CSE 101 Homework 0

Fall 2018

This homework is due on gradescope Friday October 5th at 11:59pm. Remember to justify your work even if the problem does not explicitly say so. Writing your solutions in \LaTeX is recommend though not required.

Question 1 (Program Runtimes, 20 points). Consider the following two programs:

Alg1(n)

\begin{verbatim}
m = 1
for i = 1 to n
    m = m + 2
    for j = 1 to m
        Print(j)
\end{verbatim}

and

Alg2(n)

\begin{verbatim}
m = 1
for i = 1 to n
    m = m * 2
    for j = 1 to m
        Print(j)
\end{verbatim}

For each of these programs give the asymptotic runtime as $\Theta(f(n))$ for some function $f$ and justify your work.

Solution 1.

(a) Alg1(n): The assignment $m=1$ takes constant time. The line $m = m + 2$ is run $n$ many times, which takes $\Theta(n)$ time. The inner print statement runs $m$ many times, and $m = 2n + 1$ for each $n$. The total number of times the print statement runs is

$$\sum_{i=1}^{n} 2i + 1 = n(n + 1) + n = n^2 + 2n$$

This runtime is dominated by the higher exponent of the print statement so the overall runtime is $\Theta(n^2)$

(b) Alg2(n): The assignment $m=1$ takes constant time. The line $m = m * 2$ is run $n$ many times, which takes $\Theta(n)$ time. The inner print statement runs $m$ many times, and $m = 2^n$ for each $n$. The total number of times the print statement runs is

$$\sum_{i=1}^{n} 2^i = 2^{n+1} - 2$$

The runtime is dominated by the print statement and the overall runtime is $\Theta(2^n)$
**Question 2** (Asymptotic Comparisons, 20 points). Sort the following functions of \( n \) in terms of their asymptotic growth rates. In particular, ones should go later in the list if they are larger when sufficiently large values of \( n \) are used as inputs. Which of these functions have polynomial growth rates? Remember to justify your answers.

- \( a(n) = 2^{n^{0.01}} \)
- \( b(n) = 100000n \)
- \( c(n) = n \log(n) \)
- \( d(n) = \frac{n^2}{\log^2(n)} - 20n + \sin(n) \)
- \( e(n) = n! \)

**Solution 2.** The order should be:

\[
 b(n) << c(n) << d(n) << a(n) << e(n)
\]

and the functions with polynomial growth are:

\[
 b(n), c(n), \& d(n)
\]

We should first start by simplifying the growth rate of each function.

- \( b(n) = 100000n = O(n) \)
- \( c(n) = n \log(n) = O(n \log(n)) \)
- \( e(n) = n! = O(n!) \)

and we know that \( n << n \log(n) << n! \), therefore:

\[
 b(n) << c(n) << e(n)
\]

Which leaves only \( a(n) \) & \( d(n) \) to consider:

1. For \( a(n) \), we can solve by setting \( n' = n^{100} \), then:

\[
 a(n') = 2^{(n')^{0.01}} = 2^{(n^{100})^{0.01}} = 2^n
\]

Since we know that \( n^k << 2^n << n! \) \[\forall \text{ variables } n \text{ and constants } k: n > 0, c \geq 0\]

\[
 n^k << a(n) << n!
\]

Which in turn implies,

\[
 b(n) << c(n) << a(n) << e(n)
\]

2. For \( d(n) \), we can solve by first removing the lower order terms in the equation:

\[
 d(n) = \frac{n^2}{\log^2(n)} - 20n + \sin(n) = O(n^2/\log^2(n)) - O(n) + O(1) = O(n^2/\log^2(n)) - O(n) \]

\[\text{[since } \sin(n) \leq 1 \forall n \in \mathbb{R}^+\]

Now, to determine how \( d(n) \)'s growth rate compares to the other functions, we can decompose the first term in such a way that it resembles functions we think are nearby, namely:

\[
 (n^2/\log^2(n)) = n(n/\log^2(n))
\]

Which clearly grows faster than \( O(n) \), so:

\[
 d(n) = O(n(n/\log^2(n)))
\]

But does it grow faster than \( O(n \log(n)) \)?

Since we only need to compare \( \log(n) \) & \( n/\log^2(n) \), setting \( n' = 2^n \) gives us:
(a) \( \log(n') = \log(2^n) = n \)
(b) \( \frac{n'}{\log^2(n')} = \frac{2^n}{\log(2^n) \cdot \log(2^n)} = \frac{2^n}{n \cdot \log(2^n)} = \frac{2^n}{n^2} \)

If we then multiply both equations by \( n^2 \) (since \( n \in \mathbb{R}^+ \), this will preserve the direction of inequalities), we get:

(a) \( n^2(n) = n^3 \)
(b) \( n^2(2^n/n^2) = 2^n \)

Since, \( n^k << 2^n \):

\[ n \log(n) = c(n) << d(n) \]

Altogether, this gives us:

\[ b(n) << c(n) << d(n) << a(n) << e(n) \]

As polynomial growth is defined as \( f(n) = O(n^k) \) for some constant \( k \geq 0 \), the only functions that satisfy this requirement are:

\( b(n), c(n), d(n) \)

Question 3 (Degrees and Cycles, 30 points). Recall that for a graph \( G \) that the degree of a vertex \( v \) is the number of edges with \( v \) as an endpoint. Recall that a cycle is a sequence of vertices \( v_1, v_2, \ldots, v_n \) so that there are edges \( v_1 \) to \( v_2 \), \( v_2 \) to \( v_3 \), \ldots, \( v_n-1 \) to \( v_n \) and from \( v_n \) to \( v_1 \), where additionally no edge is used more than once in this sequence.

(a) Show that for any finite graph \( G \) where all vertices of \( G \) have degree at least 2 that \( G \) contains a cycle. (Hint: Start at some vertex \( v \) of \( G \) and follow edges until a vertex is reached for the second time.) [10 points]

(b) Is it the case that every vertex of \( v \) is necessarily contained in a cycle? Prove that this is the case or provide a counter-example. [10 points]

(c) Suppose instead that every vertex of \( G \) has degree exactly equal to 2. Is it the case that every vertex of \( G \) is contained in a cycle? [10 points]

Solution 3.

(a) Let’s assume the contrary: let there be a graph \( G \) where all vertices of \( G \) have degree at least 2 and the graph does not contain a cycle. Let’s take some vertex \( v_1 \) in this graph. As its degree is at least two, there must be some other vertex, let’s say \( v_2 \), such that there is an edge \( (v_1, v_2) \). Now, as the degree of vertex \( v_2 \) is at least 2 as well, there must another vertex, let’s say \( v_3 \), such that there is an edge \( (v_2, v_3) \). Similarly, we might get that there must be edges \( (v_3, v_4), (v_4, v_5), \ldots, (v_n, v_{n+1}) \), where \( n \) is the number of vertices in graph \( G \) and is no less than 3. As we can see, there are \( n+1 \) vertices in this path. By pigeonhole principle, there must be at least one vertex that occurs in this path twice, which should be impossible, as we assumed that there are no cycle. Thus, we have a contradiction.

(b) No. Let’s take at least two separate graphs \( (G_1, G_2, \ldots, G_m), m \geq 2 \), such that any vertex in each graph has a degree at least 2. Add a vertex \( v \) and edges \( (u, u_i) \), where \( u_i \) is a vertex in \( G_i \) for all \( 1 \leq i \leq m \). Any vertex in the new graph has a degree at least 2 and there are no edges between these graphs. Thus, vertex \( v \) doesn’t belong to any cycle.
(c) Yes. Let’s assume the contrary: let there be a graph $G$, such that each vertex in it has a degree of 2 and this graph has a vertex $v$, which does not belong to any cycle. Vertex $v$ has two incident edges $(v, u)$ and $(v, w)$. Let’s denote all vertices and their incident edges, except for $v$, $(v, u)$ and $(v, w)$, reachable from $v$ through $(v, u)$ and $(v, w)$ as $G_u$ and $G_w$ respectively. There doesn’t exist an edge between these two graphs as there can’t be a cycle, therefore, these graphs are distinct (they do not have any common vertices). The sum of all degrees in each of these graphs is an odd number ($u$ and $w$ have degree 1, while all other vertices have degree 2). This, however, is impossible, as the sum of all degrees in any graph is an even number. Thus, such graphs can’t exist and we have a contradiction.

An alternative solution: let’s take any vertex $v_1$ and build a sequence of vertices $v_1, v_2, ..., v_k$ in the same way as we did in part (a). At some point, some vertex will appear twice in this sequence (otherwise, the degrees of the first and the last vertices would be 1). Let this vertex be $v_k$. But the degree of any vertex, except for $v_1$, along this path already equals 2. Thus, $v_k$ can only be equal $v_1$, which implies that $v_1$ is part of a cycle. As $v_1$ is any vertex in the graph, we conclude that each vertex in the graph must belong to some cycle.

**Question 4 (Recurrence Relation, 30 points).** Suppose that you have a function $T(n)$ defined by $T(1) = 1$ and

$$T(n) = 3T(\lfloor n/2 \rfloor) + n$$

for $n > 1$.

(a) Show that $T(2^n) = \Theta(3^n)$. (Hint: Show that $T(2^n)$ is between $3^n$ and $3^{n+1} - 2^{n+1}$.) [15 points]

(b) Consider the following “proof” that $T(n) = O(n)$ (note that this contradicts part (a)):

We proceed by strong induction on $n$. Clearly $T(1) = O(1)$, which gives us our base case. If we assume that $T(m) = O(n)$ for all $m < n$, then $T(n) = 3T(\lfloor n/2 \rfloor) + n = 3O(n) + n = O(n)$. This completes our inductive step and proves that $T(n) = O(n)$ for all $n$.

What is wrong with the above proof? (Hint: Consider what the implied constant in the $O$ term would be.) [15 points]

**Solution 4.**

(a) $T(2^n)$

$$= 3T(2^{n-1}) + 2^n$$

$$= 3^k(T(2^{n-k})) + 3^{k-1}2^{n-k} + ... + 2^n$$

$$= 3^n + 2(3^{n-1}) + ... + 2^n$$

This is a geometric series with each element with $2/3$ the value of the previous element, ending at $2^n$. Using the geometric series formula, we get

$$T(2^n) = (3^{n+1} - 2^{n+1})/(3 - 2) = 3^{n+1} - 2^{n+1}.$$  

This is dominated by the $3^{n+1}$ term so

$$T(2^n) = \Theta(3^n)$$
(b) To show that something is \( O(n) \), we need to find some constant \( C \) such that \( C \cdot n > T(n) \) for all \( n > k \) for some constant \( k \).

Instead, what this proof is doing is showing that for each value of \( n \), there is some constant \( C \) such that \( T(n) < Cn \). The values of \( C \) that this proof finds are not all the same, which is an issue. This proof shows that up to a certain point, \( T(n) \) is bounded by \( O(n) \), with no guarantee for arbitrarily large values of \( n \). Note one could use a similar argument to argue any function is \( \Theta(1) \), which is clearly not the case.

The correct approach for this problem would be to work out what value the recursion sums to and then find some \( C \) such that \( T(n) < C \cdot n \) for all \( n > k \) where \( k \) is some constant. However, this is impossible.

The actual runtime for this is \( \Theta(n^{\log_2(3)}) \).

\[
T(n) = 3T(n/2) + n \\
= 3kT(n/2^k) + 3^{k-1}n + \ldots + n \\
= 3^{\log_2(n)} = 3^{\log(n)/\log(2)} \\
= 3^{\log_3(n)\log(2)/\log(3)\log(2)} \\
= n^{\log_2(3)}
\]

Question 5 (Extra credit, 1 point). Approximately how much time did you spend working on this homework?

Solution 5.