THE UNIVERSITY OF AUCKLAND
NEW ZEALAND
Tamaki Campus

COMPSCI.220.S1.T - Algorithms and Data Structures

Assignment 1 – Introduction to Algorithm Analysis

Out: Thursday 6th March, 2008
Due: Wednesday 2nd April, 2008
Worth: 8.33% of the final course grade

This assignment is worth **50 marks** representing **8.33%** of your total course grade.

Objectives

- Learn how to evaluate complexity of a given Java code.
- Learn how to use the "Big-Oh", "Big-Omega", and "Big-Theta" notation to analyse performance of algorithms.
- Learn how to solve basic recurrence relations describing the algorithm performance.
- Learn efficiency of sorting and searching algorithms.

Requirements

You should answer the following questions (each answer must be explained in detail):

1. (*2 marks*) Work out the time complexity $T(n)$ of the following piece of code in terms of the number of operations for a given integer n:

```
for ( int i = 1; i < n*n*n; i *= n ) {
     for ( int j = 0; j < n; j += 2 ) {
          for ( int k = 1; k < n; k *= 3 ) {
               // constant number C of operations
          }
     }
}
```

2. (*3 marks*) You have found empirically that the implemented `insertionSort` and `heapSort` methods spent 5 μs and 1 μs, respectively, to sort an array of 1000 objects. Find out how much time each algorithm will spend for sorting an array of $100,000,000$ objects?

3. (*4 marks*) Suggest which of the two software packages, **A** and **B**, of the same price should be bought to maintain large data bases having each up to 10^9 records. It is known that the average time to process n records with **A** and **B** is $T_{\mathbf{A}}(n) = 0.1n \log_2 n$ milliseconds and $T_{\mathbf{B}}(n) = 2.5n$ milliseconds, respectively. Decide which package is better in the "Big-Oh" sense, work out exact conditions, in terms of the database size n, when this package outperforms the other, and recommend the best choice in your case.

4. (*5 marks*) Prove that $T(n) = 5n \log_2 n + 500n$ is $\Omega(n)$ and $O(n^{1+\varepsilon})$ where $\varepsilon > 0$ is an arbitrary small positive constant.

5. (*5 marks*) Let the processing time for two algorithms, **A** and **B**, comply to the conditions $T_{\mathbf{A}}(n)$ is $\Omega(n)$ and $T_{\mathbf{B}}(n)$ is $O(n)$, respectively. Decide whether both the algorithms can be of the same time complexity (i.e. $T_{\mathbf{A}}(n) = T_{\mathbf{B}}(n)$) and, if yes, determine under which conditions this can happen.

6. (*6 marks*) Let the processing time $T(n)$ of a certain optimisation algorithm depend on the problem size n as follows: $T(n) = 0.01n^2 \log_2 n + 6n (\log_2 n)^2$. Determine the "Big-Theta" time complexity bounds for this algorithm, decide whether the statements "$T(n)$ is $O(n^3)$" and "$T(n)$ is $\Omega(n^2)$ hold for this algorithm, and prove your decisions.

7. (*6 marks*) You have theoretically derived that the processing time $T(n)$ of a certain algorithm is both $\Omega(n)$ and $O(n^3)$. Decide whether you can conclude that $T(n)$ is $\Theta(n^2)$.

8. (*7 marks*) Determine a recurrence relation for the time $T(n)$ of processing a string of the length n with the Java method:

```
public static String reverse( String s ) {
   int n = s.length();
   if ( n <= 1 ) return s;
   else return reverse(s.substring(n/2, n)) + reverse(s.substring(0, n/2));
}
```

assuming that the Java method `String substring(int i,int j)` spends $j-i$ time units for copying a substring of elements with indices from i to $j-1$ from the string `s`, and derive a closed-form formula for $T(n)$ by solving this recurrence relation with the base condition is $T(1) = 0$ under the simplifying assumption that $n = 2^m$ with the integer m.

9. (*6 marks*) Assuming $n = 5^m$ with the integer $m = \log_5 n$, derive a closed-form formula for $T(n)$ by solving the recurrence relation $T(n) = 5T(n/5) + 5$ with the base condition $T(1) = 0$.

10. (*6 marks*) You need to select $k = 15$ most successful lottery winners from an unordered array of winning records for $n = 100,000,000$ web lottery participants all over the world. Each record contains the winner's UPI and the winnings specifying how successful this winner is. You know (possibly, from the course COMPSCI 220) two options for selecting the k higher-rank individuals:

    (a) to run k times `quickselect` with linear processing time, $T_{\text{qselect}}(n) = cn$, in order to sequentially fetch the participants having the desired ranks $n, n-1, \ldots, n-k+1$), or

    (b) to run once `quicksort` with $n \log n$ processing time, $T_{\text{qsort}} = cn \log_2(n)$, to sort the array in ascending order of winnings and fetch the k higher-rank participants.

    Providing the factors c are the same for both the algorithms and the fetching time is negligibly small comparing to the sorting time, which option does result in the fastest selection?

Submission

The due date is Wednesday, 2nd April 2008, 8:30 p.m. (ADB time) [the penalty 20% till 3rd April, 8:30 p.m., and 50% till 4th April, 8:30 p.m.; no submission afterwards].

Submit your assignment to the Assignment Dropbox (ADB) system. You have to electronically hand in a pdf-file `Assignment1.pdf` with your answers to Questions 1–10. You may use any platform and text editor to prepare your submission, but you should check at the Faculty of Science Tamaki Computer Labs that your final pdf-file is readable. In particular, you should use only most common fonts like Times New Roman or Arial. Scanned hand-written notes embedded as images into a pdf file **will not be accepted** for marking.

Marking scheme

	% of marks
Correctness of your answers to Questions 1–10	up to 40
Correctness of intermediate steps in deriving the answers	up to 30
Detailed explanations of these steps with the reference, if necessary, to the textbook	up to 30
	100