A general framework for the realistic analysis of sorting and searching algorithms. Application to some popular algorithms

Julien Clément, Thu Hien Nguyen Thi, Brigitte Vallée

A general framework for the realistic analysis of sorting and searching algorithms. Application to some popular algorithms*

Julien Clément, Thu Hien Nguyen Thi, and Brigitte Vallée

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

Abstract

We describe a general framework for realistic analysis of sorting and searching algorithms, and we apply it to the average-case analysis of five basic algorithms: three sorting algorithms (*QuickSort*, *InsertionSort*, *BubbleSort*) and two selection algorithms (*QuickMin* and *SelectionMin*). 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 $\Theta(n \log^2 n), \Theta(n^2), \Theta(n^2 \log n)$. For selection algorithms, and with respect to key comparisons, the average-case complexity of *QuickMin* is asymptotic to $2n$, of *SelectionMin* is $n - 1$. With respect to symbol comparisons, we prove that their average-case complexity remains $\Theta(n)$. In these five cases, we describe the dominant constants which exhibit the probabilistic behaviour of the source (namely, entropy, and various notions of coincidence) with respect to the algorithm.

1998 ACM Subject Classification F2.2: Pattern matching, sorting and searching – G2.1: Generating functions, permutations – G4: Algorithm design and analysis – H1.1: Information theory – I1.2: Analysis of algorithms

Keywords and phrases Probabilistic analysis of algorithms – Sorting and searching algorithms – Pattern matching – Permutations – Information theory – Rice formula – Asymptotic estimates

Digital Object Identifier 10.4230/LIPIcs.STACS.2013.598

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.

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

30th Symposium on Theoretical Aspects of Computer Science (STACS'13).
Editors: Natacha Portier and Thomas Wilke; pp. 598–609

Leibniz International Proceedings in Informatics
Schloss Dagstuhl – Leibniz-Zentrum für Informatik, Dagstuhl Publishing, Germany

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 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 [5], Fill and Nakama [6], who dealt with data composed of random uniform bits. Then, in the paper [18], 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 [4, 7].

Main results. The present paper follows the lines of the article [18], and works within the same general framework, with four specific aims:

(a) The general method has been already described in [18]: 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. 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, and a second phase where it is analyzed. We note here that the first phase may mostly be performed in an "automatic" way.

(b) We apply the method to three other popular algorithms: InsertionSort, BubbleSort and SelectionMinimum, respectively denoted in the sequel by the short names `InsSort`, `BubSort`, `SelMin` (see for instance the book [15] for a thorough description of these algorithms). With this approach we also easily recover the results about algorithms `QuickSort` and `QuickMin` already obtained in [18]. Thus we provide an unified framework for the analysis of these five algorithms in Section 2.2.

(c) We exhibit in each case the probabilistic features of the source which play a role in the analysis: each algorithm of interest is related to a particular constant of the source, which describes the interplay between the algorithm and the source, and explains how the efficiency of the algorithm depends on the source, via various notions of coincidence between words (See Proposition 5). This type of coincidence provides a good characterization of the algorithm, and our study is a tool for a better understanding of the algorithmic strategy.

(d) We discuss the robustness of the algorithms, i.e., the possible changes in the complexity behaviors, due to the change in the complexity measure, from the number of key comparisons to the number of symbol comparisons (see Discussion p. 607).

Plan of the paper. Section 1 first presents the general method, with its main steps. Then, Section 2 states the main results.

Most of the proofs and technical details are omitted due to space constraints. The full version of this paper will include them.

1 Main steps for the "realistic" analysis of a sorting algorithm

Here, we describe our general framework, already provided in [18]. 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 source, and the central notion of coincidence (Sections 1.2 and 1.3). We obtain 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) introduced in Section 1.4 and 1.5. 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 and 1.7).

1.1 The classical probabilistic model: permutations and arrival times

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 $\mathfrak{S}_n$ (usually endowed with the uniform distribution).

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 (or the arrival times) 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, $\pi(i, j)$ denotes the mean number of comparisons between U_i and U_j. The computation of $\pi(i, j)$ is the first step, described in Section 2.1. These mean numbers $\pi(i, j)$ are computed with direct probabilistic arguments. A remarkable feature is that the expectations $\pi(i, j)$ are always expressed as sums of rational functions depending on i, j or $j - i$.

1.2 General sources

Here, we consider that the keys are words produced by a general source. 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 *general source* produces infinite words of $\Sigma^{\mathbb{N}}$, and is specified by the set $\{p_w, w \in \Sigma^\star\}$ of *fundamental probabilities* p_w, where p_w is the probability that an infinite word begins with the finite prefix w. It is (only) assumed that $\sup\{p_w : w \in \Sigma^k\}$ tends to 0, as $k \to \infty$.

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

$$a_w := \sum_{\substack{\alpha, |\alpha| = |w|, \\ \alpha < w}} p_\alpha, \qquad b_w := \sum_{\substack{\alpha, |\alpha| = |w|, \\ \alpha \leq w}} p_\alpha, \qquad p_w = b_w - a_w. \tag{1}$$

Denote by $\mathcal{L}(\mathcal{S})$ the set of (infinite) words produced by the source $\mathcal{S}$, ordered via the lexicographic order. Given an infinite word $X \in \mathcal{L}(\mathcal{S})$, denote by w_k its prefix of length k.

The sequence (a_{w_k}) is increasing, the sequence (b_{w_k}) is decreasing, and $b_{w_k} - a_{w_k} = p_{w_k}$ tends to 0. Thus a unique real $P(X) \in [0,1]$ is defined as the common limit of (a_{w_k}) and (b_{w_k}), and $P(X)$ can be viewed as the probability that an infinite word Y be smaller than X. The mapping $P : \mathcal{L}(\mathcal{S}) \to [0,1]$ is strictly increasing outside the exceptional set formed with words of $\mathcal{L}(\mathcal{S})$ 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, to a number u of the interval $\mathcal{I} := [0,1]$, a word $M(u) \in \mathcal{L}(\mathcal{S})$. 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 $\mathcal{I}$. 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.

1.3 Coincidence

Here, we are interested in a more realistic cost related to the number of symbol comparisons performed by these 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* $\gamma(u,t)$ is the length of the largest common prefix of $M(u)$ and $M(t)$.

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

$$\mathcal{T} := \{(u,t) : 0 \le u \le t \le 1\}, \quad \mathcal{T}_w = (\mathcal{I}_w \times \mathcal{I}_w) \cap \mathcal{T} = \{(u,t) : a_w \le u \le t \le b_w\}. \tag{2}$$

Using the two relations

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

the following equality holds, for any integrable function g on the unit triangle $\mathcal{T}$, and will be extensively used in the sequel,

$$\int_{\mathcal{T}} [\gamma(u,t)+1] g(u,t)\, du\, dt = \sum_{w \in \Sigma^\star} \int_{\mathcal{T}_w} g(u,t)\, du\, dt. \tag{3}$$

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 parameters 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 interval $\mathcal{I}$ 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 $u < t$,

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

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 $M(u)$ and $M(t)$, 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

$$\widehat{\pi}(u,t) = \pi(N_{[0,u[} + 1, N_{[0,t[} + 2). \tag{4}$$

In our framework, expressions obtained for $\pi(i,j)$ ensure that $\widehat{\pi}(u,t)$ is always a sum of rational functions in variables $N_{[0,u[}$, $N_{[0,t[}$ and $N_{[u,t[}$, (with the relation $N_{[0,t[} = N_{[0,u[} + N_{]u,t[} + 1$).

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 sequence $\mathcal{V}$ has a variable number N of elements that obeys a Poisson law of rate Z,

$$\Pr\{N = k\} = e^{-Z}\frac{Z^k}{k!}. \tag{5}$$

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). 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

The density of the algorithm in the Poisson model, denoted by $\phi_Z(u,t)$ and defined as

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

is the mean number of key comparisons between two words $M(u')$ and $M(t')$ for $u' \in [u-du, u]$ and $t' \in [t, t+dt]$. In the model $(\mathcal{P}_Z, \mathcal{S})$, this is a main tool for computing, not only the mean number of key comparisons K_Z performed by the algorithm, but also the mean number of symbol comparisons S_Z via the formulae

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

To return to the Bernoulli model $(\mathcal{B}_n, \mathcal{S})$, the coefficients $\varphi(n,u,t)$ in the series expansion of $\phi_Z(u,t)$ defined as

$$\varphi(n,u,t) := (-1)^n\, n![Z^n]\phi_Z(u,t), \tag{6}$$

are computed in an "automatic way" from the mean numbers $\widehat{\pi}(u,t)$, themselves closely related to $\pi(i,j)$. This is the second step leading to results in Table 1 p. 608. Using Eq. (3), the sequence $\varphi(n)$ is now defined for any $n \geq 2$,

$$\varphi(n) := \int_{\mathcal{T}} (\gamma(u,t)+1)\,\varphi(n,u,t)\,du\,dt = \sum_{w\in\Sigma^\star}\int_{\mathcal{T}_w} \varphi(n,u,t)\,du\,dt, \tag{7}$$

and is easy to obtain via computations of the integral of $\varphi(n,u,t)$ on the triangles $\mathcal{T}_w$. Now, 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 $\varphi(n)$ by the equality

$$S(n) = \sum_{k=2}^{n} (-1)^k \binom{n}{k} \varphi(k), \tag{8}$$

which provides an exact formula for $S(n)$, described in Section 2.2. The expression of $S(n)$ is 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 $n \to \infty$). 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. (8). This analytic lifting gives rise to the mixed Dirichlet series itself,

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

which depends both on the algorithm (via $\varpi(s,u,t)$) and the source (via the fundamental triangles $\mathcal{T}_w$). 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. One has $\sigma_0 = 1$, except for the algorithms `InsSort` and `BubSort` where σ_0 equals 2. This is due to constant term $1/2$ appearing in the expectation $\pi(i,j)$, as seen in Table 1 p. 608 (see also Section 2.2).

The Rice Formula [12, 13] transforms a binomial sum into an integral in the complex plane. For any real $\sigma_1 \in]\sigma_0, \sigma_0+1[$, one has

$$T(n) = \sum_{k=1+\sigma_0}^{n} (-1)^k \binom{n}{k} \varpi(k) = \frac{(-1)^{n+1}}{2i\pi} \int_{\Re s=\sigma_1} G(s)\,ds, \text{ with } G(s) = \frac{n!\,\varpi(s)}{s(s-1)\dots(s-n)}. \tag{9}$$

Then, along general principles in analytic combinatorics [9, 10], 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

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

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. (8) and(9), 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 $\varphi(2)\binom{n}{2}$.

1.7 Tameness of sources

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 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 $A, B, \rho > 0$

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

where $\varpi(s)$ is meromorphic, with an only 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 $k \in \mathbb{Z}, k \neq 0$) of simple poles at points $s_k = \sigma_0 + 2ki\pi t$ with $t \neq 0$, and is of polynomial growth as $|\Im s| \to +\infty$[1].

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

Here, the main Dirichlet series $\varpi(s)$ of interest are closely related to the Dirichlet series of the source, which involve the fundamental probabilities p_w, and the ends a_w, b_w of the fundamental intervals (see Section 1.1), via a function $F : [0,1]^2 \to \mathbb{R}^+$ of class $\mathcal{C}^1$,

$$\Lambda[F](s) := \sum_{w \in \Sigma^\star} F(a_w, b_w)\, p_w^s, \qquad \Lambda_k[F](s) := \sum_{w \in \Sigma^k} F(a_w, b_w)\, p_w^s. \tag{10}$$

For $F \equiv 1$, we omit the reference to F, and we let $\Lambda := \Lambda[1]$. These series satisfy, for $\Re s > 1$, the relation[2] $|\Lambda(F, s)| \leq \|F\| \Lambda(\sigma)$. Since the equality $\Lambda_k(1) = 1$ holds for all k, the series $\Lambda(s)$ is divergent at $s = 1$, and many probabilistic properties of the source can be expressed in terms of the behavior of $\Lambda(s)$, when $\Re s$ is close to 1. For instance, the entropy $h(\mathcal{S})$ of the source $\mathcal{S}$ is defined as the limit (if it exists),

$$h(\mathcal{S}) := \lim_{k \to \infty} \frac{-1}{k} \sum_{w \in \Sigma^k} p_w \log p_w = \lim_{k \to \infty} \frac{-1}{k} \frac{d}{ds} \Lambda_k(s)_{|_{s=1}}. \tag{11}$$

Two types of properties of the source may entail tameness for the mixed series $\varpi(s)$.

▶ **Definition 4** (Tameness of Sources). (a) A source is *weakly tame* if the function $s \mapsto \Lambda(s)$ is analytic on $\Re s > 1$, and of polynomial growth when $\Im s \to \infty$ on any $\Re s \geq \sigma_1 > 1$

(b) 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)[3] and if one of the following conditions is fulfilled:

1. [S–shape] for any $F \in \mathcal{F}$, the series $\Lambda[F](s)$ is tame at $s = 1$ with a S–shape;

---

[1] 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 $\delta' < \delta$.

[2] The norm $\|\cdot\|$ is the sup-norm on $[0,1] \times [0,1]$.

[3] Then (proof omitted here) 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

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

2. [H–shape] for any $F \in \mathcal{F}$, the series $\Lambda[F](s)$ is tame at $s = 1$ with a H–shape;
3. [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.

This definition is in fact very natural, since 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 [8]. Dynamical sources, introduced by Vallée and defined in [17], may have a P–shape only if they are "similar" to simple sources. Adapting deep results of Dolgopyat [2, 3], 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 [14]. See the cited papers for more details, where all these facts, here described in a informal way, are stated in a formal way and proven.

This definition is also well-adapted to our framework since it describes situations where the mixed series $\varpi(s)$ may be proven tame. Then, the contour of the Rice integral may be shifted to the left, providing an asymptotic expansion for the mean number $S(n)$.

The weak tameness of the source is sufficient to entail the tameness at $s = 1$ (with a S–shape, and an exponent $k_0 = 0$) of series $\varpi(s)$ related to selection algorithms (namely `QuickMin` and `SelMin`). The Λ–tameness of the source is central in the analysis of sorting algorithms, as it ensures the tameness of $\varpi(s)$ related to algorithms `QuickSort`, `InsSort` and `BubSort`); moreover, the tameness shape $\varpi(s)$ is inherited from the one of the source.

2 Summary of our results.

We recall the main steps of the method.
Step 1. Computation of expected values $\pi(i, j)$.
Step 2. Automatic derivation of $\varpi(s, u, t)$; determination of the abscissa σ_0.
Step 3. Expression for the mixed Dirichlet series $\varpi(s)$, and description of the main term of the singular expression of $\varpi(s)/(s - \sigma_0)$. Interpretation of the "dominant" constants.
Step 4. Relation between tameness of the source and tameness of the mixed series $\varpi(s)$. Application of the Rice Formula. Statement of the final results.

This Section presents the results with three tables (found at the end), five propositions and a theorem. Section 2.1 summarizes Steps 1 and 2 with Propositions 2.1 and 2.2, and Table 1. Section 2.2 summarizes Step 3 with Propositions 2.3, 2.4, 2.5, and Table 2. Finally, Section 2.3 states the final result (Theorem 2.6) with Table 3. The proofs are omitted in this short version and defered to a full version of this paper.

2.1 Summary of the results for Steps 1 and 2

We present in the leftmost part of Table 1 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 5. Consider the permutation model described in Section 1.1, and denote by $\pi(i, j)$ the mean number of comparisons between the keys of rank i and j, with $i \leq j$. Then, for any of the five algorithms, the mean numbers $\pi(i, j)$ admit the expressions described in the second column of Table 1 p. 608.

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

▶ Proposition 6. Denote by $\varpi(s,u,t)$ the function which provides an analytical lifting of the sequence $\varphi(n,u,t)$ defined in Eq. (6), and by σ_0 the integer which defines the domain $\Re s > \sigma_0$ of validity of this lifting. Then, for any of the five algorithms, the functions $\varpi(s,u,t)$ admit the expressions described in the fifth column of Table 1 p. 608.

2.2 Summary of the results for Step 3 – the mixed Dirichlet series

▶ Proposition 7. Consider any general source, assumed to be weakly tame, together with the fundamental intervals $[a_w, b_w]$ defined in (1) and its Dirichlet series defined in Eq. (10). Then, for any of the five 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 2, 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

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

We now study the relation between tameness of the source and tameness of the mixed Dirichlet series.

▶ Proposition 8. Assume the source $\mathcal{S}$ to be weakly tame. Then, the mixed Dirichlet series $\varpi(s)$ relative to selection algorithms are both tame at $\sigma_0 = 1$ with order $k_0 = 0$ and a S–shape. Moreover, their abscissae δ satisfy

(*a*) `[QuickMin]` $\delta \geq 1/3$.

(*b*) `[SelMin]` $\delta > 0$ depends on an exponent a (attached to the source).

Assume the source $\mathcal{S}$ to be Λ–tame. Then, the mixed Dirichlet series $\varpi(s)$ relative to 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 $\varpi(s)$.

We finally describe the main term of the singular expression of $\varpi(s)/(s-\sigma_0)$ at $s = \sigma_0$.

▶ Proposition 9. The constants of interest which intervene in the main terms displayed in the last column of Table 2 p. 608 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.

(*iii*) The min–coincidence $a(\mathcal{S})$: this is the mean number of symbols needed to compare a uniform random word and the smallest word of the source.

(*iv*) The logarithmic coincidence $b(\mathcal{S})$: this is the mean number of symbols needed to compare two words X and Y randomly chosen as follows: the word X is uniformly drawn from the source, and Y is drawn with $Y \geq X$, according to density $1/t$.

The entropy is defined in (11). The constants $a(\mathcal{S}), b(\mathcal{S}), c(\mathcal{S})$ satisfy the inequalities $a(\mathcal{S}) < b(\mathcal{S}), c(\mathcal{S}) < 2b(\mathcal{S})$ and are defined as follows

$$a(\mathcal{S}) = \sum_{\ell \geq 0} q_\ell, \quad b(\mathcal{S}) = \sum_{w \in \Sigma^\star} \int_{\mathcal{T}_w} \frac{1}{t}\, du\, dt \quad c(\mathcal{S}) = 2 \sum_{w \in \Sigma^\star} \int_{\mathcal{T}_w} du\, dt = \sum_{w \in \Sigma^\star} p_w^2 = \Lambda(2).$$

Here q_ℓ is the probability of the prefix of length ℓ of the smallest word of the source, $\mathcal{T}_w$ is the fundamental triangle defined in (2) and $\Lambda(s)$ is defined in (10).

The constants $a(\mathcal{S}), c(\mathcal{S})$ and $h(\mathcal{S})$ are easy to compute for any memoryless source. For the unbiased source $\mathcal{M}_r$, or for the source $\mathcal{B}_p$ on the alphabet $\{0,1\}$, with $p := p_0$, one has: $a(\mathcal{M}_r) = c(\mathcal{M}_r) = \frac{r}{r-1}$, $h(\mathcal{M}_r) = \log r$, $a(\mathcal{B}_p) = \frac{1}{1-p}$, $c(\mathcal{B}_p) = \frac{1}{2p(1-p)}$ and $h(\mathcal{B}_p) = -p\log p - (1-p)\log(1-p)$. The constant $b(\mathcal{S})$ is more difficult to compute even in the memoryless case. But, for the source $\mathcal{M}_r$, one has (see [11] for details)

$$b(\mathcal{M}_r) = \sum_{\ell\geq 0}\Big(1 + \frac{1}{r^\ell}\sum_{k=1}^{r^\ell-1}\log\frac{k}{r^\ell}\Big), \qquad b(\mathcal{M}_2) \doteq 2.639689120.$$

2.3 Final step

▶ **Theorem 10.** *Consider a general source $\mathcal{S}$. For selection algorithms* `QuickMin`, `SelMin`, *we assume the source to be weakly–tame, and, for sorting algorithms* `QuickSort`, `InsSort`, `BubSort`, *we assume the source to be Λ–tame. Then, the mean number $S(n)$ of symbol comparisons performed by each algorithm on a sequence of n words independently drawn from the same source $\mathcal{S}$ admits the asymptotic behaviour described in Table 3. Here, the constants κ_i in the subdominant terms*[4] *involve the Euler constant γ together with the subdominant constant of the source*[5] *$d(\mathcal{S})$:*

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

The errors terms $E(n), F(n)$ are not of the same type for sorting algorithms and selection algorithms.

For selection algorithms, still assuming the source is weakly tame. The error term $F(n)$ is of order $O(n^{1-\delta})$, with $\delta = 1/3$ for `QuickMin`. *For* `SelMin`, *the constant δ depends on the exponent a (if it exists) attached to the source.*

For sorting algorithms, assuming a Λ–tame source with a given shape, we have

- *if the source has a S–shape with abscissa 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 abscissa 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).*

Discussion. We now compare the asymptotic estimates for the two mean numbers, the mean number $K(n)$ of key–comparisons (column 2 of Table 3) and the mean number $S(n)$ (column 3 of Table 3). 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 three algorithms: `InsSort`, `QuickMin` and `SelMin`. Of course, the constants are different for $K(n)$ and $S(n)$, and the ratios $S(n)/K(n)$ involve coincidences of various types always between *two* words, respectively uniform coincidence $c(\mathcal{S})$, logarithmic-coincidence $b(\mathcal{S})$, or min-coincidence $a(\mathcal{S})$.

(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 $S(n)/K(n)$ is asymptotic to $[1/(2h(\mathcal{S}))]\log n$.

(c) This ratio also appears in lower bounds: Combining results due to Seidel [16], with our methods, we obtain a lower bound for the number of symbol comparisons of any sorting

[4] The constant κ_2 is not computed here. Note that the computation of the subdominant term for `InsSort` needs the singular expansion of $\varpi(s)/(s-1)$ at $s = 1$.

[5] This constant, defined as the constant term in the singular expansion of $\Lambda(s)$ at $s = 1$, is easy to compute for any source $\mathcal{B}_p$: $d(\mathcal{B}_p) = (1/h(\mathcal{B}_p))^2(p\log^2 p + (1-p)\log^2(1-p))$.

algorithm on a source $\mathcal{S}$, asymptotic to $S_0(n) = [1/(2h(\mathcal{S}))]\, n \log^2 n$. Comparing with the well-known lower bound for the number of key comparisons, asymptotic to $K_0(n) = n \log n$, we observe that the ratio $S_0(n)/K_0(n)$ is also asymptotic to $[1/(2h(\mathcal{S}))]\, \log n$.

Algorithms	$\pi(i,j)$	$K(n)$	σ_0	$\varpi(s,u,t),\ \Re s > \sigma_0$
`QuickSort`	$\dfrac{2}{j-i+1}$	$2n\log n$	1	$2(t-u)^{s-2}$
`InsSort`	$\dfrac{1}{2} + \dfrac{1}{(j-i+1)(j-i)}$	$\dfrac{n^2}{4}$	2	$(s-1)(t-u)^{s-2}$
`BubSort`	$\dfrac{1}{2} + \dfrac{1}{(j-i+1)(j-i)} + \dfrac{2(i-1)}{(j-i+2)(j-i+1)(j-i)}$	$\dfrac{n^2}{2}$	2	$(s-1)(t-u)^{s-3}[t-(s-1)u]$
`QuickMin`	$\dfrac{2}{j}$	$2n$	1	$2t^{s-2}$
`SelMin`	$\dfrac{1}{i(i+1)} + \dfrac{1}{j(j-1)}$	n	1	$(s-1)[u^{s-2} + t^{s-2}]$

(a) Table 1: results for Steps 1 and 2 (Section 2.1)

Algorithms	$\varpi(s)$	σ_0	Main term of $\varpi(s)/(s-\sigma_0)$
`QuickSort`	$\dfrac{2\Lambda(s)}{s(s-1)}$	1	$\dfrac{2}{h(\mathcal{S})}\dfrac{1}{(s-1)^3}$
`InsSort`	$\dfrac{\Lambda(s)}{s}$	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	$-\dfrac{1}{2h(\mathcal{S})}\dfrac{1}{(s-2)^2}$
`QuickMin`	$2\displaystyle\sum_{w\in\Sigma^\star}\int_{a_w}^{b_w}(t-a_w)t^{s-2}dt$	1	$2b(\mathcal{S})\dfrac{1}{s-1}$
`SelMin`	$(s-1)\displaystyle\sum_{w\in\Sigma^\star}(b_w-a_w)\int_{a_w}^{b_w} u^{s-2}du$	1	$a(\mathcal{S})\dfrac{1}{s-1}$

(b) Table 2: results for Step 3 (Section 2.2)

Algorithms	$K(n)$	Dom. term of $S(n)$	Subdominant terms	Rem. term
`QuickSort`	$2n\log n$	$\dfrac{1}{h(S)}\, n\log^2 n$	$\kappa_0 n\log n\ +\ \kappa_2 n$	$E(n)$
`InsSort`	$\dfrac{n^2}{4}$	$\dfrac{c(\mathcal{S})}{4}\, n^2$	$\dfrac{1}{h(S)} n\log n\ +\ \left(\kappa_0 - \dfrac{c(\mathcal{S})}{4}\right) n$	$E(n)$
`BubSort`	$\dfrac{n^2}{2}$	$\dfrac{1}{4h(S)}\, n^2\log n$	$\left(\kappa_1 + \dfrac{c(\mathcal{S})}{4}\right) n^2$	$nE(n)$
`QuickMin`	$2n$	$2b(\mathcal{S})\, n$		$F(n)$
`SelMin`	n	$a(\mathcal{S})\, n$		$F(n)$

(c) Table 3: results for Theorem 1 (Section 2.3). *Nota.* Rem.: Remainder, Dom.: Dominant.

Figure 1 Tables summarizing results.

Conclusion. We show here the applicability of the method which has been described in the paper [18]. 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 and the tameness of sources. For these, we are truly grateful.

References

1. E. Cesaratto and B. Vallée. Gaussian distribution of trie depth for dynamical sources. submitted, 2012.
2. D. Dolgopyat. On decay of correlations in Anosov flows. *Ann. of Math.*, 147(2):357–390, 1998.
3. D. Dolgopyat. Prevalence of rapid mixing in hyperbolic flows. *Ergod. Th. & Dynam. Sys.*, 18:1097–1114, 1998.
4. J. A. Fill. Distributional convergence for the number of symbol comparisons used by Quicksort. Ann. Appl. Probab. (2012), to appear.
5. 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).
6. J. A. Fill and T. Nakama. Analysis of the expected number of bit comparisons required by quickselect. *Algorithmica*, 58(3):730–769, 2010.
7. J. A. Fill and T. Nakama. Distributional convergence for the number of symbol comparisons used by Quickselect. *CoRR*, abs/1202.2599, 2012. submitted.
8. 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.
9. P. Flajolet and R. Sedgewick. Mellin transforms and asymptotics: Finite differences and Rice's integrals. *Theor. Comput. Sci.*, 144(1&2):101–124, 1995.
10. P. Flajolet and R. Sedgewick. *Analytic Combinatorics*. Cambridge University Press, 2009.
11. P. J. Grabner and H. Prodinger. On a constant arising in the analysis of bit comparisons in Quickselect. *Quaestiones Mathematicae*, 31(4):303–306, 2008.
12. N. E. Nörlund. Leçons sur les équations linéaires aux différences finies. In *Collection de monographies sur la théorie des fonctions*. Gauthier-Villars, Paris, 1929.
13. N. E. Nörlund. *Vorlesungen über Differenzenrechnung*. Chelsea Publishing Company, New York, 1954.
14. 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.
15. R. Sedgewick. *Algorithms in C, Parts 1–4*. Addison–Wesley, Reading, Mass., 1998. 3rd ed.
16. 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.
17. B. Vallée. Dynamical sources in information theory: Fundamental intervals and word prefixes. *Algorithmica*, 29(1/2):262–306, 2001.
18. 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.