Analysis of Hoare's FIND *Algorithm with Median-of-Three Partition*

P. Kirschenhofer,[1] H. Prodinger,[1] C. Martínez[2]
[1]*Institut für Algebra und Diskrete Mathematik, Technical University of Vienna, Wiedner Hauptstraße 8-10, A-1040 Vienna, Austria*
[2]*Departament de Llenguatges i Sistemes Informàtics, Polytechnical University of Catalonia, Pau Gargallo 5, E-08028 Barcelona, Spain*

Received 28 December 1996; revised 20 August 1996

ABSTRACT: Hoare's FIND algorithm can be used to select the jth element out of a file of n elements. It bears a remarkable similarity to Quicksort; in each pass of the algorithm, a pivot element is used to split the file into two subfiles, and recursively the algorithm proceeds with the subfile that contains the sought element. As in Quicksort, different strategies for selecting the pivot are reasonable. In this paper, we consider the Median-of-three version, where the pivot element is chosen as the median of a random sample of three elements. Establishing some hypergeometric differential equations, we find explicit formulae for both the average number of passes and comparisons. We compare these results with the corresponding ones for the basic partition strategy. © 1997 John Wiley & Sons, Inc. *Random Struct. Alg.*, **10**, 143–156 (1997)

1. INTRODUCTION

The problem of *selection* consists in finding the jth smallest (or jth largest) element out of a given set of n elements. Particular important cases are finding the minimum ($j = 1$), the maximum ($j = n$) or the median ($j = \lfloor (n+1)/2 \rfloor$). The first reasonably efficient solution to this problem was devised by Hoare [6] in the early

Correspondence to: C. Martínez

Contract grant sponsor: Acciones Integradas Hispano-Austríacas (Austrian-Spanish Scientific Exchange Program)
Contract grant sponsor: ESPRIT BRA Program of the EC, Project ALCOM II; contract grant number: 7141

1960s. Hoare's FIND algorithm has expected linear running time and is based upon the same design principles as the well-known sorting algorithm Quicksort, also by Hoare [7]. In fact, FIND has also been called Quickselect, One-sided Quicksort, and the like.

Soon after Hoare's work and continuing up to the present, many other solutions have been proposed for the problem of selection. A closely related topic that has attracted much interest is that of lower bounds for this problem, both for the average and worst-case number of comparisons. Here, we mention only the worst-case $\mathscr{O}(n)$ solution to find the median of n elements, proposed by Blum, Floyd, Pratt, Rivest, and Tarjan [2] in 1973. The reader may find a comprehensive list of references in Gonnet and Baeza-Yates's *Handbook* [4] about the topic of selection.

Most of the solutions proposed so far, although better than Hoare's algorithm on a theoretical basis, are not practical because they are difficult to implement, the constant factors and hidden lower-order terms are too large, some of them have been devised for the particular instance of finding the median so their generalization to arbitrary j is difficult, etc. The simplicity, elegance, and robustness of FIND make it the algorithm of choice for most practical situations, like Quicksort for sorting. The probability of large deviations from the expected performance of FIND is probably very small, and the lower-order terms, hidden constant factors, etc. of any reasonable implementation of the algorithm are also quite small.

We now introduce the basic algorithm in detail (see Algorithm 1). We are given an array A of $n > 0$ items, and a rank j, $1 \le j \le n$. For simplicity, we assume that the elements of A are integers. Since we are presenting a recursive implementation, the procedure *find* operates in a segment $A[l:u]$ of the array, such that it must contain the sought element. The first call to *find* is, obviously, *find*$(A, j, 1, n)$.

Algorithm 1 FIND

```
{$l \le j \le u$, $A[l:u]$ contains the $j$th smallest of $A[1:n]$}

function find(var $A$ : array [1 .. $n$] of integer; $j, l, u$ : integer)
  var $r, k$ : integer;

  begin
    if $l = u$ then return $A[l]$;
    $r$ := select_pivot($A, l, u$);
    swap($A[l], A[r]$);
    {$A[l] = p$, $A[l:u]$ contains the $j$th element}
    partition($A, l, u, k$);
    {$\forall i : l \le i < k : A[i] \le p$, $A[k] = p$, and $\forall i : k < i \le u : A[i] > p$}
    if $j = k$ then return $A[k]$;
          else if $j < k$ then return find($A, j, l, k-1$);
                    else return find($A, j, k+1, u$)
  end
```

In order to select the jth smallest element, a certain element of the given segment, the *pivot*, is brought into its correct position, left of it being only smaller, right of it only larger elements. The index r of the pivot is computed by *select_pivot*. For the time being, we can assume that this function returns a value between l and u at random. Then the pivot is exchanged with $A[l]$ and the procedure *partition* is called. After the partition, the segment $A[l:u]$ is divided into three parts: the first one, $A[l:k-1]$, contains all elements smaller or equal to the pivot; $A[k]$ contains the pivot; and $A[k+1:u]$ contains all elements larger than the pivot (see Algorithm 2). Clearly, if m is the number of elements in the segment, then $m-1$ comparisons are needed to determine the correct position of the pivot. Recall that k is the final position of the pivot after the partition. Hence, if $j=k$, then $A[k]$ is returned, and we are done. Otherwise, if $j<k$, then the jth element must be in $A[1..k-1]$ and the procedure is recursively applied to that segment. If $k<j$, then the jth element lies in $A[k+1..n]$, and we call *find* on that segment.

Unlike Quicksort, where both segments of the partitioned array have to be sorted recursively, the FIND algorithm has only to work on either the left or the right segment. As a consequence, an iterative implementation of FIND is almost straightforward, and no additional auxiliary storage is needed.

Algorithm 2 Partition of the segment $A[l:u]$

```
procedure partition(var $A$ : array [$1..n$] of integer; $l, u$ : integer;
                    var $k$ : integer)

  var $i, p$ : integer;
  begin
    $k := l + 1$; $p := A[l]$;
    for $i := l + 1$ to $u$ do
      if $A[i] \le p$ then begin
                             swap($A[k]$, $A[i]$);
                             $k := k + 1$
                         end
    swap($A[l]$, $A[k]$)
  end
```

The preliminary analysis of FIND is by Hoare [7]. In 1970, Kruseman-Aretz showed that it makes approximately $2(1+\log 2)n$ comparisons to find the median of n elements [13]. One year after, Knuth presented the first detailed analysis of this algorithm [9], where he computed the expected number of comparisons to find the jth out of n. A similar approach may be used to compute the expected number of passes or calls to *find*, the expected number of exchanges in the array, etc. Devroye [3] has identified upper bounds on the moments of arbitrary order for the number of comparisons, and Mahmoud, Modarres, and Smythe [11] have investigated the expectation, variance, and distribution of the number of comparisons needed to select an element of *random rank*, that is, when j is the value of a uniformly distributed random variable, between 1 and n.

Several variations of the basic scheme may yield significant improvements in the algorithm's performance. For instance, instead of taking a random element as the pivot, a random sample of 3 elements is taken, and the middle of it is used as the pivot element. The idea is that it is more likely that no subfile is degenerate. This strategy, called the *Median-of-three* variant, is very well understood in the case of Quicksort [10, 12]. Of course, the Median-of-three strategy can also be used in the context of selection. Quite recently, Anderson and Brown [1] presented a preliminary analysis of this variant of the FIND algorithm.

In this paper, we give a precise and detailed analysis of the Median-of-three. We will find an explicit expression for the average number of passes of the recursive algorithm, when the jth element has to be selected out of n and for the average number of comparisons between elements made during the partitioning. We show how to solve the intricate recurrences arising in the analysis of this variant, and how the theory of hypergeometric differential equations comes into play in the solution.

For the rest of the paper, we will assume that the n given elements are distinct. Since the behavior of the algorithm only depends on the relative ordering of the elements and the pivot is selected from random samples, the analysis can be carried out as if the input were a random permutation of $\{1, 2, \dots, n\}$. In Section 2, we present the analysis of the expected number of passes, and in Section 3, that of the number of comparisons.

Section 4 is devoted to discuss several related results and various comparisons of the cost of the Median-of-three variant with the cost of the basic algorithm.

2. THE NUMBER OF PASSES

Let $\mathbf{P}_{n,j}(z)$ be the probability generating function of the number of passes, when selecting the jth element out of n. The coefficient of z^k in $\mathbf{P}_{n,j}(z)$, denoted $[z^k]\mathbf{P}_{n,j}(z)$ as usual, is by definition [5]

$$[z^k]\mathbf{P}_{n,j}(z) = \text{Prob}\{\text{“number of passes to select the } j\text{th out of } n\text{”} = k\}.$$

A file of size 1 or 2 is not treated recursively; hence

$$\mathbf{P}_{1,1}(z) = \mathbf{P}_{2,1}(z) = \mathbf{P}_{2,2}(z) = z.$$

For $n \geq 3$ we have

$$\mathbf{P}_{n,j}(z) = z\left[\sum_{k=1}^{j-1} \pi_{n,k}\mathbf{P}_{n-k,j-k}(z) + \pi_{n,j} + \sum_{k=j+1}^{n} \pi_{n,k}\mathbf{P}_{k-1,j}(z)\right]. \tag{1}$$

Here and in the sequel,

$$\pi_{n,k} = \frac{(k-1)(n-k)}{\binom{n}{3}} \tag{2}$$

are the *splitting probabilities*. It is easy to see that this quantity is the probability that the middle element of a random sample of 3 elements is the kth element in a random permutation of n elements. The recursion above is almost self-explanatory. If $k<j$, we have to search in the right subfile of size $n-k$, but the sought element is the $(j-k)$th of the subfile. If $k=j$, we are done, and if $k>j$, we are still looking for the jth element in the left subfile of size $k-1$.

Solving the recurrence above is a very difficult problem; however, it is straightforward to obtain recurrences for each factorial moment of the random variable "“number of passes” in a systematic way, the most important moment being the expectation. The average number $P_{n,j}$ of passes is given by $(d/dz)\mathbf{P}_{n,j}(z)|_{z=1}$. Hence we have $P_{1,1}=P_{2,1}=P_{2,2}=1$ and for $n\geq 3$

$$P_{n,j} = 1 + \sum_{k=1}^{j-1} \pi_{n,k} P_{n-k,j-k} + \sum_{k=j+1}^{n} \pi_{n,k} P_{k-1,j}. \tag{3}$$

For convenience, we set $P_{n,j}=0$ whenever $n<j$. We also note the symmetry $P_{n,j}=P_{n,n+1-j}$ which holds for obvious combinatorial reasons.

It is now natural to set up generating functions

$$P_j(z) = \sum_{n\geq j} P_{n,j} z^n.$$

Multiplying the recursion by $\binom{n}{3}z^{n-3}$ and summing up, we obtain

$$\frac{1}{6}P_j'''(z) = \frac{1}{(1-z)^4} - \sum_{k=3}^{j-1}\binom{k}{3}z^{k-3} + \sum_{k=2}^{j-1}(k-1)z^{k-2}P_{j-k}'(z) + \frac{P_j'(z)}{(1-z)^2}. \tag{4}$$

This equation holds for all $j\geq 1$ and the initial values are $P_j(0)=0$, $P_j'(0)=[j=1]$, and $P_j''(0)=2\cdot[j\leq 2]$, where the value of $[R]$ is 1 if the predicate R is true and 0 otherwise (compare [5]). It is simple to reduce the order of the differential equations; we define $Q_j(z)=P_j'(z)$ and rewrite Eq. (4) as

$$\frac{1}{6}Q_j''(z) = \frac{1}{(1-z)^4} - \sum_{k=3}^{j-1}\binom{k}{3}z^{k-3} + \sum_{k=2}^{j-1}(k-1)z^{k-2}Q_{j-k}(z) + \frac{Q_j(z)}{(1-z)^2} \tag{5}$$

with initial conditions $Q_j(0)=[j=1]$ and $Q_j'(0)=2\cdot[j\leq 2]$.

Solving the first few instances, we get

$$Q_1(z) = \frac{6}{5}\frac{1}{(1-z)^2}\log\frac{1}{1-z} + \frac{19}{25}\frac{1}{(1-z)^2} + \frac{6}{25}(1-z)^3,$$

$$Q_2(z) = \frac{6}{5}\frac{1}{(1-z)^2}\log\frac{1}{1-z} + \frac{4}{25}\frac{1}{(1-z)^2} - \frac{4}{25}(1-z)^3,$$

$$Q_3(z) = \frac{6}{5}\frac{1}{(1-z)^2}\log\frac{1}{1-z} + \frac{108}{175}\frac{1}{(1-z)^2} - \frac{6}{5}\log\frac{1}{1-z}$$
$$- \frac{24}{25} + \frac{6}{25}(1-z)^3 + \frac{18}{175}(1-z)^5,$$

etc.

It is meaningful to define the double generating function

$$\mathcal{Q}(z,u) := \sum_{j\geq 1} Q_j(z)u^j.$$

Then the system of differential equations (5) translates into a single equation, viz.,

$$\frac{1}{6}\frac{\partial^2\mathcal{Q}}{\partial z^2} - \left(\frac{1}{(1-z)^2} + \frac{u^2}{(1-uz)^2}\right)\mathcal{Q} = \frac{u}{1-u}\left(\frac{1}{(1-z)^4} - \frac{u^3}{(1-uz)^4}\right), \tag{6}$$

with the initial conditions $\mathcal{Q}(0,u) = u$ and $(\partial\mathcal{Q}/\partial z)(0,u) = 2u(1+u)$.

In order to solve this equation it is helpful to express $\mathcal{Q}(z,u)$ as

$$\mathcal{Q}(z,u) = \frac{1}{(1-z)^2(1-uz)^2}\mathcal{E}(z,u), \tag{7}$$

and substitute $z = 1 + t(1-u)/u$. With the notation

$$\mathcal{Q}(t,u) = \mathcal{E}\left(1 + \frac{1-u}{u}t, u\right), \tag{8}$$

the original differential equation (6) transforms into the *hypergeometric equation*

$$t(1-t)\frac{\partial^2\mathcal{G}}{\partial t^2} + (-4+8t)\frac{\partial\mathcal{G}}{\partial t} - 8\mathcal{G} = 6(1-u)\frac{u(1-t)^4 - t^4}{t(1-t)}, \tag{9}$$

with the initial conditions $\mathcal{G}(-u/(1-u),u) = u$ and $(\partial\mathcal{G}/\partial t)(-u/(1-u),u) = 0$.

For the reader's convenience, we recall [8] that, under certain assumptions on the parameters, the hypergeometric differential equation

$$t(1-t)g''(t) + (c-(1+a+b)t)g'(t) - abg(t) = 0$$

has a neighborhood of $t = 0$ the solution

$$g(t) = c_1 \cdot {}_2F_1\left(\begin{matrix} a, b \\ c \end{matrix} \,\middle|\, t \right) + c_2 t^{1-c} \cdot {}_2F_1\left(\begin{matrix} a-c+1, b-c+1 \\ 2-c \end{matrix} \,\middle|\, t \right),$$

where ${}_2F_1$ denotes a *hypergeometric function* [5]. In our instance, the two hypergeometric functions degenerate to polynomials ("terminating hypergeometric series"), and we get the homogeneous solution

$$\mathcal{G}^{[\text{hom}]}(t,u) = c_1(u)(1-2t) + c_2(u)t^5\left(1 - 2t + \frac{10}{7}t^2 - \frac{5}{14}t^3\right).$$

Using the *variation of parameters method* [8], inserting the initial conditions, and, resubstituting, we find

$$\mathcal{Q}(z,u) = \mathcal{Q}_1(z,u) + u^2\mathcal{Q}_1(uz, u^{-1}) + \mathcal{Q}_2(z,u), \tag{10}$$

where

$$\begin{aligned}\mathcal{Q}_1(z,u) = \frac{u}{(1-z)^2(1-uz)^2}\log\frac{1}{1-z}\Bigg\{&\frac{6}{5}(1-u) + \frac{12}{5}u(1-z) \\ &- \frac{24}{5}\frac{(1+u)u^4}{(1-u)^4}(1-z)^5 - \frac{48}{5}\frac{(1+u)u^5}{(1-u)^5}(1-z)^6 \\ &- \frac{48}{7}\frac{(1+u)u^6}{(1-u)^6}(1-z)^7 - \frac{12}{7}\frac{(1+u)u^7}{(1-u)^7}(1-z)^8\Bigg\},\end{aligned}$$

$$\begin{aligned}\mathcal{Q}_2(z,u) = \frac{u}{(1-z)^2(1-uz)^2}\Bigg\{&-\frac{(1-u)(R(u)+94)}{175} \\ &- \frac{2(uR(u) - 75 + 124u)}{175}(1-z) + \frac{u(9-5u)}{7(1-u)}(1-z)^2 \\ &- \frac{4u^2(1+u)}{7(1-u)^2}(1-z)^3 + \frac{13u^3(1+u)}{7(1-u)^3}(1-z)^4 \\ &+ \frac{2(7S(u) + 2u^4(211+379u))}{175(1-u)^4}(1-z)^5 \\ &+ \frac{2u(2S(u) + 3u^4(9+41u))}{25(1-u)^5}(1-z)^6 \\ &+ \frac{4u^2(S(u) - 3u^4(3-13u))}{35(1-u)^6}(1-z)^7 \\ &+ \frac{u^3(S(u) - 24u^4(1-u))}{35(1-u)^7}(1-z)^8\Bigg\},\end{aligned}$$

and $R(u) := -(1-u)(11-3u)(7-2u+u^2)$ and $S(u) := u^5R(u) - u^4R(u^{-1})$.

We notice that the symmetry property $P_{n,j} = P_{n,n+1-j}$ is reflected on the level of generating functions by $\mathscr{Q}(z,u) = u^2\mathscr{Q}(uz,u^{-1})$; in fact we have $\mathscr{Q}_2(z,u) = u^2\mathscr{Q}_2(uz,u^{-1})$.

Recall that $Q_j(z) = P_j'(z)$ and therefore $P_{n,j} = \frac{1}{n}[z^{n-1}u^j]\mathscr{Q}(z,u)$. To extract these coefficients from Eq. (10), we use

$$\frac{1}{n}[z^{n-1}u^r]\frac{(1-z)^i}{(1-uz)^2}\log\frac{1}{1-z} = \frac{r+1}{n}\frac{(-1)^i i!}{(n-1-r)^{\underline{i+1}}},$$

$$\text{for } 0 \le i < n-1-r,$$

$$\frac{1}{n}[z^{n-1}u^r]\frac{1}{(1-z)(1-uz)^2}\log\frac{1}{1-z} = \frac{r+1}{n}H_{n-1-r},$$

$$\frac{1}{n}[z^{n-1}u^r]\frac{1}{(1-z)^2(1-uz)^2}\log\frac{1}{1-z} = \frac{(r+1)(n-r)}{n}(H_{n-r}-1),$$

and Theorem 2.1 follows. Here, and for the rest of the paper, $n^{\underline{i}} = n(n-1)\dots(n-i+1)$ denotes the ith *falling factorial* of n and $H_n = \sum_{1\le i\le n} 1/i$ the nth harmonic number [5].

Theorem 2.1. *The average number of passes to select the jth element out of a random permutation of n elements using Hoare's* FIND *algorithm with Median-of-three partition is*

$$\begin{aligned} P_{n,j} = {} & \frac{24}{35}H_n + \frac{18}{35}H_j + \frac{18}{35}H_{n+1-j} + \frac{12}{35j} + \frac{12}{35(n+1-j)} \\ & - \frac{304}{175} - \frac{6}{7n} + \frac{18j}{35n} - \frac{12(j-1)^2}{35n^{\underline{2}}} - \frac{4(2j-3)(j-1)^{\underline{2}}}{35n^{\underline{3}}} \\ & - \frac{6(j-2)(j-1)^{\underline{3}}}{35n^{\underline{4}}} + \frac{6(2j-5)(j-1)^{\underline{4}}}{35n^{\underline{5}}} - \frac{4(j-3)(j-1)^{\underline{5}}}{35n^{\underline{6}}} \end{aligned} \tag{11}$$

for $5 \le j \le n-4$. Table I *collects also the formulae for the cases $j = 1,2,3,4$. The remaining cases, when $j \le n \le j+3$, are handled by using the symmetry $P_{n,j} = P_{n,n+1-j}$. Since $1 \le n+1-j \le 4$, the expressions for $j = 1,2,3,4$ apply.*

The formula for $P_{n,j}$ is symmetric in j and $n+1-j$, even though it is not apparent at the first glance. It is possible to produce obviously symmetric expressions for $P_{n,j}$, but they are messier and we stick to the presented form.

3. THE NUMBER OF COMPARISONS

The analysis of the average number of comparisons follows the pattern shown in Section 2.

TABLE I Average number of passes to select the jth element out of n (Theorem 2.1)

j	$P_{n,j}$
1	$\frac{6}{5}H_n - \frac{11}{25}$, for $n \geq 5$; $P_{1,1} = 1$, $P_{2,1} = 1$, $P_{3,1} = 2$, $P_{4,1} = 2$
2	$\frac{6}{5}H_n - \frac{26}{25}$, for $n \geq 5$; $P_{2,2} = 1$, $P_{3,2} = 1$, $P_{4,2} = \frac{3}{2}$
3	$\frac{6}{5}H_n - \frac{102}{175} - \frac{6}{5}\frac{1}{n^2}$, for $n \geq 7$; $P_{3,3} = 2$, $P_{4,3} = \frac{3}{2}$, $P_{5,3} = \frac{11}{5}$, $P_{6,3} = \frac{23}{10}$
4	$\frac{6}{5}H_n - \frac{209}{350} - \frac{18}{5}\frac{1}{n^2} - \frac{12}{5}\frac{1}{n^3}$, for $n \geq 8$; $P_{4,4} = 2$, $P_{5,4} = \frac{17}{10}$, $P_{6,4} = \frac{23}{10}$, $P_{7,4} = \frac{12}{5}$
$5 \leq j \leq n-4$	$\frac{24}{35}H_n + \frac{18}{35}H_j + \frac{18}{35}H_{n+1-j} + \frac{12}{35j} + \frac{12}{35(n+1-j)} - \frac{304}{175} - \frac{6}{7n} + \frac{18j}{35n} - \frac{12(j-1)^2}{35n^2} - \frac{4(2j-3)(j-1)^2}{35n^3} - \frac{6(j-2)(j-1)^3}{35n^4} + \frac{6(2j-5)(j-1)^4}{35n^5} - \frac{4(j-3)(j-1)^5}{35n^6}$

First, we introduce $\mathbf{C}_{n,j}(z)$, the probability generating function of the number of comparisons. We will not count the comparisons made to select pivots, i.e., to decide the middle element in the random samples of three elements. A file of size 1 or 2 is not treated recursively; hence

$$\mathbf{C}_{1,1}(z) = 1, \qquad \mathbf{C}_{2,1}(z) = \mathbf{C}_{2,2}(z) = z.$$

For $n \geq 3$ we have

$$\mathbf{C}_{n,j}(z) = z^{n-1}\left[\sum_{k=1}^{j-1} \pi_{n,k}\mathbf{C}_{n-k,j-k}(z) + \pi_{n,j} + \sum_{k=j+1}^{n} \pi_{n,k}\mathbf{C}_{k-1,j}(z)\right]. \quad (12)$$

The recurrence is obtained in the same way as the one for the number of passes, the factor z^{n-1} in front of everything counting for the $n-1$ comparisons needed to partition the file around the pivot element.

The average number $C_{n,j}$ of comparisons is given by $(d/dz)\mathbf{C}_{n,j}(z)|_{z=1}$. Hence, we have $C_{1,1}=0$, $C_{2,1}=C_{2,2}=1$, and for $n \geq 3$

$$C_{n,j} = n - 1 + \sum_{k=1}^{j-1} \pi_{n,k} C_{n-k,j-k} + \sum_{k=j+1}^{n} \pi_{n,k} C_{k-1,j}. \tag{13}$$

Again, we have $C_{n,j} = C_{n,n+1-j}$ for any j, $1 \leq j \leq n$. As in the analysis of the number of passes, we assume $C_{n,j}=0$ whenever $n<j$ and set up generating functions

$$C_j(z) = \sum_{n \geq j} C_{n,j} z^n.$$

It turns out that the order of differential equations for the $C_j(z)$'s, corresponding to the recurrence (13), may be reduced by using $D_j(z) = C_j'(z)$ instead. As in the analysis of the number of passes, the bivariate generating function

$$\mathscr{D}(z,u) := \sum_{j \geq 1} D_j(z) u^j,$$

satisfies a second-order partial differential equation. After appropriate transformations, the problem reduces to solving an hypergeometric differential equation. We will not repeat the intermediate steps, since they are completely analogous to those shown in the previous section. The final result is cast into the form of Theorem 3.1.

Theorem 3.1. *The average number of comparisons to select the jth element out of a random permutation of n elements using Hoare's* FIND *algorithm with Median-of-three partition is*

$$\begin{aligned} C_{n,j} = 2n &+ \frac{72}{35}H_n - \frac{156}{35}H_j - \frac{156}{35}H_{n+1-j} + \frac{36}{35j} + \frac{36}{35(n+1-j)} + \frac{113}{175} + \frac{24}{7n} \\ &+ 3j - \frac{3(j-1)^2}{n} - \frac{156j}{35n} - \frac{36(j-1)^2}{35n^2} - \frac{12(2j-3)(j-1)^2}{35n^3} \\ &- \frac{18(j-2)(j-1)^3}{35n^4} + \frac{18(2j-5)(j-1)^4}{35n^5} - \frac{12(j-3)(j-1)^5}{35n^6} \end{aligned} \tag{14}$$

for $5 \leq j \leq n-4$. *Table* II *also collects the formulae for the cases* $j = 1,2,3,4$. *The remaining cases, when* $j \leq n \leq j+3$, *are handled by using the symmetry* $C_{n,j} = C_{n,n+1-j}$. *Since* $1 \leq n+1-j \leq 4$, *the expressions for* $j=1,2,3,4$ *apply.*

TABLE II Average number of comparisons to select the jth element out of n (Theorem 3.1).

j	$C_{n,j}$
1	$2n - \frac{12}{5}H_n + \frac{12}{25}, \quad \text{for } n \geq 5$ $C_{1,1} = 0, \quad C_{2,1} = 1, \quad C_{3,1} = 2, \quad C_{4,1} = \frac{7}{2}$
2	$2n - \frac{12}{5}H_n + \frac{12}{25}, \quad \text{for } n \geq 5$ $C_{2,2} = 1, \quad C_{3,2} = 2, \quad C_{4,2} = \frac{7}{2}$
3	$2n - \frac{12}{5}H_n + \frac{314}{175} - \frac{6}{n} + \frac{12}{5n^2}, \quad \text{for } n \geq 7$ $C_{3,3} = 2, \quad C_{4,3} = \frac{7}{2}, \quad C_{5,3} = \frac{26}{5}, \quad C_{6,3} = 7$
4	$2n - \frac{12}{5}H_n + \frac{634}{175} - \frac{18}{n} + \frac{36}{5n^2} + \frac{24}{5n^3}, \quad \text{for } n \geq 8$ $C_{4,4} = \frac{7}{2}, \quad C_{5,4} = 5, \quad C_{6,4} = 7, \quad C_{7,4} = \frac{316}{35}$
$5 \leq j \leq n-4$	$2n + \frac{72}{35}H_n - \frac{156}{35}H_j - \frac{156}{35}H_{n+1-j}$ $+ \frac{36}{35j} + \frac{36}{35(n+1-j)} + \frac{113}{175} + \frac{24}{7n} + 3j - \frac{3(j-1)^2}{n}$ $- \frac{156j}{35n} - \frac{36(j-1)^2}{35n^2} - \frac{12(2j-3)(j-1)^2}{35n^3}$ $- \frac{18(j-2)(j-1)^3}{35n^4} + \frac{18(2j-5)(j-1)^4}{35n^5} - \frac{12(j-3)(j-1)^5}{35n^6}$

4. RELATED RESULTS

The procedure shown in Sections 2 and 3 can be used, in principle, to analyze many other quantities such as the average number of exchanges $X_{n,j}$, the number of calls to the partition procedure—which is not actually the same as the number of passes—etc. For instance, in the paper [1], the authors make the assumption that each partitioning phase costs $n-3$ comparisons, as opposed to the assumption of $n-1$ comparisons to partition a file of size n. The idea behind their approach is that the comparisons between the pivot and the other two elements in the sample during the partitioning can be avoided since they were already compared when selecting the pivot. In this instance, the result, very similar to the one given in

Theorem 3.1, is

$$C_{n,j} = 2n + \frac{24}{35}H_n - \frac{192}{35}H_j - \frac{192}{35}H_{n+1-j} + \frac{12}{35j} + \frac{12}{35(n+1-j)} + \frac{921}{175} + \frac{36}{7n} + 3j - \frac{3(j-1)^2}{n} - \frac{192j}{35n} - \frac{12(j-1)^2}{35n^2} - \frac{4(2j-3)(j-1)^2}{35n^3} - \frac{6(j-2)(j-1)^3}{35n^4} + \frac{6(2j-5)(j-1)^4}{35n^5} - \frac{4(j-3)(j-1)^5}{35n^6}, \tag{15}$$

which holds for $j \geq 5$ and $n+1-j \geq 5$.

Neither the previous work of Anderson and Brown [1] nor this work consider the total number of comparisons, since the comparisons made to select the pivots were not counted. Of course, the exact average number of comparisons $T_{n,j}$ can be computed using the methods presented in this paper, but it is approximately

$$T_{n,j} \sim \frac{8}{3} \times P_{n,j} + C_{n,j},$$

since we need $8/3$ comparisons on the average to select the median of three elements. Hence, for $5 \leq j \leq n-4$,

$$T_{n,j} = 2n + \frac{136}{35}H_n - \frac{108}{35}H_j - \frac{108}{35}H_{n+1-j} + \mathscr{O}(1).$$

If we assume $n-3$ comparisons to partition a file of size n, instead of $n-1$, the same estimation results.

An interesting question that we consider now is the performance of the algorithm when looking for the median $j = \lfloor (n+1)/2 \rfloor$ and when selecting an element at random, i.e., j takes any value between 1 and n with identical probability $1/n$. We compare then these performances with the corresponding ones for the basic algorithm, that is, when the pivot is randomly chosen. Let us recall that the average number of passes for the basic algorithm [9] is

$$P_{n,j} = H_j + H_{n+1-j} - 1, \tag{16}$$

while the average number of comparisons is

$$C_{n,j} = 2\big(n + 3 + (n+1)H_n - (j+2)H_j - (n+3-j)H_{n+1-j}\big). \tag{17}$$

If we specialize formula (11) for $n = 2N+1$ and $j = N+1$, then

$$\begin{aligned} P_{2N+1,N+1} &= \frac{24}{35}H_{2N+1} + \frac{36}{35}H_{N+1} - \frac{4469}{2800} + \mathscr{O}\left(\frac{1}{N}\right) \\ &= \frac{12}{7}\log N + \frac{12}{7}\gamma + \frac{24}{35}\log 2 - \frac{4469}{2800} + \mathscr{O}\left(\frac{1}{N}\right). \end{aligned}$$

On the other hand, the average number of passes to find the median in the classical case is

$$P_{2N+1,N+1} = 2H_{N+1} - 1 \sim 2\log N + 2\gamma - 1 + \mathcal{O}\left(\frac{1}{N}\right).$$

Comparing the leading coefficients, $\frac{12}{7}$ and $2 = \frac{14}{7}$, we see that the savings by the refined method are 14.3%.

By taking $n = 2N+1$ and $j = N+1$ in Eqs. (14) and (17), we obtain the average number of comparisons made to search for the median, in the case of the Median-of-three and the basic algorithm, respectively. The formulae read this time

$$\begin{aligned} C_{2N+1,N+1} &= \frac{11}{2}N + \frac{14893}{2800} + \frac{72}{35}H_{2N+1} - \frac{312}{35}H_{N+1} + \mathcal{O}\left(\frac{1}{N}\right) \\ &= \frac{11}{2}N - \frac{48}{7}\log N - \frac{48}{7}\gamma + \frac{14893}{2800} + \frac{72}{35}\log 2 + \mathcal{O}\left(\frac{1}{N}\right), \end{aligned}$$

and

$$\begin{aligned} C_{2N+1,N+1} &= 4(N + 2 + (N+1)H_{2N+1} - (N+3)H_{N+1}) \\ &= 4(1+\log 2)N - 8\log N - 8(\gamma - 1) + 4\log 2 + \mathcal{O}\left(\frac{1}{N}\right). \end{aligned}$$

The leading coefficients (of the order N), if we search for the median, are this time $\frac{11}{2} = 5.5$ and $4(1+\log 2) = 6.772588722$, so that the savings are roughly 19%. To be fair, the total number of comparisons $T_{2N+1,N+1}$ should have been used rather than $C_{2N+1,N+1}$ to compute these savings, but it actually does not matter since the main term in both quantities is $\frac{11}{2}N$.

The average number of passes and comparisons to find an element of random rank out of n elements are given by

$$\mathcal{P}_n = \frac{1}{n}\sum_{1 \le j \le n} P_{n,j},$$

$$\mathcal{C}_n = \frac{1}{n}\sum_{1 \le j \le n} C_{n,j}.$$

Comparing the values of $\mathcal{P}_n$ for the classical and the Median-of-three algorithms, we find that they are $2H_n + \mathcal{O}(1)$ and $\frac{12}{7}H_n + \mathcal{O}(1)$, respectively, and hence the average savings are 14.3%. On the other hand, $\mathcal{C}_n = 3n - 8H_n + \mathcal{O}(1)$ for the standard algorithm and $\mathcal{C}_n = \frac{5}{2}n - \frac{48}{7}H_n + \mathcal{O}(1)$ for the variant, showing that an average of 16.6% of the comparisons are saved when using the median-of-three.[1]

ACKNOWLEDGMENTS

We thank Friedrich Hubalek for valuable discussions. The computer algebra system MAPLE was heavily used in the preparation of this work.

[1]As in the example, where we considered the number of comparisons to find the median, the main term in $\mathcal{C}_n$ does not change if we use $T_{n,j}$ instead of $C_{n,j}$ to compute this "grand" average.

REFERENCES

[1] D. H. Anderson and R. Brown, Combinatorial aspects of C. A. R. Hoare's Find algorithm, *Australasian J. Combinat.*, **5**, 109–119 (1992).

[2] M. Blum, R. W. Floyd, V. R. Pratt, R. Rivest, and R. Tarjan, Time bounds for selection, *J. Comput. Syst. Sci.*, **7**, 448–461 (1973).

[3] L. Devroye, Exponential bounds for the running time of a selection algorithm, *J. Comput. Syst. Sci.*, **29**, 1–7 (1984).

[4] G. H. Gonnet and R. Baeza-Yates, *Handbook of Algorithms and Data Structures—In Pascal and C*, 2nd ed., Addison-Wesley, Reading, MA, 1991.

[5] R. L. Graham, D. E. Knuth, and O. Patashnik, *Concrete Mathematics*, Addison-Wesley, Reading, MA, 1989.

[6] C. A. R. Hoare, Find (Algorithm 65), *Commun. ACM*, **4**, 321–322 (1961).

[7] C. A. R. Hoare, Quicksort, *Comput. J.*, **5**, 10–15 (1962).

[8] E. Kamke, *Differentialgleichungen: Lösungsmethoden und Lösungen*, Teubner, Stuttgart, 1977.

[9] D. E. Knuth, Mathematical analysis of algorithms, in *Information Processing '71*, *Proc. of the 1971 IFIP Congress*, North-Holland, Amsterdam, 1972, pp. 19–27.

[10] D. E. Knuth, *The Art of Computer Programming: Sorting and Searching*, Addison-Wesley, Reading, MA, 1973, Vol. 3.

[11] H. M. Mahmoud, R. M. Modarres, and R. T. Smythe, Analysis of one-sided quicksort: an algorithm for order statistics, *RAIRO Theor. Inf. Appl.*, **29**, 255–276 (1995).

[12] R. Sedgewick, *Quicksort*, Garland, New York, 1978.

[13] M. H. van Emden, Increasing the efficiency of Quicksort, *Commun. ACM*, **13**, 563–567 (1970).