Optimal Sampling Strategies in Quicksort and Quickselect*

Conrado Martínez and Salvador Roura†

January 7, 1998

Abstract

It is well known that the performance of quicksort can be substantially improved by selecting the median of a sample of three elements as the pivot of each partitioning stage. This variant is easily generalized to samples of size $s = 2k + 1$. For large samples the partitions are better as the median of the sample makes a more accurate estimate of the median of the array to be sorted, but the amount of additional comparisons and exchanges to find the median of the sample also increases.

We show that the optimal sample size to minimize the average total cost of quicksort (which includes both comparisons and exchanges) is $s = a \cdot \sqrt{n} + o(\sqrt{n})$. We also give a closed expression for the constant factor a, which depends on the median-finding algorithm and the costs of elementary comparisons and exchanges. The result above holds in most situations, unless the cost of an exchange exceeds by far the cost of a comparison. In that particular case, it is better to select not the median of the samples, but the $(p+1)^{\mathrm{th}}$ element. The value of p can be precisely determined as a function of the ratio between the cost of an exchange and the cost of a comparison.

We also show how to apply the same ideas and techniques to the analysis of quickselect, a general-purpose selection algorithm closely related to quicksort, and get similar results to those for quicksort.

1 Introduction

Early in the sixties, C.A.R. Hoare devised two efficient algorithms, *quicksort* and *quickselect* (also known as *Hoare's* FIND *algorithm* and as *one-sided quicksort*), for internal sorting and selection, respectively, both of great theoretical and practical importance [7, 8]. These algorithms combine elegance and efficiency, and still remain among the best practical algorithms for sorting and

*This research was supported by the ESPRIT LTR Project ALCOM-IT, contract # 20244 and by a grant from CIRIT (Comissió Interdepartamental de Recerca i Innovació Tecnològica).

†Departament de Llenguatges i Sistemes Informàtics, Universitat Politècnica de Catalunya, E-08034 Barcelona, Catalonia, Spain. E-mail: {conrado,roura}@lsi.upc.es

selection. Because of their simplicity and robustness, they can be subjected to many improvements, and finely tuned implementations of these algorithms beat other methods under most circumstances.

Both algorithms are based upon the *divide-and-conquer* principle and operate using similar ideas. (A brief, but complete description is given in Section 2. Excellent sources for background information and further references on quicksort and quickselect and their analysis include [5, 10, 15, 16, 18, 17, 19].) Contrary to other divide-and-conquer algorithms, quicksort and quickselect are not guaranteed to divide the problem into subproblems of approximately the same size. Not even there is certainty that the size of the subproblems will be a fraction of the size of the original problem. Hence, their worst-case performance is far from good; both need $\Theta(n^2)$ time in the worst case, where n is the size of the input array. But in practice the partition of the current subarray in each invocation is likely to produce subarrays whose size is a fraction of the total size and therefore, the expected performance of quicksort is $\Theta(n \log n)$ and that of quickselect is $\Theta(n)$, where the hidden constant factors are quite small. Furthermore, it has been shown that the standard deviation of the number of comparisons in both algorithms is $\Theta(n)$. Thus in the case of quicksort, the probability of large deviations from the expected performance is rather low. This is not true for quickselect, as the expected cost and the standard deviation are of the same order of magnitude. However, we show in this paper that quickselect can be improved to make its deviation be $\Theta(n^{3/4})$, that is, significantly smaller than the average cost.

Quicksort with median–of–three is a well-known variant whose benefits have been endorsed both by theoretical analysis and practical experiments. In this variant of quicksort, we select a *pivot* in each recursive stage by taking a sample of three elements and using the median of the sample as the pivot. The idea is that it is more likely that no subarray is degenerate after the partitioning [2, 15, 20].

This variant is easily generalized to samples of size $s = 2k + 1$ elements, so that the $(k+1)^{\text{th}}$ element in the sample is selected as the pivot. Van Emden [21] analyzed this generalized variant of quicksort using wise information-theoretic arguments, showing that the average number of comparisons made to sort an array of size n is $q(k) \cdot n \ln n + o(n \log n)$, where the coefficient $q(k)$ steadily decreases with k, from $q(0) = 2$ to $q(\infty) = 1/\ln 2$. Thus, if the sample size is large enough, the main term in the average number of comparisons made by quicksort is as close to the theoretic optimum $n \log_2 n$ as desired. (From a practical standpoint, though, many authors recommend using samples of 3 or 5 elements only.) A bit more surprising is that the median–of–three strategy also improves the performance of quickselect. Quite recently, it has been shown that quickselect with median–of–three yields significant savings over the standard quickselect algorithm [1, 9].

Ideally, using large samples helps, but there are two problems. The first one is easily exemplified. Assume that we take $s = 1001$. This is a large sample size, indeed. But, what do we do if the size of the (sub)array to be sorted is $n < s$? Two immediate solutions (although none of them really good unless s

were small) are to pick the elements of the sample with replacement or to take samples of size $\min\{s, n\}$.

The second problem is also intuitively clear, but much more serious. As the size of the sample increases, we need to invest more resources to compute the median of the samples. Thus the savings achieved by using large samples can easily get lost in practice. The time spent in finding the median of the samples shows up in larger lower order terms of the average performance, growing with the sample size, so that they cannot be disregarded unless n is impractically large.

As far as we know, McGeoch and Tygar [12] were the first authors that analyzed the performance of quicksort with sampling when the size of the samples is not fixed, but grows as a function of n, the size of the (sub)array to be sorted. They considered several such strategies, and proved that using samples of size $\Theta(\sqrt{n}\,)$ is the best strategy among a class of possible strategies. This class includes fixed-size sampling and hybrid strategies, called two-tier, which use first a sample whose size is a function of the size of the array, and fixed-size samples in subsequent recursive calls.

In their analysis, McGeoch and Tygar took into account the comparisons made during partitioning stages and the comparisons needed to find the median of the samples. This is important, as we have already discussed before. We should point out that previous analyses did not deal with the comparisons needed to select the pivot, on the basis that the samples were of fixed size and hence, finding the median of the sample would only contribute some additional constant number of comparisons at each stage. Unfortunately, the interesting work of McGeoch and Tygar provided some partial answers and left open other important questions.

The fundamental question studied in this paper is to find the optimal value of the sample size s as a function of n, taking into account both comparisons and exchanges. We will study the general situation where we pick the $(p+1)^{\text{th}}$ element in the sample, $0 \leq p < s$, not necessarily the median of the sample. We will find that, if only comparisons are considered, the best choice is to select the median, a result already proved by Sedgewick [17] for quicksort. But if both comparisons and exchanges are taken into account, the best choice for p will depend, in the case of quicksort, on the ratio between the cost of a single exchange and the cost of a single comparison.

The basic tools for our analysis are the continuous master theorem for divide-and-conquer recurrences (CMT, for short) and several related results [13] (see also [14]). The CMT provides almost automatically all the results for quickselect and quicksort with fixed-sized sampling — some of which were already known after Van Emden's paper. These tools also allow us to prove that if the size of the sample s grows with n and is not linear, i.e. $s = \omega(1)$ and $s = o(n)$, then the average number of comparisons made by quicksort is $n \log_2 n + o(n \log n)$ and the average number of exchanges is $\frac{1}{4} n \log_2 n + o(n \log n)$. For quickselect (when selecting a random item) the average number of comparisons and exchanges are $2n + o(n)$ and $n/2 + o(n)$, respectively (Theorems 6.1 and 5.1).

The paper is organized as follows. In Section 2, we briefly recall how the algorithms work and analyze the average number of comparisons and exchanges needed to partition an array of size n, when the pivot is selected as the $(p+1)^{\text{th}}$ element of a sample of size s. This analysis is needed in subsequent sections of the paper, since partition is the common building block of both algorithms. Care is taken not to make any assumption about s nor about p in the analysis, since we consider that s is, in general, a function of n, and p a function of s. In that section, we also set up all the recurrences for the quantities of interest.

Sections 3 and 4 are devoted to the analysis of the average total cost of quickselect and quicksort with fixed-size samples, respectively. We start with the analysis of quickselect since it turns out to be slightly simpler than the analysis of quicksort. Afterwards, in Sections 5 and 6 we tackle the study of the variants where the samples are of increasing size.

Considering the lower order terms, we prove in Section 5 that the optimal sample size for quickselect is $s^* = \Theta(\sqrt{n}\,)$ and give an explicit formula for the constant factor of the main term, which depends on the cost of the algorithm used to select pivots (which may or may not be quickselect) and on the cost of elementary comparisons and exchanges.

In Section 6, we use similar techniques to analyze quicksort, and prove that the optimal sample size is $s^* = \Theta(\sqrt{n}\,)$ if we consider only comparisons. We also find the constant factor of the main term in s^*. Our results state that $\sqrt{n}$-sampling is optimal for quicksort with respect to comparisons, if we assume that the selected pivots are the medians of the samples. In [12] it was conjectured that the optimal sample size w.r.t. the number of comparisons in quicksort is $\Theta(n^{0.56})$, by curve-fitting of the exact optimal sample sizes for various values of n. These exact values were found by a dynamic-programming algorithm. Our results disprove this conjecture. Our formula —which is $s^* = \Theta(\sqrt{n}\,)$— perfectly fits the exact optimal values given there (see Figure 8).

In Sections 4 and 6, we also find and precisely quantify the surprising behavior that arises when the average total cost of quicksort is considered, not just its average number of comparisons. In this setting, the optimal sample size is still $s^* = \Theta(\sqrt{n}\,)$, but the best choice for p is not always $p = \lfloor (s-1)/2 \rfloor$: it depends on the relative cost of an exchange with respect to that of a comparison. Moreover, if we are systematically selecting as pivots the medians of the samples, then the optimal sample size is $\Theta(\sqrt{n}\,)$ when an exchange is not expensive, but if the cost of an exchange is more than (roughly) 10 times the cost of a comparison, then the optimal sample size is constant.

The last section discusses the effect of sampling in the variance of the cost of quickselect. We show that using fixed-size samples reduces the constant factor in the main term of the variance of quickselect (which is still $\Theta(n^2)$). Another important result, with obvious implications in practical terms, is that if the samples are of increasing size then the variance of quickselect is $o(n^2)$, namely $\Theta(\max\{n \cdot s, n^2/s\})$. For quicksort, we have not been able to carry out the corresponding analysis, but we conjecture that similar results hold. In particular, it would be very interesting to elucidate whether samples of increasing size reduce the order of magnitude of the variance of quicksort.

In the last section we also discuss tuned implementations of partitioning, which avoid making redundant comparisons and exchanges. The results for the tuned variants of quicksort and quickselect are basically the same as for the non-tuned variants examined in the previous sections.

A final appendix contains the proofs of several combinatorial identities, necessary for the computations made in previous sections.

2 Sampling

For the sake of completeness, we give in this section a brief description of the algorithms. We will assume that the input to the algorithms is a (sub)array of $n > 0$ distinct integers. Furthermore, we shall assume —as is usual in the analysis of comparison-based sorting algorithms— that each permutation of the n items is equally likely.

Quicksort sorts a subarray $A[l..u]$, with $u-l+1 = n$, as follows. If $l \geq u$, the subarray contains zero or one element and hence it is already sorted. Otherwise, we choose some *pivot* v among the elements of the subarray, and partition around this selected element. The pivot can be a particular element of the subarray (chosen or not at random), the median of a (small) sample of elements taken from the subarray, etc. For the rest of the paper we will assume that the pivot is selected from a sample of size s, $s \leq n$, consisting of the first s elements in the subarray. Notice that for $s = 1$ we have standard quicksort.

Partitioning means that the subarray is rearranged in a way such that there exists some position j such that $l \leq j+1 \leq u$, all elements in $A[l..j]$ are smaller than v, all elements in $A[j+2..u]$ are larger than v, and $A[j+1] = v$. Once the subarray has been partitioned, the procedure is called recursively to sort the subarrays $A[l..j]$ and $A[j+2..u]$.

There are several ways to efficiently partition a subarray, but some care must be taken to guarantee that the partitioning algorithm preserves randomness. One of the most common partitioning algorithms works as follows: first, it swaps the pivot with the element in $A[l]$. Then, it initializes two pointers $\mathit{left} := l+1$ and $\mathit{right} := u$. The left pointer scans the subarray from left to right, until an element larger than the pivot is found. After that, the right pointer scans the subarray from right to left, until it reaches an element smaller than the pivot. The elements pointed to by the pointers are then exchanged, and the procedure is repeated until the pointers meet. If $\mathit{right} = j+1$ and $\mathit{left} = j+2$ at that moment, then $A[l+1..j+1]$ contains elements smaller than the pivot and $A[j+2..u]$ contains elements larger than the pivot. A final swap between $A[l]$ and $A[j+1]$ yields the desired arrangement.

Quickselect uses similar ideas to those used in quicksort in order to select the m^{th} element out of n, with $1 \leq m \leq n$. Given a subarray with n elements that contains the m^{th} element, we begin partitioning the subarray as in quicksort. Let $j+1$ be the rank of the pivot among the n elements. If $m = j+1$ then we have found the sought element, and return the pivot. If $m < j+1$, the element we want to select must be in the subarray to the left of the pivot, so we make a recursive call to the procedure on that subarray; otherwise, $m > j+1$ and we must recursively continue the process in the subarray to the right, but now looking for the $m-j-1^{\mathrm{th}}$ element.

From now on, we assume that s, the size of the sample, is a function of the size of the subarray to be sorted. For convenience, we will use the letter n to denote the size of the current subarray, in a slight abuse of notation as n also denotes the size of the whole array. We will not write down the dependence of s on n most of the times, though.

Specifically, we consider that $s(n) = \Omega(1)$ and $s(n) = o(n)$. This implies that, for all but finitely many values of n, the inequality $s(n) \leq n$ is satisfied. This assumption includes the situation where the size of the sample is constant, $s = \Theta(1)$.

In our analysis, we also assume that the selected pivot is the $(p+1)^{\mathrm{th}}$ element $(0 \leq p < s)$ in the sample. We will denote $q = s-1-p$ the number of elements in the sample larger than the pivot. For the particular case where the sample is of odd size and the selected pivot is the median of the sample, we write $s = 2k+1$, and hence $k+1$ is the rank of the pivot, with $p = q = k$.

The average performance of quicksort and quickselect ultimately relies on the probability of (un)even partitioning. The following well known proposition given below will be used in all subsequent computations.

Proposition 2.1 *Let $\pi_{n,j}^{(s,p)}$ be the probability that the $(p+1)^{th}$ element of a sample of s elements, with $0 \leq p < s$, is the $(j+1)^{th}$ element of a random permutation of size n, where $0 \leq j < n$. Then*

$$\pi_{n,j}^{(s,p)} = \frac{\binom{j}{p}\binom{n-1-j}{q}}{\binom{n}{s}}.$$

The denominator of the right-hand side of the equation above is the number of ways to pick a sample of size s out of n elements; the numerator is the number of ways to choose p elements smaller than the pivot times the number of ways to choose $q = s-1-p$ elements larger than the pivot. Note that for the plain variants of the algorithms (that is, $s = 1$ and $p = 0$) we have $\pi_{n,j}^{(1,0)} = 1/n$ for all $0 \leq j < n$.

We consider now the average total cost of the non-recursive parts of quicksort and quickselect, which include the comparisons and exchanges made to select the pivot from the sample, plus the comparisons and exchanges to partition the array around the selected pivot.

Let $\mathsf{C}(n,s,p)$ denote the average number of comparisons to partition an array of size n when the sample has s elements. Analogously, let $\mathsf{X}(n,s,p)$ denote the average number of exchanges made by the partition. Recall that we assume that the array contains a random permutation of the n distinct elements.

The number of comparisons is $\mathsf{C}(n,s,p) = n+1$ irrespective of s and p, since the pivot must be compared with every other element in the array, and two additional comparisons are performed when the left and right pointers meet.

The following lemma gives the value of $\mathsf{X}(n,s,p)$ for any n, s and p, provided that $0 \le p < s \le n$. Its proof is given in Appendix A at the end of this paper. In that proof, we assume that the subarray to be partitioned contains the pivot in its first position followed by a random permutation of the $n-1$ remaining elements.

Lemma 2.1 *The average number of exchanges to partition a random array of size n when the pivot is the $(p+1)^{th}$ element of a sample of s elements is*

$$\mathsf{X}(n,s,p) = \frac{1}{(s+1)(s+2)(n-1)}\Big((p+1)(q+1)n^2 \\ - ((p+1)^2 + (q+1)^2 - (p+1)(q+1) + s + 1)n + 2(p+1)(q+1)\Big),$$

where $q = s - 1 - p$.

Note that if the selection of the pivots is made in-place, then the first s components of A would be arranged by the selection procedure and then $A[2..n]$ (in particular, $A[2..s]$) would probably not be a random permutation, after the selection of the pivot is performed. Hence, the computation of $\mathsf{X}(n,s,p)$ above is valid if and only if we assume that the selection of the pivot is performed in a separate area containing a copy of the s elements of the sample or we prove that the selection mechanism preserves randomness. However, it seems that, apart from brute-force solutions, no such selection mechanism exists for general s. (In [17], Sedgewick thoroughly discusses the benefits that the partition procedure is randomness-preserving —the selection of the pivot included as a part of that procedure—, and the dangers of partitioning strategies that do not have that property.)

Let $\mathsf{S}(s,p)$ denote the average total cost of the algorithm that selects the $(p+1)^{\mathrm{th}}$ out of s elements (this algorithm may or may not be quickselect or one of its variants). Efficient selection algorithms work in linear time, at least on the average, so we may safely assume $\mathsf{S}(s,p) = \beta \cdot s + o(s)$ for some constant β that depends on the chosen algorithm, the unitary costs of a comparison u_c and of an exchange u_x, and typically on the ratio $(p+1)/(s+1)$.

For example, the expected number of comparisons to select the median with standard quickselect is $2(1+\ln 2)s + o(s)$. Recall [3, 4] that the expected number of comparisons to select the $(p+1)^{\mathrm{th}}$ out of s elements is bounded by below by

$$\left(\frac{3}{2} - \left|\frac{1}{2} - \frac{p+1}{s+1}\right|\right) s + o(s),$$

and hence if $p = \lfloor (s-1)/2 \rfloor$) then $\beta \ge 1.5 \cdot u_c$. On the other hand, we could include the cost to copy the sample in a separate area in the total cost $\mathsf{S}(s,p)$, and this cost would still be linear in s. We will not do so.

We now set up the recurrences for the average total cost of quicksort and quickselect, valid for n large enough. In the recurrence for quickselect we assume that the rank m of the sought element is any value in the range $1 \ldots n$ with equal probability.

Lemma 2.2 *Let Q_n be the average total cost of quicksort to sort an array of size n when we choose as the pivot of each partitioning stage the $(p+1)^{th}$ element of a sample of s elements. Then*

$$Q_n = \mathsf{S}(s,p) + \mathsf{C}(n,s,p)\cdot u_c + \mathsf{X}(n,s,p)\cdot u_x + \sum_{0\le j<n} \left(\pi^{(s,p)}_{n,j} + \pi^{(s,p)}_{n,n-1-j}\right) Q_j.$$

Lemma 2.3 *Let F_n be the average total cost of quickselect to select a random element from an array of size n when we choose as the pivot of each partitioning stage the $(p+1)^{th}$ element of a sample of s elements. Then*

$$F_n = \mathsf{S}(s,p) + \mathsf{C}(n,s,p)\cdot u_c + \mathsf{X}(n,s,p)\cdot u_x + \sum_{0\le j<n} \frac{j}{n}\left(\pi^{(s,p)}_{n,j} + \pi^{(s,p)}_{n,n-1-j}\right) F_j.$$

Finally, we define $\xi = u_x/u_c$ to be the cost of an exchange relative to that of a comparison. For the rest of the paper we will set $u_c = 1$, and thus $u_x = \xi$; in order words, we will measure the total cost in number of comparisons, where an exchange is worth ξ comparisons. By setting $\xi = 0$, we will be only considering the number of comparisons. We will also assume that $\mathsf{S}(s,p)$ is measured using as unit the cost of a comparison.

3 Quickselect with Fixed-Size Samples

In this section we compute F_n, the expected cost of quickselect when the size s of the samples is fixed. This analysis is almost straightforward using the CMT [13].

For n large enough the *toll function*, that is, the non-recursive cost in the recurrence for F_n, is

$$\left(1 + \frac{(p+1)(q+1)}{(s+1)(s+2)}\cdot\xi\right) n + \mathcal{O}(1). \tag{1}$$

Notice that $\mathsf{S}(s,p)$ and several other terms are accounted for in the $\mathcal{O}(1)$-term above, because we assume $s = \mathcal{O}(1)$.

The first step to apply the CMT requires that we compute the *shape function* for this problem; recall that the shape function is essentially a continuous approximation to the discrete weights, which in our instance are (see Lemma 2.3)

$$\omega^{(s,p)}_{n,j} = \frac{j}{n}\left(\pi^{(s,p)}_{n,j} + \pi^{(s,p)}_{n,n-1-j}\right) = \frac{j}{n}\left(\pi^{(s,p)}_{n,j} + \pi^{(s,q)}_{n,j}\right). \tag{2}$$

It is not difficult to show that

$$\omega^{(s,p)}(z) = \frac{s!}{p!\,q!}\left(z^{p+1}(1-z)^q + z^{q+1}(1-z)^p\right) \tag{3}$$

is the shape function in this case, and then we can compute the limiting const-entropy $\mathcal{H}(s,p)$ associated to this problem (in [13], the const-entropy is denoted $\mathcal{H}_n^{(1)}$ and its limit by $\mathcal{H}^{(1)}$). This entropy is given by $\mathcal{H}(s,p) = 1 - \varphi(1)$, where

$$\begin{aligned}\varphi(x) &= \int_0^1 \omega^{(s,p)}(z)\, z^x \, dz \\ &= \frac{s!}{p!\, q!} \int_0^1 \left(z^{p+1+x}(1-z)^q + z^{q+1+x}(1-z)^p \right) dz.\end{aligned}$$

Note that $\varphi(x)$ is evaluated at $x = 1$ because the toll function is linear.

We use the equality

$$\int_0^1 z^\alpha \, (1-z)^\beta \, dz = \frac{\alpha!\beta!}{(\alpha+\beta+1)!} \tag{4}$$

to deduce

$$\mathcal{H}(s,p) = 1 - \frac{(p+1)(p+2)+(q+1)(q+2)}{(s+1)(s+2)} = \frac{2(p+1)(q+1)}{(s+1)(s+2)} > 0,$$

for every s and p. Therefore,

$$\begin{aligned}F_n &= \left(1 + \frac{(p+1)(q+1)}{(s+1)(s+2)} \cdot \xi \right) \frac{n}{\mathcal{H}(s,p)} + o(n) \\ &= \left(\frac{(s+1)(s+2)}{2(p+1)(q+1)} + \frac{\xi}{2} \right) n + o(n).\end{aligned}$$

Let

$$f_\xi(s,p) = \frac{(s+1)(s+2)}{2(p+1)(q+1)} + \xi/2$$

denote the coefficient of the main term in F_n, i.e. the constant multiplying n. We can get a better estimate of the lower order terms of F_n, as the recurrence for $G_n = F_n - f_\xi(s,p) \cdot n$ is easily computed and amenable to analysis through the CMT. Indeed,

$$\begin{aligned}G_n = \mathsf{S}(s,p) + \mathsf{C}(n,s,p) + \mathsf{X}(n,s,p) \cdot \xi + \sum_{0 \le j < n} \omega_{n,j}^{(s,p)}\, G_j \\ - f_\xi(s,p) \cdot n + f_\xi(s,p) \cdot \sum_{0 \le j < n} \omega_{n,j}^{(s,p)}\, j.\end{aligned}$$

First of all, we seek for a closed form of the last term in the formula above. It follows from the next proposition.

Proposition 3.1 *For all s and p such that $0 \le p < s$,*

$$(s+1)(s+2) \sum_{0 \le j < n} j^2 \left(\pi_{n,j}^{(s,p)} + \pi_{n,n-1-j}^{(s,p)} \right) = \\ \left((p+1)(p+2) + (q+1)(q+2) \right) n^2 \\ - 6(p+1)(q+1)n + (p-q)^2 + (s+1),$$

where $q = s - 1 - p$.

Proof. The proposition is a simple consequence of

$$\sum_{0 \le j < n} j^{\underline{\ell}} (n-1-j)^{\underline{m}} = \ell!\; m! \binom{n}{\ell + m + 1},$$

where $x^{\underline{\ell}} = x \cdot (x-1) \cdots (x - \ell + 1)$ denotes the ℓ^{th} falling factorial of x (see Proposition A.2 in Appendix A), and of the equality $j^2 = (j-x)(j-1-x) + (2x+1)(j-x) + x^2$, that we apply twice, with $x = p$ and with $x = q$. □

Using the proposition above, a few additional manipulations yield that the toll function in the recurrence for G_n is

$$\mathsf{S}(s,p) - 2 - \frac{(q+1)^2 + 7(q+1) + s(p-1) + 2}{(s+1)(s+2)} \cdot \xi + \Theta\left(\frac{1}{n}\right), \tag{5}$$

which altogether is $\Theta(1)$, if the positive and negative terms above do not cancel out. The weights for G_n are the same as for F_n and so is the shape function. As the toll function is $\Theta(1)$, the limiting const-entropy is now $\mathcal{H}(s,p) = 1 - \varphi(0) = 0$, and we are led to compute the limiting log-entropy (denoted $\mathcal{H}^{(\ln)}$ in [13]), which is

$$\mathcal{H}'(s,p) = -(c+1) \int_0^1 \omega^{(s,p)}(z)\, z^b \,\ln z \, dz,$$

with $b = c = 0$, as the toll function is $\Theta\left(n^0 (\ln n)^0\right)$. Using the equality

$$-\int_0^1 z^\alpha \,(1-z)^\beta \,\ln z \, dz = \frac{\alpha!\beta!}{(\alpha+\beta+1)!} (H_{\alpha+\beta+1} - H_\alpha) \tag{6}$$

(which can be found using some computer algebra system, and proved by induction, for instance), we find

$$\mathcal{H}'(s,p) = H_{s+1} - \frac{(p+1)H_{p+1} + (q+1)H_{q+1}}{s+1}, \tag{7}$$

where H_n denotes the n^{th} harmonic number. Recall that the harmonic numbers admit the asymptotic expansion $H_n = \ln n + \gamma + o(1)$, where $\gamma = 0.577\ldots$ is Euler's constant.

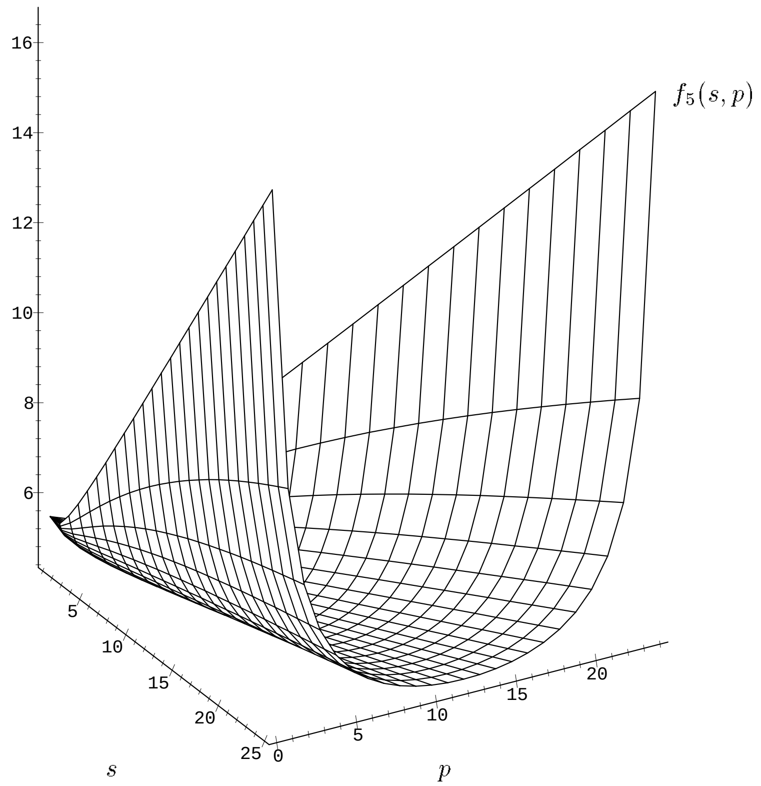

Figure 1: Values of $f_5(s, p)$.

The function in the right-hand side of Equation (7) appears often enough in our analysis that it is worth naming it. For any positive integers p and q, if $s = p + q + 1$ then

$$\mathrm{VE}(s,p) = H_{s+1} - \frac{(p+1)H_{p+1} + (q+1)H_{q+1}}{s+1}.$$

The name of this function follows after Van Emden, who first used it for the particular case where $p = q$ [21].

Once we have computed the log-entropy, it follows

$$G_n = \left(\mathsf{S}(s,p) - 2 - \frac{(q+1)^2 + 7(q+1) + s(p-1) + 2}{(s+1)(s+2)} \cdot \xi \right) \frac{\ln n}{\mathrm{VE}(s,p)} + o(\log n).$$

Again, we could get more information, defining $I_n = G_n - g_\xi(s,p) \cdot H_n$, where $g_\xi(s,p)$ is the constant factor of the main order term of G_n. Notice that in this case we use the n^{th} harmonic number, H_n, instead of the factor $\ln n$ as the main order term of G_n. This trick does not introduce any asymptotically significant difference, but makes the computations much easier. We shall not give those computations here, but in this case the toll function in the recurrence for I_n is $\mathcal{O}\left(n^{-1}\log n\right)$. Therefore, $I_n = \mathcal{O}(1)$, and no further information can be extracted using the CMT.

Gluing together all the pieces, we get the following theorem.

Theorem 3.1 *The average total cost of quickselect with fixed-size samples is*

$$\begin{aligned} F_n = &\left(\frac{(s+1)(s+2)}{2(p+1)(q+1)} + \frac{\xi}{2}\right) n \\ &+ \left(\mathsf{S}(s,p) - 2 - \frac{(q+1)^2 + 7(q+1) + s(p-1) + 2}{(s+1)(s+2)} \cdot \xi\right) \frac{\ln n}{VE(s,p)} \\ &+ \mathcal{O}(1). \end{aligned}$$

We now consider which are the values s and p that minimize $f_\xi(s,p)$. Clearly, for any fixed s, the value of p that minimizes $f_\xi(s,p)$ is $p = (s-1)/2$ if s is odd, and $p = (s-2)/2$ or $p = s/2$ if s is even. Moreover, for any $k \geq 0$,

$$f_\xi(2k+1, k) = f_\xi(2k+2, k) = 2 + \frac{1}{k+1} + \frac{\xi}{2} \tag{8}$$

is a decreasing function of k. Hence, the minimum of $f_\xi(s,p)$ is achieved if we let s be as large as possible, and we choose the median of the sample as the pivot (see Figure 1).

As we have already mentioned in the introductory section, taking very large samples irrespective of n is not good for practical purposes. In Section 5 we will see how we can do better by taking samples whose size grows with n.

4 Quicksort with Fixed-Size Samples

The analysis of quicksort with samples of fixed size is almost identical to that of quickselect in Section 3. Hence, we skip most intermediate steps, for the sake of brevity.

The shape function is now

$$\omega^{(s,p)}(z) = \frac{s!}{p!\,q!}\left(z^p(1-z)^q + z^q(1-z)^p\right). \tag{9}$$

Thus, we define

$$\varphi(x) = \frac{s!}{p!\,q!}\int_0^1 \left(z^{p+x}(1-z)^q + z^{q+x}(1-z)^p\right)\,dz,$$

and conclude that, as the toll function is linear, the limiting const-entropy is $\mathcal{H}(s,p) = 1 - \varphi(1) = 0$ for every s and p. Hence, we need to compute the limiting log-entropy

$$\mathcal{H}'(s,p) = -\frac{s!}{p!\,q!}\int_0^1 \left(z^{p+1}(1-z)^q + z^{q+1}(1-z)^p\right)\ln z\,dz.$$

We immediately find $\mathcal{H}'(s,p) = \mathrm{VE}(s,p)$. As a conclusion we have

$$Q_n = q_\xi(s,p)\,n\ln n + o(n\log n), \tag{10}$$

where

$$q_\xi(s,p) = \frac{1 + \frac{(p+1)(q+1)}{(s+1)(s+2)}\cdot\xi}{\mathrm{VE}(s,p)}. \tag{11}$$

Setting $s = 2k+1$, $p = q = k$ and $\xi = 0$, we recover the original result of Van Emden [21].

It is possible to get more information about Q_n by subtracting the main order term from Q_n and setting up a recurrence for the remaining part, i.e., for $R_n = Q_n - q_\xi(s,p)\,nH_n$. Following the same steps as when analyzing quickselect with fixed-size samples, we get $R_n = \mathcal{O}(n)$. We now summarize our findings in the following theorem.

Theorem 4.1 *The average total cost of quicksort with fixed-size samples is*

$$Q_n = \frac{1 + \frac{(p+1)(q+1)}{(s+1)(s+2)}\cdot\xi}{VE(s,p)}\,n\ln n + \mathcal{O}(n).$$

Let us shift our attention now to the constant factor $q_\xi(s,p)$ and study the values of s and p that minimize it. If we only take into account comparisons ($\xi = 0$), then the best choice for p is $\lfloor (s-1)/2 \rfloor$. As with quickselect, $q_\xi(2k+1,k)$ is a strictly decreasing function of k, and hence we should get samples as large as possible and select the medians of the samples as pivots. (This conclusion, in fact an even stronger statement, was already proved by Sedgewick [17]. He considered the case where each element of the sample could be selected as the pivot with a given probability, and showed that the best choice is to select the median with probability 1.) For moderately large values of ξ, like $\xi = 5$, nothing surprising occurs and the same conclusions hold (see Figure 2).

But if ξ is large, using as pivot the median of the sample is no longer the best choice. The best fixed-size strategy is certainly to have samples as large as possible, but the pivot should be the element of rank $p^* = p^*(s,\xi)$ in the sample, where p^* is the value that minimizes $q_\xi(s,p)$. It turns out that $p^* \neq \lfloor (s-1)/2 \rfloor$ when ξ is large (roughly, greater than 10).

For convenience, we will work with the quotients $\psi = \lim_{s\to\infty}(p/s)$ rather than with the actual ranks p for the rest of this section. In Figure 3 we can see that for $\xi = 30$ there are two valleys symmetrically disposed at both sides of the line defined by $\psi = 1/2$. In both valleys, $q_\xi(s,p)$ is identical and minimum, and we take the one with smallest p to define ψ^*, which is clearly smaller than $1/2$. What we have discussed so far could be then expressed by saying that ψ^* equals $1/2$ for moderate values of ξ, and ψ^* is smaller than $1/2$ if ξ is large.

Let

$$q_\xi(\psi) = -\frac{1+\psi(1-\psi)\xi}{\psi\ln\psi + (1-\psi)\ln(1-\psi)} \tag{12}$$

for $0 < \psi < 1$. It is easy to see that $\lim_{s\to\infty} q_\xi(s, \psi\cdot s + o(s)) = q_\xi(\psi)$.

In Figure 4 we show the (scaled) values of $q_\xi(\psi)$ for $\xi = 0$, 5, 15, 20 and 30. As shown in the figure, $q_\xi(\psi)$ is symmetric around $\psi = 1/2$, with only one minimum located at $\psi^* = 1/2$ when ξ is smaller than or equal to the threshold value τ, whereas if ξ is larger than τ we have a local maximum at $\psi = 1/2$ and two absolute minima at $\psi^* < 1/2$ and at $1-\psi^* > 1/2$, respectively. In that last case, the optimal ψ^* is the unique solution of the equation $\partial q_\xi(\psi)/\partial\psi = 0$ in the interval $(0, 1/2)$, or equivalently, the unique solution in $(0,1/2)$ of

$$\ln\psi + \xi\,\psi^2\ln\psi = \ln(1-\psi) + \xi(1-\psi)^2\ln(1-\psi). \tag{13}$$

The value τ is given by the solution of the equation

$$\left.\frac{\partial^2 q_\xi(\psi)}{\partial\psi^2}\right|_{\psi=1/2} = 0,$$

which is $\tau = 4/(2\ln 2 - 1) \approx 10.35480$. In Figure 5 we can see that $\psi^*(\xi) = 1/2$ for $\xi \in [0,\tau]$, and $\psi^*(\xi)$ strictly decreases for $\xi > \tau$, tending to 0 as ξ grows.

If s is large enough then the optimal value of p is $p^* \approx \psi^*\cdot s$. But now the question is whether taking large samples is the best choice or not. Informally speaking, the next theorem states just that.

Theorem 4.2 *For any s and p, $q_\xi(s,p) > q_\xi(\psi^*)$.*

Proof. Let

$$u(z) = \frac{s!}{p!\,q!} z^p (1-z)^q.$$

Then, because of Equations (4), (6) and (11), we have

$$q_\xi(s,p) = \frac{\int_0^1 u(z)\,(1+z(1-z)\xi)\ dz}{\int_0^1 u(z)\,(-z\ln z - (1-z)\ln(1-z))\ dz}.$$

We will use the following fact: Let $f(z)$, $g(z)$ be positive functions over the interval $[0,1]$. Let z^* be the location of a minimum of $f(z)/g(z)$, and assume

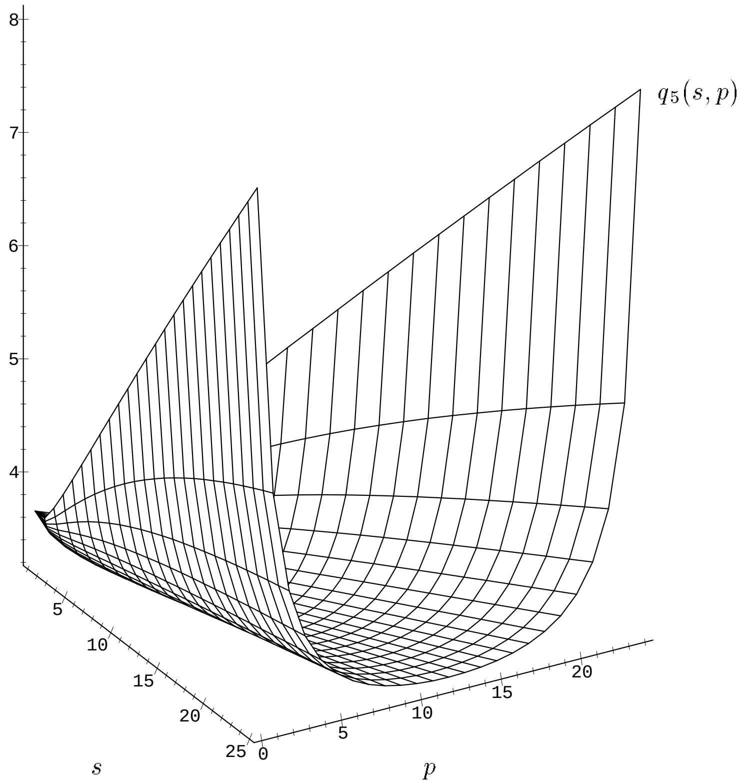

Figure 2: Values of $q_5(s,p)$.

that $0 < z^* < 1$ and $g(z) \neq 0$ for $0 < z < 1$. Then, as $f(z) \geq g(z)f(z^*)/g(z^*)$ and $u(z)$ is also positive in the interval $[0,1]$ we have

$$\frac{\int_0^1 u(z)f(z)\,dz}{\int_0^1 u(z)g(z)\,dz} \geq \frac{\int_0^1 u(z)\,[g(z)f(z^*)/g(z^*)]\;dz}{\int_0^1 u(z)g(z)\,dz} = \frac{f(z^*)}{g(z^*)}.$$

Now taking $f(z) = 1 + z(1-z)\xi$, $g(z) = -z\ln z - (1-z)\ln(1-z)$, which satisfy the assumptions, and since ψ^* is the minimum of $q_\xi(\psi) = f(\psi)/g(\psi)$, we have $q_\xi(s,p) \geq q_\xi(\psi^*)$, and the statement of the theorem is almost proved.
Finally, notice that for any ξ there is always an interval $[\psi_1, \psi_2]$ with $0 \leq \psi_1 < \psi_2 \leq 1$ such that $q_\xi(\psi) > q_\xi(\psi^*(\xi))$ and $u(z) > 0$ for every ψ in (ψ_1, ψ_2). Thus $q_\xi(s,p) > q_\xi(\psi^*)$.

$\square$

An intuitive explanation for the theorem above goes as follows. Assume that we had a black-box routine such that, given an array of size n and a value

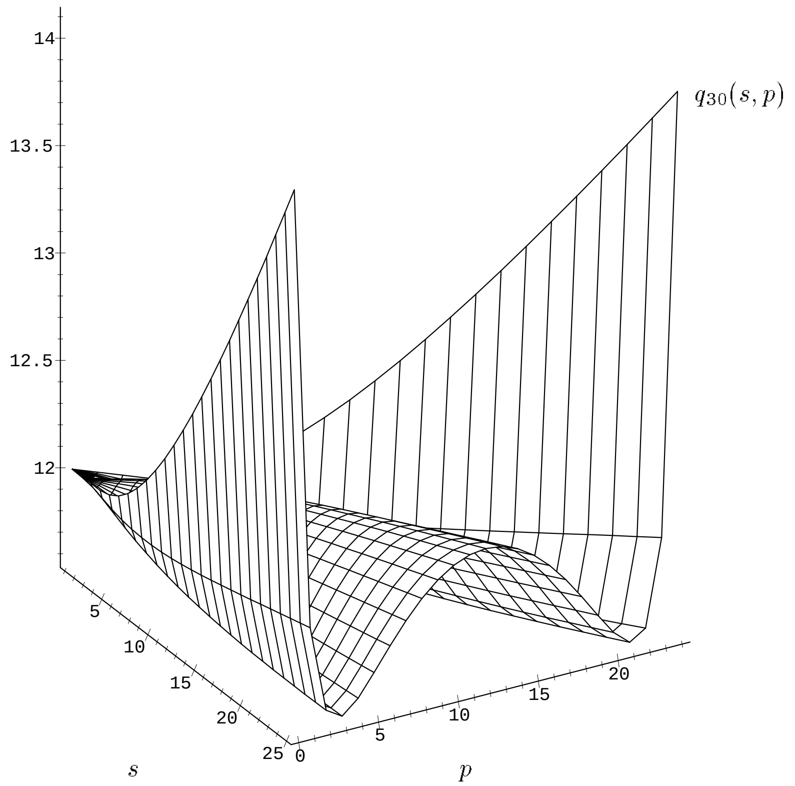

Figure 3: Values of $q_{30}(s,p)$.

$0 < \psi < 1$, it returned the $\lfloor \psi \cdot n \rfloor^{\mathrm{th}}$ element in the array *at no cost*. We could make use of such a black-box routine (with some fixed ψ) in the pivot-finding stage of quicksort. Then it is not difficult to prove that the toll function of this variant of quicksort would be $(1 + \psi(1-\psi)\xi)$, and its log-entropy $-\psi \ln \psi - (1-\psi)\ln(1-\psi)$. Hence, we conclude that the average total cost of this variant would be $q_\xi(\psi)\, n \ln n + o(n \log n)$. For this variant, it is clear that the optimal choice for ψ is ψ^*.

Now, as we are deprived from such a black-box, we should try to get the best possible estimate of the $\lfloor \psi^* \cdot n \rfloor^{\mathrm{th}}$ element in the array to use it as the pivot, which can be done by taking large samples and selecting the $\lfloor \psi^* \cdot s \rfloor^{\mathrm{th}}$ element of the sample.

Several remarks are in order. On the one hand, the fact that ψ^* tends to 0 as ξ tends to ∞ can be informally described as a smooth transition from quicksort to selection sort. Notice that if we always select the smallest element in the array as the pivot, then quicksort behaves exactly as selection sort. And if

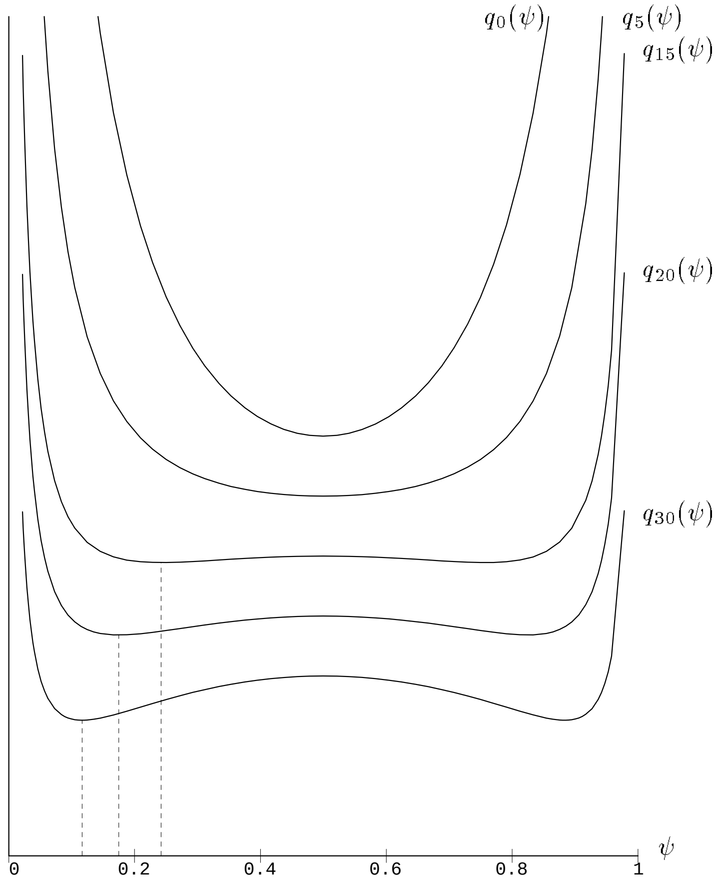

Figure 4: Values of $q_\xi(\psi)$ for $\xi = 0$, 5, 15, 20 and 30.

exchanges are very expensive then selection sort is a good choice, as it minimizes the number of exchanges to sort the array.

On the other hand, we should be aware that the analysis of the case $\xi > \tau$ is mainly of theoretical interest since in most practical situations we have $\xi \leq \tau$. If data movements were too expensive, we would sort an array of pointers to the actual records rather than sorting the records themselves.

Now we restrict our attention to the variants of quicksort that always take the median of a sample of fixed size as the pivot, irrespective of the value of ξ (and therefore are not the best theoretical alternatives if $\xi > \tau$). That is, we take $s = 2k + 1$ with $p = q = k$, for some $k \geq 0$.

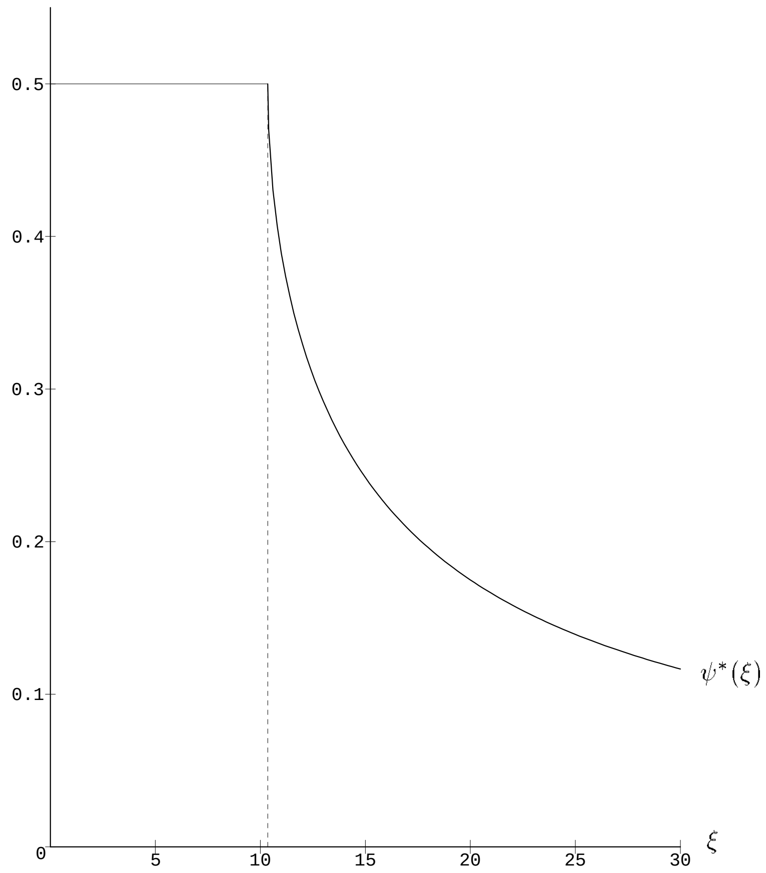

Figure 5: Values of $\psi^*(\xi)$.

This time we have

$$q_\xi(k) = q_\xi(2k+1, k) = \frac{1 + \frac{k+1}{4k+6} \cdot \xi}{\frac{1}{k+2} + \cdots + \frac{1}{2k+2}} \tag{14}$$

for every $\xi \geq 0$.[1]

The function $q_\xi(k)$ gives the constant of the main term of quicksort when we choose as pivot the median of samples of $2k+1$ elements. The (scaled) shape of this function is shown in Figure 6 for several values of ξ. For $\xi = 5$ and $\xi = 8$ the function $q_\xi(k)$ steadily decreases with k, in accordance with what we know. This behavior changes as soon as $\xi > \tau$ (dashed in the figure). For values of ξ greater than τ the function $q_\xi(k)$ has one minimum at finite distance k^*. For

[1] We overload the notation $q_\xi(\cdots)$ once again; here, $q_\xi(k) = q_\xi(2k+1, k)$.

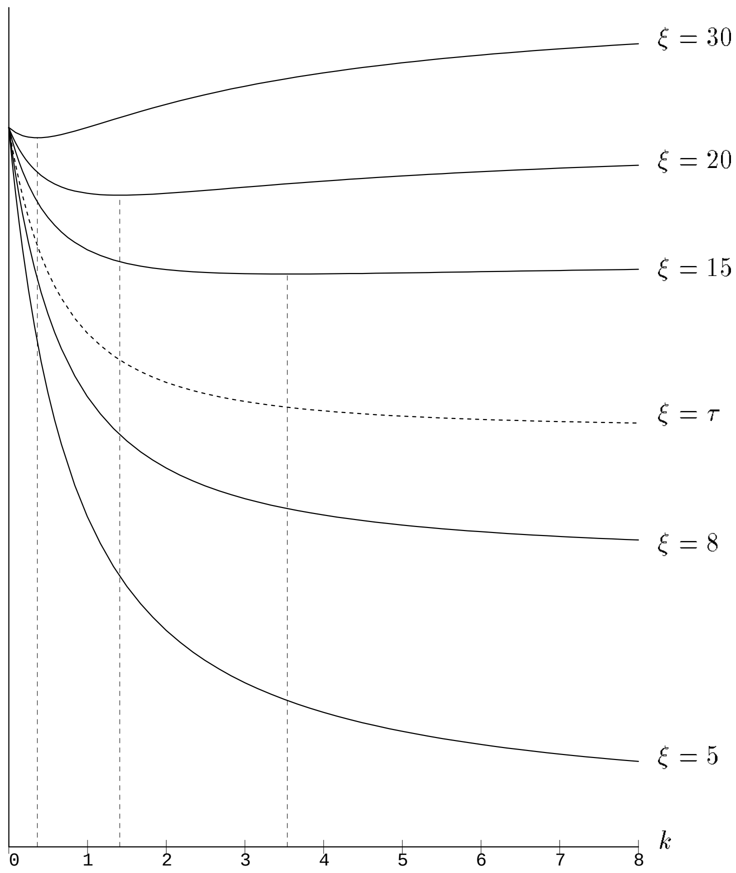

Figure 6: Values of $q_\xi(k)$ for $\xi = 5$, 8, τ, 15, 20 and 30.

instance, $k^* = 0$ if $\xi > 30$, and $k^* = 1$ if $\xi \in (20, 30)$. In Figure 6 we show the location of the minimum for $\xi = 15$, 20 and 30. The plot actually depicts the extension of $q_\xi(k)$ to the real numbers using the function $\Psi(z) = d \ln \Gamma(z)/dz$ (recall that, for positive integers, $\Psi(n+1) = H_n - \gamma$).

Notice that the location of the minima tends to 0 when ξ grows. Figure 7 shows the value of k^* as a function of ξ. There is a vertical asymptote as $\xi \to \tau^+$. For values of ξ larger than 30 we have $k^* = 0$; in other words, if we enforce choosing the median of the sample as pivot then the best alternative is plain quicksort (without sampling). Notice also that k^* is not well defined in some points. For instance, if $\xi = 20$ then $q_\xi(1) = q_\xi(2)$, and both 1 and 2 compete as optimal choices for k.

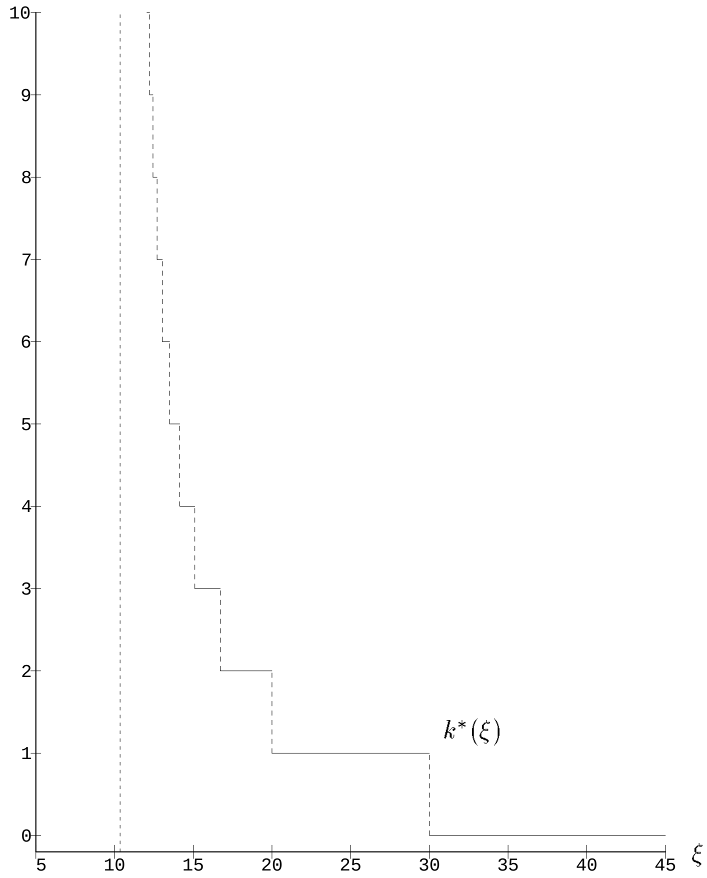

Figure 7: Values of $k^*(\xi)$.

The function

$$\xi^*(k) = \frac{4}{4(H_{2k} - H_k)\frac{k+1}{2k+3} - 1 + \frac{1}{2k+1}}$$

is the pseudoinverse of $k^* = k^*(\xi)$, in the sense that k is the optimal choice if ξ belongs to the open interval $(\xi^*(k+1), \xi^*(k))$. For instance, $\xi^*(2) = 20$ and $\xi^*(1) = 30$, and hence $k^* = 1$ when $20 < \xi < 30$. By convention, we take $\xi^*(0) = \infty$. Because of the definition, $q_{\xi^*(k)}(k) = q_{\xi^*(k)}(k-1)$ for any $k > 0$.

In fairly intuitive terms, the conclusion we have arrived at is that a good estimation of the median of the (sub)array is always profitable as far as comparisons are concerned, since the number of comparisons made while partitioning does only depend on the size of the (sub)array to be sorted. As large samples increase the probability that the selected pivot is the median of the subarray (or is close to it), it is worth making the additional effort to get a better estimation of the median. If exchanges are not taken into account or if their cost is low, the best alternative is to make s grow with n.

But from the point of view of the number of exchanges it is not so good to have a close estimate of the median. Indeed, if the pivot lies close to one of the extremes of the subarray then only a few exchanges are needed in that stage. By contrast, an uneven partition of the array means more recursive stages and more exchanges to be done in the long run. Here, we have a trade-off between the short-term profit of selecting a pivot far away from the median and the long-term gains of selecting a pivot closer to the median. So, when exchanges are expensive, we would be happy with just a good (but not too good!) estimation of the median of the array.

5 Optimal Samples for Quickselect

We already know from Section 3 that, in order to minimize the average cost of quickselect, our best choice is to select always the medians of the samples, so we will assume $s = 2k + 1$ and $p = q = k$ for the rest of this section. Also, we will sometimes simplify the notation for functions like $\mathsf{X}(n, 2k+1, k)$, writing $\mathsf{X}(n, k)$ instead.

In principle, it pays to have a sample as large as possible. The largest possible sample that makes sense is $s = n$, and it is not difficult to show that the average total cost in this case is not optimal: it is $(2 + \xi/2 + \beta)\, n + o(n)$. If $s = \Theta(n)$ the average total cost is not optimal either, since the cost of the selection of the pivot is also of the same order of magnitude as the cost of the partition. Hence, we shift our attention towards using samples whose size increases with n but are sublinear, that is, $s = \omega(1)$ and $s = o(n)$.

Under the hypotheses above, when n grows so does s, and therefore the following theorem should not be surprising.

Theorem 5.1 *Let $s = 2k+1$ and $p = q = k$, where $k = k(n)$ is a function such that $k = \omega(1)$ and $k = o(n)$. Then the average total cost of quickselect to find an element of random rank out of n is*

$$F_n = (2 + \xi/2)\, n + o(n).$$

Proof. Let $F_n(t)$ be the average cost of quickselect when we use samples of fixed size $2t+1$, and let $\pi_{n,j}^{(t)} = \pi_{n,j}^{(2t+1,t)}$. The toll function in the recurrence for $F_n(t)$ is $(1 + (t+1)/(4t+6) \cdot \xi)\, n + \mathcal{O}(1)$. Then, according to the CMT, the const-entropy associated to $F_n(t)$ is

$$\mathcal{H}_n(t) = 1 - \frac{2}{n^2} \sum_{0 \le j < n} j^2\, \pi_{n,j}^{(t)} + o(1),$$

and its limit $\mathcal{H}(t) = \lim_{n\to\infty} \mathcal{H}_n(t) = f_\xi(2t+1, t)$ (see Section 3).

Fix any $t' \geq t$. Since x^2 is a convex function, the probabilities $\pi_{n,j}^{(\cdot)}$ are symmetric in j and $n-1-j$, and the weights for t' are more concentrated around $(n-1)/2$ than the weights for t, it follows

$$\sum_{0 \leq j < n} j^2 \pi_{n,j}^{(t')} \leq \sum_{0 \leq j < n} j^2 \pi_{n,j}^{(t)},$$

and hence $\mathcal{H}_n(t') \geq \mathcal{H}_n(t)$ if n is large enough.
Under the hypotheses, the const-entropy associated to F_n is

$$\mathcal{H}_n = 1 - \frac{2}{n^2} \sum_{0 \leq j < n} j^2 \pi_{n,j}^{(k(n))} + o(1).$$

Since $k = \omega(1)$, for any fixed t we have $k(n) > t$ and $\mathcal{H}_n \geq \mathcal{H}_n(t)$, as long as n is large enough. Therefore, $\mathcal{H} = \lim_{n\to\infty} \mathcal{H}_n \geq \lim_{n\to\infty} \mathcal{H}_n(t) = \mathcal{H}(t)$. By Proposition 3.1, $\mathcal{H}(t) = 1/2 - 1/(4t+6)$. Hence, $\mathcal{H} \geq 1/2 - 1/(4t+6)$ for any t, and $\mathcal{H} \geq 1/2$. On the other hand, an upper bound $\mathcal{H}_n \leq 1/2$ can be easily derived using the probabilities

$$\pi_{n,j} = \begin{cases} 1 & \text{if } j = (n-1)/2, \\ 0 & \text{otherwise.} \end{cases}$$

Thus $\mathcal{H} = 1/2$. Using the CMT again, we conclude $F_n = (1+\xi/4)\, n / \mathcal{H} + o(n) = (2+\xi/2)\, n + o(n)$. □

Notice that, if we only consider comparisons ($\xi = 0$), then the last theorem states that the average cost of quickselect is $2n + o(n)$, which does not reach the lower bound $1.25n + o(n)$ of the minimum average number of comparisons needed to locate an element of random rank.

The answer provided by Theorem 5.1 leaves open the question we posed ourselves at the beginning. To determine the optimal sample size, we need to consider the lower order terms of the average total cost. In fact, we will get the main term of k^*, where $s^* = 2k^* + 1$ is the optimal sample size that minimizes the average total cost of quickselect.

Let us introduce the function

$$\mathcal{F}(n,k) = -\frac{1}{2} + \sum_{0 \leq j < n} \frac{j^2}{n^2} \left(\pi_{n,j}^{(k)} + \pi_{n,n-1-j}^{(k)} \right) = -\frac{1}{2} + \frac{2}{n^2} \sum_{0 \leq j < n} j^2 \pi_{n,j}^{(k)}, \quad (15)$$

which is only a slight variation of the const-entropy associated to F_n (see the proof of Theorem 5.1).

As $k = \omega(1)$ and $k = o(n)$, then $F_n = (2+\xi/2)\, n + G_n$ for some function $G_n = o(n)$. Substituting in the recurrence for F_n and using the definition of $\mathcal{F}(n,k)$, we have

$$\begin{aligned} F_n = {} & \mathsf{S}(k) + \mathsf{C}(n,k) + \mathsf{X}(n,k) \cdot \xi + 2 \sum_{0 \leq j < n} \frac{j}{n} \pi_{n,j}^{(k)} G_j \\ & + (1+\xi/4)\, n + (2+\xi/2)\ \mathcal{F}(n,k)\, n. \end{aligned}$$

The behavior of $\mathcal{F}(n,k)$ when k grows with n is stated in the following lemma.

Lemma 5.1 *If $k = \omega(1)$ and $k = o(n)$ then*

$$\mathcal{F}(n,k) = \frac{1}{4k} + \Theta\left(\max\left\{\frac{1}{k^2}, \frac{1}{n}\right\}\right).$$

Proof. Again, we use Proposition 3.1 with $s = 2k+1$ and $p = k$ to get

$$\mathcal{F}(n,k) = \frac{1}{4k} - \frac{3}{4k(2k+3)} - \frac{3}{2n} + \frac{3}{2(2k+3)n} + \frac{1}{(2k+3)n^2}.$$

□

We also need the asymptotic behavior of $\mathsf{X}(n,k)$, given in the following lemma.

Lemma 5.2 *If $k = \omega(1)$ and $k = o(n)$ then*

$$\mathsf{X}(n,k) = \frac{n}{4} - \frac{n}{8k} + \Theta\left(\frac{n}{k^2}\right).$$

Proof. Setting $s = 2k+1$ and $p = k$ in Lemma 2.1 yields

$$\mathsf{X}(n,k) = \frac{(k+1)n^2 - (k+3)n + (2k+2)}{2(2k+3)(n-1)}.$$

Now it is a simple matter to rewrite the equality above to get the statement of the lemma. □

Since $G_n = o(F_n)$, it is rather intuitive that the contribution of the sum of G_j's to the asymptotic location of k^* is irrelevant. The argument can be formalized as follows. Fix any $\delta > 0$. By hypothesis, there exists some N such that $|G_j| \le \delta \cdot j$ for every $j \ge N$. On the other hand, we are assuming $k = \omega(1)$, so when n is large enough we have $k \ge N$ and $\pi_{n,j}^{(k)} = 0$ for every $j < N$. Therefore,

$$2 \sum_{0 \le j < n} \frac{j}{n}\, \pi_{n,j}^{(k)}\, |G_j| \le 2\delta \sum_{0 \le j < n} \frac{j^2}{n}\, \pi_{n,j}^{(k)} = \delta\ \mathcal{F}(n,k)\, n + \frac{\delta}{2}\, n$$

for large n. Thus

$$F_n = \mathsf{S}(k) + \mathsf{C}(n,k) + \mathsf{X}(n,k) \cdot \xi + E_1(n) n + E_2(n)\, \mathcal{F}(n,k) n, \tag{16}$$

where we have the bounds $(1 + \xi/4) - \delta/2 \le E_1(n) \le (1 + \xi/4) + \delta/2$ and $(2 + \xi/2) - \delta \le E_2(n) \le (2 + \xi/2) + \delta$.

Notice that the functions $E_1(n)$ and $E_2(n)$ *do not depend on* k. Moreover, recall that $\mathsf{S}(k) \sim 2\beta\, k$ for some constant β, and $\mathsf{C}(n,k) = n + 1$. Then it is easy to see that, for large n, k^* belongs to the interval $[k_1, k_2]$, where k_1 is the value of k that minimizes the function

$$(1+\delta)2\beta\, k - \frac{(1+\delta)\cdot\xi}{8k}\, n + \left[\left(2+\frac{\xi}{2}\right) - \delta\right] \frac{(1-\delta)}{4k}\, n, \tag{17}$$

and similarly, k_2 minimizes the function

$$(1-\delta)2\beta\, k - \frac{(1-\delta)\cdot\xi}{8k}\, n + \left[\left(2+\frac{\xi}{2}\right) + \delta\right] \frac{(1+\delta)}{4k}\, n. \tag{18}$$

This yields

$$k_1 = \sqrt{\frac{((4+\xi)-2\delta)(1-\delta)-(1+\delta)\cdot\xi}{16(1+\delta)\beta} n},$$
$$k_2 = \sqrt{\frac{((4+\xi)+2\delta)(1+\delta)-(1-\delta)\cdot\xi}{16(1-\delta)\beta} n}.$$

The reasoning above holds for every $\delta > 0$, no matter how small we choose it. Therefore, we can conclude the following theorem.

Theorem 5.2 *Let $s^* = 2k^* + 1$ denote the optimal sample size w.r.t. the average total cost F_n of quickselect to select a random element, when the median of the sample is used as pivot. Then*

$$k^* = \sqrt{\frac{1}{4\beta}} \cdot \sqrt{n} + o\left(\sqrt{n}\right).$$

Notice that it has been enough to consider the main terms of $\mathsf{S}(k)$, $\mathsf{X}(n,k)$ and $\mathcal{F}(n,k)$ to find the main term of k^*. The procedure that we have just outlined here will be used again in Section 6. To avoid dealing with the tedious details, we will disregard terms of small order and obtain minima as if these terms did not exist. We have already shown here that it can be safely done and yields asymptotically valid conclusions.

6 Optimal Samples for Quicksort

In this section we study which are the optimal sample sizes for quicksort. We already know from Section 4 that picking the median of the sample as the pivot is not always the best choice. Therefore, we set $p = \psi \cdot s + o(s)$ for some fixed $0 < \psi \leq 1/2$ (the case $1/2 \leq \psi < 1$ is symmetrical). Notice that we do not assume $\psi = \psi^*(\xi)$.

We have the following theorem for the main term of the cost of quicksort when we make the size of the sample grow with the size of the input.

Theorem 6.1 *Let $p = \psi \cdot s + o(s)$, where $0 < \psi < 1$ and $s = s(n)$ is a function such that $s = \omega(1)$ and $s = o(n)$. Then the average total cost of quicksort is*

$$Q_n = q_\xi(\psi)\, n \ln n + o(n \log n).$$

Proof. First introduce the function

$$\nu_n(t,x) = \sum_{0 \le j < n} \pi_{n,j}^{(t, x\cdot(t-1))} \left(j \ln j + (n-1-j) \ln(n-1-j) \right),$$

where t is any integer $1 \le t \le n$ and x is a real number, $0 < x \le 1/2$. The weights $\pi_{n,j}^{(t,d)}$ are defined in terms of the Γ function in order to make sense for real d ($d = x \cdot (t-1)$ in this case). Similarly, we extend the function $\mathrm{VE}(t,d)$ for real d by using the Ψ function instead of the harmonic numbers. We will stick to the same names for these extensions, though.
Now, since $j \ln j$ is a convex function, it is easy to see that $\nu_n(t, x_1) > \nu_n(t, x_2)$ if $0 < x_1 < x_2 \le 1/2$, and $\nu_n(t_1, x) > \nu_n(t_2, x)$ if $t_1 < t_2$. Moreover, $\nu_n(t,x)$ is continuous on x.
Let $Q_n(t,x)$ be the average cost of quicksort when we use samples of fixed size t and select the $(d+1)^{\mathrm{th}}$ item from the sample as the pivot, where $d = x \cdot (t-1)$ (such a quicksort doesn't make sense for all x, but we can write down its recurrence, which is what really matters for the argument). Then, according to the CMT, the log-entropy associated to $Q_n(t,x)$ is

$$\mathcal{H}'_n(t,x) = \ln n - \frac{1}{n} \nu_n(t,x) + o(1),$$

and its limit $\mathcal{H}'(t,x) = \lim_{n\to\infty} \mathcal{H}'_n(t,x) = \mathrm{VE}(t,d)$ (see Section 4). Note that even if d is not an integer, the entropy and its limit are well-defined.
Fix any $0 < \delta < \psi$. Under the assumptions of the theorem, we have $s \ge t$ and $p \ge (\psi - \delta) \cdot (s-1)$ for large n. Therefore, the log-entropy associated to Q_n can be bounded by

$$\mathcal{H}'_n \ge \ln n - \frac{1}{n} \nu_n(t, \psi - \delta) + o(1).$$

Hence, $\mathcal{H}' = \lim_{n\to\infty} \mathcal{H}'_n \ge \lim_{n\to\infty} \mathcal{H}'_n(t, \psi - \delta) = \mathrm{VE}(t, (\psi - \delta)\cdot(t-1))$. This bound holds no matter how large we choose t or how small we choose δ. Hence, $\mathcal{H}' \ge \lim_{t\to\infty} \lim_{\delta\to 0} \mathrm{VE}(t, (\psi-\delta)\cdot(t-1)) = -(\psi \ln \psi + (1-\psi)\ln(1-\psi))$.
On the other hand, a matching upper bound $\mathcal{H}' \le -(\psi \ln \psi + (1-\psi)\ln(1-\psi))$ can be easily derived using the probabilities

$$\pi_{n,j} = \begin{cases} 1 & \text{if } j = \lfloor \psi \cdot (n-1) \rfloor, \\ 0 & \text{otherwise.} \end{cases}$$

Thus $\mathcal{H}' = -(\psi \ln \psi + (1-\psi)\ln(1-\psi))$. Using the CMT, we conclude the statement of the theorem, $Q_n = q_\xi(\psi)\, n \ln n + o(n \log n)$. □

If we only measure the number of comparisons ($\xi = 0$), then the theorem above states that *any* sample size $s = \omega(1)$ and $s = o(n)$ with $\psi = 1/2$ is asymptotically optimal w.r.t. the main term of quicksort, as the expected number of comparisons for all them is $Q_n \sim n \log_2 n$.

To investigate the optimal sample size we need to consider the lower order terms and introduce a pseudoentropy

$$\begin{aligned}\mathcal{Q}(n,s,\psi) = &-\left(H_n + \psi \ln \psi + (1-\psi)\ln(1-\psi)\right)\frac{n-1}{n} \\ &+ \sum_{0 \le j < n}\left(\pi_{n,j}^{(s,p)} + \pi_{n,n-1-j}^{(s,p)}\right)\frac{jH_j}{n},\end{aligned}$$

which plays a rôle analogous to that of $\mathcal{F}(n,k)$ in the analysis of quickselect. Here and for the rest of the section, we will assume that $p = \lceil \psi \cdot (s+1) \rceil - 1$. Any other discretization satisfying $lim_{s\to\infty}(p/s) = \psi$ will yield results similar to the one that we assume.

By Theorem 6.1 we know that we can decompose Q_n as $Q_n = q_\xi(\psi)\, nH_n + R_n$, where $R_n = o(n \log n)$. The recurrence for quicksort (Lemma 2.2) can be then rewritten as

$$\begin{aligned}Q_n = {} & \mathsf{S}(s,\psi) + \mathsf{C}(n,s,\psi) + \mathsf{X}(n,s,\psi)\cdot\xi + \sum_{0\le j<n}\left(\pi_{n,j}^{(s,p)} + \pi_{n,n-1-j}^{(s,p)}\right)R_j \\ &+ \Big(H_n + \psi\ln\psi + (1-\psi)\ln(1-\psi)\Big)q_\xi(\psi)\,(n-1) \\ &+ \mathcal{Q}(n,s,\psi)q_\xi(\psi)\,n.\end{aligned}$$

Let $\chi_\psi(s)$ be the oscillating factor induced by taking ceilings:

$$\chi_\psi(s) = \lceil \psi\cdot(s+1)\rceil - \psi\cdot(s+1). \tag{19}$$

We have the following lemma for the asymptotic behavior of $\mathcal{Q}(n,s,\psi)$.

Lemma 6.1 *If $s = \omega(1)$, $s = o(n)$ and $p = \lceil \psi\cdot(s+1)\rceil - 1$ for some ψ such that $0 < \psi \le 1/2$, then*

$$\mathcal{Q}(n,s,\psi) = \frac{1}{2s} - \Big(\ln(1-\psi) - \ln\psi\Big)\frac{\chi_\psi(s)}{s} + \mathcal{O}\left(\max\left\{\frac{1}{n}, \frac{1}{s^2}\right\}\right).$$

Proof. Using the second sum in Proposition A.2 of Appendix A yields

$$\begin{aligned}\sum_{0\le j<n}\left(\pi_{n,j}^{(s,p)} + \pi_{n,n-1-j}^{(s,p)}\right)jH_j = {} & (H_n - \mathrm{VE}(s,p))\,(n-1) \\ &+ \frac{p+1}{s+1}(H_{p+1} - H_{q+1}) + \frac{q+1}{s+1}(H_{q+1} - H_{p+1}) \\ &+ \frac{1}{p+1} + \frac{1}{q+1} - \frac{2}{s+1} - 1,\end{aligned} \tag{20}$$

where $q = s - 1 - p$ as usual.
The last two lines above are $\mathcal{O}(1)$. On the other hand, from the equalities $p + 1 = \psi \cdot (s + 1) + \chi_\psi(s)$, $H_n = \ln n + \gamma + 1/2n + \Theta(n^{-2})$ and $\ln(1 + 1/x) = 1/x + \Theta(x^{-2})$, it is not difficult to deduce

$$\begin{aligned} & -H_{s+1} + \frac{p+1}{s+1} \cdot H_{p+1} + \frac{q+1}{s+1} \cdot H_{q+1} \\ = \quad & \psi \ln \psi + (1-\psi)\ln(1-\psi) + \frac{1}{2s} - \Big(\ln(1-\psi) - \ln\psi\Big)\frac{\chi_\psi(s)}{s} + \Theta\left(\frac{1}{s^2}\right), \end{aligned}$$

and the lemma follows after a few simple manipulations. □

The factor $\ln(1-\psi) - \ln\psi$ in the statement of Lemma 6.1 is zero if and only if $\psi = 1/2$. So the perturbation $\chi_\psi(s)/s$ has to be taken into account whenever we are not selecting the median of the sample as the pivot.

We also need the asymptotic properties of $\mathsf{X}(n, s, \psi)$, given in the next lemma. It extends Lemma 5.2 for $\psi < 1/2$. The proof, very similar to others in this work, is omitted.

Lemma 6.2 *If $s = \omega(1)$, $s = o(n)$ and $p = \lceil \psi \cdot (s+1) \rceil - 1$ for some ψ such that $0 < \psi \leq 1/2$, then*

$$\mathsf{X}(n, s, \psi) = \psi(1-\psi)\, n - \psi(1-\psi)\, \frac{n}{s} + (1 - 2\psi)\, \chi_\psi(s)\, \frac{n}{s} + \mathcal{O}\left(\max\left\{\frac{n}{s^2}, 1\right\}\right).$$

We have already pointed out in Section 2 that $\mathsf{S}(s, p)$ is linear, with the constant of proportionality typically dependent on the quotient $(p+1)/(s+1)$. We assume thus, without loss of generality, that $\mathsf{S}(s, p) = \mathsf{S}(s, \psi) = \beta \cdot s + o(s)$ for some constant $\beta = \beta(\psi)$.

By a reasoning identical to that of last section we conclude that to obtain the leading term of the optimal sample size s^* it suffices to find the value of s that minimizes

$$\begin{aligned} & \beta \cdot s + \left(\frac{q_\xi(\psi)}{2} - \psi(1-\psi)\,\xi\right)\, \frac{n}{s} + \\ & \qquad \Big[(1-2\psi)\,\xi - \Big(\ln(1-\psi) - \ln\psi\Big) q_\xi(\psi)\Big]\; \chi_\psi(s)\, \frac{n}{s}. \end{aligned} \tag{21}$$

We will analyze two variants of quicksort with sample sizes depending on n. The first one is setting $\psi = \psi^*(\xi)$, i.e. choosing the optimal ratio p/s; the second one is setting $\psi = 1/2$ irrespective of ξ, that is, picking always the median of the sample. If $\xi \leq \tau$ both variants are identical, as $\psi^* = 1/2$ whenever ξ is below the threshold.

Fortunately, in any of these two variants the second line in Equation (21) vanishes (because of Equation (13)). Therefore, the analysis of the optimal sample sizes of quicksort follows similar steps to those in the analysis of quickselect, yielding the following two theorems.

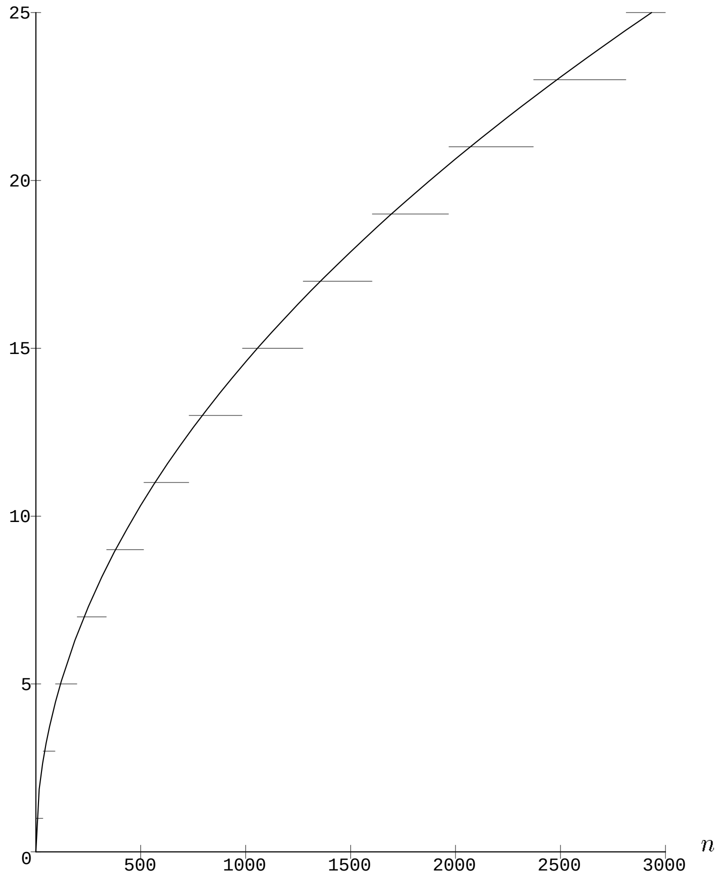

Figure 8: Exact values of the optimal samples for Q_n compared with s^*.

Theorem 6.2 *Let s^* denote the optimal sample size w.r.t. Q_n when the $\lceil \psi^*(s+1)\rceil^{th}$ element of the sample is used as the pivot. Then*

$$s^* = \sqrt{\left(\frac{q_\xi(\psi^*)}{2} - \psi^*(1-\psi^*)\xi\right)\frac{1}{\beta(\psi^*)}} \cdot \sqrt{n} + o\left(\sqrt{n}\right).$$

The next theorem deals with the particular case of selecting the median of the sample as the pivot.

Theorem 6.3 *Let s^* denote the optimal sample size w.r.t. Q_n when the median of the sample is used as the pivot and $\xi < \tau$. Then*

$$s^* = \sqrt{\left(\frac{4-(2\ln 2-1)\xi}{8\ln 2}\right)\frac{1}{\beta}} \cdot \sqrt{n} + o\left(\sqrt{n}\right),$$

where $\beta = \beta(1/2)$.

Notice that when $\xi > \tau$ the result above makes no sense, as the factor multiplying $\sqrt{n}$ is the square root of a negative number. This provides a further check for the conclusions of Section 4, and is consistent with the observation that the optimal samples in that situation have constant size.

Figure 8 shows the exact values (staircase plot) of the optimal sample sizes for the number of comparisons. The data comes from [12]. The continuous curve is the leading term of the optimal sample size s^* as stated by Theorem 6.3, setting $\xi = 0$ and assuming that standard quickselect is the median-finding algorithm ($\beta = 2(1 + \ln 2)$).

7 Related issues

7.1 Variance of quickselect

In this subsection we will study the variance of the number of comparisons made by quickselect with sampling, specifically, for the case where the selected pivot is the median of the sample. This analysis is somewhat more complicated than the analysis of its expected performance, but feasible.

Recall that the variance of the number of comparisons made by standard quickselect is $\Theta(n^2)$ [11]. Thus, the expected cost and the standard deviation are of the same order of magnitude, and this implies that there is a low but non-negligible probability that the number of comparisons actually made is much larger than the expected number.

In [13] it is shown that, under some mild assumptions, it is fairly easy to systematically obtain recurrences for $\mathbb{E}\left[X_n^2\right]$, the second moment of the random variable X_n denoting the cost of a one-branch or a two-branch divide-and-conquer algorithm. Once the recurrence for the second moment has been derived, the CMT can be used to obtain asymptotic information about $\mathbb{E}\left[X_n^2\right]$ and $\mathbb{V}\left[X_n\right]$. Quickselect fails into the class of one-branch divide-and-conquer algorithms, and thus these techniques can be applied to analyze its variance.

We assume that the median-finding algorithm used to select the pivots has quadratic variance —an assumption that is not too astringent since we are already assuming that this algorithm works in linear time on the average. Furthermore, if the variance were subquadratic, the basic conclusions of this section would not change drastically. Let $F_n^{(2)}$ denote the second moment of the number of comparisons made by quickselect to find an element of random rank out of n.

A key observation that makes the analysis relatively simple is that the number of comparisons needed to partition the array is $\mathsf{C}(n, k) = n + 1$, and this is independent of the number of comparisons made to select the pivot and the comparisons made in further invocations of quickselect in smaller subarrays.

We consider first the situation where $s = 2k + 1 = \Theta(1)$. The recurrence for

$F_n^{(2)}$ is, under the assumptions above,

$$\begin{aligned} F_n^{(2)} &= 2(n+\Theta(1))F_n - (n+\Theta(1))^2 + \sum_{0\le j<n} \omega_{n,j}^{(k)} F_j^{(2)} \\ &= \left(3 + \frac{2}{k+1}\right) n^2 + \mathcal{O}(n) + \sum_{0\le j<n} \omega_{n,j}^{(k)} F_j^{(2)}. \end{aligned}$$

The weights and shape function are the same as for the analysis of the expected cost of quickselect with fixed-size samples, so we have

$$\begin{aligned} \varphi(x) &= \int_0^1 \omega^{(k)}(z)\, z^x\, dz \\ &= \frac{2\cdot(2k+1)!}{k!\,k!} \int_0^1 z^{k+1+x}(1-z)^k\, dz, \end{aligned}$$

but we have to evaluate $\varphi(x)$ at $x=2$, because the toll function is quadratic now. This yields for the limiting const-entropy

$$\mathcal{H}(k) = 1 - \varphi(2) = 1 - \frac{k+3}{2(2k+3)} = \frac{3(k+1)}{2(2k+3)}$$

and the asymptotic behavior of $F_n^{(2)}$,

$$F_n^{(2)} = \frac{(3k+5)(4k+6)}{3(k+1)^2}\, n^2 + o(n^2).$$

Finally, subtracting F_n^2 from $F_n^{(2)}$ we get the variance.

Theorem 7.1 *The variance of the number of comparisons made by quickselect when using samples of fixed size $s = 2k+1$ is*

$$\begin{aligned} V_n &= \frac{2k+3}{3(k+1)^2}\, n^2 + o(n^2) \\ &= \left(\frac{2}{3k} + \mathcal{O}\left(\frac{1}{k^2}\right)\right) n^2 + o(n^2) \end{aligned}$$

As expected, sampling not only reduces the expected number of comparisons, it also reduces its variance: if $v(k)$ denotes the coefficient of n^2 in V_n we have

$$\begin{aligned} v(0) &= 1, \\ v(1) &= \frac{5}{12} \approx 0.41667, \\ v(2) &= \frac{7}{27} \approx 0.25926, \\ &\dots \end{aligned}$$

and $\lim_{k\to\infty} v(k) = 0$. This suggests that if we take samples of increasing size ($k = \omega(1)$, $k = o(n)$) then the variance would be $o(n^2)$. Indeed, this is what actually happens.

The analysis of this case goes along the same lines as the analysis of the expected value F_n in Section 5, and the techniques that were used to derive Theorem 5.1 yield in this case that the limiting entropy associated to $F_n^{(2)}$ is $\mathcal{H} = 3/4$. Hence, since the toll function of $F_n^{(2)}$ is $3n^2 + o(n^2)$, we have

$$F_n^{(2)} = 4n^2 + o(n^2), \tag{22}$$

and the variance is $V_n = F_n^{(2)} - F^2(n) = 4n^2 + o(n^2) - (2n + o(n))^2 = o(n^2)$.

A refinement of this calculation is possible, since we know that (see Equation (16) in Section 5)

$$F_n = 2n + \frac{n}{2k} + \Theta(k) + \Theta\left(\frac{n}{k^2}\right).$$

and therefore,

$$F_n^{(2)} = 4n^2 + \Theta\left(\max\left\{nk, \frac{n}{k^2}\right\}\right).$$

This yields the following theorem.

Theorem 7.2 *The variance V_n of the number of comparisons made by quickselect when the samples are of size $2k+1$ and $k = \omega(1)$ and $k = o(n)$, is*

$$V_n = \mathcal{O}(\max\{\frac{n^2}{k}, nk\}).$$

Notice that the upper bound given above is $o(n^2)$ for any k such that $k = \omega(1)$ and $k = o(n)$. If $k = \Theta(\sqrt{n})$ then the variance is $\mathcal{O}(n^{3/2})$ and the standard deviation is $\mathcal{O}(n^{0.75})$. This is the right choice since it simultaneously minimizes the average total cost and the order of magnitude the variance.

Concerning quicksort, we have not been able to analyze the effect of sampling in the variance. We conjecture that if fixed-size samples are used, the variance is still quadratic but the coefficient decreases with k and that when samples increase with n then the variance of quicksort satisfies Theorem 7.2 (with k replaced by s and using the appropriate pivot, namely, the $(p^*+1)^{\text{th}}$ element in the sample).

7.2 Tuning performance

The comparisons and exchanges made while selecting the pivot provide information that is mostly discarded because the partition brings the selected pivot to the left extreme of the subarray, and compares each other element of the array with the pivot and exchanges elements whenever necessary. In fact, we have assumed that the sample is copied into a separate area, and the selection process is carried out there.

We might select the pivot in-place instead. When sorting $A[l..u]$, we would take the segments $A[l..p+1]$ and $A[u-q+1..u]$ as the sample of s elements and apply a slightly modified version of quickselect which would bring the $(p+1)^{\mathrm{th}}$ element of the sample to $A[l]$, and put all smaller elements in $A[2..p+1]$ and all larger elements in $A[u-q+1..u]$. Then the partition of $A[l..u]$ would initially set $\mathit{left} := p+2$ and $\mathit{right} := u-q$.

The partitioning is thus avoiding redundant comparisons and exchanges, so $\mathsf{C}(n,s,p) = n+2-s$ and

$$\mathsf{X}(n,s,p) = \frac{(p+1)(q+1)}{(s+1)(s+2)}(n-1-s),$$

if $s < n$.

This selection-plus-partition combination is more efficient than the standard mechanism assumed in previous sections, but it does not preserve randomness, because the selection of the pivot reorganizes the elements of the sample. There seems not to exist —for general s— an efficient way to perform selection in-place and partition without redundant comparisons and exchanges while preserving randomness.

However, we may disregard the small amount of sortedness that the selection process introduces and assume that the analysis is still possible, yielding good approximate results.

The steps and computations would be absolutely analogous to those of previous sections. The results are, qualitatively speaking, identical. We give here a couple of these results. They correspond to the tuned variants of quickselect and quicksort and are valid provided that our assumption above were correct.

Theorem 7.3 *Let $s^* = 2k^*+1$ denote the optimal sample size w.r.t. the average total cost $F_n^{[tuned]}$ of the tuned variant of quickselect, when the median of the sample is used as pivot. Then*

$$k^* = \sqrt{\frac{1}{4\beta - 4 - \xi}} \cdot \sqrt{n} + o\left(\sqrt{n}\right).$$

Recall that quickselect (with or without sampling) is the median-finding algorithm in the tuned variant. Hence, we have $\beta \geq 2 + \xi/2$ (because finding the median is more costly than finding an element of random rank) and the factor $\sqrt{4\beta - 4 - \xi}$ above is always well-defined.

Theorem 7.4 *Let s^* denote the optimal sample size w.r.t. the average total cost $Q_n^{[tuned]}$ of the tuned variant of quicksort when the $\lceil \psi^*(s+1) \rceil^{th}$ element of the sample is used as pivot. Then*

$$s^* = \sqrt{\left(\frac{q_\xi(\psi^*)}{2} - \psi^*(1-\psi^*)\xi\right) \frac{1}{\beta(\psi^*) - 1 - \psi^*(1-\psi^*)\xi}} \cdot \sqrt{n} + o\left(\sqrt{n}\right).$$

Acknowledgements

We thank Hsu-Kuei Hwang for various useful comments and suggestions on an early version of this work, specially for encouraging us to study the variance of quicksort and quickselect.

References

[1] D.H. Anderson and R. Brown. Combinatorial aspects of C.A.R. Hoare's FIND algorithm. *Australasian Journal of Combinatorics*, 5:109–119, 1992.

[2] J.L. Bentley and M.D. McIlroy. Engineering a sort function. *Software—Practice and Experience*, 23:1249–1265, 1993.

[3] M. Blum, R.W. Floyd, V.R. Pratt, R.L. Rivest, and R. Tarjan. Time bounds for selection. *J. Computer and System Sciences*, 7:448–461, 1973.

[4] R.W. Floyd and R.L. Rivest. Expected time bounds for selection. *Communications of the ACM*, 18:165–173, 1975.

[5] G.H. Gonnet and R. Baeza-Yates. *Handbook of Algorithms and Data Structures - In Pascal and C*. Addison-Wesley, 2nd edition, 1991.

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

[7] C.A.R. Hoare. FIND (Algorithm 65). *Communications of the ACM*, 4:321–322, 1961.

[8] C.A.R. Hoare. Quicksort. *Computer Journal*, 5:10–15, 1962.

[9] P. Kirschenhofer, H. Prodinger, and C. Martínez. Analysis of Hoare's FIND algorithm with median-of-three partition. *Random Structures & Algorithms*, 10:143–156, 1997.

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

[11] H. Mahmoud, R. Modarres, and R. Smythe. Analysis of quickselect: An algorithm for order statistics. *RAIRO, Theoretical Informatics & Applications*, 29:255–276, 1995.

[12] C.C. McGeoch and J.D. Tygar. Optimal sampling strategies for quicksort. *Random Structures & Algorithms*, 7:287–300, 1995.

[13] S. Roura. *Divide-and-Conquer Algorithms and Data Structures*. PhD thesis, Dept. Llenguatges i Sistemes Informàtics, U. Politècnica de Catalunya, October 1997.

[14] S. Roura. An improved master theorem for divide-and-conquer recurrences. In P. Degano, R. Gorrieri, and A. Marchetti-Spaccamela, editors, *Proc. of the 24th Int. Colloquium on Automata, Languages and Programming (ICALP'97)*, volume 1256 of *Lecture Notes in Computer Science*. Springer, 1997.

[15] R. Sedgewick. The analysis of quicksort programs. *Acta Informatica*, 7:327–355, 1976.

[16] R. Sedgewick. Implementing quicksort programs. *Communications of the ACM*, 21:847–856, 1978.

[17] R. Sedgewick. *Quicksort*. Garland, 1978.

[18] R. Sedgewick. *Algorithms in C*, volume 1. Addison-Wesley, 3rd edition, 1997.

[19] R. Sedgewick and Ph. Flajolet. *An Introduction to the Analysis of Algorithms*. Addison-Wesley, 1996.

[20] R.C. Singleton. Algorithm 347: An efficient algorithm for sorting with minimal storage. *Communications of the ACM*, 12:185–187, 1969.

[21] M.H. van Emden. Increasing the efficiency of quicksort. *Communications of the ACM*, 13:563–567, 1970.

A A few combinatorial identities

In this appendix we consider some useful identities involving binomial coefficients (Proposition A.2) and prove Lemma 2.1 about $\mathsf{X}(n, s, p)$.

The material and the proofs that follow is standard and can be found in any good book on discrete mathematics, like [6]. But we introduce a few definitions in order to make the development self-contained.

The ℓ^{th} falling factorial of x is $x^{\underline{\ell}} = x \cdot (x-1) \cdots (x-\ell+1)$ for any real x and integer $\ell \geq 0$. By definition, $x^{\underline{0}} = 1$. Some obvious identities satisfied by falling factorials include $\ell^{\underline{\ell}} = \ell!$, $(-x)^{\underline{\ell}} = (-1)^{\ell}(x+\ell-1)^{\underline{\ell}}$ and $x^{\underline{\ell}} \cdot (x-\ell)^{\underline{m}} = x^{\underline{\ell+m}}$.

Given a function f from the positive integers to the positive integers, let $\Delta f(n) = f(n+1) - f(n)$ and $\mathrm{E}\, f(n) = f(n+1)$. The following proposition contains two basic results that are the discrete analogues of fundamental theorems of Calculus (the second is known *summation by parts*).

Proposition A.1 *Let u, v and U be functions from the positive integers to the positive integers, and assume $\Delta U(n) = u(n)$. Then*

1. $\sum_{a \leq i < b} u(i) = U(b) - U(a)$.
2. $\sum_{a \leq i < b} u(i) \cdot v(i) = U(b) \cdot v(b) - U(a) \cdot v(a) - \sum_{a \leq i < b} E\, U(i) \cdot \Delta v(i)$.

Many of the computations in this work involve evaluating sums of the type

$$\sum_{0\le j<n} g(j,n)\binom{j}{p}\binom{n-1-j}{q} = \frac{1}{p!\,q!}\sum_{0\le j<n} g(j,n)\, j^{\underline{p}}\,(n-1-j)^{\underline{q}},$$

for some appropriate function $g(j,n)$. For instance, to evaluate $\mathsf{X}(n,s,p)$ we will have $g = j(n-1-j)$. To compute $\mathcal{F}(n,k)$ we need to plug $g = j^2$, and for $\mathcal{Q}(n,\psi)$ we will use $g = jH_j$.

The next proposition is helpful to cope with this kind of sums.

Proposition A.2 *Let*

$$P_{\ell,m}(n) = \sum_{0\le j<n} j^{\underline{\ell}}(n-1-j)^{\underline{m}},$$
$$Q_{\ell,m}(n) = \sum_{0\le j<n} j^{\underline{\ell}}(n-1-j)^{\underline{m}} H_j.$$

Then

$$P_{\ell,m}(n) = \ell!\; m!\binom{n}{\ell+m+1}, \qquad \textit{and}$$
$$Q_{\ell,m}(n) = \ell!\; m!\binom{n}{\ell+m+1}\left(H_n - H_{\ell+m+1} + H_\ell\right).$$

Proof. Let $\ell \ge 1$. Then we can write

$$P_{\ell,0}(n) = \sum_{0\le j<n} j^{\underline{\ell}}.$$

Since $\Delta\, x^{\underline{\ell}} = \ell\, x^{\underline{\ell-1}}$

$$P_{\ell,0}(n) = \frac{n^{\underline{\ell+1}}}{\ell+1}.$$

Also, because of symmetry, $P_{0,m}(n) = n^{\underline{m+1}}/(m+1)$. If both $\ell = m = 0$ we have $P_{0,0}(n) = n$. Finally, we use summation by parts again to get the recurrence

$$P_{\ell,m}(n) = \frac{m}{\ell+1}\cdot P_{\ell+1,m-1}(n)$$

for every $\ell, m \ge 1$. Unfolding it by iteration we prove the statement of the proposition for the first sum.

Regarding the second sum, its proof is almost identical. The key observation in the proof is that $\Delta\, H_n = 1/(n+1)$. $\square$

We prove now Lemma 2.1 of Section 2 which gives a closed expression for $\mathsf{X}(n,s,p)$.

Lemma A.1 *The average number of exchanges to partition an array of size n when the pivot is the $(p+1)^{th}$ element of a sample of s elements is*

$$\mathsf{X}(n,s,p) = \frac{1}{(s+1)(s+2)(n-1)}\Big((p+1)(q+1)n^2 - ((p+1)^2+(q+1)^2-(p+1)(q+1)+s+1)n + 2(p+1)(q+1)\Big),$$

where $q = s-1-p$.

Proof. If the pivot ends at $A[j+1]$ after partitioning and there were t elements in $A[2..j+1]$ greater than the pivot, then exactly t swaps were needed to arrange the array (we do not compute a few additional swaps needed just before and after the partitioning). The probability of this to happen is

$$\frac{\binom{j}{j-t}\binom{n-1-j}{t}}{\binom{n-1}{j}},$$

by an argument largely analogous to the one we used to obtain the value of $\pi_{n,j}^{(s,p)}$ in Proposition 2.1. Thus,

$$\mathsf{X}(n,s,p) = \sum_{0\le j<n} \pi_{n,j}^{(s,p)} \sum_t t\cdot \frac{\binom{j}{j-t}\binom{n-1-j}{t}}{\binom{n-1}{j}}.$$

We can now use the "derivative" of Vandermonde's convolution,

$$\sum_t t\binom{b}{c-t}\binom{a}{t} = a\binom{a+b-1}{c-1},$$

to get

$$\sum_t t\binom{j}{j-t}\binom{n-1-j}{t} = (n-1-j)\binom{n-2}{j-1} = \frac{j(n-1-j)}{(n-1)}\binom{n-1}{j}.$$

Therefore,

$$\mathsf{X}(n,s,p) = \sum_{0\le j<n} \pi_{n,j}^{(s,p)}\cdot\frac{j(n-1-j)}{(n-1)}.$$

Finally, using the equality $j(n-1-j) = (j-p)(n-1-j-q) + (j-p)(q-p) + (pn-p^2-p)$, where $p+q+1=s$ as usual, yields

$$\mathsf{X}(n,s,p) = \frac{P_{p+1,q+1}(n) + (q-p)P_{p+1,q+1}(n) + (pn-p^2-p)P_{p,q}(n)}{p!\,q!\,(n-1)\binom{n}{s}}.$$

The rest of the proof is a simple matter of computation. $\square$