Project: Algorithms for the Selection Problem

This project is concerned with the SELECT problem, i.e. the problem of finding order statistics for sequences of integers: given a sequence $X$ of $n$ integers and a positive integer $k \leq n$, find the $k$th smallest element $z$ of $X$. You are asked to implement and experimentally analyze the following four algorithms/approaches for the SELECT problem:

(a) The worst-case linear time algorithm for the SELECT problem given in section 9.3 of the text.

(b) The same algorithm as in (a), where initially the $n$ elements are divided into groups of 7 instead of groups of 5.

(c) The RANDOMIZED–SELECT algorithm in section 9.2 of the text.

(d) A sorting–based SELECT algorithm: e.g. sort the input $X$ in non-decreasing order using heapsort and return the $k$th element of the sorted sequence.

Your program implementing these algorithms should have three command line arguments in the following order: the number of elements $n$, the integer $k$, and the name of a file containing the sequence $X$. The output of your program should be (in a single output line): for each algorithm (a–d), the order statistic found, the elapsed real–time and CPU running–time in milliseconds, and the total memory used in (bytes).

What to do

1. Develop appropriate software program(s) implementing your algorithm(s) as efficiently as you can. You may use the C, C++, or Java programming languages. You must ensure that your programs can execute on the Unix machines (gl.umbc.edu) available to you on campus.

2. Perform experiments with your software implementations of the implemented algorithms. Run your algorithms for multiple values of $n = 2^1, 2^2, 2^3, 2^4, \ldots, 2^{20}$, and $k = [9n/10], [7n/8], [4n/5], [3n/4], [2n/3], [n/2], [n/4], [n/8], [n/16]$. For each combination of values $n$ and $k$ run your algorithms for at least 30 sequences of integers, that contain uniformly random integers between 0 and $2^{20}$. Measure the running time and memory used by each algorithm on each one of these runs; these are your experimental results.
3. Analyze your experimental results. This means that you do the following

- Statistically analyze the measurements you obtained (e.g. compute averages, medians, minimums, maximums, standard deviations, etc).
- Plot the results of your statistical analyses.
- Discuss the results of your analyses (e.g. what conclusions can you draw from them?) How do they depend on \( n \)? How do they depend on \( k \)? Which of your implemented algorithms is faster and when? What is the largest problem you can solve in 3 secs? etc.

4. Write your project report.

What to Hand in

Each student must hand in a written project report in the format of a computer science technical report. This report must explain what you did, why you did it, how you did it, what you found, and what is the significance of your findings. Do not simply list experimental findings; be sure also to interpret your findings. In addition, comment on the engineering aspects of your work: What difficulties did you encounter, how did you resolve these difficulties, and what were the consequences of your solutions to these difficulties?

Whenever possible, summarize your important experimental findings in appropriate graphs. As a separate appendix to your report, include a well-documented copy of your source code. For more about technical reports, see the guide(s) posted on the course/project website.

You must submit as attachments in a single email to kalpakis@csee.umbc.edu by May 14, 2003, 23:59pm the following:

1. an electronic copy of your project report in Postscript, PDF, or MS Word. If your report is in MS Word, a hard-copy must also be submitted by the same deadline.

2. an electronic archive file (TAR or ZIP) with all your software.

Your project grade will be based on your project report, and depends on both technical and writing aspects of your report.