BTH004 - Laboratory assignment 2

The purpose of this laboratory assignment is to practice theoretical and practical algorithm analysis. In particular, you will analyze two sorting algorithms: merge sort and bubble sort. Merge sort has been discussed in rather much detail earlier in this course, and you have learnt about bubble sort earlier in your education.

The assignment should be conducted individually, and it will be examined through a written report (one per student). The report should fulfill the requirements specified below, and it should be sent to Professor Zhenbo Cheng. The deadline for submitting the report is announced by Zhenbo Cheng.

It is highly recommended that you start working with the assignment as soon as possible. It is of particular importance that you start before the scheduled supervision time, so that you can make as much use as possible of the teacher support.

Theoretical analysis

Earlier in the course, we introduced the concept of basic operations. A *basic operation* can be seen as an operation that is fundamental for the performance of a particular algorithm. The idea is that counting the number of basic operations required by an algorithm should provide a reasonably good measure of the work required by an algorithm.

For the sorting algorithms considered in this assignment, the basic operation is "comparing two numbers".

In the theoretical analysis you should, for both merge sort and bubble sort, count the number of basic operations that is required in the worst case, and describe the number of operations as a function ($f(n)$) of the input size (n) .

For example (in order to show how you should present the number of basic operations) the number of basic operations required by some algorithm for some problem can be described as a function $f(n) = 4n^3 + \log n$, where n is the input size.

Practical analysis

This part of the assignment requires that you implement the merge sort and bubble sort algorithms, using any high-level programming language you find suitable, e.g., java, c, c++, or python.

Your program should be able to read a list of values (float or integer), and then sort the list using either merge sort or bubble sort.

For randomly generated lists of varying length, e.g., 10^4, 10^5, 10^6, 10^7, and 10^8, you should measure the time required by the algorithms[1] [2]. You should also measure the initialization time, i.e., the time to read the lists from the text file, initialize data structures, etc.

Make sure that you measure CPU time if possible, and that you turn off any other applications on your computer, which might influence the running time of your sorting algorithms.

Comparison between your theoretical and practical analysis

Let us assume that the number of basic operations (comparisons) is proportional to the total number of operations performed by each of the algorithms, and that the functions (of the input size) describing the number of basic operations for each of the sorting algorithms are correct. It is then possible to compute, for each of the studied lists, a value $c(> 0)$, so that the running time $T(n) = c\dot{f}(n)$ (the running time might be measured in seconds).

For each of the considered inputs and for both sorting algorithms, you should present the calculated c values. You should also describe the observations, and conclusions, that you are able to make by comparing the c values for the two algorithms, and when you compare the c values for different input sizes for each of the two algorithms?

Requirements on report

In the report you should:

- Provide your name.
- Explain the two considered sorting algorithms using your own words (the purpose is to show that you have understood the algorithms, and it is a good idea to do this before you start analyzing and implementing the algorithms).
- Provide pseudocode for both of the two algorithms. This might be necessary in order to complete the theoretical analysis.
- Present the results from your theoretical analysis, i.e., present your estimation of the number of basic operations (comparisons) required by each of the two algorithms. The number of basic operations should be presented as a function of the input size (i.e., the number of elements to sort). In particular, you should describe how you reasoned when estimating the number of required basic operations.
- Present the results from your practical analysis.

[1] It is probably a good idea to construct a program that generates random sequences of values, which you can use as input.

[2] To simplify your implementation of the merge sort, a good idea is to only consider lists with 2^k $(k > 0)$ elements

- Answer the following question: At what input size do you consider the time required for initialization to be negligible in relation to the total running time of the algorithm?
- Present your comparison of your practical and theoretical analyses.