Data Communications and Networking Fourth Edition

Forouzan

# Transport Layer

Delivered By:
Avinash Bhagat
9463281930
avinash.bhagat@lpu.co.in

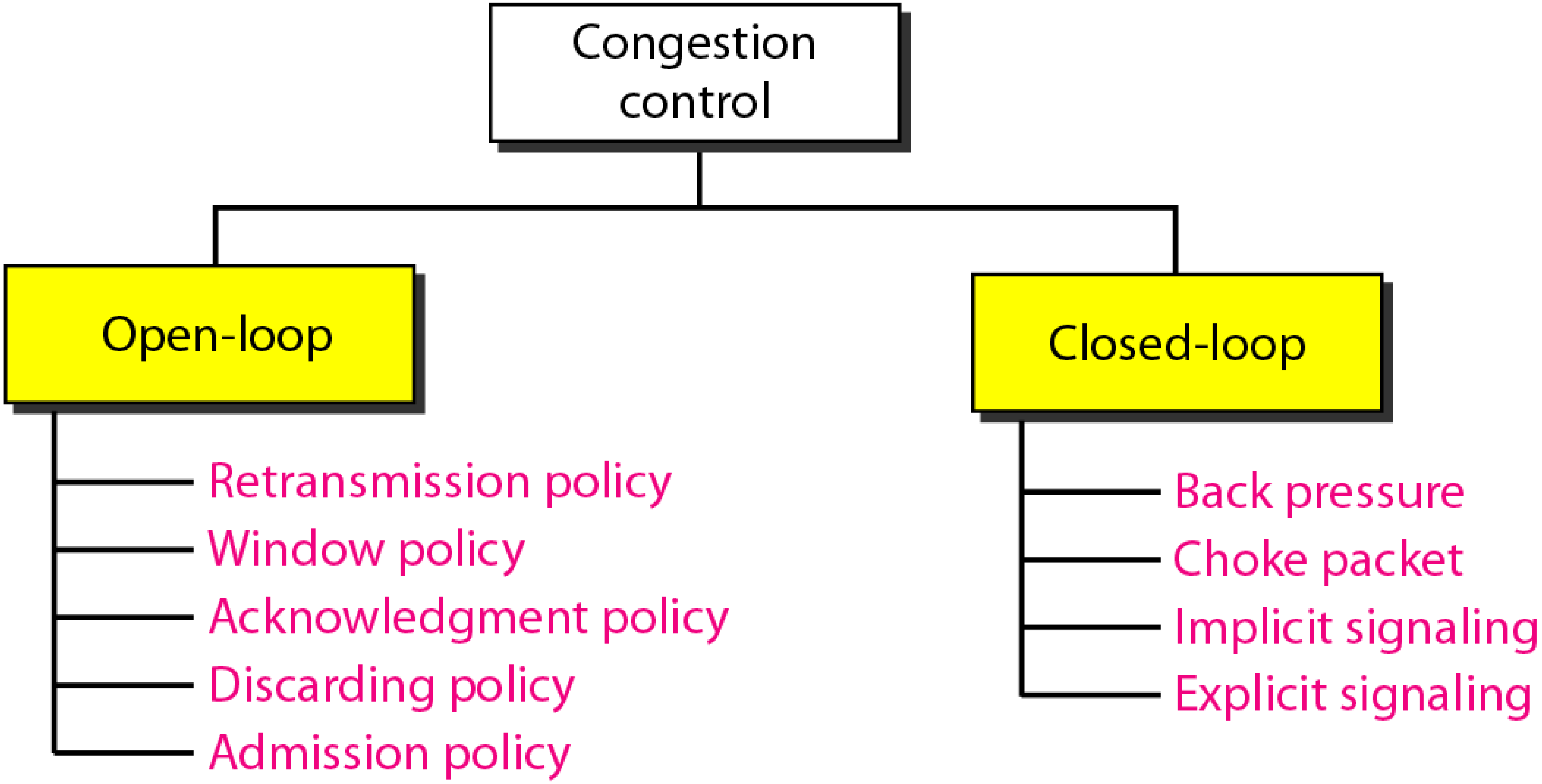
Copyright © The McGraw-Hill Companies, Inc. Permission required for reproduction or display.

1. Transport layer protocols Chapter 23
2. Congestion control and QoS Chapter 24

# Congestion Control

- An important issue in a packet-switched network is **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.

# Congestion Control Categories



- Congestion control
  - Open-loop
    - Retransmission policy
    - Window policy
    - Acknowledgment policy
    - Discarding policy
    - Admission policy
  - Closed-loop
    - Back pressure
    - Choke packet
    - Implicit signaling
    - Explicit signaling

# Open Loop Congestion Control

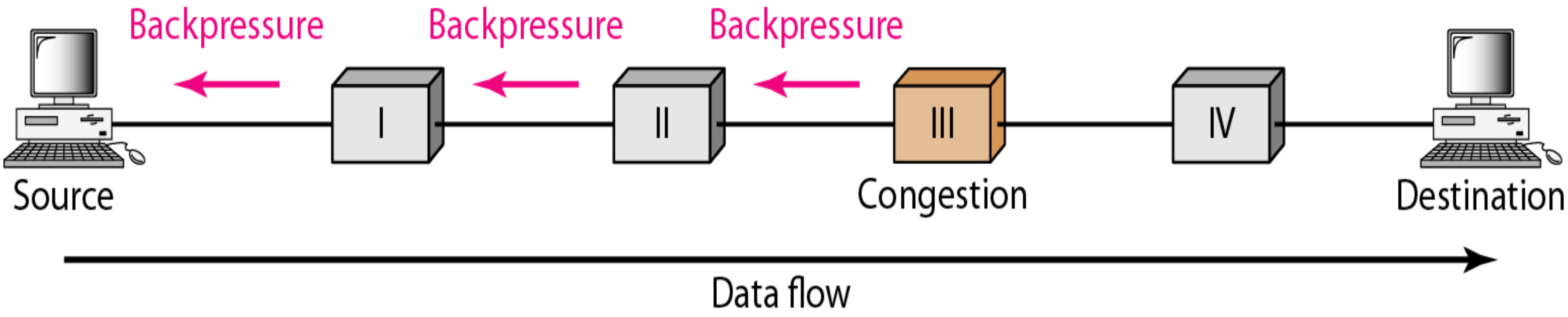
- **Retransmission Policy:**
  - If the sender feels that a sent packet is lost or corrupted, the packet needs to be retransmitted.
  - The retransmission policy and the retransmission timers must be designed to optimize efficiency and at the same time prevent congestion.
- **Window Policy:**
  - The type of window at the sender may also affect congestion.
  - The Selective Repeat window is better than the Go-Back-N window for congestion control.
  - In the *Go-Back-N* window, when the timer for a packet times out, several packets may be resent, although some may have arrived safe and sound at the receiver.

- **Acknowledgement Policy:**
  - If the receiver does not acknowledge every packet it receives, it helps prevent congestion.
- **Discarding Policy:**
  - In this policy less sensitive packets may be discarded when congestion is likely to happen
- **Admission Policy:**
  - A router can deny establishing a virtual circuit connection if there is congestion in the network or if there is a possibility of future congestion.

# Closed Loop Congestion Control

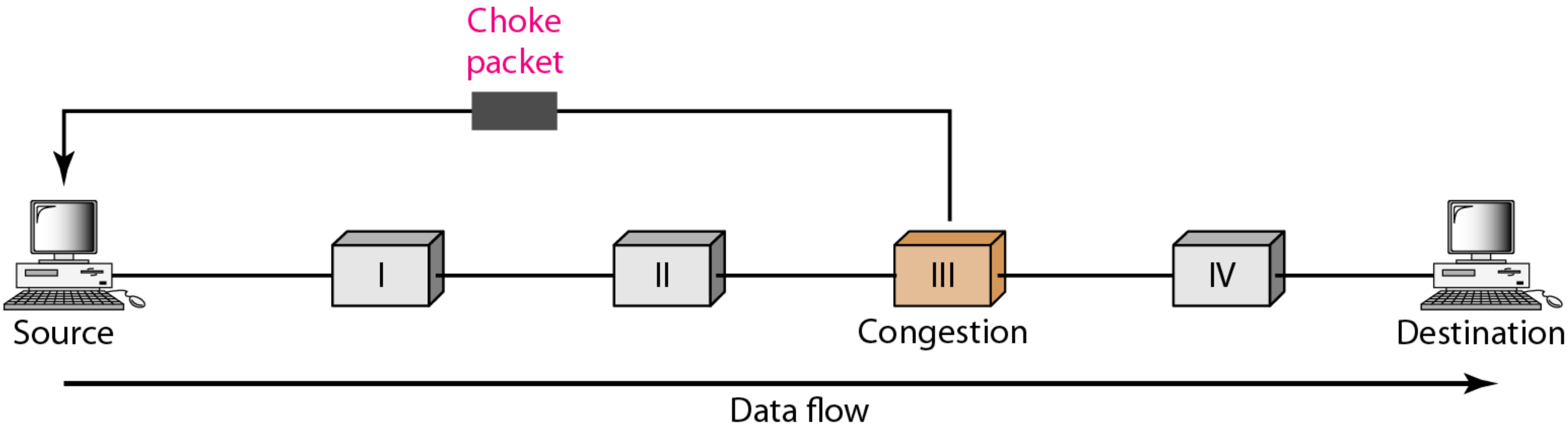
## Backpressure

- It is a node-to-node congestion control that starts with a node and propagates, in the opposite direction of data flow, to the source.
- The backpressure technique can be applied only to virtual circuit networks, in which each node knows the upstream node from which a flow of data is coming.



## Choke Packet:

- A choke packet is a packet sent by a node to the source to inform it about congestion.
- The warning is from the router, which has encountered congestion, to the source station directly.



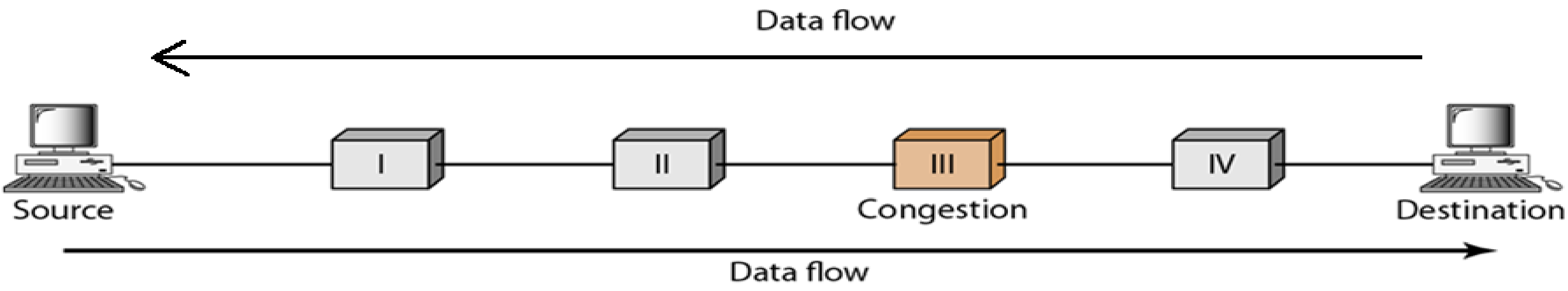
## Implicit Signalling:

- there is no communication between the congested node or nodes and the source.
- The source guesses that there is a congestion somewhere in the network from other symptoms.

# Explicit Signalling:

- The node that experiences congestion can explicitly send a signal to the source or destination.
- 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.

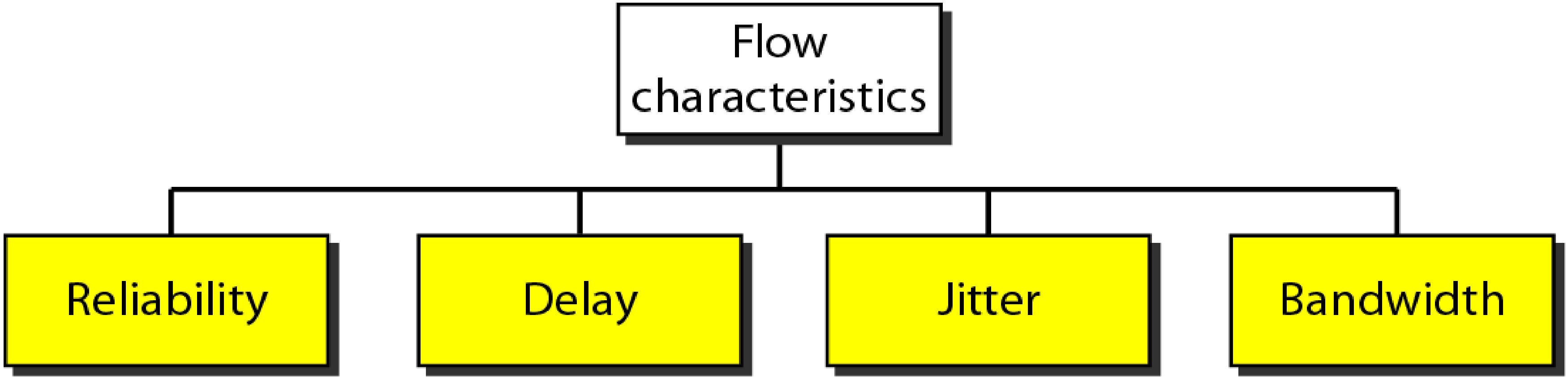
- **Backward Signaling** A bit can be set in a packet moving in the direction opposite to the congestion. This bit can warn the source that there is congestion and that it needs to slow down to avoid the discarding of packets.
- **Forward Signaling** A bit can be set in a packet moving in the direction of the congestion. This bit can warn the destination that there is congestion.



# Quality of Service (QoS)

Quality of Service can be defined as something a flow seeks to attain.

**Flow Characteristics:**

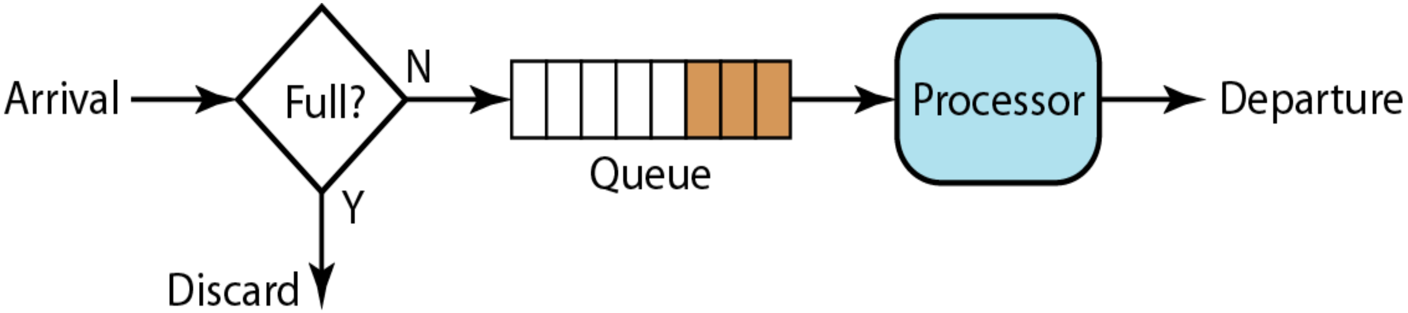


# Techniques to Improve QoS

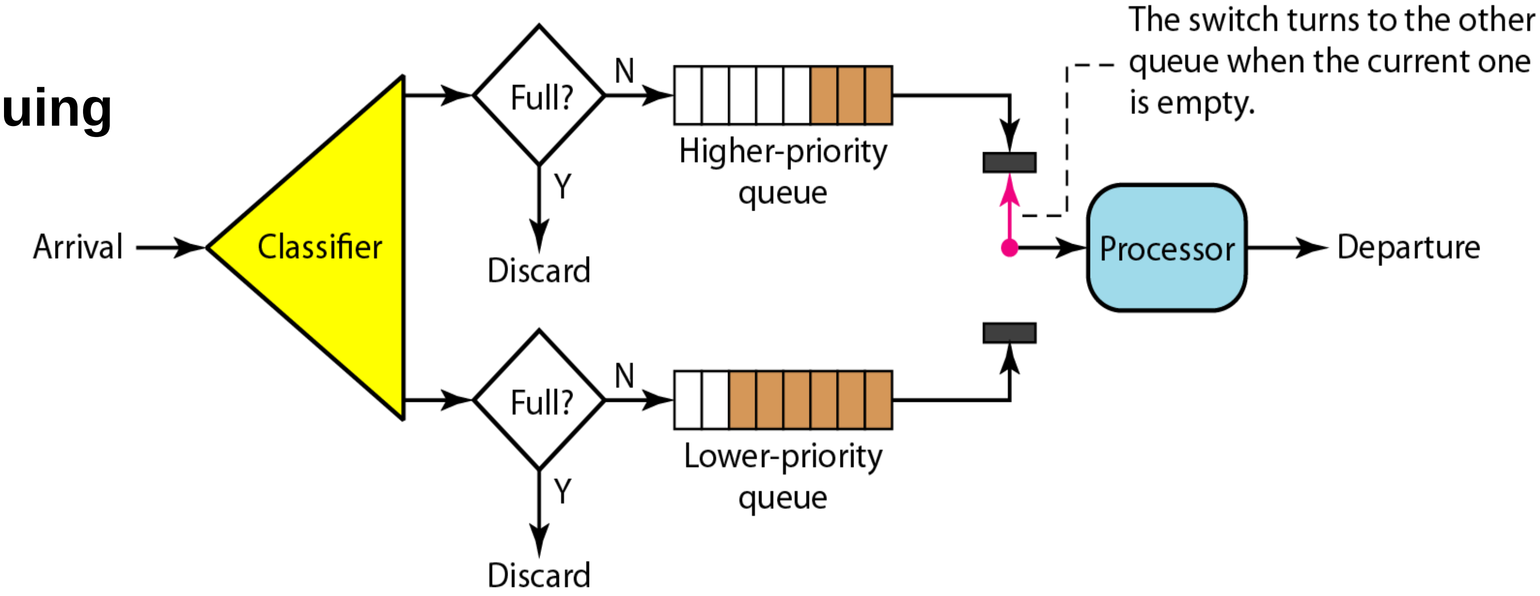
- **Scheduling**
- **Traffic Shaping**
- **Resource Reservation**
- **Admission Control**

# Scheduling

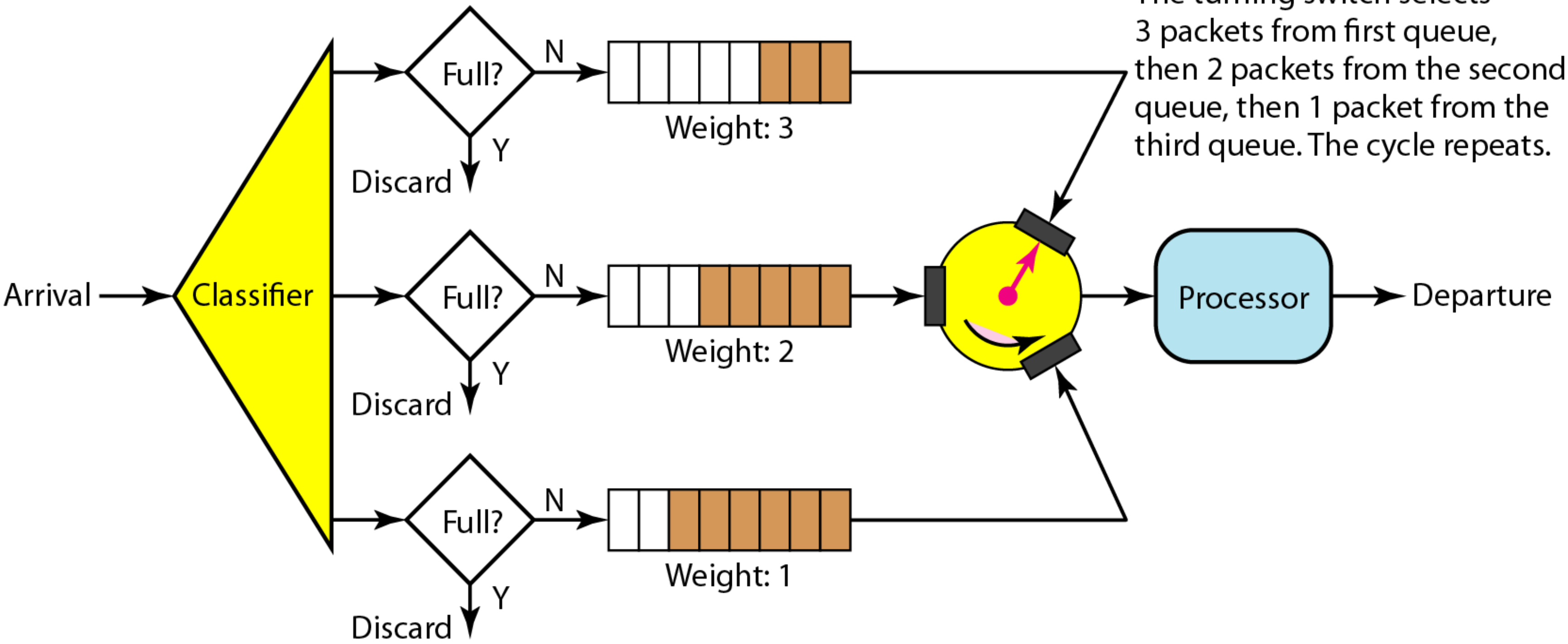
## FIFO Queuing



## Priority Queuing



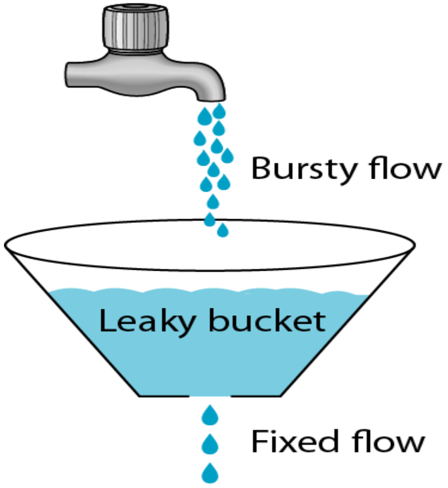
# Weighted Fair Queuing

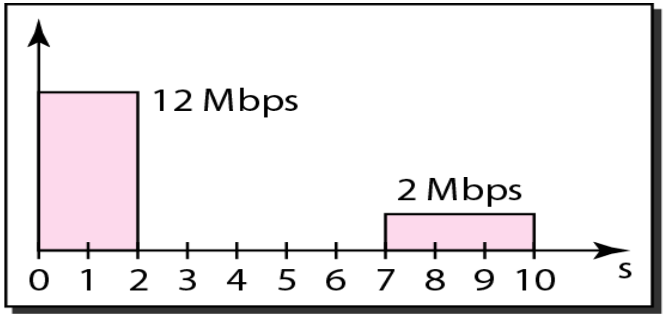


The turning switch selects 3 packets from first queue, then 2 packets from the second queue, then 1 packet from the third queue. The cycle repeats.

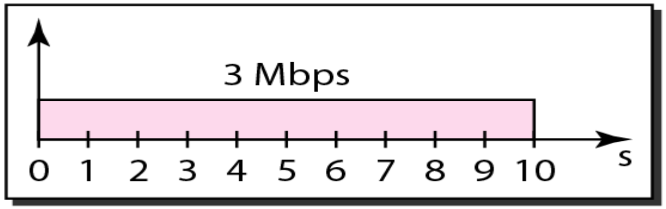
# Traffic Shaping

## Leaky Bucket



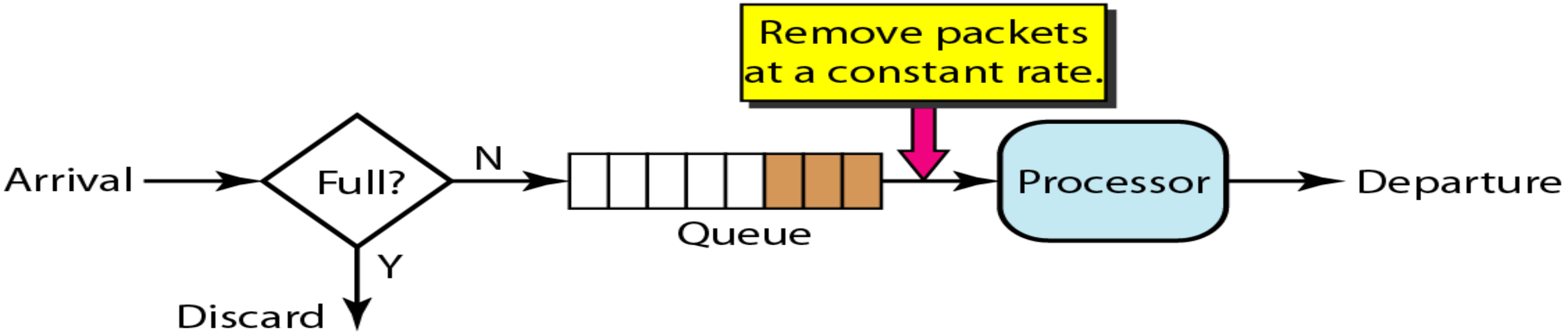


Bursty data



Fixed-rate data

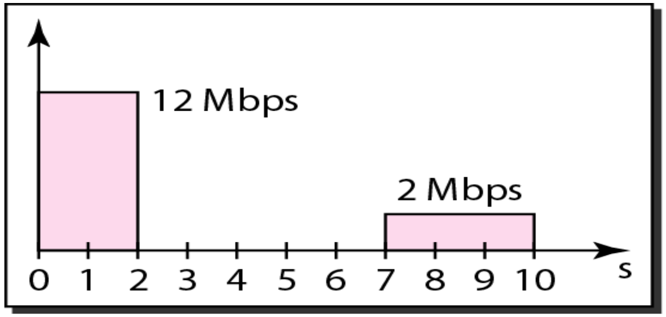
Leaky bucket algorithm



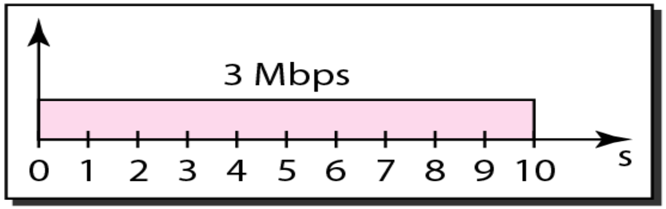
# Traffic Shaping

In the figure, we assume that the network has committed a bandwidth of 3 Mbps for a host.

The use of the leaky bucket shapes the input traffic to make it conform to this commitment.



Bursty data



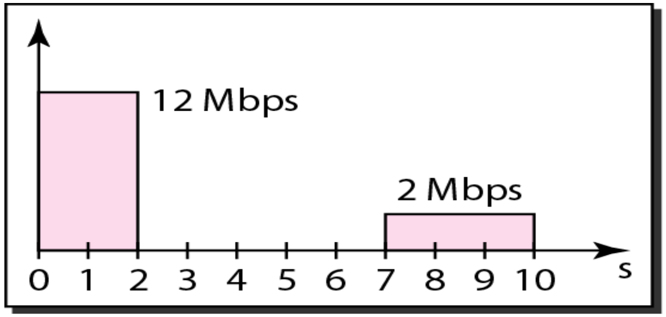
Fixed-rate data

# Traffic Shaping

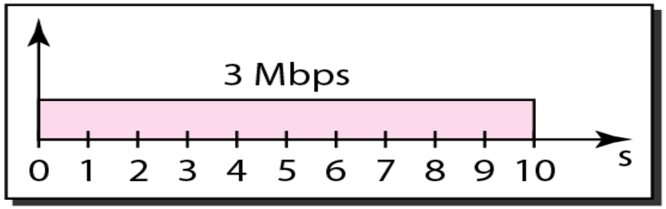
In Figure the host sends a burst of data at a rate of 12 Mbps for 2s, for a total of 24 Mbits of data.

The host is silent for 5s and then sends data at a rate of 2Mbps for 3s, for a total of 6Mbits of data.

In all, the host has sent 30 Mbits of data in lOs.



Bursty data

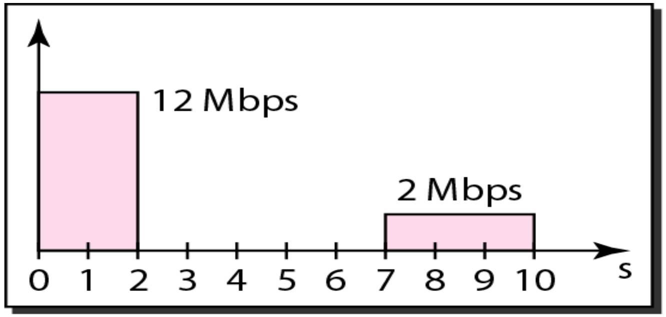


Fixed-rate data

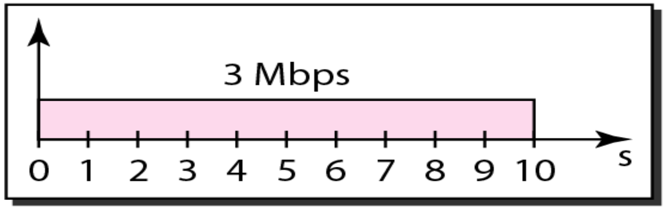
# Traffic Shaping

The leaky bucket smooth's the traffic by sending out data at a rate of 3Mbps during the same 10s.

Without the leaky bucket, the beginning burst may have hurt the network by consuming more bandwidth than is set aside for this host.



Bursty data

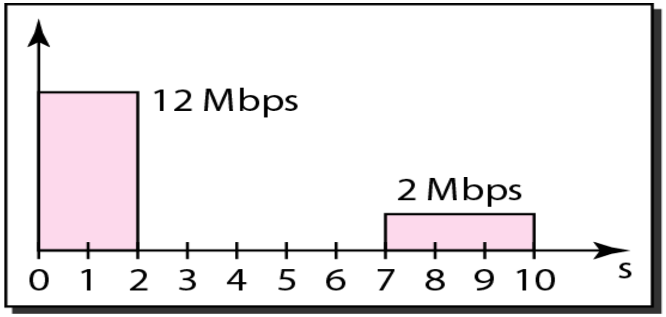


Fixed-rate data

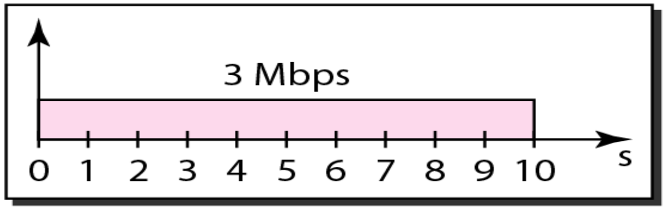
# Traffic Shaping

We can also see that the leaky bucket may prevent congestion.

As an analogy, consider the freeway during rush hour (bursty traffic). If, instead, commuters could stagger their working hours, congestion on our freeways could be avoided.



Bursty data

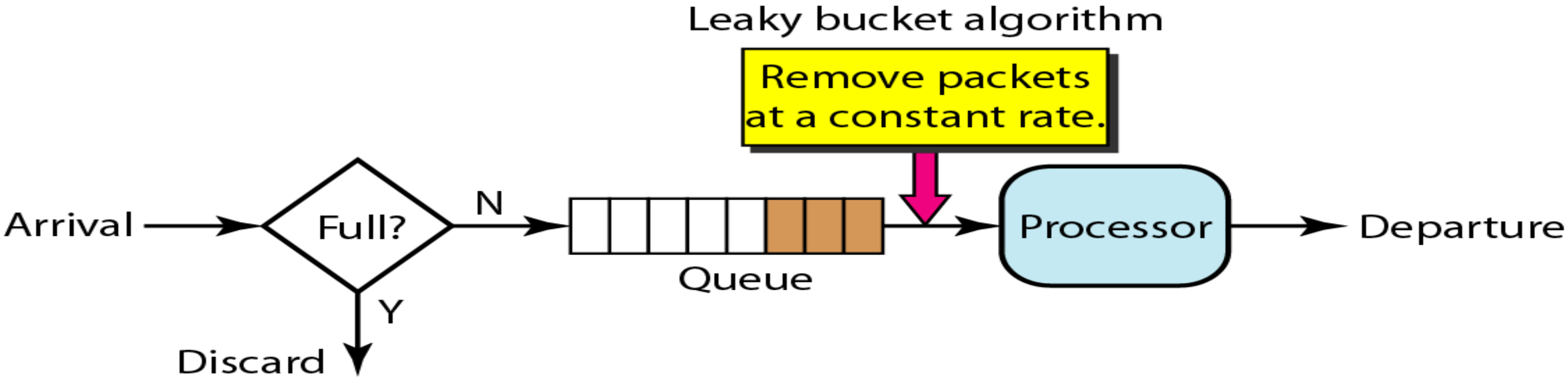


Fixed-rate data

# Traffic Shaping

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.

## Implementation of Leaky Bucket Algorithm



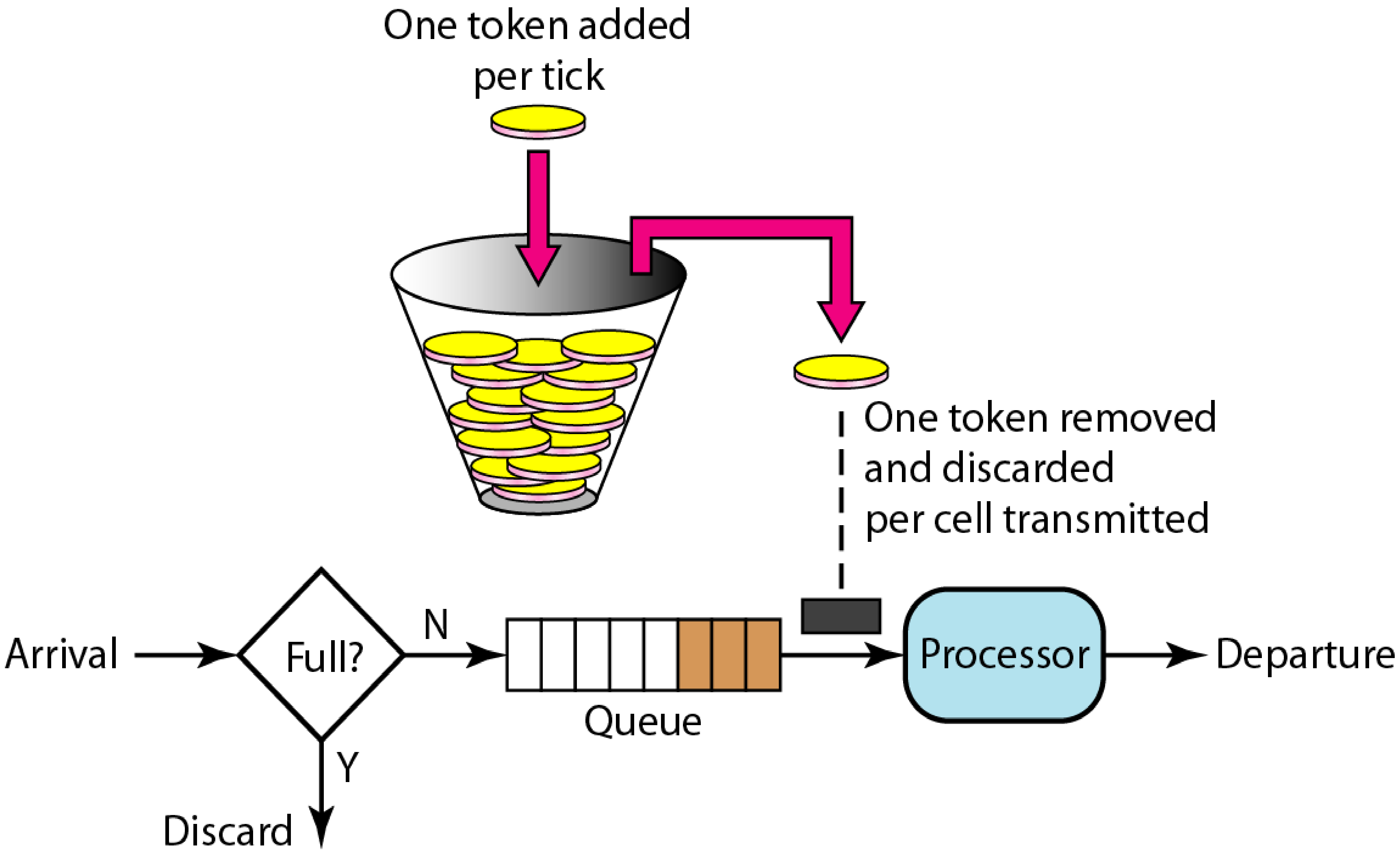
## Token Bucket

The leaky bucket is very restrictive. It does not credit an idle host.

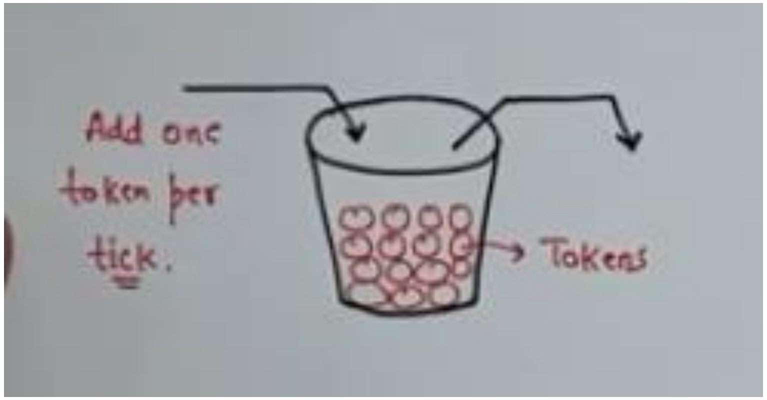
For example,

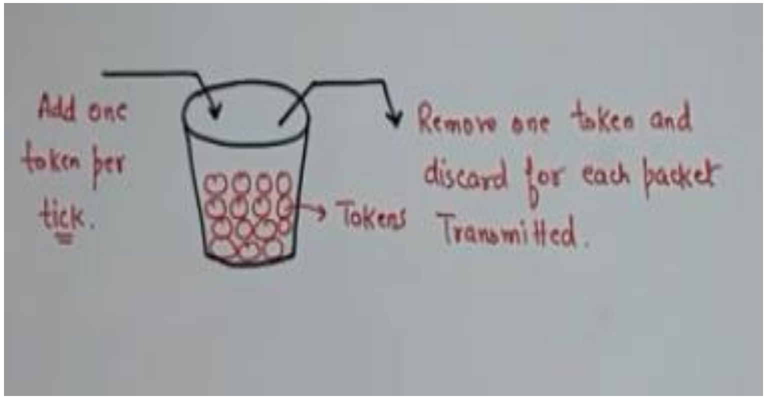
if a host is not sending for a while, its bucket becomes empty. Now if the host has bursty data, the leaky bucket allows only an average rate. The time when the host was idle is not taken into account.

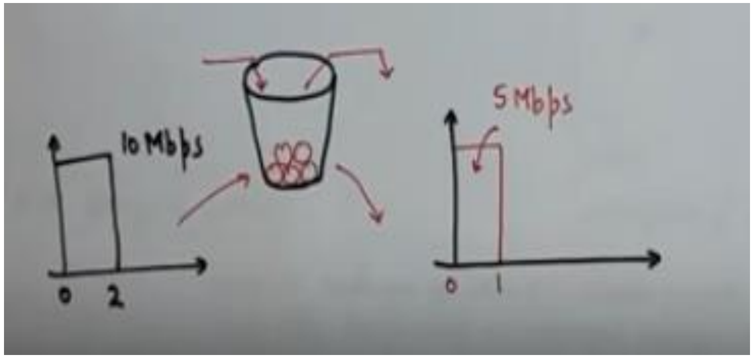
# Token Bucket

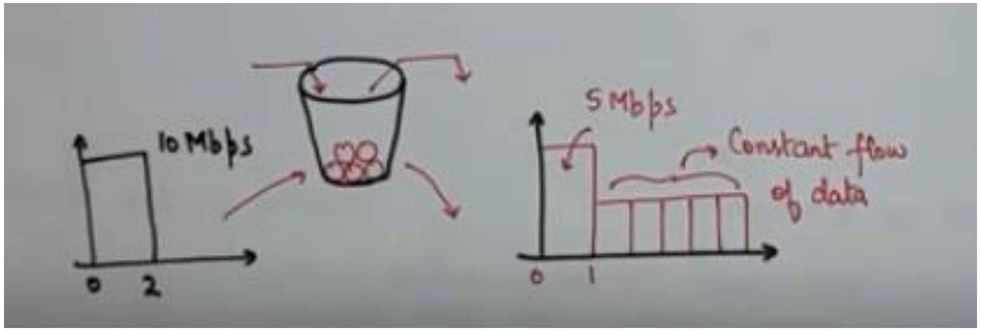


# Token Bucket









# Token Bucket

The token bucket algorithm allows idle hosts to accumulate credit for the future in the form of tokens.

For each tick of the clock, the system sends *n* tokens to the bucket.

# Token Bucket

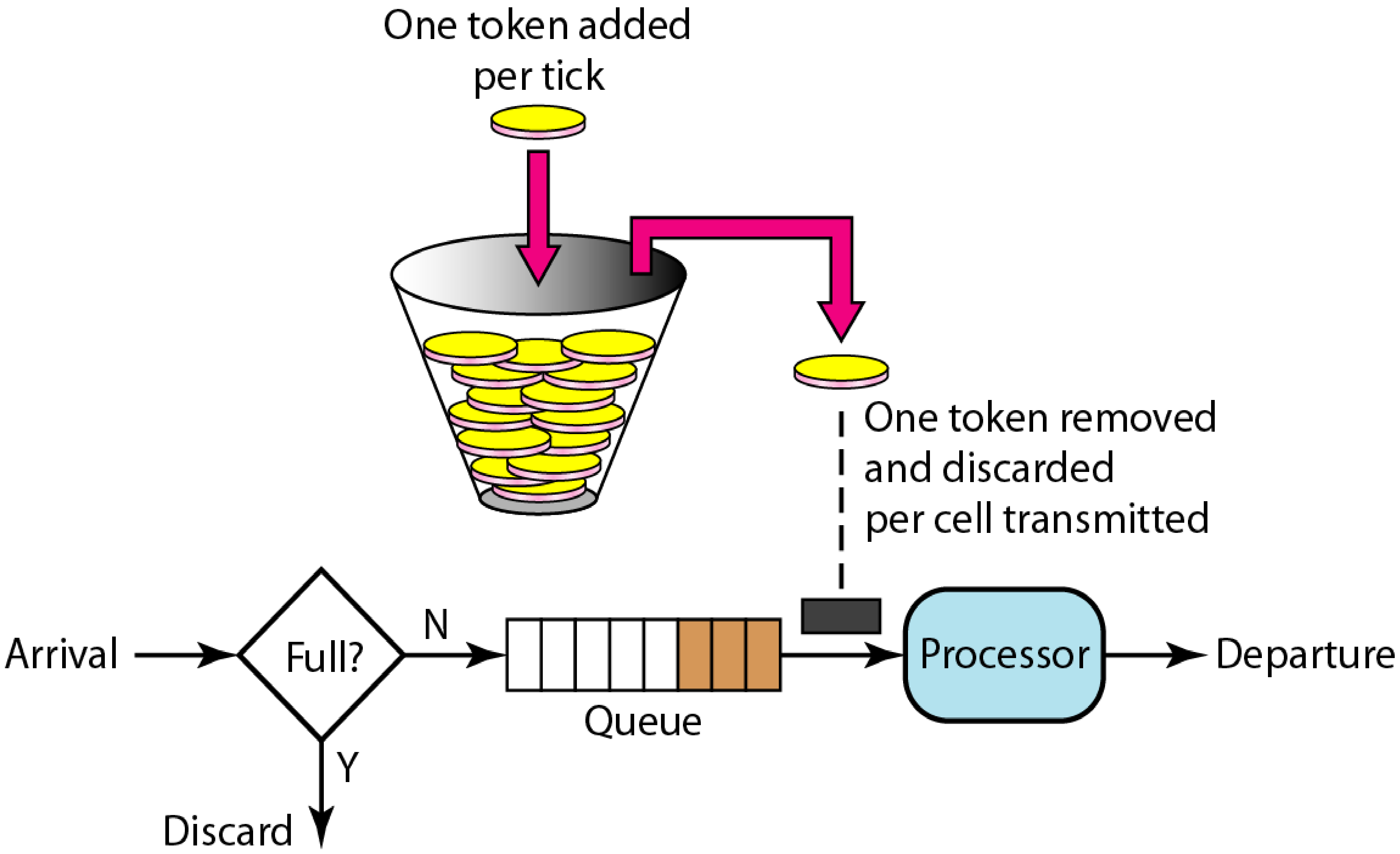
The system removes one token for every cell (or byte) of data sent. For example, if *n* is 100 and the host is idle for 100 ticks, the bucket collects 10,000 tokens. Now the host can consume all these tokens in one tick with 10,000 cells, or the host takes 1000 ticks with 10 cells per tick.

# Token Bucket

In other words, the host can send bursty data as long as the bucket is not empty.

Figure shows the idea. The token bucket can easily be implemented with a counter. The token is initialized to zero. Each time a token is added, the counter is incremented by 1. Each time a unit of data is sent, the counter is decremented by 1. When the counter is zero, the host cannot send data.

# Token Bucket



**Resource Reservation:**

- A flow of data needs resources such as a buffer, bandwidth, CPU time, and so on.
- The quality of service is improved if these resources are reserved beforehand.

**Admission Control:**

- Admission control refers to the mechanism used by a router, or a switch, to accept or reject a flow based on predefined parameters called flow specifications.