CSE 551 Assignment 1 Solutions

January 24th, 2021

Submission Instructions: Deadline is **11:59pm on 01/31**. Late submissions will be penalized, therefore please ensure that you submit (file upload is completed) before the deadline. Additionally, you can download the submitted file to verify if the file was uploaded correctly. Submit your answers electronically, in a single PDF, via *Canvas*. **Please type up the answers and keep in mind that we'll be checking for plagiarism**.
Furthermore, please note that the graders will grade 2 out of the 4 questions randomly. Therefore, if the grader decides to check questions 1 and 4, and you haven't answered question 4, you'll lose points for question 4. Hence, please answer all the questions.

1. Prove of disprove the following with valid arguments: **(5+5+5+5+5)**

   Solution:
   (i) $n! \in O(n^n)$.

   $n! \in O(n^n)$ implies,
   $n! \leq c \times n^n$, (using the definition of Big-O)
   $n \times (n-1) \times ... \times 1 \leq c \times n^n$,
   Dividing both sides by n^n, we get,
   $\frac{n}{n^n} \times \frac{(n-1)}{n^n} \times ... \times 1 \leq c$

   As $n \rightarrow \infty$, all the terms on the LHS $\rightarrow 0$. Therefore, there exists a constant $c > 0$, which will satisfy $n! \leq c \times n^n$. Therefore, **TRUE** that $n! \in O(n^n)$.

   (ii) $2n^2 2^n + nlog(n) \in \Theta(n^2 2^n)$.

   We have to show that $2n^2 2^n + nlog(n) \in \Omega(n^2 2^n)$ and $2n^2 2^n + nlog(n) \in O(n^2 2^n)$, in order to show that $2n^2 2^n + nlog(n) \in \Theta(n^2 2^n)$.

   Case 1: $2n^2 2^n + nlog(n) \in \Omega(n^2 2^n)$
   $2n^2 2^n + nlog(n) \geq c_1 \times n^2 2^n$, using the definition for Big-Omega,

Dividing both sides by $n^2 2^n$, we get,
$2 + \frac{nlogn}{n^2 2^n} \geq c_1$,
$2 + \frac{logn}{n2^n} \geq c_1$,

As $n \rightarrow \infty$, $\frac{logn}{n2^n} \rightarrow 0$. Therefore by selecting $c_1 = 1$, we can satisfy the Big-Omega inequality. Therefore, $2n^2 2^n + nlog(n) \in \Omega(n^2 2^n)$.

Case 2: $2n^2 2^n + nlog(n) \in O(n^2 2^n)$
$2n^2 2^n + nlog(n) \leq c_2 \times n^2 2^n$, using the definition for Big-O,
Dividing both sides by $n^2 2^n$, we get,
$2 + \frac{nlogn}{n^2 2^n} \leq c_2$,
$2 + \frac{logn}{n2^n} \leq c_2$,

As $n \rightarrow \infty$, $\frac{logn}{n2^n} \rightarrow 0$. Therefore by selecting $c_2 \geq 2$, we can satisfy the Big-O inequality. Therefore, $2n^2 2^n + nlog(n) \in O(n^2 2^n)$. Hence, **TRUE** that $2n^2 2^n + nlog(n) \in \Theta(n^2 2^n)$.

(iii) $10n^2 + 9 = O(n)$.
$10n^2 + 9 \leq c \times n$,
Dividing both sides by n, we get,
$10n + \frac{9}{n} \leq c$,

Since the LHS is a a function of n, we cannot find a constant which will satisfy the above inequality. Therefore, **FALSE** that $10n^2 + 9 = O(n)$.

(iv) $n^2 log(n) = \Theta(n^2)$.
Case 1: $n^2 log(n) = \Omega(n^2)$
$n^2 log(n) \geq c_1 \times n^2$
$log(n) \geq c_1$, after dividing both sides by n^2,
Selecting $c_1 = 1$ for $n > 1$ will satisfy the above equation. Therefore $n^2 log(n) = \Omega(n^2)$.

Case 2: $n^2 log(n) = O(n^2)$
$n^2 log(n) \leq c_2 \times n^2$
$log(n) \leq c_2$, after dividing both sides by n^2,

Since the LHS is a a function of n, we cannot find a constant which will satisfy the above inequality. Therefore, $n^2 log(n) \neq O(n^2)$ and by extension, **FALSE** that $n^2 log(n) = O(n^2)$.

(v) $n^3 2^n + 6n^2 3^n = O(n^3 2^n)$.
$n^3 2^n + 6n^2 3^n \leq c \times n^3 2^n$,
$1 + 6 \times \frac{n^2 3^n}{n^3 2^n} \leq c$,
$1 + 6 \times \frac{3^n}{n 2^n} \leq c$,
$1 + 6 \times \frac{1}{n} \times (\frac{3}{2})^n \leq c$,

The RHS is a function of n, therefore we cannot find a constant. Therefore, **FALSE** that $n^3 2^n + 6n^2 3^n = O(n^3 2^n)$.

2. Suppose that you have algorithms with the size running times listed below.Assume that these are the exact number of operations performed as a function of the input size n. Suppose you have a computer that can perform 10^{10} operations per second, and you need to compare a result in at most an hour of computation. For each of the algorithms, what is the largest input size n for which you would be able to get the result within an hour? **(6+6+6+7)**

   Solution: Number of operations done in 1 hour = 60 * 60 * 10^{10}.
   (i) n^2.
   Solving for $n^2 = 60 * 60 * 10^{10}$, we get,
   $n = 60 \times 10^5$.

   (ii) n^3.
   Solving for $n^3 = 60 * 60 * 10^{10}$, we get,
   $n = 33019$.

   (iii) $50n^2$.
   Solving for $50n^2 = 60 * 60 * 10^{10}$, we get,
   $n = 848528$.

   (iv) 3^n.
   Solving for $3^n = 60 * 60 * 10^{10}$, we get,
   $n = 28$.

3. Algorithm A_1 takes $10^{-3} \times 2^n$ seconds to solve a problem instance of size n and Algorithm A_2 takes $10^{-2} \times n^4$ seconds to do the same on a particular machine. **(8+8+9)**

   Solution:

(i) What is the size of the largest problem instance A_2 will be able solve in one year ?

Ans: Amount of seconds in a year = 365 * 24 * 60 * 60 = 31536000.

Now, we solve for $10^{-2} \times n^4 = 31536000$

$n = 236.97$

Now since the size of a problem instance cannot be fractional, the largest problem instance the algorithm will *finish* solving is 236.

(ii) What is the size of the largest problem instance A_2 will be able solve in one year on a machine one hundred times as fast ?

Ans: On a machine 100x fast, we have
$10^{-2} \times n^4 \times 10^{-2} = 31536000$

$n = 749.37$
Or, $n = 749$

(iii) Which algorithm will produce results faster, in case we are trying to solve problem instances of size less than 20?

Ans: Problem instance of size less than 20 implies $n = 19$. Plugging in the value of n in the amount of time taken for algorithm A_1 we get, $10^{-3} \times 2^n = 10^{-3} \times 2^{19} = 10^{-3} \times 524288 = 524.28s$. Algorithm A_2 will take $10^{-2} \times n^4 = 10^{-2} \times 19^4 = 10^{-2} \times 130321 = 1303.21s$. The process yields similar results for values $1 \leq n \leq 18$. Therefore, algorithm A_1 is faster for instances less than 20.

4. Take the following list of functions and arrange them in ascending order of growth rate. That is, if function g(n) immediately follows f(n) in your list, then it should be the case that f(n) is O(g(n)). **(25)**

(i) $f_1(n) = n^{4.5}$.
(ii) $f_2(n) = (3n)^{0.6}$.
(iii) $f_3(n) = n^4 + 20$.
(iv) $f_4(n) = 25^n$.
(v) $f_5(n) = 260^n$

Solution: $f_2 < f_3 < f_1 < f_4 < f_5$