THÉORIE DE L'INFORMATION:

MODÈLES, ALGORITHMES, ANALYSE

Brigitte Vallée
Laboratoire GREYC
(CNRS et Université de Caen, France)

Exposé au Colloquium Jacques Morgenstern, le 11 février 2010.

THÉORIE DE L'INFORMATION:

MODÈLES, ALGORITHMES, ANALYSE

Brigitte VALLÉE
Laboratoire GREYC
(CNRS et Université de Caen, France)

Exposé fondé sur des travaux communs
avec Julien CLÉMENT, Jim FILL et Philippe FLAJOLET

Plan of the talk.

– Motivations of the study

– A general model of source

– Description of the main results

– Description of the methods

Plan of the talk.

– Motivations of the study

– A general model of source

– Description of the main results

– Description of the methods

The classical framework for sorting.

The main sorting algorithms or searching algorithms

e.g., `QuickSort`, `BST-Search`, `InsertionSort`,...

deal with n (distinct) keys $U_1, U_2, \ldots, U_n$ of the same ordered set Ω.

They perform comparisons and exchanges between keys.

The unit cost is the key–comparison.

The classical framework for sorting.

The main sorting algorithms or searching algorithms

e.g., `QuickSort`, `BST-Search`, `InsertionSort`,...

deal with n (distinct) keys $U_1, U_2, \dots, U_n$ of the same ordered set Ω.

They perform comparisons and exchanges between keys.

The unit cost is the key–comparison.

The behaviour of the algorithm (wrt to key–comparisons)

only depends on the relative order between the keys.

It is sufficient to restrict to the case when $\Omega = [1..n]$.

The input set is then $\mathfrak{S}_n$, with uniform probability.

The classical framework for sorting.

The main sorting algorithms or searching algorithms

e.g., `QuickSort, BST-Search, InsertionSort`,...

deal with n (distinct) keys $U_1, U_2, \ldots, U_n$ of the same ordered set Ω.

They perform comparisons and exchanges between keys.

The unit cost is the key–comparison.

The behaviour of the algorithm (wrt to key–comparisons)

only depends on the relative order between the keys.

It is sufficient to restrict to the case when $\Omega = [1..n]$.

The input set is then $\mathfrak{S}_n$, with uniform probability.

Then, the analysis of all these algorithms is very well known,

with respect to the number of key–comparisons performed

in the worst-case, or in the average case.

Here, realistic analysis of the two algorithms QuickSort and QuickSelect

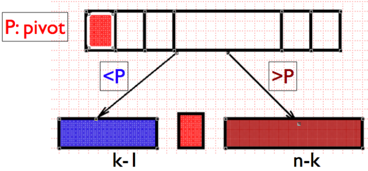
QuickSort (n, A): sorts the array A

Choose a pivot;

$(k, A_-, A_+) :=$ Partition(A);

QuickSort $(k-1, A_-)$;

QuickSort $(n-k, A_+)$.



Here, realistic analysis of the two algorithms QuickSort and QuickSelect

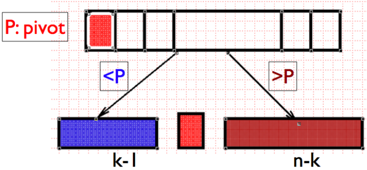
QuickSort (n, A): sorts the array A

Choose a pivot;

$(k, A_-, A_+) :=$ Partition(A);

QuickSort $(k-1, A_-)$;

QuickSort $(n-k, A_+)$.



QuickSelect (n, m, A): returns the value of the element of rank m in A.

Choose a pivot;

$(k, A_-, A_+) :=$ Partition(A);

If $m = k$ then QuickSelect := pivot

else if $m < k$ then QuickSelect $(k-1, m, A_-)$

else QuickSelect $(n-k, m-k, A_+)$;

Known results for QuickSort and QuickSelect for various values of rank m about the mean number K_n of key–comparisons

QuickSort (n)	sorts		$K_n \sim 2n \log n$
QuickMin(n)	minimum	$m = 1$	$K_n \sim 2n$
QuickMax(n)	maximum	$m = n$	$K_n \sim 2n$
QuickRand(n)		$m \in [1..n]_{\mathcal{R}}$	$K_n \sim 3n$
QuickQuant$_\alpha(n)$	α–quantile	$m = \lfloor \alpha n \rfloor$	$K_n \sim \kappa(\alpha)\, n$
QuickMed(n)	median	$m = \lfloor n/2 \rfloor$	$K_n \sim 2(1 + \log 2)n$

Known results for QuickSort and QuickSelect for various values of rank m

about the mean number K_n of key–comparisons

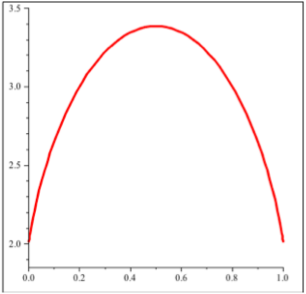
QuickSort (n)	sorts		$K_n \sim 2n \log n$
QuickMin(n)	minimum	$m = 1$	$K_n \sim 2n$
QuickMax(n)	maximum	$m = n$	$K_n \sim 2n$
QuickRand(n)		$m \in [1..n]_{\mathcal{R}}$	$K_n \sim 3n$
QuickQuant$_\alpha(n)$	α–quantile	$m = \lfloor \alpha n \rfloor$	$K_n \sim \kappa(\alpha)\, n$
QuickMed(n)	median	$m = \lfloor n/2 \rfloor$	$K_n \sim 2(1 + \log 2)n$

On the right,

the function $\kappa : \alpha \mapsto 2\,[1 + h(\alpha)]$

where $h(\cdot)$ is the entropy function

$$h(\alpha) = \alpha|\log \alpha| + (1 - \alpha)|\log(1 - \alpha)|$$



A more realistic framework for sorting.

Keys are viewed as words. The domain Ω of keys is a subset of Σ^∞, $\Sigma^\infty = \{$the infinite words on some ordered alphabet $\Sigma\}$.

The words are compared [wrt the lexicographic order].

The realistic unit cost is now the symbol–comparison.

A more realistic framework for sorting.

Keys are viewed as words. The domain Ω of keys is a subset of Σ^∞,
$\Sigma^\infty = \{$the infinite words on some ordered alphabet $\Sigma\}$.
The words are compared [wrt the lexicographic order].
The realistic unit cost is now the symbol–comparison.

The realistic cost of the comparison between two words A and B,

$$A = a_1\, a_2\, a_3 \ldots a_i \ldots \qquad \text{and} \qquad B = b_1\, b_2\, b_3 \ldots b_i \ldots$$

equals $k + 1$, where k is the length of their largest common prefix

$$k := \max\{i; \quad \forall j \le i, \quad a_j = b_j\} = \text{the coincidence}$$

A more realistic framework for sorting.

Keys are viewed as words. The domain Ω of keys is a subset of Σ^∞,
$\Sigma^\infty = \{$the infinite words on some ordered alphabet $\Sigma\}$.
The words are compared [wrt the lexicographic order].
The realistic unit cost is now the symbol–comparison.

The realistic cost of the comparison between two words A and B,

$$A = a_1\, a_2\, a_3 \ldots a_i \ldots \qquad \text{and} \qquad B = b_1\, b_2\, b_3 \ldots b_i \ldots$$

equals $k+1$, where k is the length of their largest common prefix

$$k := \max\{i; \quad \forall j \le i, \quad a_j = b_j\} = \text{the coincidence}$$

a b a b b b...
a b a a b a...

coincidence=3; #comparisons=4.

We are interested in this new cost for each algorithm:

the number of symbol–comparisons ... and its mean value S_n (for n words)

We are interested in this new cost for each algorithm:
the number of symbol–comparisons ... and its mean value S_n (for n words)

How is S_n compared to K_n? That is the question....

An initial question asked by Sedgewick in 2000...
... In order to also compare with other text algorithms.

We are interested in this new cost for each algorithm:

the number of symbol–comparisons ... and its mean value S_n (for n words)

How is S_n compared to K_n? That is the question....

An initial question asked by Sedgewick in 2000...

... In order to also compare with other text algorithms.

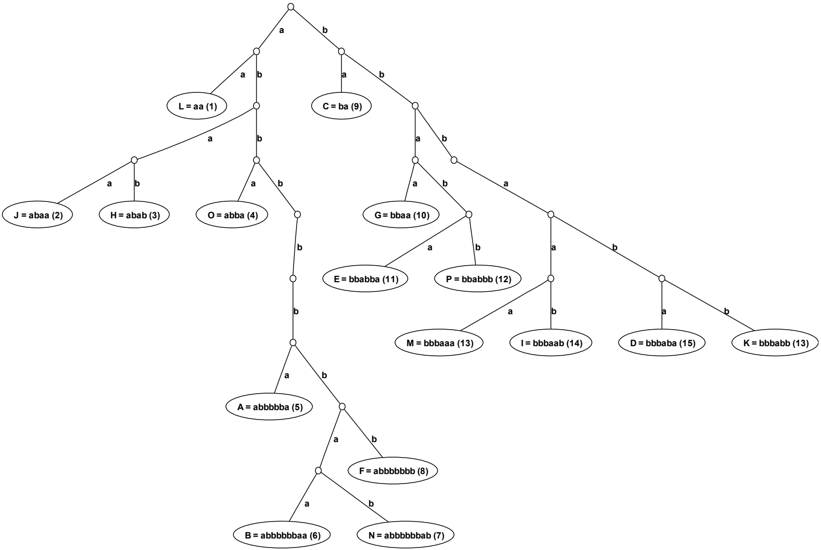
Two data structures for sorting a set of words

— the trie, for dictionary algorithms

— the binary search tree (BST) closely related to `QuickSort`

An example : A trie built on a set of words.

A = abbbbbaaabab B = abbbbbbaabaa C = baabbbabbbba D =bbbababbbaab E = bbabbaababbb
F = abbbbbbbbabb G = bbaabbabbaba H = ababbbabbbab I = bbbaabbbbbbb J = abaabbbbaabb
K = bbbabbbbbbaa L = aaaabbabaaba M = bbbaaabbbbbb N = abbbbbbabbaa O = abbababababbbb P = bbabbbaaaabb



The `Trie` structure

A finite set $\mathcal{X} = \{X_1, X_2, \ldots, X_n\}$ formed with n words.

The tree $\texttt{Trie}\,(\mathcal{X})$ built on $\mathcal{X}$ is defined by the three rules:

– If $|\mathcal{X}| = 0$, $\texttt{Trie}\,(\mathcal{X}) = \emptyset$

– If $|\mathcal{X}| = 1$, $\mathcal{X} = \{\ X\}$, $\texttt{Trie}\,(\mathcal{X})$ is a leaf labeled by X.

– If $|\mathcal{X}| \geq 2$, then $\texttt{Trie}\,(\mathcal{X})$ is formed with

  – an internal node

  – and n subtries $\texttt{Trie}\,(\mathcal{X} \setminus m_1), \ldots, \texttt{Trie}\,(\mathcal{X} \setminus m_r)$

  where $\mathcal{X} \setminus m :=$ {words of $\mathcal{X}$ that begin with m, stripped of m}.

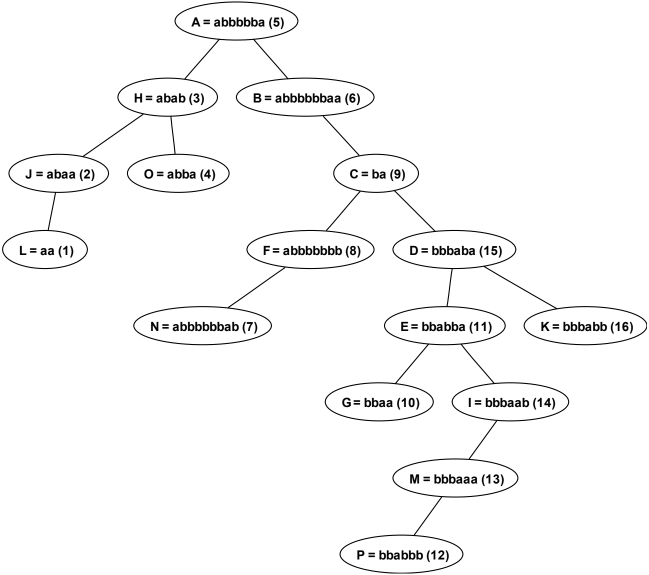
If $\mathcal{X} \setminus m \neq \emptyset$, the edge: internal node $\rightarrow \texttt{Trie}\,(\mathcal{X} \setminus m)$ has label m.

An example : The BST (binary search tree) built on the same sequence of words

A = abbbbbaaabab   B = abbbbbbaabaa   C = baabbbabbbba   D = bbbababbbaab   E = bbabbaababbb

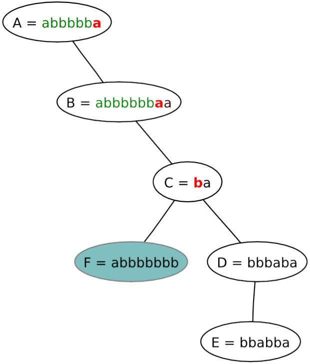
F = abbbbbbbbabb   G = bbaabbabbaba   H = ababbbabbbab   I = bbbaabbbbbbb   J = abaabbbbaabb

K = bbbabbbbbbaa   L = aaaabbabaaba   M = bbbaaabbbbbb   N = abbbbbbabbaa   O = abbababbbbb   P = bbabbbaaaabb



An example : The cost of the insertion of the key F into the BST

F =abbbbbbb



Number of symbol comparisons needed $= 16$

$=$ 7 for comparing to A

$+$ 8 for comparing to B

$+$ 1 for comparing to C

Plan of the talk.

– Motivations of the study

– A general model of source

– Description of the main results

– Description of the methods

The parametrization of a general source

A general source $\mathcal{S}$ produces infinite words

on an ordered alphabet $\Sigma := \{a_1, \ldots, a_r\}$.

For $w \in \Sigma^\star$, $p_w :=$ probability that a word begins with the prefix w.

The set $\{p_w, \quad w \in \Sigma^\star\}$ defines the source $\mathcal{S}$. We assume

$$\pi_k := \sup\{p_w, \quad w \in \Sigma^k\} \to 0 \quad \text{for } k \to \infty$$

For each length k, we consider the p_w's for $w \in \Sigma^k$,

sorted with respect to the lexicographic order on Σ^k.

The parametrization of a general source

A general source $\mathcal{S}$ produces infinite words

on an ordered alphabet $\Sigma := \{a_1, \ldots, a_r\}$.

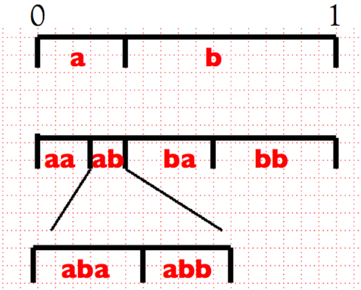
For $w \in \Sigma^\star$, $p_w :=$ probability that a word begins with the prefix w.

The set $\{p_w, \quad w \in \Sigma^\star\}$ defines the source $\mathcal{S}$. We assume

$$\pi_k := \sup\{p_w, \quad w \in \Sigma^k\} \to 0 \quad \text{for } k \to \infty$$

For each length k, we consider the p_w's for $w \in \Sigma^k$,

sorted with respect to the lexicographic order on Σ^k.



The parametrization of a general source

A general source $\mathcal{S}$ produces infinite words

on an ordered alphabet $\Sigma := \{a_1, \ldots, a_r\}$.

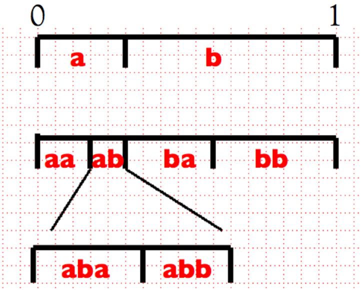
For $w \in \Sigma^\star$, $p_w :=$ probability that a word begins with the prefix w.

The set $\{p_w, \ \ w \in \Sigma^\star\}$ defines the source $\mathcal{S}$. We assume

$$\pi_k := \sup\{p_w, \ \ w \in \Sigma^k\} \to 0 \quad \text{for } k \to \infty$$

For each length k, we consider the p_w's for $w \in \Sigma^k$,

sorted with respect to the lexicographic order on Σ^k.



We define two other probabilities

$$p_w^{(-)} := \sum_{\substack{\alpha \in \Sigma^k, \\ \alpha < w}} p_\alpha, \quad p_w^{(+)} := \sum_{\substack{\alpha \in \Sigma^k, \\ \alpha > w}} p_w.$$

The parametrization of a general source

A general source $\mathcal{S}$ produces infinite words

on an ordered alphabet $\Sigma := \{a_1, \ldots, a_r\}$.

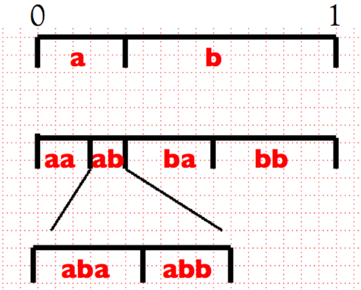
For $w \in \Sigma^\star$, $p_w :=$ probability that a word begins with the prefix w.

The set $\{p_w, \quad w \in \Sigma^\star\}$ defines the source $\mathcal{S}$. We assume

$$\pi_k := \sup\{p_w, \quad w \in \Sigma^k\} \to 0 \quad \text{for } k \to \infty$$

For each length k, we consider the p_w's for $w \in \Sigma^k$,

sorted with respect to the lexicographic order on Σ^k.



We define two other probabilities

$$p_w^{(-)} := \sum_{\substack{\alpha \in \Sigma^k, \\ \alpha < w}} p_\alpha, \quad p_w^{(+)} := \sum_{\substack{\alpha \in \Sigma^k, \\ \alpha > w}} p_w.$$

Then, for any $X \in \Sigma^\infty$,

$$\lim_{w \to X} p_w^{(-)} = 1 - \lim_{w \to X} p_w^{(+)} := P(X)$$

Consider the set $\Sigma^\infty(\mathcal{S})$ the set of infinite words emitted by $\mathcal{S}$.

The function $P : \Sigma^\infty(\mathcal{S}) \to [0,1]$ is strictly increasing almost everywhere.

Only possible exceptions: $P(X) = P(Y)$ iff

$$\exists w \in \sigma^\star, \ \ \exists t < r, \quad \text{such that} \quad X = w \cdot a_t \cdot a_r^\infty, \quad Y = w \cdot a_{t+1} \cdot a_1^\infty$$

Consider the set $\Sigma^\infty(\mathcal{S})$ the set of infinite words emitted by $\mathcal{S}$.

The function $P : \Sigma^\infty(\mathcal{S}) \to [0,1]$ is strictly increasing almost everywhere.

Only possible exceptions: $P(X) = P(Y)$ iff

$$\exists w \in \sigma^\star, \quad \exists t < r, \quad \text{such that} \quad X = w \cdot a_t \cdot a_r^\infty, \quad Y = w \cdot a_{t+1} \cdot a_1^\infty$$

Then, outside the exceptional set, each infinite word X is written as

$X = M(u)$ with $M : [0,1] \to \Sigma^\infty$.

The map M provides a parametrization of the source $\mathcal{S}$.

Via the mapping M,

[Drawing in $\mathcal{S}$ wrt the p_w's] $\equiv$ [Uniform drawing in $[0,1]$]

Consider the set $\Sigma^\infty(\mathcal{S})$ the set of infinite words emitted by $\mathcal{S}$.

The function $P : \Sigma^\infty(\mathcal{S}) \to [0,1]$ is strictly increasing almost everywhere.

Only possible exceptions: $P(X) = P(Y)$ iff

$$\exists w \in \sigma^\star, \quad \exists t < r, \quad \text{such that} \quad X = w \cdot a_t \cdot a_r^\infty, \quad Y = w \cdot a_{t+1} \cdot a_1^\infty$$

Then, outside the exceptional set, each infinite word X is written as

$$X = M(u) \text{ with } M : [0,1] \to \Sigma^\infty.$$

The map M provides a parametrization of the source $\mathcal{S}$.

Via the mapping M,

[Drawing in $\mathcal{S}$ wrt the p_w's] $\equiv$ [Uniform drawing in $[0,1]$]

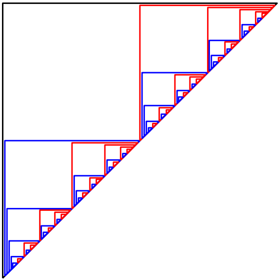
For any finite prefix $w \in \Sigma^\star$,

the set $\{u, \ M(u) \text{ begins with } w\}$ is an interval with endpoints $p_w^{(-)}, p_w^{(+)}$.

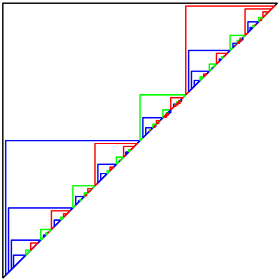
This is the fundamental interval of w. Its length equals p_w.

For any finite prefix $w \in \Sigma^\star$,
the set $\{u, \; M(u) \text{ begins with } w\}$ is an interval with endpoints $p_w^{(-)}, p_w^{(+)}$.
This is the fundamental interval of w. Its length equals p_w.

Instances of fundamental intervals for two memoryless sources.



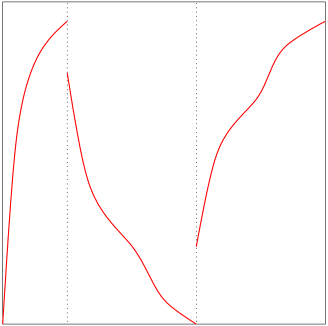
Memoryless source on $\{a, b\}$
$p_a = 1/2$, $p_b = 1/2$

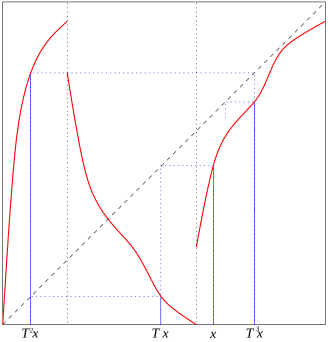


Memoryless source on $\{a, b, c\}$
$p_a = 1/2$, $p_b = 1/6$, $p_c = 1/3$

Natural instances of sources: Dynamical sources

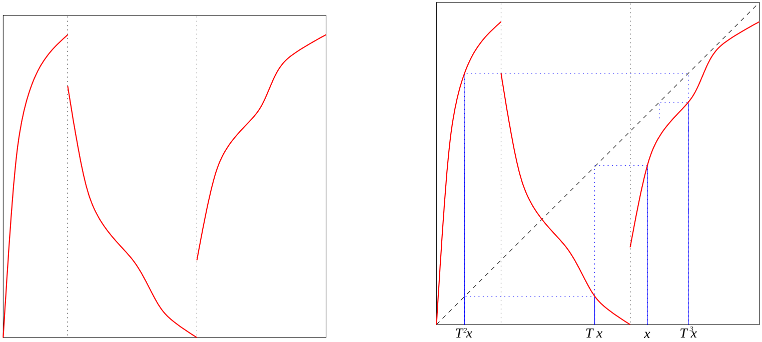
With a shift map $T : \mathcal{I} \to \mathcal{I}$ and an encoding map $\tau : \mathcal{I} \to \Sigma$,
the emitted word is $M(x) = (\tau x, \tau T x, \tau T^2 x, \ldots \tau T^k x, \ldots)$





Natural instances of sources: Dynamical sources

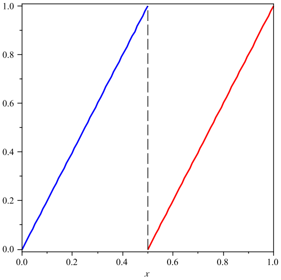
With a shift map $T : \mathcal{I} \to \mathcal{I}$ and an encoding map $\tau : \mathcal{I} \to \Sigma$,
the emitted word is $M(x) = (\tau x, \tau T x, \tau T^2 x, \ldots \tau T^k x, \ldots)$

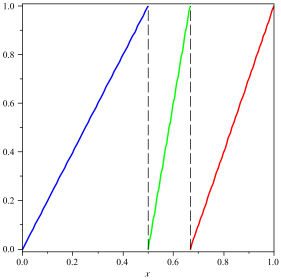


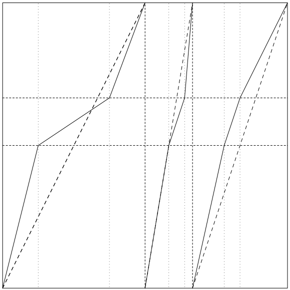
A dynamical system, with $\Sigma = \{a, b, c\}$ and a word $M(x) = (c, b, a, c \ldots)$.

Memoryless sources or Markov chains.

= Dynamical sources with affine branches....







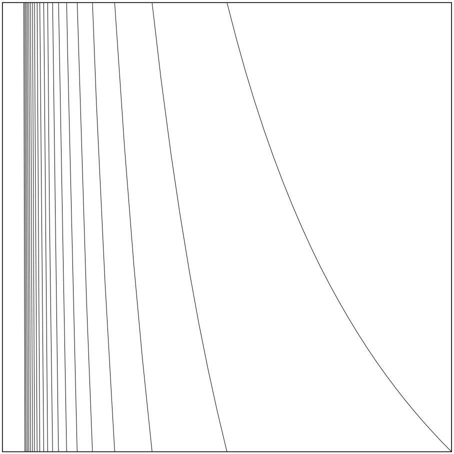
The dynamical framework leads to more general sources.

The curvature of branches entails correlation between symbols

The dynamical framework leads to more general sources.

The curvature of branches entails correlation between symbols

Example : the Continued Fraction source



A main analytical object related to any source:

the Dirichlet series of probabilities, $\Lambda(s) := \sum_{w \in \Sigma^\star} p_w^{-s}$

A main analytical object related to any source:

the Dirichlet series of probabilities, $\Lambda(s) := \sum_{w \in \Sigma^\star} p_w^{-s}$

Memoryless sources, with probabilities (p_i)

$$\Lambda(s) = \frac{1}{1-\lambda(s)} \qquad \text{with} \quad \lambda(s) = \sum_{i=1}^{r} p_i^{-s}$$

A main analytical object related to any source:

the Dirichlet series of probabilities, $\Lambda(s) := \sum_{w \in \Sigma^\star} p_w^{-s}$

Memoryless sources, with probabilities (p_i)

$$\Lambda(s) = \frac{1}{1-\lambda(s)} \qquad \text{with} \quad \lambda(s) = \sum_{i=1}^{r} p_i^{-s}$$

Markov chains, defined by – the vector R of initial probabilities (r_i)

– and the transition matrix $P := (p_{i,j})$

$$\Lambda(s) = {}^t\mathbf{1}(I - P(s))^{-1} R(s) \qquad \text{with} \quad P(s) = (p_{i,j}^{-s}), \quad R(s) = (r_i^{-s}).$$

A main analytical object related to any source:

the Dirichlet series of probabilities, $\Lambda(s) := \sum_{w \in \Sigma^\star} p_w^{-s}$

Memoryless sources, with probabilities (p_i)

$$\Lambda(s) = \frac{1}{1 - \lambda(s)} \qquad \text{with} \quad \lambda(s) = \sum_{i=1}^{r} p_i^{-s}$$

Markov chains, defined by – the vector R of initial probabilities (r_i)

– and the transition matrix $P := (p_{i,j})$

$$\Lambda(s) = {}^t\mathbf{1}(I - P(s))^{-1} R(s) \qquad \text{with} \quad P(s) = (p_{i,j}^{-s}), \quad R(s) = (r_i^{-s}).$$

A general dynamical source

$\Lambda(s)$ closely related to $(I - \mathbf{H}_s)^{-1}$

where $\mathbf{H}_s$ is the transfer operator of the dynamical system.

Plan of the talk.

– Presentation of the study

– A general model of source

– Description of the main results

– Description of the methods

What is already known about the mean number of symbol-comparisons?

The `Trie` structure is very well-studied, but only for particular sources:

the so–called simple sources: memoryless or Markov chains.

What is already known about the mean number of symbol-comparisons?

The `Trie` structure is very well-studied, but only for particular sources:

the so–called simple sources: memoryless or Markov chains.

The number of symbols comparaisons used in `QuickSort`, and `QuickSelect`, is already studied by Janson, Fill, Nakama ('06), but only

– in the case of memoryless sources,

– for `QuickSort, QuickMin, QuickMax, QuickRand`

What is already known about the mean number of symbol-comparisons?

The `Trie` structure is very well-studied, but only for particular sources:

the so–called simple sources: memoryless or Markov chains.

The number of symbols comparaisons used in `QuickSort`, and `QuickSelect`, is already studied by Janson, Fill, Nakama ('06), but only

– in the case of memoryless sources,

– for `QuickSort, QuickMin, QuickMax, QuickRand`

Here, we study the mean number of symbol-comparisons,

in the case of a general source and a general algorithm of the class.

– There are precise restrictive hypotheses on the source, and sufficient conditions under which these hypotheses hold.

– We provide a closed form for the constants of the analysis, for any source of the previous type.

– We use different methods, with limited computation...

Case of $\texttt{Trie}(n)$ [CFV 01]

Theorem 1. *For any Λ–tame source, the mean path length T_n of a trie built on n words independently drawn from the source satisfies*

$$T_n \sim \frac{1}{h_{\mathcal{S}}}\, n\, \log n.$$

and involves the entropy $h_{\mathcal{S}}$ of the source $\mathcal{S}$, defined as

$$h_{\mathcal{S}} := \lim_{k\to\infty} \left[\frac{-1}{k} \sum_{w\in\Sigma^k} p_w \log p_w \right],$$

where p_w is the probability that a word begins with prefix w.

Case of $\texttt{QuickSort}(n)$ or $\texttt{BST}(n)$ [CFFV 08]

Theorem 2. *For any Λ–tame source,*
the mean number S_n of symbol comparisons used by $\texttt{QuickSort}(n)$
(or the mean number of symbols comparisons used to built the BST)
on n words of the source satisfies

$$B_n \sim \frac{1}{h_{\mathcal{S}}} n \log^2 n.$$

and involves the entropy $h_{\mathcal{S}}$ of the source $\mathcal{S}$, defined as

$$h_{\mathcal{S}} := \lim_{k \to \infty} \left[\frac{-1}{k} \sum_{w \in \Sigma^k} p_w \log p_w \right],$$

where p_w is the probability that a word begins with prefix w.

Case of QuickSort(n) or BST(n) [CFFV 08]

Theorem 2. *For any Λ–tame source,*
the mean number S_n of symbol comparisons used by `QuickSort`(n)
(or the mean number of symbols comparisons used to built the BST)
on n words of the source satisfies

$$B_n \sim \frac{1}{h_{\mathcal{S}}} n \log^2 n.$$

and involves the entropy $h_{\mathcal{S}}$ of the source $\mathcal{S}$, defined as

$$h_{\mathcal{S}} := \lim_{k\to\infty} \left[\frac{-1}{k} \sum_{w\in\Sigma^k} p_w \log p_w \right],$$

where p_w is the probability that a word begins with prefix w.

Compared to $K_n \sim 2n \log n$, there is an extra factor equal to $1/(2h_{\mathcal{S}}) \log n$

Compared to $T_n \sim (1/h_{\mathcal{S}})\, n \log n$, there is an extra factor of $\log n$.

Case of $\texttt{QuickQuant}_\alpha(n)$ [CFFV 09]

Theorem 3. *For any* Π*–tame source,*
the mean number of symbol comparisons used by $\texttt{QuickQuant}_\alpha(n)$ *satisfies*

$$Q_n^{(\alpha)} \sim \rho_{\mathcal{S}}(\alpha)\, n \qquad \rho_{\mathcal{S}}(\alpha) = \sum_{w \in \Sigma^\star} p_w \, L\left(\frac{|\alpha - \mu_w|}{p_w}\right).$$

$$\mu_w = \frac{1}{2}\left[p_w^{(+)} + p_w^{(-)}\right] = \text{the middle of the fundamental interval}$$

The function L is an even function given by $L(y) = 2[1 + H(y)]$,

$$H(y) = \begin{cases} -(y^+ \log y^+ + \; y^- \log y^-), & \text{if } 0 \le y < 1/2 \\ 0, & \text{if } y = 1/2 \\ y^+(\log|y^+| - \log|y^-|), & \text{if } y > 1/2. \end{cases}$$

$H(y)$ is a modified entropy function expressed with $y^+ := (1/2) + y$, $y^- = (1/2) - y$.

Some particular cases for the constant $\rho_{\mathcal{S}}(\alpha)$.

Constants for `QuickMin` $(\alpha = 0 \to \epsilon = +)$ and `QuickMax` $(\alpha = 1 \to \epsilon = -)$

$$c_{\mathcal{S}}^{(\epsilon)} := 2 \sum_{w \in \Sigma^\star} p_w \left[1 - \frac{p_w^{(\epsilon)}}{p_w} \log \left(1 + \frac{p_w}{p_w^{(\epsilon)}} \right) \right].$$

Some particular cases for the constant $\rho_{\mathcal{S}}(\alpha)$.

Constants for `QuickMin` $(\alpha = 0 \to \epsilon = +)$ and `QuickMax` $(\alpha = 1 \to \epsilon = -)$

$$c_{\mathcal{S}}^{(\epsilon)} := 2 \sum_{w \in \Sigma^\star} p_w \left[1 - \frac{p_w^{(\epsilon)}}{p_w} \log \left(1 + \frac{p_w}{p_w^{(\epsilon)}} \right) \right].$$

Constant for `QuickRand` $\underline{c}_{\mathcal{S}} = \int_0^1 \rho_{\mathcal{S}}(\alpha) d\alpha$

$$\underline{c}_{\mathcal{S}} = \sum_{w \in \Sigma^\star} p_w^2 \left[2 + \frac{1}{p_w} + \sum_{\epsilon = \pm} \left[\log \left(1 + \frac{p_w^{(\epsilon)}}{p_w} \right) - \left(\frac{p_w^{(\epsilon)}}{p_w} \right)^2 \log \left(1 + \frac{p_w}{p_w^{(\epsilon)}} \right) \right] \right],$$

The constants of the analysis for the binary source.

$$h_{\mathcal{B}} = \log 2, \qquad c_{\mathcal{B}}^{(+)} = c_{\mathcal{B}}^{(-)} = c_{\mathcal{B}}^{(\epsilon)}$$

$$c_{\mathcal{B}}^{(\epsilon)} = 4 + 2\sum_{\ell \geq 0} \frac{1}{2^{\ell}} + 2\sum_{\ell \geq 0} \frac{1}{2^{\ell}} \sum_{k=1}^{2^{\ell}-1} \left[1 - k \log\left(1 + \frac{1}{k}\right)\right]$$

The constants of the analysis for the binary source.

$$h_{\mathcal{B}} = \log 2, \qquad c_{\mathcal{B}}^{(+)} = c_{\mathcal{B}}^{(-)} = c_{\mathcal{B}}^{(\epsilon)}$$

$$c_{\mathcal{B}}^{(\epsilon)} = 4 + 2\sum_{\ell \geq 0} \frac{1}{2^{\ell}} + 2\sum_{\ell \geq 0} \frac{1}{2^{\ell}} \sum_{k=1}^{2^{\ell}-1} \left[1 - k \log\left(1 + \frac{1}{k}\right)\right]$$

$$\underline{c}_{\mathcal{B}} = \frac{14}{3} + 2\sum_{\ell=0}^{\infty} \frac{1}{2^{2\ell}} \sum_{k=1}^{2^{\ell}-1} \left[k + 1 + \log(k+1) - k^2 \log\left(1 + \frac{1}{k}\right)\right]$$

The constants of the analysis for the binary source.

$$h_{\mathcal{B}} = \log 2, \qquad c_{\mathcal{B}}^{(+)} = c_{\mathcal{B}}^{(-)} = c_{\mathcal{B}}^{(\epsilon)}$$

$$c_{\mathcal{B}}^{(\epsilon)} = 4 + 2\sum_{\ell \geq 0} \frac{1}{2^{\ell}} + 2\sum_{\ell \geq 0} \frac{1}{2^{\ell}} \sum_{k=1}^{2^{\ell}-1} \left[1 - k \log\left(1 + \frac{1}{k}\right)\right]$$

$$\underline{c}_{\mathcal{B}} = \frac{14}{3} + 2\sum_{\ell=0}^{\infty} \frac{1}{2^{2\ell}} \sum_{k=1}^{2^{\ell}-1} \left[k + 1 + \log(k+1) - k^2 \log\left(1 + \frac{1}{k}\right)\right]$$

Numerically, $c_{\mathcal{B}}^{(\epsilon)} = 5.27937......,$ $\qquad c_{\mathcal{B}} = 8.20731......$

To be compared to the constants of the number of key–comparisons

$$\kappa = 2 \qquad \text{or} \qquad \kappa = 3$$

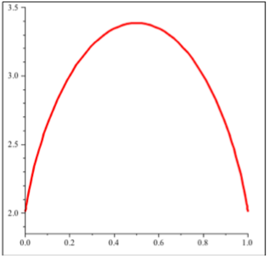
The curve $\alpha \mapsto \rho(\alpha)$ is a fractal deformation of $\alpha \mapsto \kappa(\alpha)$

$\kappa(\alpha)$ the constant of the number of key–comparisons in $\texttt{QuickQuant}_\alpha$

The curve $\alpha \mapsto \rho(\alpha)$ is a fractal deformation of $\alpha \mapsto \kappa(\alpha)$

$\kappa(\alpha)$ the constant of the number of key–comparisons in $\texttt{QuickQuant}_\alpha$

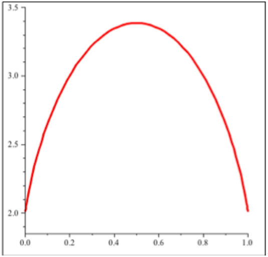
The plot of $\alpha \mapsto \kappa(\alpha)$



The curve $\alpha \mapsto \rho(\alpha)$ is a fractal deformation of $\alpha \mapsto \kappa(\alpha)$

$\kappa(\alpha)$ the constant of the number of key–comparisons in $\texttt{QuickQuant}_\alpha$

The plot of $\alpha \mapsto \kappa(\alpha)$



..... To be compared

to the plots of $\alpha \mapsto \rho(\alpha)$

for four memoryless sources

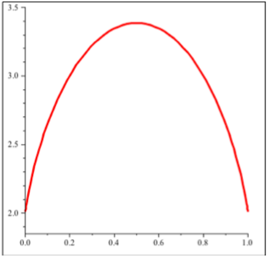
– three unbiased, $r = 2, 3, 4$

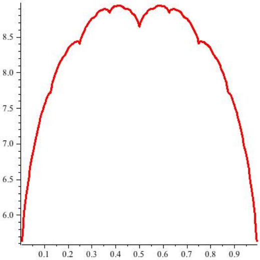
– one biased (1/3, 2/3)

The curve $\alpha \mapsto \rho(\alpha)$ is a fractal deformation of $\alpha \mapsto \kappa(\alpha)$

$\kappa(\alpha)$ the constant of the number of key–comparisons in $\mathtt{QuickQuant}_\alpha$

The plot of $\alpha \mapsto \kappa(\alpha)$





..... To be compared

to the plots of $\alpha \mapsto \rho(\alpha)$

for four memoryless sources

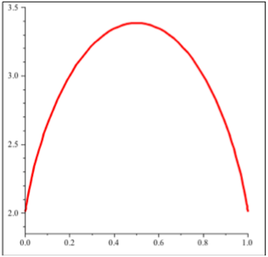
– three unbiased, $r = 2, 3, 4$

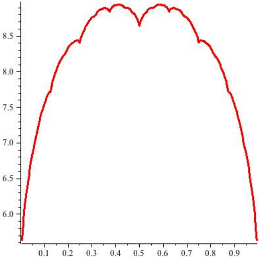
– one biased (1/3, 2/3)

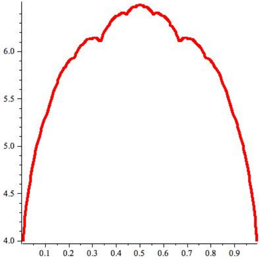
The curve $\alpha \mapsto \rho(\alpha)$ is a fractal deformation of $\alpha \mapsto \kappa(\alpha)$

$\kappa(\alpha)$ the constant of the number of key–comparisons in $\texttt{QuickQuant}_\alpha$

The plot of $\alpha \mapsto \kappa(\alpha)$







..... To be compared

to the plots of $\alpha \mapsto \rho(\alpha)$

for four memoryless sources

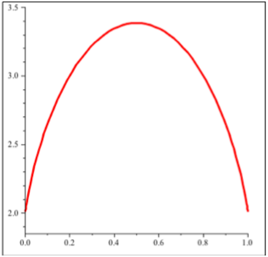
– three unbiased, $r = 2, 3, 4$

– one biased (1/3, 2/3)

The curve $\alpha \mapsto \rho(\alpha)$ is a fractal deformation of $\alpha \mapsto \kappa(\alpha)$

$\kappa(\alpha)$ the constant of the number of key–comparisons in $\texttt{QuickQuant}_\alpha$

The plot of $\alpha \mapsto \kappa(\alpha)$



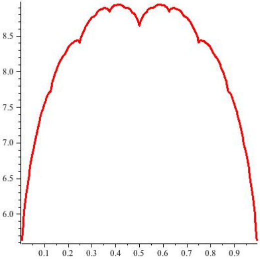
..... To be compared

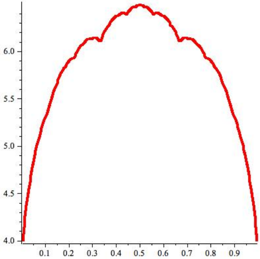
to the plots of $\alpha \mapsto \rho(\alpha)$

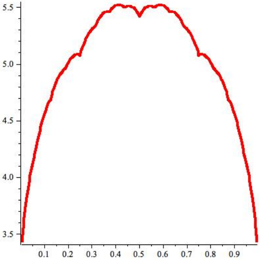
for four memoryless sources

– three unbiased, $r = 2, 3, 4$

– one biased (1/3, 2/3)



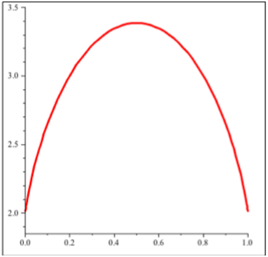




The curve $\alpha \mapsto \rho(\alpha)$ is a fractal deformation of $\alpha \mapsto \kappa(\alpha)$

$\kappa(\alpha)$ the constant of the number of key–comparisons in $\texttt{QuickQuant}_\alpha$

The plot of $\alpha \mapsto \kappa(\alpha)$



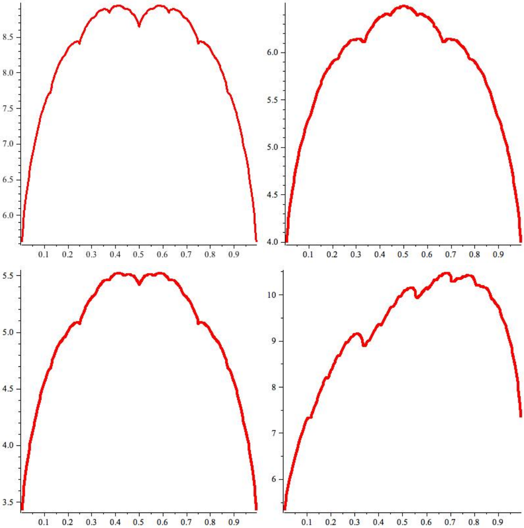
..... To be compared

to the plots of $\alpha \mapsto \rho(\alpha)$

for four memoryless sources

– three unbiased, $r = 2, 3, 4$

– one biased (1/3, 2/3)



What about the function $\alpha \mapsto \rho_{\mathcal{S}}(\alpha)$?

In the case where $\mathcal{S} =$ the unbiased memoryless source with r symbols.

$\rho_{\mathcal{S}}$ is denoted by ρ_r.

If r is odd, ρ_r is maximum at $\alpha = 1/2$ (case of `QuickMed`)

If r is even, this is not true. For which value of α, $\rho_r(\alpha)$ is maximum?

What about the function $\alpha \mapsto \rho_{\mathcal{S}}(\alpha)$?

In the case where $\mathcal{S} =$ the unbiased memoryless source with r symbols.

$\rho_{\mathcal{S}}$ is denoted by ρ_r.

If r is odd, ρ_r is maximum at $\alpha = 1/2$ (case of `QuickMed`)

If r is even, this is not true. For which value of α, $\rho_r(\alpha)$ is maximum?

Is ρ_r differentiable? Is it Hölder?

What about the function $\alpha \mapsto \rho_{\mathcal{S}}(\alpha)$?

In the case where $\mathcal{S} =$ the unbiased memoryless source with r symbols.

$\rho_{\mathcal{S}}$ is denoted by ρ_r.

If r is odd, ρ_r is maximum at $\alpha = 1/2$ (case of `QuickMed`)

If r is even, this is not true. For which value of α, $\rho_r(\alpha)$ is maximum?

Is ρ_r differentiable? Is it Hölder?

When $r \to \infty$, $\rho_r(\alpha) \to 2[1 + h(\alpha)]$

$=$ the constant which intervenes in the mean number of key–comparisons.

($h(.)$ is the entropy function)

Plan of the talk.

- Presentation of the study
- A general model of source
- Description of the main results
- Description of the methods

Three main steps for the analysis
of the mean number S_n of symbol comparisons

Three main steps for the analysis
of the mean number S_n of symbol comparisons

(1) First step (algebraic).

The Poisson model $\mathcal{P}_Z$ deals with a variable number N of keys:
N is a random variable which follows a Poisson law of parameter Z.

We first obtain nice expressions for the mean number $\widetilde{S}_Z$ ....

Three main steps for the analysis
of the mean number S_n of symbol comparisons

(1) First step (algebraic).

The Poisson model $\mathcal{P}_Z$ deals with a variable number N of keys:
N is a random variable which follows a Poisson law of parameter Z.

We first obtain nice expressions for the mean number $\widetilde{S}_Z$ ....

(2) Second step (algebraic).

It is then possible to return to the model where the number of keys is fixed.
We obtain a nice exact formula for S_n ....

from which it is not easy to obtain the asymptotics...

Three main steps for the analysis
of the mean number S_n of symbol comparisons

(1) First step (algebraic).

The Poisson model $\mathcal{P}_Z$ deals with a variable number N of keys:
N is a random variable which follows a Poisson law of parameter Z.

We first obtain nice expressions for the mean number $\widetilde{S}_Z$ ....

(2) Second step (algebraic).

It is then possible to return to the model where the number of keys is fixed.
We obtain a nice exact formula for S_n ....

from which it is not easy to obtain the asymptotics...

(3) Third step (analytic).

Then, the Rice formula provides the asymptotics of S_n ($n \to \infty$),

as soon as the source is "tame"

Λ–tame for `QuickSort` and `Tries` , Π–tame for `QuickSelect`

(1) Dealing with the Poisson Model $\mathcal{P}_Z$

– The number N of keys is drawn according to the Poisson law

$$\Pr[N = n] = e^{-Z} \frac{Z^n}{n!},$$

– Then, the N words are independently drawn from the source.

(1) Dealing with the Poisson Model $\mathcal{P}_Z$

– The number N of keys is drawn according to the Poisson law

$$\Pr[N = n] = e^{-Z}\frac{Z^n}{n!},$$

– Then, the N words are independently drawn from the source.

Two nice properties of the Poisson model.

about the number $N_{[a,b]}$ of words $M(v)$ with $v \in [a, b]$

(i) $N_{[a,b]}$ follows a Poisson law of parameter $Z(b-a)$.

(ii) For $[a, b] \cap [c, d] = \emptyset$ the variables $N_{[a,b]}$ and $N_{[c,d]}$ are independent.

The path-length of a `Trie` equals

$$\sum_{w \in \Sigma^\star} \underline{N}_w \qquad \text{with} \quad \underline{N}_w = \mathbf{1}_{[N_w \geq 2]} \cdot N_w,$$

where N_w is the number of keys which begin with prefix w.

The mean path-length in the $\mathcal{P}_Z$ model is then

$$\widetilde{T}_Z = \sum_{w \in \Sigma^\star} Z p_w [1 - e^{-Z p_w}].$$

(1) Dealing with the Poisson Model $\mathcal{P}_Z$

The mean number $\widetilde{S}_Z$ of symbol comparisons for an algorithm $\mathcal{A}$ is

$$\widetilde{S}_Z = \int_{\mathcal{T}} [\gamma(u,t) + 1]\, \widetilde{\pi}_Z(u,t)\, du\, dt$$

(1) Dealing with the Poisson Model $\mathcal{P}_Z$

The mean number $\widetilde{S}_Z$ of symbol comparisons for an algorithm $\mathcal{A}$ is

$$\widetilde{S}_Z = \int_{\mathcal{T}} [\gamma(u,t) + 1]\, \widetilde{\pi}_Z(u,t)\, du\, dt$$

where $\mathcal{T} := \{(u,t), \quad 0 \le u \le t \le 1\}$ is the unit triangle

(1) Dealing with the Poisson Model $\mathcal{P}_Z$

The mean number $\widetilde{S}_Z$ of symbol comparisons for an algorithm $\mathcal{A}$ is

$$\widetilde{S}_Z = \int_{\mathcal{T}} [\gamma(u,t) + 1]\, \widetilde{\pi}_Z(u,t)\, du\, dt$$

where $\mathcal{T} := \{(u,t), \quad 0 \le u \le t \le 1\}$ is the unit triangle

$\gamma(u,t)$:= coincidence between $M(u)$ and $M(t)$

(1) Dealing with the Poisson Model $\mathcal{P}_Z$

The mean number $\widetilde{S}_Z$ of symbol comparisons for an algorithm $\mathcal{A}$ is

$$\widetilde{S}_Z = \int_{\mathcal{T}} [\gamma(u,t) + 1]\, \widetilde{\pi}_Z(u,t)\, du\, dt$$

where $\mathcal{T} := \{(u,t), \quad 0 \le u \le t \le 1\}$ is the unit triangle

$\gamma(u,t)$:= coincidence between $M(u)$ and $M(t)$

$\widetilde{\pi}_Z(u,t)\, du\, dt$:= Mean number of key-comparisons between $M(u')$ and $M(t')$ with $u' \in [u, u+du]$ and $t' \in [t-dt, t]$ performed by the algorithm $\mathcal{A}$.

(1) Dealing with the Poisson Model $\mathcal{P}_Z$

The mean number $\widetilde{S}_Z$ of symbol comparisons for an algorithm $\mathcal{A}$ is

$$\widetilde{S}_Z = \int_{\mathcal{T}} [\gamma(u,t) + 1]\, \widetilde{\pi}_Z(u,t)\, du\, dt$$

where $\mathcal{T} := \{(u,t), \quad 0 \le u \le t \le 1\}$ is the unit triangle

$\gamma(u,t) :=$ coincidence between $M(u)$ and $M(t)$

$\widetilde{\pi}_Z(u,t)\, du\, dt :=$ Mean number of key-comparisons between $M(u')$ and $M(t')$ with $u' \in [u, u+du]$ and $t' \in [t-dt, t]$ performed by the algorithm $\mathcal{A}$.

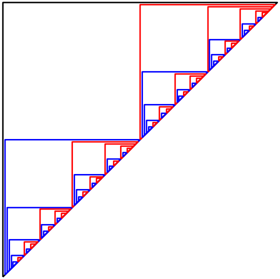
An (easy) alternative expression for $\widetilde{S}_Z$

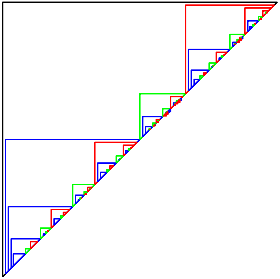
$$\widetilde{S}_Z = \sum_{w \in \Sigma^\star} \int_{\mathcal{T}_w} \widetilde{\pi}_Z(u,t)\, du\, dt$$

It involves the fundamental triangles

and separates the rôles of the source and the algorithm.

Instances of fundamental triangles.





On the left:
memoryless source on $\{a, b\}$
$p_a = 1/2$, $p_b = 1/2$

On the right :
memoryless source on $\{a, b, c\}$
$p_a = 1/2$, $p_b = 1/6$, $p_c = 1/3$

Study of the key probability $\widetilde{\pi}_Z(u,t)$ of `QuickX` (`X= Sort` or `X= Quant`$_\alpha$.)

Related question : When does `QuickX` compare two keys $M(u)$ and $M(t)$?

Study of the key probability $\widetilde{\pi}_Z(u,t)$ of `QuickX` (`X= Sort` or `X= Quant`$_\alpha$.)

Related question : When does `QuickX` compare two keys $M(u)$ and $M(t)$?

In `QuickSort`, $M(u)$ and $M(t)$ are compared
iff the first pivot chosen in $\{M(v), v \in [u,t]\}$ is $M(u)$ or $M(t)$

Study of the key probability $\widetilde{\pi}_Z(u,t)$ of `QuickX` (`X= Sort` or `X= Quant`$_\alpha$.)

Related question : When does `QuickX` compare two keys $M(u)$ and $M(t)$?

In `QuickSort`, $M(u)$ and $M(t)$ are compared
iff the first pivot chosen in $\{M(v), v \in [u,t]\}$ is $M(u)$ or $M(t)$

In `QuickMin`, $M(u)$ and $M(t)$ are compared
iff the first pivot chosen in $\{M(v), v \in [0,t]\}$ is $M(u)$ or $M(t)$

Study of the key probability $\widetilde{\pi}_Z(u,t)$ of QuickX (X= Sort or X= Quant_α.)

Related question : When does QuickX compare two keys $M(u)$ and $M(t)$?

In QuickSort, $M(u)$ and $M(t)$ are compared

iff the first pivot chosen in $\{M(v), v \in [u,t]\}$ is $M(u)$ or $M(t)$

In QuickMin, $M(u)$ and $M(t)$ are compared

iff the first pivot chosen in $\{M(v), v \in [0,t]\}$ is $M(u)$ or $M(t)$

In QuickMax, $M(u)$ and $M(t)$ are compared

iff the first pivot chosen in $\{M(v), v \in [u,1]\}$ is $M(u)$ or $M(t)$

Study of the key probability $\widetilde{\pi}_Z(u,t)$ of QuickX (X= Sort or X= Quant$_\alpha$.)

Related question : When does QuickX compare two keys $M(u)$ and $M(t)$?

In QuickSort, $M(u)$ and $M(t)$ are compared
iff the first pivot chosen in $\{M(v), v \in [u,t]\}$ is $M(u)$ or $M(t)$

In QuickMin, $M(u)$ and $M(t)$ are compared
iff the first pivot chosen in $\{M(v), v \in [0,t]\}$ is $M(u)$ or $M(t)$

In QuickMax, $M(u)$ and $M(t)$ are compared
iff the first pivot chosen in $\{M(v), v \in [u,1]\}$ is $M(u)$ or $M(t)$

And for QuickQuant$_\alpha$? Not so easy!

Study of the key probability $\widetilde{\pi}_Z(u,t)$ of `QuickX` (`X= Sort` or `X= Quant`$_\alpha$.)

Related question : When does `QuickX` compare two keys $M(u)$ and $M(t)$?

In `QuickSort`, $M(u)$ and $M(t)$ are compared
iff the first pivot chosen in $\{M(v), v \in [u,t]\}$ is $M(u)$ or $M(t)$

In `QuickMin`, $M(u)$ and $M(t)$ are compared
iff the first pivot chosen in $\{M(v), v \in [0,t]\}$ is $M(u)$ or $M(t)$

In `QuickMax`, $M(u)$ and $M(t)$ are compared
iff the first pivot chosen in $\{M(v), v \in [u,1]\}$ is $M(u)$ or $M(t)$

And for `QuickQuant`$_\alpha$? Not so easy!

The idea is to compare `QuickQuant`
with a dual algorithm, the `QuickVal` algorithm.

A parenthesis – Presentation of `QuickVal`

The `QuickVal` algorithm is the dual algorithm of `QuickSelect`,

A parenthesis – Presentation of QuickVal

The QuickVal algorithm is the dual algorithm of QuickSelect,

QuickVal (n, a, A). : returns the rank of the element a in $B = A \cup \{a\}$

    $B := A \cup \{a\}$
    QV (n, a, B);

QV (n, a, B).

    Choose a pivot in B;
    $(k, B_-, B_+) :=$ Partition(B);
    If $a =$ pivot then QV $:= k$
        else if $a <$ pivot then QV := QV $(k-1, a, B_-)$
            else QV := $k+$ QV $(n-k, a, B_+)$;

A parenthesis – Presentation of QuickVal

The QuickVal algorithm is the dual algorithm of QuickSelect,

```
QuickVal (n, a, A). : returns the rank of the element a in B = A ∪ {a}
    B := A ∪ {a}
    QV (n, a, B);

QV (n, a, B).
    Choose a pivot in B;
    (k, B−, B+) := Partition(B);
    If a = pivot then QV := k
                else if a < pivot then QV := QV (k − 1, a, B−)
                                  else QV := k+ QV (n − k, a, B+);
```

$\texttt{QuickVal}_\alpha$:= the algorithm where the key of interest is the word $M(\alpha)$

Comparison between $\texttt{QuickVal}_\alpha$ and $\texttt{QuickQuant}_\alpha$

$\texttt{QuickVal}_\alpha$:= the algorithm where the key of interest is the word $M(\alpha)$

There are two facts

– Since the rank of $M(\alpha)$ amongst n keys is close to αn (for $n \to \infty$), the probabilistic behaviours of the two algorithms are close

Comparison between $\texttt{QuickVal}_\alpha$ and $\texttt{QuickQuant}_\alpha$

$\texttt{QuickVal}_\alpha$:= the algorithm where the key of interest is the word $M(\alpha)$

There are two facts

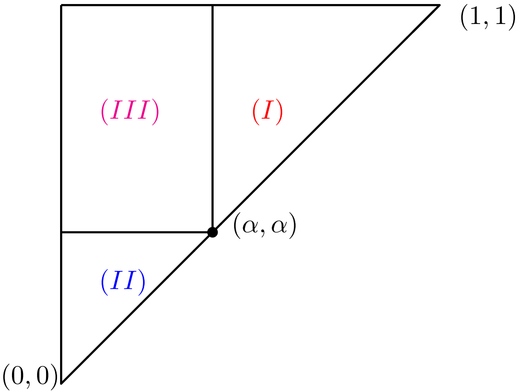
– Since the rank of $M(\alpha)$ amongst n keys is close to αn (for $n \to \infty$), the probabilistic behaviours of the two algorithms are close

– The $\texttt{QuickVal}_\alpha$ algorithm is easy to deal with since

$M(u)$ and $M(t)$ are compared in $\texttt{QuickVal}_\alpha$
iff the first pivot chosen in $\{M(v), v \in [x, y]\}$ is $M(u)$ or $M(t)$.

Here, the interval $[x, y]$ is the smallest interval that contains u, t and α.
this means : $x = \min(\alpha, u), \qquad y = \max(\alpha, t)$

The three domains for the definition of the interval $[x, y]$, the smallest interval that contains u, t, α



$$[x(u,t), y(u,t)] := \begin{cases} [\alpha, t] & \text{if } u > \alpha & (I) & \sim \texttt{QuickMin} \\ [u, \alpha] & \text{if } t < \alpha & (II) & \sim \texttt{QuickMax} \\ [u, t] & \text{if } u < \alpha < t & (III) & \sim \texttt{QuickSort} \end{cases}$$

In summary, the algorithm `QuickX` with X= `Sort` or X= $\texttt{Val}_\alpha$,

compares two words $M(u)$ and $M(t)$

iff $M(u)$ or $M(t)$ is chosen as the first pivot in $\{M(v), v \in [x, y]\}$ with

$[x, y] = [u, t]$ (`QuickSort`), $[x, y] = [\min(\alpha, u), \max(\alpha, t)]$ ($\texttt{QuickVal}_\alpha$)

In summary, the algorithm QuickX with X= Sort or X= Val_α,

compares two words $M(u)$ and $M(t)$

iff $M(u)$ or $M(t)$ is chosen as the first pivot in $\{M(v), v \in [x, y]\}$ with

$[x, y] = [u, t]$ (QuickSort), $[x, y] = [\min(\alpha, u), \max(\alpha, t)]$ (QuickVal_α)

In the Poisson model, $\widetilde{\pi}_Z(u, t)\, du\, dt = Zdu \cdot Zdt \cdot \widetilde{\mathbb{E}}_Z \left[\frac{2}{2 + N_{[x,y]}}\right]$

$\widetilde{\pi}_Z(u, t) = 2\, Z^2\, f_1(Z(y - x))$ with $f_1(\theta) := \theta^{-2}\,[e^{-\theta} - 1 + \theta]$

In summary, the algorithm QuickX with X= Sort or X= Val_α,
compares two words $M(u)$ and $M(t)$
iff $M(u)$ or $M(t)$ is chosen as the first pivot in $\{M(v), v \in [x, y]\}$ with
$[x, y] = [u, t]$ (QuickSort), $[x, y] = [\min(\alpha, u), \max(\alpha, t)]$ (QuickVal_α)

In the Poisson model, $\widetilde{\pi}_Z(u, t)\, du\, dt = Z du \cdot Z dt \cdot \widetilde{\mathbb{E}}_Z \left[\frac{2}{2 + N_{[x,y]}} \right]$

$$\widetilde{\pi}_Z(u, t) = 2\, Z^2\, f_1(Z(y - x)) \qquad \text{with} \qquad f_1(\theta) := \theta^{-2}\, [e^{-\theta} - 1 + \theta]$$

With $f_0(\theta) = \theta(1 - e^{-\theta})$, $f_1(\theta) := \theta^{-2}\, [e^{-\theta} - 1 + \theta]$,
Final expressions of the mean cost for Trie and QuickX in the $\mathcal{P}_Z$ model

$$\widetilde{T}_Z = \sum_{w \in \Sigma^\star} f_0(Z p_w) \qquad \widetilde{S}_Z = 2Z^2 \sum_{w \in \Sigma^\star} \int_{\mathcal{T}_w} f_1(Z(y - x)) du dt,$$

(2) Return to the model where the number n of keys is fixed.

Expanding f_0, f_1, $\quad f_0(\theta) = \theta[1 - e^{-\theta}], \qquad f_1(\theta) := \theta^{-2}\,[e^{-\theta} - 1 + \theta],$

and using the transfer between the two models $\quad \frac{S_n}{n!} = [Z^n]\left(e^Z \cdot \widetilde{S}_Z\right)$

there is an exact formula for S_n

$$S_n = 2\sum_{k=2}^{n}(-1)^k \binom{n}{k} \varpi(-k)$$

which involves the series ϖ at integer values $-k$.

(2) Return to the model where the number n of keys is fixed.

Expanding f_0, f_1, $\quad f_0(\theta) = \theta[1 - e^{-\theta}], \qquad f_1(\theta) := \theta^{-2}\,[e^{-\theta} - 1 + \theta]$,

and using the transfer between the two models $\qquad \dfrac{S_n}{n!} = [Z^n]\left(e^Z \cdot \widetilde{S}_Z\right)$

there is an exact formula for S_n

$$S_n = 2\sum_{k=2}^{n}(-1)^k\binom{n}{k}\varpi(-k)$$

which involves the series ϖ at integer values $-k$.

The series $\varpi(s)$ is of Dirichlet type, and depends both

- on the algorithm (via the function f_0 or f_1 and interval $[x, y]$)
- on the source (via the fundamental triangles $\mathcal{T}_w$)

In the three cases, an exact formula for S_n ....

$$S_n = 2\sum_{k=2}^{n}(-1)^k \binom{n}{k} \varpi(-k)$$

...which involves the series ϖ at integer values $-k$.

For the mean path length (Trie or BST),

$\varpi(s)$ is closely related to the Dirichlet series of the probabilities,

$$\varpi_T(s) = -s\Lambda(s) \qquad \varpi_B(s) = 2\frac{\Lambda(s)}{s(s+1)} \qquad \text{where} \quad \Lambda(s) := \sum_{w\in\Sigma^\star} p_w^{-s}$$

For QuickVal, the expression is more involved,

$$\varpi_Q(s) = 2\sum_{w\in\Sigma^\star}\int_{\mathcal{T}_w}(y-x)^{-(s+2)}\, du\, dt$$

(3) Asymptotic analysis.

Then, the residue formula transforms the sum into an integral:

$$S_n = \sum_{k=2}^{n} (-1)^k \binom{n}{k} \varpi(-k) = \frac{1}{2i\pi} \int_{d-i\infty}^{d+i\infty} \varpi(s) \frac{n!}{s(s+1)\dots(s+n)} ds,$$

with $-2 < d < -1$.

We shift the integral on the right, and there is one singularity at $s = -1$.

What is the behaviour of $\varpi(s)$ near $\Re s = -1$?

We compare it to other Dirichlet series:

(3) Asymptotic analysis.

Then, the residue formula transforms the sum into an integral:

$$S_n = \sum_{k=2}^{n} (-1)^k \binom{n}{k} \varpi(-k) = \frac{1}{2i\pi} \int_{d-i\infty}^{d+i\infty} \varpi(s) \frac{n!}{s(s+1)\dots(s+n)} ds,$$

with $-2 < d < -1$.

We shift the integral on the right, and there is one singularity at $s = -1$.

What is the behaviour of $\varpi(s)$ near $\Re s = -1$?

We compare it to other Dirichlet series:

– For `Trie`, `BST`, $\varpi_T(s), \varpi_B(s)$ are related to $\Lambda(s)$.

$$\Lambda(s) := \sum_{w \in \Sigma^\star} p_w^{-s},$$

$p_w = \Pr$ [a word begins with w],

– For `QuickVal`, $\varpi_Q(s)$ is related to $\Pi(s)$.

$$\Pi(s) = \sum_{k \geq 0} \pi_k^{-s}.$$

$\pi_k = \sup \{p_w;\ w \in \Sigma^k\}$

Study of `QuickVal` and `QuickQuant`

A function is "tame" in a region $\mathcal{R}$

if it is analytic and of polynomial growth for $|s| \to \infty$

Study of `QuickVal` and `QuickQuant`

A function is "tame" in a region $\mathcal{R}$

if it is analytic and of polynomial growth for $|s| \to \infty$

A source S is Π–tame if $\Pi(s)$ is tame on $\{\Re s < \sigma_1\}$ with $\sigma_1 > -1$.

A sufficient condition is $\pi_k \leq Ak^{-\gamma}$ with $\gamma > 1$

Most of the "natural" sources are Π–tame !

Study of `QuickVal` and `QuickQuant`

A function is "tame" in a region $\mathcal{R}$

if it is analytic and of polynomial growth for $|s| \to \infty$

A source S is Π–tame if $\Pi(s)$ is tame on $\{\Re s < \sigma_1\}$ with $\sigma_1 > -1$.

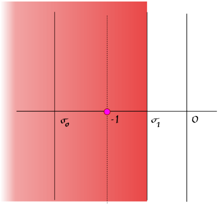
A sufficient condition is $\pi_k \leq A k^{-\gamma}$ with $\gamma > 1$

Most of the "natural" sources are Π–tame !

In this case,

(1) $\varpi(s)$ is also tame in $\{\Re s < \sigma_1\}$.

(2) The function $\alpha \mapsto \rho_{\mathcal{S}}(\alpha)$ is Hölder of exponent $\sigma_1 + 1$



(1) $\Rightarrow$ analysis of `QuickVal`

(2) $\Rightarrow$ analysis of `QuickQuant`

A nice expression for $\rho_{\mathcal{S}}(\alpha) = \sum_{w \in \Sigma^\star} \int_{\mathcal{T}_w} [\max(\alpha, t) - \min(\alpha, u)]^{-1} du dt$

Study of the mean path length of `Trie` and `BST`

$$\varpi_T(s) = -s\Lambda(s), \qquad \varpi_B(s) = 2\frac{\Lambda(s)}{s(s+1)} \quad \text{where} \quad \Lambda(s) := \sum_{w\in\Sigma^\star} p_w^{-s}$$

For any source, $\Lambda(s)$ has a singularity at $s = -1$.

Study of the mean path length of `Trie` and `BST`

$$\varpi_T(s) = -s\Lambda(s), \qquad \varpi_B(s) = 2\frac{\Lambda(s)}{s(s+1)} \quad \text{where} \quad \Lambda(s) := \sum_{w \in \Sigma^\star} p_w^{-s}$$

For any source, $\Lambda(s)$ has a singularity at $s = -1$.

A source is Λ–tame if

Study of the mean path length of Trie and BST

$$\varpi_T(s) = -s\Lambda(s), \qquad \varpi_B(s) = 2\frac{\Lambda(s)}{s(s+1)} \quad \text{where} \quad \Lambda(s) := \sum_{w\in\Sigma^\star} p_w^{-s}$$

For any source, $\Lambda(s)$ has a singularity at $s = -1$.

A source is Λ–tame if

(1) the dominant singularity of $\Lambda(s)$ is located at $s = -1$,

this is a simple pôle, whose residue equals $1/h_{\mathcal{S}}$.

Study of the mean path length of Trie and BST

$$\varpi_T(s) = -s\Lambda(s), \qquad \varpi_B(s) = 2\frac{\Lambda(s)}{s(s+1)} \quad \text{where} \quad \Lambda(s) := \sum_{w\in\Sigma^\star} p_w^{-s}$$

For any source, $\Lambda(s)$ has a singularity at $s = -1$.

A source is Λ–tame if

(1) the dominant singularity of $\Lambda(s)$ is located at $s = -1$,

this is a simple pôle, whose residue equals $1/h_{\mathcal{S}}$.

In this case, there is, at $s = -1$

a double pôle for $\dfrac{\varpi_T(s)}{s+1}$, a triple pôle for $\dfrac{\varpi_B(s)}{s+1}$

Study of the mean path length of `Trie` and `BST`

$$\varpi_T(s) = -s\Lambda(s), \qquad \varpi_B(s) = 2\frac{\Lambda(s)}{s(s+1)} \quad \text{where} \quad \Lambda(s) := \sum_{w \in \Sigma^\star} p_w^{-s}$$

For any source, $\Lambda(s)$ has a singularity at $s = -1$.

A source is Λ–tame if

(1) the dominant singularity of $\Lambda(s)$ is located at $s = -1$,

this is a simple pôle, whose residue equals $1/h_{\mathcal{S}}$.

In this case, there is, at $s = -1$

a double pôle for $\dfrac{\varpi_T(s)}{s+1}$, a triple pôle for $\dfrac{\varpi_B(s)}{s+1}$

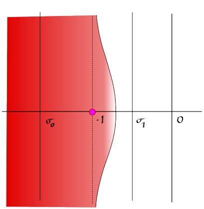
(2) $\Lambda(s)$ is tame on the right of the line $\Re s = -1$

(useful for shifting on the right...)

Different possible regions on the right of $\Re s = -1$ where $\Lambda(s)$ is tame.

Different possible regions on the right of $\Re s = -1$ where $\Lambda(s)$ is tame.

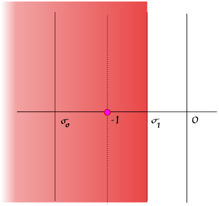
Different possible regions on the right of $\Re s = -1$ where $\Lambda(s)$ is tame.



Situation I

Hyperbolic region

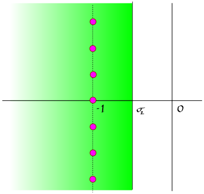
Arithmetic condition



Situation II

Vertical strip

Geometric condition

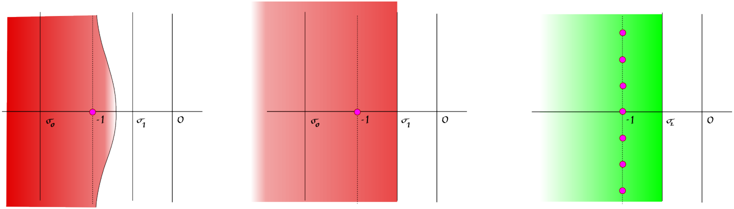


Situation III

Vertical strip with holes

Periodicity condition

Different possible regions on the right of $\Re s = -1$ where $\Lambda(s)$ is tame.



Situation I	Situation II	Situation III
Hyperbolic region	Vertical strip	Vertical strip with holes
Arithmetic condition	Geometric condition	Periodicity condition

For dynamical sources, we provide sufficient conditions (of geometric or arithmetic type), under which these behaviours hold.

For a memoryless source,

– the arithmetic condition is based on the approximability of ratios $\log p_i / \log p_j$

– the situation (II) is not possible

Conclusions.

— For any Λ–tame source,

$$T_n \sim \frac{1}{h_{\mathcal{S}}}\, n\, \log n \quad (\texttt{Trie}), \qquad B_n \sim \frac{1}{h_{\mathcal{S}}}\, n\, \log^2 n \quad (\texttt{BST})$$

Conclusions.

— For any Λ–tame source,

$$T_n \sim \frac{1}{h_{\mathcal{S}}}\, n\, \log n \quad (\texttt{Trie}), \qquad B_n \sim \frac{1}{h_{\mathcal{S}}}\, n\, \log^2 n \quad (\texttt{BST})$$

— It is easy to adapt our results to the intermittent sources, which emits "long" sequences of the same symbols. In this case,

$$T_n = \Theta(n \log^2 n). \quad (\texttt{Trie}) \qquad B_n = \Theta(n \log^3 n), \quad (\texttt{BST})$$

Conclusions.

— For any Λ–tame source,

$$T_n \sim \frac{1}{h_{\mathcal{S}}}\, n \,\log n \quad (\texttt{Trie}), \qquad B_n \sim \frac{1}{h_{\mathcal{S}}}\, n \,\log^2 n \quad (\texttt{BST})$$

— It is easy to adapt our results to the intermittent sources, which emits "long" sequences of the same symbols. In this case,

$$T_n = \Theta(n \log^2 n). \quad (\texttt{Trie}) \qquad B_n = \Theta(n \log^3 n), \quad (\texttt{BST})$$

— For any reasonable source, $Q_n = \Theta(n)$ (`QuickQuant`).

Long term research projects...

— Revisit the complexity results of the main classical algorithms, and take into account the number of symbol-comparisons...

instead of the number of key-comparisons.

Long term research projects...

— Revisit the complexity results of the main classical algorithms,
and take into account the number of symbol-comparisons...
instead of the number of key-comparisons.

— Provide a sharp "analytic" classification of sources:
Transfer probabilistic properties of sources into analytical properties of $\Lambda(s)$.