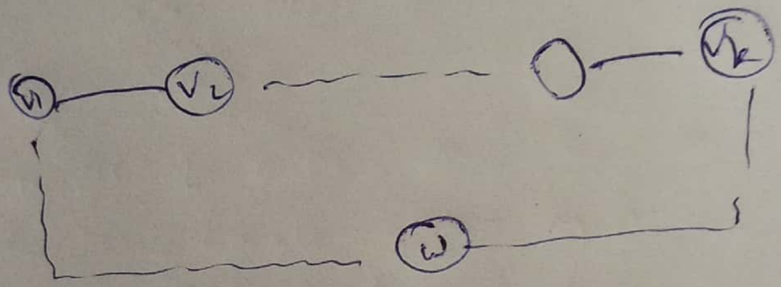
Dt 20/05/2020 Assignment - I

Name - Atul Kumar Agrawal
Branch - CSE - II
Roll No - 1602010031

4.1 Consider a network that consists of a path of nodes v_1 through v_k, & a node w connected to both v_1 & v_k, but its links go up & down repeatedly.

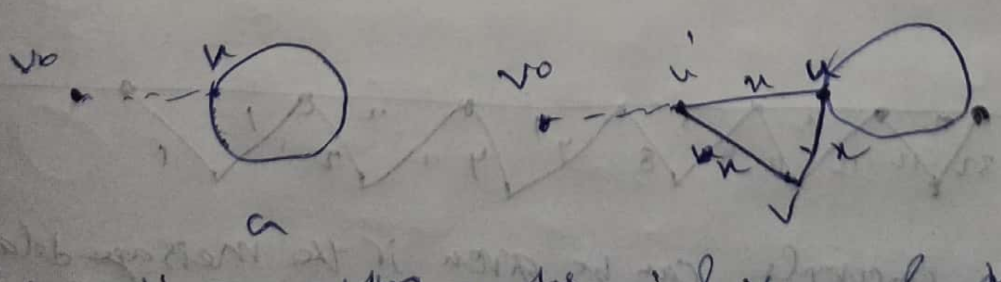


Consider a packet m with destination w, & assume the link $v_1 w$ is down; packet must move in the direction of v_k i.e. to the right. When the packet arrives in v_k, the link to w goes down but $v_1 w$ comes up; after adaptation of the routing tables the packet will move to v_1 i.e., to the left. When m arrives there, the link to w fails, but $v_k w$ is restored again, & we are back in the situation we started with.

In this example the network is always connected, yet m never reaches its destination even if the routing algo is "perfect" & instantaneously adapts tables to the current topology.

4.2 Yes, the modification is possible. A message containing D_w is relayed to those neighbours from which a (ys, w) message is received. Unfortunately, neighbouring processes may now be several rounds apart in the execution of the algorithm, i.e., a process may receive this message while already processing several pivots further. This implies that the D_w table must be stored also during later rounds, in order to be able to reply to (ys, w) messages appropriately. This causes the space complexity of the algorithm to become quadratic in N. The message complexity is reduced to quadratic, but the bit complexity remains the same because we unfortunately save only on small messages.

4.3 To construct the example, let G be a part of the network, not containing v_0, connected to v_0 only through node u in G, & with the property that if $D_u[v_0]$ improves by x while no mydist-messages are in transit in G, then k messages may be exchanged in G:-

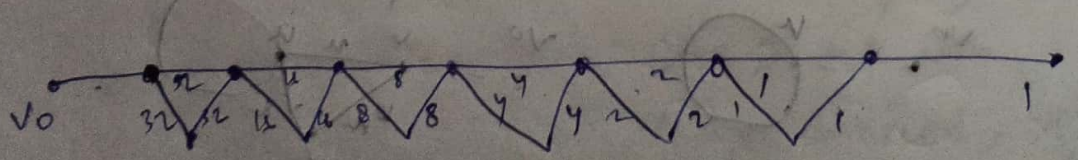


→ Construct G' by adding nodes u' & v, & three edges uu' & u'v & uv, each of weight x, & G' is connected to v_0 & only via u'. When no messages are in

Dangit in G', $D_u[V_0] = D_{u'}[V_0] + x$, & assume in this situation $D_u[V_0]$ improves by $2x$ (due to the receipt of a mydist-message). Consider the following scenario, in which u is informed about the improvement in 2 steps by following information via v.

1. u' sends an improved mydist-message to v & u.
2. v receives the message, improves its estimate, & sends an improved mydist-message to u.
3. u receives message from v & improves the estimate by x, which causes k messages to be exchanged in G.
4. u receives the mydist-message from u' & improves its estimate again by x, causing another k messages to be exchanged in G.

It is seen that when $D_{u'}[V_0]$ improves by $2x$, more than $2k$ messages may be exchanged in G'. By iterating the construction the no. of nodes & edges grows linearly, while the no. of messages grows exponentially; the resulting graph as follows:-



A similar example can be given if the message delays are guaranteed to lie within a very small range. The assumption of the shortest path measure is, however, necessary; in the minimum hop measures the complexity of the algorithm is bounded by $O(N.E)$ messages.

4.4 the following table gives the values of $D_u[v]$ & b/w Parenthesis, the value or possible values of $Nb_u[v]$. Recompute is non-deterministic w.r.t. the selection of a preferred neighbour if the minimal estimate occurs multiply.

for each neighbour w of u,

$Ndis_u[v,w]$ equals $D_w[v]$.

V \ u	A	B	C	D	E	F
A	0 (loc)	1 (A)	4 (F)	1 (A)	2 (B/D)	3 (E)
B	1 (B)	0 (loc)	3 (F)	2 (A/E)	1 (B)	2 (E)
C	4 (B/D)	3 (E)	0 (loc)	3 (E)	2 (F)	1 (C)
D	1 (D)	2 (A/E)	3 (F)	0 (loc)	1 (D)	2 (E)
E	2 (B/D)	1 (E)	2 (F)	1 (E)	0 (loc)	1 (E)
F	3 (B/D)	2 (E)	1 (F)	2 (E)	1 (F)	0 (loc)

following algorithm 4.9, node F sends upon receiving the <repair, A> message all entries of its distance table i.e. messages; <mydist, A, 3> <mydist, B, 2>, <mydist, C, 1>, <mydist, D, 2>, <mydist, E, 1> & <mydist, F, 0>.

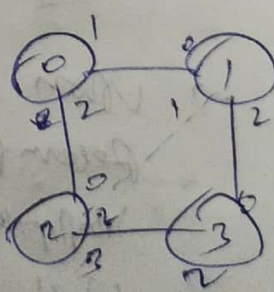
→ Upon receipt of each of these 6 messages, Recompute is executed in A & leads to an improvement for destinations F & C, after which A sends messages <mydist, F, 1> & <mydist, C, 2>.

4.5 Let G be a ring of N processes, where the cost of each edge in clockwise direction is 1 & in anti-clockwise direction the cost is k, then D_G is approximately $\frac{k}{k+1}N$. A spanning tree of G is

obtained by removing a single edge, & the distance b/w the 2 separated nodes is $N-1$ in one direction but $k.(N-1)$ in the other direction; this is about $k+1$ times D_G.

4.6 Let the label of link uw (in node u) be α_{uw}; if $\alpha_{uw} \neq l_u$, the link is used when a message with address α_{uw} is generated in u, if $\alpha_{uw} = l_u$ but the label $\alpha_{uw}+1$ doesn't occur in u, the link is used when a message with address $\alpha_{uw}+1$ is generated in u. But if $\alpha_{uw} = l_u$ & the label $\alpha_{uw}+1$ also occurs in u, the link is never used.

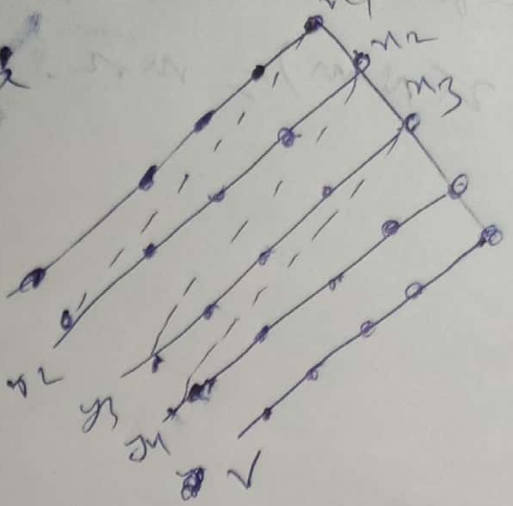
- This picture shows an example of an ILS where this occurs for the edge 12 on both sides; hence the edge 12 is not used for traffic in neither direction. The scheme is valid.



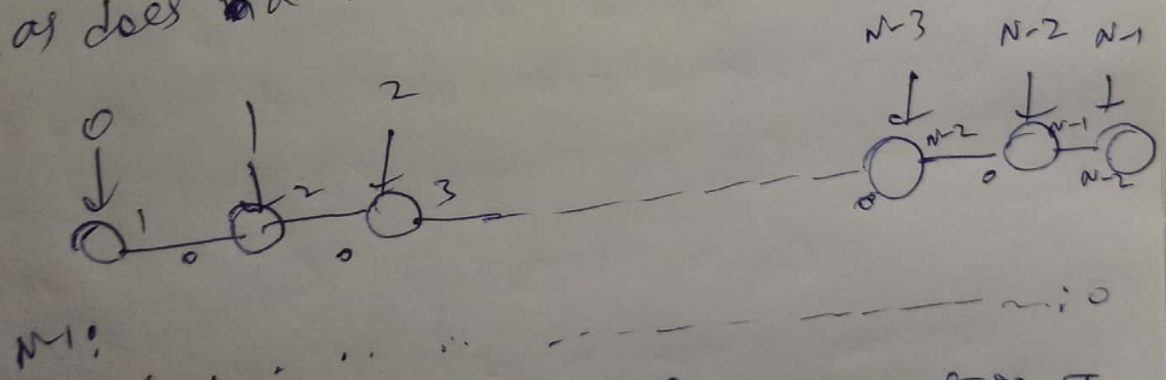
- A scheme not using edge uw is not optimal because traffic b/w u & w is sent over 2 or more hops while $d(u,w) = 1$.

4.7

Exploit figure 4.16 to design a detour running through all nodes of the network, In the network, such a detour is made by messages from u for. the nodes marked n_i send the message down the tree via the dotted front edges belong y_{i+1} has a node label b/w n_{i+1} & v (both edges are labelled with node label of the adjacent node).



4.8 A DFS tree in a ring has depth N-1; the ILS is as indicated here, and a message from 0 to N-2 travels via N-2 hops (as does a message in the opposite direction).



4.9 (1) Besides minimality, the only requirement on T is that it contains all c_i; consequently, every leaf of T is one of the c_i (otherwise, a leaf could be removed, contradicting minimality).

(2) Because a tree has N-1 edges the node degrees sum up to 2(N-1) & leaves have degree 1. With m leaves, there are N-m nodes with degree at least 2, so, the no. of nodes with degree... sum up to 2N-m-2.

degree ~~with~~ larger than 2 is at most $2N - m - 2 -$

$2(N-m) = m - 2$.