Data Communications and Networking Fourth Edition

Forouzan

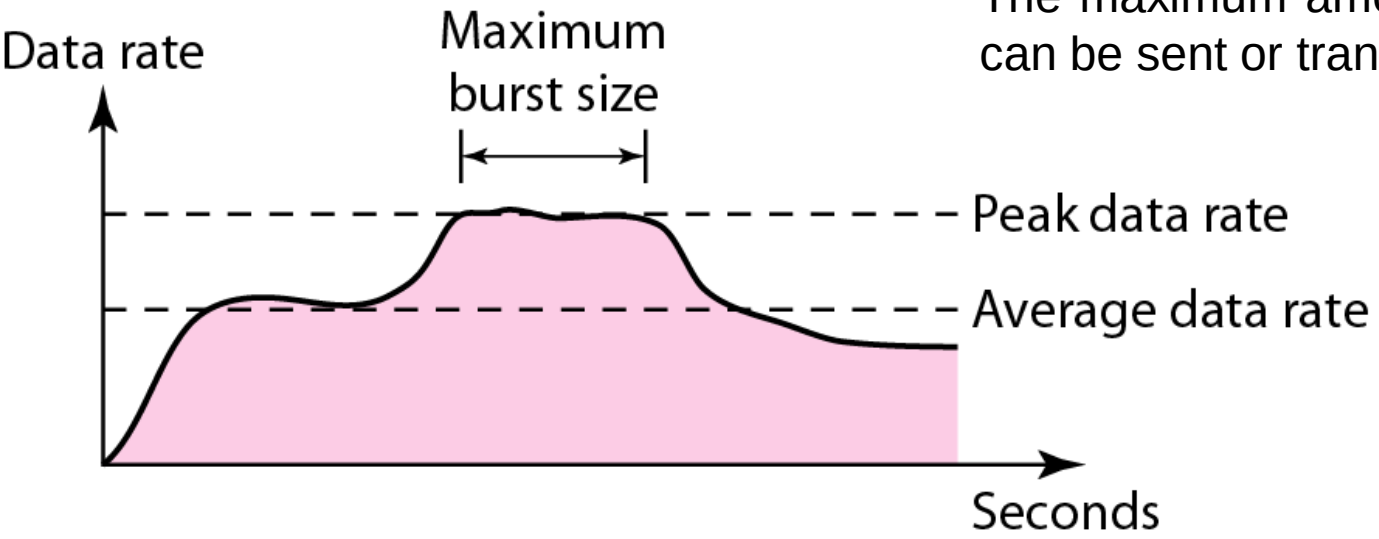
Chapter 24

Congestion Control and Quality of Service

24-1 DATA TRAFFIC

The main focus of congestion control and quality of service is data traffic. In congestion control we try to avoid traffic congestion. In quality of service, we try to create an appropriate environment for the traffic. So, before talking about congestion control and quality of service, we discuss the data traffic itself.

Figure 24.1 *Traffic descriptors*

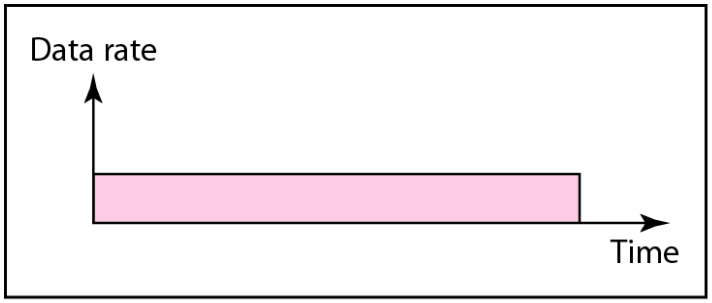


The maximum amount of traffic in bits that can be sent or transmitted in a burst.

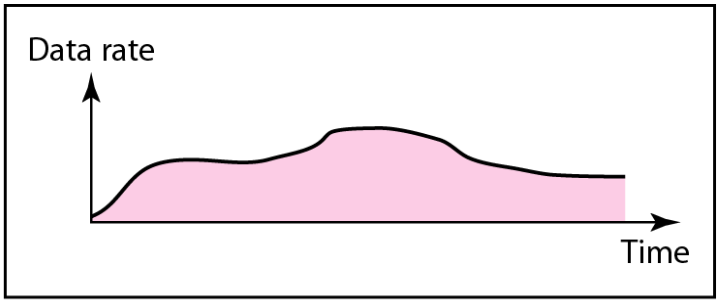
Average data rate refers to the average amount of data transferred per unit of time,

Peak data rate is the fastest **data transfer rate** for a device, typically available in short bursts during **transfer** activity, and not sustainable for long periods of time.

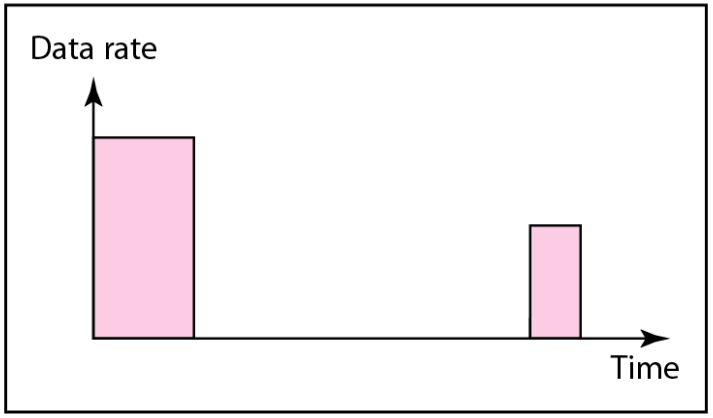
Figure 24.2 *Three traffic profiles*



a. Constant bit rate



b. Variable bit rate



c. Bursty

24-2 CONGESTION

Congestion in a network may occur if the load on the network—the number of packets sent to the network—is greater than the capacity of the network—the number of packets a network can handle. Congestion control refers to the mechanisms and techniques to control the congestion and keep the load below the capacity.

Topics discussed in this section:

Network Performance

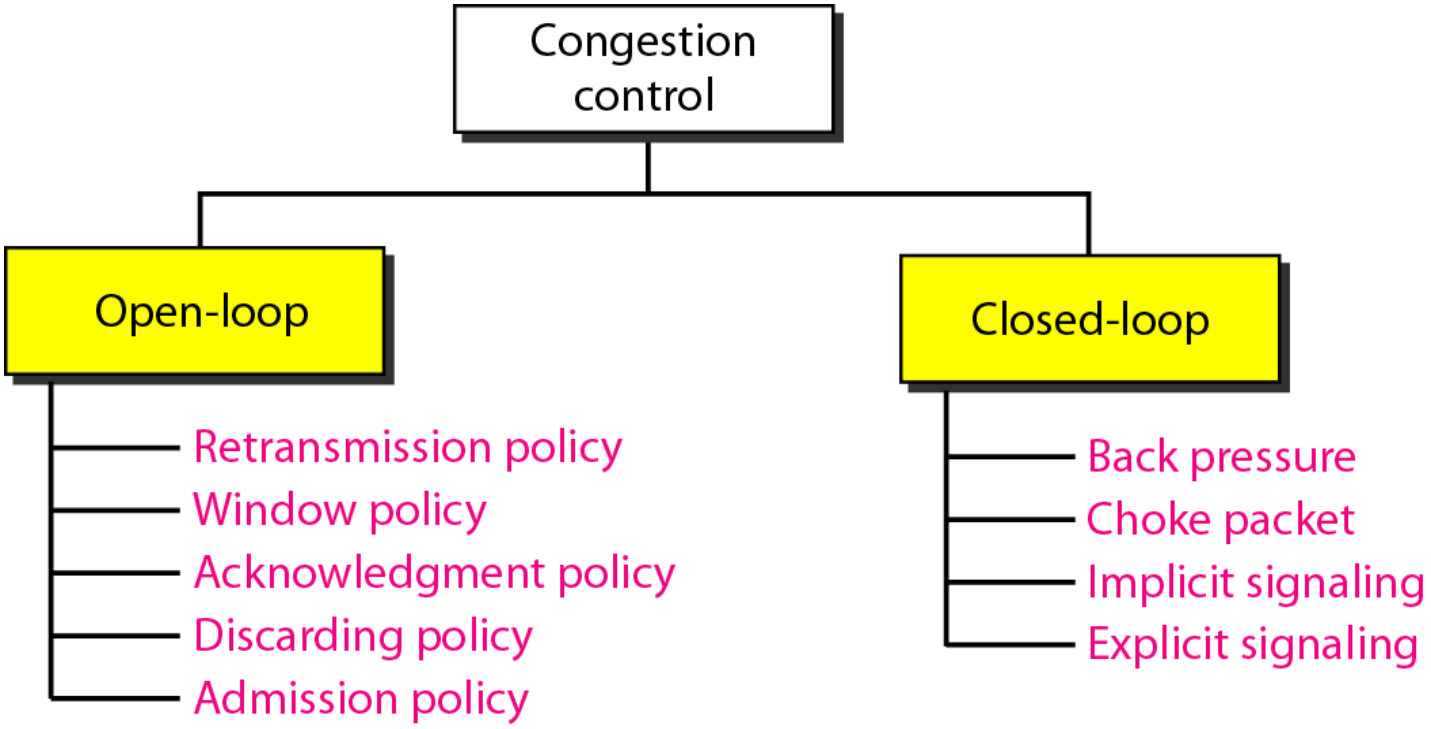
24-3 CONGESTION CONTROL

Congestion control refers to techniques and mechanisms that can either prevent congestion, before it happens, or remove congestion, after it has happened. In general, we can divide congestion control mechanisms into two broad categories: open-loop congestion control (prevention) and closed-loop congestion control (removal).

Topics discussed in this section:

Open-Loop Congestion Control
Closed-Loop Congestion Control

Figure 24.5 *Congestion control categories*



Open Loop Congestion Control

- Open loop congestion control policies are applied to prevent congestion before it happens.
- The congestion control is handled either by the source or the destination.

Retransmission Policy

- It is the policy in which retransmission of the packets are taken care.
- If the sender feels that a sent packet is lost or corrupted, the packet needs to be retransmitted.
- This transmission may increase the congestion in the network.
- To prevent congestion, retransmission timers must be designed to prevent congestion and also able to optimize efficiency.

Window Policy

- The type of window at the sender side may also affect the congestion.
- Several packets in the Go-back-n window are resent, although some packets may be received successfully at the receiver side.
- This duplication may increase the congestion in the network and making it worse.
- Therefore, Selective repeat window should be adopted as it sends the specific packet that may have been lost.

Discarding Policy

- A good discarding policy adopted by the routers is that the routers may prevent congestion and at the same time partially discards the corrupted or less sensitive package and also able to maintain the quality of a message.
- In case of audio file transmission, routers can discard less sensitive packets to prevent congestion and also maintain the quality of the audio file.

Acknowledgment Policy

- Since acknowledgement are also the part of the load in network, the acknowledgment policy imposed by the receiver may also affect congestion.
- Several approaches can be used to prevent congestion related to acknowledgment.
- The receiver should send acknowledgement for N packets rather than sending acknowledgement for a single packet.
- The receiver should send a acknowledgment only if it has to sent a packet or a timer expires.

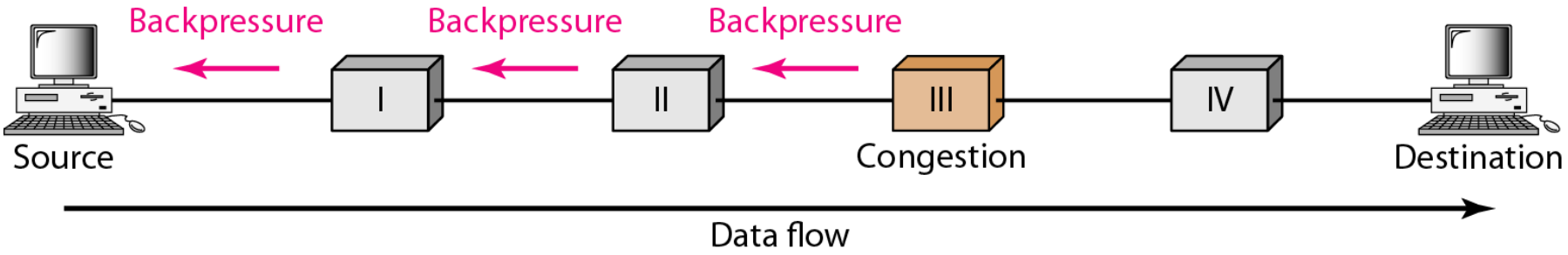
Admission Policy

- In admission policy a mechanism should be used to prevent congestion.
- Switches in a flow should first check the resource requirement of a network flow before transmitting it further.
- If there is a chance of a congestion or there is a congestion in the network, router should deny establishing a virtual network connection to prevent further congestion.

Backpressure

- Backpressure is a technique in which a congested node stop receiving packet from upstream node.
- This may cause the upstream node or nodes to become congested and rejects receiving data from above nodes.
- Backpressure is a node-to-node congestion control technique that propagate in the opposite direction of data flow.
- The backpressure technique can be applied only to **virtual circuit** where each node has information of its above upstream node.

Figure 24.6 ***Backpressure method for alleviating congestion***

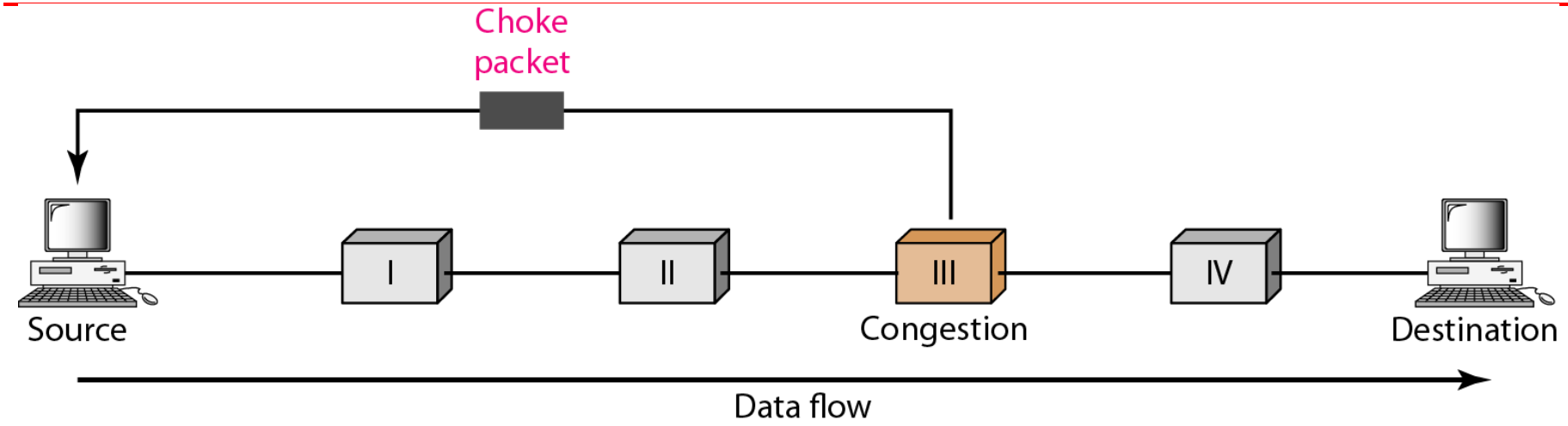


In above diagram the 3rd node is congested and stops receiving packets as a result 2nd node may be get congested due to slowing down of the output data flow. Similarly 1st node may get congested and informs the source to slow down.

Choke Packet

- Choke packet technique is applicable to both virtual networks as well as datagram subnets.
- A choke packet is a packet sent by a node to the source to inform it of congestion.
- Each router monitor its resources and the utilization at each of its output lines.
- whenever the resource utilization exceeds the threshold value which is set by the administrator, the router directly sends a choke packet to the source giving it a feedback to reduce the traffic.
- The intermediate nodes through which the packets has traveled are not warned about congestion.

Figure 24.7 *Choke packet*



Implicit Signaling

- In implicit signaling, there is no communication between the congested nodes and the source.
- The source guesses that there is congestion in a network.
- For example when sender sends several packets and there is no acknowledgment for a while, one assumption is that there is a congestion.

Explicit Signaling

- In explicit signaling, if a node experiences congestion it can explicitly sends a packet to the source or destination to inform about congestion.
- The difference between choke packet and explicit signaling is that the signal is included in the packets that carry data rather than creating different packet as in case of choke packet technique.
- Explicit signaling can occur in either forward or backward direction.

- **Forward Signaling :** In forward signaling, signal is sent in the direction of the congestion. The destination is warned about congestion. The receiver in this case adopt policies to prevent further congestion.
- **Backward Signaling :** In backward signaling signal is sent in the opposite direction of the congestion. The source is warned about congestion and it needs to slow down.

Figure 24.13 *FECN*

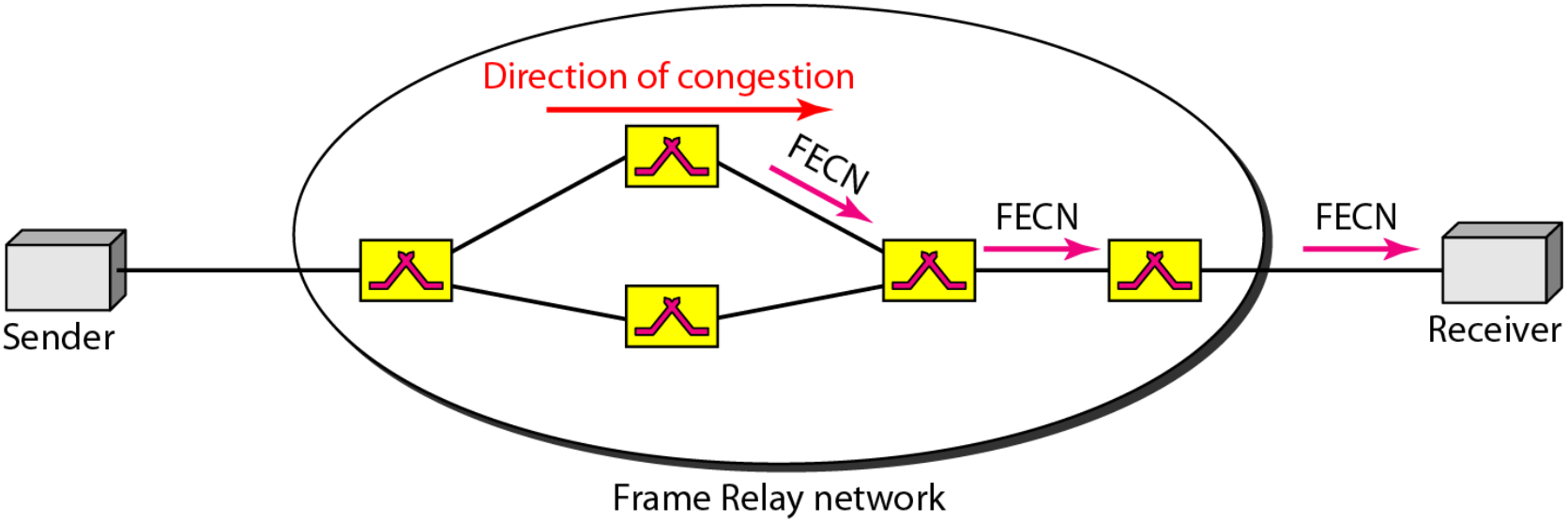


Figure 24.12 *BECN*

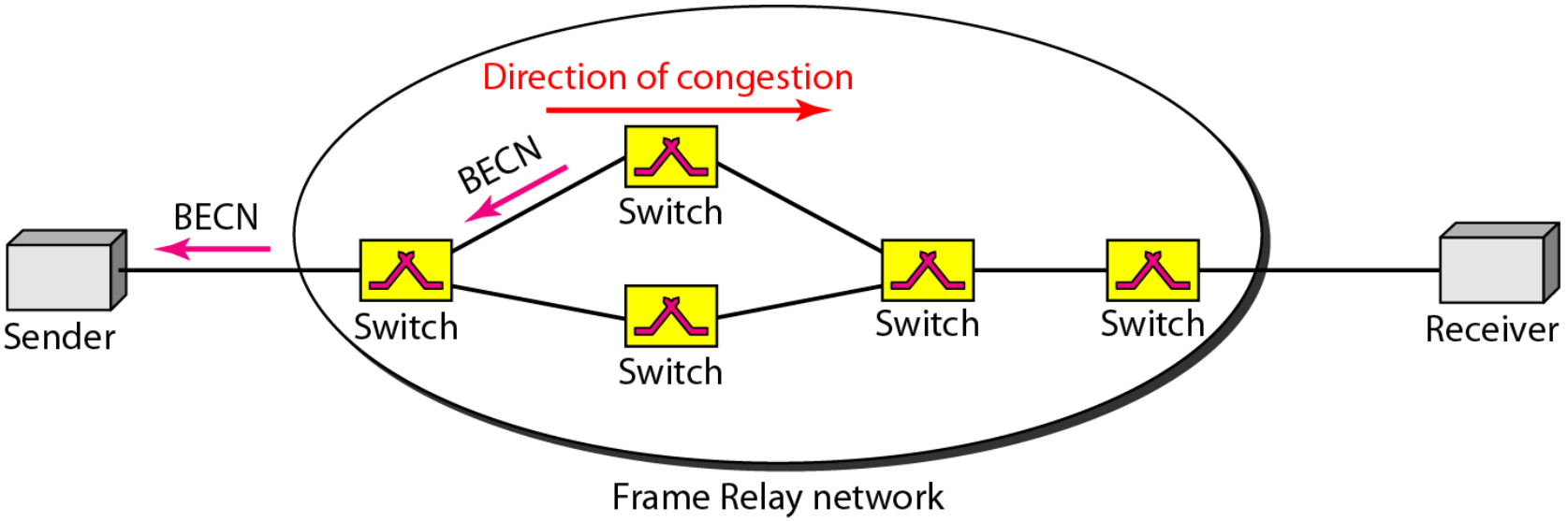
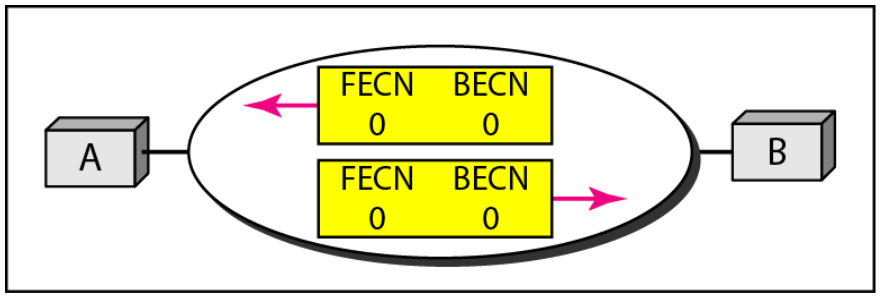
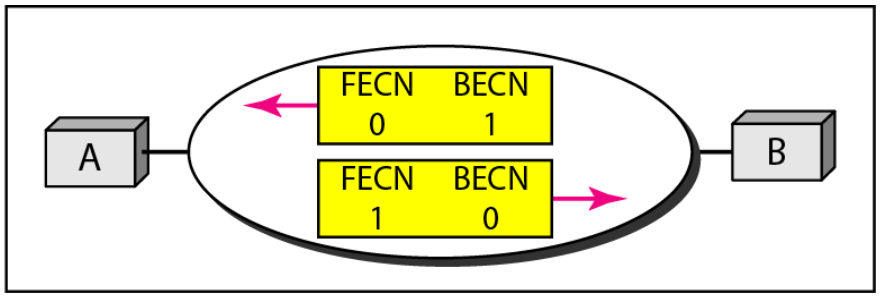


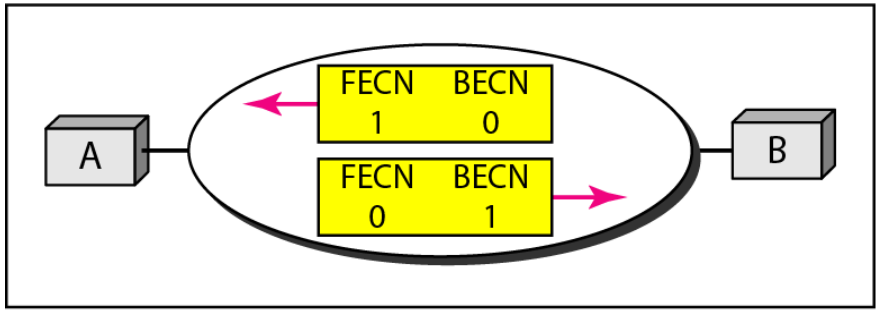
Figure 24.14 *Four cases of congestion*



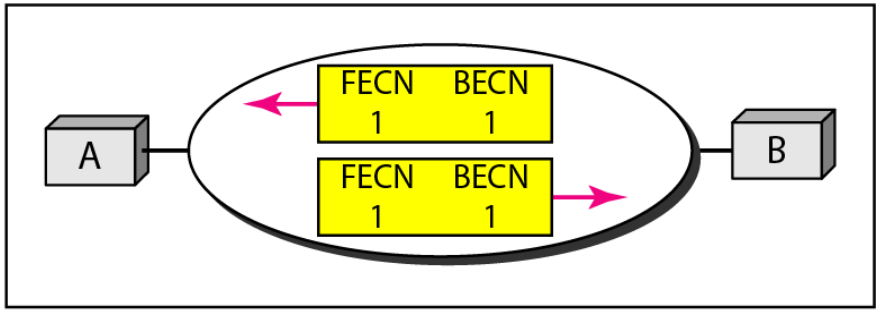
a. No congestion



b. Congestion in the direction A–B



c. Congestion in the direction B–A



d. Congestion in both directions

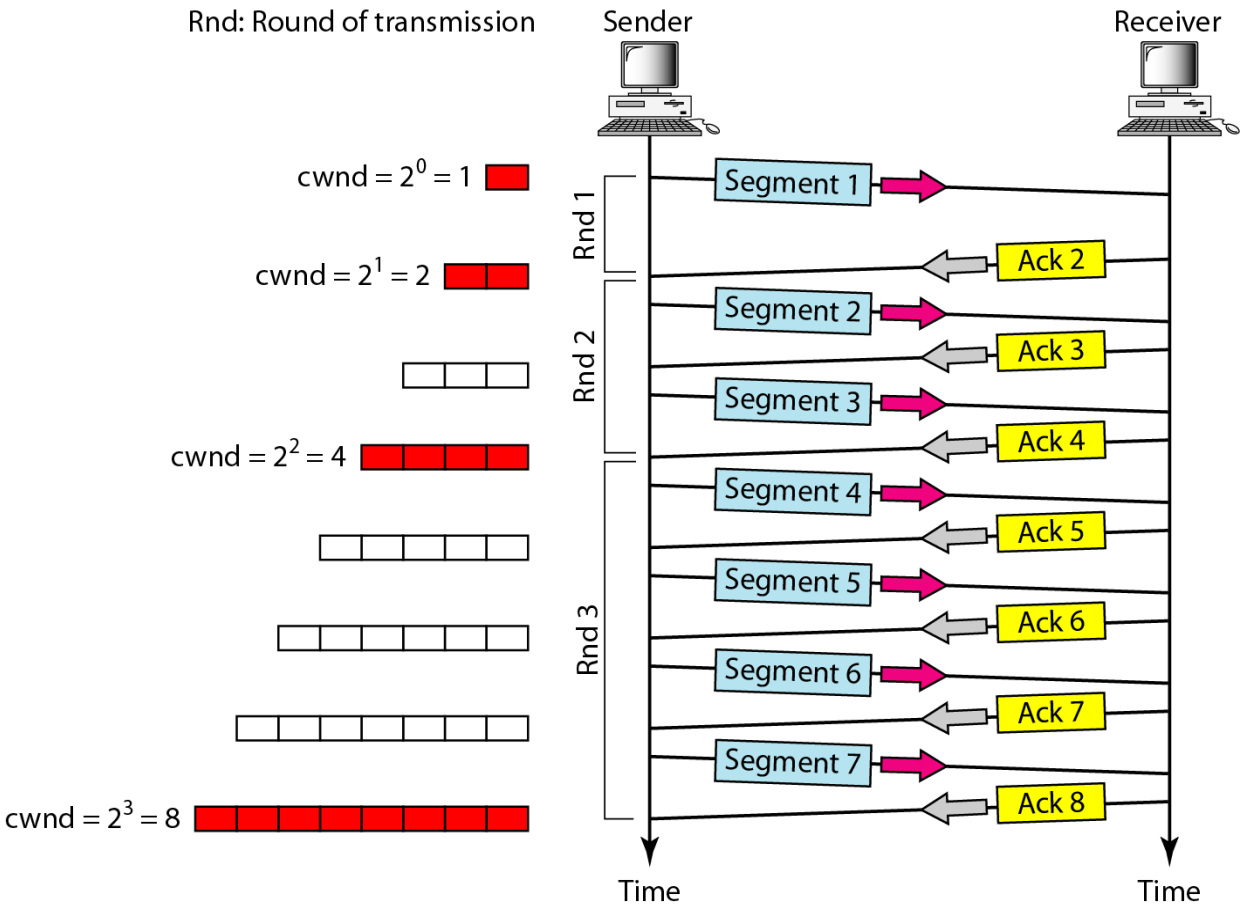
24-4 TWO EXAMPLES

To better understand the concept of congestion control, let us give two examples: one in TCP and the other in Frame Relay.

Topics discussed in this section:

Congestion Control in TCP
Congestion Control in Frame Relay

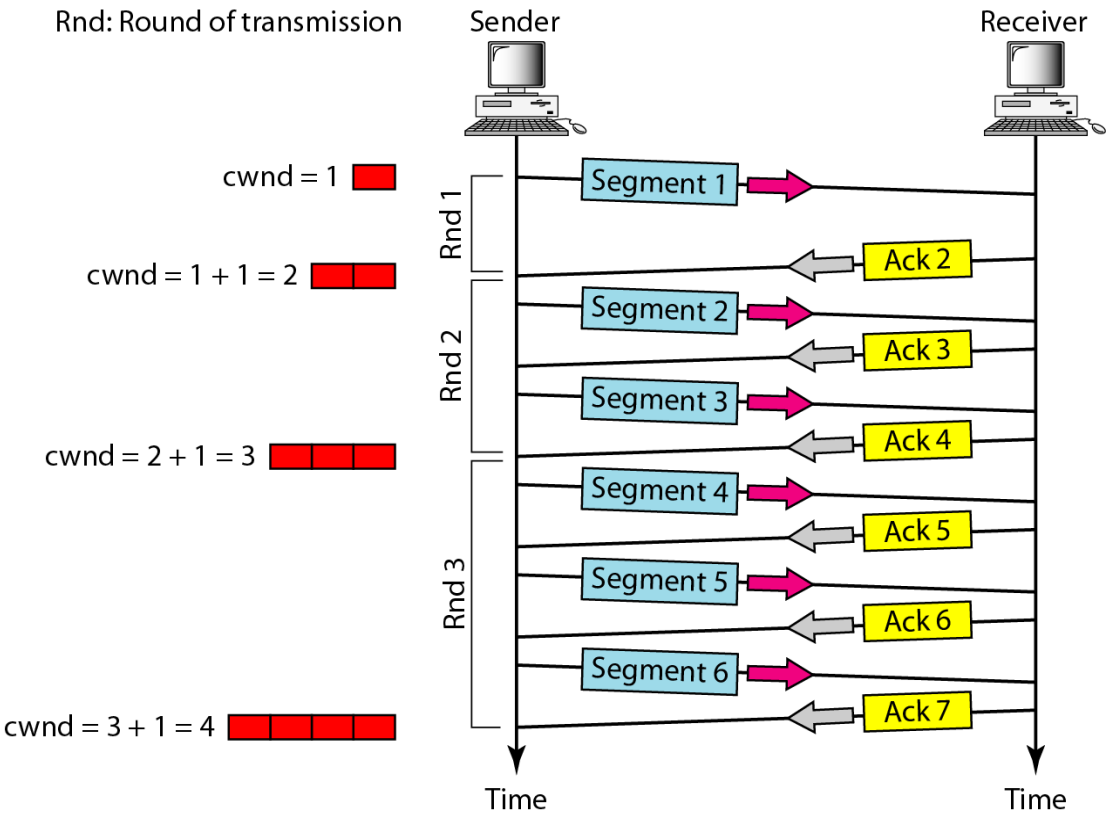
Figure 24.8 *Slow start, exponential increase*



Note

In the slow-start algorithm, the size of the congestion window increases exponentially until it reaches a threshold.

Figure 24.9 *Congestion avoidance, additive increase*



Note

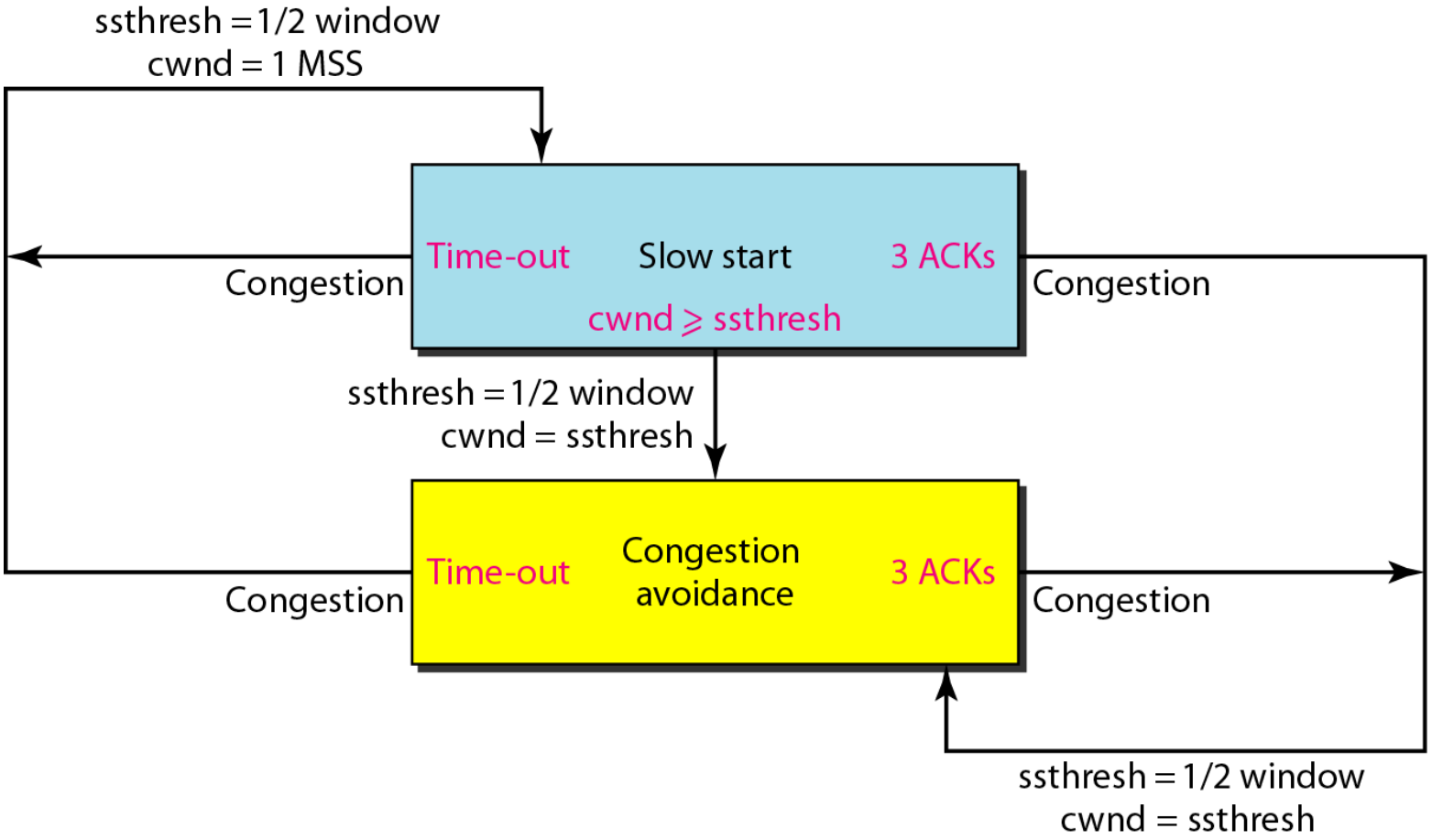
In the congestion avoidance algorithm, the size of the congestion window increases additively until congestion is detected.

Note

An implementation reacts to congestion detection in one of the following ways:

- **If detection is by time-out, a new slow start phase starts.**
- **If detection is by three ACKs, a new congestion avoidance phase starts.**

Figure 24.10 *TCP congestion policy summary*



24-5 QUALITY OF SERVICE

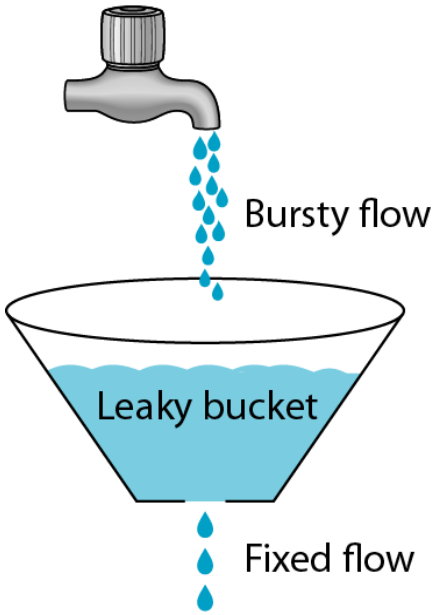
Quality of Service (QoS) is an internetworking issue. We can informally define quality of service as something a flow seeks to attain.

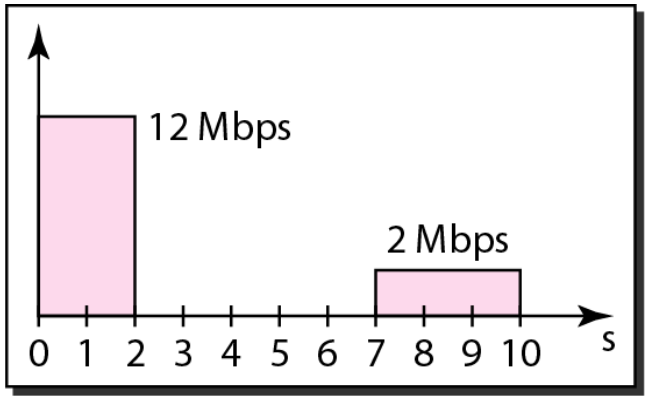
Techniques to improve QoS

There are four common methods:

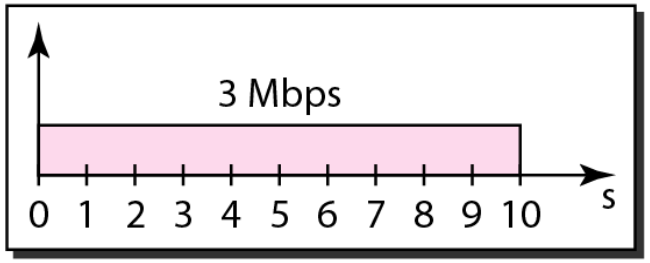
1. *Scheduling*
2. *Traffic shaping*
3. *Admission control and*
4. *Resource reservation.*

Leaky bucket

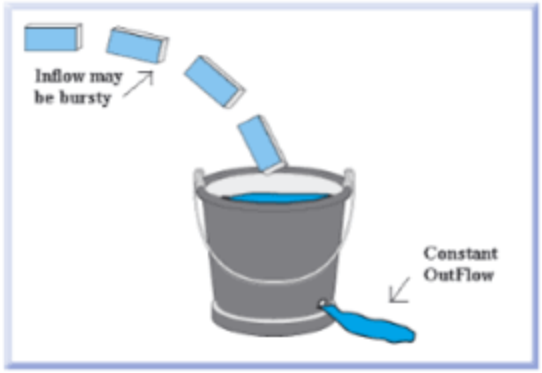




Bursty data

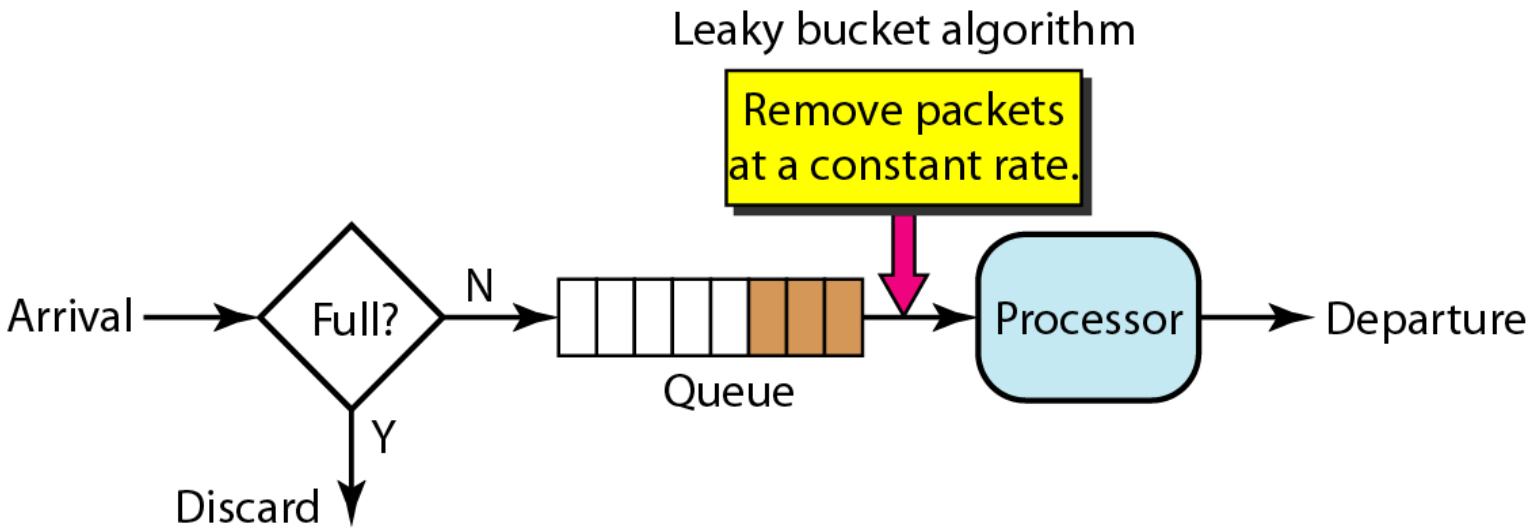


Fixed-rate data



- Consider a bucket with a small hole at the bottom, whatever may be the rate of water pouring into the bucket, the rate at which water comes out from that small hole is constant.
- Once the bucket is full, any additional water entering it spills over the sides and is lost (i.e. it doesn't appear in the output stream).
- The same idea of leaky bucket can be applied to packets.
- Conceptually each network interface contains a leaky bucket. And the following steps are performed:
  - **When the host has to send a packet, the packet is thrown into the bucket.**
  - **The bucket leaks at a constant rate, meaning the network interface transmits packets at a constant rate.**
  - **Bursty traffic is converted to a uniform traffic by the leaky bucket.**
  - **In practice the bucket is a finite queue that outputs at a finite rate.**
- This arrangement can be simulated in the operating system or can be built into the hardware.
- Implementation of this algorithm is easy and consists of a finite queue.
- Whenever a packet arrives, if there is room in the queue it is queued up and if there is no room then the packet is discarded.

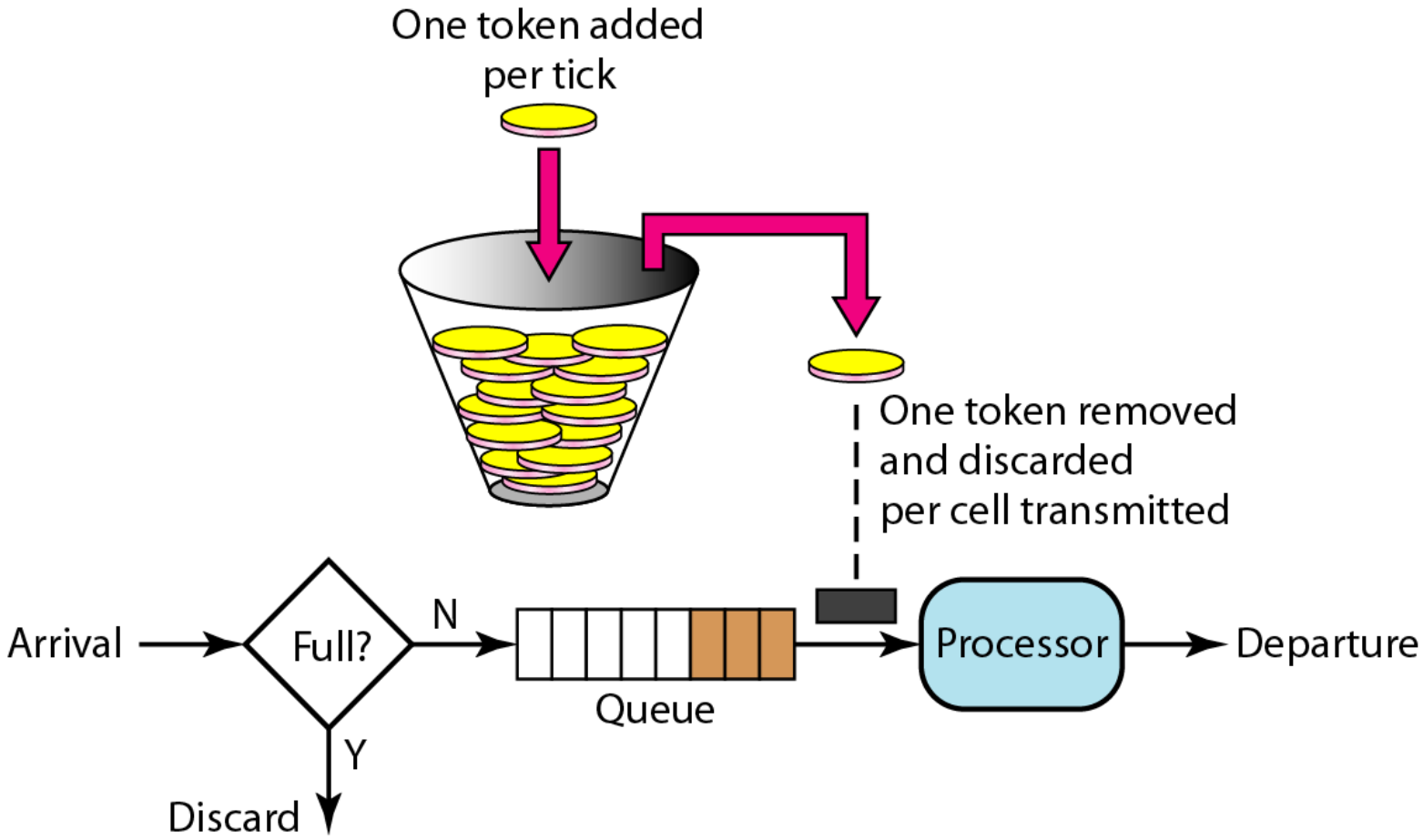
Leaky bucket implementation



Note

A leaky bucket algorithm shapes bursty traffic into fixed-rate traffic by averaging the data rate. It may drop the packets if the bucket is full.

Token bucket

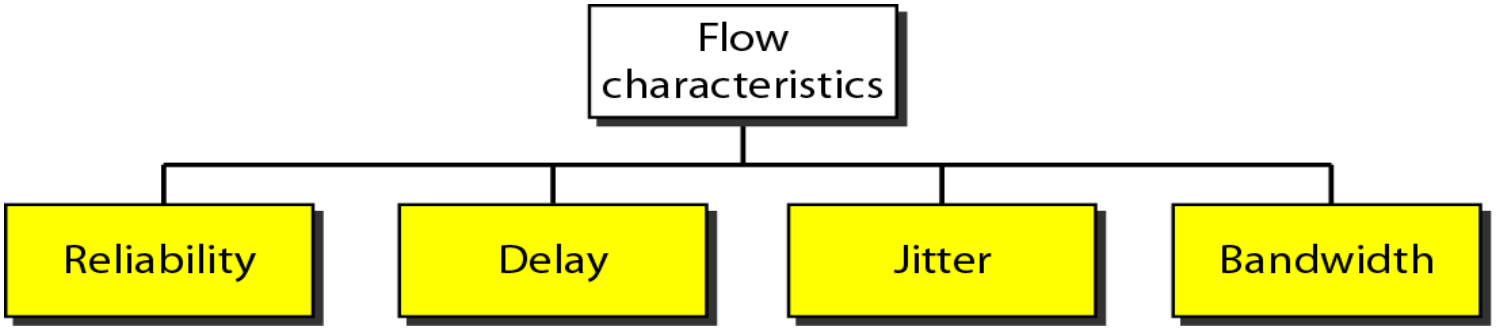


- The leaky bucket algorithm described above, **enforces a rigid pattern at the output stream**, irrespective of the pattern of the input.
- For many applications it is better to allow the output to speed up somewhat when a larger burst arrives than to loose the data.
- Token Bucket algorithm provides such a solution.
- In this algorithm leaky bucket holds token, generated at regular intervals.
- Main steps of this algorithm can be described as follows: ☐
  - **In regular intervals tokens are thrown into the bucket.** ☐
  - **The bucket has a maximum capacity.** ☐
  - **If there is a ready packet, a token is removed from the bucket, and the packet is send.** ☐
  - **If there is no token in the bucket, the packet cannot be send.**
- The token bucket algorithm is **less restrictive** than the leaky bucket algorithm, in a sense that it **allows bursty traffic**.
- However, the limit of burst is restricted by the number of tokens available in the bucket at a particular instant of time.

Note

The token bucket allows bursty traffic at a regulated maximum rate.

Flow characteristics



Input characteristics:

1. **Max. packet size (Kbits)**
2. **Token bucket rate (Mbps)**
3. **Token bucket size(GB)**
4. **Maximum Transmission Rate**