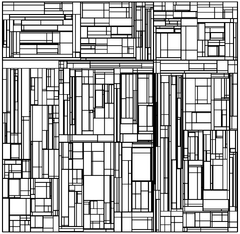
Probabilistic analysis of algorithms



Henning Sulzbach

INRIA Junior Seminar, Paris-Rocquencourt, January 2014

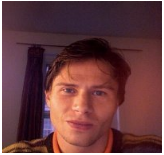
RAP - Réseaux, algorithmes et probabilités

About myself

May 2012: PhD in mathematics at the Goethe University Frankfurt, supervisor: Ralph Neininger

June 2012 - August 2013: Post-Doc at the Goethe University Frankfurt and at the McGill University of Montréal

Since Sept. 2013: FMSP (Fondation Sciences Mathématiques de Paris) Post-Doc fellowship at INRIA with host Nicolas Broutin.



Outline

Analysis of algorithms

Goal: Estimate the computational complexity of algorithms and computational problems

Examples:

Basic algorithms:	sorting & searching
Data structures:	insertion, deletion, searching, . . .
Approximation algorithms:	traveling salesman problem, . . .

Analysis of algorithms

Goal: Estimate the computational complexity of algorithms and computational problems

Complexity: storage and/or running-time

Motivation:

- rigorous verification of the performance
- separation of efficient and inefficient algorithms
- estimate quality of approximation algorithms
- improve known algorithms

Probabilistic analysis of algorithms

Classical approach: Complexity is measured in terms of the *worst-case* behavior

Random input data: *average* instead of worst-case behavior

X_n: complexity for input of size n (n large). Study

- $\mathbb{E}[X_n]$ (*mean behavior*)
- $\sigma_n^2 := \mathrm{Var}(X_n)$ (*standard deviation*)
- $\mathbb{P}(X_n \geq 2\mathbb{E}[X_n])$ (*large deviations*)
- As $X_n - \mathbb{E}[X_n] \approx \sigma_n$, the limit

$$\lim_{n\to\infty} \mathbb{P}(X_n - \mathbb{E}[X_n] \leq x\sigma_n) = F(x)$$

is plausible to exist.

An important example:

$X_n = \#$ of heads in n coin tosses:

$\mathbb{E}[X_n] = \frac{1}{2}n$, $\sigma_n = \frac{1}{2}\sqrt{n}$ and

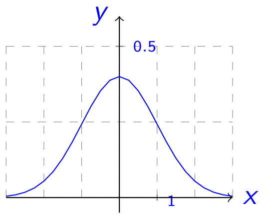
$$\lim_{n\to\infty} \mathbb{P}\left(\frac{X_n - \mathbb{E}[X_n]}{\sigma_n} \leq x\right) = \Phi(x) = \frac{1}{\sqrt{2\pi}} \int_{-\infty}^{x} e^{-t^2/2} dt$$

We say

$$\frac{X_n - \mathbb{E}[X_n]}{\sigma_n} \to \mathcal{N}$$

in distribution as $n \to \infty$ where

$$\mathbb{P}(\mathcal{N} \leq x) = \Phi(x).$$

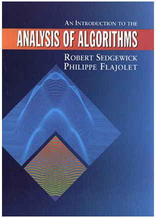


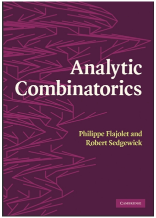
The analysis of algorithms at Rocquencourt

Project ALGO (1989 - 2012) with head

Philippe Flajolet (1948 - 2011)

Today: ALGO $\rightarrow$ RAP





Bob Sedgewick's online course:
https://www.coursera.org/course/algs4partI

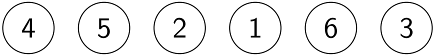
The Quicksort algorithm (Hoare 1962)

Input: n distinct numbers, say $1, \ldots, n$

Task: list of numbers in increasing order

Routine:

1. Compare all elements to the first element.
2. Proceed recursively in sublists until all lists have size 0 or 1.



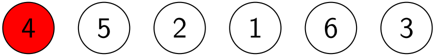
The Quicksort algorithm (Hoare 1962)

Input: n distinct numbers, say $1, \ldots, n$

Task: list of numbers in increasing order

Routine:

1. Compare all elements to the first element.
2. Proceed recursively in sublists until all lists have size 0 or 1.



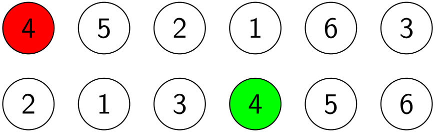
The Quicksort algorithm (Hoare 1962)

Input: n distinct numbers, say $1, \ldots, n$

Task: list of numbers in increasing order

Routine:

1. Compare all elements to the first element.
2. Proceed recursively in sublists until all lists have size 0 or 1.



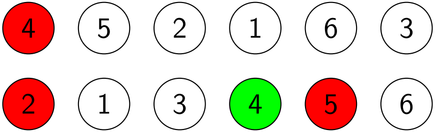
The Quicksort algorithm (Hoare 1962)

Input: n distinct numbers, say $1, \ldots, n$

Task: list of numbers in increasing order

Routine:

1. Compare all elements to the first element.
2. Proceed recursively in sublists until all lists have size 0 or 1.



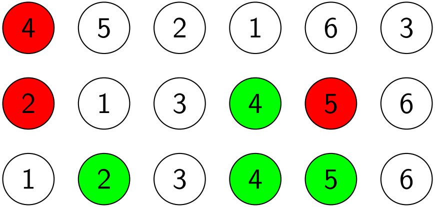
The Quicksort algorithm (Hoare 1962)

Input: n distinct numbers, say $1, \ldots, n$

Task: list of numbers in increasing order

Routine:

1. Compare all elements to the first element.
2. Proceed recursively in sublists until all lists have size 0 or 1.



Aim: Analyze the complexity of Quicksort

Analysis of Quicksort

A common choice for the complexity of Quicksort is the number of executed key comparisons X_n sorting a list of size n.

Model: input data is permuted uniformly at random

Results (Knuth 1973):

- $\mathbb{E}[X_n] \sim 2n \log n$,
- $\sigma_n \sim cn$ for $c = \sqrt{7 - 2\pi^2/3} = 0.6485\ldots$

Theorem (Régnier '89, Rösler '91)

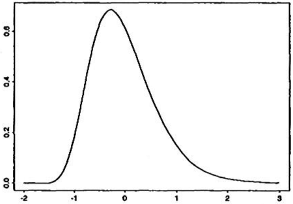
There exists a r.v. Y *with* $\mathbb{E}[Y] = 0$ *and* $\sigma_Y = 1$ *such that*

$$Y_n := \frac{X_n - \mathbb{E}[X_n]}{\sigma_n} \to Y$$

in distribution, that is $\mathbb{P}(Y_n \leq x) \to \mathbb{P}(Y \leq x)$ *for all* $x \in \mathbb{R}$.

The shape of Y

Y admits a density f, i.e. $\mathbb{P}(Y \leq x) = \int_{-\infty}^{x} f(t)dt$



- f is bounded (≤ 16) (Janson, Fill 2000)

Conjecture (Knessl, Szpankowski 1997): Very roughly,

$$\mathbb{P}(Y \geq x) \sim e^{-\alpha x \log x}, \quad x \to \infty, \alpha > 0$$

$$\mathbb{P}(Y \leq x) \sim e^{-\beta e^{-\gamma x}}, \quad x \to -\infty, \beta, \gamma > 0$$

Remember $\mathbb{P}(\mathcal{N} \geq x) \sim e^{-x^2/2}$.

A first step: The recurrence

For two random quantities Z, Z', write $Z \stackrel{d}{=} Z$ if, for all x,

$$\mathbb{P}\left(Z \leq x\right) = \mathbb{P}\left(Z' \leq x\right).$$

Then, with $L_n =$ size of the left sublist,

$$X_n \stackrel{d}{=} X^{(1)}_{L_n} + X^{(2)}_{n-L_n-1} + n - 1.$$

- L_n is uniformly distributed on $\{0, \ldots, n-1\}$,
- $(X^{(1)}_k)$ and $(X^{(2)}_k)$ are distributed like (X_k)
- $(L_n, (X^{(1)}_k), (X^{(2)}_k))$ are independent

Very often, a distributional recurrence relation is the first step of the analysis.

Quicksort is an algorithm of type *divide and conquer.*

Partial match retrieval

Input: Set of elements with m different parameters.

Output: Subset of elements where $1 \leq i < m$ parameters are given.

Example:

- Find people having certain characteristics in a data base

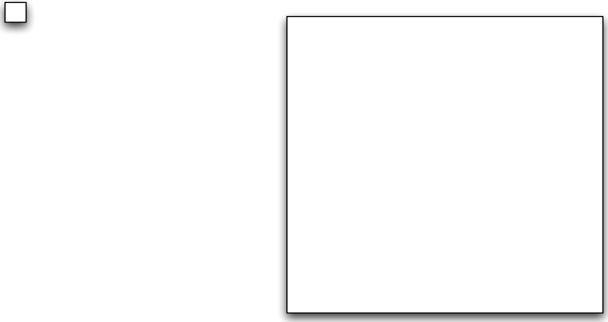
Here: quadtree with two-parameter space $[0, 1]^2$.

Quadtree - construction

Two-dimensional Quadtree

Quadtree - construction

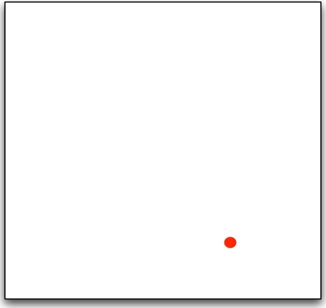
Two-dimensional Quadtree



Quadtree - construction

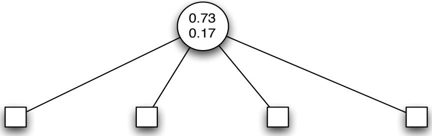
Two-dimensional Quadtree

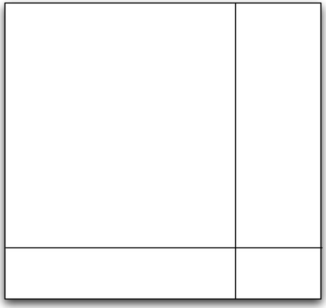
$0.73, 0.17$



Quadtree - construction

Two-dimensional Quadtree

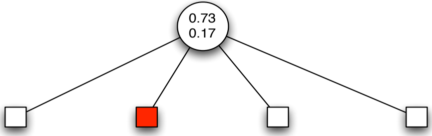


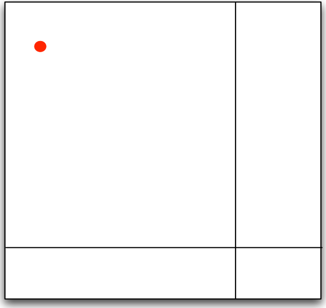


Quadtree - construction

Two-dimensional Quadtree

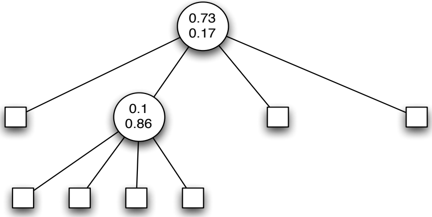
$0.1, 0.86$

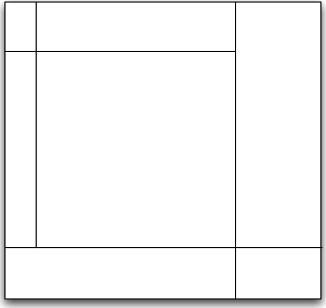




Quadtree - construction

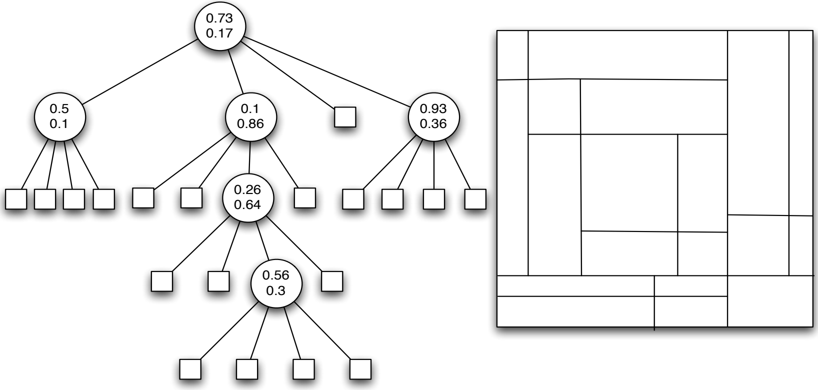
Two-dimensional Quadtree





Quadtree - construction

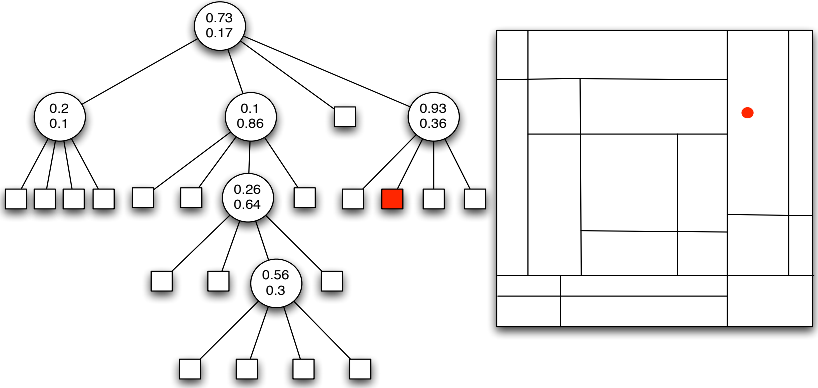
Two-dimensional Quadtree



Quadtree - construction

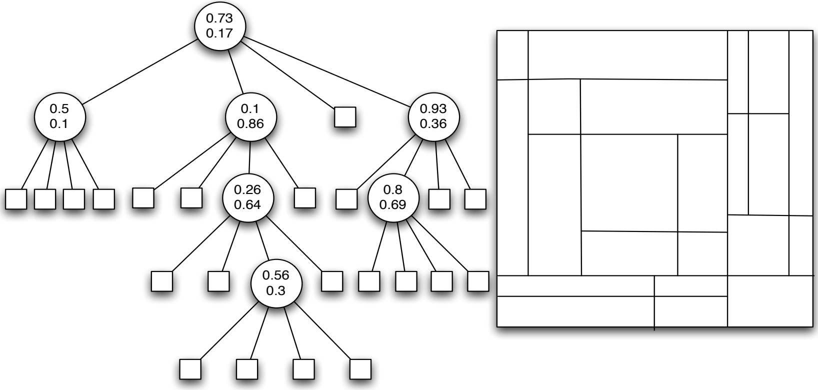
Two-dimensional Quadtree

$0.8, 0.69$



Quadtree - construction

Two-dimensional Quadtree

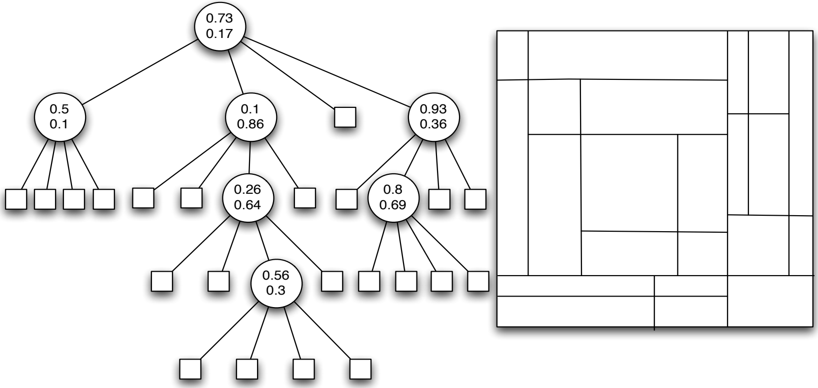


Partial match retrieval

Task: Find values with first coordinate $s = 0.2$

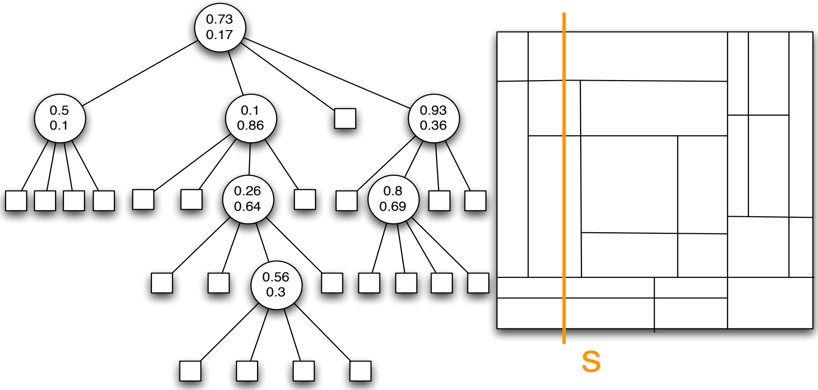
Partial match retrieval

Task: Find values with first coordinate $s = 0.2$



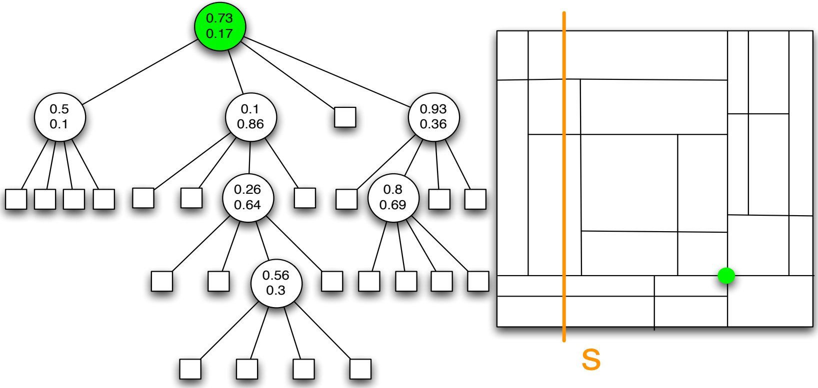
Partial match retrieval

Task: Find values with first coordinate $s = 0.2$



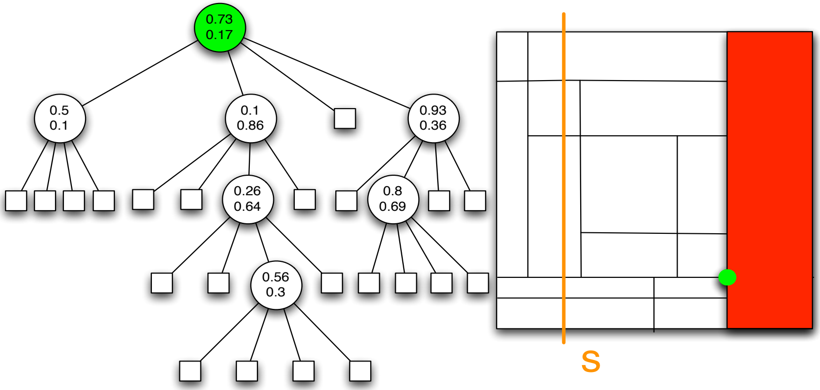
Partial match retrieval

Task: Find values with first coordinate $s = 0.2$



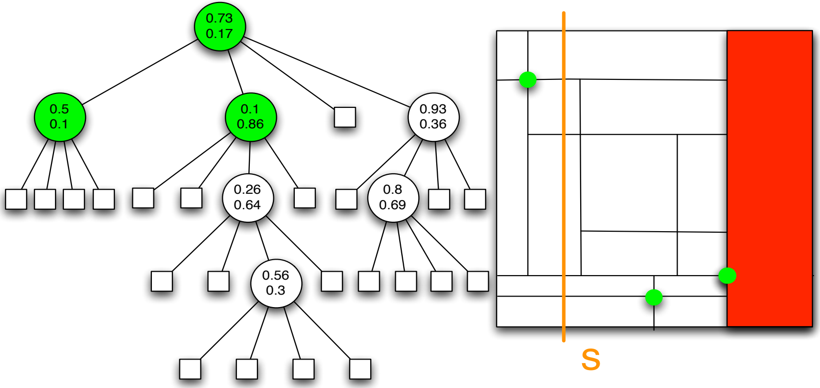
Partial match retrieval

Task: Find values with first coordinate $s = 0.2$



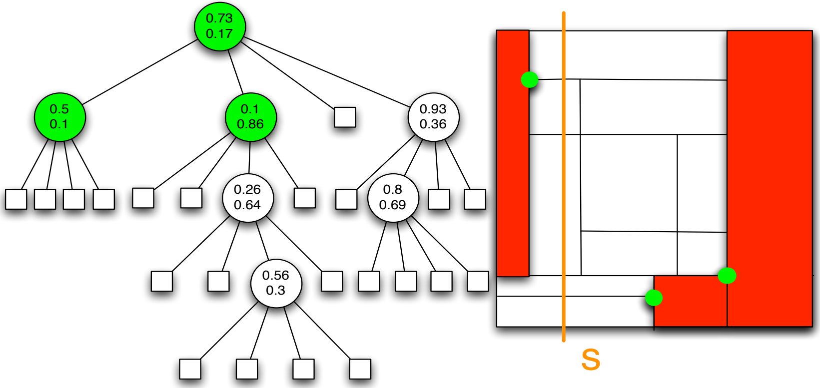
Partial match retrieval

Task: Find values with first coordinate $s = 0.2$



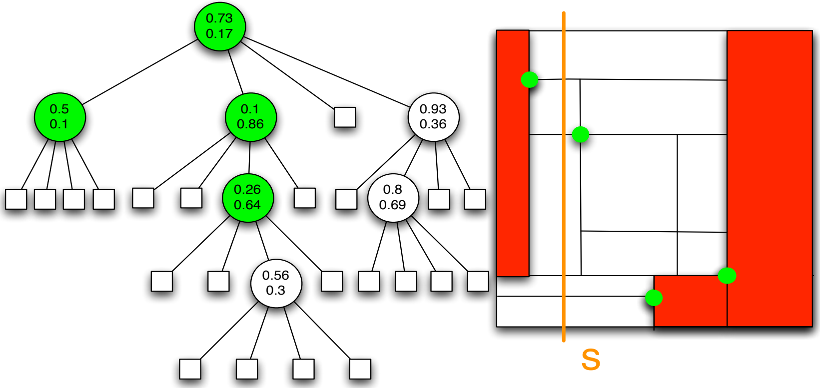
Partial match retrieval

Task: Find values with first coordinate $s = 0.2$



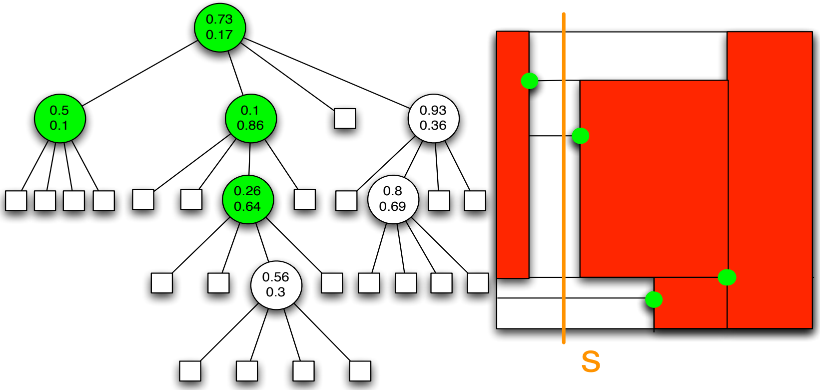
Partial match retrieval

Task: Find values with first coordinate $s = 0.2$



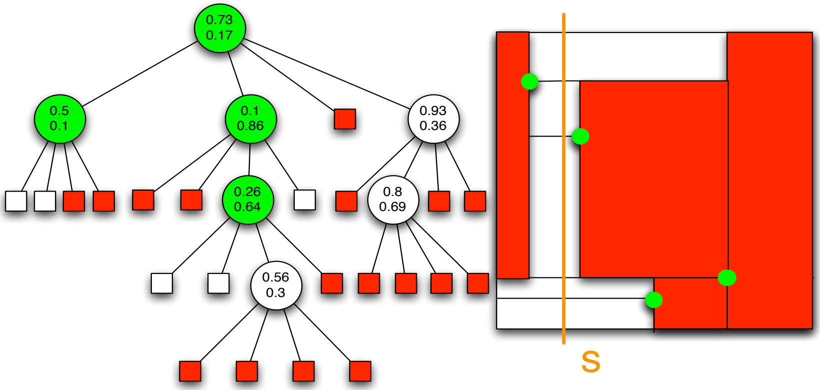
Partial match retrieval

Task: Find values with first coordinate $s = 0.2$



Partial match retrieval

Task: Find values with first coordinate $s = 0.2$

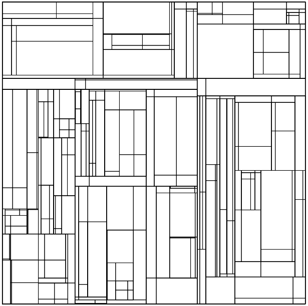


$C_n(s)$: number of visited nodes retrieving $\{s, *\}$.

Model: data points independent and uniformly distributed on $[0,1]^2$

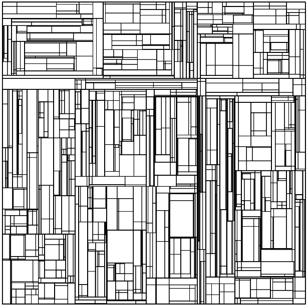
Simulations

$n = 100$



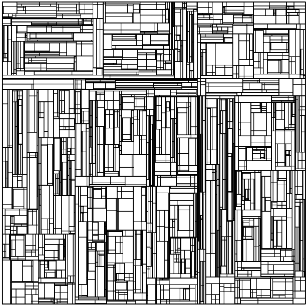
Simulations

$n = 500$

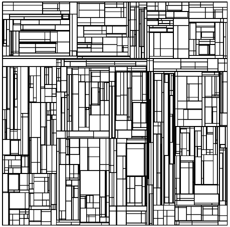


Simulations

$n = 1000$



A first result



Theorem (Flajolet et al. '93)

Let ξ be uniformly distributed on $[0,1]$, independent of the tree. Then, as $n \to \infty$,

$$\mathbb{E}\left[C_n(\xi)\right] \sim cn^{\beta},$$

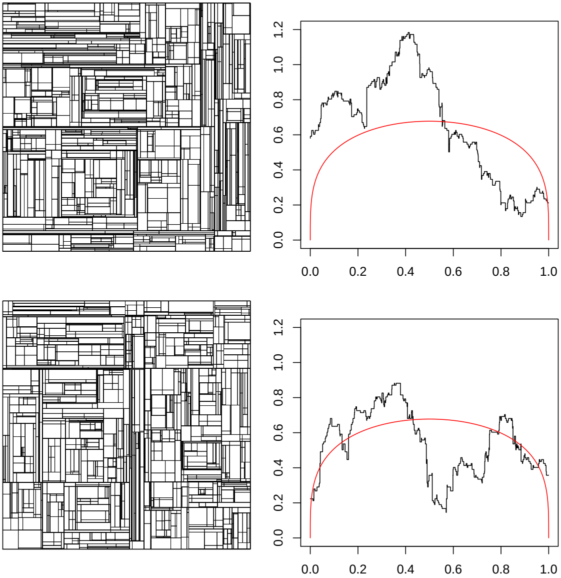
$$\beta = \frac{\sqrt{17}-3}{2} = 0.561\ldots, c > 0$$

Theorem (Curien, Joseph '12)

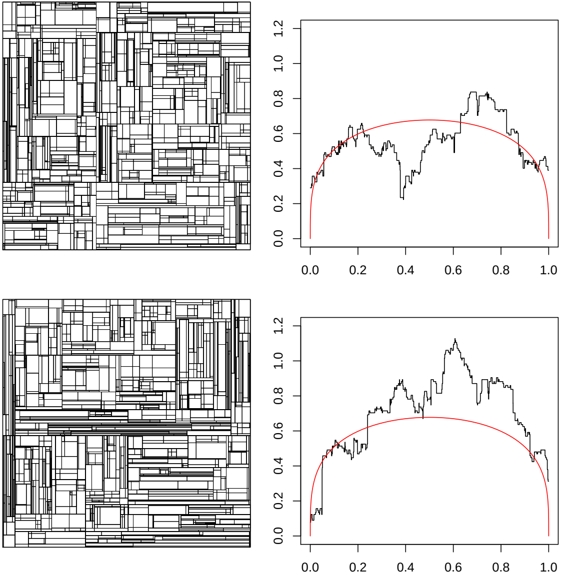
For $s \in (0,1)$, as $n \to \infty$, we have $\mathbb{E}\left[C_n(s)\right] \sim c'(s(1-s))^{\beta/2} n^{\beta}$

Open: variance, limit distribution, behavior of $C_n(s)$ and $\sup_s C_n(s)$ (worst-case).

Simulations of C_n/n^β, $n = 500$



Simulations of C_n/n^β, $n = 500$



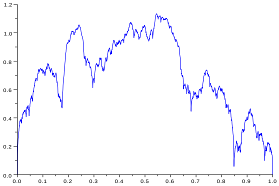
A limit theorem

Theorem (Broutin, Neininger, S.)

In distribution

$$\frac{C_n(s)}{n^\beta} \to Y(s)$$

as random continuous functions where $\mathbb{E}\left[Y(s)\right] = c'(s(1-s))^{\beta/2}$.



Corollaries

- $\frac{C_n(s)}{n^\beta} \to Y(s)$ for any fixed $s \in [0,1]$,
- $\frac{C_n(\xi)}{n^\beta} \to Y(\xi)$ with a uniform ξ,
- the supremum is of the *same* order as the average-case,

$$\frac{\sup_s C_n(s)}{n^\beta} \to \sup_s Y(s),$$

and $\mathbb{E}\left[\exp\left(\lambda \sup_s Y(s)\right)\right] < \infty$ for all $\lambda > 0$.

- $\mathrm{Var}\, C_n(s) \sim c_1 (s(1-s))^\beta n^{2\beta}$ with $c_1 = 0.136\ldots$.
- $\mathrm{Var}\, C_n(\xi) \sim c_2 n^{2\beta}$ with $c_2 = 0.447\ldots$.

But what happens at $s = 0$?

The results say little at the boundary for $C_n(0) \ll n^\beta$ as $n \to \infty$.

Theorem (Flajolet et al. '93, Curien, Joseph '12)

As $n \to \infty$,

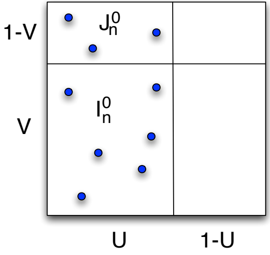
$$\mathbb{E}\left[C_n(0)\right] \sim \kappa n^{\sqrt{2}-1}$$

and $C_n(0)/n^{\sqrt{2}-1} \to C$ *for some r.v.* C. *Note*

$$0.414\ldots = \sqrt{2} - 1 < \beta$$

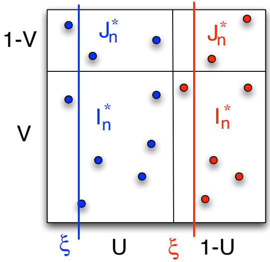
.

An intuitive explanation



$$I_n^0 \stackrel{d}{=} \text{Bin}(n-1, UV) \sim nUV$$

$$J_n^0 \stackrel{d}{=} \text{Bin}(n-1, U(1-V)) \sim nU(1-V)$$



$$I_n^* \stackrel{d}{=} \text{Bin}(n-1, XV) \sim nXV$$

$$J_n^* \stackrel{d}{=} \text{Bin}(n-1, X(1-V)) \sim nX(1-V)$$

$$X = \begin{cases} U & \xi < U \\ 1-U & \xi > U \end{cases}$$

Note: $\mathbb{P}(X > x) \geq \mathbb{P}(U > x)$, X is *larger* than U.

Conclusions

Summary:

- Sophisticated mathematical tools allow *precise* statements with *rigorous* proofs about the complexity of algorithms and operations in data structures for large input sizes.
- Conversely, the analysis of algorithms leads to the development of new mathematical ideas.

MERCI