Santa Clara University
2020 Spring Final Exam

Course	COEN279/AMTH377	Name	Jay Joshi
Department	Computer Engineering	Student ID	00001599322
Lecturer	Yuan Wang	Data/Time	2020/06/12, 2:00pm - 4:00pm
Format	open-book,		
Note	No discussion. No cell phone. no internet		

1. Multiple choices (5 questions, 4points each, 20 points total)

(1) The lower bound of all comparison sort is

✓ a. $\Omega(n\log^n)$
b. $\Omega(n^2)$
c. $\Omega(n)$
d. $O(1)$

(2) The complexity of counting sort is:

a. $\Theta(n\log^n)$
b. $\Theta(n^2)$
✓ c. $\Theta(n)$
d. $\Theta(1)$

(3). The formula $(n-1)*(n-5)$ in terms of big O notation is:

a. $O(1)$
b. $O(\log n)$
c. $O(n)$
✓ d. $O(n^2)$

(4). The complexity of the following program

```
MY_ALGORITHM ( n )
    m = n * n
    return m
```

is:

a. $O(n)$
b. $O(n^2)$
✓ c. $O(1)$
d. $O(m)$

(5) The complexity of the following algorithm :

```
for i = 1 to n
    for j = i to n
        print( i * j)
```

is:

a. $O(n)$
✓ b. $O(n^2)$
c. $O(i * j)$
d. $O(n\log^n)$

2. True or False (10 questions, 2 points each, 20 points total)

(1) A sort technique is said to be stable when the original relative order of records with equal keys are retained after sorting. True

(2) Dynamic Programming and Greedy algorithms are two techniques that can be used interchangeably. False

(3) Dynamic programming improve the efficiency over brute force method by introducing more memory usage. True

(4) Greedy algorithms is used to solve optimization problems. True

(5) All NP-complete problems are NP-hard problems True

(6) Since we are able to find NP complete problems, we know for sure that NP != P False

(7) If problem A is reducible to problem B, that means problem B is easier than problem A. False

(8) For any problem X, if we can find a NP-complete problem C, and C is reducible to X, then X is also NP-complete problem. True

(9) If algorithm has a best case complexity $\Theta(m)$ and a worse case complexity $\Theta(n)$, then the complexity of the algorithm is $\Theta(m)$ False

(10) NP-hard are problems not in NP. True

Ans - 3 input -> {5, 6, 7, 8, 9} for Matrix Chain Multiplication

matrix	A_1	A_2	A_3	A_4
dimension	5×6	6×7	7×8	8×9

i \ j	1	2	3	4
1	0	210	490	850
2		0	336	768
3			0	504
4				0

We need to use below eduation to calculate entry in above table,

$$M[i,j] = \min \begin{cases} 0 & , \text{ if } i=j \\ \min\{M[i,K] + M[K+1,j] + P_{i-1} * P_k * P_j\} & \text{ if } i<j \end{cases}$$

$M[1,2] = M[1,1] + M[2,2] + 5*6*7 = 210$

$M[2,3] = M[2,2] + M[3,3] + 6*7*8 = 336$

$M[3,4] = M[3,3] + M[4,4] + 7*8*9 = 504$

$$M[1,3] = \min \begin{cases} M[1,1] + M[2,3] + 5*6*8 = 576 \\ M[1,2] + M[3,3] + 5*7*8 = 490 & (k=2) \end{cases}$$

$$M[2,4] = \min \begin{cases} M[2,2] + M[3,4] + 6*7*9 = 882 \\ M[2,3] + M[4,4] + 6*8*9 = 768 & (k=3) \end{cases}$$

$$M[1,4] = \min \begin{cases} M[1,1] + M[2,4] + 5*6*9 = 1038 \\ M[1,2] + M[3,4] + 5*7*9 = 1029 \\ M[1,3] + M[4,4] + 5*8*9 = 850 & (k=3) \end{cases}$$

We can construct Solution table as per below,

i \ j	2	3	4
1	1	2	3
2		2	3
3			3

4. [15 points] Fibonacci number is defined like this:

$$F(n) = \begin{cases} 0 & \text{for } n = 0 \\ 1 & \text{for } n = 1 \\ F(n-1) + F(n-2) & \text{for } n > 1 \end{cases}$$

so, for example, F(6) = 8 (because up to position 6, all Fibonacci numbers are: 0 1 1 2 3 5 8)

The straight forward top down recursive algorithm to calculate fibonacci number is:

```
int FibTopDown(int n) {
    if(n==0) return 1;
    if(n==1) return 1;

    fib[n] = fibTopDown(n-1) + fibTopDown(n-2);
    return fib[n];
}
```

This problem can be solved using dynamic programming algorithm. Add the memoization to this recursive algorithm.

```
int FibTopDown(int n, int lookup[ ]){
    if(n==0){
        lookup[n]=n;
    }
    ~~else~~ eif(n==1){
        lookup[n]=n;
    }
    ~~if~~
    ~~else~~
    if lookup[n] is Null{
        lookup[n] = FibTopDown(n-1, lookup) +
                    FibTopDown(n-2, lookup);
    }
    return lookup[n];
}
```

Ans - 5 Independent Set Problem

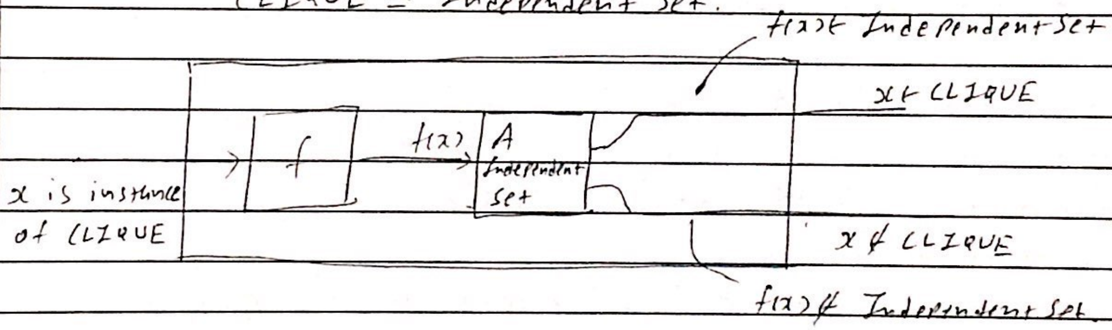
Here the decision Problem is
does the graph has a independent set of size 3?

Proof! -

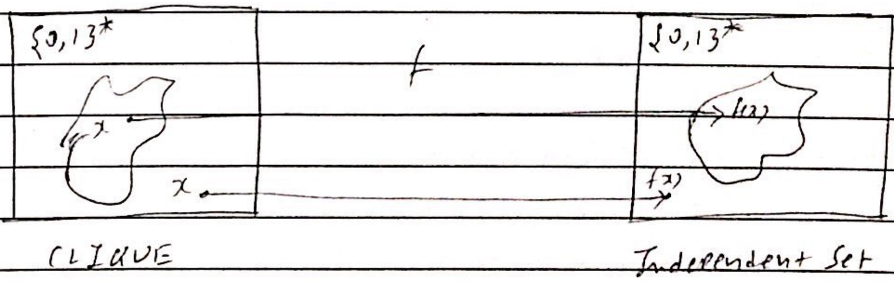
(1) Independent Set ∈ NP
given a solution, we can verify it in Polynomial time

(2) A NPcomplete Problem CLIQUE can be reduced to Independent set problem

CLIQUE ≤ Independent Set.



$\bar{G}$ is a graph/Independent set) containing exactly those edges that are not in given graph G.



① if x is Satisfiable then,
graph has a Independent Set of size 3.
and has a Clique of size 4.

② if there is a Independent Set of size 3 then,
there must be 7 Vertices in the graph
because, Independent Set size = V - Size of Clique
$= 7 - 4 = 3.$

③ F is polynomial function.

Proof is done!

∴ CLIQUE can be reduced to Independent Set, or in other words, CLIQUE can be solved by using Independent Set Problem algorithm.

Therefore, Independent Set Problem is NP-Complete problem.

Scanned with CamScanner