Rank Selection in Multidimensional Data

Amalia Duch, Rosa M. Jiménez, Conrado Martínez
Univ. Politècnica Catalunya

AofA 2011, Będlewo, Poland

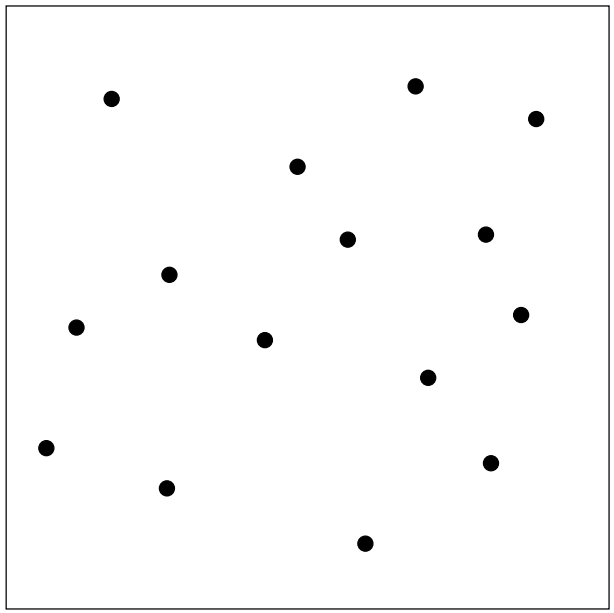
Joint work with:

Amalia Duch

Rosa M. Jiménez

Introduction

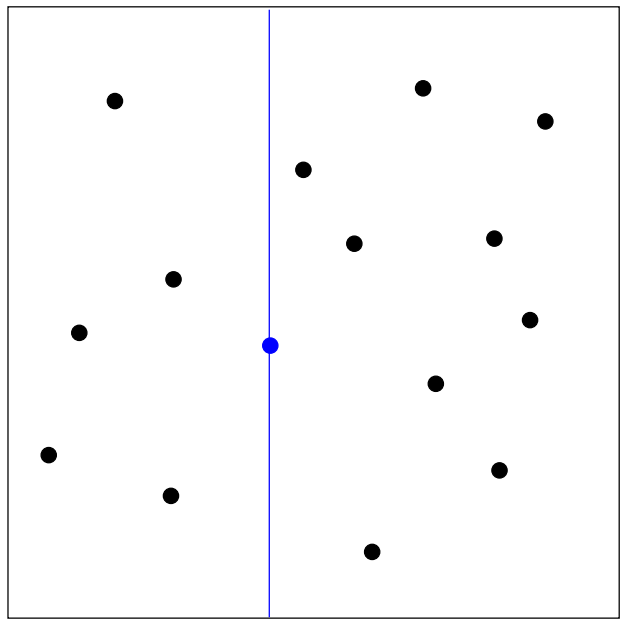
The problem: Given a collection of n multidimensional records, each with K coordinates, and values i, $1 \leqslant i \leqslant n$, and j, $1 \leqslant j \leqslant K$, find the i-th record along the j-th coordinate



n = 15
K = 2

Introduction

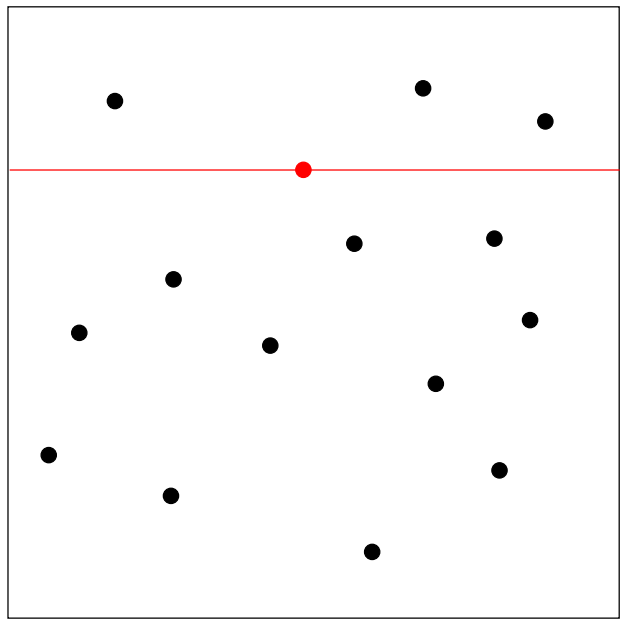
The problem: Given a collection of n multidimensional records, each with K coordinates, and values i, $1 \leqslant i \leqslant n$, and j, $1 \leqslant j \leqslant K$, find the i-th record along the j-th coordinate



$n = 15$ $i = 6$
$K = 2$ $j = 1$

Introduction

The problem: Given a collection of n multidimensional records, each with K coordinates, and values i, $1 \leqslant i \leqslant n$, and j, $1 \leqslant j \leqslant K$, find the i-th record along the j-th coordinate



$n = 15$ $i = 12$
$K = 2$ $j = 2$

Introduction

- What if the collection is organized in some multidimensional index? (e.g., a K-d tree, a quadtree, ...)
- If $K = 1$ and the collection of n records is stored in some kind of binary search tree $\Rightarrow$ (expected) time $\Theta(\log n)$, using some little extra space
- We look for an algorithm that uses space $\Theta(n)$, independent of K
- The data structure for the n records should efficiently support usual spatial queries, e.g., orthogonal range search
- We assume w.l.o.g. the n records are points from $[0, 1]^K$

K-d trees

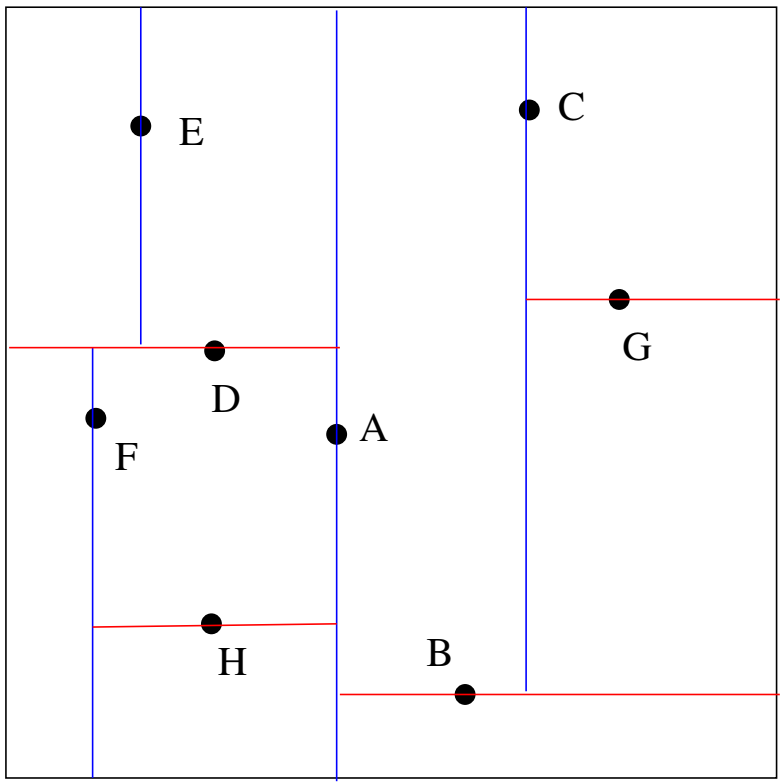
J.L. Bentley

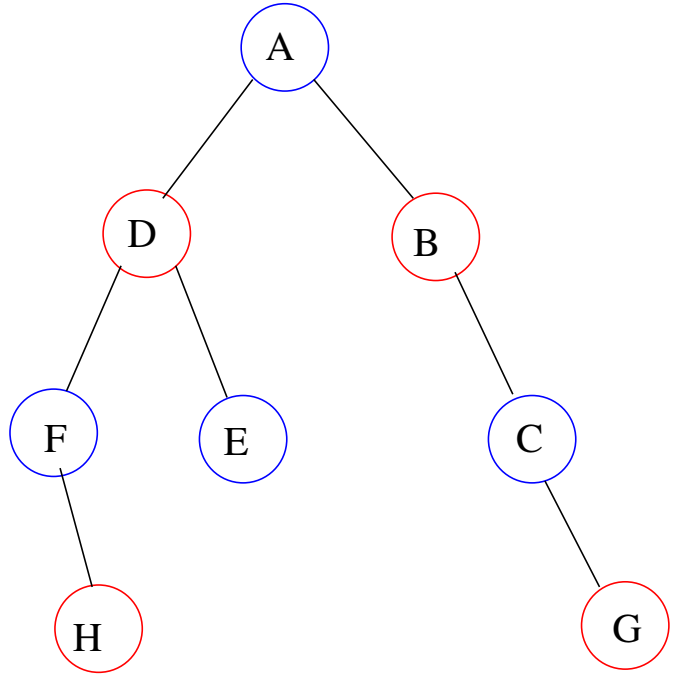
Definition

A K-d tree for a set $X \subset [0,1]^K$ is either the empty tree if $X = \emptyset$ or a binary tree where:

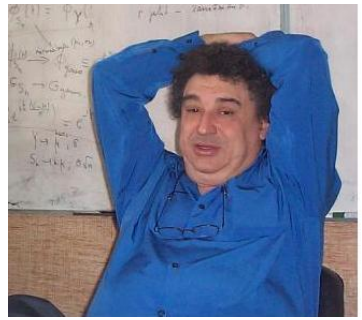
- the root contains $y \in X$ and some value j, $1 \leqslant j \leqslant K$
- the left subtree is a K-d tree for $X^- = \{x \in X \,|\, x_j < y_j\}$
- the right subtree is a K-d tree for $X^+ = \{x \in X \,|\, y_j < x_j\}$

K-d trees





K-d trees



Ph. Flajolet

C. Puech

- In a partial match query we are given a query $q = (q_1, \ldots, q_K)$ where s coordinates are specified and $K - s$ are "don't cares"
- The goal is to find all records in a collection that satisfy the query
- Flajolet and Puech (1986) showed that a partial match in a random standard K-d tree of size n has expected cost $\Theta(n^{\alpha(s/K)})$, where $\alpha(x) = 1 - x + \phi(x)$, $0 \leqslant \phi(x) < 0.07$
- Similar results have been proved for other variants of K-d trees, quadtrees, etc.

K-d trees

L. Devroye

- Orthogonal range queries ask for all records falling inside an hyperrectangle (with sides parallel to the axis); their expected cost has been analyzed by Chanzy, Devroye and Zamora-Cura (2001) and Duch and Martínez (2002):

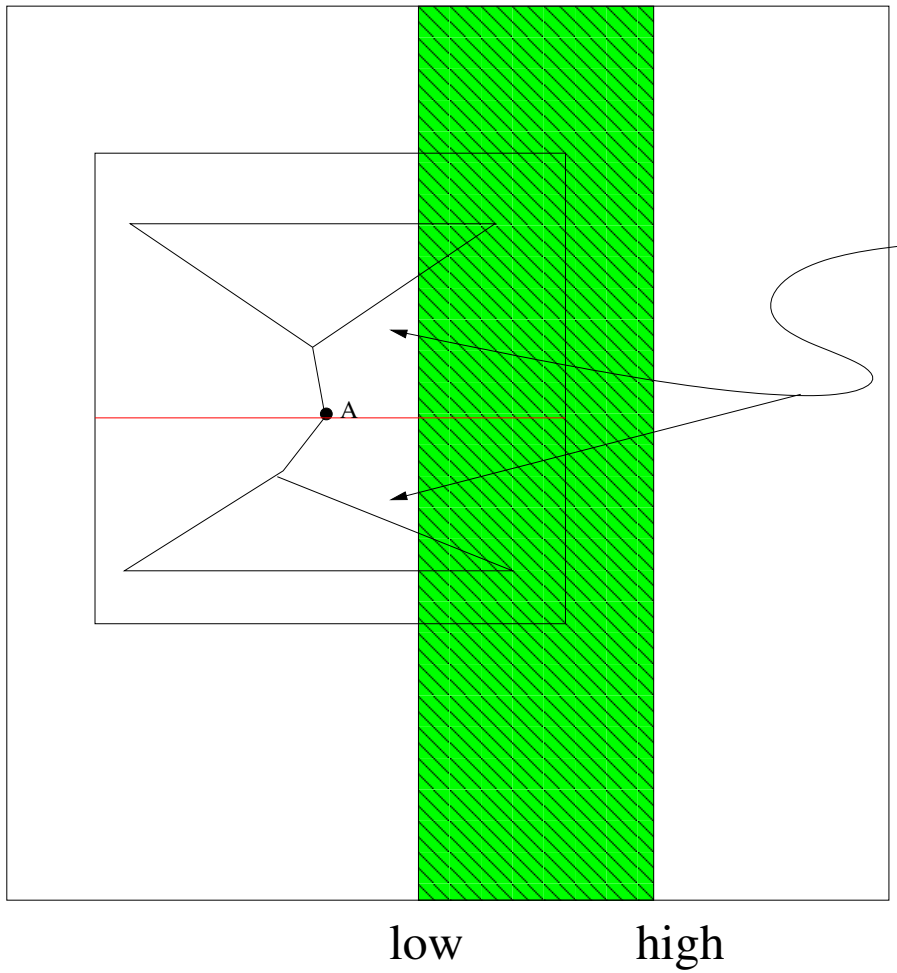
$$n \cdot \text{volume of query} + n^{\alpha(1/K)} \cdot \text{perimeter of query} + \text{l.o.t.}$$

The algorithm

Our algorithm has three main steps

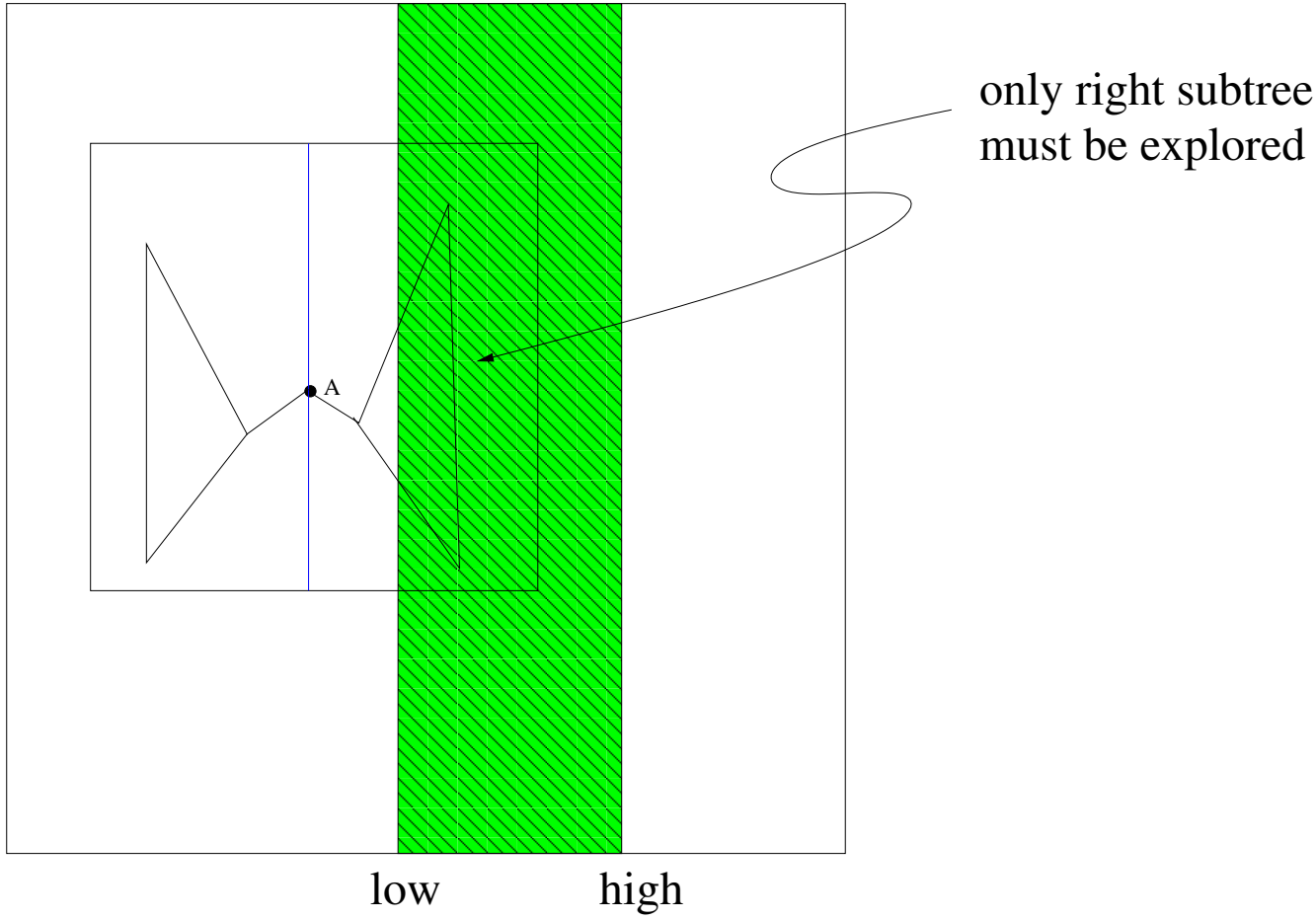
- The main loop starts with a strip $x_j \in [low, high] = [0, 1]$ and explores the K-d tree, reducing the strip in such a way that it always contains the i-th record along coordinate j
- When the main loop finishes, it has found the sought element (if it is stored in a node that discriminates w.r.t. j) or the strip does only contain nodes discriminating w.r.t. a coordinate $\neq j$; if needed, the second step performs an orthogonal range search to locate all records within the strip
- A conventional selection algorithm is used to find the sought element among the elements reported in the previous step

The algorithm: main loop

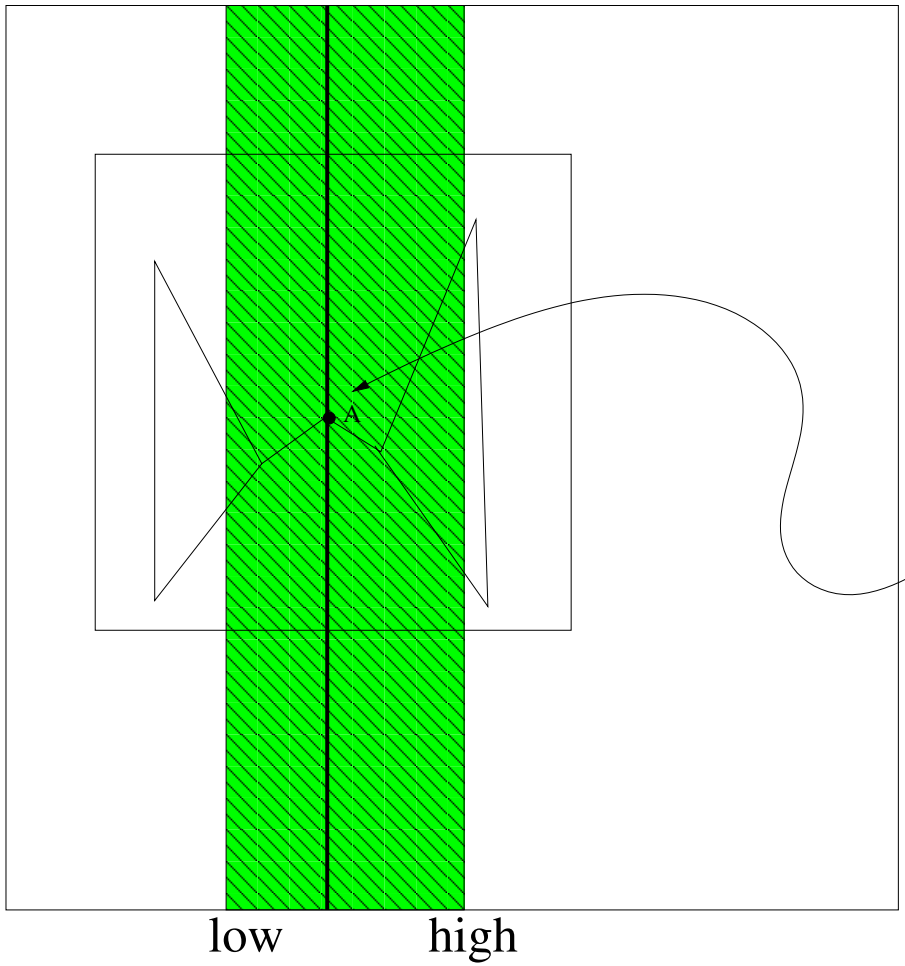


both subtrees
must be explored

The algorithm: main loop

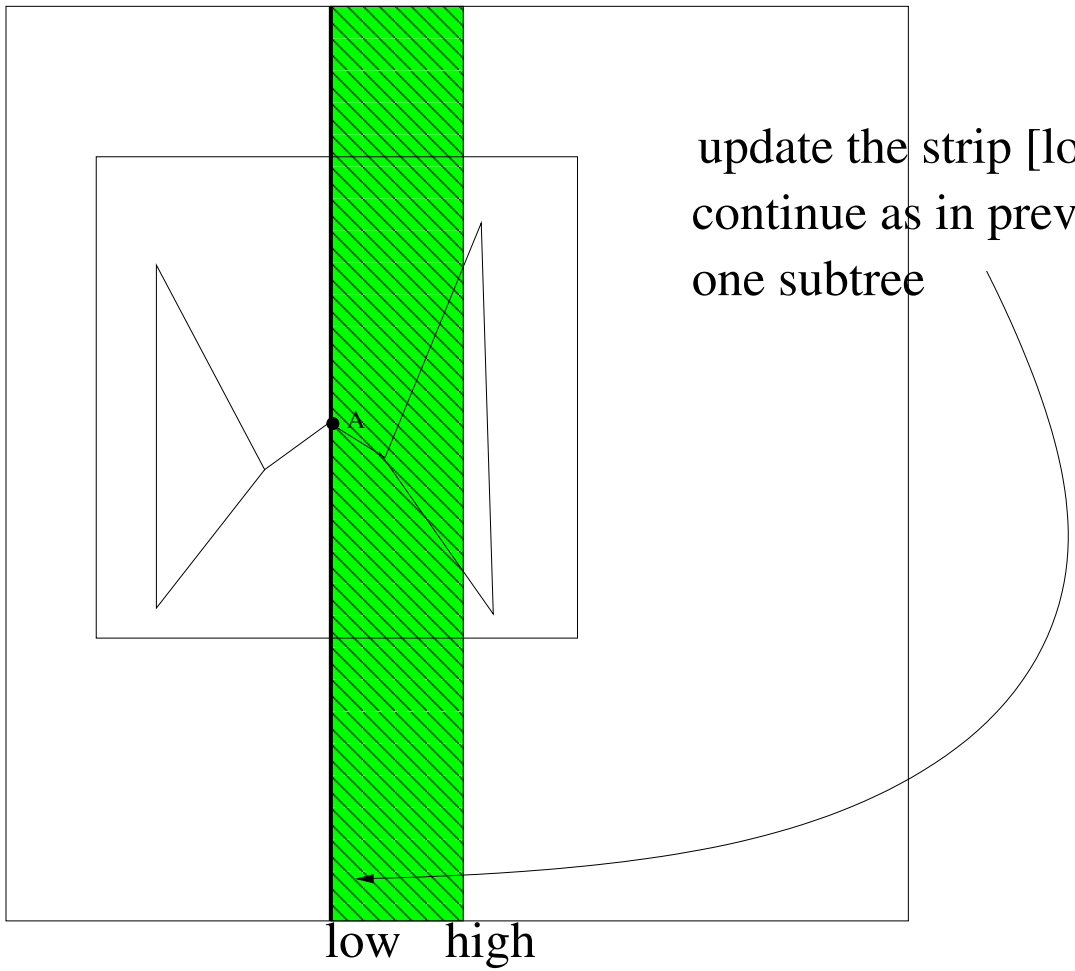


The algorithm: main loop



find the rank of the element
along coordinate j =
count how many
elements are below

The algorithm: main loop



update the strip [low, high] and
continue as in previous case exploring
one subtree

Analysis

Hypothesis for the analysis: The n records are independently drawn from a continuous distribution in $[0, 1]^K$ (standard probability model for random K-d tree)

Analysis

Five key observations

1. The number of visited nodes in the main loop is at most the number of nodes visited by an orthogonal range search with the strip $[low, high]$
2. The cost of a call to BELOW is that of a partial match with a single specified coordinate
3. The expected number of calls to BELOW is $\Theta(\log n)$
4. The main loop finds the sought point when the node discriminates along j-th coordinate or the strip $[low, high]$ contains it and no point that discriminates with respect to j
5. The strip contains $\Theta(1)$ points on average

Analysis

Five key observations

1. The number of visited nodes in the main loop is at most the number of nodes visited by an orthogonal range search with the strip $[low, high]$
2. The cost of a call to BELOW is that of a partial match with a single specified coordinate
3. The expected number of calls to BELOW is $\Theta(\log n)$
4. The main loop finds the sought point when the node discriminates along j-th coordinate or the strip $[low, high]$ contains it and no point that discriminates with respect to j
5. The strip contains $\Theta(1)$ points on average

Analysis

Five key observations

1. The number of visited nodes in the main loop is at most the number of nodes visited by an orthogonal range search with the strip $[low, high]$
2. The cost of a call to BELOW is that of a partial match with a single specified coordinate
3. The expected number of calls to BELOW is $\Theta(\log n)$
4. The main loop finds the sought point when the node discriminates along j-th coordinate or the strip $[low, high]$ contains it and no point that discriminates with respect to j
5. The strip contains $\Theta(1)$ points on average

Analysis

Five key observations

1. The number of visited nodes in the main loop is at most the number of nodes visited by an orthogonal range search with the strip $[low, high]$
2. The cost of a call to BELOW is that of a partial match with a single specified coordinate
3. The expected number of calls to BELOW is $\Theta(\log n)$
4. The main loop finds the sought point when the node discriminates along j-th coordinate or the strip $[low, high]$ contains it and no point that discriminates with respect to j
5. The strip contains $\Theta(1)$ points on average

Analysis

Five key observations

1. The number of visited nodes in the main loop is at most the number of nodes visited by an orthogonal range search with the strip $[low, high]$
2. The cost of a call to BELOW is that of a partial match with a single specified coordinate
3. The expected number of calls to BELOW is $\Theta(\log n)$
4. The main loop finds the sought point when the node discriminates along j-th coordinate or the strip $[low, high]$ contains it and no point that discriminates with respect to j
5. The strip contains $\Theta(1)$ points on average

Analysis

Theorem

The expected cost T_n *of* KDSELECT *to select the* i*-th smallest element along the* j*-th coordinate in a random relaxed* K*-d tree of size* n*, for random uniformly distributed* i *and* j*,* $i \sim$ *Unif*$(1, n)$*,* $j \sim$ *Unif*$(0, K-1)$*, is*

$$T_n = \Theta(n^\alpha \log n),$$

where $\alpha = \alpha(1/K)$ *satisfies* $1/K \leqslant \alpha(K) < 1$ *for all* $K \geqslant 2$*.*

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

Analysis

To achieve a good expected performance for a call to BELOW, it is necessary that each node contains the size of the subtree rooted at that tree

procedure BELOW(T, j, z)
  if T $= \square$ **then return** 0
  if T.$\mathrm{discr} \neq$ j **then**
    $$c \leftarrow \begin{cases} 1 & \textbf{if } T.\mathrm{key}[j] \leqslant z, \\ 0 & \text{otherwise.} \end{cases}$$
    return BELOW$(z, j, T.\mathrm{left})$ + BELOW$(z, j, T.\mathrm{right})$ + c
  else
    if $z <$ T.key[j] **then return** BELOW$(z, j, T.\mathrm{left})$
    else return $T.\mathrm{left}.\mathrm{size}$ + BELOW$(z, j, T.\mathrm{right})$

Analysis: A Refinement

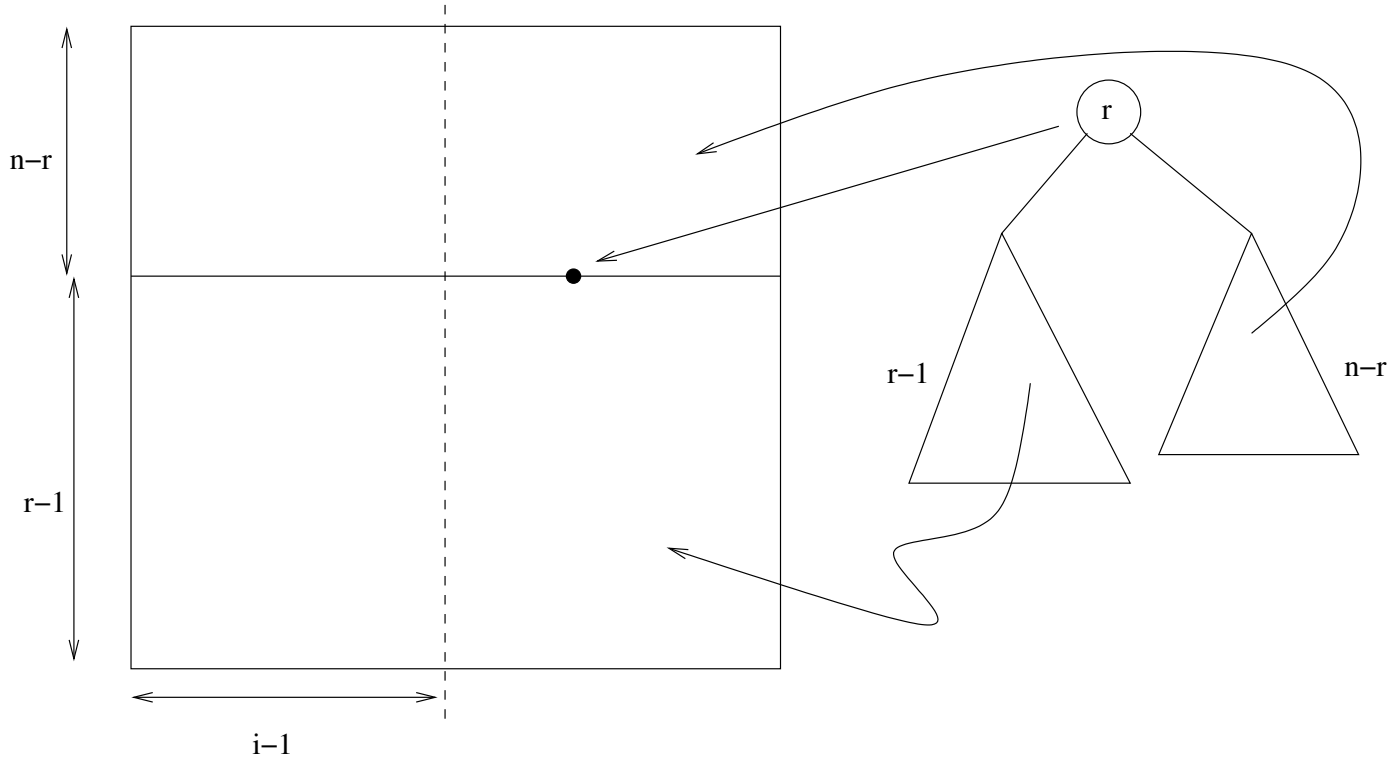
Let $C_{n,i}$ denote the expected cost of a call BELOW(T, j, z) in a random relaxed K-d tree T of size n when exactly $i-1$ of its elements satisfy $x_j \leqslant z$.
Then, for $n > 0$ and $1 \leqslant i \leqslant n+1$

$$C_{n,i} = 1 + \frac{1}{K} \cdot \frac{1}{n} \cdot \left[\sum_{r=1}^{i-1} C_{n-r,i-r} + \sum_{r=i}^{n} C_{r-1,i} \right]$$

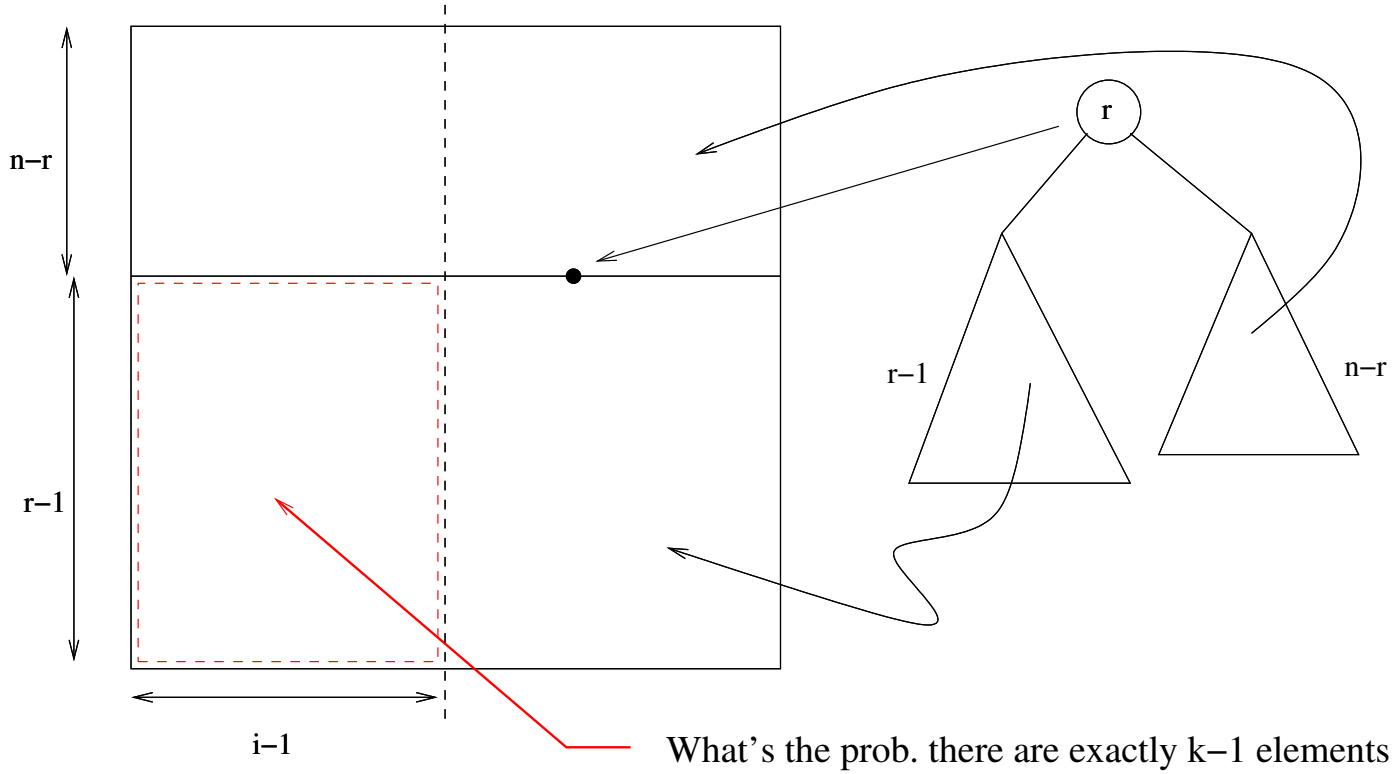
$$+ \frac{K-1}{K} \cdot \frac{1}{n} \left[\sum_{r=1}^{n} \left\{ \frac{n+1-i}{n} \sum_{k} \frac{\binom{r-1}{k-1}\binom{n-r}{i-k}}{\binom{n-1}{i-1}} \cdot (C_{r-1,k} + C_{n-r,i+1-k}) \right.\right.$$

$$\left.\left. + \frac{i-1}{n} \sum_{k} \frac{\binom{r-1}{k-1}\binom{n-r}{i-k-1}}{\binom{n-1}{i-2}} (C_{r-1,k} + C_{n-r,i-k}) \right\} \right].$$

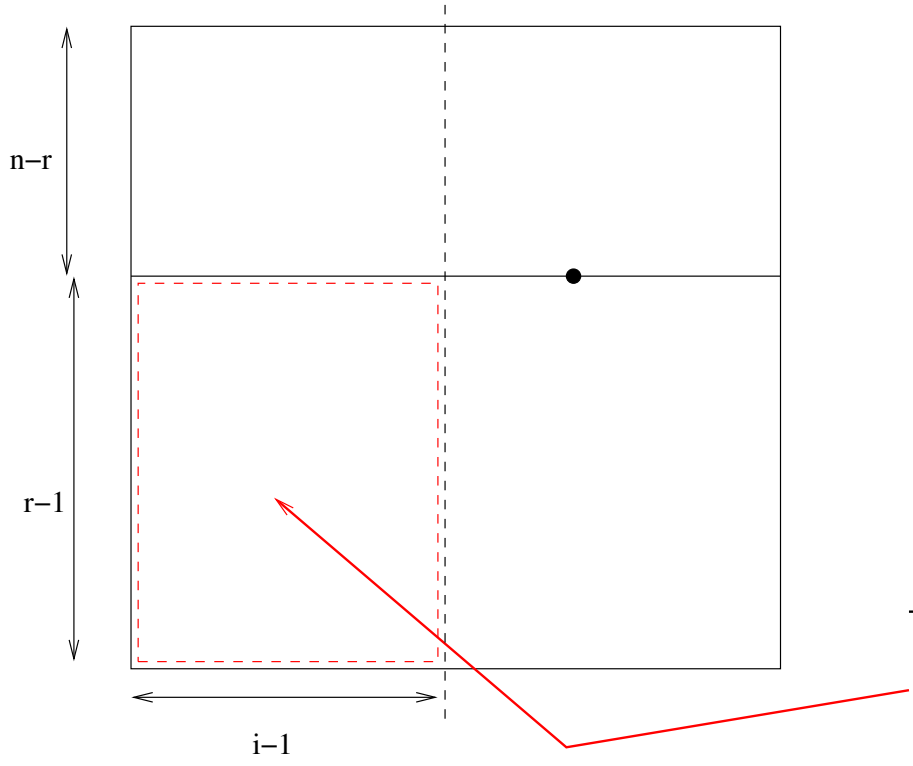
Analysis: A Refinement



Analysis: A Refinement



Analysis: A Refinement



$$\frac{\binom{r-1}{k-1}\binom{n-r}{i-k}}{\binom{n-1}{i-1}}$$

Analysis: A Refinement

Hopes to solve the recurrence for $C_{n,i}$ are dim ...
However we can prove

$$f(x) = \lim_{n\to\infty, i/n\to x} C_{n,i}/n^{\alpha},$$

for $0 < x < 1$ exists and the recurrence leads to

$$f(x) = \frac{1}{K - \frac{2(K-1)}{\alpha+1}} \left[\int_0^x f\left(\frac{x-u}{1-u}\right)(1-u)^{\alpha}\,du + \int_x^1 f\left(\frac{x}{u}\right) u^{\alpha}\,du \right].$$

Analysis: A Refinement

The equation for $f(x)$ can be solved yielding

$$f(x) = \eta \cdot (x(1-x))^{\alpha/2}, \qquad 0 \leqslant x \leqslant 1,$$

for some constant η. Determination of η follows from

$$\int_0^1 f(x)\,dx = \beta$$

where β is the constant factor of the main order term in the expected cost of a random partial match:

$$\eta = \frac{\Gamma(2\phi+2)}{\Gamma^2(\phi+1)}\beta = \frac{\Gamma(2\phi+2)}{\Gamma^2(\phi+1)}\frac{\Gamma(2\alpha+1)}{(1-1/K)(1+\alpha)\alpha^3\Gamma^3(\alpha)}.$$

Analysis: A Refinement

From the analysis of $f(x)$ it follows

$$C_{n,i} = \eta \cdot \left(\sqrt{i(n+1-i)}\right)^{\alpha}.$$

For standard K-d trees a similar formula holds, now with α the characteristic exponent in the expected cost of partial matches in standard K-d trees and the constant η which depends also on j, the specified coordinate.

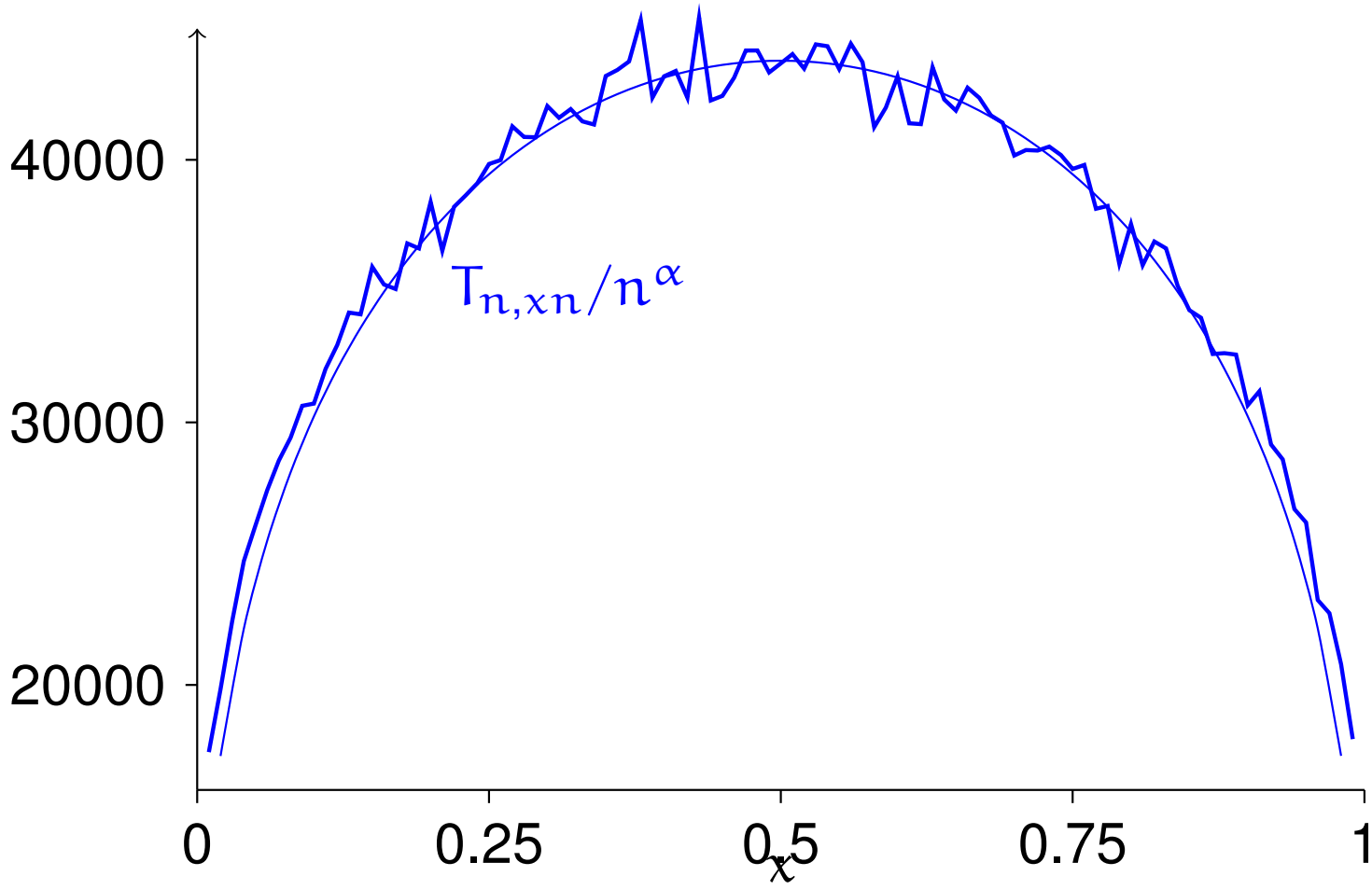
Analysis: A Refinement

The refined analysis of BELOW is the building block for a refined analysis of KDSELECT. We can show that the expected cost $T_{n,i}$ to find the i-th record along some given coordinate in a random relaxed K-d tree of size n is

$$T_{n,i} \sim n^{\alpha}(f(i/n)\ln(i) + f(1-i/n)\ln(n+1-i) - f(i/n)) + o(n^{\alpha}),$$

with $f(x) = \eta \cdot (x(1-x))^{\alpha/2}$.

Analysis: A Refinement



A plot of experimental values of $T_{n,xn}/n^{\alpha}$ (thick line) as a function of $x = i/n$, versus the theoretical predictions (thin line), for relaxed 2-d trees.

Final remarks

- A simple algorithm with sublinear expected cost
- It can easily be extended to many other multidimensional data structures
- Very little overhead: storing the size of each subtree is not very space consuming and it can also be sucessfully used for balancing (e.g., randomized relaxed K-d trees)
- Experiments show that it is competitive in practice compared to alternative solutions, for reasonably low dimensions (when K grows, $\alpha(\mathrm{K}) \rightarrow 1$)

Thanks for your attention!