Advanced information, computation, communication I
EPFL - Fall semester 2020-2021

Week 7 - solutions

November 3, 2020

Exercise 1. To show that $f(x)$ is $o(g(x))$ we need to show that

$$\lim_{x\to\infty} \frac{f(x)}{g(x)} = 0.$$

On the other hand, to show that $f(x)$ is not $o(g(x))$ we need to show that the limit above is non zero.

1. *Show that $5x$ is $o(x^2)$.*
$$\lim_{x\to\infty} \frac{5x}{x^2} = \lim_{x\to\infty} \frac{5}{x} = 0$$
2. *Show that $2x^2$ is not $o(x^2)$.*
$$\lim_{x\to\infty} \frac{2x^2}{x^2} = \lim_{x\to\infty} \frac{2}{1} = 2 \neq 0$$
3. *Show that $1/x$ is $o(x)$.*
$$\lim_{x\to\infty} \frac{\frac{1}{x}}{x} = \lim_{x\to\infty} \frac{1}{x^2} = 0$$
4. *Show that if $f(x)$ is $o(g(x))$, then $f(x)$ is $O(g(x))$.*

   If $f(x)$ is $o(g(x))$, then $\lim_{x\to\infty} \frac{f(x)}{g(x)} = 0$. I.e.
$$\forall \epsilon > 0\ \exists \delta > 0 \quad \forall x > \delta \quad \left|\frac{f(x)}{g(x)}\right| < \epsilon$$
$$\iff \quad \forall \epsilon > 0\ \exists \delta > 0 \quad \forall x > \delta \quad |f(x)| < \epsilon |g(x)|.$$
   Choose any $\epsilon > 0$ and a corresponding δ such that the above inequality holds. Then
$$\forall x > \delta \quad |f(x)| < \epsilon |g(x)|$$
   i.e. ϵ and δ are the witnesses to show that $f(x)$ is $O(g(x))$.

Exercise 2. *Which function below grows fastest when n goes to infinity?*

- ✓ $(\log_3(33))^{n-3}$
- ◯ 3^n
- ◯ $n^{3\log_3(n)}$
- ◯ $n^3 \log_3(n)$

Because $3 = \log_3(3^3) = \log_3(27) < \log_3(33)$ it follows that

$$\frac{(\log_3(33))^n}{3^n} = \left(\frac{\log_3(33)}{3}\right)^n$$

goes to infinity for $n \to \infty$. Because the constant factor $(\log_3(33))^{-3}$ does not affect the growth rate, it follows that $(\log_3(33))^{n-3}$ grows faster than 3^n.

Exercise 3. *Consider the two statements below, where c and k are constants with $k \geq 2$:*

$$n^k \textit{ is } O(k^n) \qquad\qquad (\log n)^k e^{(c+o(1))(\log n)^{1/3}(\log(\log(n)))^{2/3}} \textit{ is } e^{(c+o(1))(\log n)^{1/3}(\log(\log(n)))^{2/3}}.$$

- ○ *They are both False.*
- ○ *Only the first is True.*
- ○ *Only the second is True.*
- ✓ *They are both True.*

To easily (and informally) see that n^k is $O(k^n)$, we write n as $k^{\log_k(n)}$ and n^k as $k^{k\log_k(n)}$, so that $\frac{k^n}{n^k} = \frac{k^n}{k^{k\log_k(n)}} = k^{n-k\log_k(n)}$; with $n - k\log_k(n)$ going to ∞ for $n \to \infty$ and $k \geq 2$, it follows that $\frac{k^n}{n^k}$ goes to ∞ for $n \to \infty$.

Just looking at the formulas for the second problem, one "immediately" sees that the factor $(\log n)^k$ gets swallowed up by the $o(1)$ in the exponent, due to the presence of the "more powerful" $(\log n)^{1/3}$ $(\log(\log(n)))^{2/3}$ in the exponent. The cumbersome details are much less complicated than they look, but can be found below.

First simplify the two expressions by replacing $\log(n)$ by m, while writing (as customary) $\exp(x)$ for e^x:

$$m^k \exp\big((c+o(1))m^{1/3}(\log(m))^{2/3}\big) \text{ is } \exp\big((c+o(1))m^{1/3}(\log(m))^{2/3}\big).$$

As above, write $m^k = \exp(k\log(m))$, so $m^k \exp\big((c+o(1))m^{1/3}(\log(m))^{2/3}\big)$ becomes $\exp\big(k\log(m) + (c+o(1))m^{1/3}(\log(m))^{2/3}\big)$ which becomes

$$\exp\left(\left(c + \left(\frac{k\log(m)}{m^{1/3}(\log(m))^{2/3}}\right) + o(1)\right) m^{1/3}(\log(m))^{2/3}\right)$$

when using $k\log(m) = \left(\frac{k\log(m)}{m^{1/3}(\log(m))^{2/3}}\right) m^{1/3}(\log(m))^{2/3}$. For n and thus m going to ∞, the term $\left(\frac{k\log(m)}{m^{1/3}(\log(m))^{2/3}}\right)$ goes to zero and is thus $o(1)$; with $o(1) + o(1) = o(1)$ the result follows.

Exercise 4. *Let f be arbitrary functions from $\mathbf{N}$ to $\mathbf{R}_{>0}$.*

Let g_1, g_2 be two functions from $\mathbf{N}$ to $\mathbf{R}_{>0}$ such that g_1 and g_2 are both $\Theta(f)$.

1. *Show that the function $g_1 + g_2$ is $\Theta(f)$ or provide a counterexample.*

   The functions $g_i : \mathbf{N} \to \mathbf{R}_{>0}$, $i = 1, 2$ are $\Theta(f)$ for some function f, i.e. there exist $c_{i,j} > 0, k_i > 0$, for $j = 1, 2$, s.t.

   $$\forall x > k_i \quad c_{i,1}|f(x)| \leq |g_i(x)| \leq c_{i,2}|f(x)|.$$

   Let $k = \max\{k_i\}$. Then, for all $x > k$, $c_{1,1}|f(x)| \leq |g_1(x)| \leq c_{1,2}|f(x)|$ and $c_{2,1}|f(x)| \leq |g_2(x)| \leq c_{2,2}|f(x)|$, hence

   $$(c_{1,1} + c_{2,1})|f(x)| \leq |g_1(x)| + |g_2(x)| \leq (c_{1,2} + c_{2,2})|f(x)|.$$

   The triangle inequality tells us that $|g_1(x) + g_2(x)| \leq |g_1(x)| + |g_2(x)|$, hence with $c_2 := c_{1,2} + c_{2,2}$, we have that

   $$|g_1(x) + g_2(x)| \leq c_2|f(x)|.$$

   So we have shown that $g_1 + g_2$ is $O(f)$.

   On the other hand, since $g_i : \mathbf{N} \to \mathbf{R}_{>0}$, we have $|g_1(x) + g_2(x)| = |g_1(x)| + |g_2(x)| = g_1(x) + g_2(x)$. It follows that, for $c_1 := c_{1,1} + c_{2,1}$,

   $$c_1|f(x)| \leq |g_1(x)| + |g_2(x)| = |g_1(x) + g_2(x)|.$$

   So $g_1 + g_2$ is $\Omega(f)$.

   Overall, we have shown that $g_1 + g_2$ is $O(f)$ and $\Omega(f)$, therefore $g_1 + g_2$ is $\Theta(f)$.

2. *Show that the function $g_1 g_2$ is $\Theta(f^2)$ or provide a counterexample.*

   We use the same notation as in 1. and obtain that for all $x > k$

$$(c_{1,1}c_{2,1})f^2(x) \leq |g_1(x)| \cdot |g_2(x)| \leq (c_{1,2}c_{2,2})f^2(x).$$

   Let $c_1 := c_{1,1}c_{2,1}$ and $c_2 := c_{1,2}c_{2,2}$. We use the observation that $|g_1(x)g_2(x)| = |g_1(x)||g_2(x)|$. Then for all $x > k$,

$$c_1 f^2(x) \leq |(g_1 \cdot g_2)(x)| \leq c_2 f^2(x),$$

   i.e., $g_1 \cdot g_2$ is $\Theta(f^2)$.

Let g_3, g_4 be two functions from $\mathbf{N}$ to $\mathbf{R}$ such that g_3 and g_4 are both $\Theta(f)$.

3. *Show that the function $g_3 + g_4$ is $\Theta(f)$ or provide a counterexample.*

   Here the functions g_3, g_4 may take negative values. We can follow the reasoning in 1. to show that $g_3 + g_4$ is $O(f)$. But there exist g_3, g_4 s.t. $g_3 + g_4$ is not $\Omega(f)$. For instance, let $g_4(x) = -g_3(x)$. Then, $(g_3 + g_4)(x) = 0$ and so $\forall c > 0, \forall x > 0$ we have that $|g_3(x) + g_4(x)| = 0 < c|f(x)|$ for any function $f : \mathbf{N} \to \mathbf{R}_{>0}$. As a consequence $g_3 + g_4$ is not $\Omega(f)$ and the statement is false.

4. *Show that the function $g_3 g_4$ is $\Theta(f^2)$ or provide a counterexample.*

   Same proof as in 2., since we did not need the fact that $g_i : \mathbf{N} \to \mathbf{R}_{>0}$ in there.

Let g be a function from $\mathbf{N}$ to $\mathbf{R}_{>0}$ such that g is $O(f)$.

5. *Show that 2^g is $O(2^f)$, or provide a counterexample.*

   Take $g = 2n$ and $f = n$. We have that g is $O(f)$, but $2^g = 2^{2n}$ and $2^f = 2^n$ so that $2^g = (2^f)^2$ and 2^g is not $O(2^f)$.

Exercise 5. *Consider the two statements below, where k and ℓ are constants with $k > \ell \geq 2$ and $m \to \infty$:*

$$\log_m(k) \text{ is } \Theta(\log_m(\ell)) \qquad k^{\log_\ell(m)} \text{ is } O(\ell^{\log_k(m)}).$$

- ◯ *They are both false.*
- ✓ *Only the first is true.*
- ◯ *Only the second is true.*
- ◯ *They are both true.*

Because $\log_m(x) = \frac{\log_2(x)}{\log_2(m)}$ both $\log_m(k)$ and $\log_m(\ell)$ are of order $\frac{1}{\log_2(m)}$; in particular $\log_m(k)$ is $\Theta(\log_m(\ell))$.

Writing $k = \ell^{\log_\ell(k)}$ and $\log_k(m) = \frac{\log_\ell(m)}{\log_\ell(k)}$ the comparison for the second problem is between $k^{\log_\ell(m)} = \ell^{\log_\ell(k)\log_\ell(m)}$ and $\ell^{\log_k(m)} = \ell^{\log_\ell(m)/\log_\ell(k)}$. Because $k > \ell \geq 2$ we find that $k^{\log_\ell(m)} = \ell^{c\log_\ell(m)} = m^c$ for the constant $c = \log_\ell(k) > 1$ and that $\ell^{\log_k(m)} = \ell^{(\log_\ell(m))/c} = m^{1/c}$. It follows that $k^{\log_\ell(m)}$ is not $O(\ell^{\log_k(m)})$.

Exercise 6. *Consider the following two statements:*

$$\Big(f \text{ is } o(f)\Big) \quad \text{and} \quad \Big(f \text{ is } o(g) \text{ implies } f \text{ is } O(g)\Big).$$

- ✓ *Only the second is true.*
- ◯ *They are both false.*
- ◯ *Only the first is true.*

◯ *They are both true.*

The statement "f if $o(f)$" would imply that for any function f it is the case that $\lim_{x\to\infty} \frac{|f(x)|}{|f(x)|} = 0$. That is cleary incorrect: for instance for the function $f(x) = 1$ it is the case that $\lim_{x\to\infty} \frac{|f(x)|}{|f(x)|} = \frac{1}{1} = 1$. Thus the first statement is not correct.

The statement "f is $o(g)$" implies that $\lim_{x\to\infty} \frac{|f(x)|}{|g(x)|} = 0$, and thus that for any $\epsilon > 0$ there exists an x_0 such that $\frac{|f(x)|}{|g(x)|} < \epsilon$ for all $x > x_0$, implying that $|f(x)| < \epsilon|g(x)|$ for all $x > x_0$, which in turn implies that f is $O(g)$. Thus the second statement is correct.

Exercise 7. *Given the two statements below, where $d > 0$ is an integer constant and a_i for all $i \in \mathbf{Z}$ are positive integers with $\max_{i\in\mathbf{Z}}(a_i) = D$ for a constant $D > 0$,*

$$\sum_{i=0}^{n} a_i i^d \text{ is } \Theta(n^{d+1}) \qquad\qquad \sum_{i=0}^{d} a_i n^i \text{ is } \Theta(n^d)$$

✓ *They are both true.*

◯ *Only the first is true.*

◯ *Only the second is true.*

◯ *They are both false.*

The first summation is a sum of d-th powers which behaves like a $(d+1)$-st power of the summation bound, the second is just a polynomial of degree d and thus behaves like its highest order term.

Exercise 8. *Construct two functions f and g from $\mathbf{N}$ to $\mathbf{R}_{>0}$ such that f is not $O(g)$ and g is not $O(f)$ or prove that such functions are impossible to find.*

For instance, consider the functions defined for any $n \in \mathbb{N}$ by

$$f(n) = \begin{cases} n! & \text{if } n \text{ is even,} \\ (n-1)! & \text{if } n \text{ is odd.} \end{cases} \quad \text{and } g(n) = \begin{cases} (n-1)! & \text{if } n \text{ is even,} \\ n! & \text{if } n \text{ is odd.} \end{cases}$$

We have that f is not $O(g)$ because for any constant α, one can find an even integer $n > \alpha$ to obtain $f(n) = n! > \alpha \cdot (n-1)! = \alpha g(n)$. Similarly, g is not $O(f)$.

Exercise 9. *Consider the following algorithm, which takes as input a sequence of n integers $a_1, a_2, ..., a_n$ and produces as output a matrix $M = \{m_{ij}\}$ where m_{ij} is the minimum term in the sequence of integers $a_i, a_{i+1}, ..., a_j$ for $j \geq i$ and $m_{ij} = 0$ otherwise.*

```
initializ M so that m_ij = a_i if j ≥ i and m_ij = 0 otherwise
for i := 1 to n
  for j := i + 1 to n
    for k := i + 1 to j
      m_ij := min(m_ij, a_k)
    end for
  end for
end for
return M = {m_ij} {m_ij is the minimum term of a_i, a_{i+1}, ..., a_j}
```

1. *Show that this algorithm uses $O(n^3)$ comparisons to compute the matrix M.*
   The algorithm only makes comparisons in the line "$m_{ij} := min(m_{ij}, a_k)$" (since determining the minimum is a comparison). Thus 1 comparison is made in each iteration of the three for-loops.

   i can take on the values 1 to n (**for** $i := 1$ **to** n), thus i can take on n values.

   j can take on the values $i+1$ to n (**for** $j := i+1$ **to** n), thus j can take on $n-i$ values, which is at most $n-1$ values (when $i=1$).

k can take on the values $i+1$ to j (**for** $k := i+1$ **to** j), thus j can take on $j-i$ values, which is at most $n-1$ values (when $i=1$ and $j=n$).

The total number of comparisons is then the product of the (maximum) number of values for i, j and k in the for-loops.

$$\begin{aligned}\text{Number of comparisons} &= n \times (n-1) \times (n-1) \\ &= n(n-1)^2 \\ &= n(n^2-2n+1) \\ &= n^3-2n^2+n\end{aligned}$$

Thus the number of comparisons is n^3-2n^2+n, while $^3-2n^2+n$ is $O(n^3)$.

2. *Show that this algorithm uses $\Omega(n^3)$ comparisons to compute the matrix M. Using this fact and part (a), conclude that the algorithms uses $\Theta(n^3)$ comparisons. [Hint: Only consider the cases where $i \le \frac{n}{4}$ and $j \ge \frac{3n}{4}$ in the two outer loops in the algorithm.]*
   The number of comparisions is n^3-2n^2+n, while $^3-2n^2+n$ is $\Omega(n^3)$.
   Since the number of comparisons is $O(n^3)$ and $\Omega(n^3)$, the number of comparisons is also $\Theta(n^3)$.

Exercise 10. *What is the largest n for which one can solve within a minute using an algorithm that requires $f(n)$ bit operations, where each bit operation is carried out in 10^{-12} seconds, with these functions $f(n)$?*

a. *log n*
   Each bit operation is carried out in 10^{-12} seconds: $T = 10^{-12}$ seconds.
   The algorithm can take at most 1 minute which contains 60 seconds, while there are $\frac{t}{T} = \frac{60}{10^{-12}} = 60 \times 10^{12}$ possible bit operations in 60 seconds.
   Algorithm requires $f(n) = log\ n$ bit operations:

$$log\ n = 60 \times 10^{12}$$

   Note: The logarithm has base 2, because bits only have 2 possible values.

$$log_2\ n = 60 \times 10^{12}$$

   Let us take the exponential with base 2 of each side of the previous equation:

$$n = 2^{60 \times 10^{12}}$$

b. $1,000,000n$
   Each bit operation is carried out in 10^{-12} seconds: $T = 10^{-12}$ seconds.
   The algorithm can take at most 1 minute which contains 60 seconds, while there are $\frac{t}{T} = \frac{60}{10^{-12}} = 60 \times 10^{12}$ possible bit operations in 60 seconds.
   Algorithm requires $f(n) = 1,000,000n$ bit operations:

$$\begin{gathered}1,000,000n = 60 \times 10^{12} \\ n = 60 \times 10^6 = 60,000,000\end{gathered}$$

c. n^2
   Each bit operation is carried out in 10^{-12} seconds: $T = 10^{-12}$ seconds.
   The algorithm can take at most 1 minute which contains 60 seconds, while there are $\frac{t}{T} = \frac{60}{10^{-12}} = 60 \times 10^{12}$ possible bit operations in 60 seconds.
   Algorithm requires $f(n) = n^2$ bit operations:

$$n^2 = 60 \times 10^{12}$$

   Take the square root of each side of the previous equation:

$$n = \sqrt{60 \times 10^{12}} \approx 7.745967 \times 10^6 = 7{,}745{,}967$$