Growth of functions (Ω, O, Θ, o, ω) - Solution

Last updated: 2/27/2024

P1 (MC) For all the questions below (except for the True or False questions), the **answer can be none, one, some or all of the choices**. Write your answers on the **LEFT** side. **No justification needed.** (3 points each question)

a) If *f(N) = O(g(N))*, then *f(N) = Θ(g(N))*. True or False.

b) Mark all answers that are correct for this summation: 1+2+3 …+i +…+ N
   A) $\Theta(2^N)$ **B) $\Omega(\lg(N)*\lg(N))$** **C)** O(N) **D)** $O(N\sqrt{N})$ **E)** none of the these
   Justification: It is $\Theta(N^2)$ and $N^2 = \Omega(\lg(N)*\lg(N))$ => B

c) Give a function f(N) (other than N^3) that is $O(N^3)$: f(N) = … **N^2**……………………

d) Which of the following is **always** a correct description of the time complexity of the code below (**regardless** of what `someFunction` does)?

A. Θ(N) B. O(N) C. Ω(N) D. O(NlgN)

Justification: See Problem P2a) for a similar problem and its justification.

```
int k;
for(k=1; k <= N; k++) {
  someFunction(k);
}
```

e) You are given the option to choose one of three algorithms with time complexities:

A. $\Theta(N^2)$ B. $O(N^2)$ C. $\Omega(N^2)$

You want to choose the algorithm most likely to be the fastest (takes less time). Which one will you choose?

Justification:

Algorithm with TC $\Theta(N^2)$ will take time **"exactly"** proportional to N^2.

Algorithm with TC $O(N^2)$ will take time **at most** proportional to N^2. It could be N^2, but it could be less (e.g. N or NlgN).

Algorithm with TC $\Omega(N^2)$ will take time **at least** proportional to N^2. It could be N^2, but it could be more (e.g. N^3 or N^4lgN).

Therefore the algorithm with TC $O(N^2)$ is the only one that has a chance to take time less than N^2.

P2.

a) What can you tell about the time complexity of the code below (**regardless** of what `someFunction` does)? Give a lower, upper or tight bound (using **Ω, O,** or **Θ**). **Justify your answer.**

```
int k;
for(k=1; k <= N; k++) {
    someFunction(N);
}
```

$\Omega(N)$ ***Justification:***

someFct(N) has at least a constant number of instructions, but could be even be exponential in N. So we can only say that it has $\Omega(1)$ => TC1iter(k) = $\Omega(1)$, indep , no var change, N iterations => the code has N* $\Omega(1) = \Omega(N)$

b) What can you tell about the time complexity of the code below (**regardless** of what `someFunction` does)? Give a lower, upper or tight bound (using **Ω, O,** or **Θ**). **Justify your answer.**

```
int k;
for(k=1; k <= N; k++) {
    return someFunction(100);
}
```

$\Theta(1)$ ***Justification:***

Since someFct(100) takes a constant as an argument, whatever TC it has, it becomes a constant for this function call (e.g. 100^2), therefore

someFunction(100) has TC $\Theta(1)$ => TC1iter(k) = $\Theta(1)$, indep , no var change,

Because of the "return", the code returns in the first iteration, therefore the for-k loop will stop after the first iteration. => it does only 1 repetition =>

TC of code is reps * $TC_{1iter}(k)$ = 1 repetition * $\Theta(1) = \Theta(1)$

P3. $5N^3 + N^2 = O(N^3)$ True or False? Justification not needed, but provided here for your information:

The limit of f(N)/g(N) should be 0 or a constant.

$$\lim_{N\to\infty}\frac{f(N)}{g(N)} = \lim_{N\to\infty}\frac{5N^3+N^2}{N^3} = \lim_{N\to\infty}\frac{5N^3}{N^3} + \lim_{N\to\infty}\frac{N^2}{N^3} = \lim_{N\to\infty} 5 + \lim_{N\to\infty}\frac{1}{N} = 5 + 0 = 5$$

P4. $5N^3 + N^2 = \Theta(N^3)$ True or False? Justification not needed, but provided here for your information:

The limit of f(N)/g(N)should be a *non-zero* constant.

$$\lim_{N\to\infty}\frac{f(N)}{g(N)} = \lim_{N\to\infty}\frac{5N^3+N^2}{N^3} = \lim_{N\to\infty}\frac{5N^3}{N^3} + \lim_{N\to\infty}\frac{N^2}{N^3} = \lim_{N\to\infty} 5 + \lim_{N\to\infty}\frac{1}{N} = 5+0 = 5 \neq 0$$

P5. 500lgN= Θ(N) True or False? Justification not needed, but provided here for your information:

The limit of f(N)/g(N) = lim ((lgN)/N) = 0. For Theta it should have been a *non-zero* constant.

$$\lim_{N\to\infty}\frac{f(N)}{g(N)} = \lim_{N\to\infty}\frac{lgN}{N} = \lim_{N\to\infty}\frac{\frac{lnN}{\ln(e)}}{N} = \frac{1}{lne}\lim_{N\to\infty}\frac{(lnN)'}{(N)'} = \frac{1}{lne}\lim_{N\to\infty}\frac{\frac{1}{N}}{1} = \frac{1}{lne}\lim_{N\to\infty}\frac{1}{N} = 0$$

P6. Consider the function lgN + 300 . Select all options below that are also true about this function.

For example if you select $O(N^2)$, you are saying that this function is $O(N^2)$.

a. O(1) b. O(lgN) c. O(N) f. $O(N^2)$

e. Θ(1) f. Θ(lgN) g. Θ(N) h. $\Theta(N^2)$

i. Ω(1) j. Ω(lgN) k. Ω(N) l. $\Omega(N^2)$

Justification:

lg($\sqrt{N}$) + 300 + lgN = Θ(lgN) because it has dominant term lgN. => f

Because it is Θ(lgN) it is also O(lgN) and Ω(lgN) => b, j

Ω(1) means **at least** constant growth. Our function has growth proportional to lg(N) therefore it is true (it is correct to say) it has at least constant growth : lg($\sqrt{N}$) + 300 + lgN = Ω(1) => i

O means **at most** that much growth. Our function has growth proportional to lg(N) therefore it is true (it is correct to say) it has :

at most N growth : lg($\sqrt{N}$) + 300 + lgN = O(N) => c

at most N^2 growth : lg($\sqrt{N}$) + 300 + lgN = $O(N^2)$ => d

Extra problems, not part of any examination.

Extra1. Let $T(N) = \sum_{k=0}^{N}\left(\frac{5}{7}\right)^k = \left(\frac{5}{7}\right)^0 + \left(\frac{5}{7}\right)^1 + \left(\frac{5}{7}\right)^2 + \ldots + \left(\frac{5}{7}\right)^N$. To which of the sets below does *T(N)* belong?

A. $\Theta(1)$ (see summation slides) B. $\Theta(N)$ C. $\Theta(N^2)$ D. $\Theta(N\lg N)$ E. $\Theta(\lg N)$

Extra2. Given summation: $1 + 2^6 + 3^6 + \ldots + N^6$ Can you solve this in terms of Θ, Ω or O ?

ANS: Yes, Θ. With Approximation by integrals we get $\Theta(N^7)$:

$$1 + 2^6 + 3^6 + \cdots + N^6 = \sum_{k=1}^{N} k^6 = \Theta\big(F(N) - F(1)\big) = \Theta\big(F(N)\big) = \Theta\left(\frac{N^7}{7}\right)$$
$$= \Theta(N^7) \ \ where \ \ F(N) = \int_0^N x^6 dx$$

Extra3. – Hard – for math-lovers.

Suppose that f(N) > 0 for all N >= 0. Suppose that g(N) = f(N)/2 + $\sqrt{N}$. For each of the following, specify if it is "**definitely true**", "**definitely false**", or "**possibly true and possibly false**". **Justify** your answer (using limits or other properties). If you answer "possibly true and possibly false", provide at least one specific example of f(N) that makes the answer "true" and one specific example of f(N) that makes the answer "false".

Look at the limit g/f (because it is easier to look at than f/g).

lim (g(N)/f(N) = lim[f(N)/2 + sqrt(N)]/f(N) = ½+lim(sqrt(N)/f(N)) This limit can be a constant or infinity => f(N) = O(g(N)).

When lim(sqrt(N)/f(N)) is a constant they are Θ of each other (f(N) is the dominant term for both).

When lim(sqrt(N)/f(N)) is infinity, f(N) = o(g(N)) .

a) f(N) = O(g(N)) Definitely true. Proven above.

b) f(N) = Θ(g(N))

Case for true: f(N) = sqrt(N) Limit of g/f is 3/2

Case for false: f(N) = 10 = o(5+sqrt(N)) limit of g/f is inf.

c) f(N) = Ω(g(N))

Case for true: f(N) = f(N) = sqrt(N) Limit of f/g is 2/3 (same case that makes them Θ)

Case for false: f(N) = 10 = O(5+sqrt(N))