CISC-235 Data Structures
Assignment 1.2

January 17, 2019

1 Big-O, Big-Ω, and Big-Θ Analysis and Proofs

1) Determine the tighest upper bound (Big-O) for the following function:

$$T(n) = \frac{3}{2}n^2 + \frac{5}{2}n - 3$$

2) Prove that your answer for 1) is the tighest upper bound (Big-O). Hint: you need to prove that you can not find another slower growing function which is also an upper bound (Big-O) for the function T(n).
3) Prove that an algorithm with the complexity function $T(n) = 3nlogn + 4logn + 2$ is $O(nlogn)$.
4) Let f(n) and g(n) be asymptotically nonnegative functions. Using the formal definition of Big-Θ notation, prove that max(f(n), g(n)) = Θ(f(n)+g(n))
5) Prove or disprove (give True or False and then proof): $2^{n+1} = O(2^n)$.
6) Prove or disprove (give True or False and then proof): $2^{2n} = O(2^n)$.

2 Runtime Analysis of Programs

Let us analyse the runtime complexity of two algorithms, i.e., linear search (Algorithm A) and binary search (Algorithm B) using experimental method. Note that we have introduced these two search algorithms in previous lectures (using book orgnization as the example).

Algorithm A: Store the list S in an array or list, in whatever order the values are given. Search the list from beginning to end. If $S[i] == x$ for any value of i, stop searching and return True (i.e., found), otherwise return False.

Algorithm B: Store the list S in an array or indexed list (such as the Python list). **Sort the list**. Use binary search to determine if x is in S. Return True or False as appropriate.

When using Algorithm A, searching for a value that is in the list will require an average of n/2 comparisons, while in the worse case, searching for a value that is not in the list will require n comparisons. When using Algorithm B, searching for any value will require an average of about logn comparisons. However, sorting the list will take $O(nlogn)$ time.

If we are doing a very small number of searches, Algorithm A is preferable. However if we are doing many searches of the same list, Algorithm B is preferable since the time required to sort the list once is more than offset by the reduced time for the searches. This is what complexity theory tells us.

Your task is to conduct experiments to explore the relationship between the size of the list and the number of searches required to make Algorithm B preferable to Algorithm A. See the detailed requirement below:

1) Implement two algorithms using Python/Java/C++. When implementing Algorithm B, you must write your own sort function and your own binary search function. You may use any sort algorithm that has complexity in $O(nlogn)$.
2) For n = 1000, 2000, 5000, and 10000, conduct the following experiment:

   - Use a pseudo-random number generator to create a list S containing n integers.

   - For values of k ranging from 10 upwards:

   - Choose k target values, make sure half of which are in S and half are not in S,

   - Use Algorithm A to search the list S for the k target values. Use Algorithm B to search the list S for the k target values.

   - Determine the approximate smallest value of k for which Algorithm B becomes faster than Algorithm A – note that this may be greater than n. Call this value F(n). Create a table or graph showing your values of F(n) as a function of n.

To easily create a list of search values, half of which are in S and half of which are not: - when generating the list S, use only even integer values - randomly choose some elements of S as the first half of the search list - randomly choose odd integer values as the second half of the search list

You need to have some functions to record the running time of your algorithms, e.g., the timeit module of Python.

3 Assignment Requirements

This assignment is to be completed individually. Your need submit your documented source code and a report of your proofs, experiment, results and analysis, organized and formatted as described in the document "Assignment Reports" (in the OnQ system). Please combine all files into a .zip archive and submit it to OnQ system. The .zip archive should contain your student number.

You can insert pictures into your assignment report, especially for the proof part.
Deadline: By the end of next Thursday (Jan-25,2019)