Problem Set4

Kaiqu Liang; Zirun Hong; Jitian Zheng

April 4, 2018

Question1

(a)
Let $n \in \mathbf{Z}^+$
We know that $p = \lfloor logn \rfloor + 1$
bits is an array of length p with all 0's
Let m_k be the total steps at the k-th time executing the inner loop.
We can find that $m_2 = 1, m_4 = 2, m_8 = 3, m_{16} = 4$
$m_{2^{\lceil logn \rceil}} = \lceil logn \rceil$
Moreover,
$m_2 \geq max\{m_1, m_2\}$
$m_4 \geq max\{m_1, m_2, m_3, m_4\}$
$m_8 \geq max\{m_1, m_2, \ldots, m_8\}$
$m_{2^{\lceil logn \rceil}} \geq max\{m_1, m_2, \ldots, m_{2^{\lceil logn \rceil}}\}$
Since $\lceil logn \rceil \geq logn$, we have $2^{\lceil logn \rceil} \geq n$
Therefore, we have that at the k-th time executing the inner loop(when k = $2^{\lceil logn \rceil}$), $m_{2^{\lceil logn \rceil}} \geq max\{m_1, m_2, \ldots, m_n\}$
Since $m_{2^{\lceil logn \rceil}} = \lceil logn \rceil$, we have $\lceil logn \rceil \geq max\{m_1, m_2, \ldots, m_n\}$
There are n iterations of outer loop.
Each iteration takes at most $\lceil logn \rceil$ steps.
Therefore there are at most $n\lceil logn \rceil$ steps.
The running time of this algorithm is $n\lceil logn \rceil \in \mathcal{O}(nlogn)$

(b)
Since there are n iteration, each iteration takes at least one single step.
There are at least n steps in total.
The running time of this algorithm is $n \in \Omega(n)$

Question2

(a)
WTS: $WC(n) \in \mathcal{O}(n)$
Let n be the length of L.

For every iteration, the minimum change of i is that i decreases by 1.
Therefore, there are at most $n-1$ iterations.
Since each iteration takes a single step.
The total cost is at most $n-1$ steps.
The running time of myprogram is $n-1 \in \mathcal{O}(n)$.
Hence, $WC(n) \in \mathcal{O}(n)$

(b)
For each $n \in \mathbb{N}$, consider the input family:
$L[i] = 1$, for $i = 0, 1, 2, ...n-1$ which means that all of the elements in L is 1.
The loop will always run the else branch since $L[i]\%2 \; != 0$ for $i = 0, 1, 2, 3, ...n-1$
Then i will decrease by 1 after each iteration.
In this case, there are $n-1$ iterations and each iteration takes a single step.
The total cost is $n-1$ steps.
The running time of myprogram is $n-1 \in \Omega(n)$
Hence, $WC(n) \in \Omega(n)$

(c)
For each $n \in \mathbb{N}$,consider the input family:
$L[i] = 2$ for $i = 0, 1, 2, 3...n-1$ which means that all the elements in L is 2.
Since $L[i]\%2 == 0$, for $i = 0, 1, 2, 3...n-1$,
it will always run the if branch.
Let i_k be the value of i at the k-th iteration.
For each iteration, i will decrease by at least a half.
Hence $i_{k+1} \leq \frac{i_k}{2}$
The loop terminates when $i \leq 0$
Therefore there are $\lceil logn \rceil$ iterations.
Each iteration takes a single step.
We have the total cost is $\lceil logn \rceil$ steps.
The running time of myprogram is $\lceil logn \rceil \in \mathcal{O}(logn)$.
Hence, BC(n)$\in \mathcal{O}(logn)$

(d)
WTS: $BC(n) \in \Omega(logn)$
At the beginning, i decreases faster if we run the if branch.
But as x becomes large, there may be a situation that i decreases more in else branch than if branch.
Therefore, if we run if branch before this point and then run else branch after this point, we can get the fastest running time.
Let k be the maximum number of iterations which runs the if branch.
Let l be the remaining iterations which run the else branch.
Since if we always run the if branch, there are at most $\lceil logn \rceil$ iterations,
we have $k \leq \lceil logn \rceil$.
Let $k = m\lceil logn \rceil, 0 < m \leq 1$
After k iterations, the value of i is $\lfloor \frac{n-1}{2^k} \rfloor$, the value of x is $k+1$

Therefore $l = \lceil \frac{\lfloor \frac{n-1}{2^k} \rfloor}{k+1} \rceil$
Since $k \geq 0$, the total iteration is $k + l$

$$\begin{aligned}
k + l &\geq \lceil \frac{\lfloor \frac{n-1}{2^k} \rfloor}{k+1} \rceil \\
&\geq \frac{\frac{n-1}{2^k} - 1}{k+1} \\
&= \frac{\frac{n-1}{2^{m\lceil logn \rceil}} - 1}{m\lceil logn \rceil + 1} && \text{(by } k = m\lceil logn \rceil\text{)} \\
&\geq \frac{\frac{n-1}{2^{m(logn+1)}} - 1}{m(logn+1)+1} && \text{(by } \lceil logn \rceil \leq logn + 1\text{)} \\
&= \frac{\frac{n-1}{n^m \cdot 2^m} - 1}{m(logn+1)+1} \\
&\geq \frac{\frac{1}{2} \cdot \frac{n-1}{n^m} - 1}{logn + 2} && \text{(by } \frac{1}{2} \leq \frac{1}{2^m} < 1, 0 < m < 1\text{)} \\
&= \frac{\frac{1}{2}(n^{1-m} - n^{-m}) - 1}{logn + 2} \\
&> \frac{\frac{1}{2}(n^{1-m} - 1) - 1}{logn + 2} && \text{(by } -1 < -\frac{1}{n^m} < 0\text{)} \\
&= \frac{n^{1-m} - 3}{2(logn + 2)}
\end{aligned}$$

Therefore we have $k + l \in \Omega(\frac{n^{1-m}-3}{2(logn+2)})$
WTS: $\frac{n^{1-m}-3}{2(logn+2)} \in \Omega(logn)$
By $n^c >> (logn)^2 >> logn$, for all $c > 0$, we can directly get the limit $\lim_{n\to\infty} \frac{2(logn)^2+4logn}{n^{1-m}-3} = 0$.
But to be more precise, I will use L'Hôpital's rule to show it.

$$\begin{aligned}
\lim_{n\to\infty} \frac{logn}{\frac{n^{1-m}-3}{2(logn+2)}} &= \lim_{n\to\infty} \frac{2(logn)^2 + 4logn}{n^{1-m} - 3} \\
&\overset{L'H}{=} \lim_{n\to\infty} \frac{4logn \cdot \frac{1}{n \ln 2} + 4 \times \frac{1}{n \ln 2}}{(1-m)n^{-m}} \\
&= \lim_{n\to\infty} \frac{4logn + 4}{(1-m) \cdot \ln 2 \cdot n^{-m+1}} \\
&= \frac{4}{\ln 2(1-m)} \cdot \lim_{n\to\infty} \frac{logn + 1}{n^{-m+1}} \\
&\overset{L'H}{=} \frac{4}{\ln 2(1-m)} \cdot \lim_{n\to\infty} \frac{\frac{1}{n \ln 2}}{(-m+1)n^{-m}} \\
&= \frac{4}{(\ln 2)^2(1-m)} \cdot \lim_{n\to\infty} \frac{1}{n^{-m+1} \cdot (-m+1)} \\
&= 0
\end{aligned}$$

Therefore by theorem 5.9(ii) (if $\lim_{n\to\infty} \frac{f(n)}{g(n)} = 0$, then $f \in \mathcal{O}(g)$ and $g \notin \mathcal{O}(f)$),
We have $logn \in \mathcal{O}(\frac{n^{1-m}-3}{2(logn+2)})$
By theorem 5.3 (for all $f, g : \mathbb{N} \to \mathbb{R}^{\geq 0}, g \in \mathcal{O}(f)$, if and only if $f \in \Omega(g)$)
We have $\frac{n^{1-m}-3}{2(logn+2)} \in \Omega(logn)$
Since we know $k + l \in \Omega(\frac{n^{1-m}-3}{2(logn+2)})$
By theorem 5.4 (for all $f, g, h : \mathbb{N} \to \mathbb{R}^{\geq 0}$, if $f \in \Omega(g)$ and $g \in \Omega(h)$, then $f \in \Omega(h)$),
We have the running time is $k + l \in \Omega(logn)$
Therefore I can conclude that $BC(n) \in \Omega(logn)$

Question3

(a)
WTS : $WC_{has\_isolated}(n) \in \Theta(2^n)$
First I want to show $WC_{has\_isolated}(n) \in \mathcal{O}(2^n)$
Let the first outer loop be loop 1(line 5). Let the inner loop be loop 2(line 7). Let the last loop be loop 3(line 14).
For a fixed iteration of loop 1, there are at most n iterations in loop2 and each iteration takes a single steps. The cost of loop2(inner loop) is at most n steps.
Then for loop1(outer loop), there are at most n iterations and each iteration takes at most n steps. The total cost of loop 1 is at most n^2 steps.
Then for loop 3(last loop), there are at most 2^n iterations and each iteration takes a single steps. The cost for loop 3 is at most 2^n steps.
Therefore total cost is at most $n^2 + 2^n$ steps. The running time of $has\_isolated$ is at most $n^2 + 2^n$.
I have shown that $WC_{has\_isolated}(n) \in \mathcal{O}(2^n)$.

Second I want to show $WC(n) \in \Omega(2^n)$
For each $n \in \mathbb{N}$, consider the input family : M[0][j] = 0, for i = 0, 1, 2 . . . n - 1 which means that the adjacency matrix M satisfies that all the elements in first row are 0.
Then after executing the inner loop first time, count = 0, found_isolated is True. The loop will break. Therefore we have there are n iterations in the loop2(inner loop) and each iteration takes a single step. There are n steps in the inner loop. Since there is only one iteration of outer loop, the cost of loop 1 is n steps. Since there are 2^n iterations in loop 3(last loop) and each iteration takes a single step. The cost of loop 3 is 2^n steps. Therefore the total cost is $2^n + n$ steps.
The running time is $2^n + n \in \Omega(2^n)$, Therefore we have $WC(n) \in \Omega(2^n)$.

(b)
WTS: $BC(n) \in \Theta(n^2)$
First I want to show $BC(n) \in \Omega(n^2)$ (*)
Case 1: found_isolated is eventually True.
Then let's consider the loop 3(last loop), there are 2^n iterations and each iteration takes a single step. The cost of loop 3 is 2^n steps. The total cost is at least 2^n steps. The running

time is at least $2^n \in \Omega(n^2)$
Case 2: found_isolated is eventually False
Then the loop 2 will never break.
For a fixed iteration of loop 1(outer loop), there are n iterations in loop 2 and each iteration takes a single step. The cost of loop 2 is n steps. For loop 1, there are n iterations and each iteration takes n step. The cost of loop 1(outer loop) is n^2 steps. Since there is no iteration in loop 3. The total cost is at least n^2 steps. The running time is at least $n^2 \in \Omega(n^2)$
Therefore I 've shown that $BC(n) \in \Omega(n^2)$

Secondly, I want to show $BC(n) \in \mathcal{O}(n^2)$ $(\star)$
For each $n \in \mathbb{N}$, consider the input family:
M[i][j] = 1, for all i = 0, 1, 2 ... n - 1, j = 0, 1, 2 ... n - 1
Therefore after each iteration of loop 2, count != 0. The loop will never terminate early.
For loop 1, there are n iterations and each iteration takes n steps. The cost of loop 1 is n^2.
Since there is no iteration in loop 3, the total cost is $n^2 \in \mathcal{O}(n^2)$ steps.
I have shown that $BC(n) \in \mathcal{O}(n^2)$
By (*) and $(\star)$, I can conclude that $BC(n) \in \Theta(n^2)$

(c)
Formula: $2^{\frac{n(n-1)}{2}}$

(d)
Since M[i][j] == M[j][i] and M[i][i] = 0,
when i = 0, M[0][1], M[0][2] ... M[0][n - 1] all have two possibilities(0 or 1)
Therefore the number of the possible adjacency matrices is 2^{n-1}
When i = 1, M[1][2], M[1][3] ... M[1][n - 1] all have two possibilities(0 or 1)
the number of the possible adjacency matrices is 2^{n-2}
Similarly, when i = 2, M[2][3], M[2][4] ... M[2][n - 1] all have two possibilities(0 or 1)
the number of the possible adjacency matrices is 2^{n-3}
...
when i = n - 1, the number of the possible adjacency matrices is 1.
The total number of the possible adjacency matrices is $2^{n-1} \times 2^{n-2} \times \ldots \times 1 = 2^{\frac{n(n-1)}{2}}$

(e)
1) First of all, if we choose one of the vertex to be isolated, there will be n possibilities.
2) Secondly, after choosing a isolated vertex, we need to consider how many different graphs with $n-1$ vertices.
Since graph with n-1 vertices is like adjacency matrices of size $(n-1)-by-(n-1)$.
From Q3(c), we know the number of adjacency matrices of size $(n-1)-by-(n-1)$ is $2^{\frac{(n-2)\times(n-1)}{2}}$.
Therefore, there are $2^{\frac{(n-2)\times(n-1)}{2}}$ different graphs with $n-1$ vertex.
From 1) and 2), the number of $n-by-n$ adjacency matrices that represent a graph with at least one isolated vertex is at most $n \cdot 2^{\frac{(n-1)(n-2)}{2}}$.

(f)
Let Tn be set of all inputs to algorithm has_isolated of length n.
Since the number of adjacency matrices of size $n-by-n$ that represent valid graphs is $2^{\frac{n(n-1)}{2}}$
$|Tn| = 2^{\frac{n(n-1)}{2}}$
Therefore $AC(n) = \frac{\sum\limits_{x \in Tn} Runtime_{has\_isolated}(x)}{|Tn|} = \frac{\sum\limits_{x \in Tn} Runtime_{has\_isolated}(x)}{2^{\frac{n(n-1)}{2}}}$
First I want to show $AC(n) \in \mathcal{O}(n^2)$
Part1: the sum of runtime of has_isolated whose input is n-by-n adjacency matrices with no isolated vertex.
By Q3(c) and Q3(e), we have the number of n-by-n adjacency matrices that represent a graph with no isolated vertex is $2^{\frac{n(n-1)}{2}} - n \cdot 2^{\frac{(n-1)(n-2)}{2}}$
For these input, there is no break in loop 1 and loop 2 and there is no iteration in loop 3.
Since for loop 2, there are n iterations, each iteration takes a single step.
The cost is n steps for a fixed iteration of loop 1.
For loop 1, there are n iterations, and each iteration takes n steps.
The total cost is n^2 steps. The sum of runtime is $n^2[2^{\frac{n(n-1)}{2}} - n \cdot 2^{\frac{(n-1)(n-2)}{2}}]$

Part2: the sum of runtime of has_isolated whose input is $n - by - n$ adjacency matrices with at least one isolated vertex.
By Q3(e), we have the number of $n - by - n$ adjacency matrices that represent a graph with at least one isolated vertex is at most $n \cdot 2^{\frac{(n-1)(n-2)}{2}}$
For a fixed iteration of loop 1, there are n iterations in loop 2 and each iteration takes a single step. Therefore the cost of loop 2 is n steps. Then for loop 1, there are at most n iterations and each iteration takes n steps. Therefore there are at most n^2 steps in loop 1. Since there are 2^n iterations in loop 3 and each iteration takes a single steps, there are n^2 steps in loop 3.
Hence, we have the total cost is at most $2n^2$ steps.
Therefore the sum of runtime is at most $2n^3 \cdot 2^{\frac{(n-1)(n-2)}{2}}$
By part1 and part2:

$$\begin{aligned}
AC(n) &= \frac{\sum\limits_{x \in Tn} Runtime_{has_isolated}(x)}{2^{\frac{n(n-1)}{2}}} \\
&\leq \frac{n^2[2^{\frac{n(n-1)}{2}} - n \cdot 2^{\frac{(n-1)(n-2)}{2}}] + 2n^3 \cdot 2^{\frac{(n-1)(n-2)}{2}}}{2^{\frac{n(n-1)}{2}}} \\
&= \frac{n^2 \times 2^{\frac{n(n-1)}{2}} + n^3 \times 2^{\frac{(n-1)(n-2)}{2}}}{2^{\frac{n(n-1)}{2}}} \\
&= n^2 + \frac{n^3}{2^{n-1}}
\end{aligned}$$

Therefore $AC(n) \in \mathcal{O}(n^2 + \frac{n^3}{2^{n-1}})$
Now I want to show $n^2 + \frac{n^3}{2^{n-1}} \in \mathcal{O}(n^2)$
It's obvious that $n^2 \in \mathcal{O}(n^2)$
Let $f(n) = \frac{n^3}{2^{n-1}}, g(n) = n^2$

Since $\lim_{n\to\infty} \frac{f(n)}{g(n)} = \lim_{n\to\infty} \frac{\frac{n^3}{2^{n-1}}}{n^2} = \lim_{n\to\infty} \frac{n}{2^{n-1}} = 0$
By theorem 5.9(ii)(if $\lim_{n\to\infty} \frac{f(n)}{g(n)} = 0$, then $f \in \mathcal{O}(g)$ and $g \notin \mathcal{O}(f)$),
We have $\frac{n^3}{2^{n-1}} \in \mathcal{O}(n^2)$
By theorem 5.5(if $f \in \mathcal{O}(n)$ and $g \in \mathcal{O}(n)$, then$f + g \in \mathcal{O}(n)$),
We have $n^2 + \frac{n^3}{2^{n-1}} \in \mathcal{O}(n^2)$
I have shown that $n^2 + \frac{n^3}{2^{n-1}} \in \mathcal{O}(n^2)$
Since we also have $AC(n) \in \mathcal{O}(n^2 + \frac{n^3}{2^{n-1}})$
By theorem 5.4(for all $f, g, h : \mathbb{N} \to \mathbb{R}^{\geq 0}$, if $f \in \mathcal{O}(g)$ and $g \in \mathcal{O}(h)$, then $f \in \mathcal{O}(h)$),
We have $AC(n) \in \mathcal{O}(n^2)$

Secondly, I want to show $AC(n) \in \Omega(n^2)$
Part1: the sum of runtime of has_isolated whose input is n-by-n adjacency matrices with no isolated vertex.
The process is the same as the part1 above. The sum of runtime is $n^2[2^{\frac{n(n-1)}{2}} - n \cdot 2^{\frac{(n-1)(n-2)}{2}}]$

Part2: the sum of runtime of has_isolated whose input is $n - by - n$ adjacency matrices with at least one isolated vertex.
By Q3(e), we have the number of $n - by - n$ adjacency matrices that represent a graph with at least one isolated vertex is at most $n \cdot 2^{\frac{(n-1)(n-2)}{2}}$
For loop 1, there is at least 1 iteration and each iteration takes n steps. Hence, the cost for loop 1 is at least n steps.
Moreover, there are 2^n iterations in loop 3 and each iteration takse a single step. Therefore the cost for loop 3 is n^2 steps.
The total cost is at least $n^2 + n$ steps.
Therefore the sum of runtime is at least $(n^2 + n) \cdot 2^{\frac{(n-1)(n-2)}{2}} \cdot n$
By Part1 and Part2:

$$
\begin{aligned}
AC(n) &= \frac{\sum\limits_{x \in Tn} Runtime_{has\_isolated}(x)}{2^{\frac{n(n-1)}{2}}} \\
&\geq \frac{n^2[2^{\frac{n(n-1)}{2}} - n \cdot 2^{\frac{(n-1)(n-2)}{2}}] + (n^2+n) \cdot 2^{\frac{(n-1)(n-2)}{2}} \cdot n}{2^{\frac{n(n-1)}{2}}} \\
&= n^2 + \frac{n^2}{2^{n-1}}
\end{aligned}
$$

Therefore $AC(n) \in \Omega(n^2 + \frac{n^2}{2^{n-1}})$
Now I want to show $n^2 + \frac{n^2}{2^{n-1}} \in \Omega(n^2)$ which is equivalent to show $\exists c_0, n_0 \in \mathbb{R}^+, \forall n \in \mathbb{N}, n \geq n_0 \Rightarrow n^2 + \frac{n^2}{2^{n-1}} \geq c_0 n^2$
Let $c_0 = 1, n_0 = 1$
Let $n \in \mathbb{N}$
Assume $n \geq n_0$. Hence $n \geq 1$
WTS: $n^2 + \frac{n^2}{2^{n-1}} \geq n^2$
Since $\frac{n^2}{2^{n-1}} \geq 0 (n \geq 1)$

we have $n^2 + \frac{n^2}{2^{n-1}} \geq n^2$
I have shown that $n^2 + \frac{n^2}{2^{n-1}} \geq n^2$
Therefore $n^2 + \frac{n^2}{2^{n-1}} \in \Omega(n^2)$
Since we also have $AC(n) \in \Omega(n^2 + \frac{n^2}{2^{n-1}})$
By theorem 5.4(for all $f, g, h : \mathbb{N} \to \mathbb{R}^{\geq 0}$, if $f \in \Omega(g)$ and $g \in \Omega(h)$, then $f \in \Omega(h)$),
We have $AC(n) \in \Omega(n^2)$
Since I have shown $AC(n) \in \mathcal{O}(n^2)$ and $AC(n) \in \Omega(n^2)$.
By definition of Big-Theta, $AC(n) \in \Theta(n^2)$