CSCI E-124 Data Structures and Algorithms — Spring 2018

Problem Set 1

Due: 11:59pm, Monday, February 5th

See homework submission instructions at
`http://sites.fas.harvard.edu/~cs124/e124/problem_sets.html`

Problem 5 is worth 40% of this problem set, and problems 1-4 constitute the remaining 60%.

1 Problem 1

Indicate for each pair of expressions (A, B) in the table below the relationship between A and B. Your answer should be in the form of a table with a "yes" or "no" written in each box. For example, if A is $O(B)$, then you should put a "yes" in the first box. If the base of a logarithm is not specified, you should assume it is base-2.

A	B	O	o	Ω	ω	Θ
$\log_2 n$	$\log_3 n$					
$\log \log n$	$\sqrt{\log n}$					
$2^{((\log n)^7)}$	n^7					
$n!$	n^n					
$\log(n!)$	$\log(n^n)$					

2 Problem 2

For all of the problems below, when asked to give an example, you should give a function mapping positive integers to positive integers. (No cheating with 0's!)

- Show that if f is $o(g)$, then $f \cdot h$ is $o(g \cdot h)$ for any positive function h.
- Give a proof or a counterexample: if f is not $O(g)$, then f is $\Omega(g)$.
- Find (with proof) a function f such that $f(2n)$ is $O(f(n))$.
- Find (with proof) a function f such that $f(n)$ is $o(f(2n))$.
- Show that for all $\epsilon > 0$, $\log n$ is $o(n^\epsilon)$.

3 Problem 3

QuickSort is a simple sorting algorithm that works as follows on input $A[0], \ldots, A[n-1]$:

```
QuickSort(A):
  n = length(A)
  if n <= 1:
    return A
  else:
    mid = floor(n/2)
    smaller <-- number of elements of A less than A[mid]
    larger <-- number of elments of A larger than A[mid]

    // put all elements of A into either B or C, based on whether they're
    // smaller or bigger than A[mid], respectively
    B <-- empty array of length smaller
    C <-- empty array of length larger
    writtenB <-- 0
    writtenC <-- 0
    for i = 1 to n:
      if A[i] < A[mid]:
        B[writtenB] <-- A[i]
        writtenB <-- writtenB + 1
      else if A[i] > A[mid]:
        C[writtenC] <-- A[i]
        writtenC <-- writtenC + 1

    B <-- QuickSort(B)
    C <-- QuickSort(C)
    // "+" denotes array concatenation
    return the array B + [A[mid]] + C
```

Assume the elements of A are distinct, and that the values `smaller` and `larger` are each calculated in time $\Theta(n)$.

(a) (5 points) Construct an infinite sequence of inputs $\{A_k\}_{k=1}^{\infty}$ such that (1) A_k is an array of length n_k with $\lim_{k\to\infty} n_k = \infty$, and (2) if $f(k)$ denotes the running time of QuickSort on A_k, then $f(k) = \Theta(n_k \log n_k)$.

(b) (5 points) Do exactly the same as part (a), except this time construct a sequence yielding $f(k) = \Theta(n_k^2)$.

(c) (2 points, **bonus**) Suppose a function $T = T(n)$ is given satisfying $T(n) = \Omega(n \log n)$ and $T(n) = O(n^2)$. Then do the same as in parts (a) and (b), except this time construct a sequence yielding $f(k) = \Theta(T(n_k))$.

4 Problem 4

Give asymptotic bounds for $T(n)$ in each of the following recurrences. Hint: You may have to change variables somehow in the last one.

- $T(n) = 2T(n/2) + n^2$.
- $T(n) = 7T(n/3) + n$.
- $T(n) = 16T(n/4) + n^2$.
- $T(n) = T(\sqrt[3]{n}) + 1$.

5 Programming Problem

Solve GOBOSORT on the programming server (`https://cs124.seas.harvard.edu`).
Hint: Try to first solve the case $m = 1$ (it is helpful to model your solution after MergeSort), then build from that solution for larger m.