Name: Jayson Cheng
NUID: 28638995

CSCE 310 Assignment 1

Question 1

a) $T(n) \approx C_{op} C(n) \approx C_{op} \frac{1}{3} n^3$

$T(3n) \approx C_{op} C(3n) \approx C_{op} \frac{1}{3}(3n)^3$

when $n = 10$,

$T(10) \approx C_{op} C(10) \approx C_{op} \frac{1}{3}(10)^3$

when $n = 30$,

$T(30) \approx C_{op} C(30) \approx C_{op} \frac{1}{3}(30)^3$

$$\frac{T(30)}{T(10)} = \frac{T(3n)}{T(n)} = \frac{\cancel{C_{op} \frac{1}{3}} (3\cancel{n})^3}{\cancel{C_{op} \frac{1}{3} (n)^3}} = 27$$

$\therefore$ The algorithm is expected to run 27 times slower when working on a system of 30 equations versus a system of 10 equations.

b) $T(n) = C_{op} \frac{1}{3} n^3$

$T(N) = \frac{1}{1000} C_{op} \frac{1}{3} N^3$

$T(n) = T(N)$

$\frac{1}{3} n^3 = \frac{1}{1000} (\frac{1}{3} N^3)$

$\frac{N^3}{n^3} = \frac{1}{1000}$

$\frac{N}{n} = \sqrt[3]{1000}$

$\frac{N}{n} = 10$

$\therefore$ factor $= 10$

Question 2

a) $f_1(n) = n(n+1) = n^2 + n \approx n^2$

$f_2(n) = 2000\, n^2 \approx n^2$

$\therefore$ same order of growth

b) $f_1(n) = 100 n^2 \approx n^2$

$f_2(n) = 0.01 n^3 \approx n^3$

$\therefore$ $f_1(n)$ has lower order of growth than $f_2(n)$

c) $f_1(n) = \log_2 n$

$f_2(n) = \ln n \approx \log_e n \approx \dfrac{\log_2 n}{\log_2 e} \approx \log_2 n$

$\therefore$ same order of growth

d) $f_1(n) = \log_2^2 n = \log_2 n \cdot \log_2 n$

$f_2(n) = \log_2 n^2 = 2 \log_2 n$

$\therefore$ $f_1(n)$ has higher order of growth than $f_2(n)$

e) $f_1(n) = 2^{n-1} \approx 2^n\left(\frac{1}{2}\right) \approx 2^n$

$f_2(n) = 2^n$

$\therefore$ same order of growth

f) $f_1(n) = (n-1)!$

$f_2(n) = n! \approx n(n-1)!$

$\therefore$ $f_1(n)$ has lower order of growth than $f_2(n)$

Question 3

a) $x(n) = x(n-1) + 1$ for $n > 1$ and $x(1) = 9$

$x(n) = x(n-1) + 1$

$= x(n-2) + 1 + 1$

$= x(n-3) + 1 + 1 + 1$

$= x(n-a) + a$

$= \overset{9}{\cancel{x(1)}} + (n-1)$

$= 9 + (n-1)$

$= n + 8$

b) $x(n) = 6x(n-1) + 2$ for $n > 0$ and $x(0) = 6$

$$x(n) = 6x(n-1) + 2$$
$$= 6(6x(n-2)+2) + 2$$
$$= 6(6^2 x(n-3) + 6(2) + 2) + 2$$
$$= 6^3 x(n-3) + 6^2(2) + 6(2) + 2$$
$$= 6^n x(0) + 6^{n-1}(2) + 6^{n-2}(2) + \dots 2$$
$$= 6^n(6) + 2\left(\frac{6^n - 1}{6-1}\right)$$
$$= 6^{n+1} + 2\left(\frac{6^n-1}{5}\right)$$

c) $x(n) = x(n-1) + n^3$ for $n > 0$ and $x(0) = 6$

$$x(1) = x(0) + 1$$
$$x(2) = x(1) + 2^3$$
$$x(3) = x(2) + 3^3$$
$$x(n) = x(0) + 1 + 2^3 + 3^3 + \dots n^3$$
$$= 6 + \frac{n^2(n+1)^2}{4}$$

d) $x(n) = x\left(\frac{n}{7}\right) + n$ for $n > 1$ and $x(1) = 7$ $(n = 7^k)$

$$x(n) = x(7^{k-1}) + 7^k$$
$$= x(7^{k-2}) + 7^{k-1} + 7^k$$
$$= x(7^{k-3}) + 7^{k-2} + 7^{k-1} + 7^k$$
$$= x(7^{k-a}) + 7^{k-a+1} + 7^{k-a+2} + \dots + 7^k$$
$$= \cancel{x(1)}\,7 + 7^1 + 7^2 + 7^3 + \dots + 7^k$$
$$= 7 + \frac{7(7^k - 1)}{7-1}$$
$$= 7 + \frac{7}{6}(7^k - 1)$$
$$= 7 + \frac{7n - 7}{6}$$
$$= \frac{42 + 7n - 7}{6}$$
$$= \frac{35 + 7n}{6}$$

e) $x(n) = x\left(\frac{n}{9}\right) + 5$ for $n > 1$ and $x(1) = 4$ $(n = 9^k)$

$\log n = \log 9^k$

$k = \frac{\log n}{\log 9}$

$= \log_9 n$

$x(n) = x(9^{k-1}) + 5$

$= x(9^{k-2}) + 5 + 5$

$= x(9^{k-3}) + 5 + 5 + 5$

$= x(9^{k-k}) + 5k$

$= x(1) + 5k$

$= 4 + 5 \log_9 n$

Question 4

Algorithm: NetworkTopology (A[0.. n-1, 0.. n-1])

Input: Boolean adjacency matrix A[0.. n-1, 0.. n-1]

Output: Topologies of networks (ring / star / fully connected mesh)

```
count_1 = 0
count_2 = 0
for i ← 0 to n-1 do
    if A[0, i] = true then
        count_1 ++
    end if
    if A[1, i] = true then
        count_2 ++
    end if
end for
if count_1 = 2 then
    return ring
end if
if count_1 = n-1 then
    if count_2 = 1 then
        return star
    else if count_2 = count_1 then
        return fully connected mesh
    end if
end if
```

Question 5

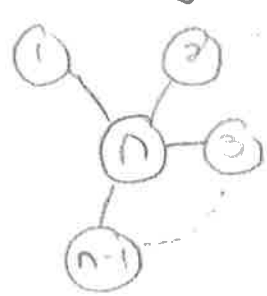
- Yes, it work correctly for every undirected graph with $n>0$ vertices.
- This algorithm determines whether the graph only contain one node and return 1 if it is true.
- If $n>1$ then it runs recursively until the adjacency matrix has only one vertex, then it will return 1
- The algorithm returns 1 if vertex $n-1$ has connection with any vertexes lower than $n-1$, else it returns 0.

Worst case:

- The algorithm calls recursively on $n-1$ vertices in $A[0 \ldots n-2, 0 \ldots n-2]$ and determines whether the last vertex is connected to any remaining vertices.
- This loop runs $n-1$ times.

$$
\begin{aligned}
T(n) &= T(n-1) + n-1 \qquad T(1)=1 \\
&= T(n-2) + n-2 + n-1 \\
&= T(n-3) + n-3 + n-2 + n-1 \\
&= T(n-a) + 2 + 3 + \ldots + n-2 + n-1 \\
&= 1 + 2 + 3 + \ldots n-2 + n-1 \\
&= \frac{n(n-1)}{2} \\
&= \frac{n^2-n}{2} \\
&= O(n^2)
\end{aligned}
$$

- The algorithm does not work on star graph when the middle element is n, because it will just return 0 when it runs recursively on $A[0 \ldots n-2, 0 \ldots n-2]$.



Question 6

a. • The first two axioms of the distance metric listed in Problem 4 is satisfied for the Hamming distance.

• For the third axioms, it is a case of triangle inequality.

• $d_M(p_1, p_2) \leq d_M(p_1, p_3) + d_M(p_3, p_2)$

let length of string = 1

if $p_1 = p_2$, then left hand side is 0

if $p_1 \neq p_2$, then left hand side is 1 and right hand side is definitely greater than or equal to 1

• This is because p_3 can be 0 or 1 and it cannot be the same as both p_1 and p_2.

∴ Hence, it is proven that all 3 axioms of the distance metric is satisfied for the Hamming distance.

b. Comparing two characters in the string of length is m.

∴ The time efficiency for the worst case is

$O(mn^2)$