Analisys of Algorithms
Escuela Politécnica Superior, UAM, 2019–2020

Set of practices n. 2

Date of delivery

- Thursday Groups: November 14th.
- Friday Groups: November 15th.

This second practice tries to experimentally determine the execution times of algorithms which use divide and conquer approach. The algorithsm to be studied are *MergeSort and QuickSort*. Of each algorithm it will be analyzed on tables of different sizes and the results obtained will be compared with the theoretical study of algorithm.

MergeSort Algorithm

1. Implement in the file `sorting.c` the sorting algorithm corresponding to the method known as *MergeSort*. The prototype of the necessary functions will be `int mergesort(int* table, int ip, int iu)` and merge `int merge(int* table, int ip, int iu, int imiddle)`. Both functions return `ERR` in case of error or the number of times the BO has been executed in case the table is ordered rightly, `table` is the table to sort, `ip` is the first element of the table, `iu` is the last element of the table, `imiddle` is the index of the midpoint of the table.

   Using the file exercise4.c of the previous practice, check that *MergeSort* algorithm orders correctly.

2. Adapting the file exercise5.c of the previous practice for using *MergeSort* algorithm, obtain the table with the average clock time, and the average, minimum and maximum number of times that the BO is executed depending on the size of the permutation. Plot these values and compare them with the theoretical result of the algorithm.

QuickSort Algorithm

3. Implement `int quicksort(int* table, int ip, int iu)` corresponding to the method known as *QuickSort*; This routine returns `ERR` in case of error or the number of basic operations in case the table is ordered rightly, where `table` is the table to sort, `ip` is the first element of the table and `iu` is the last element of the table.

   The routines `partition` and `median` to support `quicksort`, are necessary to do an efficient sorting, and they must be declared as `int split(int* table, int ip, int iu,int *pos)` and `int median(int *table, int ip, int iu,int *pos)`. Both reoutines will return `ERR` in case of error or the number of basic operations in case of table is ordered correctly, the posiition of the pivot will be returned through the argument of type pointer `pos`. In this first implementation, we will use the first element of the table as a pivot, therefore, `median` must return value 0 (or `ERR` in case of error) and return the `ip` value through the argument `pos`.

   Using the exercise4.c of the previous practice, check that *QuickSort* algorithm orders correctly.

4. Adapting the file exercise5.c of the previous practice for using *QuickSort* algorithm, obtain the table with the average clock time, and the average, minimum and maximum number of times that the BO is executed depending on the size of the permutation. Plot these values and compare them with the theoretical result of the algorithm.

5. Implement another function for pivot `int median_avg(int *table, int ip, int iu, int *pos)` which returns the average table position, so $(ip + iu)/2$. Another usual way for choosing the pivot in QuickSort is to choose the position of the intermediate value between the first element of the table, the last and the one that it is in the intermediate position ot the table, using for this operation, three additional basic operations for those calculated through `partition` or `partition_avg`. Implement another routine `int median_stat(int *table, int ip, int iu, int *pos)` which compares the values of the positions `ip`, `iu` e `(ip+ iu)/2` and returns the position containing the intermediate value between the three.

   For example, in this table of 5 elements:

$$T = (13542)$$

The routine `median_stat` must return the index 5 because in this position is the intermediate value between $1, 5$ and 2.

Modify the routine **partition** so that uses these new pivot functions and also calculate the additional basic operations (BOs) performed by `int median_stat`. Compare the average clock time, and the average, minimum and maximum number of times that the BO is executed with each of the three pivot routines implemented. Discuss reasonably the result of such comparisons.

Questions

1. Compare the empirical performance of the algorithms with the theoretical average case for each case. If the traces of the performance graphs are very sharp, why do you think this happens?
2. Analyze the result obtained with the different versions of `quicksort` (versions with different pivot election).
3. What are the best and worst cases for each algorithm? What should be modified in practice to strictly calculate each case (include average case too)?
4. Which of the two algorithms studied is empirically more efficient? Compare this result with the theoretical prediction. Which algorithm(s) is/are more efficient from the point of view of memroy management? Justify your answer.

Material to be delivered in each of the sections

Documentation: The documentation will consist of the following sections:

1. **Introduction:** it consists of a technical description of the work to be carried out, what are the objectives they intend to reach, what input data is required and what data is obtained from the output, so like any kind of comment about the practice.
2. **Printed Code:** the routine code according to the corresponding setcion. The code also includes the header of the routine.
3. **Results:** description of the results obtained, comparative graphics, of the result obtained with the theoretical ones and comments on them.
4. **Questions:** answers to theoretical questions.

All the files necessary to compile the practice and documentation have to be stored in a single compressed file, in zip format, or tgz (tgz represents a tar file compressed with gzip)
The delivery of the source codes corresponding to the practices of the subject AA will be carried out through the Web page **https://moodle.uam.es/**.
Additionally, the practices should be stored in some storage medium (pendrive, hard disk, remote virtual disk, etc) by the student for the day of the practice exam in December.
Ojo: The importance of carrying a pendrive is stressed **in addition to other storage media such as usb, remote virtual disk, email to own address, etc**, since it is not guaranteed that they can be mounted and accessed each and every one of them during the exam, which will mean the grade not pass in practices.