Randomized Algorithms (RA-MIRI): Assignment #2

1 Statement

In this programming assignment you will have to write programs to select the j-th smallest element of an array of n elements using:

1. Hoare's quickselect (see chapter 5 "Random Variables and Expectation (II)", pages 24–29, of the course slides).
2. Randomized Selection algorithm of Floy-Rivest (as given in the course slides, chapter 7 "Concentration around the Mean", pages 29–41).

Your programs will generate many arrays of size n containing each a random permutation and apply the selection algorithms to find the j-th smallest element of each array, for $j = \Delta, 2\Delta, \ldots, n$. In order to extract valid conclusions you should work with large arrays, with size $n = 20000$ (or more). It is hence of the utmost importance that your programs are efficient, as you'll be working with very large inputs and repeat many times the experiments to draw statistically meaningful conclusions.

1.1 Quickselect

In the case of quickselect, your program should keep track of the total number of comparisons $C_n^{(j)}$ made between array elements; for each value of j obtain an average number of comparisons $\overline{C}_n^{(j)}$ and a sample standard deviation $\sigma_n^{(j)}$ by applying the algorithm on a reasonably large number of random arrays of the given size. Study and plot the average value (and the deviation in a boxplot) of $\overline{C}_n^{(j)}/n$ as a function of $\alpha = j/n$ and verify that the experimental results match (or not) the theoretical prediction, namely

$$\frac{\overline{C}_n^{(\alpha n)}}{n} \approx 2 - 2(\alpha \ln(\alpha) + (1-\alpha)\ln(1-\alpha))$$

1.2 Randomized Selection

Measure the number of comparisons made to select the median in arrays of size n, for large values of n, the experimental design is similar as for quickselect, but now we will be always slecting the median $j = \lfloor (n+1)/2 \rfloor$, and will be measuring the average number of comparisons for different values of n. The comparisons needed to sort the sample **must also be accounted for** (by coding the sorting algorithm yourself or by passing a comparator function to the system's `sort` function, which keeps track of the number of comparisons made).

If you fill your input arrays with random permutations of the numbers 1 to n, it is extremely easy to check if the algorithm produces the median or not. Record the proportion of arrays in the sample for which the algorithm fails (it will be basically 0, if n is large enough).

Compare the average number of comparisons made by Randomized Selection to that of quickselect, for the value of n for which you have conducted the quickselect experiments.

1.3 Bonus extensions [optional]

Do the experiments in subsection 1.1 for several different values of n and $j = \lfloor (n+1)/2 \rfloor$, then plot $\overline{C}_n^{(n/2)}$ for both quickselect and Randomized Selection, as functions of n. Both algorithms should give you two straight lines, with different slopes, estimate the slopes from your experimental data and draw conclusions about the efficiency of the two algorithms on practical grounds.

Another nice extension you might consider is to repeat the experiments for quickselect but now with quickselect using a small sample of s elements, e.g., $s = 3$, to get a pivot for each recursive stage. For example, we could pick the median of the sample as the pivot of each recursive stage (*quickselect with median-of-three*). Or we could choose the element of the sample with rank r similar to j relative to the sizes: $r/s \approx j/n$ (*quickselect with adaptive sampling*). For instance, with $s = 3$ if $j/n < 1/3$ we'd use the smallest of the sample, if $1/3 \leq j/n < 2/3$ we'd use the median, and if $2/3 \leq j/n$ then we'd use the largest in the sample. Compare the standard variant of quickselect to one or more of the variants using pivot sampling of your choice.

2 Instructions to deliver your work

Submit your work using the FIB-Racó. The deadline for submission is November 6th, 2022 at 23:59. It must consist of a zip or tar file containing all your source code, auxiliary files and your report in PDF format. Include a README file that briefly describes the contents of the zip/tar file and gives instructions on how to produce the executable program(s) used and how to reproduce the experiments. The PDF file with your report must be called `YourLastName_YourFirstName-2.pdf`,

N.B. I encourage you to use LaTeX to prepare your report. For the plots you can use any of the multiple packages that LaTeX has (in particular, the bundle TikZ+PGF) or use independent software such as matplotlib and then include the images/PDF plots thus generated into your document.