THE HOPS-SO-FAR SCHEME

⇒ k is the length of the longest path in the routing network.

$k < N$ ∵ All routing paths are acyclic

ASSUMPTION.

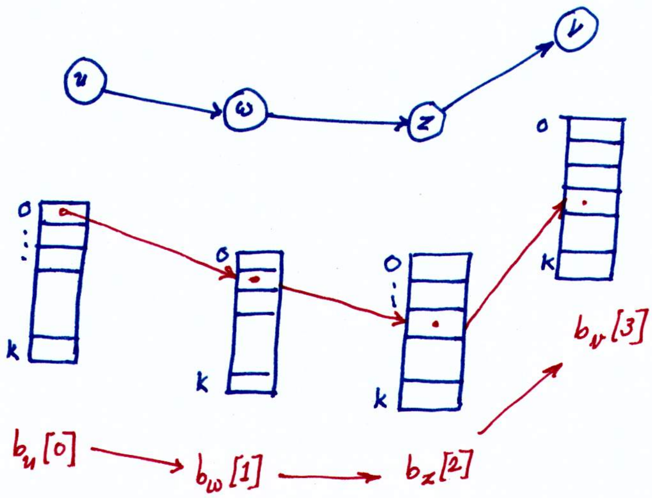
Each packet contains a hop count.

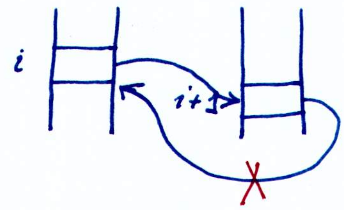
The Buffer Graph is $BG = (B, E)$

$b_u[i]\ b_w[j] \in E$ iff $\underbrace{i+1 = j}$ $\wedge$ uw is an edge of the network

↳ Every time the packet makes a hop, it uses one plane.

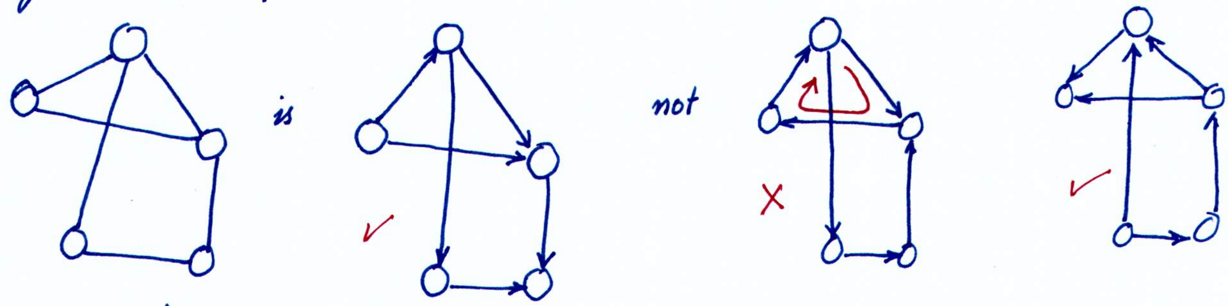
To have a deadlock, we need to have a cyclic wait condition.



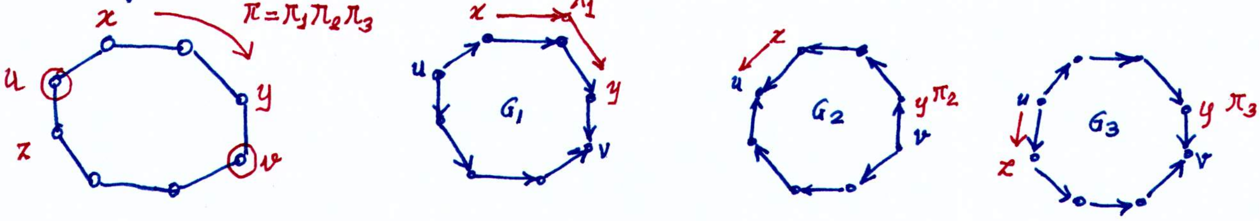


ACYCLIC ORIENTATION BASED SCHEME

An acyclic orientation of



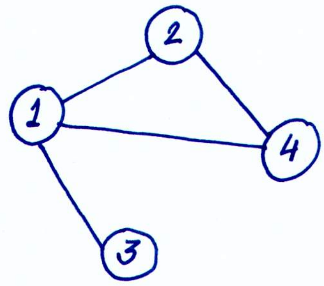
An acyclic orientation cover:



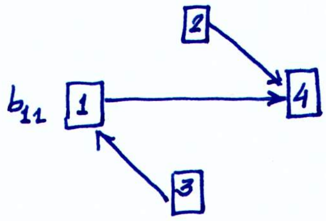
For a ring we have an acyclic orientation cover of size 3.

Algorithms differ on how buffer graphs are constructed.

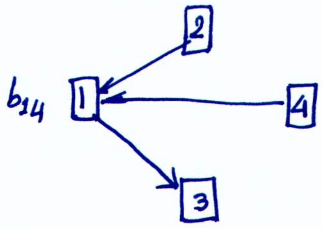
THE DESTINATION SCHEME



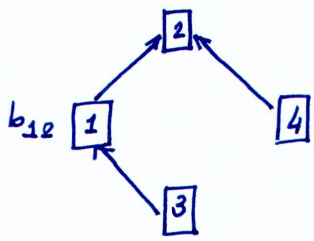
How are packets with destination 4 routed?



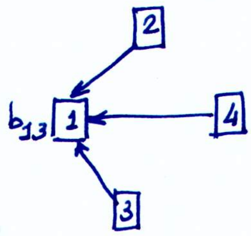
Routing to 3



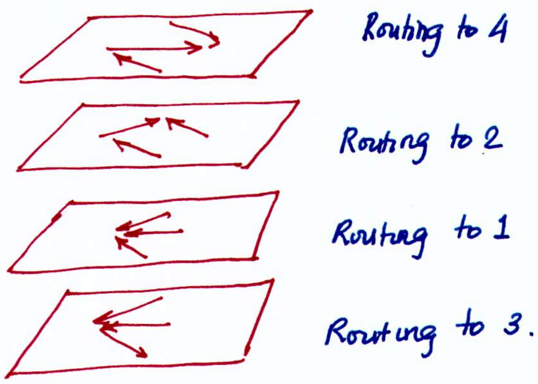
How are packets routed to 2

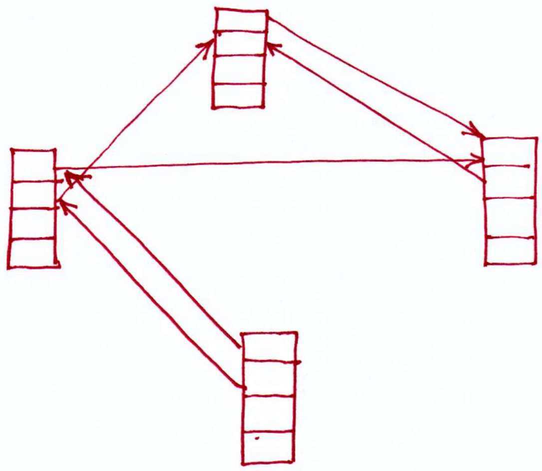


Routing to 1



Every node has N buffers. We look at the graph as a collection of 4 graphs in 4 different planes.





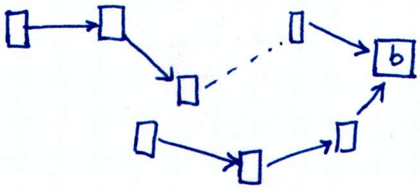
⇒ Each plane is a tree

⇒ No interconnection between the planes

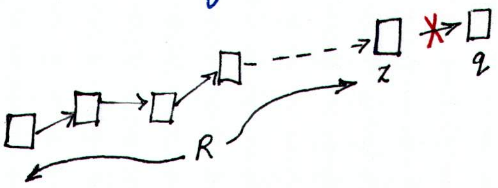
∴ All BG paths are acyclic.

Buffer Class

The buffer class of b is the length of the longest path in BG that ends in b.



Let R be the highest buffer class.



Packets in z can only be consumed.

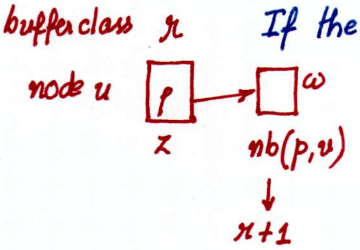
$\Rightarrow$ We use induction on R to show that for all buffer classes deadlocks cannot happen.

Buffer class $r < R$

Induction hypothesis:

For each r' with $r < r' \leq R$ no buffer of buffer class r' contains a deadlocked packet.

For nodes having $r = R$, they cannot have deadlocked packets.

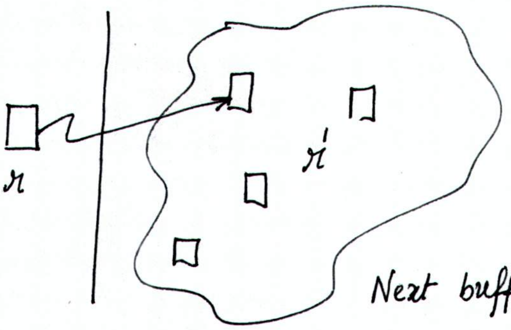


If the buffer has a packet with destination u, it is consumed.

Does not contain a deadlocked packet. $\therefore$ It will eventually become free. and p can move there.

What did we prove?

In an acyclic graph we can never have a cyclic wait of buffers.



Next buffer class.

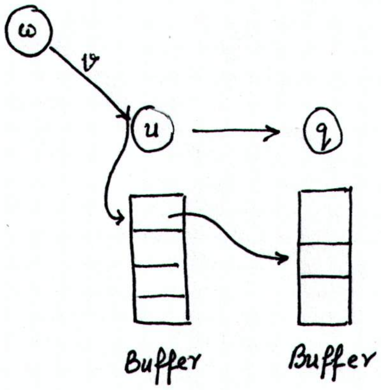
How do we construct the buffer graph?? (Using local information only)

The BG is virtually constructed in a distributed fashion.

Optimize : No. of buffers.

Deadlocks in Distributed Systems

Store and Forward Deadlock



⇒ Intermediate nodes receive packets for diff^n destinations

∴ Buffering is req^d

⇒ Packets are routed from a buffer in one node to the buffer of another node.

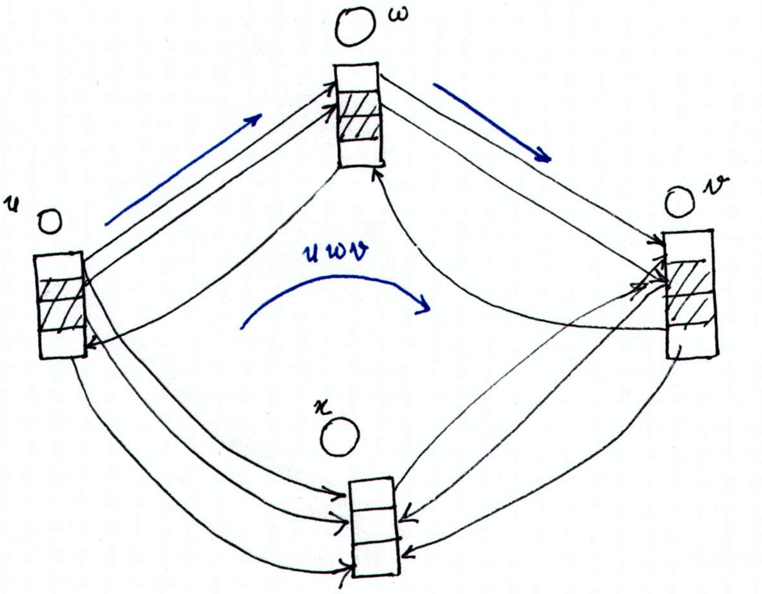
⇒ Buffers are of finite size

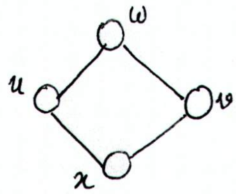
∴ Possibility of Deadlocks

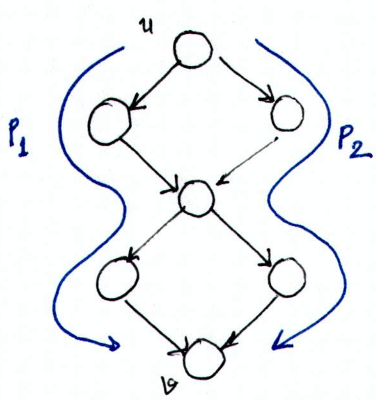
Deadlock Prevention is required.

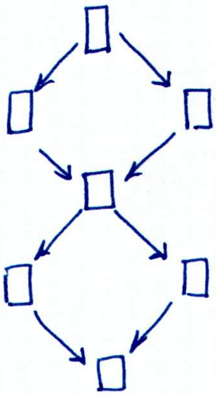
A Buffer management strategy to decide when to accept / forward a packet to a node v, and to which buffer to send it. such that no deadlocks occur.

Buffer Graphs









If any acyclic orientation cover for P of size B exists, then there exists a deadlock-free controller using only B buffers in each node.

Let $G_1, \ldots, G_B$ be the cover. For node u: $b_u[1], \ldots, b_u[B]$

We write $uw \in \vec{E_i}$ if edge uw is directed towards w in G_i.

The buffer graph is defined as follows:

$$b_u[i]\ b_w[j] \in \overrightarrow{BE} \quad \text{iff} \quad uw \in E \text{ and } (i = j \wedge uw \in \vec{E_i}) \text{ or } (i+1 = j \wedge wu \in \vec{E_i})$$

For a tree only 2 Buffers are required.