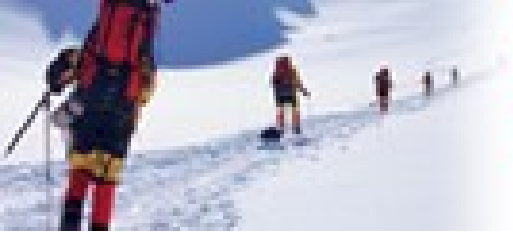


Data Communications and Networking Fourth Edition

Forouzan

Chapter 24

Congestion Control and Quality of Service

Congestion Control and Quality of Service

These are issues related not to one layer, but to three:

- The data link layer,
- The network layer, and
- The transport layer.

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.

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

Congestion Control Introduction:

- When too many packets are present in (a part of) the subnet, performance degrades. This situation is called **congestion**.
- As traffic increases too far, the routers are no longer able to cope and they begin losing packets.
- At very high traffic, performance collapses completely and almost no packets are delivered.
- Congestion happens in any system that involves waiting.

Congestion Control Introduction:

- Reasons of Congestion:
  - Slow Processors.
  - High stream of packets sent from one of the sender.
  - Insufficient memory.
  - High memory of Routers also add to congestion as becomes un manageable and un accessible. (Nagle, 1987).
  - Low bandwidth lines.

Congestion Control Introduction:

- Then what is **congestion control**? Congestion control has to do with making sure the subnet is able to carry the offered traffic.
- Congestion control and flow control are often confused but both helps reduce congestion.
  - Congestion control is a global issue – involves every router and host within the subnet
  - Flow control – scope is point-to-point; involves just sender and receiver.

General Principles of Congestion Control

- Three Step approach to apply congestion control:
  1. Monitor the system .
     - detect when and where congestion occurs.
  2. Pass information to where action can be taken.
  3. Adjust system operation to correct the problem.
- How to monitor the subnet for congestion.
  1) percentage of all packets discarded for lack of buffer space,
  2) average queue lengths,
  3) number of packets that time out and are retransmitted,
  4) average packet delay
  5) standard deviation of packet delay (jitter Control).

- Knowledge of congestion will cause the hosts to take appropriate action to reduce the congestion.
- Dividing all algorithms into
  - open loop or
  - closed loop

- The presence of congestion means that the load is (temporarily) greater than the resources can handle.
- Solution?
  - increase the resources or
  - decrease the load.
- That is not always possible. So we have to apply some congestion prevention policy.

24-3 CONGESTION CONTROL

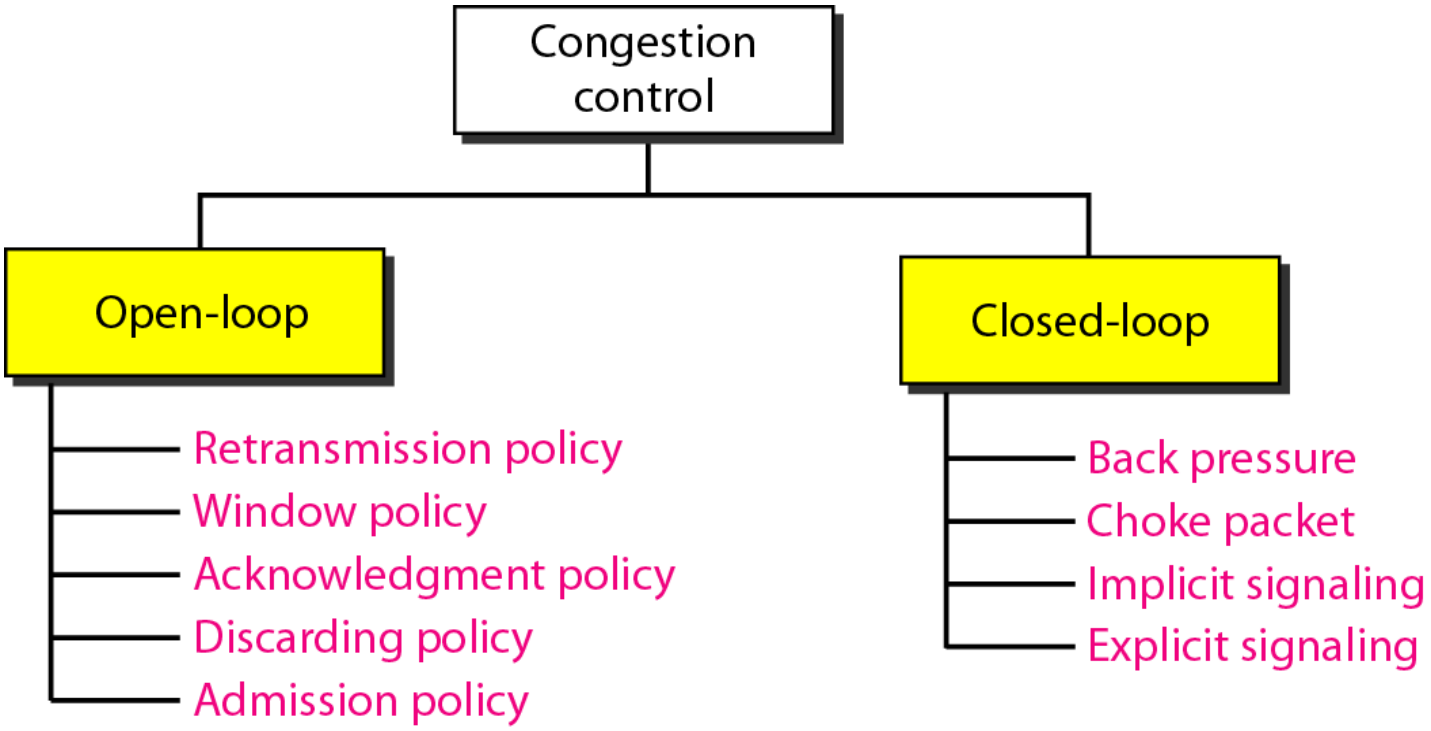
Congestion control refers to techniques and mechanisms that can either

- **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).

Figure 24.5 *Congestion control categories*



They further divide the open loop algorithms into ones that act at the source versus ones that act at the destination.

The closed loop algorithms are also divided into two subcategories:

- Explicit feedback
- Implicit feedback.

- In explicit feedback algorithms, packets are sent back from the point of congestion to warn the source.
- In implicit algorithms, the source deduces the existence of congestion by making local observations, such as the time needed for acknowledgements to come back.

Retransmission Policy

Retransmission is sometimes unavoidable. If the sender feels that a sent packet is lost or corrupted, the packet needs to be retransmitted.

Retransmission in general may increase congestion in the network. However, a good retransmission policy can prevent congestion.

The retransmission policy and the retransmission timers must be designed to optimize efficiency and at the same time prevent congestion. For example, the retransmission policy used by TCP (explained later) is designed to prevent or alleviate congestion.

The acknowledgment policy imposed by the receiver may also affect congestion.

- If the receiver does not acknowledge every packet it receives, it may slow down the sender and help prevent congestion.
- Several approaches are used in this case.
  - A receiver may send an acknowledgment only if it has a packet to be sent or a special timer expires.
  - A receiver may decide to acknowledge only *N* packets at a time. We need to know that the acknowledgments are also part of the load in a network. Sending fewer acknowledgments means imposing less load on the network.

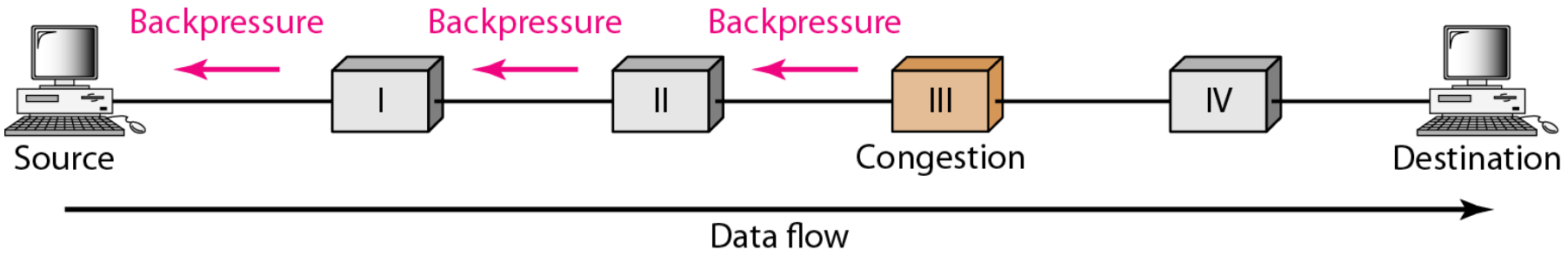
Disregarding Policy/ Load Shedding

- When buffers become full, routers simply discard packets.
- Which packet is chosen to be the victim depends on the application and on the error strategy used in the data link layer.
- For a file transfer, for, e.g. cannot discard older packets since this will cause a gap in the received data.
- For real-time voice or video it is probably better to throw away old data and keep new packets.
- Get the application to mark packets with discard priority.

Warning Bit or Backpressure:

- DECNET(Digital Equipment Corporation to connect mini computers) architecture signaled the warning state by setting a special bit in the packet's header.
- The source then cut back on traffic.
- The source monitored the fraction of acknowledgements with the bit set and adjusted its transmission rate accordingly.
- As long as the warning bits continued to flow in, the source continued to decrease its transmission rate. When they slowed to a trickle, it increased its transmission rate.
- Disadvantage: Note that since every router along the path could set the warning bit, traffic increased only when no router was in trouble.

Figure 24.6 *Backpressure method for alleviating congestion*



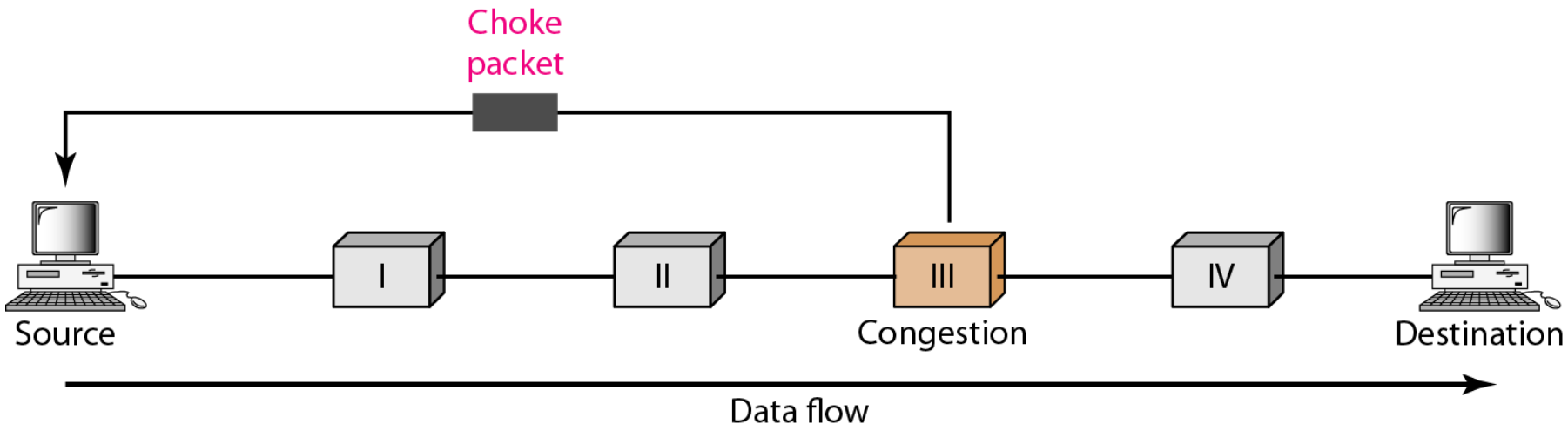
Choke Packets

- A more direct way of telling the source to slow down.
- A choke packet is a control packet generated at a congested node and transmitted to restrict traffic flow.
- The source, on receiving the choke packet must reduce its transmission rate by a certain percentage.
- An example of a choke packet is the ICMP Source Quench Packet.

Choke Packets:

- The original packet is tagged (a header bit is turned on) so that it will not generate any more choke packets farther along the path and is then forwarded in the usual way.
- Router maintains threshold. And based on it gives
  - Mild Warning
  - Stern Warning
  - Ultimatum.
- Variation: Use queue length or buffers instead of line utilization as trigger signal. This will reduce traffic. Chocks also increase traffic.

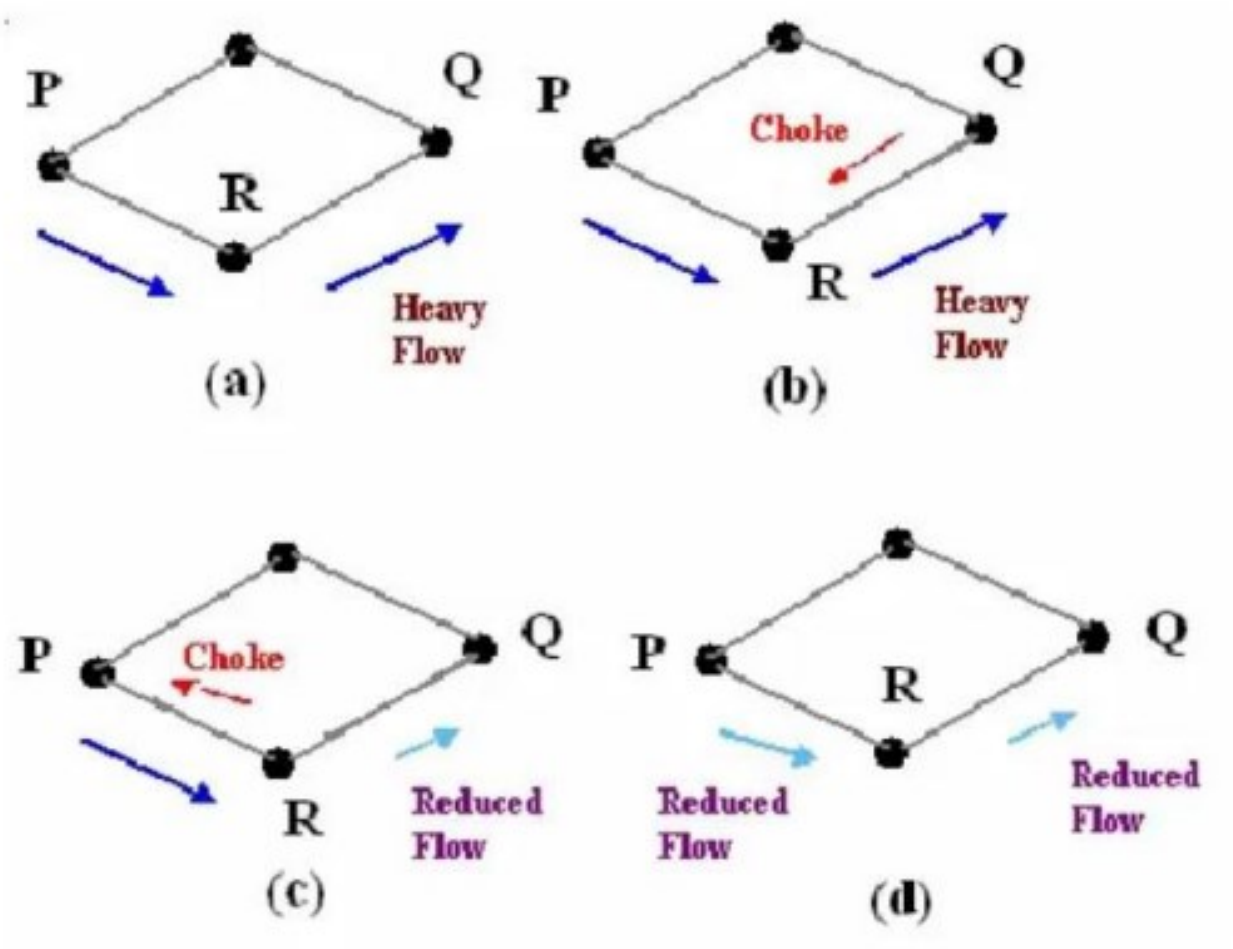
Figure 24.7 *Choke packet*



Hop-by-Hop Choke Packets

- This technique is an advancement over the Choked packet method. At high speed over long distances, sending a packet back to the source doesn't help much, because by the time the choke packet reaches the source, already a lot of packets destined for the same original the destination would be out from the source.
- So, to help this, Hop-by-Hop Choke packets are used. Over long distances or at high speeds choke packets are not very effective. A more efficient the method is to send choke packets hop-by-hop.
- This requires each hop to reduce its transmission even before the choke packet arrives at the source.

Hop-by-Hop Choke Packets



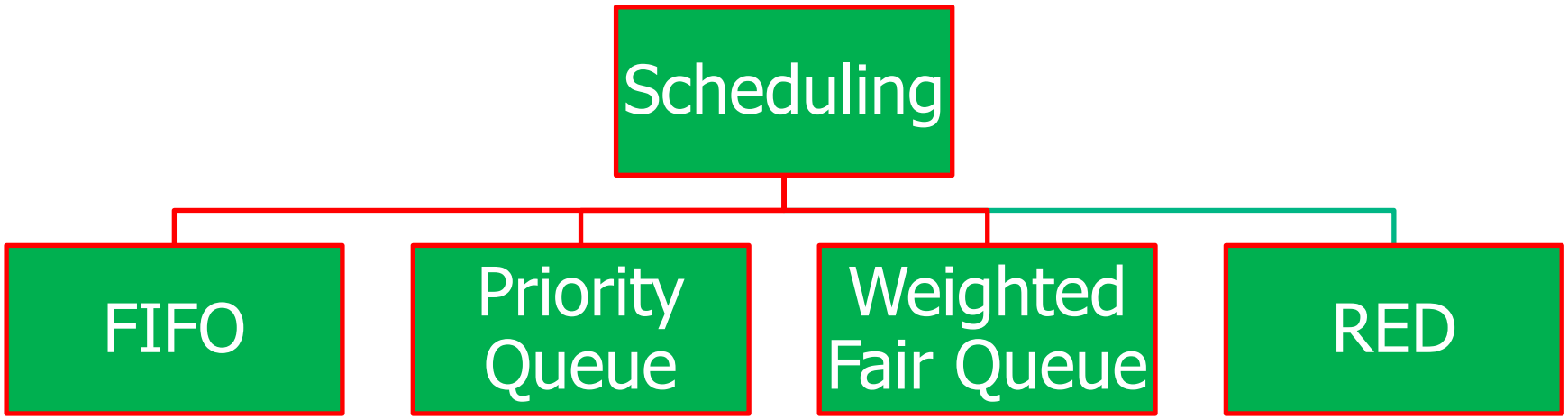
24-6 TECHNIQUES TO IMPROVE QoS

In this section, we discuss some techniques that can be used to improve the quality of service. We briefly discuss four common methods: scheduling, traffic shaping, admission control, and resource reservation.

Topics discussed in this section:

Scheduling (RED)
Traffic Shaping
Resource Reservation
Admission Control

Scheduling

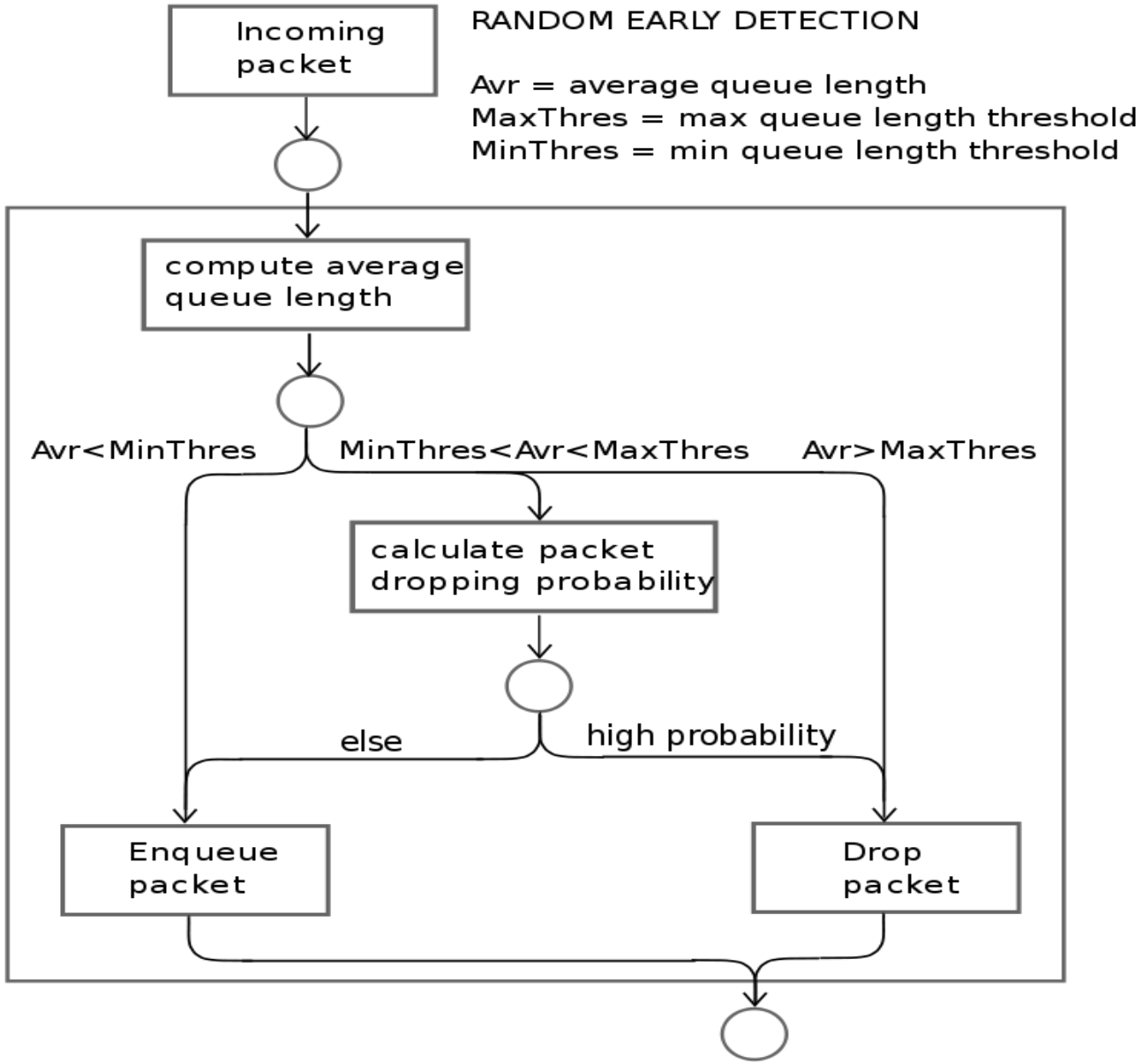


Random Early Discard (RED)

- This is a proactive approach in which the router discards one or more packets *before* the buffer becomes completely full.
- Each time a packet arrives, the RED algorithm computes the average queue length, ***avg***.
- If *avg* is lower than some lower threshold, congestion is assumed to be minimal or non-existent and the packet is queued.

RED, cont.

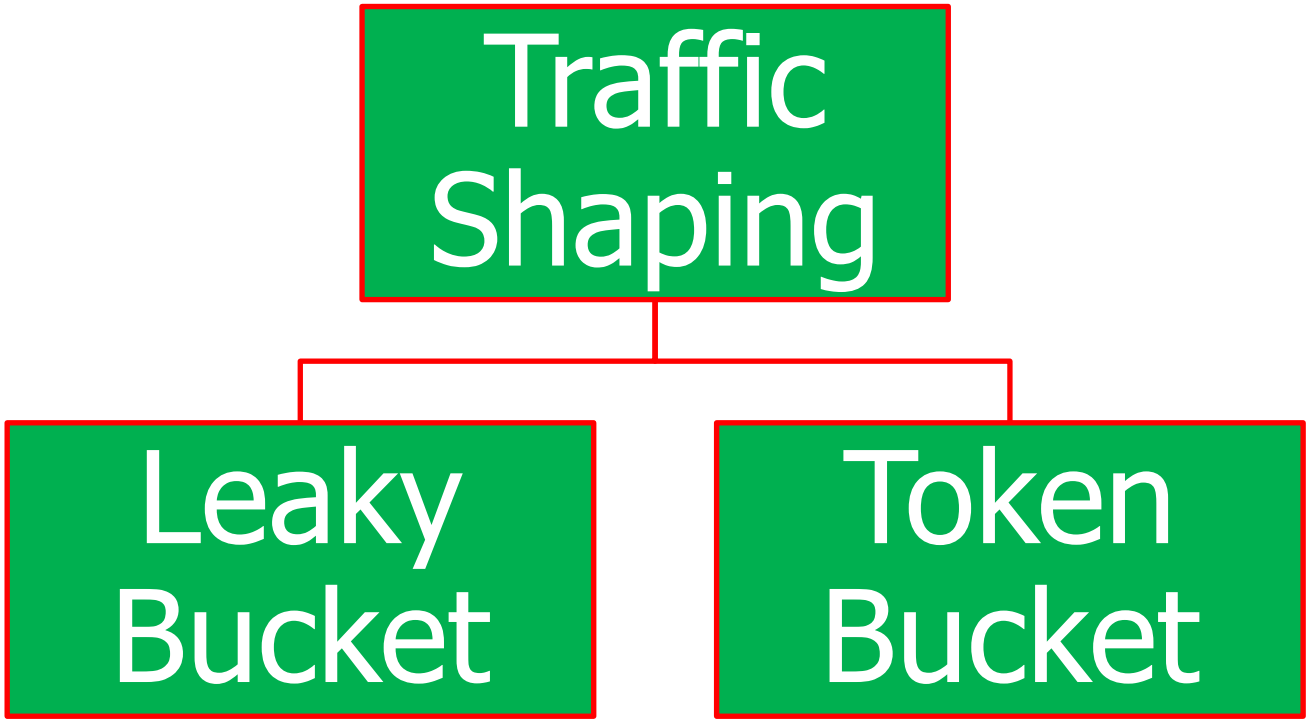
- If *avg* is greater than some upper threshold, congestion is assumed to be serious and the packet is discarded.
- If *avg* is between the two thresholds, this might indicate the onset of congestion. The probability of congestion is then calculated.


Incoming packet
RANDOM EARLY DETECTION
Avr = average queue length
MaxThres = max queue length threshold
MinThres = min queue length threshold
compute average queue length
Avr<MinThres
MinThres<Avr<MaxThres
Avr>MaxThres
calculate packet dropping probability
else
high probability
Enqueue packet
Drop packet

Traffic Shaping

- Another method of congestion control is to "shape" the traffic before it enters the network.
- Traffic shaping controls the *rate* at which packets are sent (not just how many). Used in ATM and Integrated Services networks.
- At connection set-up time, the sender and carrier negotiate a traffic pattern (shape).
- Two traffic shaping algorithms are:
  - Leaky Bucket
  - Token Bucket

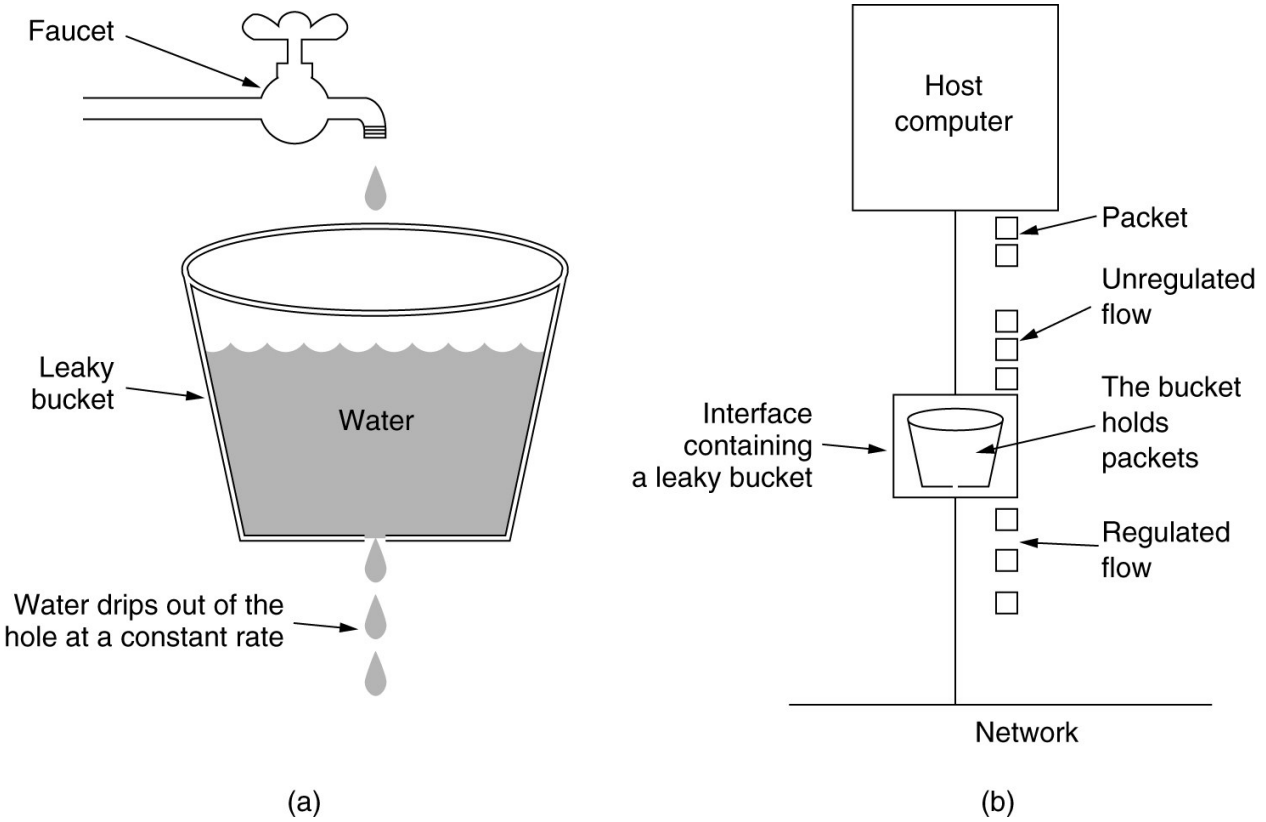
Traffic Shaping



The Leaky Bucket Algorithm

- The **Leaky Bucket Algorithm** used to control rate in a network. It is implemented as a single-server queue with constant service time. If the bucket (buffer) overflows then packets are discarded.

The Leaky Bucket Algorithm

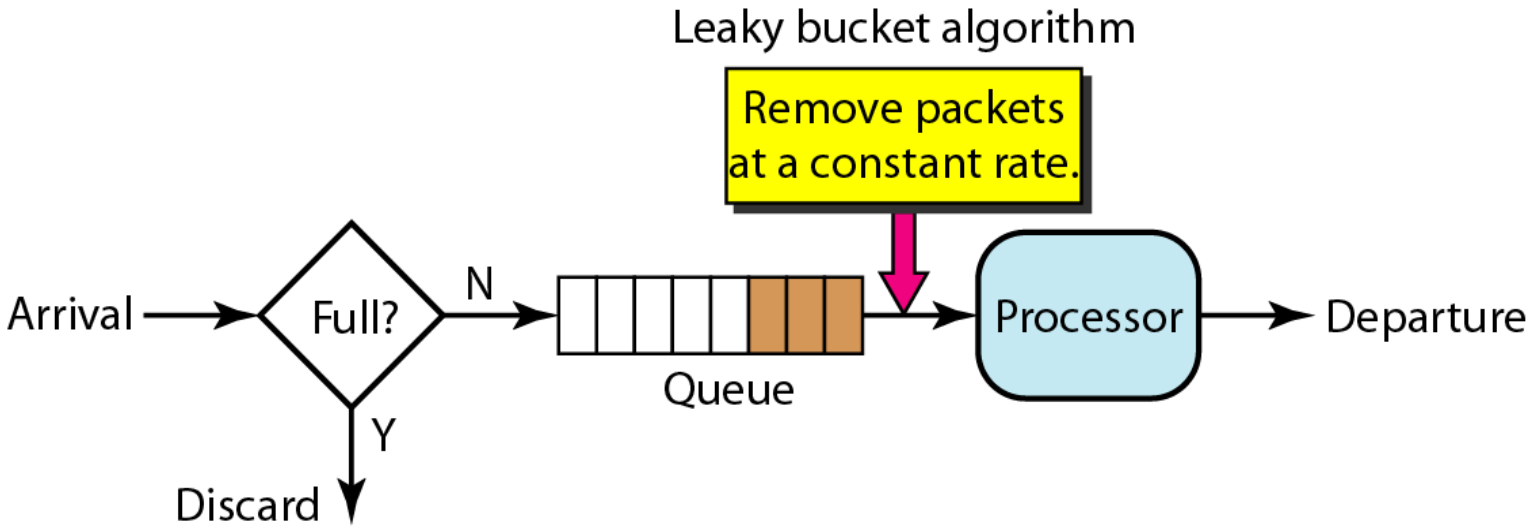


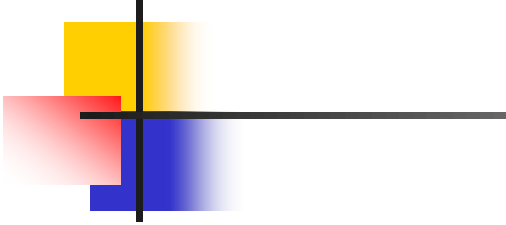
(a) A leaky bucket with water. (b) a leaky bucket with packets.

Leaky Bucket Algorithm, cont.

- The leaky bucket enforces a constant output rate (average rate) regardless of the burstiness of the input. Does nothing when input is idle.
- The host injects one packet per clock tick onto the network. This results in a uniform flow of packets, smoothing out bursts and reducing congestion.
- When packets are the same size (as in ATM cells), the one packet per tick is okay. For variable length packets though, it is better to allow a fixed number of bytes per tick. E.g. 1024 bytes per tick will allow one 1024-byte packet or two 512-byte packets or four 256-byte packets on 1 tick.

Figure 24.20 *Leaky bucket implementation*





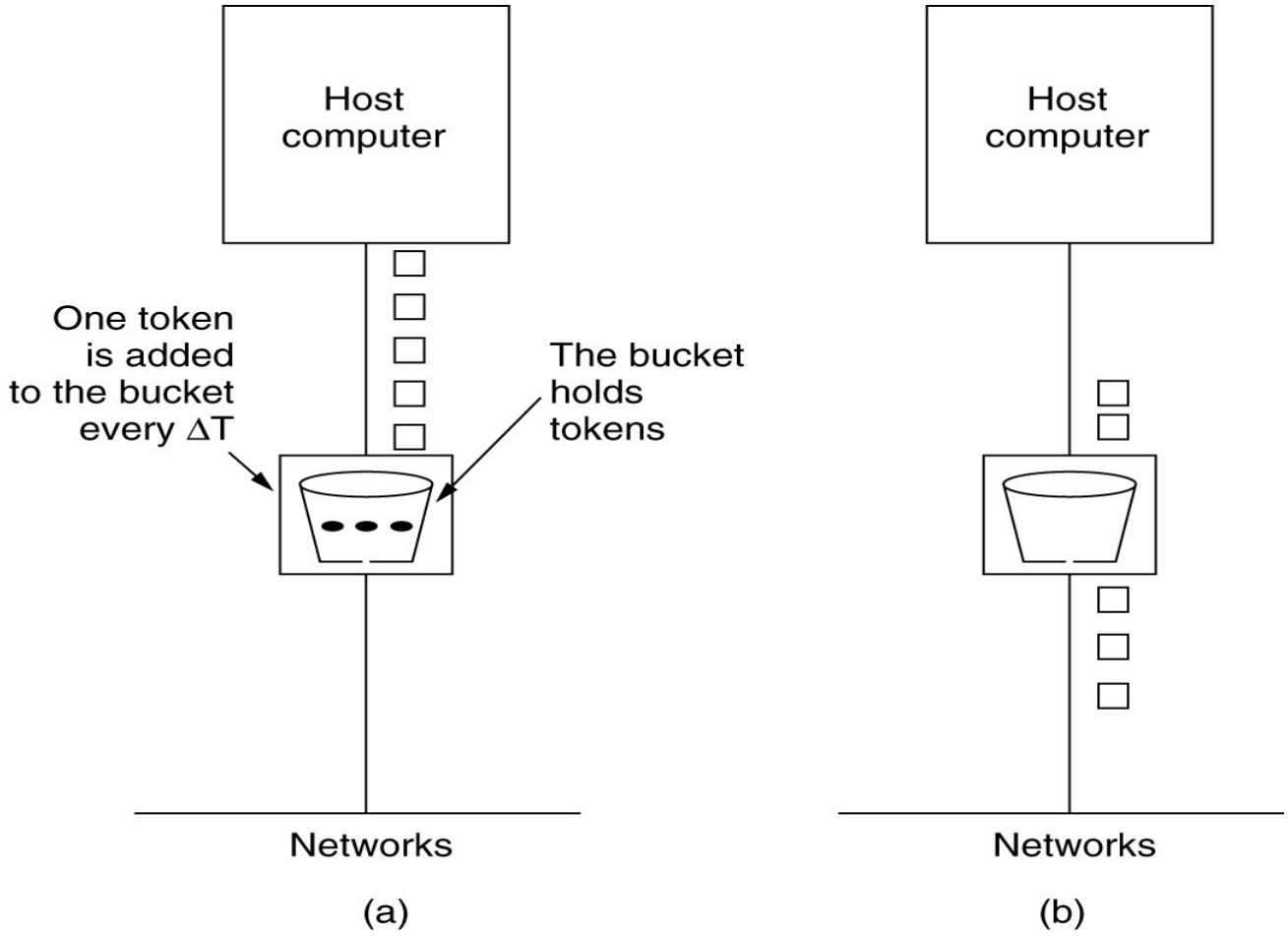
Note

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 Algorithm

- In contrast to the LB, the Token Bucket Algorithm, allows the output rate to vary, depending on the size of the burst.
- In the TB algorithm, the bucket holds tokens. To transmit a packet, the host must capture and destroy one token.
- Tokens are generated by a clock at the rate of one token every Δt sec.
- Idle hosts can capture and save up tokens (up to the max. size of the bucket) in order to send larger bursts later.

The Token Bucket Algorithm



(a) Before. (b) After.

Leaky Bucket vs Token Bucket

- LB discards packets; TB does not. TB discards tokens.
- With TB, a packet can only be transmitted if there are enough tokens to cover its length in bytes.
- LB sends packets at an average rate. TB allows for large bursts to be sent faster by speeding up the output.
- TB allows saving up tokens (permissions) to send large bursts. LB does not allow saving.

Figure 24.21 *Token bucket*