hat{r}_0}}{\binom{j_0}{r_0}}.$$

Now if we look at the contribution to the expected cost due to the traversal of T_{01} , we have that $\mathcal{N}_{01} = j_0 - n_{00}$ and the rank of the query with respect to B_{01} is $(r_0 - \hat{r}_0, *)$. The fact that the second coordinate is unspecified allow us to do the analysis above with n_{01} instead of n_{00} and we would have obtained symmetric formulas. We can exploit this

20:6 Fixed Partial Match Queries in Quadrees

north-south symmetry that will give us a factor of 2. Taking into account the visit to the root and our discussion so far we can write

$$P_{n,r_0} = 1 + \frac{2}{n} \left(\sum_{j_0=0}^{r_0-1} Q_{n-1-j_0, n-r_0} + \sum_{j_0=r_0}^{n-1} Q_{j_0, r_0} \right), \quad (3)$$

where, for $n_{0*} \geq r$, we have

$$Q_{j_0, r_0} = \sum_{n_{00}=0}^{j_0} \Pr \{ \mathcal{N}_{00} = n_{00} \mid \mathcal{N}_{\langle 0 \rangle} = j_0 \} \sum_{\hat{r}_0=0}^{r_0} \left(\frac{\binom{n_{00}}{\hat{r}_0} \binom{j_0-n_{00}}{r_0-\hat{r}_0}}{\binom{j_0}{r_0}} P_{n_{00}, \hat{r}_0} \right). \quad (4)$$

To complete the recurrence for P_{n,r_0} we need only to obtain the probability that $\mathcal{N}_{00} = n_{00}$, conditioned on $\mathcal{N}_{\langle 0 \rangle} = j_0$. Since $\mathcal{N}_{\langle 1 \rangle}$ can take any value in $[0..n-1]$ with identical probability, the number of points in B_{00} will take any value between 0 and j_0 with identical probability $1/(j_0+1)$. Plugging this probability and (4) into (3) yields to the desired recurrence for P_{n,r_0} .

An asymptotic estimate of the main term of P_{n,r_0} follows by deriving an integral equation for $f(z_0) := \lim_{n \rightarrow \infty} P_{n, z_0 n} / n^\alpha$ and solving that integral equation. We give the details of the derivation of the integral equation in the case of $K=2$ in Lemma 4.

3.2 The general case

Let $\mathbf{r} = (r_0, r_1, \dots, r_{s-1})$ be the query rank vector and let $\mathbf{j} = (j_0, \dots, j_{s-1})$ be the first s coordinates of the rank vector for the root of the random quadtree. Thus we have that j_i is the value of $|F_{\langle i \rangle}| = \mathcal{N}_{\langle i \rangle}$. These K strings of the form $\langle \mathbf{i} \rangle$ constitute a “basis” in the sense that we can obtain any region $B_{\mathbf{w}}$ by complementation ($B_{*^i 1 *^{K-i}} = B_{* \dots *} \setminus B_{\langle \mathbf{i} \rangle}$) and intersection of the appropriate $B_{\langle \mathbf{i} \rangle}$ ’s.

Like we did for $K=2$ our goal is to use the symmetries of the problem to reduce the whole analysis to the analysis of the contribution to the total cost of one particular subtree, namely, T_{0^s} . Again, call $Q_{\mathbf{j}, \mathbf{r}}$ the contribution of the recursive call in T_{0^s} , conditioned to $r_i \leq j_i$ for all i , $0 \leq i < s$. This condition guarantees that the PM search will recursively continue in that subtree.

Then, because of the $K-s$ symmetries on unspecified coordinates (like the north-south symmetry of the case $K=2$) and because of the s symmetries for specified coordinates (like the east-west symmetry when $K=2$), we can express $P_{n, \mathbf{r}}$ in terms of $Q_{\mathbf{j}, \mathbf{r}}$ ’s. In particular, considering all the possibilities for \mathbf{j} gives a factor $1/n^s$, and a summation over all bitstrings \mathbf{w} of length s to cover the cases where the query intersects $B_{\mathbf{w}}$. Finally the factor 2^{K-s} stems from the 2^{K-s} bounding boxes that each $B_{\mathbf{w}}$ contains. Hence,

$$P_{n, \mathbf{r}} = 1 + \frac{2^{K-s}}{n^s} \sum_{\mathbf{w} \in \{0,1\}^s} \sum_{j_0} \cdots \sum_{j_{s-1}} Q_{\mathbf{j}'_{\mathbf{w}}(\mathbf{j}), \mathbf{r}'_{\mathbf{w}}(\mathbf{r})}, \quad (5)$$

where the summation ranges are $r_i \leq j_i \leq n-1$ if $w_i = 0$, and $0 \leq j_i \leq r_i - 1$ if $w_i = 1$, and the rank vectors $\mathbf{j}'_{\mathbf{w}} = (j'_0, \dots, j'_{s-1})$ and $\mathbf{r}'_{\mathbf{w}} = (r'_0, \dots, r'_{s-1})$ are defined as follows: if $w_i = 0$ then $j'_i = j_i$ and $r'_i = r_i$, otherwise if $w_i = 1$ then $j'_i = n-1-j_i$ and $r'_i = n-r_i$.

For any i , $0 \leq i < K$, we will denote $\mathbf{0}^i$ the string $0^i *^{K-i}$, that is, a string of length K consisting of i zeros, followed by $K-i$ *’s.

The method to obtain a formula for $Q_{\mathbf{j}, \mathbf{r}}$ consists of the following steps: 1) First we use Lemma 5 to obtain the probability distribution of the number of data points $\mathcal{N}_{\mathbf{0}^s}$ in the

“corner” hyperrectangle, by intersecting the sets $F_{(0)}, F_{(1)}, \dots, F_{(s-1)}$, with sizes j_0, \dots, j_{s-1} , respectively. This will be expressed by $s - 1$ “hypergeometric” sums that will give us the probability that $\mathcal{N}_{0^s} = \ell_s$; 2) Given that the last $K - s$ coordinates are unspecified, and conditioned on $j_i = \mathcal{N}_{(i)}$, $0 \leq i < s$, all the potential sizes of $\mathcal{N}_{(i)} = |F_{(i)}|$, $s \leq i < K$, are equiprobable. This will be expressed by $K - s$ “uniform” sums that will allow us to derive the probability distribution for \mathcal{N}_{0^K} , and 3) Now conditioning on $\mathcal{N}_{0^K} = |F_{0^K}|$, and given \mathbf{r} we intersect F_{0^K} with each of $F'_{(0)}, F'_{(1)}, \dots, F'_{(s-1)}$ to obtain the components of $\mathbf{r}_{0^K} = (\hat{r}_0, \dots, \hat{r}_{s-1})$. We will denote $\mathcal{R}_{(i)} = |F_{0^K} \cap F'_{(i)}|$ the random variable that gives the i -th component of \mathbf{r}_{0^K} . As in the case $K = 2$, the probability distribution of the $\mathcal{R}_{(i)}$'s is hypergeometric and it will lead to s additional “hypergeometric” sums.

Therefore the general formula for $Q_{\mathbf{j}, \mathbf{r}}$ is:

$$Q_{\mathbf{j}, \mathbf{r}} = \sum_{\ell_s=0}^{j_{s-1}} \Pr \left\{ \mathcal{N}_{0^s} = \ell_s \left| \bigwedge_{i=0}^{s-1} \mathcal{N}_{(i)} = j_i \right. \right\} \times \sum_{\ell_K=0}^{\ell_s} \Pr \left\{ \mathcal{N}_{0^K} = \ell_K \left| \mathcal{N}_{0^s} = \ell_s \right. \right\} \\ \times \sum_{\mathbf{r}_{0^K}=(\hat{r}_0, \dots, \hat{r}_{s-1})} \Pr \left\{ \bigwedge_{i=0}^{s-1} \mathcal{R}_{(i)} = \hat{r}_i \left| \mathcal{N}_{0^K} = \ell_K, \bigwedge_{i=0}^{s-1} |F'_{(i)}| = r_i \right. \right\} \times P_{\ell_K, \mathbf{r}_{0^K}}. \quad (6)$$

We can expand this last expression as:

$$Q_{\mathbf{j}, \mathbf{r}} = \sum_{\ell_s=0}^{j_{s-1}} \dots \sum_{\ell_2=0}^{j_1} \left(\frac{\binom{j_0}{\ell_2} \binom{n-1-j_0}{j_1-\ell_2}}{\binom{n-1}{j_1}} \dots \frac{\binom{\ell_{s-1}}{\ell_s} \binom{n-1-\ell_{s-1}}{j_{s-1}-\ell_s}}{\binom{n-1}{j_{s-1}}} \right) \\ \times \frac{1}{\ell_s + 1} \sum_{\ell_{s+1}=0}^{\ell_s} \dots \frac{1}{\ell_{K-1} + 1} \sum_{\ell_K=0}^{\ell_{K-1}} \\ \sum_{\hat{r}_0=0}^{\ell_K \wedge r_0} \frac{\binom{\ell_K}{\hat{r}_0} \binom{j_0-\ell_K}{r_0-\hat{r}_0}}{\binom{j_0}{r_0}} \dots \sum_{\hat{r}_{s-1}=0}^{\ell_K \wedge r_{s-1}} \frac{\binom{\ell_K}{\hat{r}_{s-1}} \binom{j_1-\ell_K}{r_{s-1}-\hat{r}_{s-1}}}{\binom{j_{s-1}}{r_{s-1}}} P_{\ell_K, (\hat{r}_0, \dots, \hat{r}_{s-1})}, \quad (7)$$

where we have used $x \wedge y = \min(x, y)$ to stress the intersections that are involved in each case, e.g. \hat{r}_i ranges from 0 to $\ell_K \wedge r_i$ since the number of data points is given by $|F_{0^K} \cap F'_{(i)}|$; with $|F_{0^K}| = \mathcal{N}_{0^K} = \ell_K$ and $|F'_{(i)}| = r_i$.

To derive the integral equation corresponding to the recurrence above we can use arguments similar to those in the case $K = 2$. We give all the details of this derivation, as well as other necessary technical lemmas in Appendix A.

► **Lemma 2.** *If $f(z_0, \dots, z_{s-1}) = \lim_{n \rightarrow \infty} \frac{P_{n, \mathbf{r}}}{n^\alpha}$ exists, with $\alpha = \alpha(s/K)$ the solution of the indicial equation (1) and $z_i = \lim_{n \rightarrow \infty} r_i/n$, $0 < z_i < 1$, for all i , $0 \leq i < s$, then $f(z_0, \dots, z_{s-1})$ is the unique solution of*

$$f(z_0, \dots, z_{s-1}) = \left(\frac{2}{\alpha + 1} \right)^{K-s} \times \sum_{\mathbf{w} \in (0+1)^s} \left\{ \int_{I_{w_0}(z_0)} \dots \int_{I_{w_{s-1}}(z_{s-1})} f \left(\varphi_{w_0}(z_0, u_0), \dots, \varphi_{w_{s-1}}(z_{s-1}, u_{s-1}) \right) \right. \\ \left. \cdot \left(\psi_{w_0}(u_0) \dots \psi_{w_{s-1}}(u_{s-1}) \right)^\alpha du_{s-1} \dots du_0 \right\}, \quad (8)$$

where $I_0(z) = [0, z]$, $I_1(z) = [z, 1]$, $\psi_0(u) = 1 - u$, $\psi_1(u) = u$, $\varphi_0(z, u) = (1 - z)/(1 - u)$ and $\varphi_1(z, u) = z/u$, which satisfies the following boundary conditions:

20:8 Fixed Partial Match Queries in Quadrees

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

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

2. For any $z_i \in (0, 1)$, $0 \leq i < s$, f is symmetric with respect to the axis $z_i = 1/2$, that is,

$$f(z_0, \dots, z_i, \dots, z_{s-1}) = f(z_0, \dots, 1 - z_i, \dots, z_{s-1}).$$

3. For any i , $0 \leq i < s$,

$$\lim_{z_i \rightarrow 0^+} f(z_0, \dots, z_i, \dots, z_{s-1}) = \lim_{z_i \rightarrow 1^-} f(z_0, \dots, z_i, \dots, z_{s-1}) = 0.$$

- 4.

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

Proof. We will follow a procedure similar to the one in the proof of Lemma 4, which covers the case $K = 2$.

The steps that we will give to obtain the integral equation for general K are:

1. Apply Lemma 6 to (7) s times in the s hypergeometric sums (the last sums over the \hat{r}_i 's)
2. Convert the $K - s$ uniform sums (the middle sums over the ℓ_i 's, $s < i \leq K$) into the corresponding integral by passing to the limit. That gives $K - s$ factors $1/(\alpha + 1)$.
3. Apply Lemma 7 once to the first $s - 1$ hypergeometric sums (over the ℓ_i 's, $2 \leq i \leq s$).
4. Convert all the sums in (5) into integrals by passing to the limit.

Here, we use ℓ_i to denote the values that the random variables \mathcal{N}_{0^i} can take, like we did in subsection 3.2, and in particular in (6) and successive.

Defining $f\left(\frac{r_0}{n}, \dots, \frac{r_{s-1}}{n}\right) := P_{n,r}/n^\alpha$, where α is the solution of the indicial equation for quadrees, we get:

$$\begin{aligned} \frac{Q_{\mathbf{j},\mathbf{r}}}{n^\alpha} &= \sum_{\ell_s=0}^{j_{s-1}} \cdots \sum_{\ell_2=0}^{j_1} \left(\frac{\binom{j_0}{\ell_2} \binom{n-1-j_0}{j_1-\ell_2}}{\binom{n-1}{j_1}} \cdots \frac{\binom{\ell_{s-1}}{\ell_s} \binom{n-1-\ell_{s-1}}{j_{s-1}-\ell_s}}{\binom{n-1}{j_{s-1}}} \right) \\ &\quad \times \frac{1}{\ell_s + 1} \sum_{\ell_{s+1}=0}^{\ell_s} \cdots \frac{1}{\ell_{K-1} + 1} \sum_{\ell_K=0}^{\ell_{K-1}} \\ &\quad \sum_{\hat{r}_0=0}^{\ell_K \wedge r_0} \frac{\binom{\ell_K}{\hat{r}_0} \binom{j_0-\ell_K}{r_0-\hat{r}_0}}{\binom{j_0}{r_0}} \cdots \sum_{\hat{r}_{s-1}=0}^{\ell_K \wedge r_{s-1}} \frac{\binom{\ell_K}{\hat{r}_{s-1}} \binom{j_1-\ell_K}{r_{s-1}-\hat{r}_{s-1}}}{\binom{j_{s-1}}{r_{s-1}}} \times f\left(\frac{\hat{r}_0}{\ell_K}, \dots, \frac{\hat{r}_{s-1}}{\ell_K}\right) \left(\frac{\ell_K}{n}\right)^\alpha. \end{aligned}$$

Hence, defining $u_{0^i} = \lim_{n \rightarrow \infty} (\ell_i/n)$ for $s \leq i \leq K$, $z_i = \lim_{n \rightarrow \infty} (r_i/n)$ and $u_i =$

$\lim_{n \rightarrow \infty} (j_i/n)$ for $0 \leq i < K$ and applying Lemma 6 s times:

$$\begin{aligned} \lim_{n \rightarrow \infty} \frac{Q_{\mathbf{j}, \mathbf{r}}}{n^\alpha} &= \lim_{n \rightarrow \infty} \sum_{\ell_s=0}^{j_{s-1}} \cdots \sum_{\ell_2=0}^{j_1} \left(\frac{\binom{j_0}{\ell_2} \binom{n-1-j_0}{j_1-\ell_2}}{\binom{n-1}{j_1}} \cdots \frac{\binom{\ell_{s-1}}{\ell_s} \binom{n-1-\ell_{s-1}}{j_{s-1}-\ell_s}}{\binom{n-1}{j_{s-1}}} \right) \\ &\quad \times \frac{1}{\ell_s+1} \sum_{\ell_{s+1}=0}^{\ell_s} \cdots \frac{1}{\ell_{K-1}+1} \sum_{\ell_K=0}^{\ell_{K-1}} f\left(\frac{r_0}{j_0}, \dots, \frac{r_{s-1}}{j_{s-1}}\right) \left(\frac{\ell_K}{n}\right)^\alpha \\ &= \lim_{n \rightarrow \infty} \sum_{\ell_s=0}^{j_{s-1}} \cdots \sum_{\ell_2=0}^{j_1} \left(\frac{\binom{j_0}{\ell_2} \binom{n-1-j_0}{j_1-\ell_2}}{\binom{n-1}{j_1}} \cdots \frac{\binom{\ell_{s-1}}{\ell_s} \binom{n-1-\ell_{s-1}}{j_{s-1}-\ell_s}}{\binom{n-1}{j_{s-1}}} \right) \\ &\quad \times \frac{1}{u_0^s} \int_0^{u_0^s} \cdots \frac{1}{u_0^{K-i}} \int_0^{u_0^{K-i}} f\left(\frac{z_0}{u_0}, \dots, \frac{z_{s-1}}{u_{s-1}}\right) u_0^\alpha du_0^K \dots du_0^{s+1} \\ &= \lim_{n \rightarrow \infty} \sum_{\ell_s=0}^{j_{s-1}} \cdots \sum_{\ell_2=0}^{j_1} \left(\frac{\binom{j_0}{\ell_2} \binom{n-1-j_0}{j_1-\ell_2}}{\binom{n-1}{j_1}} \cdots \frac{\binom{\ell_{s-1}}{\ell_s} \binom{n-1-\ell_{s-1}}{j_{s-1}-\ell_s}}{\binom{n-1}{j_{s-1}}} \right) \\ &\quad \times f\left(\frac{z_0}{u_0}, \dots, \frac{z_{s-1}}{u_{s-1}}\right) \frac{u_0^\alpha}{(\alpha+1)^{K-s}}. \end{aligned}$$

Replacing u_0^s by ℓ_s/n and applying Lemma 7 once to the first $s-1$ hypergeometric sums we obtain:

$$\lim_{n \rightarrow \infty} \frac{Q_{\mathbf{j}, \mathbf{r}}}{n^\alpha} = \frac{1}{(\alpha+1)^{K-s}} f\left(\frac{z_0}{u_0}, \dots, \frac{z_{s-1}}{u_{s-1}}\right) \prod_{i=0}^{s-1} u_i^\alpha. \tag{9}$$

Finally, introduce the following notation: $I_0(z) = [0, z]$, $I_1(z) = [z, 1]$, $\varphi_0(z, u) = (1-z)/(1-u)$ and $\varphi_1(z, u) = z/u$. Plugging (9) into (5) and passing to the limit (the fourth step in the procedure that we have described) yields the stated integral equation. ◀

Conditions 1 and 2 in the lemma follow from the combinatorics of the problem. By symmetry, $P_{n, \mathbf{r}} = P_{n, \mathbf{r}'}$ for any permutation \mathbf{r}' of the rank vector \mathbf{r} . Likewise, if $\mathbf{r} = (r_0, \dots, r_i, \dots, r_{s-1})$ and $\mathbf{r}' = (r_0, \dots, r_{i-1}, n-r_i, r_{i+1}, \dots, r_{s-1})$ then $P_{n, \mathbf{r}} = P_{n, \mathbf{r}'}$. Condition 3 needs an inductive argument in the number of non-extreme ($z_i \neq 0$ and $z_i \neq 1$) coordinates. When all specified coordinates are extreme, say, $z_0 = z_1 = \dots = z_{s-1} = 0$ we must have $f = 0$; indeed, it is very easy to prove that $P_{n, (0, \dots, 0)} = o(n^\alpha)$. We do not give here a complete and detailed analysis when $s_0 \leq s$ specified coordinates are extreme; the computations and the reasoning is analogous to that carried out in [6] for K -d trees. Last but not least, Condition 4 follows by summing the expected cost $P_{n, \mathbf{r}}$ over all possible rank vectors \mathbf{r} and dividing by $(n+1)^s$: it must yield the known expected cost of a random partial match query $\beta_{s, K} n^\alpha + o(n^\alpha)$. In terms of f , we must integrate f in the domain $[0, 1]^s$ to obtain $\beta_{s, K}$. For a detailed justification the reader can refer to [6]: it is straightforward to adapt the discussion there to the case of quadtrees.

3.3 Solving the integral equation

From the integral equation (8) in Lemma 2 we can obtain an equivalent partial differential equation (PDE) by application of the differential operators

$$\Phi_j(f) = z_j(1-z_j) \frac{\partial^2 f}{\partial z_j^2} + \alpha(2z_j-1) \frac{\partial f}{\partial z_j} - \alpha(\alpha+1)f.$$

20:10 Fixed Partial Match Queries in Quadrees

Indeed, if we define the operator

$$I_i(f) = z_i^{\alpha+1} \int_{z_i}^1 f(z_0, \dots, z_{i-1}, u_i, z_{i+1}, \dots, z_{s-1}) \frac{du_i}{u_i^{\alpha+2}} + (1-z_i)^{\alpha+1} \int_0^{z_i} f(z_0, \dots, z_{i-1}, v_i, z_{i+1}, \dots, z_{s-1}) \frac{dv_i}{(1-v_i)^{\alpha+2}}$$

then the integral equation (8) in Lemma 2 can be written as

$$f = \left(\frac{2}{\alpha+1} \right)^{K-s} I_0(I_1(\dots(I_{s-1}(f)\dots)),$$

using the changes of variables $u_i := z_i/u_i$ and $v_i := (1-z_i)/(1-u_i)$.

Then, as

$$\Phi_i(I_j(g)) = \Psi_i(g) = (2z_i - 1) \frac{\partial g}{\partial z_i} - 2\alpha g$$

it follows that

$$\Phi_0(\Phi_1(\dots(\Phi_{s-1}(f))\dots)) = \left(\frac{2}{\alpha+1} \right)^{K-s} \Phi_0(\Phi_1(\dots(\Phi_{s-1}(I_0(I_1(\dots(I_{s-1}(f)\dots))\dots))\dots)).$$

Now, since Φ_i 's and Ψ_i 's commute – $\Phi_i(\Phi_j(g)) = \Phi_j(\Phi_i(g))$, $\Psi_i(\Psi_j(g)) = \Psi_j(\Psi_i(g))$ – and $\Phi_i(\Psi_j(g)) = \Psi_j(\Phi_i(g))$ for any $i \neq j$, we can manipulate the equation above to get

$$\Phi_0(\Phi_1(\dots(\Phi_{s-1}(f))\dots)) = \left(\frac{2}{\alpha+1} \right)^{K-s} \Psi_0(\Psi_1(\dots(\Psi_{s-1}(f)\dots))$$

or

$$\left(\Phi_0 \circ \Phi_1 \circ \dots \circ \Phi_{s-1} - \left(\frac{2}{\alpha+1} \right)^{K-s} \Psi_0 \circ \Psi_1 \circ \dots \circ \Psi_{s-1} \right)(f) = 0, \quad (10)$$

which is the sought PDE, succinctly expressed in terms of the linear differential operators Φ_i and Ψ_i , $i = 0, \dots, s-1$.

The resulting PDE is homogeneous and linear, hence it is natural to try to solve it by separation of variables. The shape of equation (10) also cries out to try a solution in separated variables. Therefore, we will assume that the solution to the integral equation (8) is a function: $f(z_0, z_1, \dots, z_{s-1}) = \phi_0(z_0) \cdot \phi_1(z_1) \cdot \dots \cdot \phi_{s-1}(z_{s-1})$.

Given that the function f is symmetric with respect to any permutation of its arguments, we can also safely assume that all the functions $\phi_0, \phi_1, \dots, \phi_{s-1}$ are the same function ϕ . Rather than working with the PDE itself, we may use our assumption to rewrite equation (8) as:

$$\phi(z_0) \cdot \phi(z_1) \cdot \dots \cdot \phi(z_{s-1}) = \left(\frac{2}{\alpha+1} \right)^{K-s} \prod_{i=0}^{s-1} \left(\int_0^{z_i} \phi\left(\frac{1-z_i}{1-u_i}\right) (1-u_i)^\alpha du_i + \int_{z_i}^1 \phi\left(\frac{z_i}{u_i}\right) u_i^\alpha du_i \right). \quad (11)$$

If ϕ is a solution of the following equation

$$\phi(z) = \left(\frac{2}{\alpha+1} \right)^{\frac{K-s}{s}} \left(\int_0^z \phi\left(\frac{1-z}{1-u}\right) (1-u)^\alpha du + \int_z^1 \phi\left(\frac{z}{u}\right) u^\alpha du \right), \quad (12)$$

then it would be a solution of equation (11). As shown in [4],

$$\phi(z) = \mu(z(1-z))^{\delta-1}, \quad \delta = \left(\frac{2}{\alpha+1}\right)^{\frac{K-s}{s}},$$

is such a solution, where μ is an arbitrary constant and we have discarded additional terms in the general solution based on symmetry considerations.

Because the exponent α is a solution to the indicial equation (1) it follows that $\delta = \frac{\alpha}{2} + 1$ and hence the solution to (8) is:

$$f(z_0, \dots, z_{s-1}) = \nu_{s,K} \cdot \prod_{i=0}^{s-1} (z_i(1-z_i))^{\alpha/2},$$

where $\nu_{s,K}$ is a constant that depends on s and K only. To finish our derivation and to obtain the value of $\nu_{s,K}$ we replace f by the expression above in Condition 4 of Lemma 2 and we get:

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

so we can use the expression for $\beta_{s,K}$ in Equation (2) to find an explicit formula for $\nu_{s,K}$.

To argue unicity of the solution, we should begin noticing that the linear homogeneous PDE satisfied by the function f has all real-analytic coefficients in the domain $(0, 1)^s$, because the coefficients of the operators Ψ_i and Φ_i are analytic too in that domain and the PDE results from the composition of such operators.

Moreover, the highest derivative in the PDE is $\partial^{2s} f / \partial z_0^2 \cdots \partial z_{s-1}^2$ and its coefficient $\prod_{0 \leq i < s} z_i(1-z_i)$ is clearly always positive in $(0, 1)^s$, hence, the PDE is elliptic. Then, by Holmgren's theorem, any solution is real-analytic; and from Cauchy-Kovalevskaya theorem it follows that it must be unique, since this last theorem guarantees that there is a unique real-analytic solution (see for instance [8, 11]). Altogether, these results tell us that the solution that we have found, starting from the *ansatz* that it admitted a representation in separable variables, is unique.

It remains to verify by direct substitution that $P_{n,\mathbf{r}} = f(\mathbf{r}/n)n^\alpha$ is a solution of recurrence (5) replacing the independent term by $o(1)$, which is the error resulting from approximating the summations by integrals. With this our main result follows.

► **Theorem 3.** *If $\lim_{n \rightarrow \infty} \frac{P_{n,\mathbf{r}}}{n^\alpha}$ exists then the expected cost $P_{n,\mathbf{r}}$ of a PM query with given rank vector \mathbf{r} such that $r_i = z_i n + o(n)$ for some $z_i \in (0, 1)$, $0 \leq i < s$, in a random K -dimensional quadtree of size n is*

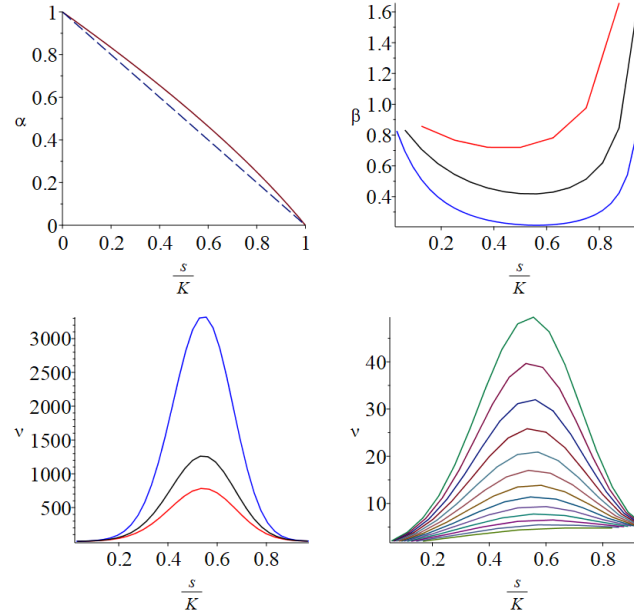
$$P_{n,\mathbf{r}} = \nu_{s,K} \left(\prod_{i=0}^{s-1} z_i(1-z_i)\right)^{\alpha/2} n^\alpha + o(n^\alpha),$$

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

$$(\alpha+2)^s (\alpha+1)^{K-s} = 2^K,$$

$$\nu_{s,K} = \frac{1}{(2^{K-s}-1)\Gamma(\alpha+1)^{K-s}\Gamma(\alpha/2+1)^{2s}} \prod_{2 \leq j \leq K} \frac{\Gamma(\alpha-\alpha_j)}{\Gamma(-\alpha_j)},$$

and the α_j 's, with $\alpha = \alpha_1 > \Re(\alpha_2) \geq \dots \geq \Re(\alpha_K)$, are the roots of the indicial equation above.



■ **Figure 3** Variation of the exponent $\alpha(s/K)$ (top-left), $\beta(s, K)$ for $K \in \{8, 16, 32\}$ (top-right) and $\nu(s, K)$ for $K \in \{30, 32, 36\}$ (bottom-left), as well as $\nu(s, K)$ for all $6 \leq K \leq 18$ (bottom-right).

Figure 3 depicts how the exponent $\alpha = \alpha(s/K)$, and the constants $\beta(s, K)$ and $\nu(s, K)$ vary with respect to s and K . In all cases, the x -axis is s/K to ease the comparison – α is a function of s/K alone, but β and ν depend on both s and K . In the graphs for $\beta(s, K)$ and $\nu(s, K)$ we have drawn three curves in each case, corresponding to $K = 8$ (red), $K = 16$ (black) and $K = 32$ (blue) in the graph for $\beta(s, K)$, and $K = 30$ (red), $K = 32$ (black) and $K = 36$ (blue) in the graph for $\nu(s, K)$. Moreover in the graph of $\alpha(s/K)$ we have also plotted $1 - s/K$ (dashed line) for reference. For fixed K , $\beta(s, K)$ is a convex function with a minimum close to $s = K/2$ but slowly shifted to the right. Likewise, for fixed K , $\nu(s, K)$ is a bell-shaped function with a single global maximum near $s = K/2$ but also slightly shifted to the right ($\nu(s, K)$ is not defined for $s = K$). If we denote $\nu^*(K) = \max_{0 < s < K} \nu(s, K)$ the graph shows that $\nu^*(K)$ grows with K . On the other hand, the graph and further numerical computations suggest that there is a limiting curve $\beta_\infty(x) = \lim_{K \rightarrow \infty} \beta(\lfloor xK \rfloor, K)$ that is a lower bound for any $\beta(s, K)$ as $K \rightarrow \infty$.

When $s = 0$ (no coordinate is specified), we have $\alpha(0) = \beta(0, K) = \nu(0, K) = 1$, despite all these constant are not well defined when $s = 0$. Notice that for $s = 0$ the partial match degenerates to a full traversal of the quadtree and visits its n nodes.

In the opposite situation, when all coordinates are specified, $s = K$, β and ν are undefined, and $\alpha(1) = 0$. The expected cost of a partial match is not $\Theta(1) = \Theta(n^0)$ but $\Theta(\log n)$ as it is actually an exact search.

4 Conclusions and Future Work

Our main result, Theorem 3, gives the main order term of the expected cost $P_{n, \mathbf{r}}$ of a PM search with a fixed query of rank vector \mathbf{q} , for quadtrees of any dimension K and any number

of specified coordinates. It can be easily translated to an equivalent result in terms of the coordinates q_i of the query, namely,

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

under the assumption of uniformity of the coordinates of the data points (see, for instance, [6]).

We show that quadtrees behave qualitatively as standard and relaxed K -d trees [6]. There we conjectured that the form of the expected cost of a PM search with fixed query would have the same “shape” for a wide variety of multidimensional data structures, excluding those producing very balanced partitions of the space (e.g., quadtries, squarish K -d trees). Duch and Lau [5] have disproved the conjecture, in its broadest terms, as it does not apply to locally balanced K -d trees. However, it seems that the conjecture might hold for hierarchical multidimensional data structures where: 1) no balancing of subtrees occurs; 2) the partition at each node follows a fixed rule independent of the current data point.

From the methodological viewpoint, we systematically exploit the many symmetries that appear in the problem to simplify its formulation and to make its mathematical manipulation feasible.

Several open problems remain. To begin with, the existence of $\lim_{n \rightarrow \infty} \frac{P_{n,r}}{n^\alpha}$, which has been rigorously proved for $K = 2$ in [3] (also in [1]); our result in that case coincides with the previous ones. We are currently working in the proof of the existence of the required limit for general K ; meanwhile, our results follow from the – yet unproven – assumption that such limit exists. We shall mention that there is compelling evidence that this is the case. On the other hand, the existence of a limiting distribution for $P_{n,r}/n^\alpha$ has been shown only for the case of standard 2-d trees and 2-dimensional quadtrees, but not for other data structures or larger dimensions, and this is a question worth of further study.

Another goal for future research, more technical in nature but also more ambitious, is to develop tools that would allow a straightforward, (semi-)automatic derivation of the recurrences or distributional equations, the proof of the existence of the limiting distribution, the corresponding integral equations for the expectation and other higher order moments, etc. This kind of techniques would ease the obtainment of results, such as the ones in previous literature and the ones in this paper, for many other multidimensional data structures and it might also open the door for “universality” results such as the ones conjectured in [6].

References

- 1 Nicolas Broutin, Ralph Neininger, and Henning Sulzbach. Partial match queries in random quadtrees. In Yuval Rabani, editor, *Proc. of the 23rd Annual ACM-SIAM Symposium on Discrete Algorithms (SODA)*, pages 1056–1065, 2012.
- 2 H.-H. Chern and H.-K. Hwang. Partial match queries in random quadtrees. *SIAM J. Comput.*, 32:904–915, 2003.
- 3 N. Curien and A. Joseph. Partial match queries in two-dimensional quadtrees: A probabilistic approach. *Advances in Applied Probability*, 43:178–194, 2011.
- 4 A. Duch, R. M. Jiménez, and C. Martínez. Selection by rank in k -dimensional binary search trees. *Random Structures and Algorithms*, 2012. doi:10.1002/rsa.20476.
- 5 Amalia Duch and Gustavo Lau. Partial match queries in relaxed K -dt trees. In *Proc. of the Fourteenth ACM-SIAM Workshop on Analytic Algorithmics and Combinatorics (ANALCO)*, pages 131–138, 2017. doi:10.1137/1.9781611974775.13.

- 6 Amalia Duch, Gustavo Lau, and Conrado Martínez. On the cost of fixed partial match queries in k-d trees. *Algorithmica*, 75(4):684–723, 2016. doi:10.1007/s00453-015-0097-4.
- 7 Philippe Flajolet, Gaston Gonnet, Claude Puech, and John Michael Robson. Analytic variations on quad trees. *Algorithmica*, 10:473–500, 1993.
- 8 Gerald B. Folland. *Introduction to Partial Differential Equations*. Princeton University Press, 2nd edition, 1995.
- 9 Steven G. Krantz and Harold R. Parks. *A Primer of Real Analytic Functions*. Birkhäuser, 2nd edition, 2002.
- 10 S. Ross. *A First Course in Probability*. Prentice Hall, Upper Saddle River, New Jersey, 8th edition, 2010.
- 11 Daniel Zwillinger. *Handbook of Differential Equations*. Academic Press, 3rd edition, 1997.

A Technical Lemmas

► **Lemma 4.** *If $f(z) = \lim_{n \rightarrow \infty} \frac{P_{n,r}}{n^\alpha}$ exists, with $\alpha = \alpha(1/2)$ the solution of the indicial equation (1) when $s = 1$ and $K = 2$, and $z = \lim_{n \rightarrow \infty} r/n$, $0 < z < 1$, then*

$$f(z) = \frac{2}{\alpha + 1} \left(\int_0^z f\left(\frac{1-z}{1-u}\right) (1-u)^\alpha du + \int_z^1 f\left(\frac{z}{u}\right) u^\alpha du \right). \quad (13)$$

The symmetry $P_{n,r_0} = P_{n,n-r_0}$ implies that in general $f(z) = f(1-z)$ and in particular $f\left(\frac{1-z}{1-u}\right) = f\left(1 - \frac{1-z}{1-u}\right)$ from where it follows that equation (13) is the same as the one for standard 2d-trees (see [4]).

Proof. Let $f(r_0/n) := P_{n,r_0}/n^\alpha$. Then we have that

$$\frac{P_{a,(b,*)}}{n^\alpha} = f\left(\frac{b}{a}\right) \left(\frac{a}{n}\right)^\alpha$$

and therefore, substituting into (4)

$$\begin{aligned} \frac{Q_{j_0,r_0}}{n^\alpha} &= \frac{1}{j_0 + 1} \sum_{n_{00}=0}^{j_0} \sum_{\hat{r}_0=0}^{r_0} \left(\frac{\binom{n_{00}}{\hat{r}_0} \binom{j_0-n_{00}}{r_0-\hat{r}_0}}{\binom{j_0}{r_0}} f\left(\frac{\hat{r}_0}{n_{00}}\right) \left(\frac{n_{00}}{n}\right)^\alpha \right) \\ &= \frac{1}{j_0 + 1} \sum_{n_{00}=0}^{j_0} \sum_{\hat{r}_0=0}^{r_0} \left(\frac{\binom{n_{00}}{\hat{r}_0} \binom{j_0-n_{00}}{r_0-\hat{r}_0}}{\binom{j_0}{r_0}} f\left(\frac{\hat{r}_0}{j_0} \frac{j_0}{n} \frac{n}{n_{00}}\right) \left(\frac{n_{00}}{n}\right)^\alpha \right) \end{aligned}$$

The last sum is the expected value of a function of a hypergeometric random variable. Passing to the limit when $n \rightarrow \infty$, Lemma 6 allows us to exchange the expected value and the function. Therefore passing to the limit when $n \rightarrow \infty$, with $z = \lim_{n \rightarrow \infty} (r/n)$, $u_{0*} = \lim_{n \rightarrow \infty} (j_0/n)$, $u_{00} = \lim_{n \rightarrow \infty} (n_{00}/n)$, and assuming that f is real analytic in Lemma 6 we can apply it to get:

$$\begin{aligned} \lim_{n \rightarrow \infty} \frac{Q_{j_0,r_0}}{n^\alpha} &= \frac{1}{u_{0*}} \int_0^{u_{0*}} f\left(\frac{u_{00}}{u_{0*}} \frac{z}{u_{0*}} \frac{u_{0*}}{u_{00}}\right) u_{00}^\alpha du_{00} = \frac{1}{u_{0*}} \int_0^{u_{0*}} f\left(\frac{z}{u_{0*}}\right) u_{00}^\alpha du_{00} \\ &= \frac{1}{\alpha + 1} f\left(\frac{z}{u_{0*}}\right) u_{0*}^\alpha. \end{aligned}$$

and similarly

$$\begin{aligned} \lim_{n \rightarrow \infty} \frac{Q_{n-1-j_0,n-r_0}}{n^\alpha} &= \frac{1}{1-u_{0*}} \int_0^{1-u_{0*}} f\left(\frac{1-z}{1-u_{0*}}\right) u_{00}^\alpha du_{00} \\ &= \frac{1}{\alpha + 1} f\left(\frac{1-z}{1-u_{0*}}\right) (1-u_{0*})^\alpha \end{aligned}$$

Since $j_0 = 0 \implies u_{0*} = 0$, $j_0 = r_0 \implies u_{0*} = z_0$ and in the limit $j_0 = r_0 - 1 \implies u_{0*} = z_0$, $j_0 = n - 1 \implies u_{0*} = 1$ and $\frac{\Delta j_0}{n} \rightarrow du_{0*}$ replacing in (3) and passing to the limit we obtain this integral equation:

$$\begin{aligned} f(z_0) &= 2 \int_0^{z_0} \frac{1}{1-u_{0*}} f\left(\frac{1-z_0}{1-u_{0*}}\right) \int_0^{1-u_{0*}} u_{00}^\alpha du_{00} du_{0*} \\ &\quad + 2 \int_{z_0}^1 \frac{1}{u_{0*}} f\left(\frac{z_0}{u_{0*}}\right) \int_0^{u_{0*}} u_{00}^\alpha du_{00} du_{0*} \\ &= 2 \int_0^{z_0} \frac{1}{1-u_{0*}} f\left(\frac{1-z_0}{1-u_{0*}}\right) \frac{(1-u_{0*})^{\alpha+1}}{\alpha+1} du_{0*} + 2 \int_{z_0}^1 \frac{1}{u_{0*}} f\left(\frac{z_0}{u_{0*}}\right) \frac{u_{0*}^{\alpha+1}}{\alpha+1} du_{0*}. \end{aligned}$$

Replacing now in (3), passing to the limit $n \rightarrow \infty$ and, to simplify, replacing u_{0*} by u we get the integral equation (13) in the statement of the Lemma. \blacktriangleleft

► **Lemma 5.** *Given a random K dimensional quadtree with n data points the conditional probability that $\mathcal{N}_{0^K} = \ell_K$ given that $\mathcal{N}_{\langle i \rangle} = n_{\langle i \rangle}$ for $0 \leq i \leq K - 1$ is:*

$$\Pr \left\{ \mathcal{N}_{0^K} = \ell_K \mid \bigwedge_{i=0}^{K-1} \mathcal{N}_{\langle i \rangle} = n_{\langle i \rangle} \right\} = \sum_{\ell_{K-1}=0}^{n_{\langle i \rangle K-2}} \cdots \sum_{\ell_3=0}^{n_{\langle 2 \rangle}} \sum_{\ell_2=0}^{n_{\langle 1 \rangle}} \left(\frac{\binom{\ell_1}{\ell_2} \binom{n-1-\ell_1}{n_{\langle 1 \rangle}-\ell_2}}{\binom{n-1}{n_{\langle 1 \rangle}}} \frac{\binom{\ell_2}{\ell_3} \binom{n-1-\ell_2}{n_{\langle 2 \rangle}-\ell_3}}{\binom{n-1}{n_{\langle 2 \rangle}}} \cdots \frac{\binom{\ell_{K-2}}{\ell_{K-1}} \binom{n-1-\ell_{K-2}}{n_{\langle K-2 \rangle}-\ell_{K-1}}}{\binom{n-1}{n_{\langle K-2 \rangle}}} \frac{\binom{\ell_{K-1}}{\ell_K} \binom{n-1-\ell_{K-1}}{n_{\langle K-1 \rangle}-\ell_K}}{\binom{n-1}{n_{\langle K-1 \rangle}}} \right). \quad (14)$$

Proof. In the base case $K = 2$ given n , $\mathcal{N}_{\langle 0 \rangle} \equiv \mathcal{N}_{0*} = n_{0*}$ and $\mathcal{N}_{\langle 1 \rangle} \equiv \mathcal{N}_{*0} = n_{*0}$, the probability that the intersection of the rectangles $B_{\langle 0 \rangle} = B_{0*}$ and $B_{\langle 1 \rangle} = B_{*0}$ contains $\ell_2 = n_{00}$ nodes is the probability of having $\ell_2 = n_{00}$ successes in n_{*0} draws without replacement from a population of size $n - 1$ that contains n_{0*} successes. It is $n - 1$ instead of n because the root cannot be in the intersections. Therefore the distribution is hypergeometric:

$$\Pr \{ \mathcal{N}_{00} = n_{00} \mid \mathcal{N}_{0*} = n_{0*}, \mathcal{N}_{*0} = n_{*0} \} = \frac{\binom{n_{0*}}{n_{00}} \binom{n-1-n_{0*}}{n_{*0}-n_{00}}}{\binom{n-1}{n_{*0}}}.$$

Assume that the lemma is true for K dimensions. We can do the inductive step based on writing the intersection of $K + 1$ sets as an intersection of K sets followed by the intersection of two sets:

$$\bigcap_{i=0}^K F_{*^i 0^{*K-i}} = \left(\bigcap_{i=0}^{K-1} F_{*^i 0^{*K-i}} \right) \cap F_{*^K 0} = F_{0^{K*}} \cap F_{*^K 0} = F_{0^{K+1}}.$$

Taking into account all the possible values of $\mathcal{N}_{0^{K*}}$, we have:

$$\begin{aligned} &\Pr \left\{ \mathcal{N}_{0^{K+1}} = n_{0^{K+1}} \mid \bigwedge_{i=0}^K \mathcal{N}_{*^i 0^{*K-i}} = n_{*^i 0^{*K-i}} \right\} \\ &= \sum_{n_{0^{K*}}=0}^{n_{*^K-1} 0^{*0}} \left(\Pr \left\{ \mathcal{N}_{0^{K*}} = n_{0^{K*}} \mid \bigwedge_{i=0}^{K-1} \mathcal{N}_{*^i 0^{*K-i}} = n_{*^i 0^{*K-i}} \right\} \right. \\ &\quad \left. \times \Pr \left\{ \mathcal{N}_{0^{K+1}} = n_{0^{K+1}} \mid \mathcal{N}_{0^{K*}} = n_{0^{K*}}, \mathcal{N}_{*^K 0} = n_{*^K 0} \right\} \right) \\ &= \sum_{n_{0^{K*}}=0}^{n-1} \left(\Pr \left\{ \mathcal{N}_{0^{K*}} = n_{0^{K*}} \mid \bigwedge_{i=0}^{K-1} \mathcal{N}_{*^i 0^{*K-i}} = n_{*^i 0^{*K-i}} \right\} \times \frac{\binom{n_{0^{K*}}}{n_{0^{K+1}}} \binom{n-1-n_{0^{K*}}}{n_{*^K 0}-n_{0^{K+1}}}}{\binom{n-1}{n_{*^K 0}}} \right), \end{aligned}$$

20:16 Fixed Partial Match Queries in Quadrees

applying the inductive hypothesis (14) (adding a $*$ to the end of each string) completes the proof. Notice that we have used ℓ_i instead of $n_{0^i * \kappa^{-i}}$ and $n_{(i)} = n_{*^i 0 * \kappa^{-1-i}}$ in the statement of the theorem. \blacktriangleleft

► **Lemma 6.** *Given a random two dimensional quadtree let \mathcal{N}_{0*} , \mathcal{N}_{*0} and \mathcal{N}_{00} be respectively the random variables of the number of nodes west, south and south-west of the root. If f is a real analytic function [9] in $(0, 1)$, $\lim_{n \rightarrow \infty} n_{0*}/n = u_{0*}$ and $\lim_{n \rightarrow \infty} n_{*0}/n = u_{*0}$, where $u_{0*}, u_{*0} \in (0, 1)$, then*

$$\begin{aligned} \lim_{n \rightarrow \infty} \mathbb{E} \left\{ f \left(\frac{\mathcal{N}_{00}}{n} \right) \mid \mathcal{N}_{0*} = n_{0*}, \mathcal{N}_{*0} = n_{*0} \right\} &= \lim_{n \rightarrow \infty} \sum_{n_{00}=0}^{n_{*0}} \left(\frac{\binom{n_{0*}}{n_{00}} \binom{n-n_{0*}}{n_{*0}-n_{00}}}{\binom{n}{n_{*0}}} f \left(\frac{n_{00}}{n} \right) \right) \\ &= \lim_{n \rightarrow \infty} \sum_{n_{00}=0}^{n_{*0}} \left(\frac{\binom{n_{0*}}{n_{00}} \binom{n-n_{0*}}{n_{*0}-n_{00}}}{\binom{n}{n_{*0}}} f \left(\frac{n_{00}}{n} \right) \right) \\ &= f(u_{0*} u_{*0}). \end{aligned} \quad (15)$$

Proof. For simplicity, in the hypergeometric probability formulas we have replaced $n-1$ by n as in the limit they are the same.

Since f is real analytic all derivatives of f exist in $(0, 1)$ and we can write, for some $x_0 \in (0, 1)$,

$$f(x) = \sum_{i=0}^{\infty} a_i (x - x_0)^i = \sum_{i=0}^{\infty} a_i \sum_{k=0}^i \binom{i}{k} (-x_0)^{i-k} x^k.$$

Since the series on the right side converges we can use the linearity of expectations:

$$\mathbb{E} \{ f(x) \} = \sum_{i=0}^{\infty} a_i \sum_{k=0}^i \binom{i}{k} (-x_0)^{i-k} \mathbb{E} \{ x^k \}.$$

Therefore we only need to prove the lemma for $f(x) = x^k$. If $X_{n,m,N}$ is a hypergeometric random variable with parameters n , m , and N then [10]:

$$\mathbb{E} \{ X_{n,m,N}^k \} = \frac{nm}{N} \mathbb{E} \{ (X_{n-1,m-1,N-1} + 1)^{k-1} \}.$$

Based on that it is easy to prove by induction that for every $k \in \mathbb{N}$ there are integers $c_{k,i}$, with $c_{k,k} = 1$, such that:

$$\mathbb{E} \{ X_{n,m,N}^k \} = \sum_{i=0}^k c_{k,i} \frac{n^i m^i}{N^i}.$$

Therefore if $f(x) = x^k$:

$$\begin{aligned} \mathbb{E} \left\{ f \left(\frac{\mathcal{N}_{00}}{n} \right) \mid \mathcal{N}_{0*} = n_{0*}, \mathcal{N}_{*0} = n_{*0} \right\} &= \mathbb{E} \left\{ \frac{\mathcal{N}_{00}^k}{n^k} \mid \mathcal{N}_{0*} = n_{0*}, \mathcal{N}_{*0} = n_{*0} \right\} \\ &= \frac{\sum_{i=0}^k c_{k,i} \frac{n_{0*}^i n_{*0}^i}{n^i}}{n^k} = \sum_{i=0}^k c_{k,i} \frac{n_{0*}^i n_{*0}^i}{n^i n^k}. \end{aligned}$$

In the last sum the only term that does not go to zero as $n \rightarrow \infty$ is the last one, where $i = k$. Given that $c_{k,k} = 1$, we have:

$$\lim_{n \rightarrow \infty} \mathbb{E} \left\{ \frac{\mathcal{N}_{00}^k}{n^k} \mid \mathcal{N}_{0*} = n_{0*}, \mathcal{N}_{*0} = n_{*0} \right\} = \lim_{n \rightarrow \infty} \frac{\frac{n_{0*}^k n_{*0}^k}{n^k}}{n^k} = \lim_{n \rightarrow \infty} \left(\frac{n_{0*}}{n} \right)^k \left(\frac{n_{*0}}{n} \right)^k = u_{0*}^k u_{*0}^k.$$

That proves the lemma for $f(x) = x^k$. \blacktriangleleft

The lemma can be generalised to any dimension K using mathematical induction on the number of dimensions, again assuming that the function f is real analytic (in several variables).

► **Lemma 7.** *Given a random quadtree let $\mathcal{N}_{(i)}$ be the random variable of the number of data points that have their i -th coordinate less than the i -th coordinate of the root and the rest of the coordinates undetermined and let \mathcal{N}_{0^k} be the random variable of the size of the cuboid where all the coordinates have values lower than the respective coordinates of the root. If f is real analytic in $(0, 1)^K$, $\lim_{n \rightarrow \infty} n_{(i)}/n = u_i$ for $0 \leq i < K$, where $u_i \in (0, 1)$, then*

$$\lim_{n \rightarrow \infty} \mathbb{E} \left\{ f \left(\frac{\mathcal{N}_{0^k}}{n} \right) \middle| \bigwedge_{i=0}^{K-1} \mathcal{N}_{(i)} = n_{(i)} \right\} = f \left(\prod_{i=0}^{K-1} u_i \right).$$

Proof. The base case $K = 2$ has been proved. Assume that the lemma is true for K dimensions. Then:

$$\begin{aligned} & \lim_{n \rightarrow \infty} \mathbb{E} \left\{ f \left(\frac{\mathcal{N}_{0^{k+1}}}{n} \right) \middle| \bigwedge_{i=0}^K \mathcal{N}_{*^i 0^{*k-i}} = n_{*^i 0^{*k-i}} \right\} \\ &= \lim_{n \rightarrow \infty} \sum_{n_{0^{k+1}}=0}^{n-1} \Pr \left\{ \mathcal{N}_{0^{k+1}} = n_{0^{k+1}} \middle| \bigwedge_{i=0}^K \mathcal{N}_{*^i 0^{*k-i}} = n_{*^i 0^{*k-i}} \right\} f \left(\frac{n_{0^{k+1}}}{n} \right) \\ &= \lim_{n \rightarrow \infty} \sum_{n_{0^{k+1}}=0}^{n-1} \sum_{n_{0^{k*}}=0}^{n-1} \left(\Pr \left\{ \mathcal{N}_{0^{k*}} = n_{0^{k*}} \middle| \bigwedge_{i=0}^{K-1} \mathcal{N}_{*^i 0^{*k-i}} = n_{*^i 0^{*k-i}} \right\} \right. \\ & \quad \times \Pr \left\{ \mathcal{N}_{0^{k+1}} = n_{0^{k+1}} \middle| \mathcal{N}_{0^{k*}} = n_{0^{k*}}, \mathcal{N}_{*^k 0} = n_{*^k 0} \right\} f \left(\frac{n_{0^{k+1}}}{n} \right) \\ &= \lim_{n \rightarrow \infty} \sum_{n_{0^{k*}}=0}^{n-1} \Pr \left\{ \mathcal{N}_{0^{k*}} = n_{0^{k*}} \middle| \bigwedge_{i=0}^{K-1} \mathcal{N}_{*^i 0^{*k-i}} = n_{*^i 0^{*k-i}} \right\} \\ & \quad \times \mathbb{E} \left\{ f \left(\frac{\mathcal{N}_{0^{k+1}}}{n} \right) \middle| \mathcal{N}_{0^{k*}} = n_{0^{k*}}, \mathcal{N}_{*^k 0} = n_{*^k 0} \right\} \\ &= \lim_{n \rightarrow \infty} \sum_{n_{0^{k*}}=0}^{n-1} \Pr \left\{ \mathcal{N}_{0^{k*}} = n_{0^{k*}} \middle| \bigwedge_{i=0}^{K-1} \mathcal{N}_{*^i 0^{*k-i}} = n_{*^i 0^{*k-i}} \right\} \times f \left(\lim_{n \rightarrow \infty} \frac{n_{0^{k*}} n_{*^k 0}}{(n-1)n} \right) \\ &= \lim_{n \rightarrow \infty} \mathbb{E} \left\{ f \left(\lim_{n \rightarrow \infty} \frac{\mathcal{N}_{0^{k*}} n_{*^k 0}}{(n-1)n} \right) \middle| \bigwedge_{i=0}^{K-1} \mathcal{N}_{*^i 0^{*k-i-1 0}} = n_{*^i 0^{*k-i-1 0}} \right\}. \end{aligned}$$

Replacing $n-1$ by n , because in the limit they are equivalent, and using the induction hypothesis (adding 0 at the end of each string) we have:

$$\begin{aligned} & \lim_{n \rightarrow \infty} \mathbb{E} \left\{ f \left(\frac{\mathcal{N}_{0^{k+1}}}{n} \right) \middle| \bigwedge_{i=0}^K \mathcal{N}_{*^i 0^{*k-i}} = n_{*^i 0^{*k-i}} \right\} = f \left(\lim_{n \rightarrow \infty} \left(\prod_{i=0}^{K-1} \frac{n_{*^i 0^{*k-i-1 0}}}{n_{*^k 0}} \right) \frac{n_{*^k 0}}{n} \right) \\ &= f \left(\lim_{n \rightarrow \infty} \prod_{i=0}^K \frac{n_{*^i 0^{*k-i}}}{n} \right) = f \left(\prod_{i=0}^K u_{*^i 0^{*k-i}} \right). \end{aligned}$$

◀

► **Lemma 8.** *The real function $f(x) = x^a(1-x)^a$ is real analytic, i. e. it is infinitely differentiable and agrees with its Taylor series, in the interval $(0, 1)$ for any real number a .*

20:18 Fixed Partial Match Queries in Quadrees

Proof. By the binomial series, or Newton's generalized binomial theorem, $f_1(x) = (1 - x)^a$ is real analytic in $(-1, 1)$ and $f_2(x) = x^a = (1 + (x - 1))^a$ is real analytic in $(0, 2)$. Therefore their product, $f(x)$, is real analytic in $(0, 1)$. ◀

Chapter 8

Quad- K -d trees

The main result of the article in this chapter [DLM16b] is an asymptotic formula for the expected cost of random PM queries in several families of random quad- K -d trees, see Theorem 1 (page 162). This result unifies into a single theorem the fact that random PM queries in both relaxed K -d trees and quad trees satisfy $\hat{P}_{n,s,\kappa} = \Theta(n^{\alpha\rho})$. Furthermore, we have promising on-going work that could generalize both Theorem 4.4 and Theorem 4.6. A node is of type m if it discriminates by m coordinates. In that case it has 2^m children. We call τ_m the probability that a node is of type m . We obtain the case of relaxed K -d trees when $\tau_1 = 1$ and $\tau_i = 0$ if $i \neq 1$, while when $\tau_\kappa = 1$ and $\tau_i = 0$ if $i \neq \kappa$ we have the case of quad trees.

Note that in this article we use P , with subscripts, for the expected cost of random PM queries instead of \hat{P} .

8.1 Article: Random PM in Quad- K -d Trees

This article is [DLM16b]:

Amalia Duch, Gustavo Lau, and Conrado Mart'mez. "Random Partial Match in Quad- K -d Trees". In: *LATIN 2016: Theoretical Informatics - 12th Latin American Symposium, Ensenada, Mexico, April 11-15, 2016, Proceedings*. 2016, pp. 376–389. doi: [10.1007/978-3-662-49529-2_28](https://doi.org/10.1007/978-3-662-49529-2_28)

Attention!

Pages 154 to 168 of the thesis are available at the editor's web
https://link.springer.com/chapter/10.1007/978-3-662-49529-2_28

Part IV

**Conclusions and Open
Problems**

The main contribution of this thesis is to deepen and generalize previous work done in the average-case analysis of partial match queries in several types of multidimensional search trees. In Table 7 we show references to the papers that include contributions to the analysis of the expected cost for several trees and query types. The ones included in this thesis are highlighted in gray. As you can see, for K -d trees only the case of $s = 1$ was previously known for fixed PM queries and we generalize it to $s > 1$. We introduce a new type of multidimensional tree, the relaxed K -dt tree in [DL17], and analyze both random and fixed PM queries in them. For quad trees we analyze the case $K > 2$ while previously only the case $K = 2$ was studied. Finally, we analyze random PM queries in the family of relaxed quad- K -d trees obtaining a result that has previous results for both quad trees and relaxed K -d trees as particular cases.

Table 7: Analysis of the expected cost of partial match queries

Data structure	Query type	Standard	Relaxed	Squarish
K -d trees ($t = 0$)	Random	[FP86; CH06]	[DEM98; MPP01]	[DJZ00]
	Fixed	$s = 1$: [DJM14] $s > 1$: [DLM16a]	$s = 1$: [DJM14] $s > 1$: [DLM14]	
K -dt trees ($t > 0$)	Random	[CLF89; CH06]	[DL17]	
	Fixed		[DL17]	
Quad trees	Random	[Fla+93; CH03]		
	Fixed	$K = 2$: [CJ11; BNS13] $K > 2$: [DLM18]		
Quad- K -d trees	Random		[DLM16b]	
	Fixed			

We summarize current knowledge regarding the expected cost of partial match queries in multidimensional point search trees in Table 8.

Regarding random PM queries, our results can be summarized in two very general unpublished theorems: Theorem 6.1 for relaxed K -dt trees and Theorem 6.2 for standard K -dt trees. The expected cost of random PM queries in them is:

$$\hat{P}_{n,t,\mathbf{u}} = \beta_{t,\mathbf{u}} n^{\alpha_{t,\rho,\rho_0}} + o(n^{\alpha_{t,\rho,\rho_0}}).$$

Table 8: Asymptotic expected cost of partial match queries

	Relaxed K -d trees Standard K -d trees Quad trees Quad- K -d trees	Relaxed K -dt trees Standard K -dt trees $t > 0$	Squarish K -d trees
	$\alpha > 1 - s/K$	$\alpha > 1 - s/K$	$\alpha = 1 - s/K$
Random	$\beta_{\mathbf{u}} n^\alpha$	$\beta_{\mathbf{u}} n^\alpha$	$\theta(n^{1-s/K})$ Also true for Standard K -dt trees as $t \rightarrow \infty$
Fixed	$\nu_{\mathbf{u}} n^\alpha \left(\prod_{q_i \neq * } q_i(1 - q_i) \right)^{\alpha/2}$ Conjecture for Quad- K -d trees	$\neq \nu_{\mathbf{u}} n^\alpha \left(\prod_{q_i \neq * } q_i(1 - q_i) \right)^{\alpha/2}$ [DL17] (Conjecture for Standard)	$\nu_\rho n^{1-s/K}?$ Conjecture supported by experimental evidence

For relaxed and standard K -dt trees α_{t,ρ,ρ_0} is respectively the unique solution in the interval $(0, 1)$ of the equations:

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

$$\left(\frac{(t+2)^{\overline{t+1}}}{2(\alpha+t+1)^{\overline{t+1}}} \right)^{\rho_0} \left(\frac{(t+2)^{\overline{t+1}}}{(\alpha+t+2)^{\overline{t+1}}} \right)^{\rho - \rho_0} \left(\frac{(t+2)^{\overline{t+1}}}{(\alpha+t+1)^{\overline{t+1}}} \right)^{1 - \rho} = 1.$$

Note that the two equations are very similar. As we mentioned on [Chapter 6](#), in the case of relaxed K -dt trees the left-hand side is a weighted arithmetic mean and in the case of standard ones is a geometric one of the same three terms.

Regarding fixed PM queries, in [\[DLM14; DLM16a\]](#) we analyzed them for relaxed and standard K -d trees in the case $s > 1$, generalizing the results of [\[DJM14\]](#). There we conjectured that

$$P_{n,\mathbf{q}} = \nu_{\mathbf{u}(\mathbf{q})} \cdot \left(\prod_{i:q_i \text{ is specified}} q_i(1 - q_i) \right)^{\alpha/2} \cdot n^\alpha + o(n^\alpha) \quad (8.1)$$

for quad trees and variants of K -d trees but not for squarish K -d trees. For the latter we conjectured that:

$$P_{n,\mathbf{q}} = \nu_\rho n^{1-\rho} + o(n^{1-\rho}) \quad (8.2)$$

In [\[DL17\]](#) we analyzed fixed PM queries in relaxed K -dt trees and, even though we could not find an explicit formula for them, we disproved the conjecture [\(8.1\)](#) for $t = 1$.

In [DLM18] we performed the analysis of the expected cost of fixed PM queries for quad trees in the case $K > 2$, generalizing the result of [CJ11]. However, our analysis needs the assumption of the existence of $\lim_{n \rightarrow \infty} \frac{P_{n,r}}{n^\alpha}$. With this result we proved that the conjecture (8.1) is correct for quad trees of any dimension.

Open problems

Regarding the expected cost of random PM queries, in the case of random relaxed and standard K -d trees with s_0 extreme specified coordinates we have the exponent α 's of expected cost of random PM queries—Theorem 6.1 and Theorem 6.2—but we still have to find formulas for their β 's. This implies a laborious and cumbersome computation but the analysis seems to be quite feasible.

For relaxed quad- K -d trees we have promising on-going work on a more precise version of Theorem 1 of [DLM16b]. In that theorem we only stated that $\hat{P}_{n,s,K,\tau} = \Theta(n^{\alpha_{s,K,\tau}})$, but we are close to having a proof that the conjecture on page 10 of [DLM16b] is true for random relaxed quad- K -d trees, that is

$$\hat{P}_{n,s,K,\tau} = \beta_{s,K,\tau} n^{\alpha_{s,K,\tau}} + o(n^{\alpha_{s,K,\tau}}).$$

Actually, we have an exact formula for $\hat{P}_{n,s,K,\tau}$ and only the final asymptotic analysis is missing. This new result would generalize both Theorem 4.4 for quad trees and Theorem 4.6 for relaxed K -d trees.

For quad- K -d trees as well, we would like to define a class of them that includes standard K -d trees and perform the analysis of random PM queries in them. More generally, it remains to prove or disprove the conjecture on page 10 of [DLM16b]: that the average cost of random PM queries in all random quad- K -d trees is of the form $P_n = \beta n^\alpha + o(n^\alpha)$.

Beyond multidimensional trees, another task is to analyze random PM queries in quad tries and quad- K -d tries. This was done for standard K -d tries in [FP86] and for relaxed K -d tries in [MPP01].

Regarding the expected cost of fixed PM queries, as mentioned in [DLM18], for quad trees we need to prove that $\lim_{n \rightarrow \infty} \frac{P_{n,r}}{n^\alpha}$ exists to fully generalize the result about the expected cost that was proved for $K = 2$ in [CJ11]. Maybe the “bounding errors” approach that we applied to K -d trees could be applied to quad trees.

We would like to find a way to solve the differential equations that can be obtained using Proposition 5.2 of [DL17] to find an explicit formula for the expected cost of fixed PM queries in relaxed K -d trees with $t > 0$. Once that is done, we would try to compute the expected cost of fixed PM queries for standard K -d trees with $t > 0$.

For the case of quad- K -d trees we do not have any results about the expected cost of fixed PM queries. A natural place to start analyzing them would be the subclass of quad- K -d trees for which we analyzed random PM queries in [DLM16b], that is the relaxed ones. We also would like to define a class of

quad- K -d trees that includes standard K -d trees; once the analysis of random PM queries is done for them we would try to perform the analysis of fixed PM queries. The final goal would be to prove the conjecture (8.1) for quad- K -d trees or characterize for which families of them it is true.

In the case of squarish K -d trees we would like to prove or disprove the conjecture (8.2) about the the expected cost of fixed PM queries in them. Based on the experiments done for [DLM16a], we think that the higher order term is independent of the coordinates of the query and its pattern \mathbf{u} but depends on s and K . We also think that such conjecture is valid for K -d tries, quad tries and quad- K -d tries but it remains to analyze PM queries in all of them.

Regarding the variance of fixed PM queries, we would like to prove or disprove the related conjecture made about it in [DLM16a]:

$$\mathbb{V}[\mathcal{P}_{n,\mathbf{q}}] = \nu_{\mathbf{u}(\mathbf{q})}^{(2)} \cdot \left(\prod_{i=0}^{s_R-1} q_i(1-q_i) \right)^{\alpha/2} \cdot n^{2\alpha} + o(n^{2\alpha}).$$

A good place to start would be to compute the variance of fixed PM queries for relaxed K -dt trees and then proceed to standard K -dt trees. In both cases the variance is unknown even for $t = 0$.

Regarding the probability distribution of fixed PM queries in K -d trees, quad trees and quad- K -d trees the existence of a limiting distribution for $\mathcal{P}_{n,\mathbf{r}}/n^\alpha$ remains to be proved. That has been done, using the contraction method, only for the case of standard 2-d trees and 2-dimensional quad trees [BNS13]. More ambitious yet is to prove the following conjectures mentioned on page 30 of [DLM16a]. We denote by $X_n \xrightarrow{(d)} X$ the convergence in distribution, as $n \rightarrow \infty$, of the sequence of random variables X_n to the random variable X . The first conjecture is that for multidimensional search trees, other than squarish K -d trees, where no balancing of subtrees occurs:

$$\left(\frac{\mathcal{P}_{n,\mathbf{q}}}{n^\alpha} \right) \xrightarrow{(d)} \frac{\Gamma^s(\alpha+2)}{\Gamma^{2s}(\alpha/2+1)} \left(\prod_{i:q_i \neq *, q_i \notin \{0,1\}} (q_i(1-q_i))^{\alpha/2} \right) \cdot \Phi_{\mathbf{u}(\mathbf{q})}, \quad (8.3)$$

where $\Phi_{\mathbf{u}(\mathbf{q})}$ is the random variable such that

$$\frac{\hat{\mathcal{P}}_{n,\mathbf{u}}}{n^\alpha} \xrightarrow{(d)} \Phi_{\mathbf{u}(\mathbf{q})}.$$

The second conjecture is that for squarish K -d trees:

$$\frac{\mathcal{P}_{n,\mathbf{q}}}{n^{1-\rho}} \xrightarrow{(d)} \nu_\rho \quad (8.4)$$

for some constant ν_ρ that depends on the ratio $\rho = s/K$.

Analogous to the analysis of fixed PM queries we could try to perform the analysis of other associative queries with fixed parameters, for nearest neighbor queries with a given fixed query point \mathbf{q} and for orthogonal range queries with a fixed hyperrectangle query $Q = [\ell_0, u_0] \times \cdots \times [\ell_{K-1}, u_{K-1}]$.

We would like to be able to define combinatorial structures so that by applying the symbolic method of analytic combinatorics we could directly get the generating function equations, at least for random PM queries. Some progress was made in this direction by [MPP01] while analyzing the randomized PM in relaxed K -d trees. One avenue would be to try to use multivariate generating functions. For example, for the case $s = 1$:

$$P(z, y, u) = \sum_T \sum_{0 \leq r \leq |T|} \mathbb{P}[T] \frac{z^{|T|}}{|T|!} y^r u^{P_{r,T}},$$

where the first sum is over all the trees T under consideration, $\mathbb{P}[T]$ is the probability of generating the tree T , $|T|$ denotes the size of T , r is the rank of the PM query (only one given that $s = 1$), and $P_{r,T}$ is the cost of searching rank r in T .

Finally, as we mentioned in [DLM18], a more general and ambitious goal is to develop (semi-)automatic tools for the derivation of the recurrences or distributional equations, the corresponding integral equations for the expectation and other higher order moments and even for the proof of the existence of the limiting distribution.

Part V

Appendices

Appendix A

Theorems used in proofs

A.1 Roura's Continuous Master Theorem

Roura's Continuous Master Theorem (CMT) [Rou01] applies to a wide class of full-history recurrences whose coefficients can be well-approximated asymptotically by a so-called *shape function* $\omega : [0, 1] \rightarrow \mathbb{R}$. The shape function describes the coefficients only depending on the *ratio* j/n of the subproblem size j and the current size n (not depending on n or j itself) and it smoothly continues their behavior to any real number $z \in [0, 1]$. This continuous point of view also allows computation of precise asymptotics for complex discrete recurrences via fairly simple integrals.

Definition 16 (Roura [Rou01]). *Let $\omega(z) \geq 0$ be a function over $[0, 1]$ such that $\int_0^1 \omega(z) dz$ exists and is at least 1. Furthermore, assume that there is some $\mu < 0$ such that $\int_0^1 \omega(z) z^\mu dz$ also converges. Then we say that $\omega(z)$ is a shape function.*

Definition 17 (Roura [Rou01]). *We say that*

$$F_n = \begin{cases} b_n & \text{for } 0 \leq n < N, \\ t_n + \sum_{0 \leq k < n} \omega_{n,k} F_k & \text{for } n \geq N \end{cases} \quad (\text{A.1})$$

is a continuous recursive definition of F_n iff there exist some shape function $\omega(z)$, some constant $0 < q \leq 1$ and some function $M_n = \Theta(n^q)$ with integer values such that, if we define $z_{n,j} = j/M_n$ for every $0 \leq j \leq M_n$, $I_{n,j} = [z_{n,j} \cdot n, z_{n,j+1} \cdot n)$ for every $0 \leq j < M_n$, and

$$\varepsilon_{n,j} = \left| \sum_{k \in I_{n,j}} \omega_{n,k} - \int_{z_{n,j}}^{z_{n,j+1}} \omega(z) dz \right|$$

for every $0 \leq j < M_n$, then $\sum_{0 \leq j < M_n} \varepsilon_{n,j} = \mathcal{O}(n^{-\rho})$ for some $\rho > 0$.

Lemma A.1 (Roura, Lemma 7.2 of [Rou01]). *Let*

$$\omega_{n,k} = A \frac{(a_1 n + b_1 k + c_1) \dots (a_m n + b_m k + c_m)}{(n + d_1) \dots (n + d_{m+1})}$$

be the weights of a given recurrence as in (A.1), where $A > 0$ is a constant, $m \geq 0$ and a_i, b_i, c_i are constants such that $a_i b_i \neq 0$ for all $1 \leq i \leq m$. Define

$$\omega(z) := A(a_1 + b_1 z) \dots (a_m + b_m z).$$

If

$$\int_0^1 \omega(z) dz \geq 1$$

then the given recurrence is continuous and $\omega(z)$ is its shape function.

Theorem A.2 (Continuous Master Theorem (CMT), Roura [Rou01]). *Let F_n be a function defined by a continuous recursive definition, where the toll function t_n satisfies $t_n \sim \tilde{K} n^a \ln^b(n)$ as $n \rightarrow \infty$ for constants $\tilde{K} \neq 0$, $a \geq 0$ and $b > -1$. With $H = 1 - \int_0^1 z^a \omega(z) dz$, we have the following cases:*

1. *If $H > 0$, then $F_n \sim \frac{t_n}{H}$.*
2. *If $H = 0$, then $F_n \sim \frac{t_n \ln(n)}{\tilde{H}}$ with $\tilde{H} = -(b+1) \int_0^1 z^a \ln(z) \omega(z) dz$.*
3. *If $H < 0$ (including the case $H = -\infty$), then $F_n = \mathcal{O}(n^\alpha)$ ($F_n = \Theta(n^\alpha)$), if $F_n \geq 0$ for every $n \geq 0$), where α is the unique non-negative real solution of $\int_0^1 z^\alpha \omega(z) dz = 1$.*

A.2 Flajolet and Odlyzko's Transfer Lemma

The work of Flajolet and Odlyzko [FO90] includes several Transfer Theorems where, as stated in [FS09], the “general objective is to translate an approximation of a function near a singularity into an asymptotic approximation of its coefficients.” We refer the reader to Chapter VI of the magnum opus of Flajolet and Sedgewick [FO90] for a full description of singularity analysis of generating functions. The particular Transfer Lemma that we have used in our work is the one below. It is written in terms of asymptotic equivalence [Bru58]:

$$f(x) \sim g(x) \iff \lim_{x \rightarrow \infty} \frac{f(x)}{g(x)} = 1.$$

The asymptotic equivalence can be written in terms of Little-O notation:

$$f(x) \sim g(x) \iff f(x) = g(x)(1 + o(1)) = g(x) + o(g(x)).$$

Lemma A.3 (Flajolet and Odlyzko, Corollary 2 of [FO90], Corollary VI.1 of [FS09]). *Assume that $f(z)$ is analytic in $\Delta \setminus \{1\}$, and that as $z \rightarrow 1$ in Δ ,*

$$f(z) \sim K(1 - z)^\alpha,$$

with $\alpha \notin \{0, 1, 2, \dots\}$. Then, the Taylor coefficients of f satisfy

$$f_n \sim \frac{K}{\Gamma(-\alpha)} n^{-\alpha-1}.$$

Appendix B

Proofs

B.1 Random PM queries with extreme coordinates in K -dt trees

In this section we are going to prove [Theorem 6.1](#) and [Theorem 6.2](#), so we restate them here.

Theorem B.1. ([Theorem 6.1](#)). *The expected cost $\hat{P}_{n,t,\rho,\rho_0}$ of a random PM query—where exactly s , $0 < s < K$, coordinates are specified and exactly s_0 , $0 \leq s_0 \leq s$, of the specified coordinates are extreme—in a random relaxed K -dt tree of size n satisfies*

$$\hat{P}_{n,t,\rho,\rho_0} = \beta_{t,\rho,\rho_0} n^{\alpha_{t,\rho,\rho_0}} + o(n^{\alpha_{t,\rho,\rho_0}}),$$

where both β_{t,ρ,ρ_0} and α_{t,ρ,ρ_0} depend on t and the ratios $\rho = s/K$ and $\rho_0 = s_0/K$. The exponent α_{t,ρ,ρ_0} is the unique solution in the interval $(0, 1)$ of the equation:

$$\rho_0 \frac{(t+2)^{\overline{t+1}}}{2(\alpha+t+1)^{\overline{t+1}}} + (\rho - \rho_0) \frac{(t+2)^{\overline{t+1}}}{(\alpha+t+2)^{\overline{t+1}}} + (1-\rho) \frac{(t+2)^{\overline{t+1}}}{(\alpha+t+1)^{\overline{t+1}}} = 1. \quad (\text{B.1})$$

Theorem B.2. ([Theorem 6.2](#)). *The expected cost $\hat{P}_{n,t,\mathbf{u}}$ of a random PM query with pattern \mathbf{u} —where exactly s , $0 < s < K$, coordinates are specified and exactly s_0 , $0 \leq s_0 \leq s$, of the specified coordinates are extreme—in a random standard K -dt tree of size n satisfies*

$$\hat{P}_{n,t,\mathbf{u}} = \beta_{t,\mathbf{u}} n^{\alpha_{t,\rho,\rho_0}} + o(n^{\alpha_{t,\rho,\rho_0}}),$$

where $\beta_{t,\mathbf{u}}$ depends on t and the query pattern \mathbf{u} and α_{t,ρ,ρ_0} depends on t and the ratios $\rho = s/K$ and $\rho_0 = s_0/K$ due to it being the unique solution in the interval $(0, 1)$ of the equation:

$$\left(\frac{(t+2)^{\overline{t+1}}}{2(\alpha+t+1)^{\overline{t+1}}} \right)^{\rho_0} \left(\frac{(t+2)^{\overline{t+1}}}{(\alpha+t+2)^{\overline{t+1}}} \right)^{\rho-\rho_0} \left(\frac{(t+2)^{\overline{t+1}}}{(\alpha+t+1)^{\overline{t+1}}} \right)^{1-\rho} = 1. \quad (\text{B.2})$$

We prove both theorems using the method developed in [Lau88; CLF89] which follows classic asymptotic analysis methods [Bru58]. On a side note, the first step used there was to establish recurrences for the probability generating functions of the cost of the randomized partial match algorithm. To obtain recurrences for the first two moments those recurrences are differentiated and evaluated at $z = 1$.

Here we follow the second step used in [Lau88; CLF89], that is solve asymptotically the recurrence for the expected value. To achieve that the binomial coefficients that appear as coefficients are approximated by polynomials and lower order terms are discarded to obtain a system of asymptotic equations. Then the sequences—discrete functions—are extended to infinitely differentiable functions on \mathbb{R} and such system is translated to a system of Euler-Cauchy differential equations. The characteristic equation of this system gives the equation for α . Some solutions of the system are discarded because they would imply cost functions that are non-increasing. Finally, it is verified that the candidate solution is indeed a solution of the system of asymptotic equations. Here we omit all those details explained in [Lau88] and just give a sketch of the proof.

Given that the proofs of the theorems above have many parts in common we are going to show them together.

Proof. Sketch. Following [DL17] we define $w_{t,n,j}$ as the probability that in a random K -dt tree of size n the left subtree has size j , $0 \leq j \leq n - 1$. As in [CLF89], we have:

$$w_{t,n,j} = \begin{cases} \frac{1}{n} & \text{if } n < 2t + 1 \\ \frac{\binom{j}{t} \binom{n-1-j}{t}}{\binom{n}{2t+1}} & \text{if } n \geq 2t + 1 \end{cases} \quad (\text{B.3})$$

with the symmetry $w_{t,n,j} = w_{t,n,n-1-j}$.

Given the fixed rule to assign discriminants to nodes in standard K -dt trees we need to introduce some notation. We will abbreviate $(i + 1) \bmod K$ as $i \oplus 1$. Let $\mathbf{u}^{(i)}$ be \mathbf{u} shifted to the left i times, in particular we have $\mathbf{u}^{(K)} = \mathbf{u}^{(0)} = \mathbf{u}$. Let $g_i(n) := \hat{P}_{n,t,\mathbf{u}^{(i)}}^{(i)}$ be the expected cost of a PM query with pattern $\mathbf{u}^{(i)}$ in a random K -dt tree of size n where the root discriminates by the i coordinate and where s_0 specified coordinates of the query are extreme, $0 \leq s_0 \leq s$, and the remaining $s - s_0$ specified coordinates are randomly drawn from $(0, 1)$. Without loss of generality we assume that the extreme specified coordinates have value 0.

The first step is to set up a system of recurrences for $g_i(n)$. If $u_i = *$ the search has to continue in both subtrees. If $u_i = E$ the search has to continue only in one subtree, given our assumption that the extreme specified coordinates are 0 it is on the left subtree. If $u_i = S$ the search continues on the left subtree with probability $(j + 1)/(n + 1)$ and on the right subtree with probability

$(n - j)/(n + 1)$. Therefore:

$$g_i(n) = 1 + \sum_{j=0}^{n-1} w_{t,n,j} \left(g_{i\oplus 1}(j) + g_{i\oplus 1}(n - j - 1) \right) \quad \text{if } u_i = *$$

$$g_i(n) = 1 + \sum_{j=0}^{n-1} w_{t,n,j} g_{i\oplus 1}(j) \quad \text{if } u_i = E$$

$$g_i(n) = 1 + \sum_{j=0}^{n-1} w_{t,n,j} \left(\frac{j+1}{n+1} g_{i\oplus 1}(j) + \frac{n-j}{n+1} g_{i\oplus 1}(n - j - 1) \right) \quad \text{if } u_i = S$$

Taking into account the symmetry $w_{t,n,j} = w_{t,n,n-1-j}$:

$$g_i(n) = 1 + 2 \sum_{j=0}^{n-1} w_{t,n,j} g_{i\oplus 1}(j) \quad \text{if } u_i = *$$

$$g_i(n) = 1 + \sum_{j=0}^{n-1} w_{t,n,j} g_{i\oplus 1}(j) \quad \text{if } u_i = E$$

$$g_i(n) = 1 + 2 \sum_{j=0}^{n-1} w_{t,n,j} \frac{j+1}{n+1} g_{i\oplus 1}(j) \quad \text{if } u_i = S$$

In the case of relaxed K -dt trees instead of this system of recurrences we have a single one because the discriminant coordinate is chosen randomly in each node, so the probability of $u_i = *$ is $(K - s)/K$, the probability of $u_i = E$ is s_0/K and the probability of $u_i = S$ is $(s - s_0)/K$, so the recurrence is:

$$\begin{aligned} g_i(n) &= 1 + 2 \frac{K - s}{K} \sum_{j=0}^{n-1} w_{t,n,j} g_{i\oplus 1}(j) \\ &\quad + \frac{s_0}{K} \sum_{j=0}^{n-1} w_{t,n,j} g_{i\oplus 1}(j) \\ &\quad + 2 \frac{s - s_0}{K} \sum_{j=0}^{n-1} w_{t,n,j} \frac{j+1}{n+1} g_{i\oplus 1}(j). \end{aligned}$$

This is recurrence (4.3) of [DL17].

Following the steps done in [Lau88; CLF89], we obtain that $g_i(x) = \beta_i x^\alpha$ where, for standard K -dt trees:

$$\beta_i = \frac{(t+2)^{\overline{t+1}}}{(\alpha+t+1)^{\overline{t+1}}} \beta_{i+1} \quad \text{if } u_i = *$$

$$\beta_i = \frac{(t+2)^{\overline{t+1}}}{2(\alpha+t+1)^{\overline{t+1}}} \beta_{i+1} \quad \text{if } u_i = E$$

$$\beta_i = \frac{(t+2)^{\overline{t+1}}}{(\alpha+t+2)^{\overline{t+1}}} \beta_{i+1} \quad \text{if } u_i = S,$$

and for relaxed K -dt trees:

$$\beta_i = \left(\frac{K-s}{K} \frac{(t+2)^{\overline{t+1}}}{(\alpha+t+1)^{\overline{t+1}}} + \frac{s_0}{K} \frac{(t+2)^{\overline{t+1}}}{2(\alpha+t+1)^{\overline{t+1}}} + \frac{s-s_0}{K} \frac{(t+2)^{\overline{t+1}}}{(\alpha+t+2)^{\overline{t+1}}} \right) \beta_{i+1}.$$

For standard K -dt trees, using $\rho = s/K$ and $\rho_0 = s_0/K$:

$$\beta_0 = \left(\frac{(t+2)^{\overline{t+1}}}{2(\alpha+t+1)^{\overline{t+1}}} \right)^{\rho_0} \left(\frac{(t+2)^{\overline{t+1}}}{(\alpha+t+2)^{\overline{t+1}}} \right)^{\rho-\rho_0} \left(\frac{(t+2)^{\overline{t+1}}}{(\alpha+t+1)^{\overline{t+1}}} \right)^{1-\rho} \beta_K. \quad (\text{B.4})$$

Due to the cyclical way of choosing the discriminant coordinate in standard K -dt trees we have that $\beta_K = \beta_0$. That constant cannot be zero, so we divide by it and obtain equation (B.2).

For relaxed K -dt trees, using $\rho = s/K$ and $\rho_0 = s_0/K$:

$$\beta_0 = \left(\rho_0 \frac{(t+2)^{\overline{t+1}}}{2(\alpha+t+1)^{\overline{t+1}}} + (\rho - \rho_0) \frac{(t+2)^{\overline{t+1}}}{(\alpha+t+2)^{\overline{t+1}}} + (1 - \rho) \frac{(t+2)^{\overline{t+1}}}{(\alpha+t+1)^{\overline{t+1}}} \right) \beta_1. \quad (\text{B.5})$$

Given that all the nodes choose the discriminant coordinate in the same way in relaxed K -dt trees we have $\beta_1 = \beta_0$. Since it cannot be zero we divide by it and obtain equation (B.1). □

B.2 Lemma for alternative proofs

As mentioned at the end of [Section 3.1.3](#), during the work done to analyze fixed PM queries in [\[DLM16a\]](#) we discovered a lemma that significantly simplifies that analysis because it leads to a proof that does not require knowing the expected cost of random PM queries with extreme specified coordinates, ([Theorem 5](#) of that article).

In [\[DLM16a\]](#) the expected cost of random PM queries with extreme specified coordinates is only used to prove—by mathematical induction—constraint (c) of [Proposition 1](#) (see [page 36](#) of [\[DLM16a\]](#)). Then that constraint is used only to obtain the initial condition $\phi(0) = 0$ (see [page 38](#) of [\[DLM16a\]](#)). Finally, we make use of such initial condition just to apply the same procedure used in [\[DJM14\]](#) to solve the hypergeometric differential equation that is obtained.

[Lemma B.1](#) below allows us to solve the differential equation that appears on [page 38](#) of [\[DLM16a\]](#)—and similar ones in [\[DJM14; DLM18\]](#)—without using the initial condition $\phi(0) = 0$ by taking advantage of the symmetry of constraint (b) of [Proposition 1](#) of [\[DLM16a\]](#) and analogous ones in the other papers. We summarize that constraint below by the condition $\phi(z) = \phi(1 - z)$, which ultimately comes from the symmetry that to search for rank r has the same cost as to search for rank $n - r$.

Lemma B.1. *Given a function $\phi : \mathbb{R} \rightarrow \mathbb{R}$ such that $\phi(z) = \phi(1 - z)$, $\alpha \in (0, 1)$ and $\delta = \frac{\alpha+2}{2}$, the differential equation*

$$z(1 - z)\phi''(z) + (\alpha - \delta)(2z - 1)\phi'(z) - \alpha(\alpha + 1 - 2\delta)\phi(z) = 0 \quad (\text{B.6})$$

has a general solution of the form

$$\phi(z) = \mu (z(1 - z))^{\alpha/2}$$

where $\mu \in \mathbb{R}$ is a constant.

Proof. Equation (B.6) is an homogeneous hypergeometric differential equation [\[AS64\]](#) [\[DLMF, Sec. 15.10\]](#):

$$z(1 - z)\frac{d^2w}{dz^2} + [c - (a + b + 1)z]\frac{dw}{dz} - abw = 0$$

with $a = -\alpha$, $b = 2\delta - \alpha - 1$ and $c = \delta - \alpha$. Since $0 < \alpha < 1$ and $\delta = \frac{\alpha+2}{2}$ we have that c is not an integer, therefore in the neighbourhood of the singular point 0 we have the general solution:

$$\phi(z) = C_1 \cdot {}_2F_1\left(\begin{matrix} -\alpha, 2\delta - \alpha - 1 \\ \delta - \alpha \end{matrix} \middle| z\right) + C_2 \cdot z^{1-\delta+\alpha} \cdot {}_2F_1\left(\begin{matrix} 1 - \delta, \delta \\ \alpha + 2 - \delta \end{matrix} \middle| z\right)$$

where C_1 and C_2 are arbitrary constants and ${}_2F_1\left(\begin{smallmatrix} a, b \\ c \end{smallmatrix} \middle| z\right)$ denotes the standard hypergeometric function [\[AS64; GKP94\]](#) [\[DLMF, Chap. 15\]](#). Using $\delta = \frac{\alpha+2}{2}$ the second term simplifies to:

$$C_2 \cdot z^{\alpha/2} \cdot {}_2F_1\left(\begin{matrix} -\alpha/2, \frac{\alpha+2}{2} \\ \frac{\alpha+2}{2} \end{matrix} \middle| z\right) = C_2 (z(1 - z))^{\alpha/2}$$

where, in the last step, we have used the identity [AS64] [DLMF, Eq. 15.4.6]:

$${}_2F_1\left(\begin{matrix} a, b \\ b \end{matrix} \middle| z\right) = {}_2F_1\left(\begin{matrix} b, a \\ b \end{matrix} \middle| z\right) = (1-z)^{-a} \quad (\text{B.7})$$

Therefore the general solution is:

$$\phi(z) = C_1 \cdot {}_2F_1\left(\begin{matrix} -\alpha, 1 \\ \frac{2-\alpha}{2} \end{matrix} \middle| z\right) + C_2 (z(1-z))^{\alpha/2}$$

Given that $\phi(z) = \phi(1-z)$ and that the second term also has that symmetry it must be the case that the first term has it as well:

$${}_2F_1\left(\begin{matrix} -\alpha, 1 \\ \frac{2-\alpha}{2} \end{matrix} \middle| z\right) = {}_2F_1\left(\begin{matrix} -\alpha, 1 \\ \frac{2-\alpha}{2} \end{matrix} \middle| 1-z\right) \quad (\text{B.8})$$

Given that $0 < \alpha < 1$ we have $\frac{2-\alpha}{2} - (-\alpha) - 1 = \alpha/2 > 0$ and $\frac{2-\alpha}{2} < 1$, therefore we can use the connection between the hypergeometric functions of z and $1-z$ [Bai64] (where the formula below is given subject to the conditions $\Re(c-a-b) > 0$ and $\Re(c) < 1$) [DLMF, Eq. 15.10.21]:

$$\begin{aligned} {}_2F_1\left(\begin{matrix} a, b \\ c \end{matrix} \middle| z\right) &= \frac{\Gamma(c)\Gamma(c-a-b)}{\Gamma(c-a)\Gamma(c-b)} \cdot {}_2F_1\left(\begin{matrix} a, b \\ a+b+1-c \end{matrix} \middle| 1-z\right) \\ &+ \frac{\Gamma(c)\Gamma(a+b-c)}{\Gamma(a)\Gamma(b)} \cdot {}_2F_1\left(\begin{matrix} c-a, c-b \\ 1+c-a-b \end{matrix} \middle| 1-z\right) (1-z)^{c-a-b} \end{aligned}$$

to obtain that

$$\begin{aligned} {}_2F_1\left(\begin{matrix} -\alpha, 1 \\ \frac{2-\alpha}{2} \end{matrix} \middle| z\right) &= \frac{\Gamma(\frac{2-\alpha}{2})\Gamma(\frac{\alpha}{2})}{\Gamma(\frac{\alpha+2}{2})\Gamma(-\frac{\alpha}{2})} \cdot {}_2F_1\left(\begin{matrix} -\alpha, 1 \\ \frac{2-\alpha}{2} \end{matrix} \middle| 1-z\right) \\ &+ \frac{\Gamma(\frac{2-\alpha}{2})\Gamma(-\frac{\alpha}{2})}{\Gamma(-\alpha)\Gamma(1)} \cdot {}_2F_1\left(\begin{matrix} \frac{\alpha+2}{2}, -\frac{\alpha}{2} \\ \frac{\alpha+2}{2} \end{matrix} \middle| 1-z\right) (1-z)^{\alpha/2} \end{aligned}$$

Using (B.7) again and also (B.8):

$$\begin{aligned} {}_2F_1\left(\begin{matrix} -\alpha, 1 \\ \frac{2-\alpha}{2} \end{matrix} \middle| z\right) &= \frac{\Gamma(\frac{2-\alpha}{2})\Gamma(\frac{\alpha}{2})}{\Gamma(\frac{\alpha+2}{2})\Gamma(-\frac{\alpha}{2})} \cdot {}_2F_1\left(\begin{matrix} -\alpha, 1 \\ \frac{2-\alpha}{2} \end{matrix} \middle| z\right) \\ &+ \frac{\Gamma(\frac{2-\alpha}{2})\Gamma(-\frac{\alpha}{2})}{\Gamma(-\alpha)\Gamma(1)} \cdot (z(1-z))^{\alpha/2} \end{aligned}$$

Therefore:

$${}_2F_1\left(\begin{matrix} -\alpha, 1 \\ \frac{2-\alpha}{2} \end{matrix} \middle| z\right) = C_3 \cdot (z(1-z))^{\alpha/2}$$

where

$$C_3 = \frac{\Gamma(\frac{\alpha+2}{2})\Gamma(\frac{2-\alpha}{2})\Gamma^2(-\frac{\alpha}{2})}{\left(\Gamma(\frac{\alpha+2}{2})\Gamma(-\frac{\alpha}{2}) - \Gamma(\frac{2-\alpha}{2})\Gamma(\frac{\alpha}{2})\right)\Gamma(-\alpha)\Gamma(1)}$$

Making $\mu = C_1 C_3 + C_2$ we obtain

$$\phi(z) = \mu (z(1-z))^{\alpha/2}.$$

□

Bibliography

- [AS64] Milton Abramowitz and Irene A. Stegun, eds. *Handbook of Mathematical Functions*. New York: Dover Publications, 1964.
- [Ada+09] Andrew Adams, Natasha Gelfand, Jennifer Dolson, and Marc Levoy. “Gaussian KD-trees for Fast High-dimensional Filtering”. In: *ACM Trans. Graph.* 28.3 (July 2009), 21:1–21:12. ISSN: 0730-0301. DOI: [10.1145/1531326.1531327](https://doi.org/10.1145/1531326.1531327).
- [Bai64] W. N. Bailey. *Generalized Hypergeometric Series*. 1935; 2nd ed. Cambridge University Press, 1964.
- [Bel65] C. J. Bell. “An investigation into the principles of the classification and analysis of data on an automatic digital computer”. PhD thesis. Leeds University, 1965.
- [Ben75] Jon Louis Bentley. “Multidimensional Binary Search Trees Used for Associative Searching”. In: *Commun. ACM* 18.9 (1975), pp. 509–517. DOI: [10.1145/361002.361007](https://doi.org/10.1145/361002.361007).
- [Ber+14] Nikolett Bereczky, Amalia Duch, Krisztián Németh, and Salvador Roura. “Quad-K-d Trees”. In: *LATIN 2014: Theoretical Informatics - 11th Latin American Symposium, Montevideo, Uruguay, March 31 - April 4, 2014. Proceedings*. Ed. by Alberto Pardo and Alfredo Viola. Vol. 8392. Lecture Notes in Computer Science. Springer, 2014, pp. 743–754. ISBN: 978-3-642-54422-4. DOI: [10.1007/978-3-642-54423-1.64](https://doi.org/10.1007/978-3-642-54423-1.64).
- [Ber+16] Nikolett Bereczky, Amalia Duch, Krisztián Németh, and Salvador Roura. “Quad-kd trees: A general framework for kd trees and quad trees”. In: *Theor. Comput. Sci.* 616 (2016), pp. 126–140. DOI: [10.1016/j.tcs.2015.12.030](https://doi.org/10.1016/j.tcs.2015.12.030).
- [BNS12] Nicolas Broutin, Ralph Neininger, and Henning Sulzbach. “Partial match queries in random quadtrees”. In: *Proceedings of the Twenty-Third Annual ACM-SIAM Symposium on Discrete Algorithms, SODA 2012, Kyoto, Japan, January 17-19, 2012*. Ed. by Yuval Rabani. SIAM, 2012, pp. 1056–1065. ISBN: 978-1-61197-210-8. DOI: [10.1137/1.9781611973099.83](https://doi.org/10.1137/1.9781611973099.83).

- [BNS13] Nicolas Broutin, Ralph Neininger, and Henning Sulzbach. “A limit process for partial match queries in random quadtrees and 2-d trees”. In: *Annals of Applied Probability* 23.6 (Dec. 2013), pp. 2560–2603. DOI: [10.1214/12-AAP912](https://doi.org/10.1214/12-AAP912).
- [Bru58] N. G. de Bruijn. *Asymptotic Methods in Analysis*. North-Holland, 1958.
- [CDZ01] Philippe Chanzy, Luc Devroye, and Carlos Zamora-Cura. “Analysis of range search for random k -d trees”. In: *Acta Informatica* 37.4/5 (2001), pp. 355–383. DOI: [10.1007/s002360000044](https://doi.org/10.1007/s002360000044).
- [CH03] Hua-Huai Chern and Hsien-Kuei Hwang. “Partial Match Queries in Random Quadtrees”. In: *SIAM Journal on Computing* 32.4 (2003), pp. 904–915. DOI: [10.1137/S0097539702412131](https://doi.org/10.1137/S0097539702412131).
- [CH06] Hua-Huai Chern and Hsien-Kuei Hwang. “Partial Match Queries in Random k -d Trees”. In: *SIAM Journal on Computing* 35.6 (2006), pp. 1440–1466. DOI: [10.1137/S0097539703437491](https://doi.org/10.1137/S0097539703437491).
- [CLF89] Walter Cunto, Gustavo Lau, and Philippe Flajolet. “Analysis of KDT-Trees: KD-Trees Improved by Local Reorganisations”. In: *Algorithms and Data Structures, Workshop WADS '89, Ottawa, Canada, August 17-19, 1989, Proceedings*. Ed. by Frank K. H. A. Dehne, Jörg-Rüdiger Sack, and Nicola Santoro. Vol. 382. Lecture Notes in Computer Science. Springer, 1989, pp. 24–38. ISBN: 3-540-51542-9. DOI: [10.1007/3-540-51542-9_4](https://doi.org/10.1007/3-540-51542-9_4).
- [CJ11] Nicolas Curien and Adrien Joseph. “Partial match queries in two-dimensional quadtrees: a probabilistic approach”. In: *Advances in Applied Probability* 43.1 (2011), pp. 178–194. DOI: [10.1239/aap/1300198518](https://doi.org/10.1239/aap/1300198518).
- [Dev93] Luc Devroye. “On the Expected Height of Fringe-Balanced Trees”. In: *Acta Informatica* 30.5 (1993), pp. 459–466. DOI: [10.1007/BF01210596](https://doi.org/10.1007/BF01210596).
- [DJZ00] Luc Devroye, Jean Jabbour, and Carlos Zamora-Cura. “Squarish k -d Trees”. In: *SIAM Journal on Computing* 30.5 (2000), pp. 1678–1700. DOI: [10.1137/S0097539799358926](https://doi.org/10.1137/S0097539799358926).
- [Duc04] Amalia Duch. “Design and Analysis of Multidimensional Data Structures”. PhD thesis. Universitat Politècnica de Catalunya, 2004.
- [DEM98] Amalia Duch, Vladimir Estivill-Castro, and Conrado Martínez. “Randomized K -Dimensional Binary Search Trees”. In: *Algorithms and Computation, 9th International Symposium, ISAAC '98, Taejeon, Korea, December 14-16, 1998, Proceedings*. 1998, pp. 199–208. DOI: [10.1007/3-540-49381-6_22](https://doi.org/10.1007/3-540-49381-6_22).
- [DJM14] Amalia Duch, Rosa M. Jiménez, and Conrado Martínez. “Selection by rank in K -dimensional binary search trees”. In: *Random Struct. Algorithms* 45.1 (2014), pp. 14–37. DOI: [10.1002/rsa.20476](https://doi.org/10.1002/rsa.20476).

- [DL17] Amalia Duch and Gustavo Lau. “Partial Match Queries in Relaxed K -dt trees”. In: *Proc. of the Fourteenth ACM-SIAM Workshop on Analytic Algorithmics and Combinatorics (ANALCO)*. 2017, pp. 131–138. DOI: [10.1137/1.9781611974775.13](https://doi.org/10.1137/1.9781611974775.13).
- [DLM14] Amalia Duch, Gustavo Lau, and Conrado Martínez. “On the Average Performance of Fixed Partial Match Queries in Random Relaxed K -d Trees”. In: *25th International Conference on Probabilistic, Combinatorial and Asymptotic Methods for the Analysis of Algorithms (AofA 2014)*. Ed. by M. Bousquet-Mélou and M. Soria. Discrete Mathematics & Theoretical Computer Science (Proceedings). 2014, pp. 97–108. URL: <https://hal.inria.fr/hal-01077251v2/document#page=106>.
- [DLM16a] Amalia Duch, Gustavo Lau, and Conrado Martínez. “On the Cost of Fixed Partial Match Queries in K -d Trees”. In: *Algorithmica* 75.4 (2016), pp. 684–723. DOI: [10.1007/s00453-015-0097-4](https://doi.org/10.1007/s00453-015-0097-4).
- [DLM16b] Amalia Duch, Gustavo Lau, and Conrado Martínez. “Random Partial Match in Quad- K -d Trees”. In: *LATIN 2016: Theoretical Informatics - 12th Latin American Symposium, Ensenada, Mexico, April 11-15, 2016, Proceedings*. 2016, pp. 376–389. DOI: [10.1007/978-3-662-49529-2_28](https://doi.org/10.1007/978-3-662-49529-2_28).
- [DLM18] Amalia Duch, Gustavo Lau, and Conrado Martínez. “Fixed Partial Match Queries in Quadtrees”. In: *29th International Conference on Probabilistic, Combinatorial and Asymptotic Methods for the Analysis of Algorithms (AofA 2018)*. Ed. by J. A. Fill and M. D. Ward. Vol. 110. Leibniz International Proceedings in Informatics (LIPIcs). Dagstuhl, Germany: Schloss Dagstuhl–Leibniz-Zentrum fuer Informatik, 2018, 20:1–20:18. ISBN: 978-3-95977-078-1. DOI: [10.4230/LIPIcs.AofA.2018.20](https://doi.org/10.4230/LIPIcs.AofA.2018.20).
- [DM02] Amalia Duch and Conrado Martínez. “On the average performance of orthogonal range search in multidimensional data structures”. In: *J. Algorithms* 44.1 (2002), pp. 226–245. DOI: [10.1016/S0196-6774\(02\)00213-4](https://doi.org/10.1016/S0196-6774(02)00213-4).
- [Fel68] William Feller. *An Introduction to Probability Theory and its Applications, Volume I*. New York, NY: John Wiley & Sons, 1968.
- [FB74] Raphael A. Finkel and Jon Louis Bentley. “Quad Trees: A Data Structure for Retrieval on Composite Keys”. In: *Acta Inf.* 4 (1974), pp. 1–9. DOI: [10.1007/BF00288933](https://doi.org/10.1007/BF00288933).
- [Fla+93] Philippe Flajolet, Gaston H. Gonnet, Claude Puech, and J. M. Robson. “Analytic Variations on Quadrees”. In: *Algorithmica* 10.6 (1993), pp. 473–500. DOI: [10.1007/BF01891833](https://doi.org/10.1007/BF01891833).
- [FO90] Philippe Flajolet and Andrew M. Odlyzko. “Singularity Analysis of Generating Functions”. In: *SIAM J. Discrete Math.* 3.2 (1990), pp. 216–240. DOI: [10.1137/0403019](https://doi.org/10.1137/0403019).

- [FP86] Philippe Flajolet and Claude Puech. “Partial match retrieval of multidimensional data”. In: *J. ACM* 33.2 (1986), pp. 371–407. DOI: [10.1145/5383.5453](https://doi.org/10.1145/5383.5453).
- [FS09] Philippe Flajolet and Robert Sedgewick. *Analytic Combinatorics*. Cambridge University Press, 2009.
- [FS13] Philippe Flajolet and Robert Sedgewick. *An Introduction to the Analysis of Algorithms*. 2nd ed. Addison-Wesley, 2013.
- [Fol95] Gerald B. Folland. *Introduction to Partial Differential Equations*. 2nd ed. Princeton University Press, 1995.
- [GKP94] Ronald L. Graham, Donald E. Knuth, and Oren Patashnik. *Concrete Mathematics*. 2nd ed. Reading, Mass.: Addison-Wesley, 1994.
- [Hoa62] C. A. R. Hoare. “Quicksort”. In: *Comput. J.* 5.1 (1962), pp. 10–15. DOI: [10.1093/comjnl/5.1.10](https://doi.org/10.1093/comjnl/5.1.10).
- [JKK92] Norman L. Johnson, Adrienne W. Kemp, and Samuel Kotz. *Univariate Discrete Distributions*. 2nd ed. New York, NY: John Wiley & Sons, 1992.
- [KH07] Ahmed Khamayseh and Glen Hansen. “Use of the spatial kD-tree in computational physics applications”. In: 2 (June 2007).
- [Knu73] Donald E. Knuth. *The Art of Computer Programming: Sorting and Searching*. 1st ed. Vol. 3. Addison-Wesley, 1973.
- [Knu97] Donald E. Knuth. *The Art of Computer Programming: Fundamental Algorithms*. 3rd ed. Vol. 1. Addison-Wesley, 1997.
- [Knu98] Donald E. Knuth. *The Art of Computer Programming: Sorting and Searching*. 2nd ed. Vol. 3. Addison-Wesley, 1998.
- [Lau88] Gustavo Lau. “Árboles multidimensionales de búsqueda binaria con reorganización local”. Available in gustavolau.com. MA thesis. Universidad Simón Bolívar, Caracas, Venezuela, 1988.
- [LVV03] Aristidis Likas, Nikos A. Vlassis, and Jakob J. Verbeek. “The global k-means clustering algorithm”. In: *Pattern Recognition* 36 (2003), pp. 451–461.
- [Mac80] George W. Mackey. “Harmonic analysis as the exploitation of symmetry—a historical survey”. In: *Bull. Amer. Math. Soc. (N.S.)* 3 (July 1980), pp. 543–698. URL: <https://projecteuclid.org/443/euclid.bams/1183546470>.
- [Mah92] Hosam M. Mahmoud. *Evolution of Random Search Trees*. John Wiley & Sons, 1992.
- [MPP01] Conrado Martínez, Alois Panholzer, and Helmut Prodinger. “Partial Match Queries in Relaxed Multidimensional Search Trees”. In: *Algorithmica* 29.1 (2001), pp. 181–204. DOI: [10.1007/BF02679618](https://doi.org/10.1007/BF02679618).

- [Mea80] Donald J. R. Meagher. *Octree Encoding: A New Technique for the Representation, Manipulation and Display of Arbitrary 3-D Objects by Computer*. Tech. rep. Rensselaer Polytechnic Institute, 1980.
- [MS18] Dinesh P. Mehta and Sartaj Sahni, eds. *Handbook of Data Structures and Applications*. 2nd ed. CRC Press, 2018.
- [Nei99] Ralph Neininger. “Limit laws for random recursive structures and algorithms”. PhD thesis. University of Freiburg, Freiburg im Breisgau, Germany, 1999. URL: <http://www.freidok.uni-freiburg.de/volltexte/13/pdf/13%5C.1.pdf>.
- [Nei00] Ralph Neininger. “Asymptotic distributions for partial match queries in K-d trees”. In: *Random Struct. Algorithms* 17.3-4 (2000), pp. 403–427. DOI: [10.1002/1098-2418\(200010/12\)17:3/4\(403::AID-RSA11\)3.0.CO;2-K](https://doi.org/10.1002/1098-2418(200010/12)17:3/4(403::AID-RSA11)3.0.CO;2-K).
- [DLMF] *NIST Digital Library of Mathematical Functions*. <http://dlmf.nist.gov/>, Release 1.0.23 of 2019-06-15. F. W. J. Olver, A. B. Olde Daalhuis, D. W. Lozier, B. I. Schneider, R. F. Boisvert, C. W. Clark, B. R. Miller and B. V. Saunders, eds. URL: <http://dlmf.nist.gov/>.
- [Ore82] Jack A. Orenstein. “Multidimensional Tries Used for Associative Searching”. In: *Inf. Process. Lett.* 14.4 (1982), pp. 150–157. DOI: [10.1016/0020-0190\(82\)90027-8](https://doi.org/10.1016/0020-0190(82)90027-8).
- [PM85] Patricio V. Poblete and J. Ian Munro. “The Analysis of a Fringe Heuristic for Binary Search Trees”. In: *J. Algorithms* 6.3 (1985), pp. 336–350. DOI: [10.1016/0196-6774\(85\)90003-3](https://doi.org/10.1016/0196-6774(85)90003-3).
- [Rös91] Uwe Röslér. “A limit theorem for ”Quicksort””. In: *ITA* 25 (1991), pp. 85–100. DOI: [10.1051/ita/1991250100851](https://doi.org/10.1051/ita/1991250100851).
- [Rou19] Tim Roughgarden. “Beyond worst-case analysis”. In: *Commun. ACM* 62.3 (2019), pp. 88–96. DOI: [10.1145/3232535](https://doi.org/10.1145/3232535).
- [Rou97] Salvador Roura. “Divide-and-Conquer Algorithms and Data Structures”. PhD thesis. Universitat Politècnica de Catalunya, 1997.
- [Rou01] Salvador Roura. “Improved master theorems for divide-and-conquer recurrences”. In: *J. ACM* 48.2 (2001), pp. 170–205. DOI: [10.1145/375827.375837](https://doi.org/10.1145/375827.375837).
- [Sam90] Hanan Samet. *The Design and Analysis of Spatial Data Structures*. Addison-Wesley, 1990.
- [Sam06] Hanan Samet. *Foundations of Multidimensional and Metric Data Structures*. Morgan & Kaufman Publ., 2006.
- [Sea49] H. L. Seal. “The historical development of the use of generating functions in probability theory”. In: *Bulletin de l’Association des Actuairees Suisses* 49 (1949), pp. 209–228.
- [Sin69] Richard C. Singleton. “Algorithm 347: an efficient algorithm for sorting with minimal storage [M1]”. In: *Commun. ACM* 12.3 (1969), pp. 185–187. DOI: [10.1145/362875.362901](https://doi.org/10.1145/362875.362901).

- [Sul12] Henning Sulzbach. “On a Functional Contraction Method”. PhD thesis. Goethe Universität Frankfurt am Main, 2012.
- [Tre04] Christopher Tremblay. *Mathematics for Game Developers*. Gale virtual reference library. Thomson Course Technology/Premier Press, 2004. ISBN: 9781592000388.
- [WW76] A. Walker and Derick Wood. “Locally Balanced Binary Trees”. In: *Comput. J.* 19.4 (1976), pp. 322–325. DOI: [10.1093/comjnl/19.4.322](https://doi.org/10.1093/comjnl/19.4.322).
- [Wil90] Herbert S. Wilf. *generatingfunctionology*. Academic Press, 1990.
- [Zam00] Carlos Zamora-Cura. “Analysis of Random Trees”. PhD thesis. McGill University, 2000.
- [Zwi97] Daniel Zwillinger. *Handbook of Differential Equations*. 3rd ed. Academic Press, 1997.