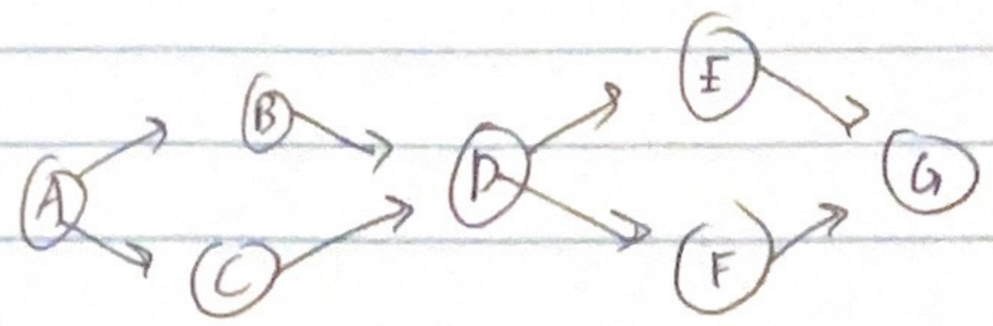
HW-8

Abhilash Ambati

1.) let G1 be the graph



let V and V1 represent A and B.
finding path from V to V1

Step 1: Toplogical sort (linear time)

A B C D E F G.

step 2: apply forward propegation

	1	1	1	1+1=2	2	2
T_S =	A	B	C	D	E	F
	↑	↑	↑	↑	↑	↑

If our pointer is pointing at V
then mark the V node with 1 as in 2 stack
we look at A and B as so $\therefore$ 1 to = 1.

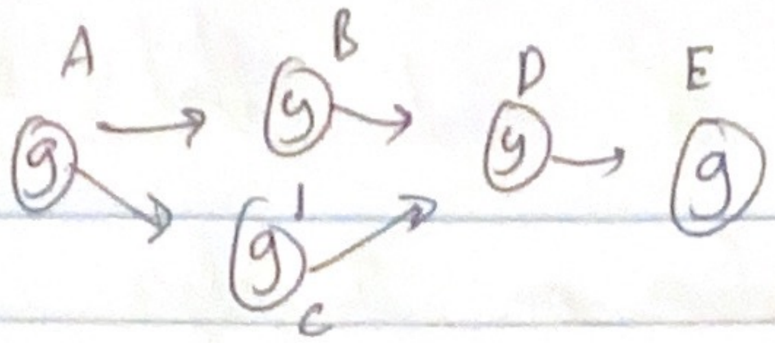
B = 1.

then check. A to C since we get to
another Node E to = 1 C = 1

we keep repeating for B and D, C to D
and D to E → E-2.

$\therefore$. there are two path's to F from A.

2.)

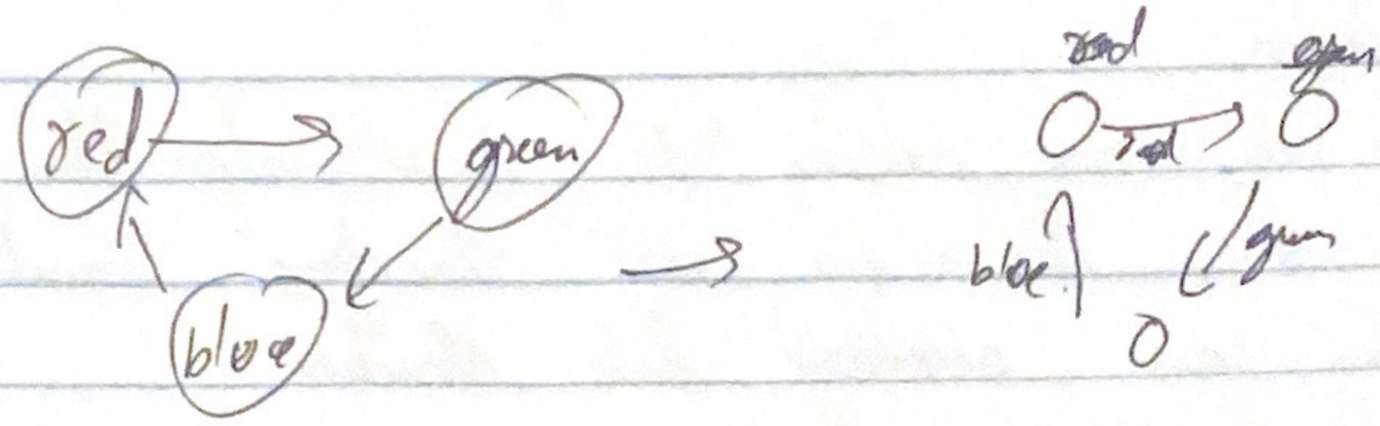


we are using the same algorithm as problem 1 but in this we are tracking multiple ways using an auxiliary data structure

there are three colonyes in our table and each node will hold one colony by using forward propogation, we will make sure that each table holds the final results.

now for g>y act sum the paths in the node, we get 1+1=2 paths.

3.) convert G into G' such that G' is a graph with colours on the edges.



now use FA and cartesian product rules.

G(colors on edge)
U.V (nodes on Is)
T (regular expression).

Suppose: M_1 : recognize the graph G's and node U,V
M_2 : recognize at regular expression r.

now find the ~~cost~~ cartesian product M.

$M_1 \times M_2 = M$ such that M color sequence satisfying r from U to V.

now. we have two cases:

① M is a DAG here we use the Alg from. Prof Lin if.

② M is ~~not~~ not A DAG so we apply SCC

1.) reverse every edge, keep ids on nodes.

2) run DFS again, storing u, nodes that are most recently finish

mark all edges that are still remain
Remain non-marked edges.

4) Remain node still connects are part of SCC
the rest are their own SCC.

DFS will show SCCS. M or SS has > 1 node return ∞.

6) count = 0.

start with either 0," or 1101, then we either have 1101 to go to final state or we repeat outside count ++