Assignment - 5

Question 2

1.

→ Prove: Insert method of amortized dictionary having n elements has cost of $O(\log n)$.

Algorithm for insertion:

- Put the insertion element in H array of size 1.

- Initialize counter $i = 0$.

- Check A_i, if empty copy H to A_i and break.

- otherwise, merge $H \leftarrow (A_i, H)$ and increment i by 1.

Let us consider, two sorted lists each with sizes 'L'.

→

Merge cost of A_i & H is at most $2 \cdot 2^i$ as merging of two lists each with size l costs $2l$.

During n insertions,

A_0 will be subjected to merge $n/2$ times with a cost of 2.

A_1 will be subjected to merge $n/4$ times with a cost of 4

Similarly, A_i will be subjected to merge $n/2^{i+1}$ times with a cost of 2^{i+1}.

$\therefore$ Total of $O(\log(n))$ always will be subjected to merge with $O(n)$ cost.

Using aggregate analysis: the amortized cost:

$$O(T(n))/n = O(n)\,O(\log(n))\,/\,O(n)$$
$$= \underline{O(\log(n))}$$

2. Prove that the searching operation of amortized dictionary will cost $O(\log^2 n)$

→ Each level has 2^i elements in it, where i is the level.

To search an element at a level, we use binary search which costs us $O(\log n)$.

Here we ~~have~~ can calculate the levels using $\log(n+1)$, where n are the total number of elements.
Let's say, $K = \log(n+1)$

∴ In worst case we will search

$K \times \log(n)$ number of times.

$= O(\log(n+1) \times \log(n))$

$\approx O(\log^2(n))$.

Assignment - 5

Question - 3

Master theorem :

Given:

$$T(n) = aT\left(\frac{n}{b}\right) + O(n^k \log^p n)$$

$\because$ $a \geq 1$ $\quad$ $k \geq 0$

$b > 1$ $\quad$ $p =$ real number

Cases: :

I] If $a > b^k$, then $T(n) = O(n^{\log_b a})$

: If $a = b^k$ then,

II] a) If $p > -1$ then $T(n) = O(n^{\log_b a} \log^{p+1} n)$

b) If $p = -1$ then $T(n) = O(n^{\log_b a} \log(\log n))$

c) If $p < -1$ then $T(n) = O(n^{\log_b a})$

III] If $a < b^k$

a) If $p \geq 0$ then $T(n) = O(n^k \log^p n)$

b) If $p < 0$ then $T(n) = O(n^k)$

1. $T(n) = 8 \cdot T(n/3) + 2^n$

→ The above equation is not in the form of master's theorem.

To convert it,

as 2^n will be a constant $\forall\, n \in \mathbb{R}$

$\therefore$ we can substitute 2^n by a constant 'c'.

$\therefore T(n) = 8 \cdot T(n/3) + c$

Here, $a = 8$, $b = 3$, $k = 0$, $p = 0$

$\therefore b^k = 3^0 = 1$ $\therefore a > b^k$

According to case I]

$T(n) = O(n^{\log_b a})$

$= O(n^{\log_3 8}) = O(n^{3\log_3 2})$

2. $T(n) = 3 \cdot T(n/3) + \frac{n}{2}$

$\longrightarrow$ Here $a = 3$, $b = 3$, $k = 1$

$b^k = 3^1 = 3$

$\therefore a = b^k$

According to Case II]:

Here, $p = 0$ $\therefore p > -1$

$\therefore$

$T(n) = O(n^{\log_b a} \log^{p+1} n)$

$= O(n^{\log_3 3} \log n)$

$= O(n \log n)$

3. $T(n) = 2 \cdot T\left(\frac{n}{4}\right) + \sqrt{n}$

→ We can write the given equation as:

$$T(n) = 2 \cdot T\left(\frac{1 \cdot n}{4}\right) + n^{1/2}$$

Here : $a = 2$, $b = 4$, $k = 1/2$, $p = 0$

$b^k = 4^{1/2} = 2$

$\therefore a = b^k$

According to Case II]

$p = 0 \quad \therefore p > -1$

$$T(n) = O(n^{\log_b a} \log^{p+1} n)$$

$$= O(n^{\log_4 2} \log n)$$

$$= O(n^{1/2 \log_4 4} \log n) = O(n^{1/2} \log n)$$

$$\boxed{\therefore \text{ANS} = O(\sqrt{n} \log n)}$$

4. $T(n) = 4 \cdot T\left(\frac{n}{2}\right) + 3n$

⟶ Here, $a = 4$, $b = 2$, $k = 1$, $p = 0$

$b^k = 2$

$\therefore a > b^k$

$\therefore$ According to case I]:

$T(n) = \cancel{O(n^{\log_a b})}\ O(n^{\log_b a})$

$= O(n^{\log_2 4})$

$= O(n^{\log_2 2^2})$

$= O(n^{2\log_2 2}) = O(n^2)$

$\boxed{\therefore \text{Ans}: O(n^2)}$

Assignment - 5

Question - 4

→ Consider $n = 10$
The perfect squares in the range 1 to 10 are 1, 4 and 9.

If the number is perfect square, cost is that number.
After completing the iteration,
the total cost can be calculated as.

number(i)		cost
1	→	1
2	→	0
3	→	0
4	→	4
5	→	0
6	→	0
7	→	0
8	→	0
9	→	9
10	→	0

The total cost is $1 + 4 + 9 = 14$

Using the aggregate analysis,

The amortized cost can be given as
$\frac{\text{total cost}}{n}$, $\because n =$ number of elements

$= \frac{14}{10} = 1.4$

$\boxed{\therefore \text{Ans} = 1.4}$

Generally,

In n elements, there are $k = \text{floor}(\sqrt{n})$

$\therefore$ The aggregate amortized cost can be given as

$$\left(\sum_{i=1}^{k} i^2\right) / n$$

$$= \boxed{\frac{k(2k+1)(k+1)}{6n}}$$