CS 317, Assignment 1

Answers

1. Consider the following algorithm which receives as input an array A of size n:

   $i \leftarrow 1$
   while $i \leq n$ **do**
   $\quad sum_i \leftarrow 0$
   $\quad prod_i \leftarrow 1$
   $\quad$**for** $j = 1$ TO n **do**
   $\quad\quad sum_i \leftarrow sum_i + A_j$
   $\quad$**for** $j = 1$ TO n **do**
   $\quad\quad prod_i \leftarrow prod_i + A_j$
   $\quad i \leftarrow i + 1$

   - State how many times each loop is executed and justify your answer.
   - Give the running time of the algorithm in Θ notation. Explain how you reached the answer.

Answer:

The while loop executes n times since i starts at 1, is incremented at each step, and the loop keeps iterating as long as $i \leq n$ (that is the loop condition). Both of the inner for loops iterates exactly n times; that is how for loop work.

For the running time, let's take assignment as the operation to be counted. Then the body of the while loop features 3 assignments and the two for loops. Each for loop takes time n as argued above, for an overall running time of $3 + n$. The while loop in turn iterates n times as explained above, for an overall running time of $n(n+3) = n^2 + 3n = \Theta(n^2)$.

---

2. Consider the following algorithm which receives as input two numbers m and n and sets $result$ to $m \times n$; `div` is the integer division operator.

   $result \leftarrow 0$
   repeat
   $\quad$**if** m IS ODD **then**
   $\quad\quad result \leftarrow result + n$
   $\quad m \leftarrow m$ `div` 2
   $\quad n \leftarrow n + n$
   until $m < 1$:

- State (as a function of m and n) how many times the loop is executed and justify your answer.
- Give the running time of the algorithm in Θ notation as a function of m and n. Explain how you reached the answer.

ANSWER:

Let m_k be the value of m at the end of iteration k of the loop, with $m_0 = m$ the value before the loop starts executing. Given the assignment to m inside the loop we have $m_k = m_{k-1}/2 = (m_{k-2}/2)/2 = ((m_{k-3}/2)/2)/2 = \cdots$. We hypothesize that $m_k = m_{k-i}/2^i$ and so $m_k = m_{k-k}/2^k = m_0/2^k = m/2^k$. We verify this by induction over k as follows[1]: For $k = 0$ we have $m_0 = m/2^0 = m/1 = m$, as desired. Now $m_{k+1} = m_k/2$ and $m_k = m/2^k$ by inductive hypothesis. Therefore $m_{k+1} = m/2^k/2 = m/2^{k+1}$, again as desired.

Now the loop iterates as long as $m_k \geq 1$ that is, $m/2^k \geq 1$ that is, $m \geq 2^k$ or equivalently $k \leq \log m$. It follows that the loop iterates $\log m$ times.

How many operations do we have inside the loop? We have either 3 or 2 of those, depending on the actual value of the input m. Clearly we can choose an m such that we always perform 3 operations per loop and so the worst-case running time is $3 \log m = \Theta(\log m)$. Actually, in the best case (when $m = 2^k$ for some positive k) the running time would be $2 \log m$ which is still $\Theta(\log m)$ just like in the worst case.

Note in passing that the running time does not depend on n, but only in m.

---

3. Prove each of the following statements without using limits. Justify your answer.

   (a) $3n^3 + 2n^2 + n \in \Omega(n^2)$

   ANSWER:

   We need to show that $3n^3 + 2n^2 + n \geq cn^2$ for some constant c and large enough n. If we choose $c = 2$ the relation becomes $3n^3 + n \geq 0$ which is clearly true for any $n \geq 0$.

   (b) $2^n \in \Theta(2^{n-2})$

   ANSWER:

   We need to show that $2^{(n-1)} \in O(2^{n-2})$ and also $2^{(n-1)} \in \Omega(2^{n-2})$.
   To show that $2^n \in O(2^{n-2})$ we need to find a constant c such that $2^n \leq c2^{n-2}$ for large enough n. For $c = 4$ the relation becomes $2^n \leq 2^n$, clearly true for any $n \geq 0$.

[1] By now we all know that a quantity that halves at every iteration produces a logarithmic number of iterations, but in such a first assignment this has to be proven for the record.

To show that $2^n \in \Omega(2^{n-2})$ we need to find a constant c such that $2^n \geq c2^{n-2}$ for large enough n. Once more $c = 4$ works well since the relation becomes $2^n \geq 2^n$, clearly true for any $n \geq 0$.

(c) $(\log n^2) \in o(\log n)^2$

ANSWER:

We need to show that $\log n^2 \leq c(\log n)^2$ for any c and large enough n. This is equivalent to $\log n + \log n \leq c(\log n) \times (\log n)$ that is, $2 \leq c(\log n)$, that is, $\log n \geq 2/c$ or $n \geq 2^{2/c}$. This means that no matter what constant c we choose there is always a threshold N (namely, $N = 2^{2/c}$) such that the relation is true for any $n \geq N$.

(d) $2^{(n+1)} \in O(4^n)$

ANSWER:

We need to show that $2^{(n+1)} \leq c4^n$ for some constant c and any large enough n. This is equivalent to $2^{n+1} \leq c2^{2n}$. We can choose $c = 1$, case in which the relation is true for any $n \geq 1$ (since $2n \geq n + 1$ for any such an n).

(e) $2^{2n} \notin \Theta(2^n)$

ANSWER:

It must be the case that either $2^{2n} \notin O(2^n)$ or $2^{2n} \notin \Omega(2^n)$. We can try both, but we can also notice that 2^{2n} seems to grow faster than 2^n (since the exponent is twice as large) so we suspect that $2^{2n} \notin O(2^n)$. We will try to prove this by contradiction:
Assume that $2^{2n} \in O(2^n)$ and so $2^{2n} \leq c2^n$ for some constant c and large enough n. This is equivalent with $2^n \leq c$ or $n \leq \log c$. No matter what constant c we choose, there will be a threshold for n (namely, $\log c$) over which the relation becomes false. That is, the relation cannot be true for arbitrarily large n, a contradiction.

---

4. For each relation below find *all* the $\mathbb{X} \in \{O, \Omega, \Theta, o, \omega\}$ that make the relation true. Justify your answer *using limits*.

(a) $3n^3 + 2n^2 + n \in \mathbb{X}(n^3)$

ANSWER:

$\lim_{n\to\infty} \frac{n^3+2n^2+n}{n^3} = \lim_{n\to\infty} \left(\frac{n^3}{n^3} + \frac{2n^2}{n^3} + \frac{n}{n^3}\right) = \lim_{n\to\infty} \left(1 + \frac{2}{n} + \frac{1}{n^2}\right) = 1 + 0 + 0 = 1.$ Therefore $3n^3 + 2n^2 + n \in \Theta(n^3)$ and so $3n^3 + 2n^2 + n \in O(n^3)$ and also $3n^3 + 2n^2 + n \in \Omega(n^3)$.

(b) $(n \log n)^2 \in \mathbb{X}(n^2 \log n^2)$

Answer:

$\lim_{n\to\infty} \frac{(n \log n)^2}{n^2 \log n^2} = \lim_{n\to\infty} \frac{n^2 (\log n)^2}{n^2 2 \log n} = \lim_{n\to\infty} \frac{\log n}{2} = \infty$. Therefore $(n \log n)^2 \in \omega(n^2 \log n^2)$ and so it is also the case that $(n \log n)^2 \in \Omega(n^2 \log n^2)$.

(c) $n^2 + 2^n \in \mathbb{X}(n2^n)$

Answer:

$\lim_{n\to\infty} \frac{n^2+2^n}{n2^n} = \lim_{n\to\infty} \left(\frac{n^2}{n2^n} + \frac{2^n}{n2^n}\right) = \lim_{n\to\infty} \left(\frac{n}{2^n} + \frac{1}{n}\right) = \lim_{n\to\infty} \frac{n}{2^n} + 0 = \lim_{n\to\infty} \frac{n}{2^n} = \lim_{n\to\infty} \frac{n'}{(2^n)'} = \lim_{n\to\infty} \frac{1}{2^n \ln n} = 0$. It follows that $n^2 + 2^n \in o(n2^n)$ and therefore it is also the case that $n^2 + 2^n \in O(n2^n)$.

(d) $(n-1)! \in \mathbb{X}(n!)$

Answer:

$\lim_{n\to\infty} \frac{(n-1)!}{n!} = \lim_{n\to\infty} \frac{1}{n} = 0$ (I used the fact that $n! = (n-1)! \times n$, well known from the recursive implementation of the factorial function). That is, somehow counterintuitively $(n-1)! \in o(n!)$ (and so $(n-1)! \in O(n!)$). The factorial function grows so fast that it kind of grows faster than itself!

(e) $n \log n \in \mathbb{X}(\sqrt{n})$

Answer:

$\lim_{n\to\infty} \frac{n \log n}{\sqrt{n}} = \lim_{n\to\infty} \frac{\sqrt{n}^2 \log n}{\sqrt{n}} = \lim_{n\to\infty} \sqrt{n} \log n = \infty$. That is, $n \log n \in \omega(\sqrt{n})$ and so $n \log n \in \Omega(\sqrt{n})$.
