ADSA EndSem

NAME: N. KAUSIK

ROLL NO: CS21M037

Question Paper Downloaded: 8:58 am

Network Turned Off: 8:58 am

Total Number of Sheets: 20

Honour Code: I have not engaged in any dishonest means to answer this question paper. I have maintained a high level of decency and dignity in attempting this question paper. I understand that my quest for knowledge is life-long, while this exam will end in just 180 minutes. I understand with pride, dignity and decency that I do not have to resort to cheating in exams.

9:00 am 20/11/2021
Chennai, Tamil Nadu
600042

1) For Quick Sort,

Worst case is when at each iteration (recursion), the array of size 'n' is partitioned into 2 subarrays of sizes $n-1$ and 0 (excluding pivot element).

eg. If we take pivot element as first element, for an array which is in descending order, quick sort always partitions into $n-1$ and 0.

Worst case T.C $\Rightarrow T(n) = T(n-1) + T(0) + \Theta(n)$

By expanding, $T(n) = n \cdot \Theta(n) = \Theta(n^2)$.

For Randomised Quick Sort,

Here, even though we pick a random element as pivot, still the worst case is that at each iteration (recursion) we pick the pivot such that array is partitioned as $n-1$ and 0.

$\therefore$ Worst case T.C $\Rightarrow \Theta(n^2)$

Worst Case T.C of Quick Sort and Randomised Quick Sort is <u>Same</u> and is $\Theta(n^2)$.

For expected time of Randomised Quick Sort, during each partition, on average, the array is split into 2 parts where both have a fraction of elements of full array. $(\frac{n}{2}, \frac{n}{2})$.

∴ the recursion tree will have a depth (expected) of $\log n$. To partition, time taken is $\Theta(n)$.

Hence Expected T.C $= \Theta(n \log n)$.

2) $f_1(n) = \begin{cases} n & , \text{ if } n \text{ is even} \\ 2n & , \text{ if } n \text{ is odd} \end{cases}$

$f_2(n) = \begin{cases} 2n+1 & , \text{ if } n \text{ is even} \\ n & , \text{ if } n \text{ is odd} \end{cases}$

Suppose n is even,

$f_1(n) = n$, $f_2(n) = 2n+1$

For $c_1 = \frac{1}{4}$, $c_2 = 1$, $n_0 = 1$,

$c_1 f_2(n) = \frac{n}{2} + \frac{1}{4} \leq f_1(n) = n \leq 2n+1 = c_2 f_2(n)$ $\forall n \geq n_0$.

$\therefore f_1(n) = \Theta(f_2(n))$ if n is even.

Suppose n is odd,

$f_1(n) = 2n$, $f_2(n) = n$

For $c_1 = 1$, $c_2 = 4$, $n_0 = 1$,

$c_1 f_2(n) = n \leq f_1(n) = 2n \leq 4n = c_2 f_2(n)$ $\forall n \geq n_0$

$\therefore f_1(n) = \Theta(f_2(n))$ if n is odd.

$\therefore$ Asymptotic relationship is that,

$$f_1(n) = \Theta(f_2(n))$$

3) Rahul Baadal Priya Jose Ameen
2 5 4 2 5

Init array C = [0 0 0 0 0]
and records R = [Null Null Null Null Null]

For Looping through records,

→ Rahul : 2 C = [0, 1, 0, 0, 0]
R = [., (Rahul), ., ., .] [. → Null]

→ Baadal : 5 C = [0, 1, 0, 0, 1]
R = [., (Rahul), ., ., (Baadal)]

→ Priya : 4 C = [0, 1, 0, 1, 1]
R = [., (Rahul), ., (Priya), (Baadal)]

→ Jose : 2 C = [0, 2, 0, 1, 1]
R = [., (Rahul, Jose), ., (Priya), (Baadal)]

→ Ameen : 5 C = [0, 2, 0, 1, 2]
R = [., (Rahul, Jose), ., (Priya), (Baadal, Ameen)]

Output Seated Order,
Rahul, Jose, Priya, Baadal, Ameen
2 2 4 5 5

4) A universal class of hash functions is defined as a collection of hash functions that map a universe U of keys to $\{0, 1, 2, \ldots m-1\}$ that satisfies, for every pair of distinct keys $k, l \in U$, number of hash functions h belonging to the class of hash functions where $h(k) = h(l)$ is AT MOST $\frac{|\mathcal{H}|}{m}$ where $\mathcal{H}$ is the given class of hash functions.

Example,

$$\mathcal{H} = \{ h_{ab} : a \in Z'_p , b \in Z_p \}$$

where $Z_p = \{0, 1, \ldots p-1\}$, $Z'_p = \{1, 2, \ldots p-1\}$

$$h_{ab}(k) = ((ak+b) \bmod p) \bmod m.$$

For example we can take $p = 37$ for $m = 12$.

Then $h_{7,8}(11) = ((7 \times 11 + 8) \bmod 37) \bmod 12$

$= (85 \bmod 37) \bmod 12 = 11 \bmod 12$

$= 11$,,

This example is indeed universal as,

if we consider 2 keys k, l, $k \neq l$,

Let,

$r = (ak+b) \bmod p$

$s = (al+b) \bmod p$ $\Rightarrow r - s = a(k-l) \bmod p$

Since $k \neq l$, $k-l \neq 0$, $a \neq 0$ and as p is prime,

$r - s \neq 0 \Rightarrow r \neq s$.

$\therefore$ Solving, $a = ((r-s)(k-l)^{-1} \bmod p) \bmod p$.

$b = (r - ak) \bmod p$.

Since we choose a ad b uniformly at random,
$\therefore$ r and s are uniformly likely to be any pair of values $(r \neq s)$.

# of possibilities $= p(p-1)$

For k and l to collide, $r \neq s$ but $r = s \pmod m$.
If we take any r, there are $p-1$ possibilities for s for $r \neq s$. For $s = r \pmod m$,

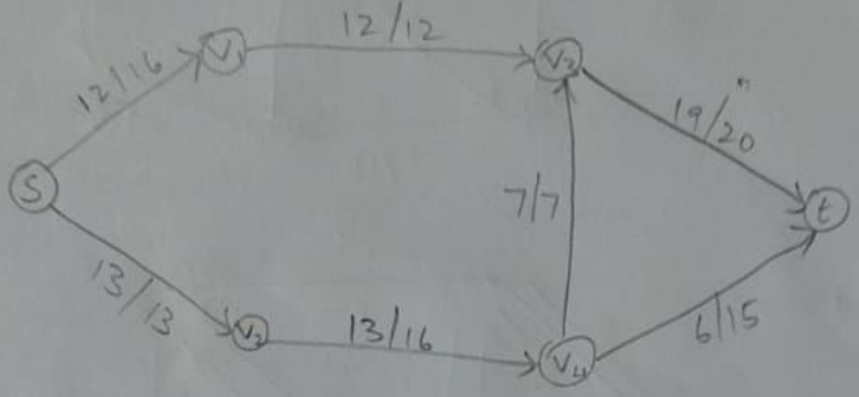
$$\# = \lceil p/m \rceil - 1 \leq \frac{p-1}{m}.$$

$\therefore$ Prob that s collides with $r \leq \frac{\left(\frac{p-1}{m}\right)}{p-1} = \frac{1}{m}$.

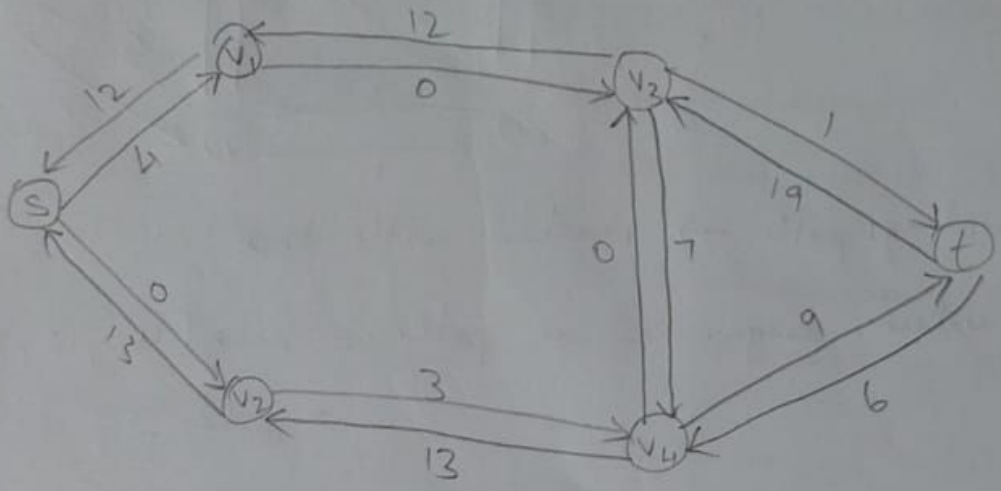
$\therefore$ The condition is satisfied, given example is a universal family of hash functions.

5) a) 25

b)



c)



d) Min s-t cut in residual network ⇒

$S = \{s, v_1, v_2\}$ $T = \{v_3, v_4, t\}$

net flow = 3 − 12 − 13 = −22

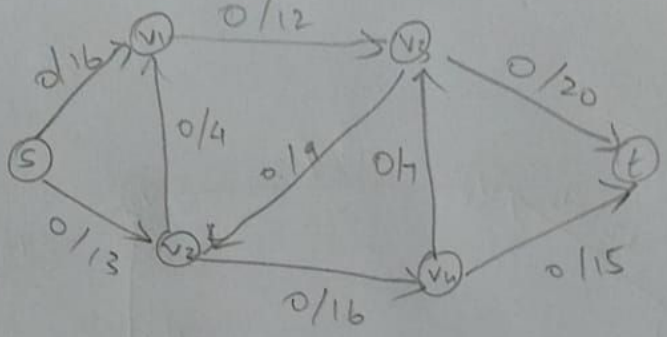
Min s-t cut in given flow network ⇒

$S = \{s, v_1, v_2\}$ $T = \{v_3, v_4, t\}$

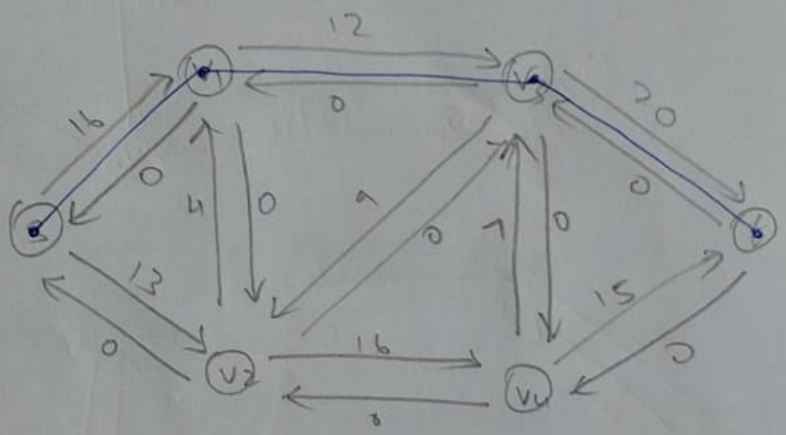
net flow = 12 + 11 − 4 = 19

Both cuts are same.

c) Initial flow,



Residual Network,



Augmenting path → marked with Pea

flow value augmented on path = min(16, 12, 20)

= 12.

6.) i) $u.d < u.f$

ii) Possibilities,

$u.d < u.f < v.d < v.f$

$u.d < v.d < v.f < u.f$

Other 2 cases with u and v reversed.

iii) Edges, (u, v)

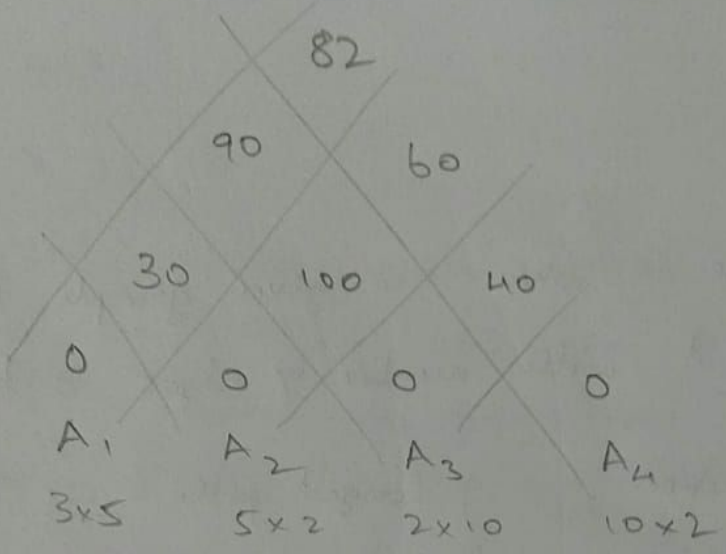
→ Tree Edge : Edge present in the DFS tree

→ Forward Edge : If v is a descendent of u but (u, v) is not part of DFS tree

→ Back Edge : If u is a descendent of v but (u, v) is not part of DFS tree

→ Cross Edge : u and v are not related as descendent of one another.

7)



8)

9) a)

b) yes instance is when the given graph G_1 has a path in it with number of edges $\geq k$.

c) witness function = { Longest path, if # edges of longest path $\geq k$; any path, otherwise }

Verifier ⇒ i) Verify if witness gives a valid path in G_1 (all edges in path are in G_1)

ii) check # edges in path $\geq k$.

Return 1 if both are satisfied, else 0.

∴ This is NP.

10. 2-SAT,

i) It is in P.

ii) 2-SAT can be converted into a graph problem and solved using BFS / DFS. T.C ⇒ O(V+E) and so it is solved in poly time.
Hence P.

3-SAT,

i) It is NP hard and is in NP

ii) ∴ It is in NP-Complete

iii) It doesnt have a deteeministic poly time algo.

11)

Part II

1) Selection Problem,

Input: Array of values A, i - Select i^{th} minimum.

Output: Value of i^{th} minimum.

A) If we divide into groups of 3,

the time comp becomes,

$$T(n) = T\left(\lceil \frac{n}{3} \rceil\right) + T\left(\frac{4n}{6}\right) + O(n) \geq T\left(\frac{n}{3}\right) + T\left(\frac{2n}{3}\right) + O(n)$$

$$T(n) \geq c\left(\frac{n}{3}\right)\log\left(\frac{n}{3}\right) + c\left(\frac{2n}{3}\right)\log\left(\frac{2n}{3}\right) + O(n)$$

If we assume for nlogn.

$$\geq cn\log n + O(n)$$

$\therefore T(n)$ is NOT linear.

B) If divide by 7,

$$T(n) \leq T\left(\frac{n}{7}\right) + T\left(\frac{10n}{14}\right) + O(n) \leq cn\left(\frac{1}{7} + \frac{5}{7}\right) + O(n)$$

$$\leq O(n)$$

$\therefore T(n)$ is linear.

2) No, it will not give optimum number of mult.

Example,

$(2\times1) \cdot (1\times2) \cdot (2\times5)$
$A_1 \qquad A_2 \qquad A_3$

This algorithm,

does $(A_1 \cdot A_2)\, A_3 \Rightarrow$ # mult $= 2\times1\times2 + 2\times2\times5$
$= 24$

But optimum is

$A_1 (A_2 \cdot A_3) \Rightarrow$ # mult $= 1\times2\times5 + 2\times1\times5$
$= 20.$

This is a greedy algorithm and it does not guarantee optimum solution for MCM.

This is because if we choose based on min mult, it might result in a larger matrix which causes large # of mult in later stages.

3) A) The greedy strategy selects, $\{a_1, a_2, a_3, a_4\}$, it rejects a_5 since its deadline is 1 and its penalty is least among $\{a_1 \ldots a_5\}$.

It also rejects a_6 as its deadline is 4 but there are $\{a_1, a_2, a_3, a_4\} \Rightarrow 4$ tasks with higher penalties before it.

It accepts a_7.

Schedule $\Rightarrow \{a_2, a_4, a_1, a_3, a_7, a_5, a_6\}$

Penalty $\Rightarrow w_5 + w_6 = 30 + 20 = 50$

B)

a_i	1	2	3	4	5	6	7
d_i	4	2	4	3	1	4	6
w_i	10	20	30	40	50	60	70

By greedy strategy,

a_1 and a_2 are rejected as they have least weights.

All other tasks are accepted.

$\therefore$ Schedule $\Rightarrow \{a_5, a_3, a_4, a_6, a_7, a_1, a_2\}$

Penalty $\Rightarrow w_1 + w_2 = 10 + 20 = 30$

7) a) P - Deterministic Polynomial time algorithm exists

NP - Non-deterministic Polynomial time algorithm exists

b) P is set of all decision problems that can be solved by a deterministic polynomial time algorithm.

NP is set of all decision problems that can be solved by a non-deterministic polynomial time algorithm.

$P \subseteq NP$

c) Decision Problem, M is max bipartite matching,

$\forall M' : |M| \geq |M'|$ is True

M' is a matching

This belongs to NP.

d) Decision Problem, f is max flow

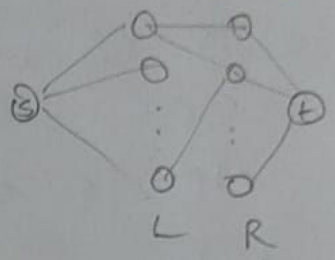
$\forall f' : |f| \geq |f'|$ is True

f' is a valid flow

Since this can be solved using Ford Fulkerson Method, and it is poly time algo, it is in P.

e) To reduce the max bipartite graph problem to a max flow problem,

i) Add a source s and terminal t vertices to the bipartite graph.
Connect s to all vertices in L
t to all vertices in R.



ii) Give unit capacity to every edge.
Also we restrict that flow is always integer valued.
So, any edge can be either 0 or 1.

iii) Then we solve maximum flow and the resultant without s and t is the maximum bipartite graph.

8) 3-Colouring problem

It is where we give colours (any one of 3 colours) to the nodes of a graph such that no 2 vertices connected by an edge have the same colour.

A node can be assigned any 1 colour among 3.

It is in NP as we can verify if any adjacent nodes have same colour in poly time.

It is in NP hard as it can we can do a reduction from 3-SAT problem which is NP-hard.

∴ It is NP-Complete.