Homework 1

CS 2500

SP 2024

#1

Prob. 1

Implement Karatsuba's algorithm in Python.

Prob 2

Empirical run-time analysis of Karatsuba's algorithm in Python. → Similar to Insertion Sort

Prob. 3

Let $a(n) = n^2 \log_2 n$, $b(n) = 2^n$, $c(n) = 2^{2^n}$, $d(n) = n^{\log_2 n}$

and $e(n) = n^2$.

Claim 3.1: $e(n) = n^2 = O(2^n) = O[b(n)]$

Proof: If $e(n) = O[b(n)]$, there are two constants (k, n_0) s.t. $e(n) \le k \cdot b(n) \ \forall \ n \ge n_0$.

i.e. $n^2 \le k \cdot 2^n \ \forall \ n \ge n_0$.

Applying logarithm on both sides,

$$2 \log n \le \log k + n \cdot \log 2$$

$$\Rightarrow \log n \le \frac{\log k}{2} + \left(n \log \frac{2}{2}\right)$$

P.T.O.

We know

$\log n \leq n - 1 \quad \forall\ n \geq 0.$

Since $\frac{\log 2}{2} \cdot n + \frac{\log k}{2} < n + \frac{\log k}{2}$

We can choose $\frac{\log k}{2} = 0$

$\Rightarrow \quad k = 1$ and $n_0 = 1.$ $\square$

<u>Claim 3.2</u>: $b(n) = 2^n = O(2^{2^n}) = O[c(n)].$

<u>Proof</u>: We need to find two constants k, n_0 s.t.

$\forall\ n \geq n_0, \quad 2^n \leq k \cdot 2^{2^n}.$

Apply log on both sides, we have

$n \log 2 \leq \log k + 2^n \cdot \log 2$

Let $k = 1$, then $n \cancel{\log 2} \leq 2^n \cancel{\log 2}$

(or) $n \leq 2^n.$

Note that $n \leq 2^n \quad \forall\ n \geq 1.$

$\therefore\ k = 1$ and $n_0 = 1.$ $\square$

#3

<u>Claim 3.3</u>: $d(n) = n^{\log n} = O[2^n] = O[b(n)]$.

<u>Proof</u>: Need to find (k, n_0) s.t.

$$n^{\log n} \leq k \cdot 2^n \quad \forall \quad n \geq n_0.$$

Applying logs on both sides,

$$(\log n) \cdot (\log n) \leq \log k + n \log 2$$

If $k = 1$, then

$$(\log n)^2 \leq n \log 2.$$

Multiplying $\frac{1}{4}$ on both sides,

$$\left[\frac{1}{2} \log n\right]^2 \leq \frac{\log 2}{4} \cdot n$$

(or) $$\left[\log\left(\sqrt{n}\right)\right]^2 \leq \frac{\log 2}{4} \cdot \left(\sqrt{n}\right)^2$$

We know $\log x \leq x \quad \forall \quad x \geq 0.$

$\therefore \left(\log \sqrt{n}\right)^2 \leq \frac{\log 2}{4} \left(\sqrt{n}\right)^2 \quad \forall \quad n \geq$ [redacted].

$\Rightarrow k = 1, \ n_0 =$ [redacted]. $\square$

<u>Claim 3.4</u>: $a(n) = n^2 \log n = O[n^{\log n}] = O[d(n)]$

<u>Proof</u>: Note that $n^2 \log n < n^3$ since $\log n < n$ $\forall\, n \geq 0$.

Now, $n^3 < n^{\log n}$ for any $\log n > 3$ (or) $n > 2^3 = 8$.

$\therefore \quad n^2 \log n < \underset{k}{\underset{\shortparallel}{1}}\, n^{\log n} \quad \forall\, n > n_0 = 8$

$\square$

<u>Claim 3.5</u>: $e(n) = n^2 = O[n^2 \log n] = O[a(n)]$.

<u>Proof</u>: Omitted for the sake of brevity as it is a trivial exercise.

<u>Claim 3.6</u>: Let $f(n) \preccurlyeq g(n)$ means $f(n) = O[g(n)]$.

Then,
$$e(n) \preccurlyeq a(n) \preccurlyeq d(n) \preccurlyeq b(n) \preccurlyeq c(n).$$

<u>Proof</u>: Combine the results in Claims 3.1 to 3.5.

$\square$

Prob. 4

```
Incremental Search (A)
    for i = 1 to A.length
        if A[i] = i
            return TRUE
    return FALSE
```

$T_1(n) = O(n)$

In the worst case, i.e. when A does not contain i s.t. $A[i] = i$.

Now, consider divide-and-conquer style.

```
Binary Search (A)
    mid = A.length / 2

    if mid ≥ 1
        if A[mid] = mid
            return TRUE
        else if A[mid] > mid
            Binary Search (A[1: mid-1])
        else
            Binary Search (A[mid+1 : n])
    else
        return FALSE.
```

Note that $T_2(n) \le T_2\left(\frac{n}{2}\right) + c$. Since $a = 1 = 2^0 = b^d$,

we have $T_2(n) = O(n^0 \log n) = \underline{O(\log n)}$

Prob. 5

$$T(n) \leq 7 \cdot T\left(\frac{n}{3}\right) + O(n^2)$$

Here $a = 7$, $b = 3$ and $d = 2$.

$$\therefore \quad a = 7 < 3^2 = b^d$$

Then, from Master theorem, as per CASE 2,

we have $T(n) = O(n^2)$

Prob 6 [Extra Credit]

$$T(n) \leq T\left(\lfloor \sqrt{n} \rfloor\right) + 1 \quad , \quad T(1) = 1$$

Note that this is not a standard recurrence.

Consider a sequence $n = 2^k, 2^{k-1}, \ldots, 2$.

Then, $T(2^k) \leq T(2^{k-1}) + 1$ — (1)

$T(2^{k-1}) \leq T(2^{k-2}) + 1$ — (2)

Substituting (2) in (1), we obtain

$$T(2^k) \leq T(2^{k-2}) + 2 \quad \text{— (3)}$$

P.T.O

#7

Iteratively expanding the RHS, we get

$$T(2^k) \leq T(2^{k-i}) + i$$

$$\forall\ i = 1, 2, \dots, k.$$

$\therefore$ when $i = k$,

$$T(2^k) \leq T(2^0) + k$$

$$= T(1) + k = k+1$$

Since $n = 2^k$, we have $k = \log_2 n$.

$$\therefore\ \underline{\underline{T(n) \leq \log_2 n + 1}} \qquad \square$$

i.e. $\underline{\underline{T(n) = O(\log n)}}$