Assignment 1: CS21003 Algorithms 1

Prof. Partha Pratim Chakrabarti and Palash Dey
Indian Institute of Technology, Kharagpur

Submit by 11:59 PM of January 26, 2022

1. Compute asymptotic complexity of $T(n)$ in terms of Θ for the following recurrence.

$$T(n) = \begin{cases} T\left(\lceil n^{\frac{1}{4}} \rceil\right) + T\left(\lceil n^{\frac{6}{11}} \rceil\right) + 11 \log n & \text{if } n \geqslant 518 \\ 1 & \text{otherwise} \end{cases}$$

[4 Marks]

Solution sketch. Putting $n = 2^h$ and ignoring ceilings,

$$T\left(2^h\right) = T\left(2^{\frac{h}{4}}\right) + T\left(2^{\frac{6h}{11}}\right) + 11h$$

Defining $S(h) = T\left(2^h\right)$,

$$S(h) = S\left(\frac{h}{4}\right) + S\left(\frac{6h}{11}\right) + 11h$$

Solving the above recurrence relation using substitution method, $S(h) = \Theta(h)$ Thus we have

$$T(n) = T\left(2^h\right) = S(h) = \Theta(h) = \Theta(\log n)$$

□

2. Prove or disprove:

   (a) There exists a constant $c > 1$ such that $(\log_2 n)! = \mathcal{O}\left(n^c\right)$

   Solution sketch. We have the following.

$$(\log_2 n)! \geqslant \left(\frac{\log_2 n}{2}\right)^{\frac{\log_2 n}{2}} = 2^{\frac{\log_2 n}{2}(\log_2 \log_2 n - 1)}$$

   On the other hand, we have the following for any constant $c > 1$.

$$n^c = 2^{c \log_2 n}$$

   Hence, the statement is false. □

   (b) If $k \log k = \Theta(n)$ then $k = \Theta\left(\frac{n}{\log n}\right)$

[2+2 Marks]

   Solution sketch. We have the following.

$$\frac{n}{\log n} = \Theta\left(\frac{k \log k}{\log k - \log \log k}\right) = \Theta\left(\frac{k \log k}{\log k}\right) = \Theta(k)$$

□

3. Find the fallacy: $2^n = \mathcal{O}(2^{n-1}) = \mathcal{O}(2^{n-2}) = \cdots = \mathcal{O}(1)$

[2 Marks]

Solution sketch. The formula $\mathcal{O}(f_1 \cdots f_k) = \mathcal{O}(f_1) \cdots \mathcal{O}(f_k)$ holds only for finite k. In particular, the above formula does not hold for arbitrarily growing non-constant (equivalently countably infinite) k. □

4. There are n food items to be cooked. Two different cookers H_1 and H_2 are available. Each food item f_i requires $h1_i$ and $h2_i$ times to be cooked on cookers H_1 and H_2 respectively. We are to develop a cooking plan so that the total time to cook all the n food items is minimized.

   (a) Develop a recursive definition for the problem.

   (b) Prove the correctness of your algorithm.

   (c) Analyze the complexity of your algorithm based on (a).

   (d) Develop the recursion tree and present its properties.

   (e) Choose the way to develop your algorithm, justifying your choices.

   (f) Decide on the data structures.

   (g) Present the final algorithm.

   (h) Analyze its complexity.

   (i) Show the working on (a), (d) and (h) on a non-trivial example of 10 food items.

   (j) Would your algorithm change if H_2 required double the time on each food item compared to cooker H_1? Explain in details and justify your answer both logically, analytically as well as with illustrative examples.

[10 Marks]