CS341 Algorithms

Assignment 1

Reuben Drummond

1. (a) [9 marks] Prove or disprove each of the following statements.

   i. if $\log(f(n)) \in \Omega(\log(g(n)))$ then $f(n) \in \Omega(g(n))$.

   Consider functions for $k \in \mathbb{N}$,

   $$f(n) = n^k$$
   $$g(n) = n^{k+1}$$

   We have that,

   $$\log f(n) = kn$$
   $$\log g(n) = (k+1)\, n$$

   It is true that $\log f(n) \in \Omega\left(g\left(n\right)\right)$ since,

   $$\exists c, n_0 > 0 : \forall n > n_0, c \log g\left(n\right) \leq \log f\left(n\right) \tag{1}$$

   And we can select a $c \forall n_0$ which satisfies the definition,

   $$\begin{aligned} c \log g\left(n\right) &\leq \log f\left(n\right) \\ ckn &\leq (k+1)\, n \\ c &\leq \frac{(k+1)}{k} \quad (\text{since } n > 0) \end{aligned}$$

   Assume that $f\left(n\right) \in \Omega g\left(n\right)$. By **??**, we can see that,

   $$\begin{aligned} cg\left(n\right) &\leq f\left(n\right) \\ cn^{k+1} &\leq n^k \\ c &\leq \frac{n^{k+1}}{n^k} = n \end{aligned}$$

   Clearly, we cannot select a finite value of c for which the condition for 1 hold, which is a contradiction to the claim that $f\left(n\right) \in \Omega g\left(n\right)$.

   Since we have found a counter example we can disprove the original statement.

   ii. If $f(n) \in O(h(n))$ and $g(n) \in O(h(n))$ then $\dfrac{f(n)}{g(n)} \in O(1)$.

   Consider the functions,

   $$f\left(n\right) = n$$
   $$g\left(n\right) = 1$$
   $$h\left(n\right) = n^2$$

Clearly, $f(n), g(n) \in h(n)$ since,

$$\exists c_1, n_0 : \forall n > n_0, f(n) \leq c_1 h(n)$$
$$\exists c_2, n_0 : \forall n > n_0, g(n) \leq c_2 h(n)$$

Now consider,

$$\frac{f(n)}{g(n)} = \frac{n}{1} = n \notin O1$$

We can therefore disprove the original statement by counter example.

iii. If $f(n) \in o(g(n))$ then $\log(f(n)) \in o(\log(g(n)))$.
Consider functions such that $f(n) \in o(g(n))$,

$$f(n) = n$$
$$g(n) = n^2$$

If $\log f(n) \in o(\log g(n))$ we must have $\forall c > 0$,

$$\begin{aligned} \log f(n) &\leq c \log g(n) \\ \log n &\leq c \log n^2 \\ \log n &\leq c2 \log n \\ 1 &\leq c2 \quad \text{since } n > 0 \\ \frac{1}{2} &\leq c \\ c &\geq \frac{1}{2} \end{aligned}$$

Since $c \geq \frac{1}{2}$, we conclude that $\log f(n) \notin o(\log g(n))$. Therefore, we have disproven the original statement by counter example.

(b) [6 marks] Analyze the following pseudocode and give a tight (Θ) bound on the running time as a function of n. You can assume that all individual instructions (including logarithm) are elementary, i.e., take constant time. Show your work.

```
l := 0;
for i = n + 1 to n^2 do
  for j = 1 to ⌈log i⌉ do
    l := l + 1
  od
od.
```

There are two loops which can be expressed as a nested sum. Note that the execution time of operations are denoted as c_1 through to c_4 in order. Expressed as a sum, we have that the running time is,

$$\begin{aligned}\text{running time} &= c_1 + \sum_{i=n+1}^{n^2} \left(c_2 + \sum_{j=0}^{\log i} (c_3 + c_4) \right) \\ &= c_1 + \sum_{i=n+1}^{n^2} (c_2 + (c_3 + c_4) \log i)\end{aligned}$$

We can say that,

$$\sum_{i=n+1}^{n^2} (c_2 + (c_3 + c_4) \log i) \leq \int_x^{x^2} (c_2 + (c_3 + c_4) \log x)\, dx$$

Since $\log x$ is monotonically increasing and the sum can be viewed as a Riemann sum of width 1.

Noting that,

$$\int \log_b x dx = \frac{1}{\ln b} (x \ln x - x) + c$$

we have,

$$\begin{aligned}\text{running time} &\leq c_1 + \int_n^{n^2} (c_2 + (c_3 + c_4) \log_b x)\, dx \\ &\leq c_1 + \left[c_2 x + (c_3 + c_4) \frac{1}{\ln b} (x \ln x - x) \right]_n^{n^2} \\ &= c_1 + c_2 (n^2 - n) + (c_3 + c_4) \frac{1}{\ln b} \left[(n^2 \ln n^2 - n^2) - (n \ln n - n) \right]\end{aligned}$$

Clearly, the highest order term is proportional to $n^2 \log n$. Therefore, we can conclude the tight bound is $\Theta(n^2 \log n)$.

2. [10 marks] Solve the following recurrence relations to obtain a closed-form big-O expression for $T(n)$. In each question, assume $T(c)$ is bounded by a constant for any small constant c.

   (a) $T(n) \leq 9\,T(\frac{n}{3}) + n^2$

   (b) $T(n) \leq 4\,T(\frac{n}{4}) + n \log n$

   (c) $T(n) \leq \sqrt{n} \cdot T(\sqrt{n}) + n$

3. [10 marks] Stavros and Jessica are discussing about runtime of an algorithm which is given by the following recurrence relation:

$$T(m) = T\left(\frac{m}{3}\right) + T\left(\frac{2m}{3}\right) + c^2 m, \quad T(1) = d.$$

Stavros: I think $T(m) \in O(m \log m)$. Let's use recursion tree method to show this. The longest path from the root to a leaf is $\log_{\frac{3}{2}} m$. after drawing the tree we can see the cost of each level (including the last one) is less than or equal to $c^2 m$. So, the total cost is

$$c^2 m \log_{\frac{3}{2}} m \in O(m \log m).$$

Jessica: I believe $T(m) \in \omega(m \log m)$. Actually based on your argument, $T(m)$ is exponential in terms of m. I agree with the upper bound on the height of the recursion tree which is $\log_{\frac{3}{2}} m$. However, using the fact that the tree may have $2^{\log_{\frac{3}{2}} m}$ leaves (based on its height) and the cost of a leaf being the constant d, we can see that the cost of leaves is exponential in m. That is why $T(m)$ is in $\omega(m \log m)$ and this implies that $T(m)$ cannot be in $O(m \log m)$.

Explain in **details**, what is correct and what is wrong in the above discussion. More precisely, provide details on which part of their argument is correct and which part is not. Moreover, provide a tight bound for $T(m)$ and justify your answer only for the upper bound.

Starvos is correct that the longest path to the root is $\log_{\frac{3}{2}} m$. He is wrong that each level is proportional to $c^2 m$. For levels less than $\log_3 m$ (the depth of the shortest branch), indeed each level has a cost of $c^2 m$ as a simple illustration shows. However, when there are levels where nodes have reached the base case, we cannot necessarily say the level has a cost $\leq c^2 m$. A simply way to see this is to consider the last level which has one node of cost d. Taking $c = 1$, we can see if $d > m$ then one level has cost $> c^2 m$ which disproves Starvos' claim. However, this is largely unimportant; what is more important to note is that each level has a cost which is proportional to m, which I believe is what Stevos' was trying to get at.

One of Starvos' key mistakes is his failure to consider the leaves, which Jesica addresses. She claims that based on the height of the tree, there are $2^{\log_{\frac{3}{2}} m}$ leaves. This is not quite correct as $2^{\log_{\frac{3}{2}} m}$ is an upper bound for the number of leaves ,since branches terminate (reach the base case) at different points. In saying this, we can use this as an upper bound for our analysis to find the big-O runtime so we can proceed with Jesica's reasoning.

Jessica continues by suggesting that the tree is leave-heavy; that is, the cost of the base cases are proportional to the number of leaves $2^{\log_{\frac{3}{2}} m}$ which she says is exponential. However, $2^{\log_{\frac{3}{2}} m}$ is not exponential:

$$2^{\log_{\frac{3}{2}} m} = m^{\log_{\frac{3}{2}} 2} \approx m^{1.71}$$

If it were exponential, Jessica would be correct in reaching the conclusion that $m \log m$ is a strict lower bound; that is, $T(m) \in \omega(m \log m)$. If this were true, it would also be correct that $T(m) \notin O(m \log m)$, since ω and O are mutually exclusive.

Appendices