CS 3510 Design & Analysis of Algorithms 1/15/2022

Homework 1

Due:1/22/2022

This assignment is due on **11:59 PM EST, Saturday, January 22, 2022**. You may turn it in 1 day late for no penalty or 2 days late for a 10% penalty. On-time submissions receive 3% extra credit. Note that a late submission means late feedback, which means less time to study before an exam.

You should submit a typeset or *neatly* written pdf on Gradescope. The grading TA should not have to struggle to read what you've written; if your handwriting is hard to decipher, you will be required to typeset your future assignments.

You may collaborate with other students, but any written work should be your own. Write the names of the students you work with at the top of your assignment.

Unless otherwise noted, you should *always* justify your answer. Please give a written description for all algorithms; algorithms may NOT be specified only in pseudocode.

When the base of a log is unspecified, it is assumed to be base 2.

1. **Big-O**

   (10 points) For the following list of functions, cluster the functions of the same order (i.e.,f and g are in the same group if and only if $f = O(g)$ AND $g = O(f)$) into one group, and then rank the groups in increasing order. You do not have to justify your answer.

   - (a) $n\log(n)$
   - (b) $n^{1.01}$
   - (c) $n\sqrt[3]{n}$
   - (d) $2^{\log_3(n)}$
   - (e) $n + \log(n)$
   - (f) $2^{\log(\log(n))}$
   - (g) $10n\log(10n) + 10$
   - (h) $(\log(n))^{10}$
   - (i) $\log(n^{10})$
   - (j) $42n$

   The form of the answer should be, for example, $\{a, b\} < \{c, d\} < \ldots$.

   $\{f, i\} < \{e\} < \{h\} < \{d\} < \{j\} < \{b\} < \{c\} < \{a, g\}$

2. **Slow Multiplication**

(5 points) Consider the following multiplication algorithm:

```
def slowmult(x,y):
    result = 0
    for i from 1 to x (inclusive):
        result += y
    return result
```

Assume x and y have n bits each. What is the running time of this algorithm in terms of n? (It is not $O(n^2)$!) Justify your answer.

Since the iterative part of this code is dependent solely on x and not y, y is used purely as a medium of addition and not iteration. The runtime of the code is only $O(n)$.

3. **Fast Multiplication**

Let x and y be two n-bit numbers, where n is divisible by 3. Let x_L, x_M and x_R consist of the first third, middle third, and final third of the digits of x, so that $x = 2^{\frac{2n}{3}} x_L + 2^{\frac{n}{3}} x_M + x_R$, and define y_L, y_M, and y_R analogously.

(a) (3 points) Express xy in terms of $x_L, x_M, x_R, y_L, y_M, y_R$. Simplify your answer.
$xy = 2^{\frac{4n}{3}} x_L y_L + 2^n (x_L y_M + x_M y_L) + 2^{\frac{2n}{3}} (x_L y_R + x_M y_M + x_R y_L) + 2^{\frac{n}{3}} (x_M y_R + x_R y_M) + x_R y_R$

(b) (7 points) Give a recursive algorithm that calculates the above polynomial with a recurrence relation of $T(n) = 6T(n/3) + O(n)$.

We are going to first split each of x and y into respective third as described in the problem. We will then recursively call the algorithm with the first thirds, middle thirds, and final thirds of the x and y to get $x_L y_L, x_M y_M, x_R y_R$ respectively. It would then be useful to notice that $(x_L + x_M)(y_L + y_M) = x_L y_L + x_L y_M + x_M y_L + x_M y_M$ so $(x_L + x_M)(y_L + y_M) - x_L y_L - x_M y_M = x_L y_M + x_M y_L$. Similarly, $(x_M + x_R)(y_M + y_R) - x_M y_M - x_R y_R = x_M y_R + x_R y_M$. Lastly, $(x_L + x_M + x_R)(y_L + y_M + y_R) - (x_L + x_M)(y_L + y_M) - (x_M + x_R)(y_M + y_R) + 2(x_M y_M) = x_L y_R + x_M y_M + x_R y_L$. These are all desired outcomes to calculate xy so we are able to simplify the algorithm by using these alternatives and match them up to the corresponding coefficients as indicated from part (a).

Function $Multi(x, y)$:
$x_L \leftarrow x[0 : n/3]$
$x_M \leftarrow x[n/3 : 2n/3]$
$x_R \leftarrow x[2n/3 : n]$
$y_L \leftarrow y[0 : n/3]$
$y_M \leftarrow y[n/3 : 2n/3]$
$y_R \leftarrow y[2n/3 : n]$
$A \leftarrow Multi(x_L, y_L)$
$B \leftarrow Multi(x_M, y_M)$
$C \leftarrow Multi(x_R, y_R)$
$D \leftarrow Multi(x_L + x_M, y_L + y_M)$
$E \leftarrow Multi(x_M + x_R, y_M + y_R)$
$F \leftarrow Multi(x_L + x_M + x_R, y_L + y_M + y_R)$
return $2^{\frac{4n}{3}} A + 2^n (D - A - B) + 2^{\frac{2n}{3}} (F - D - E + 2B) + 2^{\frac{n}{3}} (E - B - C) + C$