CS 4349 — Advanced Algorithm Design and Analysis, Homework 2

https://iniyanjoseph.com | iniyanijoseph@gmail.com

Question 1 - We prove the following using the Master Method

a

$$T(n) = T\left(\frac{8n}{11}\right) + n$$

$$\log_{\frac{11}{8}} 1 = 0$$

$$n = \Omega(n^0) = \Omega(1)$$

We must now prove regularity

$$\frac{8n}{11} < c \cdot n$$

Thus, T(n) = $\Theta(n)$

b

$$T(n) = 16T\left(\frac{n}{4}\right) + n^2$$

$$\log_4 16 = 2$$

$n^2 = \Theta(n^2)$, so T(n) is $\Theta(n^2 \lg n)$

c

$$T(n) = 4T\left(\frac{n}{2}\right) + n^2 \lg n$$

$\log_2 4 = 2$ $n^2 \log n = O(n^{2+\varepsilon})$

$$4 \cdot n^2 \lg\left(\frac{n}{2}\right) =$$

$$4 \cdot n^2 \lg n - 1 \leq$$

$$4n^2 \lg n - 1 = \Omega(n^{2+\varepsilon})$$

We must now prove regularity

$$4 \cdot \left(\frac{n}{2}\right)^2 \cdot \lg\left(\frac{n}{2}\right) = n^2 \cdot \lg\left(\frac{n}{2}\right) < c \cdot n^2 \lg n$$

Thus, T(n) is $\Theta(n^2 \lg n)$

d

$$T(n) = 8T\left(\frac{n}{3}\right) + n^2$$

$\log_3 8 \approx 1.893$

$$n^2 = \Omega(n^{\log_3 8 + \varepsilon})$$

We must now prove regularity

$$\frac{8n^2}{3^2} = \frac{8n^2}{9} < c \cdot n^2$$

Thus, T(n) = $\Theta(n^2)$

Question 2

a

No. $5^{2n} \neq O(5^n)$. Recall that $5^{2n} = \left(5^2\right)^n = 25^n$.
Now let us proceed by contradiction. Assume there is constants n_0, c such that $c \cdot 25^n \leq 5^n$ for all $n \geq n_0$ and $c \geq 1$.
Rearranging, we get

$$c \cdot 25^n \leq 5^n$$
$$c \cdot \left(5^n\right)^2 \leq 5^n$$
$$c \cdot 5^n \leq 1$$

but this is clearly false.

b

No. $2.0000000001 > 2$. This means that $n^2 \neq O(n^{2.0000000001})$
We can show this through contradiction. Assume there is constants n_0, c such that $c \cdot n^{2.0000000001} \leq n^2$ for all $n \geq n_0$ and $c \geq 1$.
This implies that $c \cdot n^2 \cdot n^{0.0000000001} \leq n^2$ for all $n \geq n_0$.
Rearranging, we get that $c \cdot n^2 \leq \frac{n^2}{n^{0.0000000001}} = (n^{1.9999999999})$
However, this cannot be true for n > 1, thus a contradiction is derived.

Question 3

a

Input: A vector of length n called A
Output: A boolean answering the proposition $\forall_{i=1}^{n-1} a_i \leq a_{i+1}$ (i.e if A is sorted)

b

Input: The input size is $O\left(\sum_{(a \in A)} \lceil \log_2 |a| \rceil\right)) = O(n \cdot \lceil \log_2 \max |A| \rceil)$. In practice, every integer is allocated the same number of bits (usually 31 for the positive part), and thus in implementation, is $\Theta(n)$

c

Output: The output size is 1 (1 bit valued at either 1 or 0 representing either T or F)

d

```
1 for each i = 1,...,n-1 do
2 |  if a_i > a_{i+1} do
3 |  |  return F
4 return T
```

e

Input: A vector of length n called A. We will assume $2 \mid n$
Output: A boolean answering the proposition $\forall_{i=1}^{n-2} a_i < a_{i+2}$ (i.e if the even indices of A are sorted in increasing order)

```
1 for each i = 1,3,5,...,n-2 do
2 |  if a_i ≥ a_{i+1} do
3 |  |  return F
4 return T
```

Question 4

a

Input: Two n digit integers which are a power of 2, which we denote X and Y

b

Output: $X \cdot Y$

c

If n=1, the running time is $\Theta(\log_2^2 n)$, as we must repeatedly add to find the product. Since this operation is performed once for each digit in n, each digit is added to itself

d

Divide each integer into two halves :

$$x_a = x_1 \cdot x_2 \cdot x_3 \cdot, ..., \cdot x_{\frac{n}{2}}$$
$$x_b = x_{\frac{n}{2}+1} \cdot x_{\frac{n}{2}+2} \cdot x_{\frac{n}{2}+3} \cdot, ..., \cdot x_n$$
$$y_a = y_1 \cdot y_2 \cdot y_3 \cdot, ..., \cdot y_{\frac{n}{2}}$$
$$y_b = y_{\frac{n}{2}+1} \cdot y_{\frac{n}{2}+2} \cdot y_{\frac{n}{2}+3} \cdot, ..., \cdot y_n$$

Each part of these has length $\frac{n}{2}$

e

Before combining, let us make some observations. $2^{\frac{n}{2}}x_a + x_b = X$ Similarly, $2^{\frac{n}{2}}y_a + y_b = Y$
$\left(2^{\frac{n}{2}}x_a + x_b\right) \cdot \left(2^{\frac{n}{2}}y_a + y_b\right) = 2^{\frac{n}{2}}y_a \cdot \left(2^{\frac{n}{2}}x_a + x_b\right) + y_b \cdot \left(2^{\frac{n}{2}}x_a + x_b\right) =$

$$2^n y_a x_a + 2^{\frac{n}{2}} y_a x_b + 2^{\frac{n}{2}} y_b x_a + y_b x_b =$$
$$2^n y_a x_a + 2^{\frac{n}{2}} (y_a x_b + y_b x_a) + y_b x_b =$$

We will write this as

$$z_2 + 2^{\frac{n}{2}} z_1 + z_0 =$$

We can compute each of these three terms separately. (Conquer)

f

Combining can also simply be seen as the addition of these three terms can be done through addition. We will assume addition is $O(n)$, as each digit must be added.

g

$$T(n) = 3T\left(\frac{n}{2}\right) + n =$$

We now find an asymptotic solution to this recurrence using the master theorem. Recall that the master theorem solves recurrences of the form

$$aT\left(\frac{n}{b}\right) + O(f(n))$$

with $a > 0, b > 1$ and $f(n) > 0$
Let $g(n) = n^{\log_b a}$, which in this case $= n^{\log_2 3}$

$$f(n) = O(g(n))$$

The first case of the master theorem applies, so this method of multiplication is $\Theta\left(n^{\log_2 3}\right)$
Note: This is the Karatsuba algorithm for multiplication. While this faster asymptotic time complexity, this is considered a "Galactic" algorithm, where the constant factor of performing the algorithm practice dominate the asymptotically larger term of the algorithm.

Question 5

a

Assume that $c \cdot 4T\left(\frac{k}{2}\right) + k = O(k^2)$ for $k < n$. We must show that

$$4T\left(\frac{n}{2}\right) + O(n) \leq c \cdot n^2 =$$

.

$$4T\left(\frac{n}{2}\right) + d \cdot n \leq$$

$$4\left(c \cdot \frac{n^2}{4}\right) + d \cdot n$$

$$c \cdot n^2 + d \cdot n$$

$\leq e \cdot n^2$ for some $n_0 > 0$ Thus, $c \cdot 4T\left(\frac{n}{2}\right) + n = O(n^2)$

b

Assume that $c \cdot 2T\left(\frac{k}{2}\right) + k = O(n \log n)$ for $k < n$. We must show that $c \cdot \left(2T\left(\frac{n}{2}\right) + O(n)\right) \leq c \cdot n \log n$.

$$2T\left(\frac{n}{2}\right) + d \cdot n =$$

$$2(c \cdot n \log n) + d \cdot n$$

$$2c \cdot n \log n + d \cdot n$$

$\leq e \cdot n \log n$ for some $n_0 > 0$ Thus, $c \cdot 4T\left(\frac{n}{2}\right) + n = O(n \log n)$