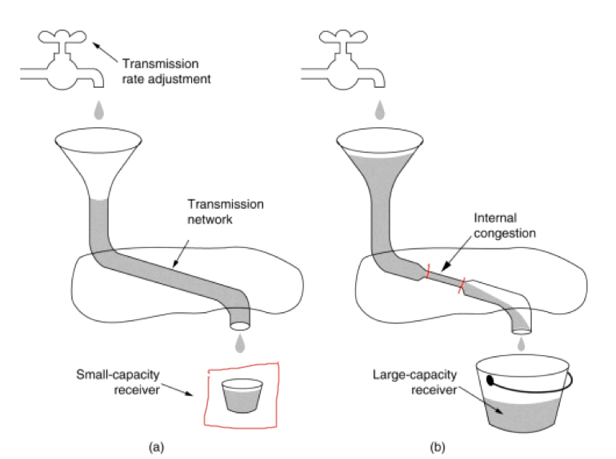
We have to regulate the speed of water so that the glass doesn’t get over-flown.

  
A-part

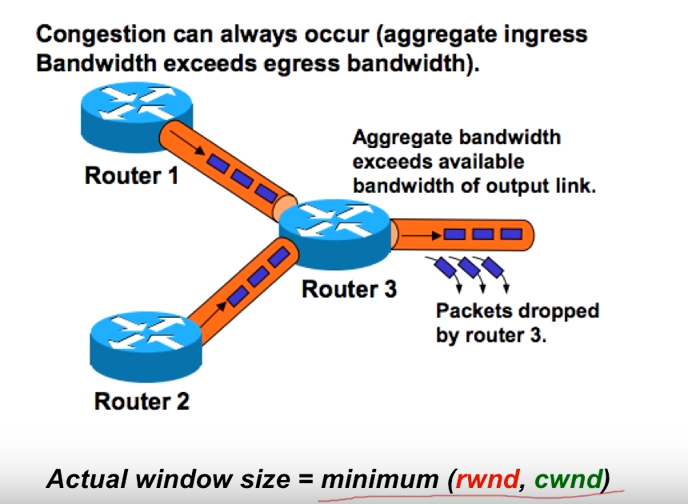
* Even-though we have the small capacity receiver, the pipe is free of congestion. Hence the receiver will not be over-flow.

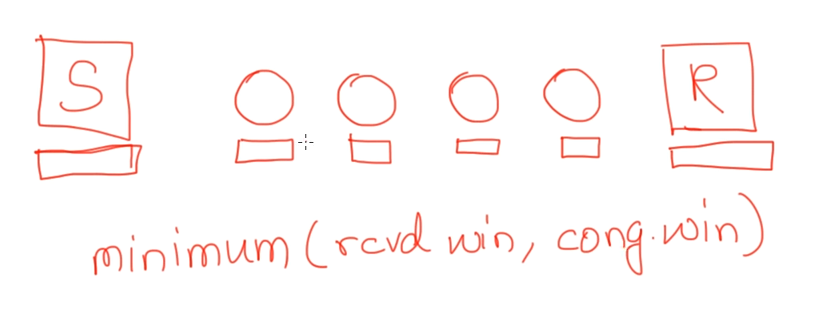
B-part

* The smaller part (causing congestion) cannot able to process the water as the normal pipe.
* As a result, the pipe is congested and the water is up-to the funnel.

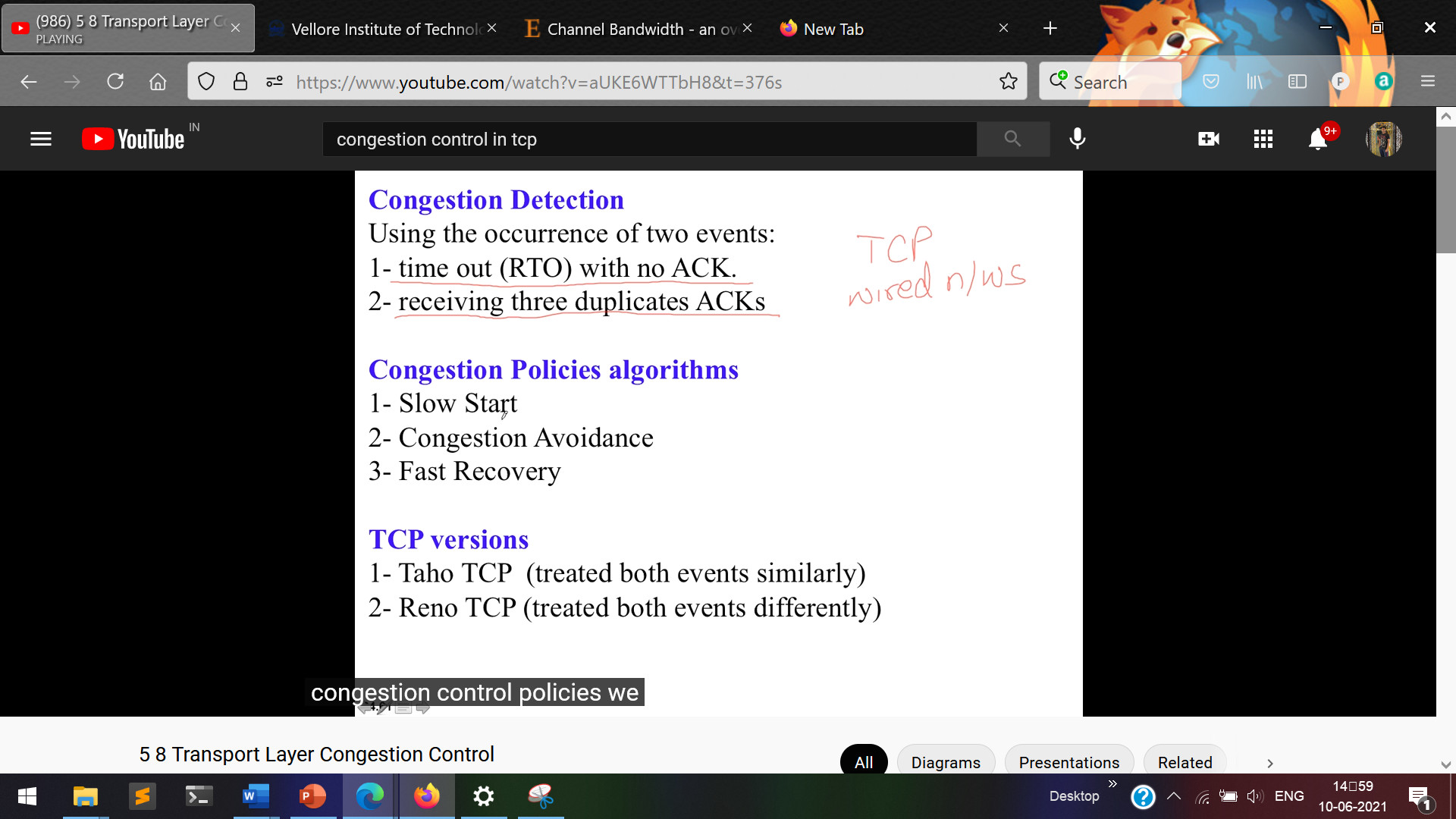
Device to device 🡪 Flow control   
Process to Process 🡪 Congestion control

In normal road, there will be traffic control mechanism  
In data-transport

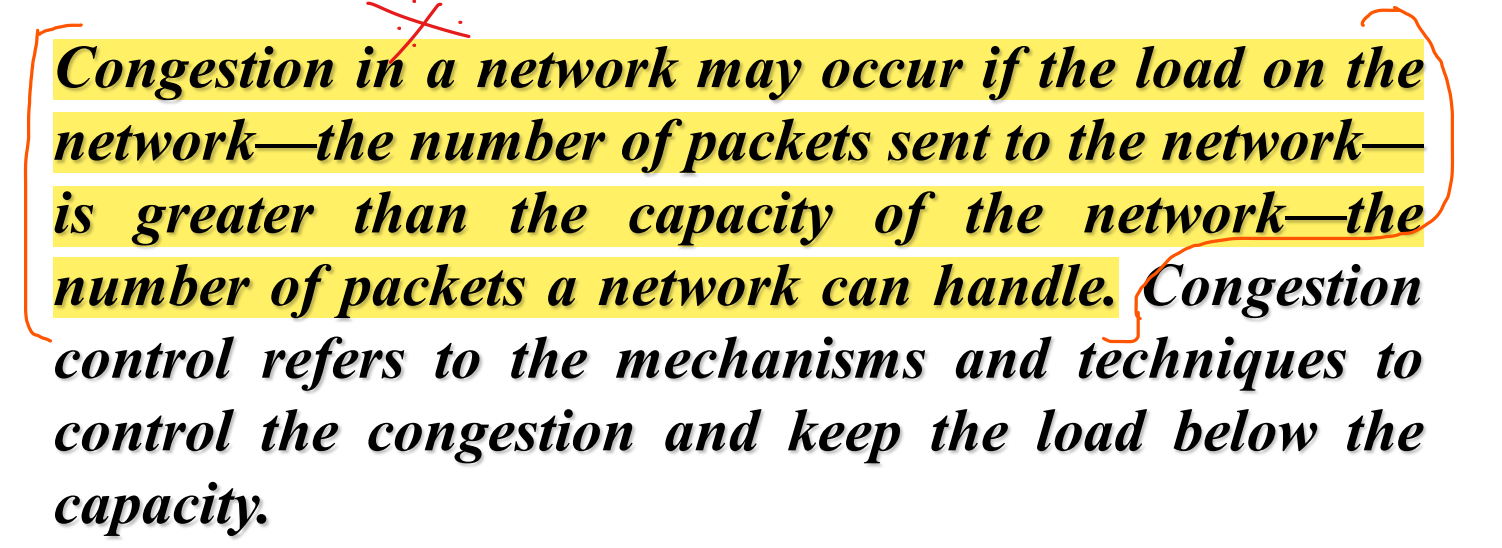
  
Router-3 cannot able to proceed further with router-1 and router-2 packets. So it will drop the packets.



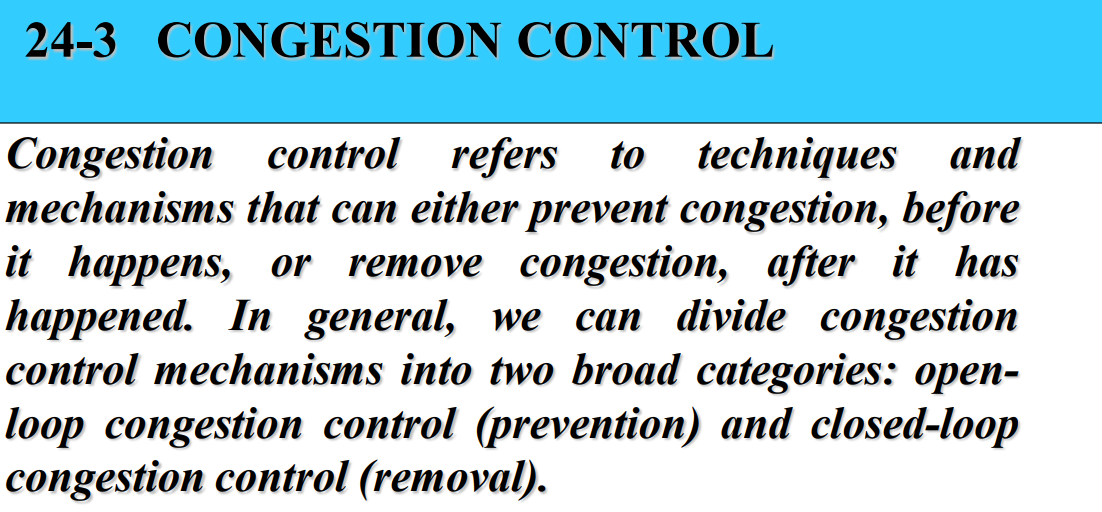
* Sender, receiver window 🡪 some-what similar size.
* In-between routers 🡪 buffers will be smaller in size.
* Sender sends the data-packets from their window, and the inter-mediate routers cannot able to handle these packets, so these routers are being flooded with packets.
* Sender-window doesn’t depend only upon the receiver window but also depends upon the intermediate router buffer. So there will be a high chance of losing the packets.
* ***There-fore the sender's window size is determined not only by the receiver but also by congestion in the network.***



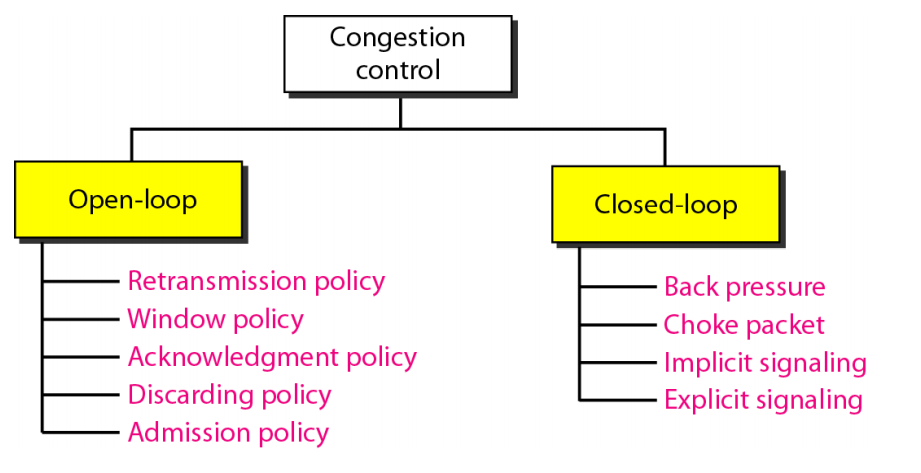
# **Congestion**



# **Congestion Control**

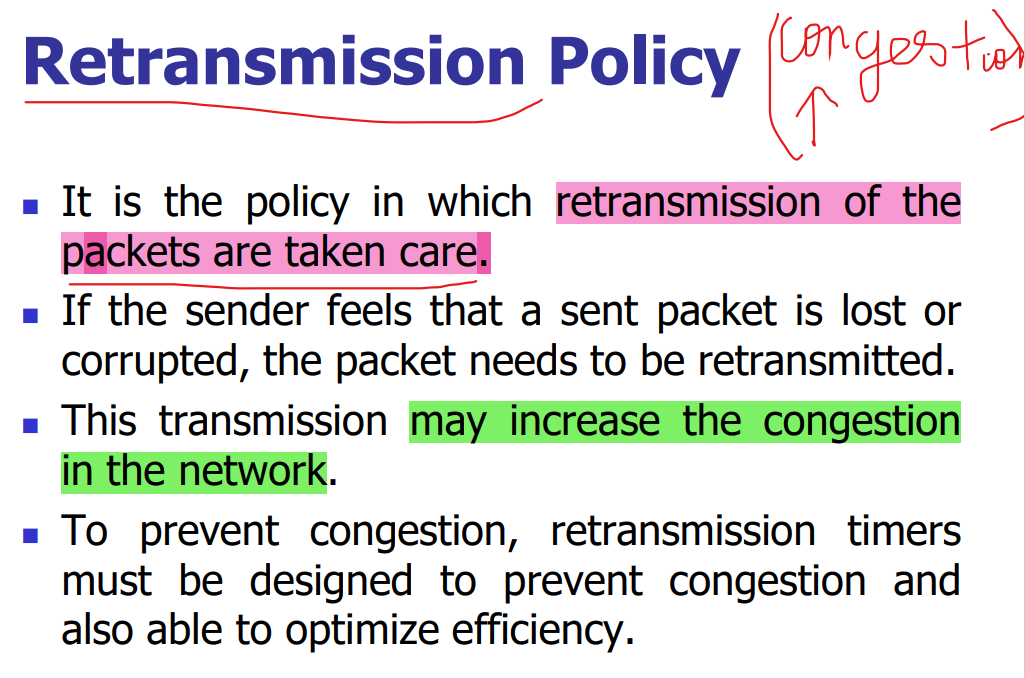


Open Loop Congestion Control 🡪 Prevent the congestion before it happens.  
Close Loop Congestion Control 🡪 Removes the congestion after it happens.



# **Open Loop Congestion Control**

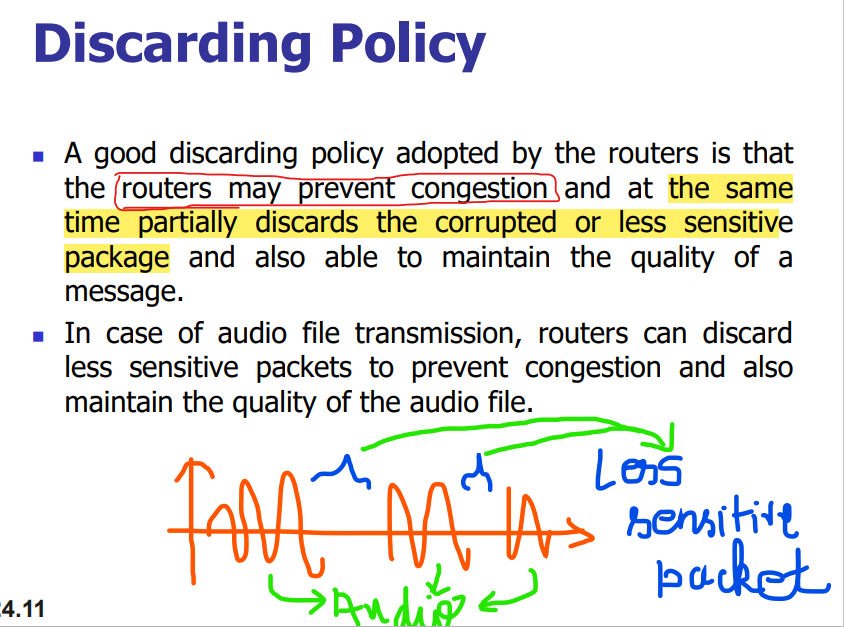
## **Retransmission Policy**



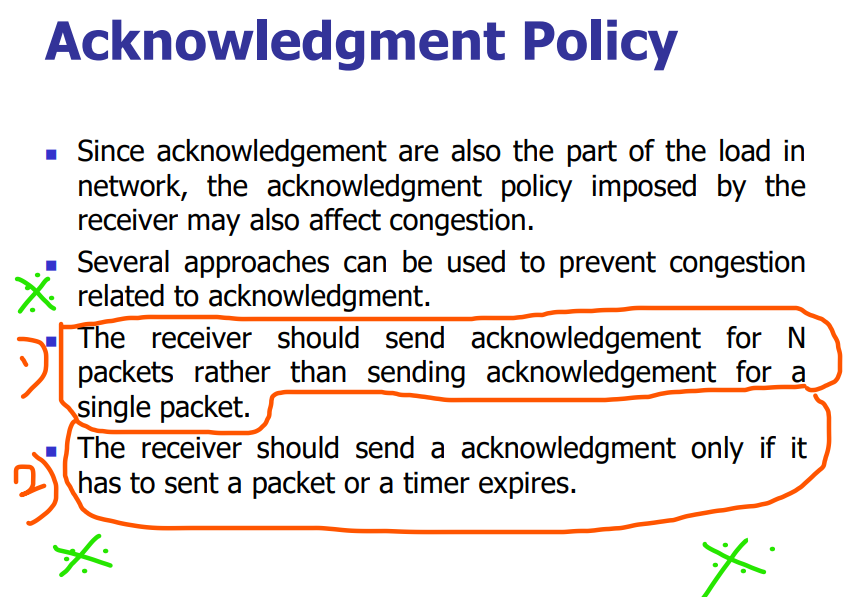
## **Window Policy**



## **Discarding Policy**

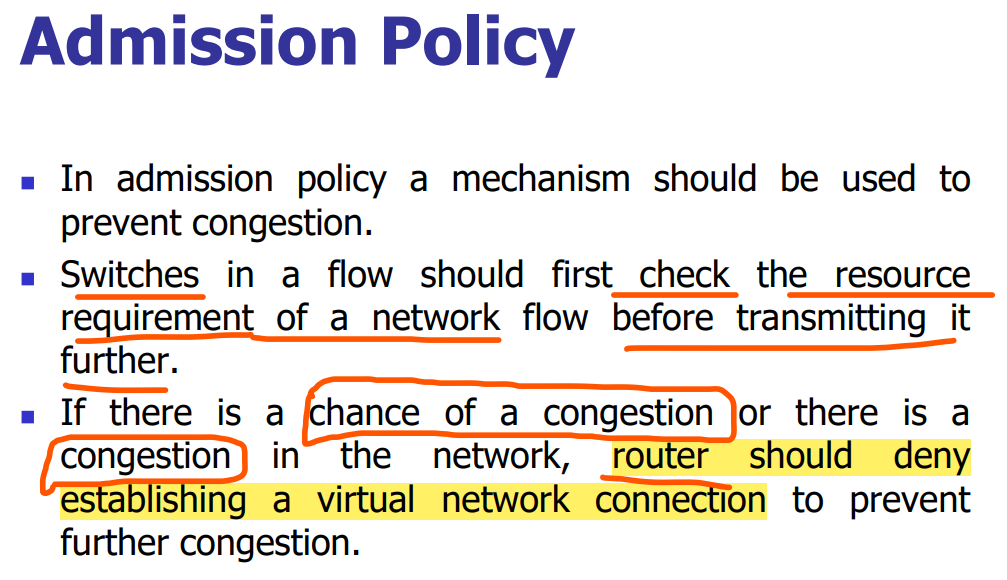
  
Corrupted packets are dropped.  
We cannot simply drop the packet. There are also some mechanism to drop this packet.

## **Acknowledgement Policy**



By reducing the number of acknowledgment packets, the congestion can be reduced.  
Receiver should not send the acknowledgment for each and every packet, instead receiver should send the acknowledgment for N packets.

## **Admission Policy**

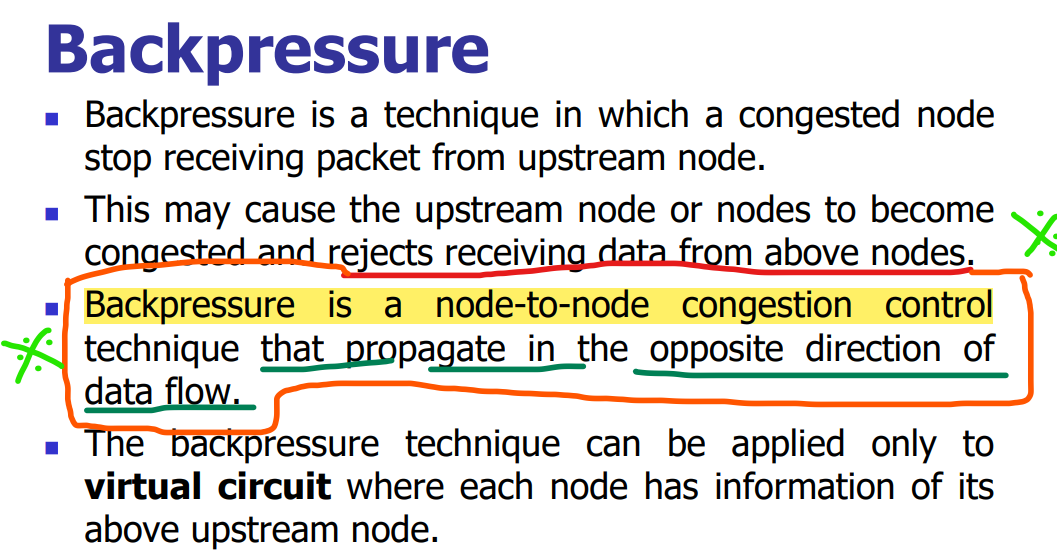
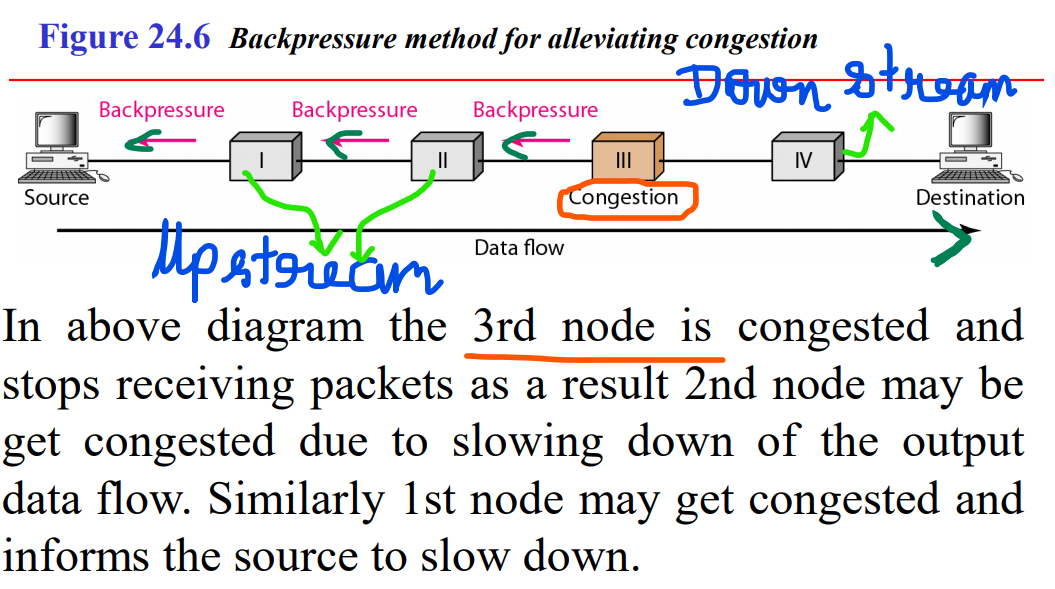


# **Closed-Loop Congestion Control**

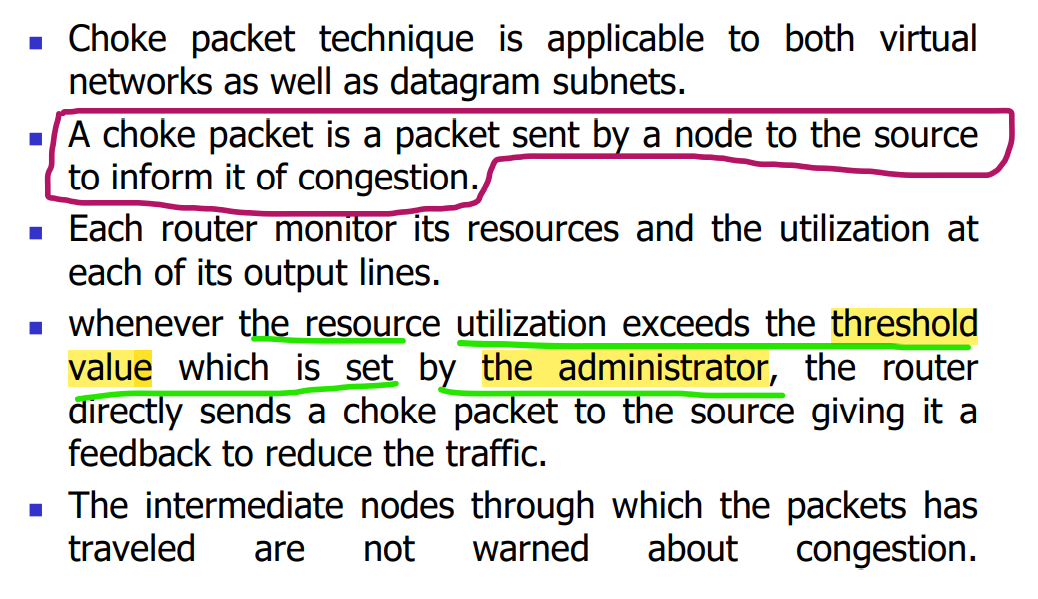
Closed loop congestion control technique is used to treat or alleviate congestion after it happens.

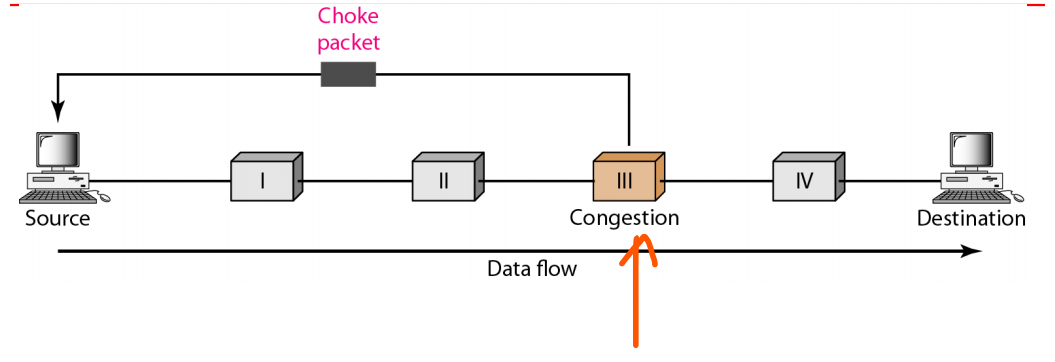
## **Back Pressure**

Eg: Dam  
Dams are built across rivers in-order to store the water rather than letting the river water to mix-up with the sea.  
During storing the water, the flow of the water is stopped/slowed down and due to this some back pressure is created which further resists the back-ward movement of the water.

  
Forward movement 🡪 Data-flow   
Back-ward movement 🡪 Back-Pressure

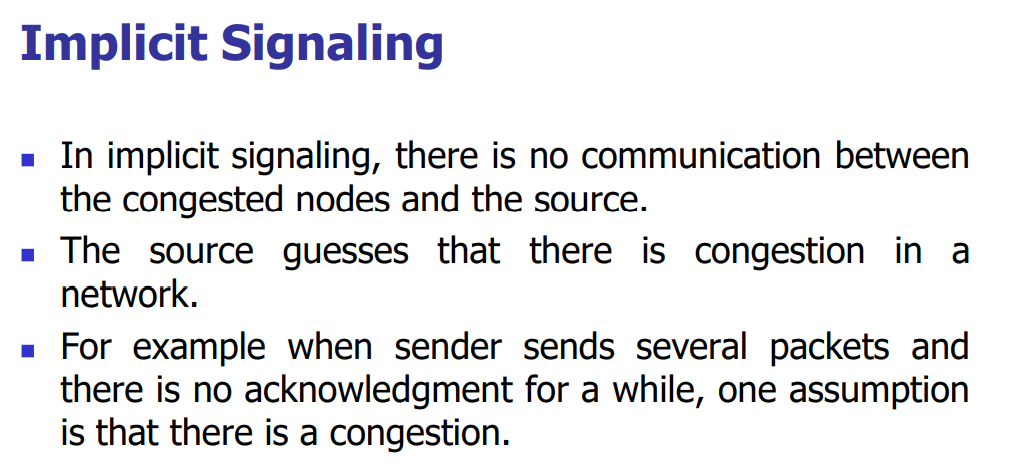
## **Choke Packet**

Congested Router 🡪 A congestion is there, source please stop sending the further packets.  
The intermediate router will not warn about the congestion. These intermediated nodes still passes the packets. Only the congested router will send the choked packet to the source.  




* This congested router’s utilization exceeds the threshold value set by the administrator, so it sends the choked packet to the source.
* Unlike the back-pressure, the congestion node will not disturb the up-stream routers.
* Choke packet are sent directly from congested node to the source.

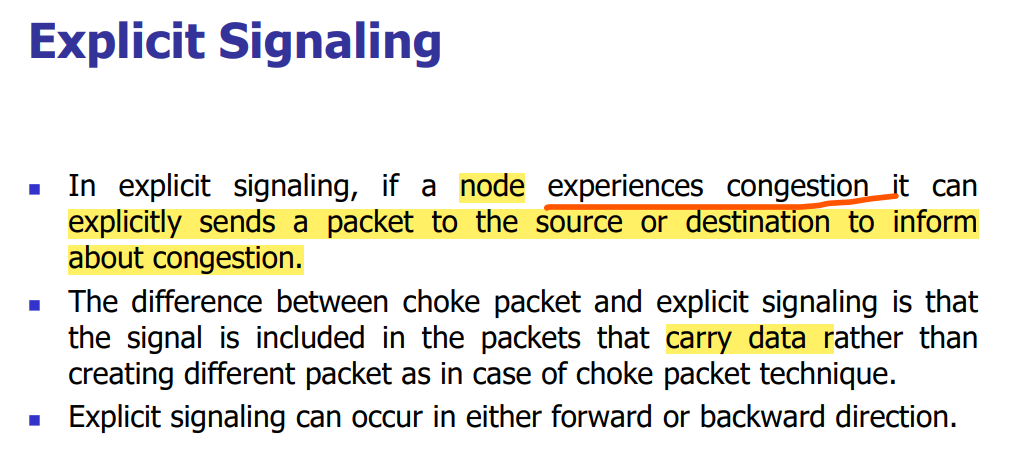
## **Implicit Signaling**



**Sender**  
1st 1 hour sends and receives the acknowledgement without any delay.  
2nd 1 hour sends and receives the acknowledgement with 1 minute delay.  
3rd 1 hour sends and receives the acknowledgement with 5 minute delay.

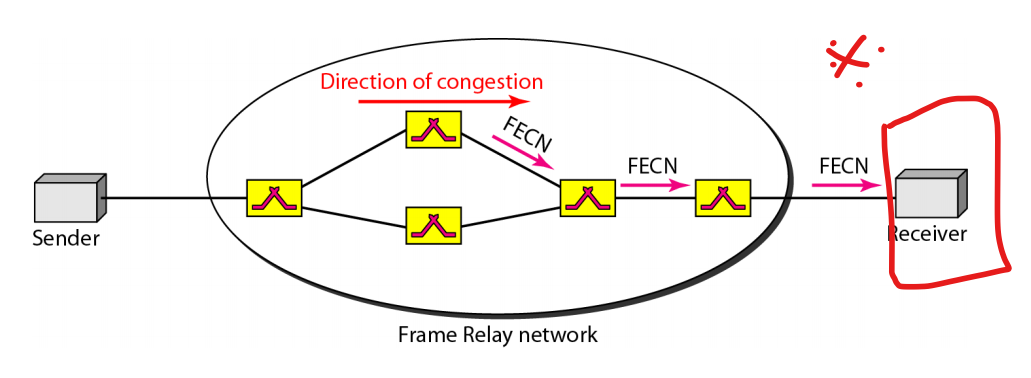
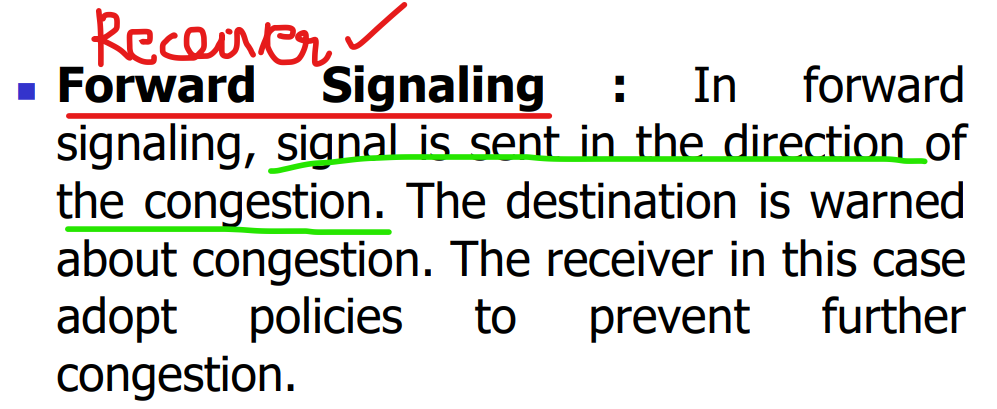
* Now the sender can clearly understand that there is some congestion in-between so I will reduce the packets. This is implicit.
* No routers in-between told the sender that there is some congestion like choke packer, the ***sender by-itself understands*** that there is some congestion in-between by seeing the delay for the acknowledgement.

## **Explcit Signaling**

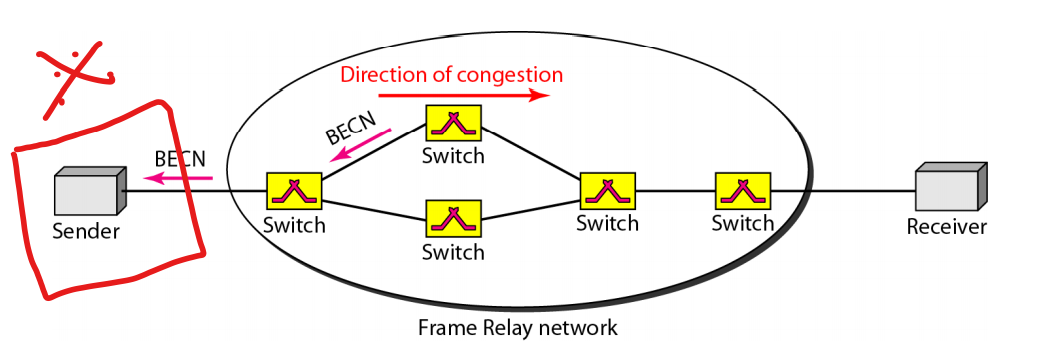


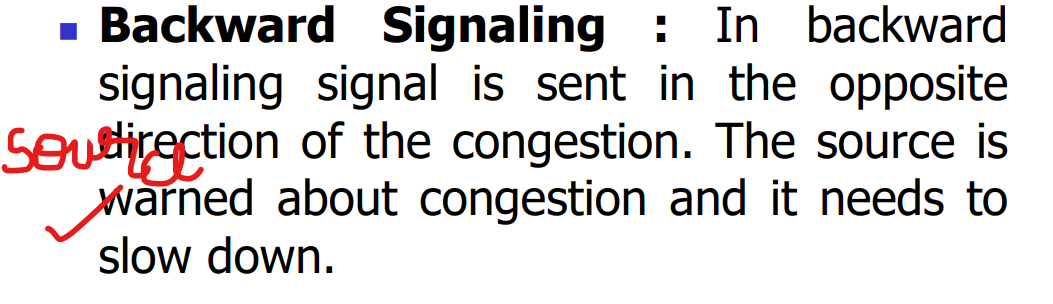
Choke packet 🡪 a special extra packet will be sent to the source to inform about the congestion. Only sent to the source.  
Explicit Signaling 🡪 Data + info about the congestion in a particular router. This is sent both to the source and destination.

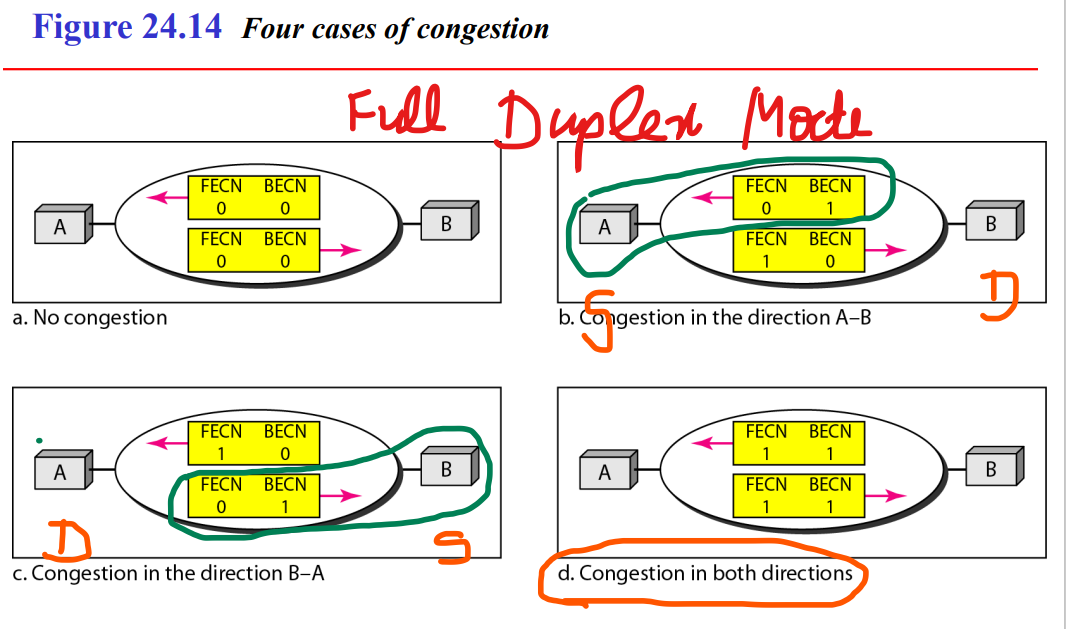
### **Forward Explicit Congestion Signaling**

### **Backward Explicit Congestion Signaling**





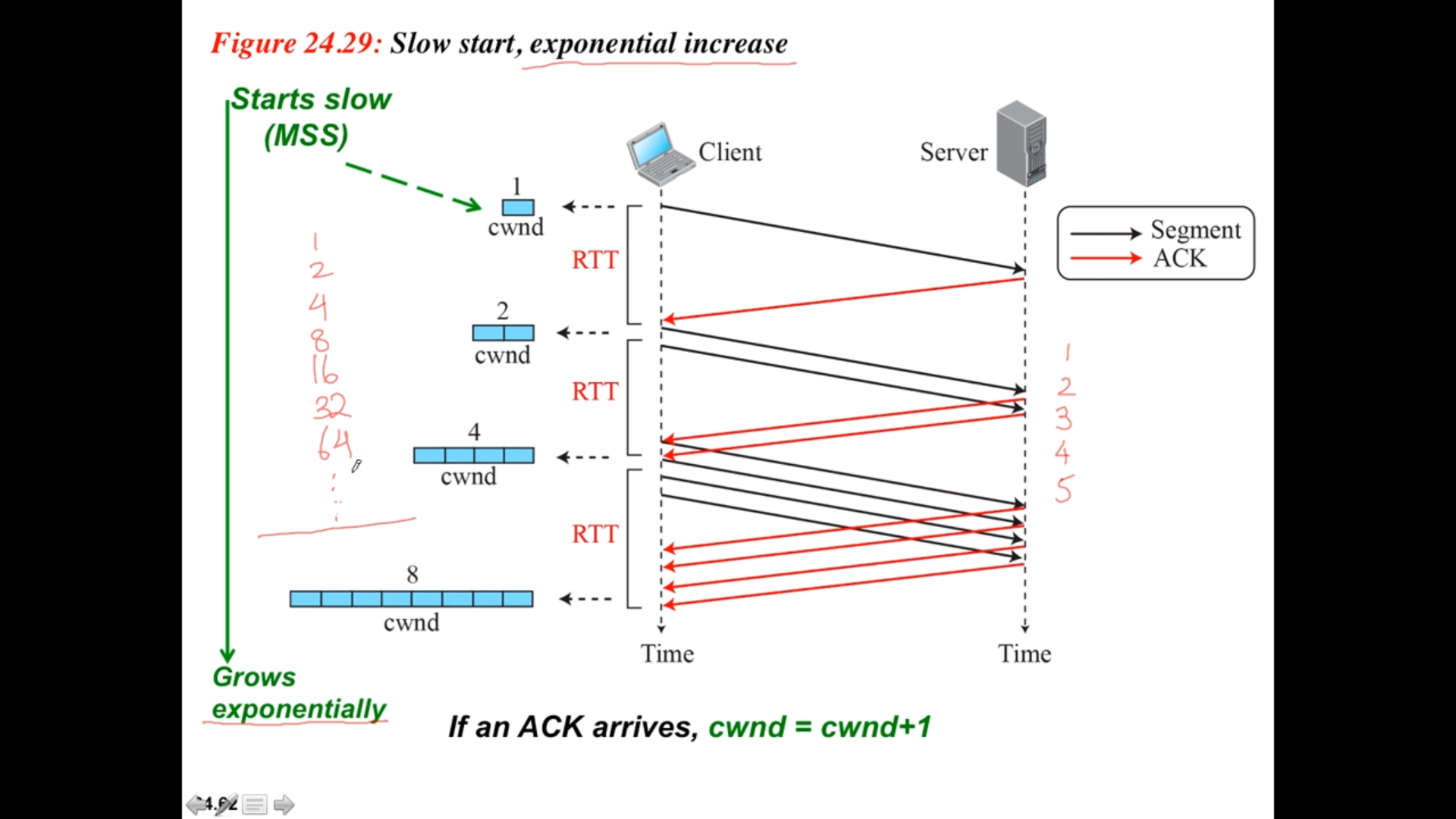


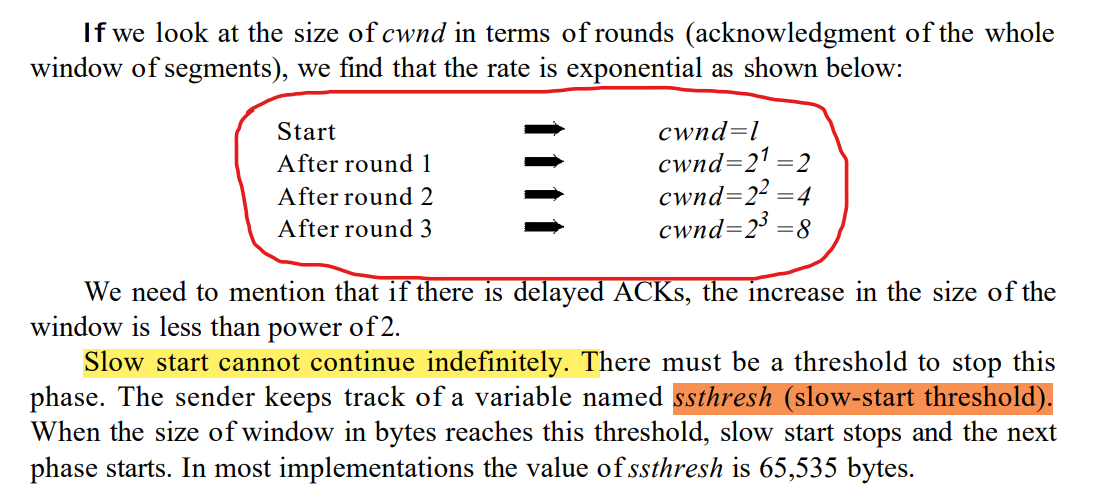
# **Congestion-Control in TCP**

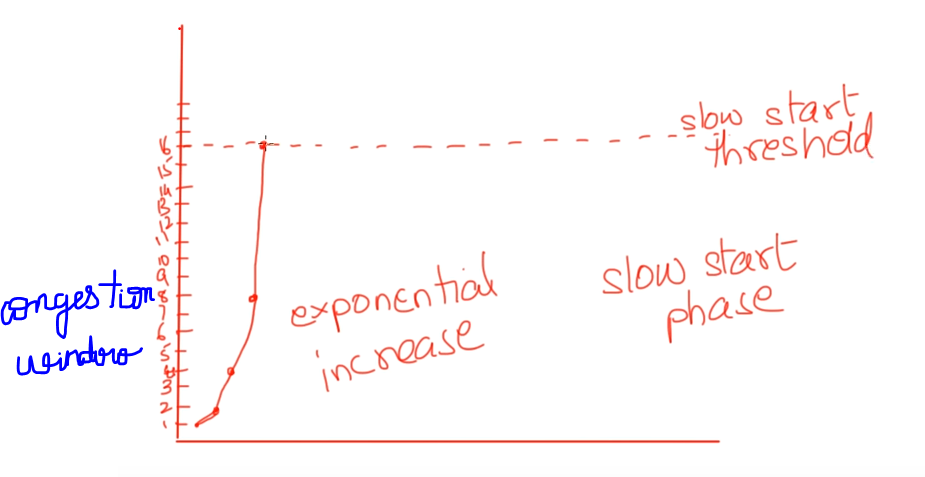
## **Congestion Policies Algorithms**

### **Slow-start : Exponential Increase**

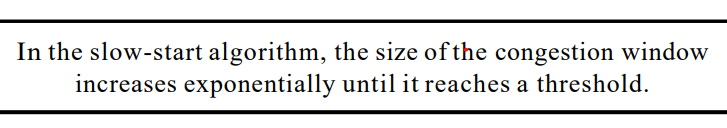
* Exponential Increase One of the algorithms used in TCP congestion control is called slow start. This algorithm is based on the idea that the size of the congestion window starts with one maximum segment size (MSS).
* The MSS is determined during connection establishment by using an option of the same name. The size of the window increases one MSS each time an acknowledgment is received.
* As the name implies, the window starts slowly, but grows exponentially.





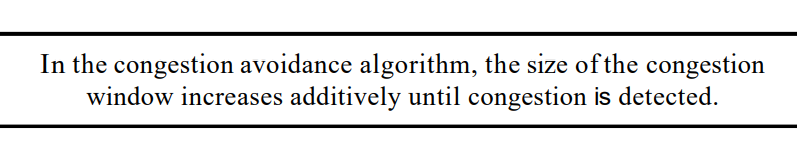
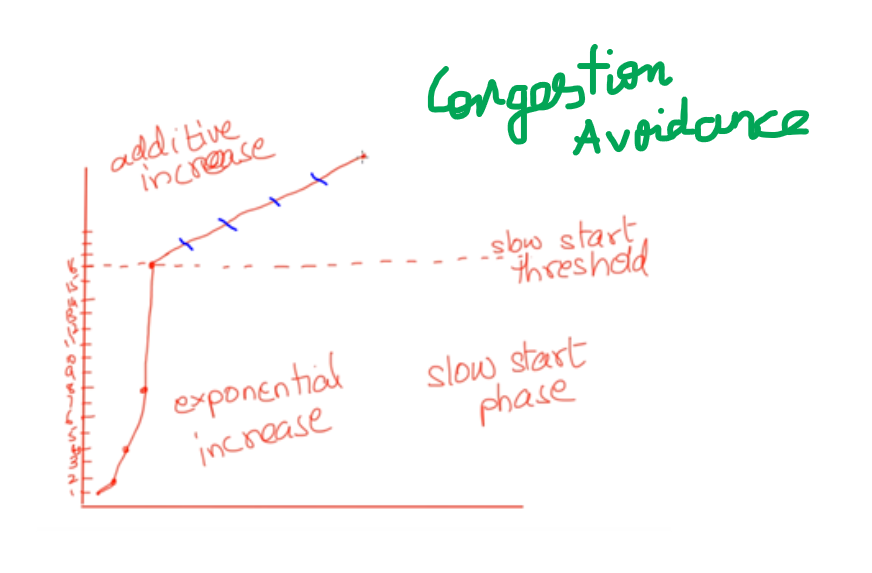
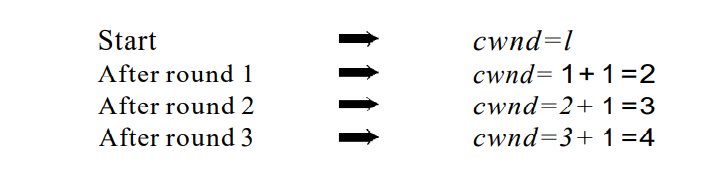
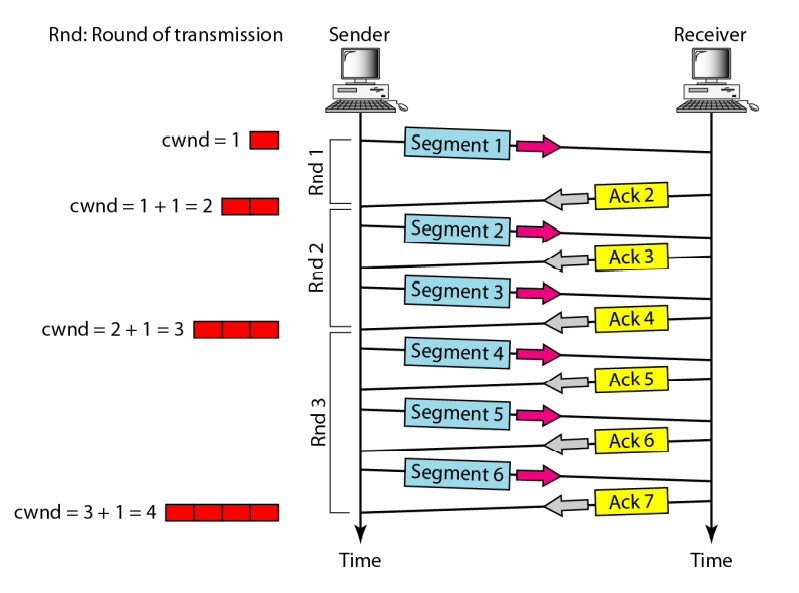


* If we go on increase the window-size more than the slow-start threshold, the congestion takes place.



### **Congestion Avoidance : Additive Increase**

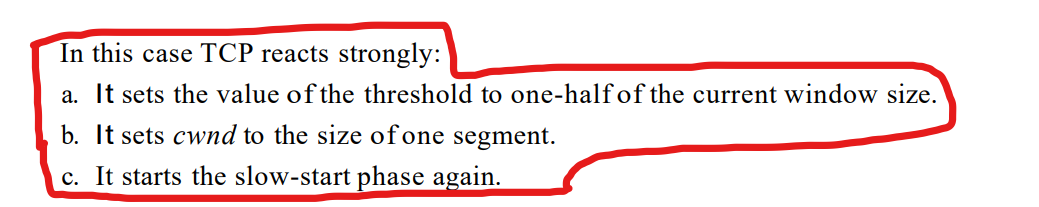
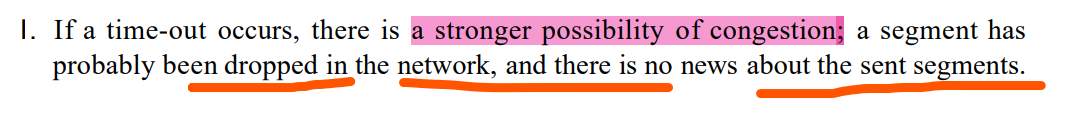
* When the size of the congestion window reaches the slow-start threshold, the slow-start phase stops and the additive phase begins instead of exponential-one.
* The increase of 1 packet goes on until there is a loss of packets (congestion).
* In this algorithm, each time the whole window of segments is acknowledged (one round), the size of congestion window is increased by 1.
* In this case, after the sender has received acknowledgments for a complete window size of segments, the size of the window is increased by one segment.

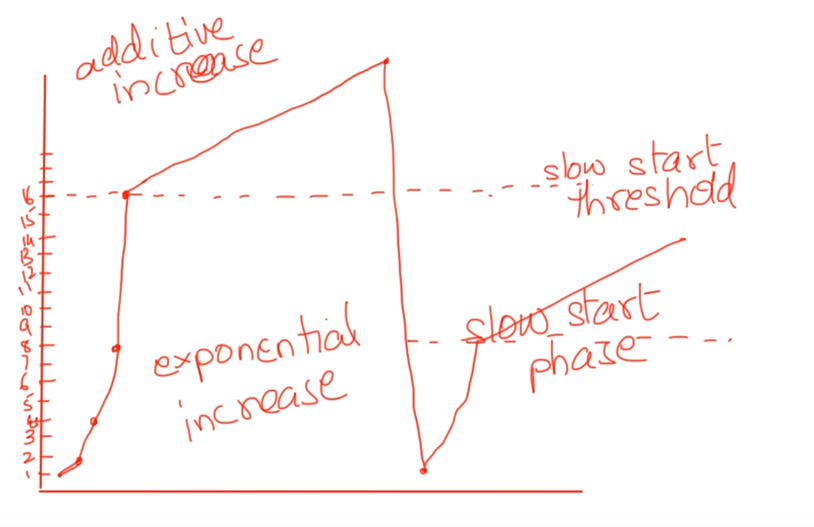


### **Congestion Detection : Multiplicative Decrease**

* If congestion occurs, the congestion window size must be decreased. The only way the sender can guess that congestion has occurred is by the need to retransmit a segment.
* However, retransmission can occur in one of two cases: when a timer times out or when three ACKs are received. In both cases, the size of the threshold is dropped to one-half, a multiplicative decrease.

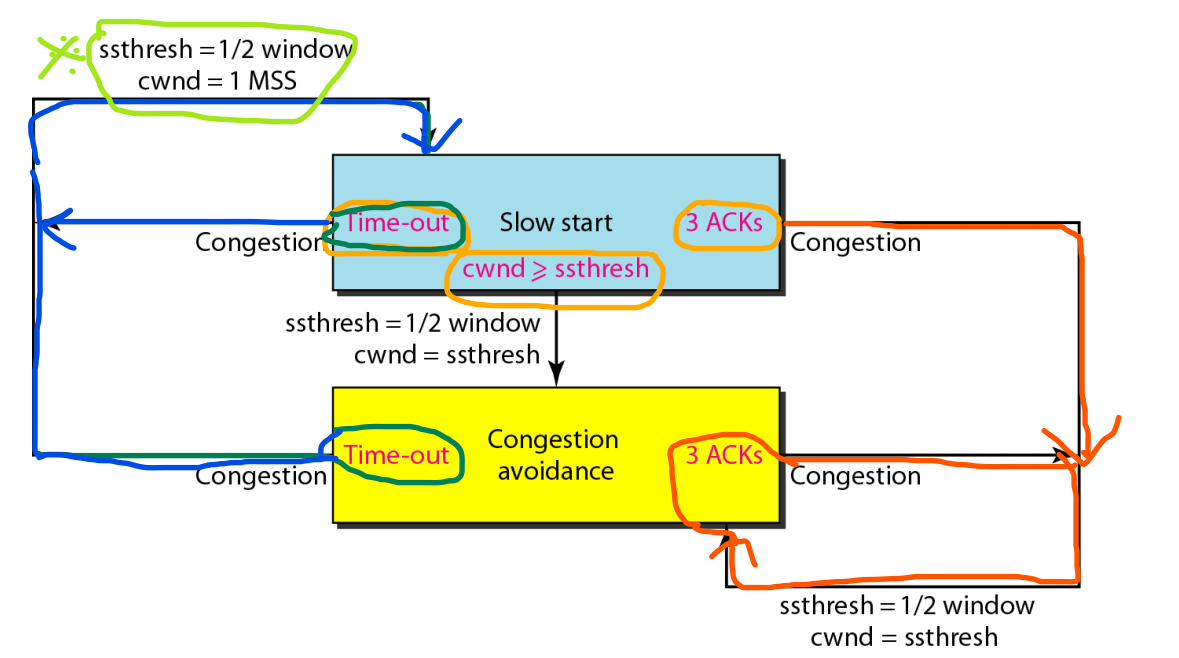
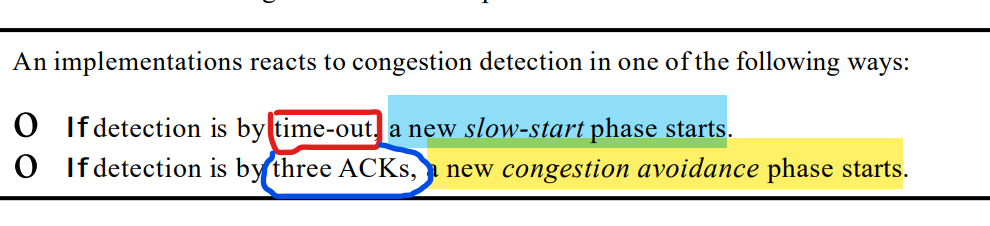
#### **If Time-out occurs:**

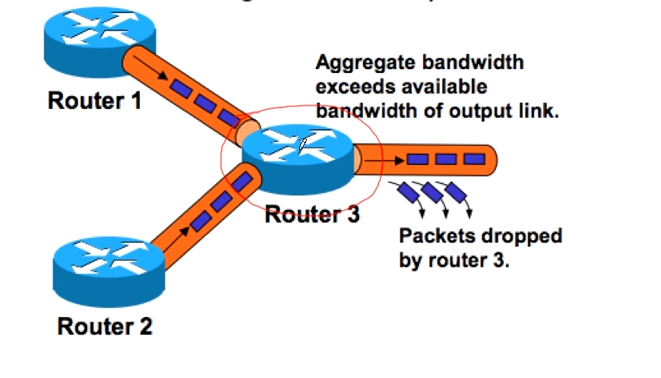




* Here time-out occurs so the window-size is reduced.
* It is brought down to 1, and a new threshold value is set to 8 and then the slow-state phase starts.

#### **If Three-Acknowledgement are received:**

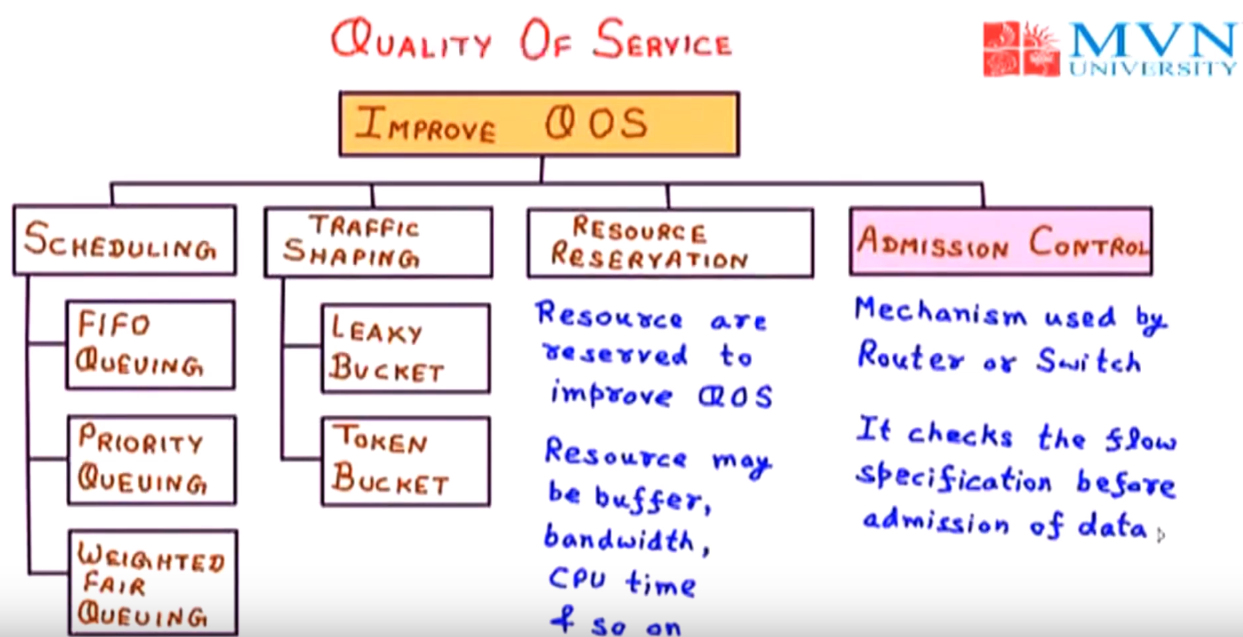




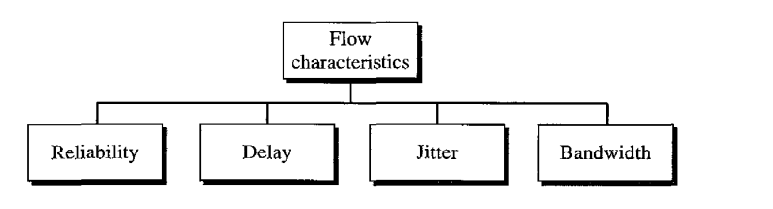
Here the router-3 faces the congestion. In-order to avoid it Router-1 and Router-2 must reduce the speed.

# **Quality of Service**

Textual 🡪 No problem when it is delayed.  
Video 🡪 Both the audio and video must be in a sync.



## **Flow characteristics**

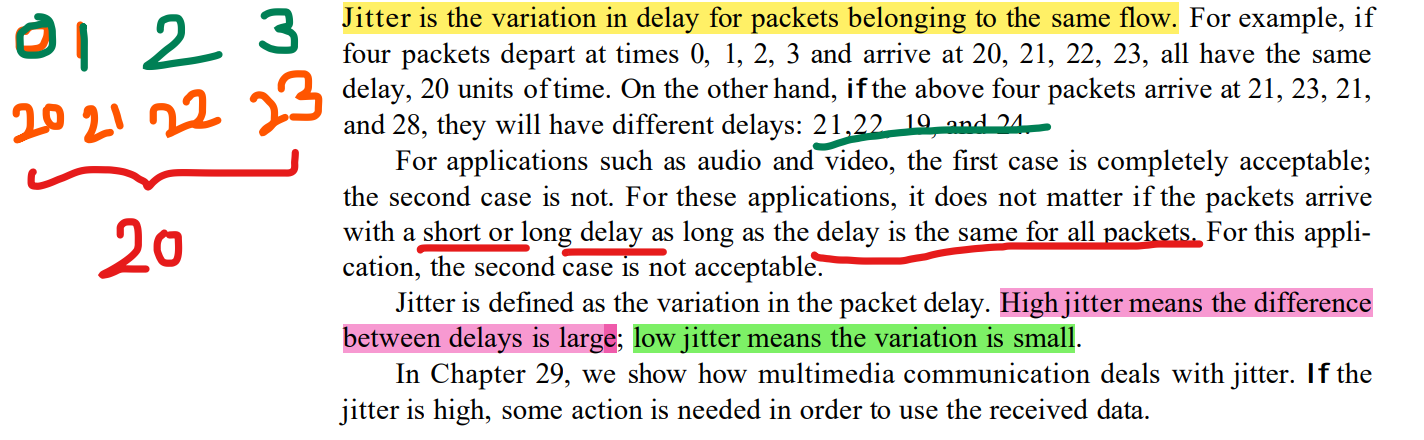


### **Reliability**

Live-streaming video 🡪 struck in-between, no problem we will watch (less reliable)  
Mail, File transfer 🡪 we should require the entire content of the mail without even skipping a line.

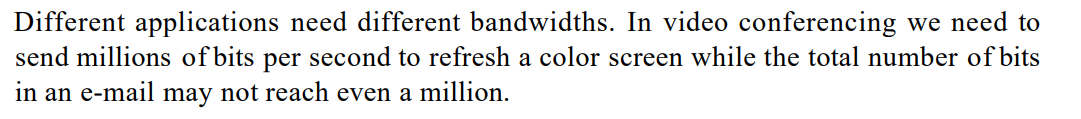
### **Delay**

### **Jitter**



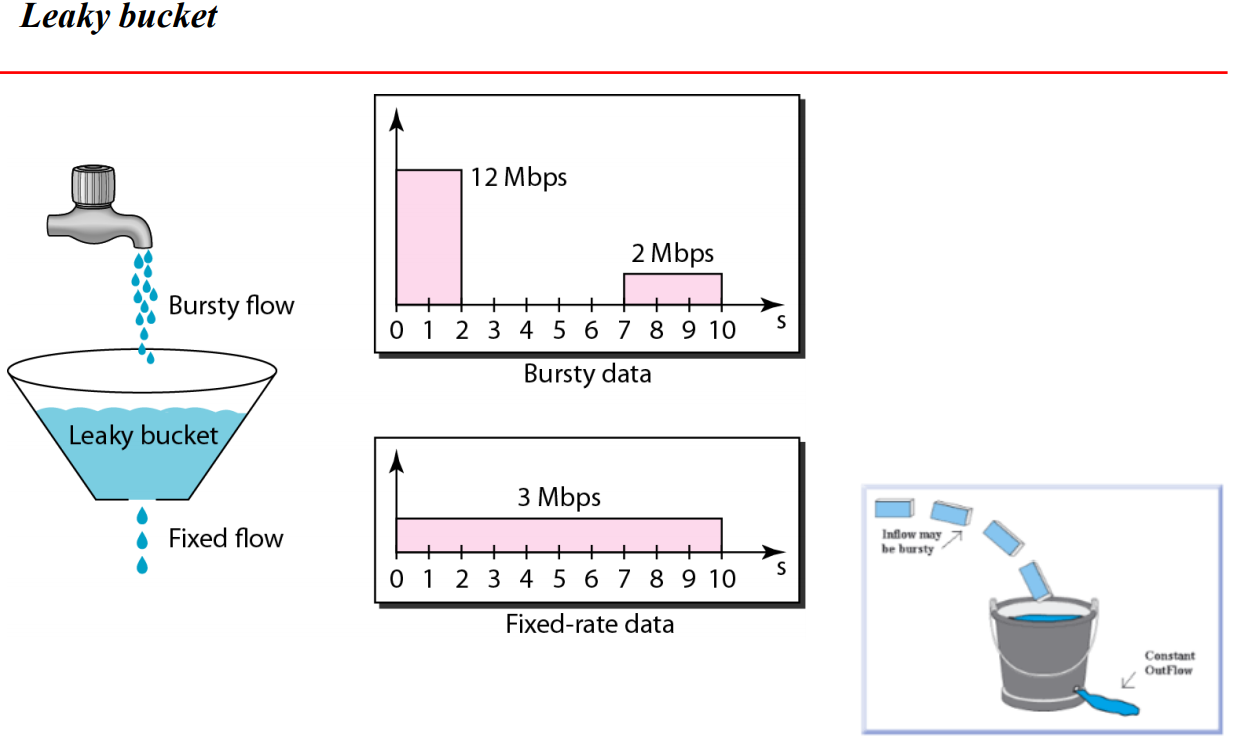
Delay difference smaller 🡪 low jitter.  
Delay difference large 🡪 high jitter.

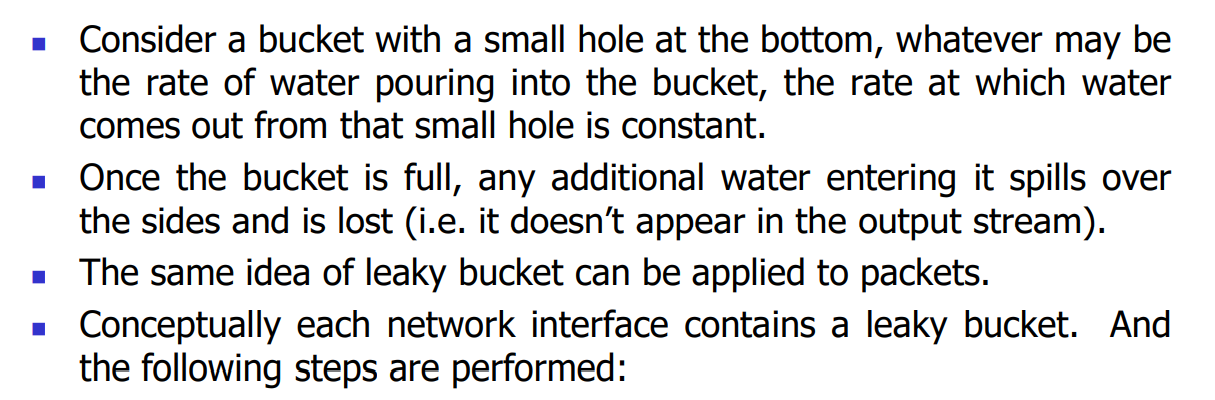
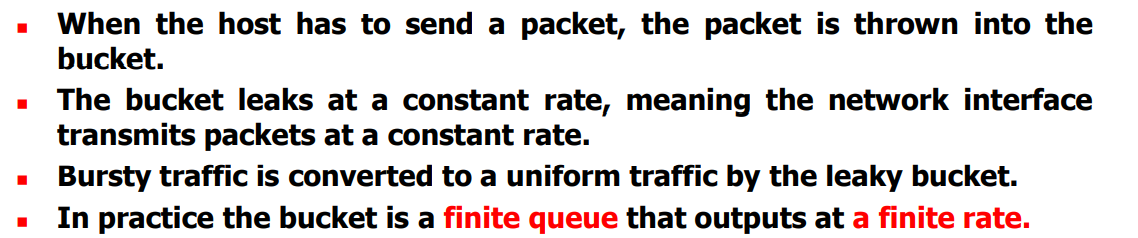
### **Bandwidth**

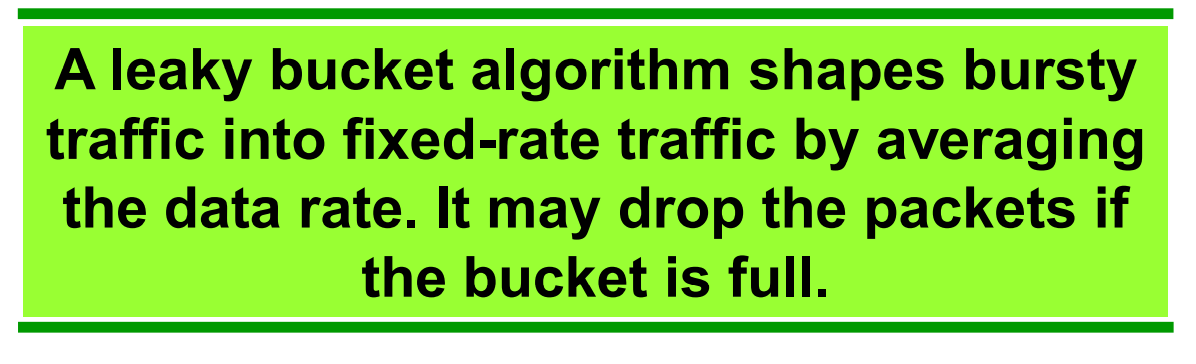
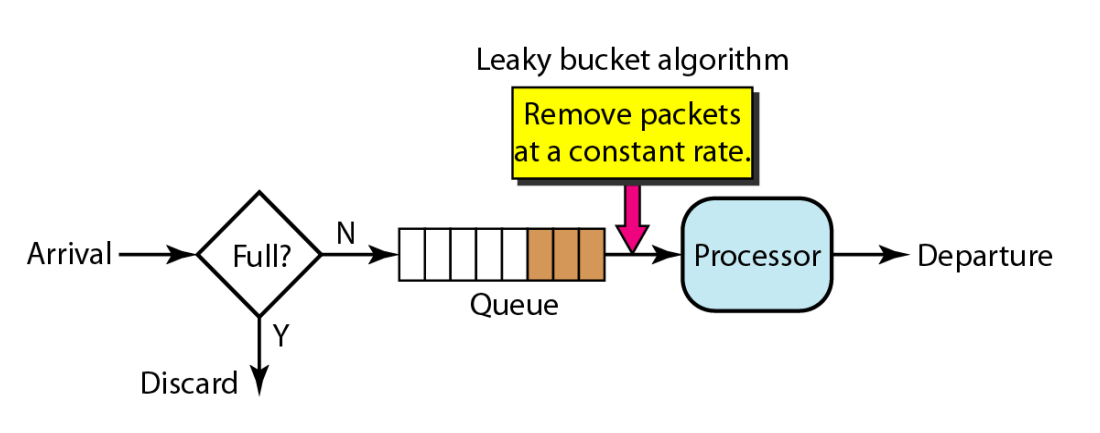
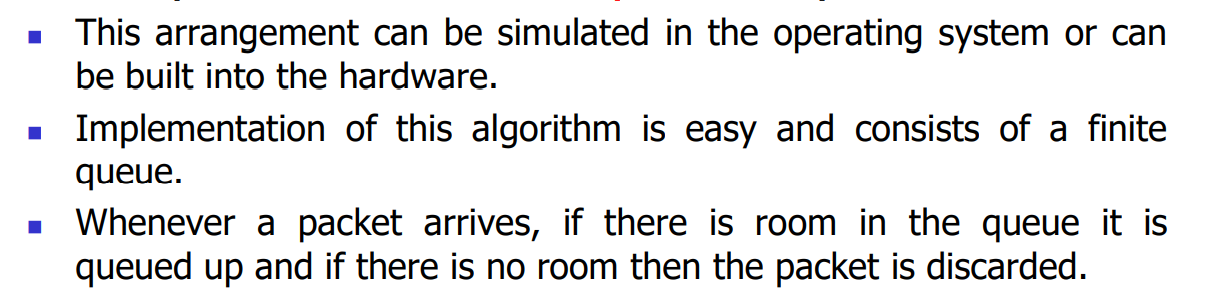


## **Traffic Shaping**

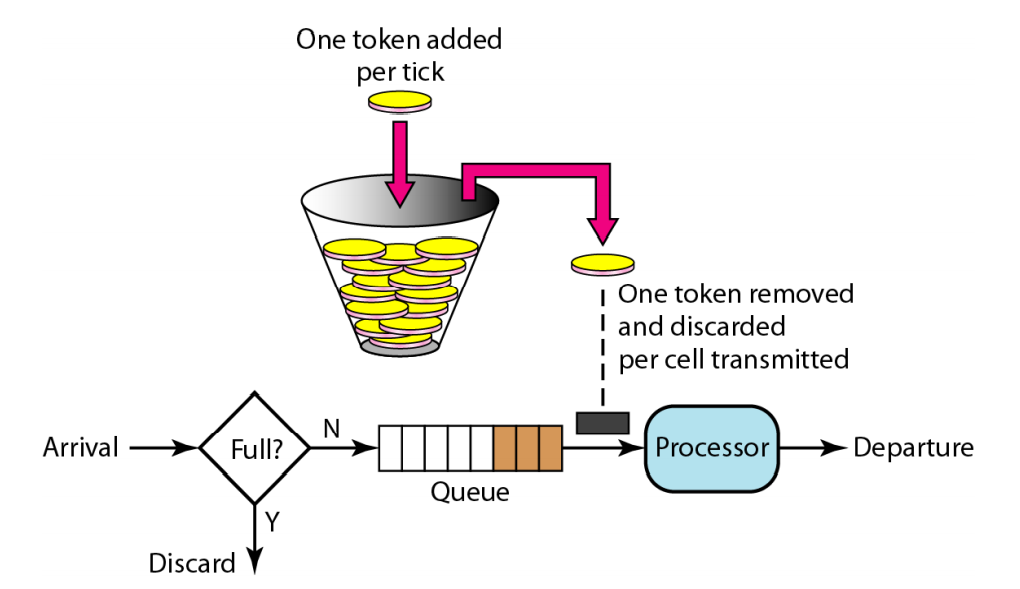
### **Leaky Bucket**

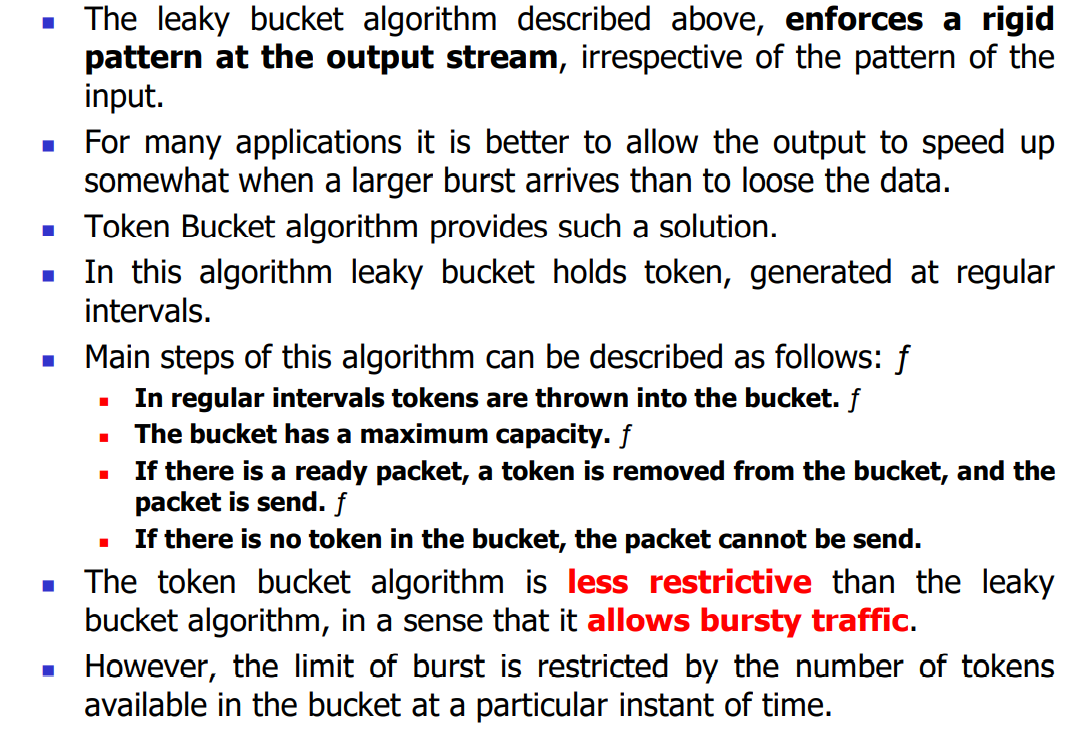


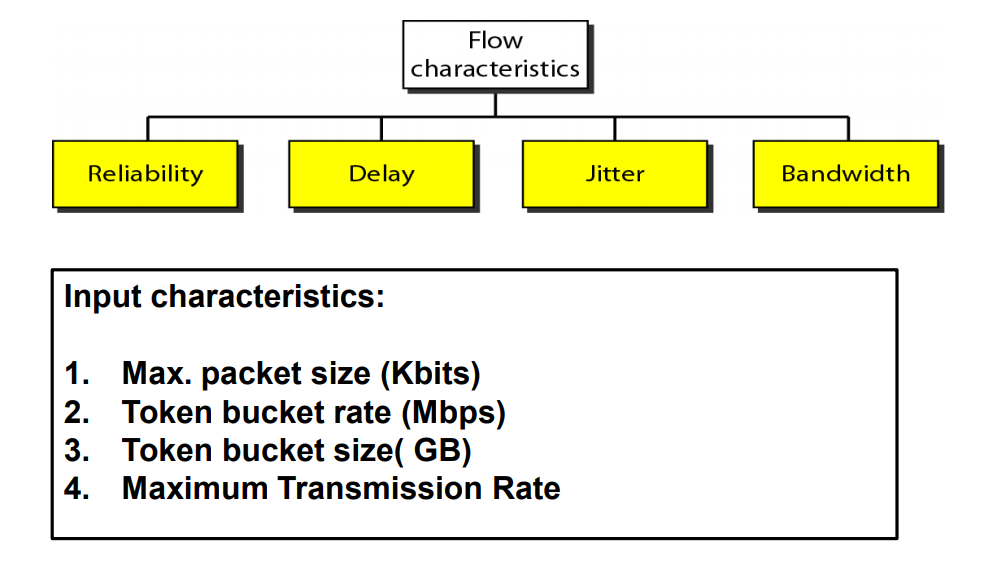
Bursty-data : Taking our whole project backup every month-end.  
Fixed-rate data : Live video streaming.  




### **Token-Bucket**

  
***Large tokens in the bucket***, more number of tokes are sent to the processor.  
***Less tokens in the bucket***, then small number of tokens are sent to the processor.

Leaky-bucket 🡪 Even-though the bucket is full/empty the water comes out a constant rate.  
Token-bucket 🡪 The speed can be regulated according the number of tokens in the bucket.



# **Integrated Services**

