Algorithm Analysis, Term I/2018-19

Assignment 1

Due: Thu, Sep 20, 2018 @ 11.59pm

Directions:

- Your solutions must be typeset. LaTeX is recommended.
- You must upload your solutions as a PDF file on Canvas before the deadline.
- You don't have to include your solutions to the programming problems in the PDF file.

Problem 1

Prove the following statements. You must show your work.

(a) $3n^3 + 75n^2 + 8\log_2 n \in O(n^3)$

(b) $1 + 3 + 5 + \ldots + (2n - 1) \in O(n^2)$.

(c) Show that $1 + 2 + 4 + 8 + \ldots + 2^n$ is $O(2^n)$.

(d) Show that $1 + \frac{1}{2} + \frac{1}{4} + \ldots + \frac{1}{2^n}$ is $O(1)$.

Problem 2

Bubble sort is a popular, but inefficient, sorting algorithm. It works by repeatedly swapping adjacent elements that are out of order.

```
BUBBLE_SORT(A)
1: for i=1 to A.length - 1
2:     for j=A.length down to i+1
3:         if A[j] < A[j-1]
4:             Exchange A[j] with A[j-1]
```

(a) Let A' denote the output of `BUBBLE_SORT(A)`. To prove that `BUBBLESORT` is correct, we need to first show that it terminates and that

$$A'[1] \leqslant A'[2] \leqslant \ldots \leqslant A'[n], \tag{1}$$

where $n = A.length$. In order to show that `BUBBLESORT` actually sorts, what else do we need to show?

(b) State precisely a loop invariant for the for loop in lines 2–4, and prove that this loop invariant holds. Your proof should use the structure of the loop invariant proof presented in class.

(c) Using the termination condition of the loop invariant proven in part (b), state another loop invariant for the for loop in lines 1–4 that will allow you to prove inequality (1). Your proof should use the structure of the loop invariant proof presented in class.

(d) What is the worst-case running time of bubblesort? How does it compare to the running time of insertion sort?

Problem 3

For some fixed positive integer c, consider the summation

$$S(n) = 1^c + 2^c + 3^c + \ldots + n^c.$$

(a) Show that $S(n)$ is $O(n^{c+1})$.

(b) Show that $S(n)$ is $\Omega(n^{c+1})$. *Hint:* Look just at the second half of the series.

Problem 4

Recall the definition of logarithm base two: saying $p = \log_2 m$ is the same as saying $m = 2^p$. In this class, we will typically write log to mean $\log_2$.

(a) How many bits are needed to write down a positive integer n? Give your answer in big-O notation, as a function of n.

(b) How many times does the following piece of code print "hello"? Assume n is an integer, and that division rounds down to the nearest integer. Give your answer in big-O notation, as a function of n.

```
while n > 1:
   print "hello"
   n := n/2
```

(c) In the following code, subroutine $A(n)$ takes time $O(n^3)$. What is the overall running time of the loop, in big-O notation as a function of n? Assume that division by two takes linear time.

```
while n > 1:
   A(n)
   n := n/2
```

Problem 5

Consider the following algorithm:

```
def foo(alist, m, val):
      length = len(alist)
      left, right = 0, 0
      while left < length and alist[left] <= val:
            right = min(length - 1, left + m)
            if alist[left] <= val and alist[right] >= val:
                  break
            left += m;
      if left >= length or alist[left] > val:
            return -1
      right = min(length - 1, right)
      i = left
      while i <= right and alist[i] <= val:
            if alist[i] == val:
                  return i
            i += 1
      return -1
```

Suppose `alist` is an ordered list of n integers (increasing), `val` and m are integers.

(a) Describe briefly what `foo()` does?

(b) According to your answer in the previous part, show that `foo()` works correctly.

(c) What is the worst-case running time of `foo()`?

(d) What is the best value of m if we want to minimize the number of comparisons?

Problem 6

The following statements are all true:

$$
\begin{aligned}
24n^2 + 10n + 20 &= O(24n^2 + 10n + 20) \\
24n^2 + 10n + 20 &= O(24n^2) \\
24n^2 + 10n + 20 &= O(n^{10}) \\
24n^2 + 10n + 20 &= O(n^2)
\end{aligned}
$$

However, the last one is the simplest, cleanest, and tightest of them, and we will refer to it as the minimal big-O form. Write the following expressions in minimal big-O notation.

(a) $100n^3 + 3n$
(b) $200n \log (200n)$
(c) $100n^2 2^n + 3^n$
(d) $100n \log n + 20n^3 + \sqrt{n}$
(e) $1^3 + 2^3 + \ldots + n^3$.