04-Report

June 17, 2020

1 Sorting: Homework

1.1 Exercise 3

Argue about the following statement and answer the questions

```
a) HEAP SORT on a array A whose length is n takes time O(n).
b) HEAP SORT on a array A whose length is n takes time Ω(n).
c) What is the worst case complexity for HEAP SORT?
d) QUICK SORT on a array A whose length is n takes time O(n^3).
e) What is the complexity of QUICK SORT?
f) BUBBLE SORT on a array A whose length is n takes time Ω(n).
g) What is the complexity of BUBBLE SORT?
```

Solutions:

a) By considering the pseudocode of `heap sort`, we know that `build_max_heap` has complexity $\Theta(n)$ and `extract_min` costs $\mathcal{O}\log(i)$ (please remember that `extract_min` is called $|A|-2$ times) so $T_h(n) = \Theta(n)+\sum_{i=2}^{n}\mathcal{O}\log(i) \leq \mathcal{O}(n)+\mathcal{O}(\sum_{i=2}^{n}\log n)$ so overall complexity of heap sort is $\mathcal{O}(n\log n)$ which makes statement wrong.

b) Worst case of heap sort is $\Omega(n\log n)$ proof. Therefore this statement wrong as well. There is only one case where a and b would be true where complexity of `extract_min` is $\Theta(1)$ (best case).

c) Worst case complexity of Heap Sort is the when `extract_min` called n times. Calculations are already done in point a. $\Theta(n)+\sum_{i=2}^{n}\mathcal{O}\log(i) \leq \mathcal{O}(n)+\mathcal{O}(\sum_{i=2}^{n}\log n) = \mathcal{O}(n\log n)$

d) `Quicksort` may always has the worst unbalanced partitions possible. By summing all the partitioning times for each level and using the arithmetic series we can see that worst-case running time is $\Theta(n^2)$. `Quicksort` best occurs when partitions are evenly balanced as possible and in this case complexity is $\Theta(n\log n)$. Since $\mathcal{O}(n\log n) \in \mathcal{O}(n^3)$ we can say that it is true but it is always recommend to use tight boundaries since it may spawn bad performance analysis.

e) It is already stated in point d. Complexity of `Quicksort` is for worst-case $\Theta(n^2)$ and for best-case $\Theta(n\log n)$

f) `Bubble sort` has one swap-block which cost $\Theta(1)$ and nested for-loop costs $\Theta(i)$ so overall complexity is $\sum_{i=2}^{n}\Theta(i).\Theta(1) = \Theta(n^2)$. $\Omega(n^2) \in \Omega(n)$ so statement is true but is it stated before since the boundary is not tight it may spawn bad performance analysis.

g) As it is explained in point f complexity of `Bubble sort` is $\Theta(n^2)$.

1.2 Exercise 4

Solve the following recursive equation:

$$T(n) = \begin{cases} \Theta(1) & n = 32 \\ 3 * T(n/4) + \Theta(n^{3/2}) & otherwise \end{cases}$$

After setting recursion tree, I sum all the cost for each level and following equation is obtained

$$cn^{3/2} + 3c(\frac{n}{4})^{3/2} + 9c(\frac{n}{16})^{3/2} + ... + 3^i c(\frac{n}{4^i})^{3/2} \tag{1}$$

where i is the level of recursion tree. This equation will go until the base case which is $n = 32$, $(\Theta(1))$. However we can bound it with ∞. By doing this also I can use the properties of geometric series.

So we can rewrite the equation 1 as the following form with previous clarificaitons:

$$\begin{aligned} cn^{3/2}[1 + \frac{3}{4^{3/2}} + \frac{9}{16^{3/2}} + ... + \frac{3^i}{4^{3i/2}}] &\leq cn^{3/2} \sum_{i=0}^{\infty} (\frac{3}{4^{3/2}})^i & (2) \\ &= cn^{3/2} \sum_{i=0}^{\infty} (\frac{3}{8})^i & (3) \\ &= cn^{3/2} \frac{8}{5} \in \mathcal{O}(n^{3/2}) & (4) \end{aligned}$$

Hence we have $T(n) \in \mathcal{O}(n^{3/2})$