Towards a Realistic Analysis of Some Popular Sorting Algorithms

Julien Clément, Thu Hien Nguyen Thi, Brigitte Vallée

Combinatorics, Probability and Computing (20XX) 00, 1–**??**.
doi:

Realistic analysis of some popular sorting algorithms

J. CLÉMENT, T. H. NGUYEN THI and B. VALLÉE[†]

Université de Caen/ENSICAEN/CNRS – GREYC – Caen, France

Received

We describe a general framework for realistic analysis of sorting algorithms, and we apply it to the average-case analysis of three basic sorting algorithms (*QuickSort*, *InsertionSort*, *BubbleSort*). Usually, the analysis deals with the mean number of key comparisons, but, here, we view keys as words produced by the same source, which are compared via their symbols in the lexicographic order. The "realistic" cost of the algorithm is now the total number of symbol comparisons performed by the algorithm, and, in this context, the average-case analysis aims to provide estimates for the mean number of symbol comparisons used by the algorithm. For sorting algorithms, and with respect to key comparisons, the average-case complexity of *QuickSort* is asymptotic to $2n \log n$, *InsertionSort* to $n^2/4$ and *BubbleSort* to $n^2/2$. With respect to symbol comparisons, we prove that their average-case complexity becomes $\Theta(n \log^2 n), \Theta(n^2), \Theta(n^2 \log n)$. In these three cases, we describe the dominant constants which exhibit the probabilistic behaviour of the source (namely, entropy and coincidence) with respect to the algorithm.

Introduction

There are two main classes of sorting and searching algorithms: the first class gathers the algorithms which deal with keys, while the algorithms of the second class deal with words (or strings). Of course, any data is represented inside a computer as a sequence of bits (that is a binary string). However, the point of view is different: the key is viewed as a "whole", and its precise representation is not taken into account, whereas the structure of a word, as a sequence of symbols, is essential in text algorithms. Hence, for basic algorithms of the first class (sorting, searching), the unit operation is the comparison between keys, whereas for text algorithms of the second class, comparisons between symbols are considered.

There exist two important drawbacks to this usual point of view. First, it is difficult to compare algorithms belonging to these two different classes, since they are analyzed

[†] Thanks to the two ANR Projects: ANR BOOLE (ANR 2009 BLAN 0011) and ANR MAGNUM (ANR 2010 BLAN 0204).

with respect to different costs. Second, when the keys are complex items, not reduced to single machine words, it is not realistic to consider the total cost of their comparison as unitary. This is why Sedgewick proposed in 1998 to analyze basic algorithms (sorting and searching) when dealing with words rather than with "atomic" keys; in this case, the realistic cost for comparing two words is the number of symbols comparisons needed to distinguish them in the lexicographic order and is closely related to the length of their longest common prefix, called here the *coincidence*. There are two factors which influence the efficiency of such an algorithm: the strategy of the algorithm itself (*which words are compared?*) and the mechanism which produces words, called the source (*what makes two words distinguishable?*).

The first results in the area are due to Fill and Janson [9] who dealt with data composed of random uniform bits in the case of the QuickSort algorithm. Then, in [28], a general framework towards a realistic analysis based on the number of symbol comparisons is provided, when the source which emits symbols is (almost completely) general. Furthermore, these principles are applied to two algorithms, QuickSort and QuickSelect. Later on, a study of the distribution of the complexity was performed in the same framework [8, 10].

Main results. The present paper follows the lines of the article [28], and works within the same general framework. We consider that the input keys are words on a finite totally ordered *alphabet* Σ. What we call a probabilistic *source* produces infinite words on the alphabet Σ, as described in Definition 1. The set of *keys* (words, data items) is then $\Sigma^{\mathbb{N}}$, endowed with the strict lexicographic order. The sources of symbols that we deal with include memoryless sources, and Markov chains, as well as many non-Markovian sources with unbounded memory from the past. A central idea is the modelling of the source via its *fundamental probabilities*, namely the probabilities that a word of the source begins with a given prefix.

We wish to estimate the mean number of symbol comparisons performed by three popular sorting algorithms, when they deal with words produced by a given source. We describe our main results as follows:

(a) The general method has been already described in [28]: it was shown that a Dirichlet series denoted by $\varpi(s)$ characterizes the behavior of an algorithm with respect to the source. We wish here to highlight the main principles, in order to make easier its application to various algorithms. This is done in Section 1. As it is often the case in analytical combinatorics, there are two main phases in the method, a first phase where the series $\varpi(s)$ is built (with algebraic tools), and a second phase where it is analyzed (with analytical tools). We note here that the first phase may mostly be performed in an "automatic" way (Section 3).

(b) We apply the method to two other popular algorithms: InsertionSort, and BubbleSort, respectively denoted in the sequel by the short names `InsSort`, `BubSort` (see for instance the book [23] for a thorough description of these algorithms). With this general approach, we provide in Theorem 5.1 an unified framework for the analysis of these three algorithms, and we also recover the results about `QuickSort` already obtained in [28].

(*c*) The generating function of the source plays a fundamental rôle in the analysis. This series $\Lambda(s)$ (of Dirichlet type) defined in Eq. (1.2), and introduced for the first time in [27], collects the fundamental probabilities of the source, and intervenes in the expression of the Dirichlet series $\varpi(s)$ which characterizes the behavior of an algorithm with respect to the source (table of Figure 7). Our analyses deal with the case when the series $\Lambda(s)$ is "tame enough"; in this case, the source itself is called "tame" (Section 1.8 for a precise definition)[1]. We exhibit, for each algorithm, a particular constant of the source (namely the entropy or the coincidence) closely related to the Λ-function, which describes the interplay between the algorithm and the source, and explains how the efficiency of the algorithm depends on the source (Proposition 4.3). Our study is thus a tool for a better understanding of the algorithmic strategy.

(*d*) We also discuss in Section 5 the robustness of the algorithms, i.e., the two possible changes in the complexity behaviors: the first one due to the change in the complexity measure, from the number of key comparisons to the number of symbol comparisons, or the second one due to the tameness of the source.

(*e*) We then discuss the faithfulness of the algorithms. This notion was recently introduced by Seidel [24], and we obtain here a natural characterization of this notion, from which we easily prove that the algorithms `QuickSort` and `InsSort` are faithful, whereas the algorithm `BubSort` is not faithful. Seidel used the faithfulness property to obtain an interesting relation between the two measures of interest — mean number of key comparisons and mean number of symbol comparisons — in the case of a faithful algorithm. Adapting the main ideas of Seidel to our framework, we obtain an alternative proof for the expression of the mean number of symbol comparisons, in the case of the two faithful algorithms (Theorem 6.4).

(*f*) There is a close relation between the two analyses — faithful sorting algorithms and trie parameters —. We are then led to revisit trie analyses, and compare the two possible methods, the Rice methodology and the Poisson–Mellin tools (Proposition 6.3).

(*g*) We finally show in Section 6 that combining ideas of Seidel and our general framework leads to an asymptotic lower bound for the number of symbol comparisons performed by any comparison-based algorithm using the standard string comparison procedure (Theorem 6.5).

Relation between the four articles on related subjects. There are two extended abstracts: the paper [28], in the proceedings of ICALP 2009, and the paper [4], in the proceedings of STACS 2013. Then, there are two long papers which are journal versions of the previous ones.

The paper [28] was the first paper devoted to the subject "analysis with respect to symbol comparisons" in the case of general (non-ambiguous) sources; this short paper just mentioned the main steps of the methodology: even if the algebraic steps are well

[1] The word "tame" was proposed by Philippe Flajolet and used for the first time in [28]. Later on, most of the papers which deal with probabilistic sources use similar notions and the word "tame" is now largely used.

described, the analytic steps are just mentioned. It then focused on the `QuickSort` and `QuickSelect` algorithms, for which it states the main results, without proofs.

The paper [4] was the second paper devoted to the subject; it wished to perform two tasks: first it shows the generality of the method, and designs a quite general framework for the "analysis with respect to symbol comparisons". It provides a precise description of the algebraic steps, and also explains the analytic steps, with the introduction of the various notions of tameness. The second aim of [4] was to apply the method to five algorithms: amongst them, one again finds `QuickSort` and `QuickMin`, but also other classical algorithms, as InsertionSort, BubbleSort and Selection-Minimum. Again, this is only a short paper which does not contain proofs.

The two journal versions provide all the details for the methodology, and precise analyses for algorithms of interest; each of the two journal versions is devoted to a class of algorithms. The present paper focuses on the three sorting algorithms `QuickSort`, `InsSort`, `BubSort` and makes precise all the notions of tameness that are adapted to sorting algorithms, whereas the second paper [2] adapts the general method to the algorithms of the `QuickSelect` class, for which it provides a complete analysis, together with a precise description of the convenient notions of tameness for searching algorithms.

Plan of the paper. Section 1 first describes the general method, with its main four steps. Then, each following section (from Section 2 to Section 5) is devoted to one of these steps. Finally, Section 6 is devoted to the comparison of our results to the approach of Seidel, and description of asymptotic lower bounds.

1. General method for a "realistic" analysis of sorting algorithms.

Here, we describe our general framework, already provided in [28]. We insist on the main steps, and the notions developed here are somewhat different from the previous paper. We first characterize in Section 1.1 the strategy of the algorithm (which keys are compared? with which probability?), then we describe the model of source in Section 1.2 together, with the particular cases of "simple" sources, then the central notion of coincidence (Section 1.3). Section 1.4 is devoted to various probabilistic models, that lead in Section 1.5 to an exact formula for the mean number of symbol comparisons, which involves the mixed Dirichlet series $\varpi(s)$ (depending on the source *and* the algorithm). In order to obtain asymptotic estimates, we deal with tameness properties of the source, which entail tameness for the series $\varpi(s)$, and finally the asymptotic estimates (Sections 1.6, 1.7, and 1.8). We conclude by describing the plan for the following sections.

1.1. The classical probabilistic model based on permutations.

Consider a totally ordered set of keys $\mathcal{U} = \{U_1 < U_2 < \cdots < U_n\}$ and any algorithm $\mathcal{A}$ which only performs comparisons and exchanges between keys. The initial input is the sequence $(V_1, V_2, \ldots, V_n)$ defined from $\mathcal{U}$ by the permutation $\sigma \in \mathfrak{S}_n$ via the equalities $V_i = U_{\sigma(i)}$. The execution of the algorithm does not actually depend on the input sequence, but only on the permutation σ which defines the input sequence from the

final (ordered) sequence. Then, the permutation σ is the actual input of the algorithm and the set of all possible inputs is the set $\mathfrak{S}_n$.

The strategy of the algorithm $\mathcal{A}$ defines, for each pair (i, j), with $1 \leq i < j \leq n$, the subset of $\mathfrak{S}_n$ which gathers the permutations σ for which U_i and U_j are compared by the algorithm $\mathcal{A}$, when the input sequence is $(U_{\sigma(1)}, U_{\sigma(2)}, \ldots, U_{\sigma(n)})$. For efficient algorithms, the two keys U_i and U_j are compared only once, but there exist other algorithms (the `BubSort` algorithm for instance) where U_i and U_j may be compared several times. In all cases, considering the usual uniform distribution on permutations, $\pi(i, j)$ denotes the mean number of comparisons[2] between U_i and U_j. The computation of $\pi(i, j)$ is the first step of our method, and the results are found in Section 2 and summarized in the table of Figure 4.

1.2. Sources.

Here, we consider that the keys are words produced by a general source on a finite alphabet. A general source $\mathcal{S}$ built on the alphabet Σ produces at each discrete time $t = 0, 1, \ldots$ a symbol from Σ. If X_n is the symbol emitted at time $t = n$, a source produces the infinite word $(X_0, X_1, \ldots, X_n, \ldots)$. For any finite prefix $w \in \Sigma^\star$, the probability p_w that a word produced by the source $\mathcal{S}$ begins with the finite prefix w is called the fundamental probability of prefix w. The probabilities p_w, $w \in \Sigma^\star$, completely define the source $\mathcal{S}$. By convention, we denote open and closed intervals of real numbers $]a, b[$ and $[a, b]$, whereas (a, b) denotes a pair of real numbers.

Definition 1. Let Σ be a totally ordered alphabet of cardinality r. A source over the alphabet Σ produces infinite words of $\Sigma^{\mathbb{N}}$, and is specified by the *fundamental probabilities* p_w, $w \in \Sigma^\star$, where p_w is the probability that an infinite word begins with the finite prefix w. When the two following properties hold,

(*i*) $p_w > 0$ for any $w \in \Sigma^\star$, (*ii*) $\pi_k := \max\{p_w : w \in \Sigma^k\}$ tends to 0, as $k \to \infty$,

the source is said to be *non-ambiguous*.

In the sequel, all the sources are assumed to be non-ambiguous.

The sets Σ^k, for $k \geq 1$ and the set $\Sigma^{\mathbb{N}}$ are endowed with the strict lexicographic order (derived from the order on Σ) and denoted by '$<$'. For any prefix $w \in \Sigma^\star$, we denote by $|w|$ the length of w (i.e., the number of the symbols that it contains) and a_w, b_w, p_w the probabilities that a word produced by the source begins with a prefix α of the same length as w, which satisfies $\alpha < w$, $\alpha \leq w$, or $\alpha = w$, meaning

$$a_w := \sum_{\substack{\alpha, |\alpha|=|w| \\ \alpha<w}} p_\alpha, \qquad b_w := \sum_{\substack{\alpha, |\alpha|=|w| \\ \alpha\leq w}} p_\alpha, \qquad p_w = b_w - a_w. \tag{1.1}$$

[2] Strictly speaking, we should denote the expected value by $\pi(n, i, j)$ since it could depend on n for some algorithms (the simplest one being the selection of the maximum symmetric to the selection of the maximum). However for clarity's sake and also since the algorithms we chose to analyse have no such dependency, we omit the cardinality n.

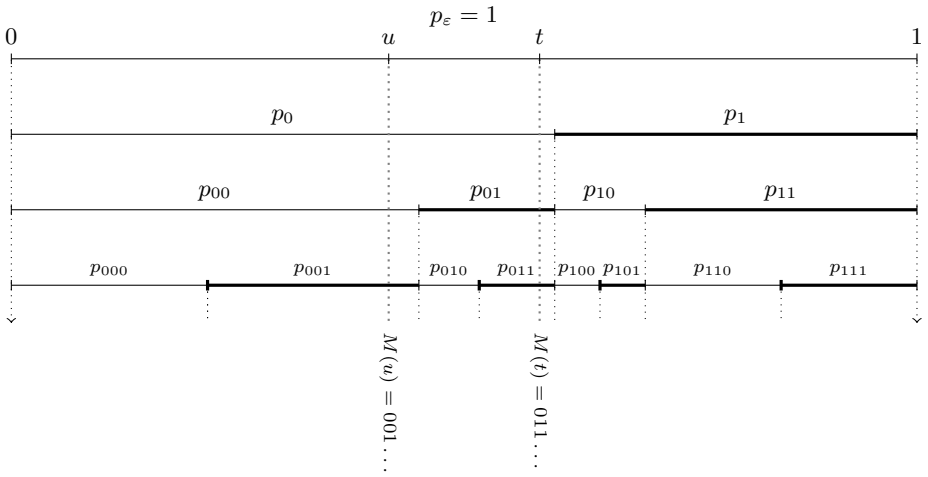


Figure 1: The parameterization of a source.

Thus, for a given k, when the prefix w varies in Σ^k, this gives rise to a partition of the unit interval with subintervals of length p_w (see Figure 1). When the prefixes $w \in \Sigma^k$ are ordered in increasing lexicographic order, and the subintervals are arranged from left to right, then, the subinterval corresponding to prefix w has left (respectively, right) endpoint equal to a_w (resp., b_w).

Parameterization of the source. Consider the set $\Sigma^{\mathbb{N}}$ of (infinite) words produced by the source $\mathcal{S}$, ordered via the lexicographic order. Given an infinite word $X \in \Sigma^{\mathbb{N}}$, denote by w_k its prefix of length k. The sequence $(a_{w_k})_{k\geq 0}$ is increasing, the sequence $(b_{w_k})_{k\geq 0}$ is decreasing, and $b_{w_k} - a_{w_k} = p_{w_k}$ tends to 0 when k tends to infinity. Thus a unique real $P(X) \in [0, 1]$ is defined as the common limit of (a_{w_k}) and (b_{w_k}), and $P(X)$ is simply the probability that an infinite word Y be smaller than X. The mapping $P : \Sigma^{\mathbb{N}} \to [0, 1]$ is surjective and strictly increasing outside the exceptional set formed with words of $\Sigma^{\mathbb{N}}$ which end with an infinite sequence of the smallest symbol or with an infinite sequence of the largest symbol.

Conversely, almost everywhere (except on the set $\{a_w, w \in \Sigma^\star\}$), there is a mapping M which associates a number u of the interval $\mathcal{I} := [0, 1]$ with a word $M(u) \in \Sigma^{\mathbb{N}}$. Hence, the probability that a word Y be smaller than $M(u)$ equals u. The lexicographic order on words ('$<$') is then compatible with the natural order on the interval $\mathcal{I}$, namely, $M(t) \leq M(u)$ if and only if $t \leq u$. The interval $\mathcal{I}_w := [a_w, b_w]$, of length p_w, gathers (up to a denumerable set) all the reals u for which $M(u)$ begins with the finite prefix w. This is the fundamental interval of the prefix w.

Dirichlet series $\Lambda(s)$. Our study involves the Dirichlet series of the source, defined as

$$\Lambda(s) := \sum_{w\in\Sigma^\star} p_w^s, \qquad \Lambda_k(s) := \sum_{w\in\Sigma^k} p_w^s. \tag{1.2}$$

Since the equalities $\Lambda_k(1) = 1$ hold, the series $\Lambda(s)$ is divergent at $s = 1$, and the probabilistic properties of the source can be expressed in terms of the regularity of Λ near $s = 1$, as we will see later.

For instance, the entropy $h(\mathcal{S})$ relative to a probabilistic source $\mathcal{S}$ is defined as the limit (if it exists) of a quantity that involves the fundamental probabilities

$$h(\mathcal{S}) := \lim_{k\to\infty} \frac{-1}{k} \sum_{w\in\Sigma^k} p_w \log p_w = \lim_{k\to\infty} \frac{-1}{k} \frac{d}{ds} \Lambda_k(s) \Big|_{s=1}. \tag{1.3}$$

For "good" sources, the Λ series is convergent at $s = 2$, and the constant $c(\mathcal{S}) := \Lambda(2)$ is called the coincidence of the source.

Simple sources: memoryless sources and Markov chains. A memoryless source over the alphabet Σ (possibly infinite) is a source where the symbols $X_i \in \Sigma$ are independent and identically distributed. The source is thus defined by the set $(p_j)_{j\in\Sigma}$ of probabilities, with $p_j = \Pr[X_k = j]$ for any $k \geq 0$. The Dirichlet series Λ, Λ_k are expressed with

$$\lambda(s) = \sum_{i\in\Sigma} p_i^s, \qquad \text{under the form} \qquad \Lambda_k(s) = \lambda(s)^k, \qquad \Lambda(s) = \frac{1}{1-\lambda(s)}. \tag{1.4}$$

In this case, the entropy equals $h(\mathcal{S}) = -\sum_i p_i \log p_i = -\lambda'(1)$.

A Markov chain over the finite alphabet Σ, is defined by the vector R of initial probabilities $(r_i)_{i\in\Sigma}$ together with the transition matrix $P := \big(p_{i|j}\big)_{(i,j)\in\Sigma\times\Sigma}$, whose each column has a sum equal to 1. Here, one has

$$r_i := \Pr[X_0 = i], \qquad p_{i|j} = \Pr[X_{k+1} = i \mid X_k = j]$$

for any $(i, j) \in \Sigma$ and $k \geq 0$. We denote by $P(s)$ the matrix with general coefficient $p_{i|j}^s$, and by $R(s)$ the vector of components r_i^s. Then

$$\Lambda_k(s) = {}^t\mathbf{1} \cdot P(s)^{k-1} \cdot R(s), \qquad \Lambda(s) = 1 + {}^t\mathbf{1} \cdot (I - P(s))^{-1} \cdot R(s). \tag{1.5}$$

If, moreover, the matrix P is irreducible and aperiodic, then, for any real $s > 0$, the matrix $P(s)$ has a unique dominant eigenvalue $\lambda(s)$. For $s = 1$, the matrix $P = P(1)$ has a unique fixed vector with positive components π_i, whose sum equals 1. The entropy of the source is then equal to

$$h(\mathcal{S}) = -\lambda'(1) = -\sum_{(i,j)\in\Sigma^2} \pi_j \, p_{i|j} \, \log p_{i|j}.$$

Other instances of "simple" sources: intermittent sources. Intermittent sources are an interesting particular case of a source of VLMC type (Variable Length Markov Chain), where the dependency from the past is unbounded. An intermittent source has two regimes, depending whether it emits a particular symbol $\underline{\sigma} \in \Sigma$ or not. Consider a source with an alphabet of finite cardinality $r \geq 2$. The source is *intermittent of exponent* $a > 0$ *with respect to* $\underline{\sigma}$ if one has the following conditional probability distribution for the emission of each symbol in the word given the prefix preceding it. Define the

event $\mathcal{S}_k$ as $\mathcal{S}_k := \{$the prefix ends with a sequence of exactly k occurrences of $\underline{\sigma}\}$. Then the conditional distribution of the next symbol emitted depends on the length k; more precisely, one has $\Pr[\sigma \mid \mathcal{S}_0] = 1/r$ and, for $k \geq 1$,

$$\Pr[\underline{\sigma} \mid \mathcal{S}_k] = \left(1 - \frac{1}{k+1}\right)^a, \qquad \Pr[\sigma \mid \mathcal{S}_k] = \left(1 - \left(1 - \frac{1}{k+1}\right)^a\right) \frac{1}{r-1} \quad \text{for } \sigma \neq \underline{\sigma}.$$

Then, in the case of a binary alphabet $\Sigma := \{0, 1\}$, when the source is intermittent with respect to 0, the probability of the prefixes 0^k and 0^k1 are respectively equal to

$$p_{0^k} = \frac{1}{2} \cdot \frac{1}{k^a}, \qquad p_{0^k1} = \frac{1}{2}\left(\frac{1}{k^a} - \frac{1}{(k+1)^a}\right),$$

and, with the language description $\{0, 1\}^\star = (0^\star 1)^\star \cdot 0^\star$, the series $\Lambda(s)$ admits the expression

$$\Lambda(s) = \frac{1 + 2^{-s}\zeta(as)}{1 - 2^{-s}[1 + \Sigma_a(s)]} \qquad \text{with} \quad \Sigma_a(s) = \sum_{k \geq 1} \left[\frac{1}{k^a} - \frac{1}{(k+1)^a}\right]^s.$$

[Here, $\zeta(\cdot)$ is the Riemann zeta function.]

Dynamical sources. An important subclass is formed by *dynamical sources*, which are closely related to dynamical systems on the interval [27]. One starts with a partition $\{\mathcal{I}_\sigma\}$ indexed by symbols $\sigma \in \Sigma$, a coding map $\tau : \mathcal{I} \to \Sigma$ which equals σ on $\mathcal{I}_\sigma$, and a shift map $T : \mathcal{I} \to \mathcal{I}$ whose restriction to each $\mathcal{I}_\sigma$ is increasing, invertible, and of class $\mathcal{C}^2$. Then the word $M(u)$ is the word that encodes the trajectory $(u, Tu, T^2u, \dots)$ via the coding map τ, namely, $M(u) := (\tau(u), \tau(Tu), \tau(T^2u), \dots)$. All memoryless (Bernoulli) sources and all Markov chain sources belong to the general framework of Definition 1: they correspond to a piecewise linear shift, under this angle. For instance, the standard binary system is obtained by $T(x) = \{2x\}$ ($\{\cdot\}$ is the fractional part). Dynamical sources with a non-linear shift allow for correlations that depend on the entire past (e.g., sources related to continued fractions obtained by $T(x) = \{1/x\}$).

For complete dynamical systems, which emit all the possible infinite words, there is an operator, called the "secant transfer operator", and denoted by $\mathbb{H}_s$ which provides a generalization of the transition matrix of Markov chains. The Dirichlet series $\Lambda(s)$ defined in (1.2) can be expressed with the quasi-inverse $(I - \mathbb{H}_s)^{-1}$ of this operator (see [27]).

1.3. Geometry of the source and coincidence.

We are interested in a more realistic cost related to the number of symbol comparisons performed by sorting algorithms, when the keys are words independently produced by the same source. The words are ordered with respect to the lexicographic order, and the cost for comparing two words (measured as the number of symbol comparisons needed) is closely related to the coincidence, defined as follows.

Definition 2. The *coincidence function* $\gamma(u, t) : [0, 1] \times [0, 1] \to \mathbb{N} \cup \{+\infty\}$ is the length of the largest common prefix of words $M(u)$ and $M(t)$.

More precisely, the realistic cost of the comparison between $M(u)$ and $M(t)$ equals $\gamma(u,t)+1$.

We represent the pair of words $(M(u), M(t))$ with $u \le t$ by the point (u,t) of the triangle $\mathcal{T} := \{(u,t) : 0 \le u \le t \le 1\}$, and the *fundamental triangles*

$$\mathcal{T}_w = (\mathcal{I}_w \times \mathcal{I}_w) \cap \mathcal{T} = \{(u,t) : a_w \le u \le t \le b_w\} \tag{1.6}$$

define the *level sets* of the function γ. Indeed, the coincidence $\gamma(u,t)$ is at least ℓ if and only if $M(u)$ and $M(t)$ have the same common prefix w of length ℓ, so that the parameters u and t belong to the same fundamental interval $\mathcal{I}_w$ relative to a prefix w of length ℓ. Then, the two relations

$$\mathcal{T} \cap [\gamma \ge \ell] = \bigcup_{w \in \Sigma^\ell} \mathcal{T}_w, \qquad \sum_{\ell \ge 0} \mathbf{1}_{[\gamma \ge \ell]} = \sum_{\ell \ge 0} (\ell+1) \mathbf{1}_{[\gamma = \ell]},$$

entail the following equality which deals with the functional $\mathcal{J}$ and holds for any integrable function g on the unit triangle $\mathcal{T}$,

$$\mathcal{J}[g] := \int_{\mathcal{T}} [\gamma(u,t)+1] g(u,t)\, du\, dt = \sum_{w \in \Sigma^\star} \int_{\mathcal{T}_w} g(u,t)\, du\, dt. \tag{1.7}$$

This functional $\mathcal{J}$ will be extensively used in the sequel.

The following figure represents the family of triangles $\mathcal{T}_w$, which defines the "geometry" of the source for two memoryless sources.

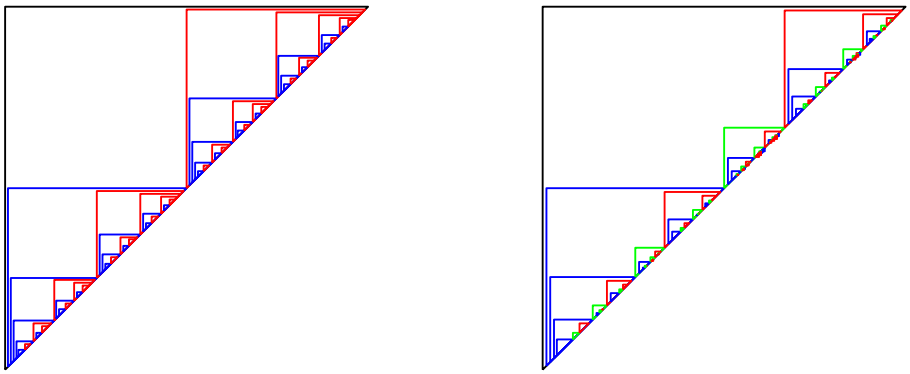


Figure 2: The geometry of two memoryless sources. On the left, the case of $\Sigma := \{a,b\}$ with $p_a = p_b = 1/2$. On the right, the case of $\Sigma := \{a,b,c\}$ with $p_a = 1/2, p_b = 1/6, p_c = 1/3$.

The sum of all the triangle areas involves the Λ series defined in (1.2), under the form

$$\frac{1}{2} \sum_{w \in \Sigma^\star} p_w^2 = \frac{1}{2} \Lambda(2)$$

and equals $(1/2)\, c(\mathcal{S})$ where $c(\mathcal{S})$ is the coincidence of the source, already mentioned in Section 1.2.

1.4. Average-case analysis – various models.
The purpose of average-case analysis of structures (or algorithms) is to characterize the mean value of their "costs" under a well-defined probabilistic model that describes the initial distribution of its inputs.

Here, we adopt the following general model for the set of inputs: we consider a finite sequence $\mathcal{V} = (V_1, \ldots, V_n)$ of infinite words independently produced by the same source $\mathcal{S}$. Such a sequence $\mathcal{V}$ is obtained by n independent drawings $v_1, v_2, \ldots, v_n$ in the unit interval $\mathcal{I} = [0, 1]$ via the mapping M, and we set $V_i := M(v_i)$. We assume moreover that $\mathcal{V}$ contains two given words $M(u)$ and $M(t)$, with $u < t$. The variables $N_{[0,u[}$, $N_{[0,t[}$ respectively denote the number of words of $\mathcal{V}$ strictly less than $M(u)$, strictly less than $M(t)$. These variables define the ranks of $M(u)$ and $M(t)$ inside the set $\mathcal{V}$, via the relations, valid for $u < t$,

$$\operatorname{Rank} M(u) = N_{[0,u[} + 1, \quad \operatorname{Rank} M(t) = N_{[0,u[} + N_{]u,t[} + 2,$$

where the respective translations of 1 and 2 express that $M(u)$ and $M(t)$ belong to $\mathcal{V}$.

We first consider the number of key comparisons between $M(u)$ and $M(t)$, and deal with the mean number $\widehat{\pi}(u, t)$ of key comparisons performed by the algorithm between two words $M(u)$ and $M(t)$ *chosen as keys*, where the mean is taken with respect to all the permutations of $\mathcal{V}$. The mean number $\widehat{\pi}(u, t)$ is related to the mean number $\pi(i, j)$ via the equality

$$\widehat{\pi}(u, t) = \pi(N_{[0,u[} + 1, N_{[0,u[} + N_{]u,t[} + 2). \tag{1.8}$$

In our framework, expressions obtained for $\pi(i, j)$ ensure that $\widehat{\pi}(u, t)$ is always a sum of rational functions in variables $N_{[0,u[}$ and $N_{]u,t[}$.

When the cardinality n of $\mathcal{V}$ is fixed, and words $V_i \in \mathcal{V}$ are independently emitted by the source $\mathcal{S}$, this is the Bernoulli model denoted by $(\mathcal{B}_n, \mathcal{S})$. However, it proves technically convenient to consider that the cardinality N of the sequence $\mathcal{V}$ is a random variable that obeys a Poisson law of rate Z,

$$\Pr\{N = k\} = e^{-Z} \frac{Z^k}{k!}. \tag{1.9}$$

In this model, called the Poisson model of rate Z, the rate Z plays a role much similar to the cardinality of $\mathcal{V}$. When it is relative to probabilistic source $\mathcal{S}$, the model, denoted by $(\mathcal{P}_Z, \mathcal{S})$, is composed with two main steps:

(a) The number N of words is drawn according to the Poisson law of rate Z;
(b) Then, the N words are independently drawn from the source $\mathcal{S}$.

Note that, in the Poisson model, the variables $N_{[0,u[}, N_{]u,t[}$ are themselves independent Poisson variables of parameters Zu and $Z(t - u)$ (respectively). This implies that the sum $N_{[0,u[} + N_{]u,t[}$ is also a Poisson variable of parameter Zt. The expectation $\widehat{\pi}(u, t)$ is itself a random variable which involves these variables.

1.5. Exact formula for the mean number of symbol comparisons.
We first work in the Poisson model, where the previous independence properties lead to easier computations, then we return to the Bernoulli model.

Density in the Poisson model. The density of the algorithm at the point (u,t) of triangle $\mathcal{T}$ is defined as the mean number of key comparisons between a pair of words $(M(u'), M(t'))$ for $u' \in [u-du, u]$ and $t' \in [t, t+dt]$. In the Poisson model, the two intervals $[u-du, u]$ and $[t, t+dt]$ are disjoint for $u < t$, and the probability that the words $M(u'), M(t')$ are both chosen as keys for some $u' \in [u-du, u]$ and $t' \in [t, t+dt]$ is $Zdu \cdot Zdt$. Now, *conditionally*, given that $M(u)$ and $M(t)$ are *both chosen as keys*, the mean number of comparisons between the two words $M(u), M(t)$ is $\mathbb{E}_Z[\widehat{\pi}(u,t)]$. Thus, in the Poisson model, the mean number of key comparisons performed by the algorithm between two words $M(u')$ and $M(t')$ for $u' \in [u-du, u]$ and $t' \in [t, t+dt]$ equals

$$Zdu \cdot Zdt \cdot \mathbb{E}_Z[\widehat{\pi}(u,t)]. \tag{1.10}$$

Then, the density $\phi_Z(u,t)$ in the Poisson model satisfies

$$\phi_Z(u,t) = Z^2 \cdot \mathbb{E}_Z[\widehat{\pi}(u,t)]. \tag{1.11}$$

In our framework, the series expansion of $\phi_Z(u,t)$, written as[3]

$$\phi_Z(u,t) = \sum_{k\geq 2} (-1)^k \frac{Z^k}{k!} \varphi(k,u,t), \tag{1.12}$$

is easy to obtain, as its coefficients $\varphi(k,u,t)$

$$(-1)^k \varphi(k,u,t) := k![Z^k]\phi_Z(u,t) \quad \text{for } k \geq 2, \qquad \varphi(k,u,t) = 0 \quad \text{for } k = 0, 1, \tag{1.13}$$

are computed in an "automatic way" during the second step (done in Section 3) leading to results described in Figure 5.

Expectation in the Poisson model. In the model $(\mathcal{P}_Z, \mathcal{S})$, the density ϕ_Z is an easy main tool for computing the mean number of key comparisons K_Z *and* the mean number of symbol comparisons S_Z performed by the algorithm. The mean number K_Z is obtained by taking the integral over the triangle $\mathcal{T}$, namely

$$K_Z = \mathcal{L}[\phi_Z], \qquad \text{with} \quad \mathcal{L}[\Phi] = \int_{\mathcal{T}} \Phi(u,t)\, du\, dt.$$

Since the product $[\gamma(u,t)+1]\,\phi_Z(u,t)$ is the mean number of symbol comparisons between two words $M(u')$ and $M(t')$ for (u',t') close to (u,t), the mean number of symbol comparisons S_Z is obtained via the formula

$$S_Z = \mathcal{J}[\phi_Z] = \int_{\mathcal{T}} [\gamma(u,t)+1]\, \phi_Z(u,t)\, du\, dt,$$

where the functional $\mathcal{J}$ was introduced in (1.7).

This is a general phenomenon: there is the same type of formula for the mean number of comparisons in the two models — key comparisons and symbol comparisons —. For symbol comparisons, the functional $\mathcal{J}$ replaces the integral $\mathcal{L}$.

[3] The sign $(-1)^k$ is just put here to get positive coefficients $\varphi(k,u,t)$, (as we will see it in Section 3), and the condition $n \geq 2$ is related to Eq. (1.11).

Using now the linearity of the functional $\mathcal{J}$, the series expansion of the expectation S_Z, defined as

$$S_Z = \sum_{k\geq 0}(-1)^k \frac{Z^k}{k!}\varphi(k).$$

involves the sequence $\varphi(k)$, defined for any $k \geq 2$ as

$$\varphi(k) := \mathcal{J}[(u,t) \mapsto \varphi(k,u,t)] = \sum_{w\in\Sigma^\star} \int_{\mathcal{T}_w} \varphi(k,u,t)\, du\, dt. \tag{1.14}$$

This sequence is easy to compute with integrals of $\varphi(k,u,t)$ on the triangles $\mathcal{T}_w$. It depends both on the algorithm (via the sequence of functions $\varphi(k,u,t)$) and the source (via the fundamental triangles $\mathcal{T}_w$).

Expectation in the Bernoulli model. It is now easy to return to the Bernoulli model $(\mathcal{B}_n, \mathcal{S})$, where we are interested in the mean number $K(n)$ of key comparisons and the mean number $S(n)$ of symbol comparisons performed by the algorithm. The mean number $S(n)$ of symbol comparisons used by the algorithm when it deals with n words independently drawn from the same source is related to S_Z and then to $\varphi(n)$ by the equalities

$$S_Z = e^{-Z}\sum_{n\geq 2}\frac{Z^n}{n!}S(n), \qquad S(n) = \sum_{k=2}^{n}(-1)^k\binom{n}{k}\varphi(k), \tag{1.15}$$

which provide an exact formula for $S(n)$, described in Proposition 4.1 in Section 4. The expression of $S(n)$ is thus obtained in an "automatic" way, from the expectations $\pi(i,j)$.

1.6. Asymptotic estimates for the mean number of symbol comparisons. However, the previous formula does not give an easy or straightforward access to the asymptotic behaviour of $S(n)$ (when $n \to \infty$).

Analytic lifting. In order to get asymptotic estimates, we first need an analytic lifting $\varpi(s,u,t)$ of the coefficients $\varphi(k,u,t)$, that is an analytic function $\varpi(s,u,t)$ which coincides with $\varphi(k,u,t)$ at integer values $s = k$ in the summation of Eq. (1.12). This analytic lifting gives rise to the mixed Dirichlet series itself,

$$\varpi(s) := \mathcal{J}[\varpi(s,u,t)] = \int_{\mathcal{T}}[\gamma(u,t)+1]\varpi(s,u,t)\, du\, dt = \sum_{w\in\Sigma^\star}\int_{\mathcal{T}_w}\varpi(s,u,t)\, du\, dt, \tag{1.16}$$

which depends both on the algorithm (via $\varpi(s,u,t)$) and the source (via the fundamental triangles $\mathcal{T}_w$). This Dirichlet series is studied in Section 4.

The abscissa σ_0. For each algorithm, the existence of this analytic lifting is granted in a domain $\Re s > \sigma_0$. However, the value of σ_0 depends on the algorithm, and there are two classes of algorithms. The first class is formed by `InsSort` and `BubSort`. Here, there is a constant term $1/2$ which appears in the expectation $\pi(i,j)$, as seen in Figure 4 (see also Section 4). Then, the analytic lifting of $\varpi(s)$ exists only for integers $k \geq 3$, and thus for

$\Re s > 2$, and the abscissa σ_0 equals 2. This constant 1/2 will be considered in a separate way, and its contribution will be added at the end of the computation. There is a unique algorithm in the second class, namely the `QuickSort` algorithm, for which the analytic lifting exists for $k \geq 2$, and thus for $\Re s > 1$, and the abscissa σ_0 equals 1.

The Rice formula. The Rice Formula [20, 21] transforms a binomial sum into an integral in the complex plane. For any real $\sigma_1 \in]\sigma_0, \sigma_0 + 1[$, one has

$$T(n) := \sum_{k=1+\sigma_0}^{n} (-1)^k \binom{n}{k} \varpi(k) = \frac{(-1)^{n+1}}{2i\pi} \int_{\Re s=\sigma_1} G(s)\, ds, \text{ with } G(s) = \frac{n!\, \varpi(s)}{s(s-1)\dots(s-n)}. \tag{1.17}$$

Then, along general principles in analytic combinatorics [15, 16], the integration line can be pushed to the left, as soon as $G(s)$ (closely related to $\varpi(s)$) has good analytic properties: we need a region $\mathcal{R}$ on the left of $\Re s = \sigma_0$, where $\varpi(s)$ is of polynomial growth (for $|\Im s| \to \infty$) and meromorphic. With a good knowledge of its poles, we finally obtain a residue formula

$$T(n) = (-1)^{n+1} \left[\sum_s \operatorname{Res}\left[G(s)\right] + \frac{1}{2i\pi} \int_{\mathcal{C}_2} G(s)\, ds \right], \tag{1.18}$$

where $\mathcal{C}_2$ is a curve of class $\mathcal{C}^1$ enclosed in $\mathcal{R}$ and the sum is extended to all poles s of $G(s)$ inside the domain delimited by the vertical line $\Re s = \sigma_1$ and the curve $\mathcal{C}_2$.

The dominant singularities of $G(s)$ provide the asymptotic behaviour of $T(n)$, and the remainder integral is estimated using the polynomial growth of $G(s)$ when $|\Im(s)| \to \infty$. According to Eq. (1.15) and (1.17), and in the cases where $\sigma_0 = 2$, we have to add to $T(n)$ the term corresponding to the index $k = 2$, where the analytical lifting ϖ does not coincides with φ. For algorithms `BubSort` and `InsSort`, the additional term is of the form $\varphi(2)\binom{n}{2}$. Finally, depending on the value of σ_0 the mean number $S(n)$ of symbol comparisons is

$$S(n) = T(n) = \sum_{k=2}^{n} (-1)^k \binom{n}{k} \varpi(k) \qquad (\text{if } \sigma_0 = 1) \tag{1.19}$$

$$S(n) = \binom{n}{2} \varphi(2) + T(n) = \binom{n}{2} \varphi(2) + \sum_{k=3}^{n} (-1)^k \binom{n}{k} \varpi(k) \qquad (\text{if } \sigma_0 = 2). \tag{1.20}$$

1.7. Tameness properties of the Dirichlet series $\varpi(s)$.

We first describe three cases of possible regions $\mathcal{R}$ where good properties of $\varpi(s)$ will make possible such a shifting to the left in the Rice formula.

Definition 3. A function $\varpi(s)$ is *tame* at σ_0 if one of the three following properties holds:

(a) [S-shape] (shorthand for Strip shape) there exists a vertical strip $\Re(s) > \sigma_0 - \delta$ for some $\delta > 0$ where $\varpi(s)$ is meromorphic, has a sole possible pole (of order $k_0 \geq 0$) at $s = \sigma_0$ and is of polynomial growth as $|\Im s| \to +\infty$.

(*b*) [*H*-shape] (shorthand for Hyperbolic shape) there exists an hyperbolic region $\mathcal{R}$, defined as, for some $A, B, \rho > 0$

$$\mathcal{R} := \{s = \sigma + it; \quad |t| \geq B, \quad \sigma > \sigma_0 - \frac{A}{|t|^\rho}\} \bigcup \{s = \sigma + it; \quad \sigma > \sigma_0 - \frac{A}{B^\rho}, |t| \leq B\},$$

where $\varpi(s)$ is meromorphic, with a sole possible pole (of order $k_0 \geq 0$) at $s = \sigma_0$ and is of polynomial growth in $\mathcal{R}$ as $|\Im s| \to +\infty$.

(*c*) [*P*-shape] (shorthand for Periodic shape) there exists a vertical strip $\Re(s) > \sigma_0 - \delta$ for some $\delta > 0$ where $\varpi(s)$ is meromorphic, has only a pole (of order $k_0 \geq 0$) at $s = \sigma_0$ and a family (s_k) (for $k \in \mathbb{Z} \setminus \{0\}$) of simple poles at points $s_k = \sigma_0 + 2ki\pi t$ with $t \neq 0$, and is of polynomial growth as $|\Im s| \to +\infty$[4].

There are three parameters relative to the tameness: the integer k_0 is the *order*, and, when they exist, the real δ is the *width*, and the real ρ is the *exponent*.

Such tameness properties, together with the Rice formula, entail the following asymptotic properties for the sum $T(n)$ defined in (1.17).

Proposition 1.1. *The following holds for the sequence $T(n)$, when it is related to $\varpi(s)$ by the Rice formula (1.17), with $\sigma_0 \in \{1, 2\}$. If $\varpi(s)$ is tame at $s = \sigma_0$ with order k_0, then there exists a polynomial Q of degree k_0 such the following asymptotics hold, depending on the tameness shape:*

(*a*) *With a S-shape and width δ_0, for any $\delta < \delta_0$, one has, for $n \to \infty$,*

$$(-1)^{n+1} T(n) = n^{\sigma_0} Q(\log n) + O(n^{\sigma_0 - \delta}).$$

(*b*) *With a H-shape and exponent β_0, then, for any β with $\beta < 1/(\beta_0 + 1)$, one has, for $n \to \infty$,*

$$(-1)^{n+1} T(n) = n^{\sigma_0} Q(\log n) + O\left(n^{\sigma_0} \cdot \exp[-(\log n)^\beta]\right)$$

(*c*) *With a P-shape and width δ_0, then, for any $\delta < \delta_0$, one has, for $n \to \infty$,*

$$(-1)^{n+1} T(n) = n^{\sigma_0} \left(Q(\log n) + \Phi(n)\right) + O(n^{\sigma_0 - \delta})$$

where $n^{\sigma_0} \cdot \Phi(n)$ is the part of the expansion brought by the family of the non real poles of $G(s)$ located on the vertical line $\Re s = \sigma_0$.

Note that, from Eq. (1.18), the dominant part of the asymptotics comes from considering poles in the region $\mathcal{R}$ (residue calculus) and the error term comes from the evaluation of the integral on the curve $\mathcal{C}_2$ in Eq. (1.18).

1.8. Tameness of sources.

Here, the main Dirichlet series $\varpi(s)$ of interest relative to sorting algorithms are closely related to the Dirichlet series $\Lambda(s)$ of the source, already defined in Eq. (1.2).

[4] More precisely, this means that $\varpi(s)$ is of polynomial growth on a family of horizontal lines $t = t_k$ with $t_k \to \infty$, and on vertical lines $\Re(s) = \sigma_0 - \delta'$ with some $\delta' < \delta$.

Series $\Lambda(s)$ and its extensions. We recall the definition of these Dirichlet series,

$$\Lambda(s) = \sum_{w\in\Sigma^*} p_w^s, \qquad \Lambda_k(s) = \sum_{w\in\Sigma^k} p_w^s.$$

There are also extensions of $\Lambda(s)$ and $\Lambda_k(s)$ which involve the fundamental probabilities p_w, together with the ends a_w, b_w of the fundamental intervals (see Section 1.2), via a function $F : [0,1]^2 \to \mathbb{R}^+$ of class $\mathcal{C}^1$,

$$\Lambda[F](s) := \sum_{w\in\Sigma^\star} F(a_w, b_w)\, p_w^s, \qquad \Lambda_k[F](s) := \sum_{w\in\Sigma^k} F(a_w, b_w)\, p_w^s. \tag{1.21}$$

For $F \equiv 1$, we recover the classical function $\Lambda := \Lambda[1]$. On the halfplane $\sigma := \Re s > 1$, these series satisfy the relation $|\Lambda[F](s)| \le \|F\|\Lambda(\sigma)$, where the norm $\|\cdot\|$ is the sup-norm on $[0,1]\times[0,1]$.

Tameness of sources. We now describe properties of the source that may entail tameness for the mixed series $\varpi(s)$.

Definition 4. [Tameness of Sources] Denote by $\mathcal{F}$ the set of functions $F : [0,1]^2 \to \mathbb{R}^+$ of class $\mathcal{C}^1$. A source is Λ-*tame* if $\Lambda(s)$ admits at $s = 1$ a simple pole, with a residue equal to $1/h(\mathcal{S})$, (where $h(\mathcal{S})$ is the entropy of the source) and if one of the following conditions is fulfilled:

[S-shape] for any $F \in \mathcal{F}$, the series $\Lambda[F](s)$ is tame at $s = 1$ with a S-shape;
[H-shape] for any $F \in \mathcal{F}$, the series $\Lambda[F](s)$ is tame at $s = 1$ with a H-shape;
[P-shape] for any $F \in \mathcal{F}$, the series $\Lambda[F](s)$ is tame at $s = 1$, with a P-shape for $F \equiv 1$. For $F \not\equiv 1$, $\Lambda[F](s)$ has either a S-shape, or a P-shape.

Remark. If $\Lambda(s)$ admits at $s = 1$ a simple pole, with a residue equal to $1/h(\mathcal{S})$, then any series $\Lambda[F](s)$ for any $F \in \mathcal{F}, F > 0$, admits at $s = 1$ a simple pole, with a residue equal to (see Section 4.5 for details)

$$\frac{1}{h(\mathcal{S})}\int_0^1 F(x,x)dx.$$

About various notions of tameness. This definition is in fact very natural, as it describes various possible behaviors of classical sources. "Most of the time", the simple sources (memoryless sources or aperiodic Markov chains) are Λ-tame. They never have a S-shape, but they may have a H-shape or a P-shape, according to arithmetic properties of their probabilities [14]. Dynamical sources, introduced by Vallée and defined in [27], may have a P-shape only if they are "similar" to simple sources. Adapting deep results of Dolgopyat [6, 7] to the "secant transfer operator" $\mathbb{H}_s$, it is possible to prove that dynamical sources are "most of the time" Λ-tame with a S-shape [1], but they may also have a H-shape [22]. The intermittent sources defined in Section 1.2 are Λ-tame as soon as the parameter a satisfies $a > 1$. And, for $a = 1$, the source is not Λ-tame but the Dirichlet series $\Lambda(s)$ is tame at $s = 1$ of order 2, with a H-shape given by the Riemann

ζ function. For more details, see the cited papers where all these facts, here described in a informal way, are stated in a formal way and proven.

Relation between tameness of sources and tameness of the mixed series $\varpi(s)$. This tameness notion for sources is also well-adapted to our framework since it describes situations where the mixed series $\varpi(s)$ may be proven tame. More precisely, as it is proven in Proposition 4.2 of Section 4, the Λ-tameness of the source is central in the analysis, as it ensures the tameness of the mixed series $\varpi(s)$; moreover, the tameness shape of $\varpi(s)$ is inherited from the one of the source. It is then possible to shift the contour of the Rice integral to the left, providing an asymptotic expansion for the mean number $T(n)$ as it is obtained in Proposition 1.1. It is easy to return to $S(n)$ with (1.15). This leads to the final result for the asymptotics of $S(n)$, in Theorem 5.1 (see Section 5).

Plan of the following sections; application to the three algorithms under study. We have drawn the general framework of our study. We now apply it to the analysis of three "popular "algorithms which are precisely described in Section 2.1. Each step of the method is then performed:

- *First Step* (Section 2). Computation of expected values $\pi(i, j)$.
- *Second Step* (Section 3). Automatic derivation of $\varpi(s, u, t)$ and determination of the abscissa σ_0.
- *Third Step* (Section 4). Expression for the mixed Dirichlet series $\varpi(s)$, relation between tameness of the source and tameness of the mixed series $\varpi(s)$ and description of the main term of the singular expression of $\varpi(s)/(s - \sigma_0)$. Interpretation of the "dominant" constants.
- *Final Step* (Section 5). Application of the Rice Formula and statement of the final results (with a discussion about robustness and possible extensions).

Moreover, the last Section (Section 6) compares our point of view with Seidel's one, and describes asymptotic lower bounds.

2. First Step. Computation of the mean numbers $\pi(i, j)$.

Section 2.1 describes the three algorithms of interest, Section 2.2 explains the main principles that we adopt for the computation of $\pi(i, j)$, and Section 2.2 states the results of the first step. Then, the sequel of the Section is devoted to the proof of the results, and each of the following subsections studies one algorithm.

2.1. Description of the three algorithms.

We first briefly recall in Figure 3 the three algorithms under study (for precisions, see [23]).

2.2. General approach for the computation of $\pi(i, j)$.

When the ordered sequence $\mathcal{U}$ is given under the permutation σ, the input sequence is $(V_1, V_2, \ldots, V_n)$, with $V_i = U_{\sigma(i)}$ and we adopt the point of view given by the arrival

Procedure `QuickSort`$(V, \text{left}, \text{right})$

```
/* Sorts the subarray V[left..right].                                  */
/* Recursive function to be called for an array V[1..n]: QuickSort
   (V, 1, n)                                                          */
/* Partition (V, i, j) rearranges the subarray V[i..j] according to its
   first element V[i], called the pivot, and compares each element to
   the pivot: the keys that are smaller than the pivot are placed on
   its left in the array, whereas the keys that are greater are
   placed on its right, and thus the pivot is at the right place.
   Partition returns the rank of the pivot.                           */
if left < right then
    k ← Partition(V, left, right)
    QuickSort (V, left, k − 1)
    QuickSort (V, k + 1, right)
end
```

Procedure `InsSort`(V, n)

```
/* Sorts the array V[1..n]                                            */
for i from 2 to n do
    for j from i downto 2 do
        if V[j − 1] ≥ V[j] then
            swap(V[j], V[j − 1])
        end
        else Break /* exit the inner loop                             */
    end
end
```

Procedure `BubSort`(V, n)

```
/* Sorts the array V[1..n]                                            */
for i from 1 to n − 1 do
    for j from n downto i + 1 do
        if V[j − 1] > V[j] then
            swap(V[j − 1], V[j])
        end
    end
end
```

Figure 3: The three algorithms under study: `QuickSort`, `InsSort`, and `BubSort`.

times. The arrival time of U_i, denoted by $\tau(U_i)$ is the position of U_i in the input array. Of course, there is a simple relation between the two points of view since $\tau(U_i) = j$ if and only if $V_j = U_i$ (meaning also $\sigma(j) = i$ since there is a bijection between arrival times and permutations).

There are two types of comparisons between two keys U_i and U_j: the *positive* comparisons which occur when U_i and U_j arrive in the good order in the initial array ($\tau(U_i) < \tau(U_j)$), and the *negative* comparisons which occur when U_i and U_j arrive in the wrong order ($\tau(U_i) > \tau(U_j)$). The mean number of positive and negative comparisons between two keys U_i and U_j is denoted respectively by $\pi^+(i,j)$ and $\pi^-(i,j)$. These mean numbers $\pi^{\pm}(i,j)$ are often computed in a separate way, with direct probabilistic arguments dealing with the arrival times. A remarkable feature is that the expectations $\pi^{\pm}(i,j)$ are always expressed as sums of rational functions depending on i, j or $j - i$. The mean number of key comparisons is $\pi(i,j) = \pi^+(i,j) + \pi^-(i,j)$.

We will see that the event "U_i and U_j are compared" is generally "similar" to an event of the type "The arrival times of the keys U_i and U_j into a given subset $\mathcal{V}$ of keys are the first two (resp. the last two)". For a subset $\mathcal{V}$ of cardinality ℓ, the probability of such an event is $1/\ell(\ell-1)$. Moreover, the subset $\mathcal{V}$ is often a subset $\mathcal{U}_{[x,y]}$ which gathers all the keys whose rank belongs to the interval $[x,y]$, with three main cases, according to the algorithms: $[x,y] = [1,i]$, $[x,y] = [1,j]$, or $[x,y] = [i,j]$, which entails that ℓ belongs to $\{i, j, j-i+1\}$.

Then, we obtain a general form for the mean numbers $\pi(i,j)$ and their analogs $\widehat{\pi}(u,t)$, which makes possible the automatic transfer performed in Section 3.

Summary of the results for Step 1. We present in Figure 4 the expressions for the mean number $\pi(i,j)$ of key comparisons between U_i and U_j, for each algorithm of interest. With these expressions, it is easy to recover the estimates for the mean number $K(n)$ of key comparisons (recalled in the third column).

Proposition 2.1. *Consider the uniform permutation model described in Sections 1.1 (and also 2.2), and denote by $\pi(i,j)$ the mean number of comparisons between the keys of rank i and j, with $i < j$. Then, for any of the three algorithms, the mean numbers $\pi(i,j)$ admit the expressions described in the second column of Figure 4.*

2.3. Algorithm `QuickSort`.

This algorithm is based on the "Divide and Conquer" principle. All the keys are compared to the first key of the array that is used as a pivot. During the `Partition` stage, the keys that are smaller than the pivot are placed on its left in the array, whereas the keys that are greater are placed on its right. After this partitioning, the pivot is at the right place.

Then, the `QuickSort` algorithm recursively sorts the two sub-arrays. While the pivot does not belong to the subset $\mathcal{U}_{[i,j]}$, this set is not separated by the pivot. When the pivot belongs to the subset $\mathcal{U}_{[i,j]}$, the keys U_i and U_j may be compared only if U_i or U_j is a pivot. This event coincides with the event "U_i or U_j is the first key-in inside the subset $\mathcal{U}_{[i,j]}$". After such a comparison, the keys are separated and no longer compared.

Algorithms	$\pi(i,j)$	$K(n)$
`QuickSort`	$\dfrac{2}{j-i+1}$	$2n\log n$
`InsSort`	$\dfrac{1}{2}+\dfrac{1}{(j-i+1)(j-i)}$	$\dfrac{n^2}{4}$
`BubSort`	$\dfrac{1}{2}+\dfrac{1}{(j-i+1)(j-i)}+\dfrac{2(i-1)}{(j-i+2)(j-i+1)(j-i)}$	$\dfrac{n^2}{2}$

Figure 4: Results for Step 1.

Then, the mean number of key comparisons is $\pi(i,j) = 2/(j-i+1)$ (the probability of the event).

2.4. Algorithm `InsSort`.
There are $n-1$ phases in the algorithm. During the i-th phase, the key V_i of the array is inserted into the left sub-array which contains an already sorted sequence built on the set $\{V_1, V_2, \ldots, V_{i-1}\}$.

First case. U_i and U_j arrive in the wrong order in the initial array ($\tau(U_i) > \tau(U_j)$). In the phase when U_i is inserted into the left sub-array, this sub-array already contains U_j with $U_j > U_i$, and the key U_i is always compared and exchanged with U_j. This event is defined as "Inside the two keys set $\{U_i, U_j\}$, U_j is the first-in key, and U_i is the second-in key" and the probability of such an event is $1/2$ so that $\pi^-(i,j) = 1/2$.

Second case. U_i and U_j arrive in the good order in the initial array ($\tau(U_i) < \tau(U_j)$). The comparison does not always occur. In the phase when U_j is inserted into the left sub-array, this left sub-array already contains the key U_i. If this left sub-array contains one of the keys of the subset $\mathcal{U}_{[i,j[}$, then U_j "meets" (i.e., is compared to) this key before meeting U_i and remains on its right. Finally, the comparison between U_i and U_j occurs only if the subset $\mathcal{U}_{]i,j[}$ arrives after U_j. This defines the event "U_i is the first-in key and U_j is the second-in key inside the set $\mathcal{U}_{[i,j]}$". The probability of such an event and the expected value of symbol comparisons $\pi^+(i,j)$ is

$$\frac{1}{(j-i+1)(j-i)}.$$

2.5. Algorithm `BubSort`.
As its name says, the algorithm pushes the smallest keys to the left of the array as the air bubbles on to the surface of a liquid. The algorithm performs $n-1$ phases. During each phase, the algorithm steps through the array, compares each pair of adjacent keys and swaps them if they are in the wrong order. The i-th phase aims at finding the key of rank i and place it in the position i of the array. After the i-th phase, the keys of $\mathcal{U}_{[1..i]}$ are at their right places. The `BubSort` algorithm may perform several comparisons

between two keys U_i and U_j. We are now interested in the first comparison between U_i and U_j and we distinguish two cases:

First case. U_i and U_j arrive in the right order in the initial array ($\tau(U_i) < \tau(U_j)$). If there is one key of $\mathcal{U}_{]i,j[}$ which arrives after U_i and before U_j, it will stay between U_i and U_j in the array thereafter, and will prevent U_i and U_j from meeting each other. If it arrives after U_j, it will eventually come between U_i and U_j in the array before these two keys meet each other. Hence, there is a comparison between U_i and U_j only if all the keys of the subset $\mathcal{U}_{]i,j[}$ arrive before both U_i and U_j. This coincides with the event "the key U_j is the last-in and the key U_i arrived just before inside the subset $\mathcal{U}_{[i,j]}$". The probability that the first comparison between U_i and U_j occurs is

$$\frac{1}{(j-i+1)(j-i)}.$$

Second case. U_i and U_j arrive in the wrong order in the initial array ($\tau(U_j) < \tau(U_i)$). The first comparison between U_i and U_j occurs just before they are swapped. The probability of the event "U_j is the first-in key and U_j is the second-in key in $\{U_i, U_j\}$" is $1/2$.

Subsequent comparisons. There might be subsequent comparisons between two keys. Note that, in both previous cases, immediately after the first comparison (either positive or negative) U_i and U_j are in the right order and in consecutive positions. A necessary condition for having at least one subsequent comparison between U_i and U_j is that all the keys of $\mathcal{U}_{]i,j[}$ are still on the left of U_i after this point (for the same reasons exposed previously in Case 1). Now we also remark that any key U_ℓ with $\ell \in [1, i[$ which arrived after $\mathcal{U}_{]i,j[}$ and before U_i in the first case, and after $\mathcal{U}_{]i,j[}$ and before U_j in the second case, will be the cause of a stop of key U_i during some latter phases (such a key U_ℓ will never be swapped with U_i because of its smaller value). Also each time a key U_i is stopped during a phase by a key from $\mathcal{U}_{[1,i[}$, the set of keys from $\mathcal{U}_{[1,i[}$ between $\mathcal{U}_{]i,j[}$ and U_i decreases by one during the same phase. After such a phase, as all keys to the right of U_i are in $\mathcal{U}_{[j,n]}$, the key U_j during the next phase will be swapped until reaching U_i (and results in a comparison). In conclusion the number of subsequent comparisons is exactly the number of keys from $\mathcal{U}_{[1,i[}$ which arrived after $\mathcal{U}_{]i,j[}$ and before U_i in the first case and before U_j in the second case. For any $\ell \in [1\,..\,i[$, the probabilities that U_ℓ arrives after $\mathcal{U}_{]i,j[}$ and before U_i (and U_j arrives after U_i — Case 1) or after $\mathcal{U}_{]i,j[}$ and before U_j (and U_i arrives after U_j — Case 2) have the same expression

$$\frac{1}{(j-i+2)(j-i+1)(j-i)}.$$

Using independence of events for $\ell \in [1, i[$, this yields that the mean number of subsequent (positive) comparisons (summing up for both Cases 1 an 2) is

$$\frac{2(i-1)}{(j-i+2)(j-i+1)(j-i)}.$$

To conclude, one has

$$\pi^+(i,j) = \frac{1}{(j-i+1)(j-i)} + \frac{2(i-1)}{(j-i+2)(j-i+1)(j-i)}, \quad \pi^-(i,j) = \frac{1}{2}.$$

3. Second Step: Automatic transfer.

We start with expressions for $\pi(i,j)$ obtained in the previous Section, and, with Equation 1.8, we transfer them into expressions for $\widehat{\pi}(u,t)$. There are two algorithms for which $\widehat{\pi}(u,t)$ contains a constant term $1/2$ —namely `InsSort` and `BubSort`— and we denote by $\widetilde{\pi}(u,t)$ the term $\widehat{\pi}(u,t)$ from which this constant is removed. This means:

$$\widetilde{\pi}(u,t) := \begin{cases} \widehat{\pi}(u,t) - 1/2 & \text{for } \texttt{InsSort} \text{ and } \texttt{BubSort} \\ \widehat{\pi}(u,t) & \text{for } \texttt{QuickSort}. \end{cases}$$

This random variable is always expressed as a linear combination of "basic" functions whose Poisson expectations are easy to compute, as it is shown in Proposition 3.1. It is then possible to compute in an automatic way the coefficients $\varphi(k,u,t)$ described in Equation (1.14) and their analytic liftings $\varpi(s,u,t)$.

3.1. Main principles of the automatic transfer.

We first explain how we transfer the mean number $\pi(i,j)$ of key-comparisons into the Dirichlet terms $\varpi(s,u,t)$.

Proposition 3.1. *Consider a variable X which follows a Poisson law of rate Z, and, for $m \geq 1$, the variable*

$$G_m(X) := \frac{1}{(X+1)(X+2)\dots(X+m)}.$$

Then, the following holds:

(a) For any of the three algorithms, the random variable $\widetilde{\pi}(u,t)$, equal to $\widehat{\pi}(u,t)$ up to the possible constant term $1/2$, can be expressed in the "basis" G_m, as displayed in the second column of the table in Figure 5.

(b) Denote by $F_m(Z)$ the expectation of the variable $G_m(X)$. Then, the two sequences

$$\beta_m(n,\lambda) = (-1)^n n! [Z^n] \left(Z^2 F_m(\lambda Z) \right), \qquad \gamma_m(n,\lambda) := (-1)^n n! [Z^n] \left(Z^3 F_m(\lambda Z) \right).$$

admit the following expressions, resp. for $n > 1$ and $n > 2$,

$$\beta_m(n,\lambda) = \frac{1}{(m-1)!} \frac{n(n-1)}{n+m-2} \lambda^{n-2}, \qquad \gamma_m(n,\lambda) = \frac{-1}{(m-1)!} \frac{n(n-1)(n-2)}{n+m-3} \lambda^{n-3}. \tag{3.1}$$

(c) For any of the three algorithms, there exists an integer σ_0, for which the coefficients $\varphi(n,u,t)$ of the density $\phi_Z(u,t)$ can be expressed for $n > \sigma_0$ as a linear combination of $\beta_m(n,\lambda)$ and $\gamma_m(n,\lambda)$ for $\lambda \in \{u, t, t-u\}$, as displayed in the third column of Figure 5. The integer σ_0 is displayed in the fourth (and last) column of Figure 5.

Algorithms	$\widetilde{\pi}(u,t)$ (in the "basis" π_i)	$\varphi(n,u,t)$ (in the "basis" β_i, γ_j)	σ_0
`QuickSort`	$2[G_1(N_{[u,t[}) - G_2(N_{[u,t[})]$	$2[\beta_1(n,t-u) - \beta_2(n,t-u)]$	1
`InsSort`	$G_2(N_{[u,t[})$	$\beta_2(n,t-u)$	2
`BubSort`	$G_2(N_{[u,t[}) + 2N_{[0,u[} \cdot G_3(N_{[u,t[})$	$\beta_2(n,t-u) + 2u\gamma_3(n,t-u)$	2

Figure 5: Automatic transfer

Proof.
Assertion (a). We begin with the expressions of $\pi(i,j)$ displayed in Figure 4, and we obtain with (1.8) expressions for the random variable $\widetilde{\pi}(u,t)$ displayed in the second column of Figure 5.

Assertion (b). Recall that

$$F_m(Z) := \mathbb{E}_Z[G_m(X)] = e^{-Z} \sum_{k\geq 0} \pi_m(k) \frac{Z^k}{k!}.$$

We first compute the coefficients $\alpha_m(n)$ of $F_m(Z)$

$$\alpha_m(n) := (-1)^n n! [Z^n] F_m(Z) = \frac{1}{(m-1)!} \frac{1}{n+m}. \tag{3.2}$$

Then, the coefficients $\beta_m(n,\lambda)$ and $\gamma_m(n,\lambda)$ are related to $\alpha_m(n)$, resp. for $n>1$ and $n>2$

$$\beta_m(n,\lambda) = n(n-1)\,\lambda^{n-2}\alpha_m(n-2), \qquad \gamma_m(n,\lambda) = -n(n-1)(n-2)\,\lambda^{n-3}\alpha_m(n-3),$$

which proves, with (3.2), the expressions (3.1) of Assertion (b).

Assertion (c). Now, the third column is obtained from the second one by "taking the expectations" in the Poisson model, multiply by Z^2 and extracting the coefficient of order n. All the expressions of the second column are linear combinations, except the term $N_{[0,u[} \cdot \pi_3(N_{]u,t[})$, which involves the product of two independent variables, whose expectation is thus the product of expectations, namely

$$Z^2 \cdot \mathbb{E}_Z(N_{[0,u[}) \cdot \mathbb{E}_Z\left(G_3(N_{]u,t[})\right) = u \cdot Z^3 F_3(Z(t-u)).$$

□

The explicit expressions of $\varphi(n,u,t)$ deduced from the decompositions described in the third column of Figure 5 together with the expressions (3.1) yield expressions for $\varpi(s,u,t)$ reported in the following Figure 6 (for $s>\sigma_0$). We recall that the link between $\varphi(n,u,t)$ and $\varpi(s,u,t)$ is essentially — but not only — a change of variable $n \to s$. The precise link was described in Section 1.6.

3.2. Summary of the results for Step 2.
We then obtain the expressions for the analytic lifting $\varpi(s,u,t)$, via the "automatic" derivation taking into account the similar expressions for quantities $\pi(i,j)$.

Proposition 3.2. *Denote by $\varpi(s,u,t)$ the function which provides an analytical lifting of the sequence $\varphi(n,u,t)$ defined in Eq. (1.13), and by σ_0 the integer which defines the domain $\Re s > \sigma_0$ of validity of this lifting. Then, for any of the three algorithms, the value of the real σ_0 is provided in the second column of Figure 6 and the functions $\varpi(s,u,t)$ admit the expressions described in the third column of Figure 6.*

Algorithms	σ_0	$\varpi(s,u,t)$ on $\{\Re s > \sigma_0\}$
`QuickSort`	1	$2(t-u)^{s-2}$
`InsSort`	2	$(s-1)(t-u)^{s-2}$
`BubSort`	2	$(s-1)(t-u)^{s-3}[t-(s-1)u]$

Figure 6: Results for Step 2.

4. Third Step — Study of the mixed Dirichlet series

We start with the "local" expression of $\varpi(s,u,t)$ and obtain in Proposition 4.1 the expressions for the series $\varpi(s)$ thanks to a sum of integrals over all the fundamental triangles $\mathcal{T}_w$, as described in Eq. (1.16). Proposition 4.2 states the main relations between tameness of the source and tameness of the mixed Dirichlet series $\varpi(s)$. Then, Proposition 4.3 describes the constants which intervene in the dominant term of $S(n)$. The remainder of Section 4 is devoted to the proofs of Propositions 4.2 and 4.3.

4.1. Expression for the mixed Dirichlet series.
We describe here the expression of the Dirichlet series $\varpi(s)$ as a function of the geometry of the source, described by the fundamental intervals $\mathcal{I}_w := [a_w, b_w]$ of length $p_w = b_w - a_w$. More precisely, the Dirichlet series $\Lambda(s)$ of the source, defined in (1.2), or its extension $\Lambda[F]$ defined in (1.21) will play a fundamental role in the sequel.

Proposition 4.1. *Consider any (non-ambiguous) source, together with the fundamental intervals $[a_w, b_w]$ defined in (1.1) and its Dirichlet series defined in Eq. (1.2). Then, for any of the three algorithms, the mixed Dirichlet series $\varpi(s)$ (defined in Section 1.6) admit in the domain $\Re s > \sigma_0$, the expressions displayed in the second column of Table of Figure 7, together with the values of σ_0 in the third column. Depending on the value of*

σ_0 the mean number $S(n)$ of symbol comparisons is

$$S(n) = \sum_{k=2}^{n} (-1)^k \binom{n}{k} \varpi(k) \qquad \textit{(if } \sigma_0 = 1\textit{)},$$

$$S(n) = \binom{n}{2} \frac{\Lambda(2)}{2} + \sum_{k=3}^{n} (-1)^k \binom{n}{k} \varpi(k) \qquad \textit{(if } \sigma_0 = 2\textit{)}.$$

Remark. This is a reformulation of Eq. (1.19-1.20) using the fact that $\varphi(2) = \Lambda(2)/2$.

Algorithms	$\varpi(s)$	$\varpi(s)$	σ_0	Main term of $\varpi(s)/(s-\sigma_0)$
`QuickSort`	$\dfrac{2\Lambda(s)}{s(s-1)}$	$\dfrac{2}{s(s-1)} \displaystyle\sum_{w\in\Sigma^\star} p_w^s$	1	$\dfrac{2}{h(\mathcal{S})} \dfrac{1}{(s-1)^3}$
`InsSort`	$\dfrac{\Lambda(s)}{s}$	$\dfrac{1}{s} \displaystyle\sum_{w\in\Sigma^\star} p_w^s$	2	$\dfrac{c(\mathcal{S})}{2} \dfrac{1}{(s-2)}$
`BubSort`	$-\Lambda[F_0](s-1)$	$-\displaystyle\sum_{w\in\Sigma^\star} a_w p_w^{s-1}$	2	$-\dfrac{1}{2h(\mathcal{S})} \dfrac{1}{(s-2)^2}$

Figure 7: Results for Step 3.

4.2. Relations between tameness of the source and tameness of the mixed Dirichlet series.

We now study the relation between various notions of tameness: tameness of the source defined in Section 1.8 and tameness of the mixed Dirichlet series, defined in Section 1.7.

Proposition 4.2. *The following holds for the mixed Dirichlet series $\varpi(s)$ relative to the three sorting algorithms:*

(*i*) *Each mixed Dirichlet series is closely related to the Dirichlet series $\Lambda(s)$ of the source, and this relation is displayed in the first column of the tabular of Figure 7.*

(*ii*) *Assume the source $\mathcal{S}$ to be Λ-tame. Then, the mixed Dirichlet series $\varpi(s)$ relative to the three sorting algorithms satisfy the following:*

- (*a*) `[QuickSort]` *$\varpi(s)$ is tame at $\sigma_0 = 1$ with order $k_0 = 2$.*
- (*b*) `[InsSort]` *$\varpi(s)$ is tame at $\sigma_0 = 1$ with order $k_0 = 1$.*
- (*c*) `[BubSort]` *$\varpi(s)$ is tame at $\sigma_0 = 2$ with order $k_0 = 1$.*

Moreover, the source $\mathcal{S}$ gives its shape of tameness to the series $\varpi(s)$.

4.3. Main constants of interest.

We now describe the main constants which intervene in the dominant terms of the singular expression of $\varpi(s)/(s-\sigma_0)$ at $s = \sigma_0$.

Proposition 4.3. *The constants of interest which intervene in the main terms displayed in the last column of Figure 7 are:*

(*i*) *The entropy $h(\mathcal{S})$ of the source.*
(*ii*) *The coincidence $c(\mathcal{S})$, namely the mean number of symbols needed to compare two random words produced by the source.*

The entropy $h(\mathcal{S})$ is defined in (1.3). The constant $c(\mathcal{S})$ satisfies

$$c(\mathcal{S}) = 2 \sum_{w \in \Sigma^\star} \int_{\mathcal{T}_w} du\, dt = \sum_{w \in \Sigma^\star} p_w^2 = \Lambda(2).$$

Here, $\mathcal{T}_w$ is the fundamental triangle defined in (1.6) and $\Lambda(s)$ is defined in (1.2).

The proofs of Propositions 4.2 and 4.3 are given in the following sections. In the sequel, σ denotes the real part of s, i.e., $\sigma := \Re s$.

4.4. Case of `QuickSort` and `InsSort`.

The integral of $\varpi(s,u,t) = (s-1)(t-u)^{s-2}$ on the fundamental triangle $\mathcal{T}_w$ equals $(1/s)p_w^s$. This entails the nice formulae for both $\varpi(s)$ (where Q and I stand respectively for `QuickSort` and `InsSort`)

$$\varpi_Q(s) = 2\frac{\Lambda(s)}{s(s-1)}, \qquad \varpi_I(s) = \frac{\Lambda(s)}{s}.$$

Then, the functions $s \mapsto \varpi(s)$ are tame at $s = 1$. Moreover, the shape of tameness of $\varpi(s)$ at $s = 1$ coincides with the shape of Λ-tameness of the source. For `InsSort`, the function $\varpi_I(s)$ has a simple dominant pole at $s = 1$ with a residue equal to $1/h(S)$, whereas, for `QuickSort`, the function $\varpi_I(s)$ has a dominant pole at $s = 1$ of order 2. Moreover, the singular expressions of the functions $\varpi(s)/(s-1)$ can be easily computed from the singular expression of $\Lambda(s)$.

4.5. Case of `BubSort`.

The integral of $\varpi(s,u,t) = (s-1)(t-u)^{s-3}[t-(s-1)u]$ on the fundamental triangle equals $-a_w {p_w}^{s-1}$. Then, the Dirichlet series $\varpi(s)$ admits the expression

$$\varpi(s) = -\sum_{w \in \Sigma^\star} a_w {p_w}^{s-1} = -\Lambda[F_0](s-1),$$

where $F_0(x,y) = x$. By hypothesis, the series $s \mapsto \Lambda[F_0](s)$ is tame at $s = 1$. Then, the series $\varpi(s)$ is tame at $s = 2$, with the same shape of tameness as the series $s \mapsto \Lambda[F_0](s)$. We now study its precise behaviour at $s = 2$. We remark, with the relation $\Lambda_\ell(1) = \Lambda_\ell(1)^2 = 1$, the equality

$$2\Lambda_\ell[F_0](1) = 2\sum_{w \in \Sigma^\ell} a_w p_w = 2\sum_{w \in \Sigma^\ell} \left[\sum_{w' < w} p_{w'} \right] p_w = \Lambda_\ell(1)^2 - \sum_{w \in \Sigma^\ell} p_w^2 = \Lambda_\ell(1) - \Lambda_\ell(2). \tag{4.1}$$

The series

$$L(s) := \sum_{\ell \geq 0} L_\ell(s) \qquad \text{with} \quad L_\ell(s) := 2\Lambda_\ell[F_0](s) - \Lambda_\ell(s),$$

is convergent at $s = 1$ and satisfies

$$L(1) = \sum_{\ell \geq 0} L_\ell(1) = - \sum_{\ell \geq 0} \Lambda_\ell(2) = -\Lambda(2) = -c(\mathcal{S}),$$

where $c(\mathcal{S})$ is the coincidence of the source defined in Proposition 2.5. Since $\Lambda(s)$ admits a simple pole at $s = 1$ with a residue equal to $1/h(\mathcal{S})$, then $\Lambda[F_0](s)$ admits a simple pole at $s = 1$ with a residue equal to $1/2h(\mathcal{S})$. More precisely, as the singular expansion of $\Lambda(s)$ at $s = 1$ is

$$\Lambda(s) = \frac{1}{h(\mathcal{S})} \frac{1}{s-1} + d(\mathcal{S}) + O(s-1),$$

the singular expansion of $\varpi(s)$ at $s = 2$ is

$$\varpi(s) = -\frac{1}{2h(\mathcal{S})} \frac{1}{s-2} + \frac{1}{2}(c(\mathcal{S}) - d(\mathcal{S})) + O(s-2).$$

Note that Equation (4.1) can be generalized to any function F of class $\mathcal{C}^1$. The sum of interest can be viewed as a Riemann sum on the fundamental intervals of depth ℓ, so that

$$\sum_{w \in \Sigma^\ell} F(a_w, b_w) p_w = I[F] + \rho_\ell[F],$$

with for some constant C

$$I[F] := \int_0^1 F(t,t)dt \quad \text{and} \quad |\rho_\ell[F]| \leq C \cdot \sup_{(x,y)\in[0,1]^2} \left| \frac{\partial}{\partial x} F(x,y) \right| \cdot \Lambda_\ell(2)\,,$$

Then, for any function F whose integral $I[F]$ is not zero, the Dirichlet series $\Lambda[F](s)$ has a residue at $s = 1$ equal to $I[F]/h(\mathcal{S})$.

5. Final step and Discussion

In this section, we obtain the final result, namely the asymptotic expansion of the mean number $S(n)$ of symbol comparisons.

5.1. The final result.

With the results of the previous Section, it is now possible to use the Rice formula and obtain the asymptotics of the mean number $S(n)$ of symbol comparisons. We start with the expression of the mixed Dirichlet $\varpi(s)$ obtained in Proposition 4.1 together with Propositions 4.2 and 4.3, and apply the main principles described in Section 1.6.

Theorem 5.1. *Consider a (non-ambiguous) source $\mathcal{S}$, assumed to be Λ-tame. Then, the mean number $S(n)$ of symbol comparisons performed by each sorting algorithm on a*

sequence of n words independently drawn from the same source $\mathcal{S}$ admits the asymptotic behaviour described in Table of Figure 8.

(*i*) [Dominant terms.] *The constants in the dominant terms are already described in Proposition 4.3.*

(*ii*) [Subdominant terms.] *Here, the constants κ_i in the subdominant terms involve the Euler constant γ together with the subdominant constant of the source $d(\mathcal{S})$ equal to the constant term in the singular expansion of $\Lambda(s)$ at $s = 1$,*

$$d(\mathcal{S}) = \lim_{s\to 1}\left[\Lambda(s) - \frac{1}{h(\mathcal{S})}\frac{1}{s-1}\right],$$

under the form

$$\kappa_0 = \frac{2}{h(\mathcal{S})}(\gamma - 2) + 2d(\mathcal{S}), \qquad \kappa_1 = \frac{1}{8h(\mathcal{S})}(2\gamma - 3) + \frac{d(\mathcal{S})}{4}.$$

(*iii*) [Error terms.] *Assuming a Λ-tame source with a given shape, we have*

- *if the source has a S-shape with width δ, then $E(n) = O(n^{1-\delta})$;*
- *if the source has a H-shape with exponent ρ, then $E(n) = n \cdot O\left(\exp[-(\log n)^{\rho}]\right)$;*
- *if the source has a P-shape with width δ, then $E(n) = n \cdot \Phi(n) + O(n^{1-\delta})$ where $n \cdot \Phi(n)$ is the expansion given by the family of imaginary poles (s_k) of $\Lambda(s)$.*

Remarks. The constant κ_2 depends on other terms in singular expansions at $s = 1$, and is not computed here. Note that the computation of the subdominant term for `InsSort` needs the singular expansion of $\varpi(s)/(s-1)$ at $s = 1$. This subdominant constant $d(\mathcal{S})$ is often easy to compute, for instance for a binary memoryless source $\mathcal{B}_p$ of probabilities $(p, 1-p)$,

$$d(\mathcal{B}_p) = \frac{1}{h(\mathcal{B}_p)^2}\; p\log^2 p + (1-p)\log^2(1-p).$$

Algorithms	$K(n)$	Dominant term of $S(n)$	Subdominant terms of $S(n)$	Remainder term of $S(n)$
`QuickSort`	$2n\log n$	$\frac{1}{h(S)}\, n\log^2 n$	$\kappa_0 n\log n \,+\, \kappa_2 n$	$E(n)$
`InsSort`	$\frac{n^2}{4}$	$\frac{c(\mathcal{S})}{4}\, n^2$	$\frac{1}{h(S)} n\log n \,+\, \left(\kappa_0 - \frac{c(\mathcal{S})}{4}\right) n$	$E(n)$
`BubSort`	$\frac{n^2}{2}$	$\frac{1}{4h(S)}\, n^2\log n$	$\left(\kappa_1 + \frac{c(\mathcal{S})}{4}\right) n^2$	$nE(n)$

Figure 8: Results for Theorem 1.

5.2. Beyond Λ-tameness?

In this paper, we insist on sources which are Λ-tame, as they are the most natural. However, our results can be extended to other sources, whose Dirichlet series Λ fulfills more general tameness properties.

Proposition 5.2. *Consider a source $\mathcal{S}$ and its Dirichlet series $\Lambda(s)$. The following holds:*

(*i*) *If the Dirichlet series Λ is tame at $s = s_0$, with $s_0 \in]1, 2[$ and order 1, then the asymptotic order of the mean number $S(n)$ of symbol comparisons performed by each of the three algorithms is described in the second column of Figure 9.*

(*ii*) *If the Dirichlet series is Λ-tame at $s = 1$ with order $k_0 \geq 1$, then the asymptotic order of the mean number $S(n)$ of symbol comparisons performed by each of the three algorithms is described in the third column of Figure 9.*

Algorithms	Asymptotic order of $S(n)$ when Λ is tame at $s = s_0$, ($s_0 \in]1, 2[$) with order 1	Asymptotic order of $S(n)$ when Λ is tame at $s = 1$ with order k_0
`QuickSort`	n^{s_0}	$n \log^{1+k_0} n$
`InsSort`	n^2	n^2
`BubSort`	n^{1+s_0}	$n^2 \log^{k_0} n$

Figure 9: Results for Proposition 5.2.

This result applies to intermittent sources with parameter a defined in Section 1.2. More precisely, Assertion (*i*) applies for $a \in]1, 1/2[$, and Assertion (*ii*) applies for $a = 1$, with $k_0 = 2$.

5.3. Robustness.

We now compare the asymptotic estimates for the two mean numbers, the mean number $K(n)$ of key-comparisons (column 2 of Figure 8) and the mean number $S(n)$ (column 3 of Figure 8). There are two types of algorithms

(*a*) The "robust" algorithms for which $K(n)$ and $S(n)$ are of the same order. This is the case for only one algorithms, the `InsSort` algorithm, and the ratio $S(n)/K(n)$ involves the coincidence $c(\mathcal{S})$, precisely described in Section 4.3.

(*b*) The algorithms for which $S(n)$ and $K(n)$ are not of the same order, here `QuickSort` and `BubSort`. In both cases, the ratio $S(n)/K(n)$ satisfies

$$\frac{S(n)}{K(n)} \sim \frac{1}{2h(\mathcal{S})} \log n. \tag{5.1}$$

We will see later in Section 6.6 that the same ratio also appears in lower bounds.

6. Alternative proofs and lower bounds.

In Section 6.1, we recall the approach due to Seidel [24] together with the notion of faithfulness he introduced (Section 6.2). We show that this approach leads to analysis of trie parameters (Section 6.3); we then make a "detour" via trie parameters in Section 6.4, and we describe the two main methods that can be used in such analyses: the Rice methodology and the Poisson-Mellin approach; when the first one may be applied (this is the case for our analyses in previous sections), this greatly simplifies the probabilistic analysis; we also explain how to deal with the second method when the first one cannot be applied. Then, in Section 6.5, we return to faithful sorting algorithms and describe how to mix the two approaches (Seidel's one and the analysis of Section 6.3). Finally, we show in Section 6.6 that combining the approach of Seidel and ours leads to a lower bound for the mean number of symbol comparisons of any sorting algorithm using words emitted by a Λ-tame source $\mathcal{S}$.

6.1. Another way for relating the number of key comparisons and symbol comparisons.

Seidel [24] proves the following result that is now described in our framework.

Proposition 6.1. [Seidel] *Consider a set $\mathcal{U}$ of words that are independently emitted by the same source $\mathcal{S}$. Denote by $\mathcal{S}[\mathcal{U},\sigma]$ (resp. $\mathcal{K}[\mathcal{U},\sigma]$) the number of symbol (resp. key) comparisons performed by the algorithm when the input set $\mathcal{U}$ is under the permutation σ. More generally, for a subset $\mathcal{V}\subset\mathcal{U}$, denote by $\mathcal{S}[\mathcal{U},\mathcal{V},\sigma]$ (resp. $\mathcal{K}[\mathcal{U},\mathcal{V},\sigma]$) the number of symbol (resp. key) comparisons performed by the algorithm on the subset $\mathcal{V}$ when the input set $\mathcal{U}$ is under the permutation σ. The values $\mathcal{S}[\mathcal{U}],\mathcal{S}[\mathcal{U},\mathcal{V}]$, (resp. $\mathcal{K}[\mathcal{U}],\mathcal{K}[\mathcal{U},\mathcal{V}]$) are the expected values of $\mathcal{S}[\mathcal{U},\sigma],\mathcal{S}[\mathcal{U},\mathcal{V},\sigma]$ (resp. $\mathcal{K}[\mathcal{U},\sigma],\mathcal{K}[\mathcal{U},\mathcal{V},\sigma]$) when σ is a uniform random permutation of $\mathfrak{S}_n$.*

Consider the subset $\mathcal{U}_{\langle w\rangle}$ of $\mathcal{U}$ which gathers all the words which begin by the prefix w, and the set $P(\mathcal{U})$ of common prefixes of $\mathcal{U}$, defined as the prefixes w for which the cardinality $|\mathcal{U}_{\langle w\rangle}|$ is at least equal to 2. The following relations hold:

$$\mathcal{S}[\mathcal{U},\sigma]=\sum_{w\in P(\mathcal{U})}\mathcal{K}[\mathcal{U},\mathcal{U}_{\langle w\rangle},\sigma],\qquad \mathcal{S}[\mathcal{U}]=\sum_{w\in P(\mathcal{U})}\mathcal{K}[\mathcal{U},\mathcal{U}_{\langle w\rangle}]. \tag{6.1}$$

Proof. The following equality

$$\mathcal{S}[\mathcal{U},\sigma]=\sum_{1\le i<j\le n}(c(U_i,U_j)+1)\,\mathcal{K}[\mathcal{U},\{U_i,U_j\},\sigma]$$

involves the coincidence $c(U_i,U_j)$ between the two keys U_i and U_j (namely the length of their longest common prefix). Seidel considers the subset $\mathcal{U}_{\langle w\rangle}$ of $\mathcal{U}$ which gathers all the words which begin by the prefix w, and the set $P(\mathcal{U})$ of common prefixes of $\mathcal{U}$, defined as the prefixes w for which the cardinality $|\mathcal{U}_{\langle w\rangle}|$ is at least equal to 2. All these sets only depend on $\mathcal{U}$, not on the permutation σ.

Seidel remarks that $c(U_i,U_j)+1$ is also the number of prefixes $w\in\mathcal{P}(\mathcal{U})$ —including the empty prefix— which are shared by U_i and U_j. This is also the number of subsets of

the form $\mathcal{U}_{\langle w\rangle}$ — including the total set $\mathcal{U}$ — which contain U_i and U_j. Then,

$$\mathcal{S}[\mathcal{U},\sigma] = \sum_{1\le i<j\le n} \sum_{\substack{w\in P(\mathcal{U})\\ U_i,U_j\in\mathcal{U}_{\langle w\rangle}}} \mathcal{K}[\mathcal{U},\{U_i,U_j\},\sigma] = \sum_{w\in P(\mathcal{U})} \sum_{\substack{1\le i<j\le n\\ U_i,U_j\in\mathcal{U}_{\langle w\rangle}}} \mathcal{K}[\mathcal{U},\{U_i,U_j\},\sigma].$$

Now the equality

$$\sum_{\substack{1\le i<j\le n\\ U_i,U_j\in\mathcal{U}_{\langle w\rangle}}} \mathcal{K}[\mathcal{U},\{U_i,U_j\},\sigma] = \mathcal{K}[\mathcal{U},\mathcal{U}_{\langle w\rangle},\sigma],$$

holds, and entails the first equality of Proposition 6.1. As the subsets $\mathcal{U}_{\langle w\rangle}$ do not depend on the permutation σ, averaging over σ leads to the second equality of Proposition 6.1. □

6.2. Faithfulness.
Seidel introduces the notion of a faithful algorithm, and we consider here a slightly different notion, the notion of a strongly faithful algorithm, on which we give an alternative point of view. We first recall the notation $\mathcal{U}_{[i,j]}$ (already used in Section 2.2) which denotes the subset formed with the keys of $\mathcal{U}$ whose rank k belongs to the interval $[i,j]$.

Definition 5. [Seidel] An algorithm is strongly faithful if, for any $n\ge 2$, any subset $\mathcal{U}$ of cardinality n, and any pair (i,j), with $1\le i<j\le n$, the mean number of key comparisons $\mathcal{K}[\mathcal{U},\mathcal{U}_{[i,j]}]$ performed by the algorithm only depends on the cardinality $j-i+1$ of the subset $\mathcal{U}_{[i,j]}$.

There is an easy translation of this notion in our framework.

Lemma 6.2. *For a sorting algorithm $\mathcal{A}$, the following three assertions are equivalent*
(*a*) *The algorithm is* strongly faithful.
(*b*) *The mean number of key comparisons $\pi(i,j)$ performed by $\mathcal{A}$ between the two keys U_i and U_j only depends on the difference $j-i$ between their ranks.*
(*c*) *The density $\phi_Z(u,t)$ of $\mathcal{A}$ in the Poisson model only depends on the difference $t-u$.*

The algorithms `Quicksort` *and* `InsSort` *are strongly faithful. The third algorithm* `BubSort` *is not strongly faithful.*

Proof. Denote by $P(i,j) := \mathcal{K}[\mathcal{U},\mathcal{U}_{[i,j]}]$ the expectation of the total number of key comparisons between any pair of two keys of $\mathcal{U}_{[i,j]}$. By definition, an algorithm is strongly faithful if $P(i,i+k)$ does not depend on i, and only depends on k.

The relation

$$P(i,i+\ell) = \sum_{\substack{(i',j')\\ i\le i'<j'\le i+\ell}} \pi(i',j') = P(i,i+\ell-1) + \pi(i,i+\ell) + \sum_{k=1}^{\ell-1} \pi(i+k,i+\ell) \quad (6.2)$$

holds for $\ell\ge 2$ between the two sequences $P(i,j)$ and $\pi(i,j)$. We will use it to prove that the following two assertions are equivalent, with a recurrence on ℓ.

(a) For any $k \leq \ell$, the expected values $P(i, i+k)$ do not depend on i.
(b) For any $k \leq \ell$, the expected values $\pi(i, i+k)$ do not depend on i.

If $\ell = 1$, there is only one pair of keys in $\mathcal{U}_{[i,i+1]}$ and $P(i, i+1) = \pi(i, i+1)$. Then, the lemma is true for $\ell = 1$.

Assume now that the lemma holds for $k < \ell$. We prove that it holds for $k = \ell$.

Assume first that Assertion (b) holds for $k \leq \ell$. Then, none of the terms $\pi(i+k, i+\ell)$ for $k \in [0\,..\,\ell-1]$ depends on i. Furthermore, by recurrence hypothesis, Assertion (a) holds for $k \leq \ell - 1$, and $P(i, i+\ell-1)$ does not depend on i. Then, with Eq. (6.2), it is the same for $P(i, i+\ell)$. and Assertion (a) holds for $k \leq \ell$.

Conversely, assume that Assertion (a) holds for $k \leq \ell$. Then, none of the two terms $P(i, i+\ell)$ or $P(i, i+\ell-1)$ depends on i. Furthermore, by recurrence hypothesis, Assertion (b) holds for $k < \ell$ and all the terms $\pi(i+k, i+\ell)$ are independent of i for $k \in [1\,..\,\ell]$. Then, with Eq. (6.2), it is the same for $\pi(i, i+\ell)$, and Assertion (b) holds for $k \leq \ell$. □

For a strongly faithful algorithm, the mean number $\mathcal{K}[\mathcal{U}, \mathcal{U}_{\langle w \rangle}]$ only depends on the cardinality N_w of $\mathcal{U}_{\langle w \rangle}$. It equals $K(N_w)$, where $K(n)$ is the mean number of key comparisons of the algorithm in the permutation model, and Relation (6.1) entails the equality

$$\mathcal{S}[\mathcal{U}] = \sum_{w \in \mathcal{P}(\mathcal{U})} K(N_w), \tag{6.3}$$

where N_w is the number of words of $\mathcal{U}$ which begin with the prefix w. We remark that, for any sorting algorithm, the equalities $K(0) = K(1) = 0$ hold. Then, the previous relation can be written as

$$\mathcal{S}[\mathcal{U}] = \sum_{w \in \mathcal{P}(\mathcal{U})} K(N_w) = \sum_{w \in \Sigma^\star} K(N_w).$$

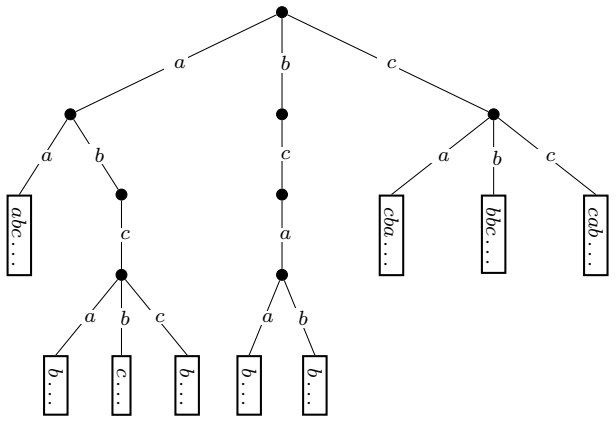
6.3. A relation between faithful sorting algorithms and trie parameters. Consider, more generally, a function $f : \mathbb{N} \to \mathbb{R}$ which satisfies $f(0) = f(1) = 0$ and $f(k) \geq 0$ for $k \geq 2$, and a random variable defined by the relation

$$\mathcal{R}[\mathcal{U}] := \sum_{w \in \Sigma^\star} f(N_w), \tag{6.4}$$

where N_w is the number of words of $\mathcal{U}$ which begin with the prefix w. We now explain why such a random variable defines an additive parameter on the trie $T(\mathcal{U})$.

The trie $T(\mathcal{U})$. A trie is a tree structure, used as a dictionary, which compares words via their prefixes. Since its introduction [17, 5] the trie has become a fundamental data structure in computer science [11]. Given a finite set $\mathcal{U} = \{U_1, U_2, \ldots, U_n\}$ formed with n (infinite) words emitted by the source, the trie $T(\mathcal{U})$ built on the set $\mathcal{U}$ is defined recursively by the following three rules:

(i) If $|\mathcal{U}| = 0$, $T(\mathcal{U}) = \emptyset$.
(ii) If $|\mathcal{U}| = 1$, $\mathcal{U} = \{U\}$, $T(\mathcal{U})$ is a leaf labeled by U.



$U_1 = aaabc\ldots$
$U_2 = abcab\ldots$
$U_3 = abcbc\ldots$
$U_4 = abccb\ldots$
$U_5 = bcaab\ldots$
$U_6 = bcabb\ldots$
$U_7 = cacba\ldots$
$U_8 = cbbbc\ldots$
$U_9 = cccab\ldots$

Figure 10: The trie $T(\mathcal{U})$ associated with a set $\mathcal{U}$ of nine (infinite) words on the alphabet $\Sigma := \{a, b, c\}$

(*iii*) If $|\mathcal{U}| \geq 2$, then $T(\mathcal{U})$ is formed with an internal node and r subtries respectively equal to

$$T(\underline{\mathcal{U}}_{\langle 0\rangle}), \ldots, T(\underline{\mathcal{U}}_{\langle r-1\rangle}),$$

where $\underline{\mathcal{U}}_{\langle\sigma\rangle}$ denotes the subset consisting of words of $\mathcal{U}_{\langle\sigma\rangle}$, stripped of their initial symbol σ. If the set $\mathcal{U}_{\langle\sigma\rangle}$ is nonempty, the edge which links the subtrie $T(\underline{\mathcal{U}}_{\langle\sigma\rangle})$ to the internal node is labelled with the symbol σ.

Then, the internal nodes are used for directing the search, and the leaves contain the words of $\mathcal{U}$. There are as many leaves as words in $\mathcal{U}$. The internal nodes are labelled by prefixes w for which the cardinality N_w of the subset $\mathcal{U}_{\langle w\rangle}$ is at least 2 (see Figure 10).

Additive parameters on tries. Trie analysis aims at describing the average shape of a trie (number of internal nodes, external path length, height). We focus here on additive parameters, whose (recursive) definition exactly copies the (recursive) definition of the trie. Consider a function $f : \mathbb{N} \to \mathbb{R}$ which satisfies $f(0) = f(1) = 0$ and $f(k) \geq 0$ for $k \geq 2$, together with a random variable $R[\mathcal{U}]$, associated with f, and defined on the trie $T(\mathcal{U})$ as:

(*i*) If $|\mathcal{U}| \leq 1$, then $R[\mathcal{U}] = 0$;
(*ii*) If $|\mathcal{U}| \geq 2$, then $R[\mathcal{U}] = f(|\mathcal{U}|) + \sum_{\sigma\in\Sigma} R[\mathcal{U}_{\langle\sigma\rangle}]$.

Iterating the recursion, we obtain exactly Equation (6.4). The cost f is the "toll" that is "paid" at each internal node of the trie. In particular, when $f(k) = 1$ for $k \geq 2$, the variable $\mathcal{R}[\mathcal{U}]$ equals the number of internal nodes in the trie $T(\mathcal{U})$, and when $f(k) = k$ for $k \geq 2$, the variable $\mathcal{R}[\mathcal{U}]$ equals the external path length of the trie $T(\mathcal{U})$. These trie parameters have been very deeply studied: first in the case when words are emitted by a simple source (see [25] for instance), and later on, when the words are produced by a general (non-ambiguous) source (see [3, 1]).

However, even for simple sources, these existing analyses are usually performed with the Poisson–Mellin tools, need Depoissonization techniques, and do not precisely deal with the tameness of the source[5]. The following result has thus two main purposes: it first deals with the usual cases $f(k) = 1$ or $f(k) = k$ and explains how the Rice methodology provides in these cases very natural proofs, with precise error terms, that do not need Depoissonization techniques. It also makes more precise the role played by the tameness of the source. Second, it describes the method that can be used in the case of the "toll" $f(k) = k \log k$, where it does not seem possible to apply directly the Rice methodology.

6.4. Analysis of additive trie parameters.
The following result compares the two methodologies, the Rice methodology and the Poisson-Mellin approach, it is thus of independent interest, and also important in our context.

Proposition 6.3. *Consider, a source $\mathcal{S}$ assumed to be Λ-tame. For each set $\mathcal{U}$ of infinite words independently produced by $\mathcal{S}$, consider the trie $T(\mathcal{U})$ and a trie parameter $\mathcal{R}[\mathcal{U}]$ defined by Relation (6.4) from a toll function $f : \mathbb{N} \to \mathbb{R}$ which satisfies $f(0) = f(1) = 0$ and $f(k) \geq 0$ for $k \geq 2$. Then, the mean value $R(n)$ of the random variable $\mathcal{R}[\mathcal{U}]$ in the Bernoulli model $(\mathcal{B}_n, \mathcal{S})$ satisfies the following:*

(*i*) *In the case when $f(k) = k(k-1)$, then $R(n) = \Lambda(2)\, n(n-1)$.*
(*ii*) *Define the degree of f as $\deg(f) := \inf\{c, f(k) = O(k^c)\}$ and assume that $d := \deg f$ belongs to $[0, 2[$. Then, $R(n)$ is written as*

$$R(n) = \sum_{k=2}^{n} (-1)^k \binom{n}{k} \Lambda(k) r(k), \qquad \text{with} \qquad r(n) = \sum_{k=2}^{n} (-1)^k \binom{n}{k} f(k),$$

and there exist analytic liftings $\rho(s)$, for $n \mapsto r(n)$, and $\varpi(s)$, for $n \mapsto R(n)$, on the half-plane $\Re s > d$.
(*iii*) *In the case when $\rho(s)$ is tame at $s = \max(d, 1)$, the Rice methodology can be applied. This arises in the two cases $f(k) = 1$ and $f(k) = k$. The function ρ, and the asymptotic behavior of the mean value R are described in Figure 11.*
(*iv*) *In the case when $f(k) = k \log k$, the function $\rho(s)$ has a pole of order 2 at $s = 1$, and, even if it is not proven to be tame at $s = 1$, Poisson–Mellin tools prove the asymptotic behaviour for $R(n)$ given in Figure 11.*

Remark. There exist two extensions of Proposition 6.3 (that will be described in [26]), in the following cases:

(*i*) The source is not Λ-tame, but its Λ series fulfills nice tameness properties as described in Section 5.2.
(*ii*) The function f admits an analytic lifting that satisfies $f(n) = n^a \log^b n$, with $b \in \mathbb{N}$ and $a \in]0, 2[$.

[5] There is an exception in [1].

Toll function	Lifting $\rho(s)$	Lifting $\varpi(s)$	Dominant term of $R(n)$
$f(k)=1$	$s-1$	$(s-1)\Lambda(s)$	$\dfrac{n}{h(\mathcal{S})}$
$f(k)=k$	s	$s\Lambda(s)$	$\dfrac{1}{h(\mathcal{S})} n\log n$
$f(k)=k\log k$	$-\dfrac{\zeta'(s)}{\Gamma(-s)}+H_2(s)$	$-\dfrac{\zeta'(s)\Lambda(s)}{\Gamma(-s)}+\Lambda(s)H_2(s)$	$\dfrac{1}{2h(\mathcal{S})} n\log^2 n$

Figure 11: Results for Proposition 6.3.

Proof of Proposition 6.3. We first work inside the Poisson model $(\mathcal{P}_Z, \mathcal{S})$ where the cardinality N follows a Poisson law of rate Z. Then, the cardinality N_w follows a Poisson law of rate Zp_w. Denote by $F(Z)$ the expectation of the variable $f(N)$ and $G(Z)$ the expectation of the variable $\mathcal{R}[\mathcal{U}]$ in the Poisson model of rate Z, namely

$$F(Z)=e^{-Z}\sum_{k\geq 2}\frac{Z^k}{k!}f(k), \qquad G(Z)=e^{-Z}\sum_{k\geq 2}\frac{Z^k}{k!}R(k), \tag{6.5}$$

where $R(n)$ is the expectation of $\mathcal{R}[\mathcal{U}]$ in the Bernoulli model in $(\mathcal{B}_n, \mathcal{S})$. If we wish to adopt the technics "à la Rice", as in Section 1.5, we write $F(Z), G(Z)$ under the form

$$F(Z)=\sum_{n\geq 0}(-1)^n\frac{Z^n}{n!}r(n), \quad \text{with} \quad r(n)=\sum_{k=2}^{n}(-1)^k\binom{n}{k}f(k), \tag{6.6}$$

$$G(Z)=\sum_{n\geq 0}(-1)^n\frac{Z^n}{n!}\varphi(n), \quad \text{with} \quad \varphi(n)=\sum_{k=2}^{n}(-1)^k\binom{n}{k}R(k). \tag{6.7}$$

Averaging Relation (6.4) in the Poisson model of rate Z entails the equality

$$G(Z)=\sum_{w\in\Sigma^\star}\mathbb{E}_Z[f(N_w)]=\sum_{w\in\Sigma^\star}F(Zp_w). \tag{6.8}$$

Assertion (i). Consider first the case where $f(k)=k(k-1)$. Then, Relation (6.5) entails the equality $F(Z)=Z^2$, and, with (6.8), the equality $G(Z)=\Lambda(2)Z^2$. This implies the *exact* equality $R(n)=n(n-1)\,\Lambda(2)$.

Assertion (ii). Relations (6.6), (6.7) and (6.8) entail the equality

$$\varphi(n)=\left(\sum_{w\in\Sigma^\star}p_w^n\right)r(n)=\Lambda(n)r(n), \tag{6.9}$$

and, inverting the triangular set of relations (6.7) leads to the binomial relations

$$R(n) = \sum_{k=2}^{n} (-1)^k \binom{n}{k} \varphi(k), \qquad \text{and then} \quad R(n) = \sum_{k=2}^{n} (-1)^k \binom{n}{k} \Lambda(k) r(k).$$

If there exists an analytic lifting $\rho(s)$ of $r(n)$, there is an analytic lifting $\varpi(s)$ for $\varphi(n)$, equal to $\Lambda(s)\rho(s)$, and we can use the Rice Formula, as previously in Section 1.6, as soon as the analytic liftings can be proven tame. Previously, for each of the three algorithms, we computed such an analytic lifting $\varpi(s)$, described in the second column of Table of Figure 7, which is moreover proven to be tame. However, in the present general setting, the existence of a simple lifting $\varpi(s)$ for $\varphi(n)$ is not directly granted, but it can be obtained by the cycle "Poisson–Mellin–Newton–Rice" well described in [12], that we now state in our setting.

For a sequence f of general term $f(k)$, with $f(k) \in \mathbb{C}$, the valuation $\mathrm{val}(f)$ is the smallest index of non-zero elements of f and the degree $\deg(f)$ as the infimum of all c such that $f(k) = O(n^c)$. The "good" case for finding a simple lifting $\rho(s)$ of the sequence $r(n)$ arises when the inequality $\mathrm{val}(f) > \deg(f)$ holds, as we explain. Here, one always has $\mathrm{val}(f) = 2$, and when $\deg(f)$ satisfies $\deg(f) < 2$, the inequality $\mathrm{val}(f) > \deg(f)$ always holds.

In the case when the inequality $\mathrm{val}(f) > \deg(f)$ holds, we use the Mellin transform in order to build $\rho(s)$. The Mellin transform [13] transforms a function g into a function g^* of a complex variable s defined as

$$g^*(s) := \int_0^{+\infty} g(x)\, x^{s-1} dx.$$

When the sequence f satisfies $\mathrm{val}(f) = 2$ and $\deg(f) = d < 2$, the Mellin transform $F^*(s)$ of the function $F(Z)$ exists on the vertical strip $\langle -2, -d \rangle$, and

$$\begin{aligned} F^*(s) &= \sum_{k \geq 2} \frac{f(k)}{k!} \int_0^{\infty} e^{-z} z^k z^{s-1} dz = \sum_{k \geq 2} \frac{f(k)}{k!} \Gamma(k+s) \\ &= \Gamma(s) \left[\sum_{k=2}^{\infty} f(k) \frac{s(s+1)\dots(s+k-1)}{k!} \right]. \end{aligned}$$

If we now let

$$\rho(s) := \frac{F^*(-s)}{\Gamma(-s)} = \sum_{k=2}^{\infty} (-1)^k f(k) \frac{s(s-1)\dots(s-k+1)}{k!}, \tag{6.10}$$

the function $\rho(s)$ is expressed as a Newton interpolation series in the vertical strip $\langle d, 2 \rangle$, which moreover satisfies $\rho(n) = r(n)$, with (6.6). On general grounds, Newton series converge in half-planes, via classical results due to Nörlund [20, 21], and here the function $\rho(s)$ is analytically continuable on the half plane $\Re s > d$. Finally, the function $\rho(s)$ defined in (6.10) provides the analytic lifting we look for, on the half plane $\Re s > d$. As the Dirichlet series Λ of the source has a pole at $s = 1$, the product $s \mapsto \varpi(s) = \rho(s)\Lambda(s)$ is analytic on the halfplane $\Re s > \max(d, 1)$. And its tameness (needed to apply the Rice methodology) depends both on the relative position of d and 1, and the tameness of ρ.

We now give two instances of such a situation, in the context of Assertions (*iii*) and (*iv*).

Assertion (*iii*). We now consider the cases $f(k) = 1$ or $f(k) = k$. In both cases, the degree d satisfies $\max(d, 1) = 1$, and we study the tameness of ρ at $s = 1$. The sequences $r(n)$ satisfy respectively

$$r(n) = \sum_{k=2}^{n} (-1)^k \binom{n}{k} = 0 - 1 + n = n - 1$$

$$r(n) = \sum_{k=2}^{n} (-1)^k \binom{n}{k} k = -n \sum_{k=1}^{n-1} (-1)^k \binom{n-1}{k} = n,$$

and the analytic liftings $\rho(s)$ are respectively $s - 1$ and s, whereas the analytic liftings of the sequence φ are $\varpi(s) = (s-1)\Lambda(s)$ and $s\,\Lambda(s)$. In the case of a source Λ-tame, the analytic lifting $\varpi(s)$ is tame at $s = 1$, and we apply the Rice methodology.

Assertion (*iv*). The case $f(k) = k \log k$ is different. The degree d equals 1, and we study the tameness of ρ at $s = 1$. We write

$$F^*(s) = \sum_{k\geq 2} \frac{f(k)}{k!} \int_0^\infty e^{-z} z^k z^{s-1} dz = \sum_{k\geq 2} \frac{f(k)}{k} \frac{\Gamma(k+s)}{\Gamma(k)}.$$

The ratio of Gamma Functions can be estimated with the Stirling Formula,

$$\frac{\Gamma(k+s)}{\Gamma(k)} = \frac{(k+s)^{k+s}}{k^k} \frac{e^{-k-s}}{e^{-k}} \sqrt{\frac{k+s}{k}} \left[1 + O\left(\frac{1}{k}\right)\right] = k^s \left[1 + O\left(\frac{|s|}{k}\right)\right],$$

where the O-term is uniform with respect to k. Then, the Mellin transform of F satisfies, for $f(k) = k \log k$,

$$F^*(s) = \sum_{k\geq 2} k^s \log k \left[1 + O\left(\frac{|s|}{k}\right)\right] = -\zeta'(-s) + H_1(s), \tag{6.11}$$

where $H_1(s)$ is analytic and of polynomial growth in $\Re s < 0$. Then $F^*(-s)$ is tame at $s = 1$ with order 2, and the relation

$$\rho(s) = \frac{F^*(-s)}{\Gamma(-s)} = -\frac{\zeta'(s)}{\Gamma(-s)} + \frac{H_1(s)}{\Gamma(-s)},$$

proves that the function $\rho(s)$ has a pole of order 1 at $s = 1$. However, the function $1/\Gamma(-s)$, even though it is analytic on the halfplane $\Re(s) > 1$, is not tame there. And it does not seem possible to directly prove that $\rho(s)$ is tame at $s = 1$. We cannot *a priori* apply the Rice methodology, and we then follow the Poisson–Mellin approach, that we now describe.

We return to the Mellin transforms $F^*(s)$ and $G^*(s)$. Due to Relation (6.8), the function $G(Z)$ is an harmonic sum[6] with base function F and frequencies p_w. With

[6] The function g is an harmonic sum with base function f and frequencies μ_k if $g(z)$ is written as $g(z) = \sum_k f(\mu_k z)$.

classical properties [13], its Mellin transform $G^*(s)$ factorises as

$$G^*(s) = \left(\sum_{w \in \Sigma^\star} p_w^{-s} \right) \cdot F^*(s) = \Lambda(-s) \cdot F^*(s). \tag{6.12}$$

The singular expressions of $F^*(s)$ and $G^*(s)$ at $s = -1$ are, with (6.11) and (6.12),

$$F^*(s) \asymp \frac{1}{(s+1)^2}, \qquad G^*(s) \asymp \frac{1}{h(\mathcal{S})} \frac{1}{(s+1)^3}.$$

The tamenesses of $F^*(s)$ and $\Lambda(s)$ are enough to deduce, using standard Mellin inverse transform [13], the estimates, for $Z \to \infty$,

$$F(Z) = Z \log Z (1 + o(1)), \qquad G(Z) = \frac{1}{2h(\mathcal{S})} Z \log^2 Z (1 + o(1)). \tag{6.13}$$

Now, we wish to return in the Bernoulli model, with Depoissonization techniques, which need a good behaviour of $G(Z)$ with respect to cones. For $\theta < \pi/2$, the cone S_θ is the set of complex numbers Z whose argument $\arg Z$ satisfies the inequality $|\arg Z| \leq \theta$. We use the following theorem of [18].

Theorem A [Jacquet and Szpankowski] *Let $G(Z)$ be the Poisson transform of a sequence $R(n)$ that is assumed to be an entire function of Z. Assume that, in a linear cone S_θ (with $\theta < \pi/2$), the following two conditions simultaneously hold for some real numbers $A, B, r > 0, \beta$, and $\alpha < 1$:*

(*a*) *For $Z \in S_\theta$,* $\quad |Z| > r \Longrightarrow |G(Z)| \leq B|Z|^\beta$.

(*b*) *For $Z \notin S_\theta$,* $\quad |Z| > r \Longrightarrow |G(Z)e^Z| \leq A \exp(\alpha |Z|)$.

Then, one has $R(n) \sim G(n)$ for $n \to \infty$.

We apply this result to the Poisson transform $G(Z)$ of the sequence $R(n)$. Assertion (a) is easy to deduce from (6.13). For Assertion (b), we first study $F(Z)$ and observe that $F(Z)$ can be written as

$$F(Z)e^Z = Z^2 L(Z) \quad \text{with} \quad L(Z) = \sum_{k=0} \frac{Z^k}{k!} \ell(k) \quad \text{and} \quad \ell(k) := \frac{1}{k+1} \log(k+2). \tag{6.14}$$

Since the function $\ell(k)$ admits an analytical continuation to the half plane $\Re(Z) > 0$, we use a very useful result of [19].

Lemma B. *Let $\ell(Z)$ be an analytic continuation of a sequence $\ell(n)$ which is $O(|Z|^\beta)$ in a linear cone. Then, for some θ_0, and for all linear cones S_θ with $\theta < \theta_0$, there exist $\alpha < 1$ and $A > 0$ such that the exponential generating function $L(Z)$ of $\ell(n)$ satisfies*

$$Z \notin S_\theta \Longrightarrow |L(Z)| \leq A \exp(\alpha |Z|).$$

With Eq. (6.14), the sequence $\ell(k)$ satisfies hypotheses of Lemma B, and the conclusion holds for the exponential generating function $L(Z)$ of $\ell(n)$. This exponential bound is

then transfered to G, as we now explain. First, we have in accordance with (6.8) and (6.14)

$$G(Z)e^Z = e^Z \sum_{w\in\Sigma^\star} F(Zp_w) = Z^2 \sum_{w\in\Sigma^\star} p_w^2 L(p_w Z)\, \exp(Z - p_w Z).$$

We denote by x the real part of Z and consider the cone $\widehat{S}_\alpha$ defined by the inequality $x > \alpha|Z|$. When Z does not belong to $\widehat{S}_\alpha$, it is the same for all the complex numbers $p_w Z$, and, with Lemma B, each term of the previous sum satisfies the inequality

$$|L(p_w Z)\exp(Z - p_w Z)| \le A \exp\left(\alpha p_w |Z| + x(1 - p_w)\right) \le A\exp(\alpha|Z|).$$

Finally, we have shown:

$$Z \notin \widehat{S}_\alpha \Longrightarrow |G(Z)e^Z| \le B|Z|^2 \exp(\alpha|Z|) \qquad \text{with} \quad B := A\Lambda(2),$$

Now, for $|Z|$ large enough, and $Z \notin \widehat{S}_\alpha$, we obtain $|G(Z)e^Z| \le C\exp(\alpha'|Z|)$ with $\alpha' \in]\alpha, 1[$ and a given constant C. Finally, Assertion (b) of Theorem A holds. Applying Theorem A to $G(Z)$ entails the estimate $R(n) \sim G(n)$ and ends the proof for Assertion (iv) of Proposition 6.3.

□

6.5. An alternative proof for `QuickSort` and `InsSort`.
It is clear that two algorithms of the studied family —`QuickSort` and `InsSort`— are strongly faithful whereas the last one `BubSort` is not strongly faithful.

In the case of the two faithful algorithms, the following result easily follows from Proposition 6.3 and Relation (6.3). Then, the approach of Seidel, combined with our methods, provides an alternative approach for our main theorem, at least for the algorithms `QuickSort` and `InsSort`. However, this approach cannot be applied to `BubSort` that is not strongly faithful, and we only provide here the asymptotic main terms.

Theorem 6.4. *Consider a strongly faithful algorithm which sorts words that are independently drawn from the same source, assumed to be* Λ*-tame.*

(*i*) *If the mean number* $K(n)$ *of key comparisons is* $An^2 + O(n)$*, then the mean number of symbol comparisons satisfies* $S(n) = A\,c(\mathcal{S})\,n^2 + O(n\log n)$*, and* $S(n)/K(n)$ *is asymptotic to* $c(\mathcal{S})$*.*

(*ii*) *If the mean number* $K(n)$ *of key comparisons is* $A\,n\log n + O(n)$*, then the mean number* $S(n)$ *of symbols comparisons is asymptotic to* $A/(2h(\mathcal{S}))\cdot n\log^2 n$ *and* $S(n)/K(n)$ *is asymptotic to* $\log n/(2h(\mathcal{S}))$*.*

6.6. An asymptotic lower bound for $S(n)$.
Combining our methods described in Proposition 6.3 together with the approach of Seidel for lower bounds, we obtain an asymptotic lower bound $\underline{S}(n)$ for the mean number of symbol comparisons for any sorting algorithm (not necessarily faithful) using the standard string comparison procedure and dealing with words of a Λ-tame source $\mathcal{S}$.

Theorem 6.5. *For a Λ-tame source $\mathcal{S}$, the following asymptotic lower bound $\underline{S}(n)$ holds for the mean number of symbol comparisons performed by any key-comparison based sorting algorithm and dealing with words emitted by $\mathcal{S}$,*

$$\underline{S}(n) \sim \frac{1}{2\log 2}\frac{1}{h(\mathcal{S})} n \log^2 n.$$

Remark. This lower bound shows that `QuickSort` is quasi-optimal in the model of symbol-comparisons, as it is quasi-optimal in the model of key-comparisons.
When the source is not Λ-tame, and its Dirichlet series $\Lambda(s)$ is tame at $s = 1$ with order $k_0 > 1$, then the order of the asymptotic lower bound becomes $\Theta(n \log^{1+k_0} n)$, and `QuickSort` remains quasi-optimal,

Proof. We use the same notations as in Section 6.1. We consider a set $\mathcal{U}$ of n distinct words, and a key-comparison based sorting algorithm $\mathcal{A}$. The set $\mathcal{U}$ is presented as input to algorithm $\mathcal{A}$ in order given by some permutation σ, and we denote by $\mathcal{K}[\mathcal{U}, \mathcal{U}_{\langle w\rangle}, \sigma]$ the number of comparisons performed by $\mathcal{A}$ on the subset $\mathcal{U}_{\langle w\rangle}$ when $\mathcal{U}$ is input under permutation σ.

We denote by L the function $L(n) = \log_2(n!)$, which appears in the lower bound for the mean number of key comparisons. We fix a subset $\mathcal{U}$ and an algorithm $\mathcal{A}$ and we say that a permutation σ is k-good for $\mathcal{V} \subset \mathcal{U}$ if $\mathcal{K}[\mathcal{U}, \mathcal{V}, \sigma] \geq L(|\mathcal{V}|) - k$. If it is not k-good, it is said to be k-bad. We will use the following lemma due to Seidel [24].

Lemma 6.6. [Seidel] *For any subset $\mathcal{U}$ of cardinality n, and any algorithm $\mathcal{A}$, there is a set $\mathfrak{S} \subset \mathfrak{S}_n$ for which the following holds:*
(i) The cardinality $|\mathfrak{S}|$ satisfies: $|\mathfrak{S}| \geq n![1 - (1/n)]$;
(ii) All the elements of $\mathfrak{S}$ are $(2\log n)$-good for any $\mathcal{U}_{\langle w\rangle}$.

We will prove the Lemma later on. We first explain how it entails the proof of Theorem 6.5. Indeed, with the second relation of Eq. (6.1), Lemma 6.6 entails the inequality, for any set $\mathcal{U}$ of cardinality n,

$$S[\mathcal{U}] \geq \left(1 - \frac{1}{n}\right)\left(R_1[\mathcal{U}] - 2\log n\, R_2[\mathcal{U}]\right),$$

where the parameters R_1 and R_2 are respectively associated with the toll functions $f(k) = L(k) := \log_2(k!)$ and $f(k) = 1$. Now, Proposition 6.3 provides the asymptotic behaviour for the mean values $R_1(n)$ and $R_2(n)$, namely

$$R_1(n) \sim \frac{1}{2\log 2}\frac{1}{h(\mathcal{S})} n \log^2 n, \qquad R_2(n) = \Theta(n).$$

This proves that the mean number of symbol comparisons of the algorithm $\mathcal{A}$ admits the asymptotic lower bound

$$S(n) \geq \underline{S}(n), \qquad \text{with} \quad \underline{S}(n) \sim \frac{1}{2\log 2}\frac{1}{h(\mathcal{S})} n \log^2 n,$$

and Theorem 6.5 is proven. □

Proof of Lemma 6.6. Consider the decision tree $\mathcal{D}$ associated with the algorithm $\mathcal{A}$. The set $\mathcal{U}_{\langle w\rangle}$ is an order contiguous subrange of $\mathcal{U}$, i.e., $\mathcal{U}_{\langle w\rangle} = \mathcal{U}_{[i,j]}$. Consider the set of permutations $\underline{\mathfrak{S}}$ whose restriction to the set $[1\,..\,n] \setminus [i\,..\,j]$ is fixed. Thus $|\underline{\mathfrak{S}}| = N_w!$. Each leaf of $\mathcal{D}$ corresponds to a permutation σ. Take the leaves that correspond to permutations in $\underline{\mathfrak{S}}$ along with their rootpaths. They induce a subtree of $\mathcal{D}$. We contract all the paths in this tree by removing all the non branching nodes; there results a binary tree $\underline{\mathcal{D}}$ that represents a valid decision tree for a sorting algorithm on $\mathcal{U}_{\langle w\rangle}$. Since $\underline{\mathcal{D}}$ is a binary tree with $N_w!$ leaves, for any $k > 0$, there can be at most a $1/2^k$- fraction of these leaves that have distance less than $L(N_w) - k$ from the root. In other words, at most a $1/2^k$-fraction of the permutations $\sigma \in \underline{\mathfrak{S}}$ are k-bad for $\mathcal{U}_{\langle w\rangle}$. Since this is true for any which way the permutation values outside $[i, j]$ were fixed, we get that for any $k > 0$, the fraction of all permutations that are k-bad for $\mathcal{U}_{\langle w\rangle}$ is at most 2^{-k}.

Now, we observe that, although the trie $T(\mathcal{U})$ can have arbitrarily many nodes w, there are only at most $n-1$ different sets $\mathcal{U}_{\langle w\rangle}$. There is a clear equality between the number of different sets $\mathcal{U}_{\langle w\rangle}$ and the number of branching nodes in the trie. We now prove that the number of branching nodes in a trie with n leaves is at most $n-1$, with an easy recursion. For $n = |\mathcal{U}| = 2$, there is at most 1 branching node. For a cardinality $n := |\mathcal{U}| \geq 2$, we consider the first branching node in the trie (the one with the smallest level). Then, each subtrie of cardinality $n_i < n$ has at most $n_i - 1$ branching nodes, and there are at most $1 + \sum_{i\in\Sigma}(n_i - 1) \leq n - 1$ branching nodes.

Thus choosing $k \geq 2\log n$ ensures that there is a subset $\mathfrak{S}$ of $\mathfrak{S}_n$ whose cardinality is at least $(1 - n2^{-k})n! \geq [1 - (1/n)]n!$ and whose elements are k-good for all the subsets $\mathcal{U}_{\langle w\rangle}$. □

6.7. Relation between various lower bounds.

The well-known lower bound $\underline{K}(n)$ for any sorting algorithm using key comparisons is asymptotic to $n\log_2 n$. Then, we have proven that

$$\frac{\underline{S}(n)}{\underline{K}(n)} \sim \frac{1}{2h(\mathcal{S})}\log n, \tag{6.15}$$

and this is the same ratio as the ratio which appears in (5.1) for the non-robust algorithms, namely `QuickSort` and `BubSort`.

There is also a lower bound in information theory which states that the number $D(n)$ of symbol comparisons used by any sorting algorithm on words which uses the symbol representation of words satisfies

$$D(n) \geq \underline{D}(n), \qquad \text{with} \quad \underline{D}(n) \sim \frac{1}{h(\mathcal{S})} n\log n,$$

and the following ratio between the two asymptotic lower bounds for sorting words holds

$$\frac{\underline{S}(n)}{\underline{D}(n)} \sim \frac{1}{2\log 2}\log n.$$

Conclusion.

We show here the applicability of the method which has been described in the paper [28]. We describe a new point of view on the basic algorithms, and their analysis, which can be (partially) automatized. Our dream is to revisit all standard algorithms from a student book, with this point of view, and perform their realistic analysis.

Acknowledgements. This paper greatly benefited from many discussions we had with Philippe Flajolet, on the topics of the Rice formula, on the cycle "Poisson–Mellin–Newton–Rice" and the tameness of sources. For these, we are truly grateful, and we dedicate this paper to his memory.

References

[1] E. Cesaratto and B. Vallée. Gaussian distribution of trie depth for dynamical sources. submitted, 2012.

[2] J. Clément, J. Fill, T. Nguyen Thi, and B. Vallée. Realistic analysis of the Quickselect algorithm. *submitted*, 2014.

[3] J. Clément, P. Flajolet, and B. Vallée. Dynamical sources in information theory: A general analysis of trie structures. *Algorithmica*, 29(1):307–369, 2001.

[4] J. Clément, T. Nguyen Thi, and B. Vallée. A general framework for the realistic analysis of sorting and searching algorithms. Application to some popular algorithms. In *STACS*, pages 598–609, 2013.

[5] R. De La Briandais. File searching using variable length keys. In *Papers Presented at the the March 3-5, 1959, Western Joint Computer Conference*, IRE-AIEE-ACM '59 (Western), pages 295–298, New York, NY, USA, 1959. ACM.

[6] D. Dolgopyat. On decay of correlations in Anosov flows. *Ann. of Math.*, 147(2):357–390, 1998.

[7] D. Dolgopyat. Prevalence of rapid mixing in hyperbolic flows. *Ergod. Th. & Dynam. Sys.*, 18:1097–1114, 1998.

[8] J. A. Fill. Distributional convergence for the number of symbol comparisons used by Quicksort. *Ann. Appl. Probab.*, 23:1129–1147., (2013).

[9] J. A. Fill and S. Janson. The number of bit comparisons used by Quicksort: an average-case analysis. In *Proceedings of the ACM-SIAM Symposium on Discrete Algorithms (SODA)*, pages 300–307, 2004. Long version *Electron. J. Probab.* 17, Article 43, 1-22 (2012).

[10] J. A. Fill and T. Nakama. Distributional convergence for the number of symbol comparisons used by QuickSelect. *Advances in Applied Probability*, 45:425–450, 2013.

[11] P. Flajolet. The ubiquitous digital tree. In *STACS 2006, 23rd Annual Symposium on Theoretical Aspects of Computer Science, Marseille, France, February 23-25, 2006, Proceedings*, volume 3884 of *Lecture Notes in Computer Science*, pages 1–22. Springer, 2006.

[12] P. Flajolet. A journey between Rice, Mellin and Poisson. *Personal communication*, 2008.

[13] P. Flajolet, X. Gourdon, and P. Dumas. Mellin transforms and asymptotics: Harmonic sums. *Theoretical Comput. Sci.*, 144(1–2):3–58, June 1995.

[14] P. Flajolet, M. Roux, and B. Vallée. Digital trees and memoryless sources: from arithmetics to analysis. *Proceedings of AofA'10, DMTCS, proc AM*, pages 231–258, 2010.

[15] P. Flajolet and R. Sedgewick. Mellin transforms and asymptotics: Finite differences and Rice's integrals. *Theor. Comput. Sci.*, 144(1&2):101–124, 1995.

[16] P. Flajolet and R. Sedgewick. *Analytic Combinatorics*. Cambridge University Press, 2009.

[17] E. Fredkin. Trie memory. *Commun. ACM*, 3(9):490–499, Sept. 1960.

[18] P. Jacquet and W. Szpankowski. Analytical de-Poissonization and its applications. *Theoretical Computer Science*, 201(1-2):1–62, 1998.
[19] P. Jacquet and W. Szpankowski. Entropy computations for discrete distributions: towards analytic information theory. *IEEE International Symposium on Information Theory*, 1998.
[20] N. E. Nörlund. Leçons sur les équations linéaires aux différences finies. In *Collection de monographies sur la théorie des fonctions*. Gauthier-Villars, Paris, 1929.
[21] N. E. Nörlund. *Vorlesungen über Differenzenrechnung*. Chelsea Publishing Company, New York, 1954.
[22] M. Roux and B. Vallée. Information theory: Sources, Dirichlet series, and realistic analyses of data structures. In *Proceedings 8th International Conference Words 2011*, volume 63 of *EPTCS*, pages 199–214, 2011.
[23] R. Sedgewick. *Algorithms in C, Parts 1–4*. Addison–Wesley, Reading, Mass., 1998. 3rd ed.
[24] R. Seidel. Data-specific analysis of string sorting. In *Proceedings of the Twenty-First Annual ACM-SIAM Symposium on Discrete Algorithms, SODA*, pages 1278–1286, 2010.
[25] W. Szpankowski. *Average case analysis of algorithms on sequences*. Interscience series in Discrete Mathematics and Optimization. Wiley, 2001.
[26] B. Vallée. Rice or Poisson-Mellin? *Preprint in preparation*.
[27] B. Vallée. Dynamical sources in information theory: Fundamental intervals and word prefixes. *Algorithmica*, 29(1/2):262–306, 2001.
[28] B. Vallée, J. Clément, J. A. Fill, and P. Flajolet. The number of symbol comparisons in QuickSort and QuickSelect. In S. A. *et al.*, editor, *Proceedings of ICALP 2009, Part I*, volume 5555 of *Lecture Notes in Computer Science*, pages 750–763. Springer-Verlag, 2009.