Informatique théorique et Applications/Theoretical Informatics and Applications
(vol. 23, n° 3, 1989, p. 317 à 333)

COMBINATORIAL ANALYSIS OF QUICKSORT ALGORITHM (*)

by P. HENNEQUIN [1]

Communicated by P. FLAJOLET

Abstract. – *We study probability distributions of several characteristic parameters on various forms of Quicksort algorithm: median-of-k, cutting of small lists. A constant use of generating functions leads to a more synthetic description and analysis of the combinatorial structure of the algorithm. This approach allows us in particular to extend the known results for the distributions of running times. We obtain average values, but also higher moments of the distribution of the cost function, both exactly and asymptotically. We give for all k means and variances of median-of-k algorithm with insertion sort on small files, and we compute higher moments in the standard case.*

Résumé. – *On étudie les distributions de probabilités des paramètres caractéristiques de différentes variantes de l'algorithme Quicksort: médiane de k éléments, tri différent sur les petites listes. L'utilisation de séries génératrices fournit une description synthétique de la structure combinatoire de l'algorithme et permet d'étendre les résultats connus sur les distributions de probabilités des coûts d'exécution. On obtient ainsi les valeurs moyennes et les variances des coûts pour l'algorithme Quicksort avec médiane d'un nombre quelconque d'éléments et tri par insertions sur les petites listes. La distribution des coûts est caractérisée plus précisement par la détermination des moments d'ordre plus élevé dans le cas standard.*

1. INTRODUCTION

Sorting algorithms have been extensively studied by D. E. Knuth [7] who gives detailed analyses of various combinatorial parameters that determine e. g. their average performance. R. Sedgewick [9] studied more specifically a number of variations based on the Quicksort scheme. In this note we extend some results of R. Sedgewick concerning average behaviours and higher moments of the distributions of the running times of Quicksort algorithm: we prove in particular that if limiting distributions exist they cannot be

(*) Received May 1987.
[1] École Polytechnique, C.M.A.P.-L.I.X., 91128 Palaiseau, France.
[2] This work has been partly supported the P.R.C. Mathématiques-Informatique.

normal. More quantitative information can also be derived from our analysis concerning the tuning of the selection for a partitioning element.

These results are derived by an algebraic method which follows the general gramework proposed by P. Flajolet [3] and J. M. Steyaert [10] for analyzing the average performance of algorithms. We make an extensive use of this *symbolic operator method* (instead of classical recurrence relations) which allows the more general and easier derivations needed in our study.

2. COMBINATORIAL DESCRIPTION

We assume in our analysis that keys to be sorted are *all distinct*. The general scheme for Quicksort is then described by:

```
function $Q$sort ($L$: list): list;
  if size $(L) \leq 1$ then $Q$sort$:= L$
                 else Select an element $v$ in $L$;
                      Partition $L$ into $L1 := \{x \in L \mid x < v\}$ and $L2 := \{x \in L \mid x > v\}$
                      $Q$sort$:= Q$sort$(L1).v.Q$sort$(L2)$.
```

The sorting method depends only on the relative order of keys and not on their specific values, so that we can assimilate the set of lists of size n to the set $\mathbf{S}_n$ of permutations over $[1 \ldots n]$: we associate to each key its position in the sorted list. We then denote by σ, σ_1, σ_2 the permutations associated to L, $L1$, $L2$ and by σ_0 the permutation obtained from σ by deleting the partitioning key. Furthermore, in order to simplify the notation, we will identify a pair $(\sigma_1, \sigma_2) \in \mathbf{S}_p \times \mathbf{S}_q$ with the pair (σ_1, σ_2') obtained by adding $p+1$ to the labels of σ_2: $(132,21) \Leftrightarrow (132,65)$.

2.1. Hypotheses

The following hypotheses are commonly assumed in average case analyses of Quicksort:

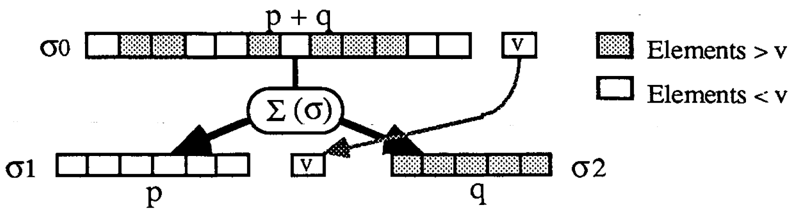
$H\,1$ (*Hypothesis on data distribution*): *Lists of size n are permutations in S_n with the uniform distribution* (*probability of each permutation* $= 1/n!$).

$H\,2$ (*Hypothesis on the algorithm*): *The partitioning method preserves* equally likely *distributions on produced permutations. In other words, each pair $(\sigma_1, \sigma_2) \in S_p \times S_q$ comes from the partition around the value $\nu = p+1$ of the same number of permutations σ_0 in S_{p+q}.*

H 1 is satisfied, in particular, when keys are drawn *independently* from a continuous distribution and gives a (classical) probabilistic model in order to define the *average* behaviour of the algorithm. Condition H2 is related to the partitioning algorithm; it is crucial for most analyses since it allows the translation of Quicksort's recursive structure into recurrences on costs. In fact it is a consequence of the more restrictive (but very reasonable in practice) hypothesis H 2′ which allows a better understanding of the partitioning mechanism and which is easier to verify for a given partitioning algorithm.

H 2′: The partitioning algorithm uses only comparisons with the partitioning key v.

Schema of partitioning algorithm



H 2′ ⇒ H 2: Let p and q be fixed ($v=p+1$). In a general way, we can characterize a partitioning method be giving for each on the permutation $\Sigma(\sigma_0)$ which takes σ_0 to $\sigma_1 . \sigma_2$. If we call mark $M(\sigma_0)$ the set of positions in σ_0 of the p smallest keys, condition H 2′ implies that $\Sigma(\sigma_0)$ depends only on $M(\sigma_0)$ since the algorithm cannot discern keys $<v$ (resp. $>v$). Thus the $p!q!$ permutations of σ_0 which have a same mark [and so a same $\Sigma(\sigma_0)$], produce by partition once and only once each pairs (σ_1, σ_2) of permutations of the p smallest and q greatest keys of σ_0; and so H 2. Finally, going from a given pair (σ_1, σ_2) to one of the $(p+q)!/p!q!$ antecedent σ_0 consists in a special kind of *shuffle* (*Qshuffle*) of σ_1 and σ_2: for each shuffle, i. e. for each mark $M(\sigma_0)$, we first execute on σ_1 two permutations fixed by $\Sigma(\sigma_0)$.

Remark: In this combinatorial description, we also suppose that the partitioning algorithm is deterministic ($\Sigma(\sigma)$ and v are well defined for each list) and particulary that keys to be selected are taken in fixed positions. However, random executions which respect hypotheses H 1 and H 2 (for example selection of keys in random places) lead exactly to the same probabilistic behaviour since they correspond at each step to the same distribution of lists and keys.

2.2. Formal description. Partitional product

A *class of combinatorial structures* is a denumerable set together with a size function denoted $|.|$; we consider here the class of permutations $\mathbf{S} = \bigcup \mathbf{S}_n$. For a cost function C defined on permutations (for example, the number of comparisons for sorting using Quicksort), we introduce the weighted class $\mathbf{C}(\mathbf{S})$ consisting formally of elements $C(\sigma).\sigma$ (with size $= |\sigma|$) for σ all in $\mathbf{S}$. Then we associate to $\mathbf{S}$ and $\mathbf{C}(\mathbf{S})$ the exponential generating functions defined by:

$$S(z) = \sum_{\sigma \in \mathbf{S}} \frac{z^{|\sigma|}}{|\sigma|!} = \sum_n n! \frac{z^n}{n!} = \frac{1}{1-z}$$

and

$$C(z) = \sum_{\sigma \in \mathbf{S}} C(\sigma) \frac{z^{|\sigma|}}{|\sigma|!} = \sum_n C_n \frac{z^n}{n!};$$

hence C_n is the total cost over $\mathbf{S}_n$ and, from H 1, $C_n/n!$ is precisely the average cost.

If $\sigma \in \mathbf{S}$ partitions into σ_1, σ_2 we can write $C(\sigma) = C_0(\sigma) + C(\sigma_1) + C(\sigma_2)$ with $C_0(\sigma) =$ cost of one sorting stage. We are going now to describe this recursive definition of $\mathbf{S}$ by using the partitional product (*see* Foata [5]) which is a kind of cartesian product for labelled objects:

DEFINITION: — *A* bipartition of $[1 \ldots n]$ *is a pair* (I_1, I_2) *such that* $I_1 \cup I_2 = [1 \ldots n]$ *and* $I \cap I_2 = \varnothing$.

— *The* partitional product *of two structures* s_1, s_2 *of size* p *and* q *is the set* $s_1 \otimes s_2$ *of pairs* $[(s_1, I_1), (s_2, I_2)]$ *where* (I_1, I_2) *is a bipartition of* $[1 \ldots p+q]$ *such that* $|I_1| = p$ *and* $|I_2| = q$. *The size of elements in* $s_1 \otimes s_2$ *is then* $p+q$ *and* $\operatorname{card}(s_1 \otimes s_2) = (p+q)!/p!\,q!$.

— *The partitional product of two classes, denoted by* $\otimes$, *is the union of partitional products of all pairs of the cartesian product.*

— *We call* rooting *of a class* $\mathbf{C}$, *denoted by* $\mathbf{o}(\mathbf{C})$, *the class consisting of elements of* $\mathbf{C}$ *with a size incremented by* 1 $(|x|_{\mathbf{o}(\mathbf{C})} = |x|_{\mathbf{C}} + 1)$.

We divide the characteristic decomposition $\sigma \to (\sigma_1, \sigma_2)$ into two steps: selection $\sigma \to (\sigma_0, p)$ where σ_0 is obtained from σ by deleting in a fixed place the key $v = p+1$, and partition around v, $(\sigma_0, p) \to (\sigma_1, \sigma_2)$. Consider the class

$\mathbf{U}=\{(\sigma,j)/\sigma\in\mathbf{S}, j\in[0\ldots|\sigma|]\}$ with $|(\sigma,j)|_{\mathbf{U}}=|\sigma|$, we can state now:

PROPOSITION 1: *The mapping* $\sigma \dashrightarrow (\sigma_0,p)$ *defines an isomorphism between* $\mathbf{S}-\mathbf{S}_0$ *and* $\mathbf{o}(\mathbf{U})$.

PROPOSITION 2: *The mapping* $(\sigma_0,p)\dashrightarrow(\sigma_1,\sigma_2)$ *defines on isomorphism between* $\mathbf{U}$ *and* $\mathbf{S}\otimes\mathbf{S}$.

Remark: Two classes are isomorphic if their sets are isomorphic and if their size functions are equivalent; their generating functions are then equal.

Proofs: (1) Consider $(\sigma_0,p)\in\mathbf{o}(\mathbf{U})$, the only antecedent σ in $\mathbf{S}-\mathbf{S}_0$ is obtained from σ_0 by incrementing labels $>p$ and inserting the key $v=p+1$ in fixed place. The correspondence of size functions is given then by the rooting operator and so the isomorphism.

(2) We have a plain correspondence between bipartitions of $[1\ldots n]$ and marks of lists of size n. So under hypothesis H 2′, we can associate one to one the elements of partitional product $\sigma_1\otimes\sigma_2$ with the antecedents $\sigma_0(|\sigma_0|=|\sigma_1|+|\sigma_2|)$ of (σ_1,σ_2): we execute a *Qshuffle* according to each bipartition; p is then fixed by $p=|\sigma_1|$. In the general case (condition H 2), the isomorphism is not natural but comes from a counting argument. Indeed, the number of antecedents to a pair (σ_1, σ_2) of permutations of size p and q is exactly the cardinal of $\sigma_1\otimes\sigma_2:(p+q)!/p!\,q!$.

As a corollary, we can now identify $\mathbf{S}$ with $\mathbf{o}(\mathbf{S}\otimes\mathbf{S})+\mathbf{S}_0$ acording to the partitioning decomposition and expand the cost function C on this description of S, we so obtain:

$$\mathbf{S}=\mathbf{o}(\mathbf{S}\otimes\mathbf{S})+\mathbf{S}_0 \qquad \text{and} \qquad \mathbf{C}(\mathbf{S})=\mathbf{C}_0(\mathbf{S})+\mathbf{o}(\mathbf{C}(\mathbf{S})\otimes\mathbf{S})+\mathbf{o}(\mathbf{S}\otimes\mathbf{C}(\mathbf{S}))$$

The partitional product of two classes corresponds to the product of the exponential generating series (*see* [3]); in the same way the rooting operator corresponds to taking antiderivatives on these series. So, we can translate the previous formal descriptions into the equations:

$$C(z)=C_0(z)+2\int_0^z C(x)\,S(x)\,dx \qquad \text{and} \qquad S(z)=\int_0^z S^2(x)\,dx+1$$

From which as expected, we get $S(z)=(1-z)^{-1}$.

2.3. Variants

We consider here two generalizations of the Quicksort algorithm which lead to the same kind of solution. The first one consists in using for small lists (size $\leqq M$) a sorting procedure (insertion sort for example) more efficient than Quicksort on small files. To describe this *cutting of small lists* variant we introduce a cutting operator on classes defined by:

$$\mathbf{T_M}(\mathbf{C})=\{c\in\mathbf{C}/|c|>M\}=\mathbf{C}-\mathbf{P_M}(\mathbf{C})$$

with

$$\mathbf{P_M}(\mathbf{C})=\{c\in\mathbf{C}/|c|\leqq M\}$$

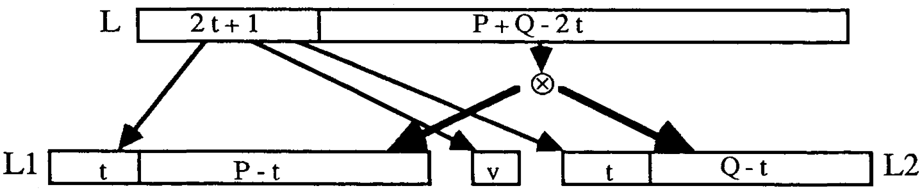
If a class $\mathbf{C}$ is described in a general way by $\mathbf{C}=\text{constr}(\mathbf{C},\ldots)$, changing the definition of objects of size $\leqq M$ leads to the new description: $\mathbf{C}=\mathbf{T_M}[\text{constr}(\mathbf{C},\ldots)]+\mathbf{C}_1$ where $\mathbf{C}_1$ is the class consisting of new elements of size $\leqq M$. The exponential generating series $C_1(z)$ is then a polynomial of degree M (operator $\mathbf{T_M}$ corresponds trivially to a cutting operation T_M on series).

The second variant consists in selecting the partitioning key as median element of $k=2t+1$ keys. We thus obtain a more centered distribution for partitioning key. Indeed, the probability that a list of size $n=p+q+1$ partitions itself into lists of size p and q is:

$$P_{p,q}=\binom{p}{t}\binom{q}{t}\Big/\binom{n}{2t+1}$$

We suppose here that during the determination of the partitioning key v, we also partition (equally likely) the list L_k of the k concerned elements into two sublists $L1_t$ and $L2_t$. To simplify explanations, we suppose also that these lists consist of first elements of lists L, $L1$ and $L2$. We write then: $L=L_k.L'$, $L1=L1_t.L1'$, $L2=L2_t.L2'$. ($|L|\geqq k$)

Schema of median-of-2t+1 partition



As previously, the lists L_k are generated from $L1_t$ and $L2_t$ by the construction $\mathbf{o}(L1_t\otimes L2_t)$. On other hand, the rank of the key v is equally likely distributed in L' (but not in L) so $L1' \otimes L2'$ describes exactly the set of antecedent lists L'. The behaviours of lists L', $L1'$, $L2'$ and of lists L_k, $L1_t$, $L2_t$ are then independent except for the contribution of the key v. So for fixed $L1'$ and $L2'$, the $(t!)^2$ pairs $(L1, L2)$ obtained by considering all permutations of $L1_t$ and $L2_t$ have exactly as antecedents the set of lists obtained by concatenation of the $k!$ permutations of L_k with elements of $L1' \otimes L2'$.

Definition: *We call* i-rooting *the operator* $\mathbf{o}_i$ *defined by* $\mathbf{o}_i(\mathbf{C}) = \mathbf{S}_i \times \mathbf{C}$ (*with cartesian product size*). *If* $\mathbf{C}$ *is formed of lists, we replace the cartesian product by concatenation and we denote by* $\mathbf{D}_i$ *the inverse operator which consists in deleting the first* i *elements of a list.*

This allows us now to describe the permutations of size $\geqq k$ with respect to their partitioning decomposition:

Proposition 3: *The decomposition of cost for median-of-k Quicksort is described by:*

$$\mathbf{T}_k(\mathbf{S}) = \mathbf{o}_k(\mathbf{D}_t(\mathbf{S}) \otimes \mathbf{D}_t(\mathbf{S}))$$

$$\mathbf{C}(\mathbf{S}) = \mathbf{C}_0(\mathbf{S}) + \mathbf{o}_k(\mathbf{D}_t(\mathbf{C}(\mathbf{S})) \otimes \mathbf{D}_t(\mathbf{S})) + \mathbf{o}_k(\mathbf{D}_t(\mathbf{S}) \otimes \mathbf{D}_t(\mathbf{C}(\mathbf{S})))$$

We consider now the Quicksort algorithm with the two previous variants combined (with $M \geqq k$). We remark that $\mathbf{C} = \mathbf{o}_i(\mathbf{A}) \Rightarrow C^{(i)}(z) = i!\, A(z)$ where ${}^{(i)}$ denotes the i-th derivative, so the cost C is described by the following equation over generating series:

$$C(z) = T_M\Big[C_0(z) + 2\frac{k!}{t!\,t!} \int \underset{k \text{ times}}{\cdots} \int C^{(t)}(x)\, S^{(t)}(x)\, dx^k\Big] + C_1(z)$$

This equation appears also in the analysis of a variant of binary tree search in D. H. Greene [6].

After simple manipulations, we obtain the characteristic differential equation of degree k:

$$\text{(I)} \quad \frac{(1-z)^k C^{(k)}(z)}{k!} = \frac{(1-z)^k C_0^{(k)}(z)}{k!} + 2\frac{(1-z)^t C^{(t)}(z)}{t!} + (1-z)^k Q_{M-k}(z)$$

where $Q_{M-k}(z)$ is a polynomial of degree $M-k$.

3. RESOLUTION

3.1. Method

Let $X=1-z$ and $\tilde{C}(X)=C(1-X)=C(z)$, $\tilde{C}_0(X)=C_0(1-X)=C_0(z)$ $(\ldots)$. Consider now the differential oprator Θ defined by $\Theta.F(X)=XF'(X)$, we have easily by induction on n:

$$\binom{\Theta}{n}.F(X)=\Theta.(\Theta-1)\ldots(\Theta-n+1).\frac{F(X)}{n!}=\frac{X^n F^{(n)}(X)}{n!}$$

So, we can rewrite the characteristic equation as:

$$(\mathrm{I}) \Leftrightarrow P_t(\Theta).\tilde{C}(X)=(-1)^k\binom{\Theta}{k}.\tilde{C}_0(X)+X^k\tilde{Q}_{M-k}(X)$$

where polynomial $P_t(x)=(-1)^k\binom{x}{k}-2(-1)^t\binom{x}{t}$

Proposition 4: *If $R(x)$ is a polynomial such that $R(x)=(x-\beta)^m.Q(x)$ where $Q(x)$ has the roots $\alpha_1, \alpha_2, \ldots, \alpha_n$ with multiplicity $m_1, m_2, \ldots, m_n$; the solution $F(X)$ of $R(\Theta).F(X)=X^\beta \mathrm{Ln}^k(X)$ is:*

$$F(X)=X^\beta\sum_{i=0}^{k}\frac{k!}{(k-i)!}\left(\frac{1}{Q(x)}\right)^{(k-i)}_{x=\beta}\frac{\mathrm{Ln}^{m+i}(X)}{(i+m)!}$$
$$+\sum_{i=1}^{n}\sum_{j=0}^{m_i-1}\lambda_{ij}X^{\alpha_i}\mathrm{Ln}^j(X)+\sum_{j=0}^{m-1}\mu_j X^\beta \mathrm{Ln}^j(X)$$

where λ_{ij} and μ_j are in the same domain (real or complex) as α_i and β.

The proof of proposition 4 is not very hard and consists in successive inductions on the degree of $R(x)$, on k (for $m=0$ and $m=1$), and then on m.

So, to solve an equation of type $R(\Theta).F(X)=G(X)$, we try to expand $G(X)$ in terms of the form X^β $\mathrm{Ln}^k(X)$ which become by resolution terms of the form $X^\beta \mathrm{Ln}^i(X)$ with $i=m\ldots k+m$ and m is the multiplicity of β in $R(x)$. Moreover, in our problem, the predominant contribution to the cost comes from X^β $\mathrm{Ln}^k(X)$ factors where the real part of β, $\mathrm{Re}(\beta)$, is minimal and k is maximal. Indeed, we have the asymptotical form for the n-th coefficient of the series (*see* [4]):

$$[z^n](1-z)^\beta \mathrm{Ln}^k(1-z)=a_{\beta,k}n^{-(\beta+1)}\mathrm{Ln}^k(n)$$

3.2. General solution

In this paragraph, we do not detail proofs which come directly from classical algebraic manipulations and from propositon 4.

LEMMA: (*a*) *The roots of $P_t(x)$ are the integers $0, 1, \ldots, t-1$; -2; the quantity $3t+2$ if t is odd; and the $2p$ complex roots: $\alpha_1, \alpha_2, \ldots, \alpha_p$ together with their conjugates $\bar{\alpha}_1, \bar{\alpha}_2, \ldots, \bar{\alpha}_p (p=[t/2])$.*

(*b*) *All the roots of $P_t(x)$ have a real part greater than -2.*

THEOREM 1: *The exponential generating function of cost for the Quicksort algorithm has the general form:*

$$\tilde{C}(X)=F(X)+\frac{\lambda_0}{X^2}+\sum_{j=1}^{p}(\lambda_j X^{\alpha_j}+\bar{\lambda}_j X^{\bar{\alpha}_j})+R_M(X)+(\mu \operatorname{Ln}(X)+\lambda)X^{3t+2}$$

where:

$R_M(X)$ is a polynomial of degree M; λ_j are complex for $j=1\ldots p$;

λ, μ, λ_0 are real; $\lambda=0$ if t even; $\mu=0$ if t even or t odd and $M<3t+2$;

and $F(X)$ is a particular solution of: $P_t(\Theta).F(X)=-X^k\,\tilde{C}_0^{(k)}(X)/k!$.

THEOREM 2: *The average cost for lists of size n ($n>M$, $n>3t+2$ if t odd) is given by:*

$$c_n=\frac{C_n}{n!}=[z^n]F(1-z)+\lambda_0(n+1)-\frac{\mu}{(3t+3)\binom{n}{3t+3}}+\sum_{j=1}^{p}B_n^j$$

where $B_n^j=2(-1)^n\operatorname{Re}\left(\lambda_j\binom{\alpha_j}{n}\right)$ is an oscillating term with amplitude $O(n^{-\operatorname{Re}(\alpha_j+1)})$.

Let NR denote the set consisting of complex roots of $P_t(x)$ and of root $3t+2$ when t is odd; we have, as a corollary, the asymptotic form for the average cost:

$$c_n=[z^n]F(1-z)+\lambda_0(n+1)+O(n^{b_t}) \qquad \text{with} \quad b_t=\max_{r\in NR}(-\operatorname{Re}(r+1))<1.$$

Example: If we consider the number of comparisons, we have for example:

$$c_n^0=C_n^0/n!=n+1-2t \qquad \text{and so} \qquad \tilde{C}_0(X)=1/X^2-2t/X.$$

The contribution $F(X)$ is given then by: $P_t(\Theta).F(X)=(k+1)/X^2-2t/X$

Thus, we have the particular solution:

$$F(X)=\frac{k+1}{P_t'(-2)}\frac{\operatorname{Ln}(X)}{X^2}-\frac{2t}{P_t(-1)X}$$

By extracting coefficients, we obtain the average number of comparisons which gives the major contribution to time behaviour of algorithm: (H_p denotes the p-th harmonic number)

$$\frac{C_n}{n!}=\frac{1}{H_{2t+2}-H_{t+1}}(n+1)(H_{n+1}-1)+2t+\lambda_0(n+1)+O(n^{b_t})$$

3.3. Integration constants

In general the constants λ_i and μ are determined from the values of the average cost for lists of size $M+1$ to $M+t+1$, for example by a linear system. Here, we want to show briefly how we can compute these constants and especially λ_0 by using once more operator Θ and generating functions. For this we introduce the polynomial $A_t(x)=P_t(x)/((x+2)\ (x-t+1)\ (x-t+2)\ldots x)$ which has for roots the t elements of NR. If we apply now $A_t(\Theta)$ on the formula of theorem 1, the terms generated by the roots of $A_t(x)$ desappear and we have:

$$A_t(\Theta).[\tilde{C}(X)-F(X)-R_M(X)]=A_t(\Theta).[\lambda_0/X^2]=A_t(-2)\lambda_0/X^2$$

Moreover when we apply on a series a polynomial in Θ of degree p, the coefficient of z^n in the result depends only on the coefficients of orders n to $n+p$ in the initial series. So, if we consider the coefficient of z^{M+1} in the previous formula, we can replace the average cost series by a series $G(X)$ which has the same coefficients of orders $M+1$ to $M+t+1$; we obtain thus:

PROPOSITION 5: *The linear contribution to the average cost is given by:*

$$\lambda_0=\frac{1}{A_t(-2)(M+2)}[z^{M+1}]\,A_t(\Theta).[G(X)-F(X)]$$

where $G(X)$ is a solution of:

$$\frac{(t-\Theta)(t+1-\Theta)\ldots(2t-\Theta)}{(t+2)(t+3)\ldots(2t+2)}.[G(X)-\tilde{C}_0(X)]=\tilde{C}_1(X)$$

The equation for $G(X)$ is obtained by replacing $C(z)$ by $C_1(z)$ in right side of the characteristic equation (I); this translates exaclty the fact that lists of size $M+1$ to $M+t+1$ partition directly into small lists (size $\leq M$) and so their costs are determined in one recurrence step from $C_1(z)$.

In that manner, we extend the results of Sedgewick obtained for $t<2$ through solving of $2t+1$ encased recurrences, by giving for all value of t the linear term of average costs. By changing polynomial $A_t(x)$, we can also compute the other integration constants but they correspond in practice to (very) negligible contributions.

3.4. Results

We are now able to give average values over lists of size $n(n>M$ and $n>3t+2)$ of the characteristic parameters used by Sedgewick. We therefore generalize the results known for the values $t=0$ and $t=1$ (*see* [9]):

Number of partitioning stages ($c_n^0=1$): $\bar{A}_n=K_t\dfrac{n+1}{M+2}-1+O(n^{b_t})$

Number of key comparisons during partitioning stages ($c_n=n+1-2t$):

$$\bar{C}_n=K_t(n+1))(H_{n+1}-H_{M+2}+\frac{1}{2K_t}+\frac{K_t}{2}(H^{(2)}_{2t+2}-H^{(2)}_{t+1}))-2t\bar{A}_n+O(n^{b_t})$$

Number of key exchanges ($c_n^0=$minimum required number $+1$):

$$\bar{B}_n=\frac{(t+1)\bar{C}_n+(t+3)\bar{A}_n}{4t+6}$$

And if we use insertion sort for small lists, we have the linear contributions:

Number of really inserted keys ($c_n^1=n-H_n$):

$$\bar{D}_n=\frac{n+1}{M+2}\left(M+4-K_t\left(H_{M+1}+\frac{2t+1}{t+1}\right)\right)+O(n^{b_t})$$

Number of moved keys ($c_n^1=n(n-1)/4$):

$$\bar{E}_n=K_t(n+1)\left(\frac{(M+3)(t+1)}{4(2t+3)}+\frac{1}{2M+4}-\frac{1}{K_t}\right)+O(n^{b_t})$$

With $K_t=(H_{2t+2}-H_{t+1})^{-1}$ and $b_t<1$; or for particular values of t:

t	0	1	2	3	4
K_t... b_t...	2 $-\infty$	1.7143 -6	1.6216 -6.5	1.5760 -5.5	1.5489 -4.52832
t	5	6	7	$t\to\infty$	
K_t... b_t...	1.5309 -3.75090	1.5181 -3.14282	1.5086 -2.66256	$1/\mathrm{Ln}(2)\approx 1.442$ $1-\pi^2/\mathrm{Ln}^3(2)\,t$	

So we can see that the use of median-of-k procedure (leading to a more centered distribution of the partitioning key) allows to gain directly on the asymptotic behaviour of cost; however the gain becomes less interesting as soon as k is greater than 5. On the other hand, the variant of cutting small lists influences only the linear term of the cost but is not negligible in practice for usual sizes of lists. We can see in particular (simulations or explicit formula for total running time [9]) that an optimal value of M ($M=10$ to 15) permits to gain around 10% on the sorting of lists of size $<5{,}000$.

4. VARIANCE AND MOMENTS

In this section, we want to describe more precisely the distribution of costs around their average values. We compute in particular the variances of previous parameters for all t and the first moments of the number of comparisons in the standard case $t=0$.

4.1. definitions and equations

Let $C_{n,k}$ denotes the probability that the cost is equal to k for lists of size n; from the structure of the algorithm we have the characteristic recurrence ($P_{p,q}$ previously defined):

$$\text{(II)} \qquad (\text{if } n<M)\; C_{n,k}=\sum_{p+q=n-1} P_{p,q} \sum_{i+j+r=k} R(p,q,r)\, C_{p,i}\, C_{q,j}$$

with $R(p,q,r)$ = probability that the partitioning cost is r, if sublists are of size p and q.

Consider now the series $C_n(u)=\Sigma_k C_{n,k}u^k$ and $C(z,u)=\Sigma_n C_n(u)z^n$, we can express all the moments of the distribution by iterate differentiations with respect to u of $C(z, u)$ in $u=1$:

$$c_n=\sum_k k\, C_{n,k}=C'_n(1)=[z^n]\left.\frac{\partial C(z,u)}{\partial u}\right|_{u=1},$$

$$\operatorname{Var}(C_n)=C''_n(1)-C'_n(1)-(C'_n(1))^2\ldots$$

From (II), we obtain easily the following recurrence on series $C_n(u)$ (for $n>M$):

(II *a*)
$$C_n(u)=\sum_{p+q=n-1} P_{p,q}\, R_{p,q}(u)\, C_p(u)\, C_q(u)$$

where

$$R_{p,q}(u)=\sum_r R(p,q,r)\, u^r$$

In the general case, we cannot deduce directly from (II *a*) a simple functional equation for $C(z, u)$ since $R_{p,q}(u)$ does not factorize in the form $R_{p,q}(u)=f(u)\, R_p(u)\, R_q(u)$. We then have to consider the successive derivatives of (II *a*) in $u=1$ and to translate "by hand" the recurrences thus obtained into functional equations for the series of moments: $M_i(z)=\Sigma_n C_n^{(i)}(1)\, z^n$. This operation is not trivial and not always possible but allows in particular to compute (same kind of equations as in part (3) the variance of the number of exchanges during partitioning stages. We will see further that the moments of the other parameters can be computed more simply from an algebraic equation for the bivariate series $C(z, u)$.

To characterize the moments, we also introduce the *semi-invariants* or *cumulants* $\kappa_i(n)$ of order i, defined by (*see* Knuth [7], vol. 1):

$$C_n(e^t)=\exp\left(\sum_{i=1}^{\infty}\kappa_i(n)\frac{t^i}{i!}\right)\quad \text{or}\quad \kappa_i(n)=\left(\frac{\partial}{\partial t}\right)^i \operatorname{Ln}(C_n(e^t))$$

Main properties of cumulants

– $\kappa_1(n)$ is the average cost and $\kappa_2(n)$ is the variance.

– In ordinary cases, the cumulants characterize completely a distribution law and the convergence of all cumulants implies the convergence of the law [the normal distribution is characterized by: $\kappa_i(n)=0$ for $i>2$].

– The cumulants $\kappa'_i(n)$ of the normed centered law are: $\kappa'_i(n)=\kappa_i(n)/\sigma^i(n)$ where $\sigma(n)$ is the standard deviation.

4.2. An interesting case

We assume here that the cost of one partitioning stage is the same for all lists of size n and has the form $C_0(n)=a(n-k)+b$ $(a=1, b=2$ gives the number of comparisons). We can then factorize $R_{p,q}(u)=u^b\, u^{p-t} u^{q-t}$ and translate easily the relation (II a) into a differential equation over $C(z, u)$:

$$\text{(II } b\text{)} \qquad \frac{1}{k!}\frac{\partial^k}{\partial z^k} C(z, u)=u^b\left(\frac{1}{t!}\frac{\partial^t}{\partial z^t} C(zu^a, u)\right)^2+P_{M-k}(z, u)$$

with $P_{M-k}(z, u)$ polynomial of degree $M-k$ in z.

We thus obtain a synthetic equation for $C(z, u)$ which characterizes implicitly the distribution of costs. A direct analysis of this equation (exact or asymptotic determination of $[z^n]\, C(z, u)$) is yet difficult owing to the coupling of z and u. Mellin transform techniques (*see* [3]) sometimes used in similar cases, seem to fail because we have at the same time a product form (square) and derivatives with respect to z of $C(z, u)$.

However, we can obtain easily by successive differentiations with respect to u of relation (II b) and setting $u=1$, equations for the series of moments $M_i(z)=(\partial/\partial z)^i\, C(z, u)\,|_{u=1}$ of the form:

$$P_t(\Theta).\, \tilde{M}_i(X)=f_i(X, \tilde{M}_1(X), \tilde{M}_2(X), \ldots, \tilde{M}_{i-1}(X))$$

We can then solve these successive differential equations by hand $(i=2)$ or using a symbolic manipulation language like Maple or Macsyma. We obtain in this way first moments of the distribution but the shapes of the other moments and of the limiting distribution (if it exists) can only be conjectured. In a recent work [8], M. Regnier has used a non-constructive argument to establish the *existence* of this distribution whose density function still remains to be explicitly characterized.

4.3. Results

(a) After (long) computations, we obtain variances of previously defined parameters:

$$\operatorname{Var}(A_n),\ \operatorname{Var}(D_n),\ \operatorname{Var}(E_n) \text{ are } O(n)$$

$$\operatorname{Var}(B_n)=C_t(n+1)^2+O(n\operatorname{Ln}(n))$$

$$\operatorname{Var}(C_n)=K_t^2\left(A_t-\frac{\pi^2}{6}\right)(n+1)^2-B_t(n+1)H_{n+1}+O(n)$$

where:

$$K_t=\frac{1}{H_{2t+2}-H_{t+1}};\qquad A_t=\frac{(2t+3)\,H^{(2)}_{2t+2}-(t+2)\,H^{(2)}_{t+1}}{t+1};$$

$$B_t=K_t^3\,(H^{(2)}_{2t+2}-H^{(2)}_{t+1});$$

$$C_t=\frac{(t+1)^2\,K_t^2\left(A_t-\dfrac{\pi^2}{6}\right)}{4\,(2t+3)^2}-\frac{K_t\,(t+1)}{(4t+6)^2\,(t+2)}+\frac{1}{4\,(2t+3)\,(2t+5)}.$$

For specific values of t (with $r-1/\mathrm{Ln}(2)\approx 1.4426950408$) we have:

t	0	1	2	3
$K_t^2(A_t-\pi^2/6)$. . . .	42026	.11489	.05301	.03430
B_t.	2	.87464	.55554	.40634
$C_t\times 10^5$.	56.288	30.979	19.720	13.663
t	4	5	6	$t\to\infty$
$K_t^2(A_t-\pi^2/6)$. . . .	.01972	.01138	.01021	$(r/2\,t)^2$
B_t.	.32013	.26403	.22462	$r^3/2\,t$
$C_t\times 10^5$.	10.025	7.6675	6.0532	$((2-r)/8\,t)^2.10^5$

So we can see the great advantage of the use of median-of-k procedure which allows to appreciably recenter the distribution of costs (reduction of extreme cases). The gain on the average costs becomes less tangible for $k\geqq 5$.

(*b*) With the help of Maple, we obtain also the first cumulants of the distribution for the number of comparisons in the standard case $t=0$ (ζ is the Riemann zeta function):

$$\kappa_2(n)=(7-4\,\zeta(2))\,(n+1)^2+O\,(n\,\mathrm{Ln}\,(n))$$

$$\kappa_3(n)=(16\,\zeta(3)-19)\,(n+1)^3+(12\,\zeta(2)-24)\,(n+1)^2+O\,(n\,\mathrm{Ln}\,(n))$$

$$\kappa_4(n)=\left(\frac{937}{9}-96\,\zeta(4)\right)(n+1)^4+\Big)\frac{332}{3}-96\,\zeta(3)\Big)(n+1)^3$$
$$+\left(\frac{2324}{9}-124\,\zeta(2))\,(n+1)^2+O\,(n\,\mathrm{Ln}\,(n)\right)$$

$$\kappa_5(n)=\left(768\,\zeta(5)-\frac{85981}{108}\right)(n+1)^5$$
$$+\left(960\,\zeta(4)-\frac{18835}{18}\right)(n+1)^4+\left(1360\,\zeta(3)-\frac{165995}{108}\right)(n+1)^3+\dots$$

The shape of these cumulants, with regard to intermediate computations, allows us to conjecture (this has been verified for the twelve first cumulants) that for $i>2$ we have:

$$\kappa_i(n)=K_i n^i+O(n^{i-1})$$

where

$$K_i=(-1)^i(\alpha_i-2^i(i-1)!\,\zeta(i))$$

and α_i rational.

This implies then that the normed centered law of the number of comparisons converges to a limiting distribution characteirzed by the cumulants $\kappa_i'=K_i/(7-4\zeta(2))^{i/2}$:

i	3	4	5	6	7
K_i.	.2326	.2080	.2401	.3301	.5132
κ_i'.	.8549	1.178	2.097	4.459	10.66
i	8	9	10	11	12
K_i.	.8630	1.4953	2.4750	3.1777	−0.8591
κ_i'.	27.66	73.938	188.78	373.89	−155.93

Final remark: From Chebyshev's inequality using the moment of order 4, we can give an upper bound for the probability that the number of comparisons C_n is far of his average value:

$$\operatorname{Prob}(|C_n-\bar{C}_n|/\bar{C}_n>E)<0.0461216\,(E\,\operatorname{Ln}(n/M))^{-4}$$

So for $n=1\,000$ and $M=10$, we have with probability $>99\%$ a relative error E less than 21%.

REFERENCES

1 A. V. Aho, J. E. Hopcroft and J. D. Ullman, *The Design and Analysis of Computer Algorithms*, Addison Wesley, Reading, 1974.

2 L. Comtet, *Analyse Combinatoire*, P.U.F., Paris, 1970.

3 P. Flajolet, *Mathematical Analysis of Algorithms and Data Structures*, I.N.R.I.A., Res. Rep., Vol. 400, 1985.

4 P. Flajolet and C. Puech, *Partial match retrieval of multidimensional data*, J.A.C.M. 33, Vol. 2, 1986, pp. 371-407.

5 D. Foata, *La série génératrice exponentielle dans les problèmes d'énumération*, S.M.S., Montreal University Press, 1974.

6 D. H. Greene, *Labelled Formal Languages and Their Uses*, Ph. D. Thesis, 1983.

7 D. E. Knuth, *The Art of Computer Programming*, Vol. 1: *Fundamental Algorithms*, Vol. 3, *Sorting and searching*, Addison Wesley, Reading, 1968, 1973.

8 M. Regnier, *A limiting Distribution for Quicksort*, RAIRO, Th. Informatics and Applications (to appear).

9 R. Sedgewick, *Quicksort*, Garland pub. co., New York, 1980.

10 J. M. Steyaert, Complexité et structure des algorithmes, Thesis Paris, 1984.