



# **Capítol 3. Core Network**

3.1 Nivell d'enllaç

3.2 Control de la congestió i QoS

3.3 Commutació d'etiquetes MPLS

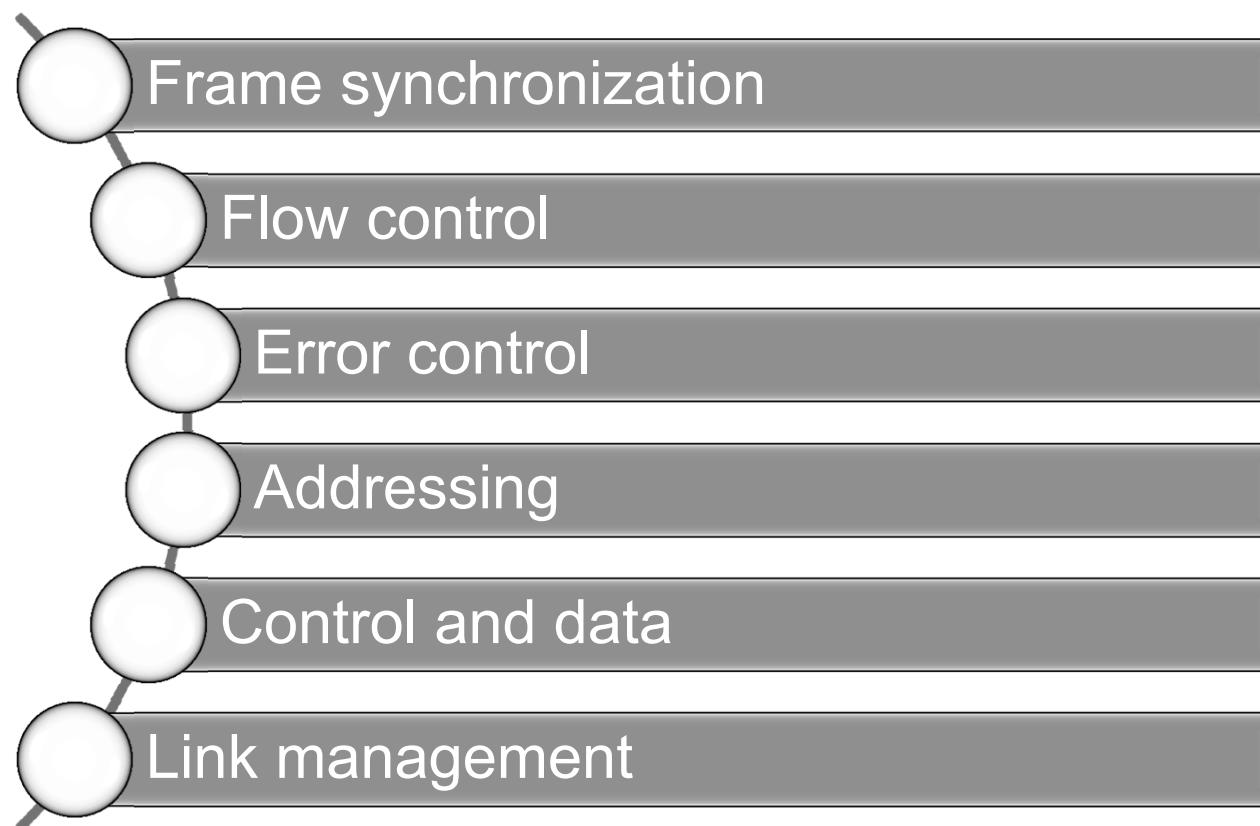
3.4 Software Defined Networking (SDN)



## **3.1 Nivell d'enllaç**

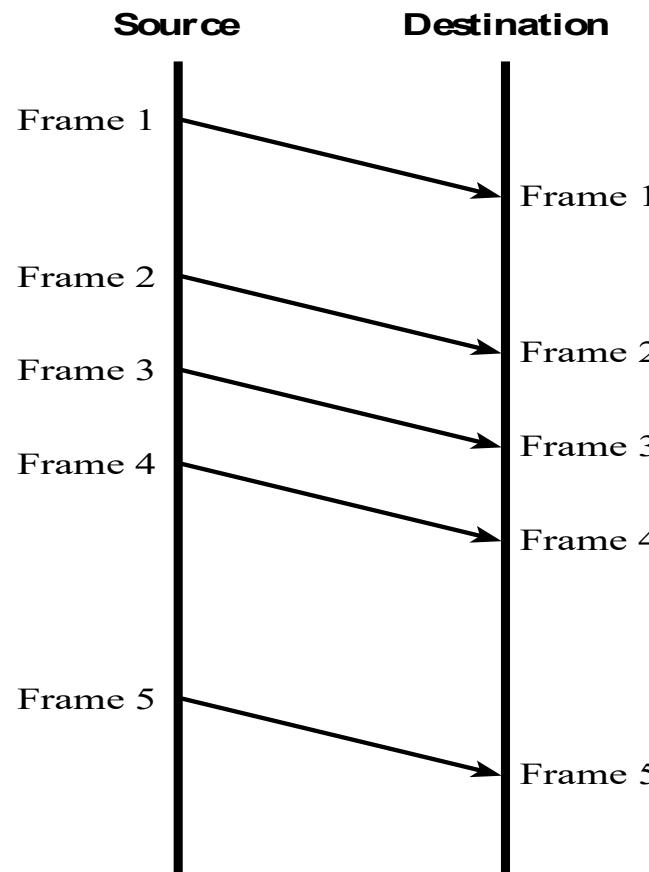
# Data Link Control Protocols

- Requirements and objectives for effective data communication between two directly connected transmitting-receiving stations:

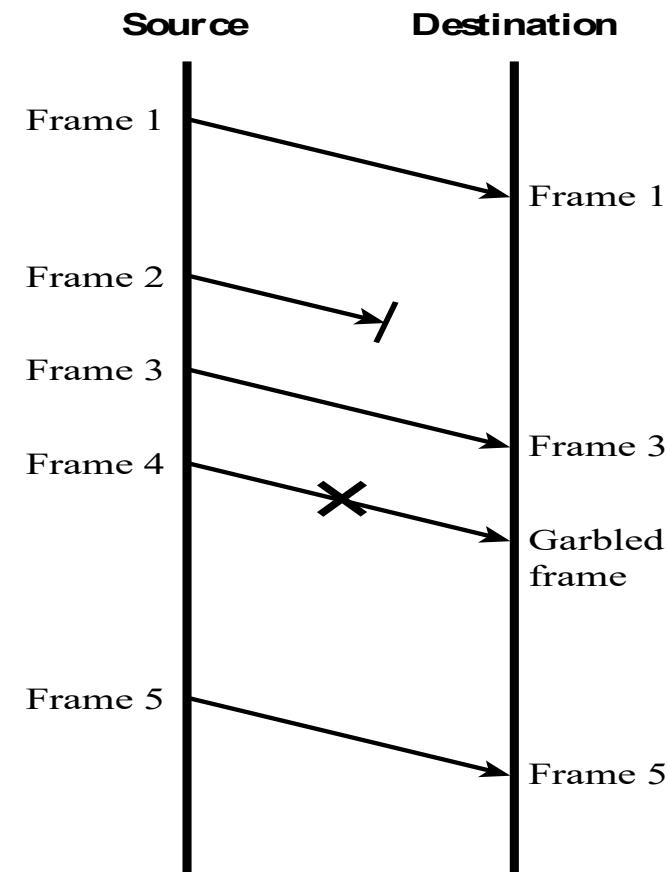


# Flow Control

- Technique for assuring that a transmitting entity does not overwhelm a receiving entity with data
  - The receiving entity typically allocates a data buffer of some maximum length for a transfer
  - When data are received, the receiver must do a certain amount of processing before passing the data to the higher-level software
- In the absence of flow control, the receiver's buffer may fill up and overflow while it is processing old data



(a) Error-free transmission

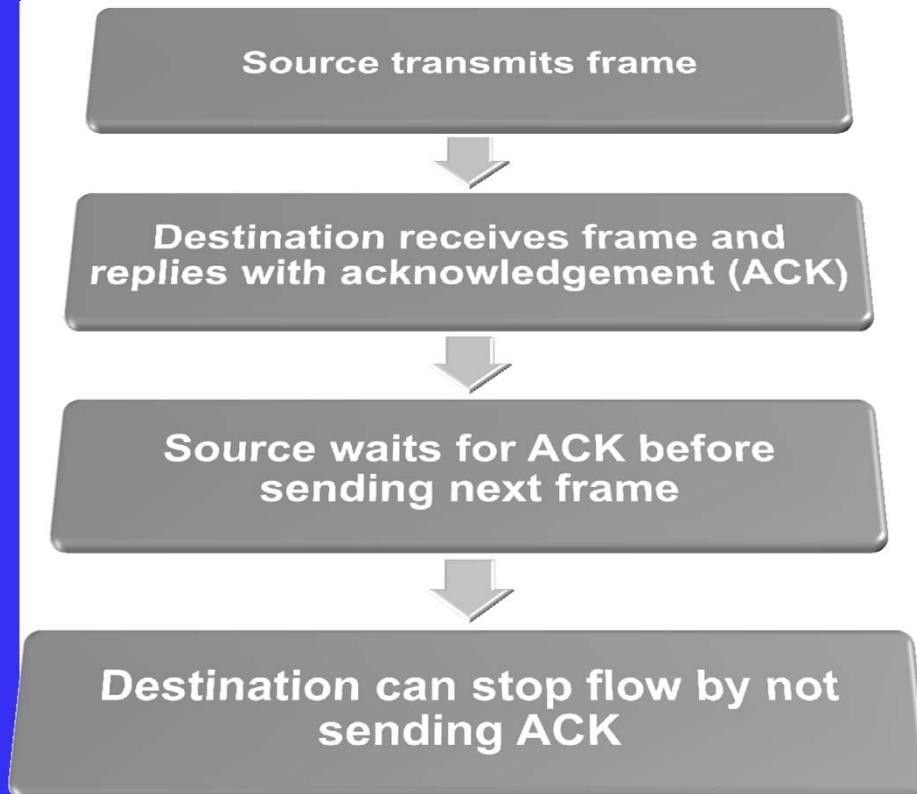


(b) Transmission with losses and errors

**Figure 7.1 Model of Frame Transmission**

# ***Stop-and-Wait Flow Control***

- Simplest form of flow control



- It is often the case that a source will break up a large block of data into smaller blocks and transmit the data in many frames
  - The buffer size of the receiver may be limited
  - The longer the transmission, the more likely that there will be an error, necessitating retransmission of the entire frame
  - On a shared medium it is usually desirable not to permit one station to the medium for an extended period, thus causing long delays at the other sending station

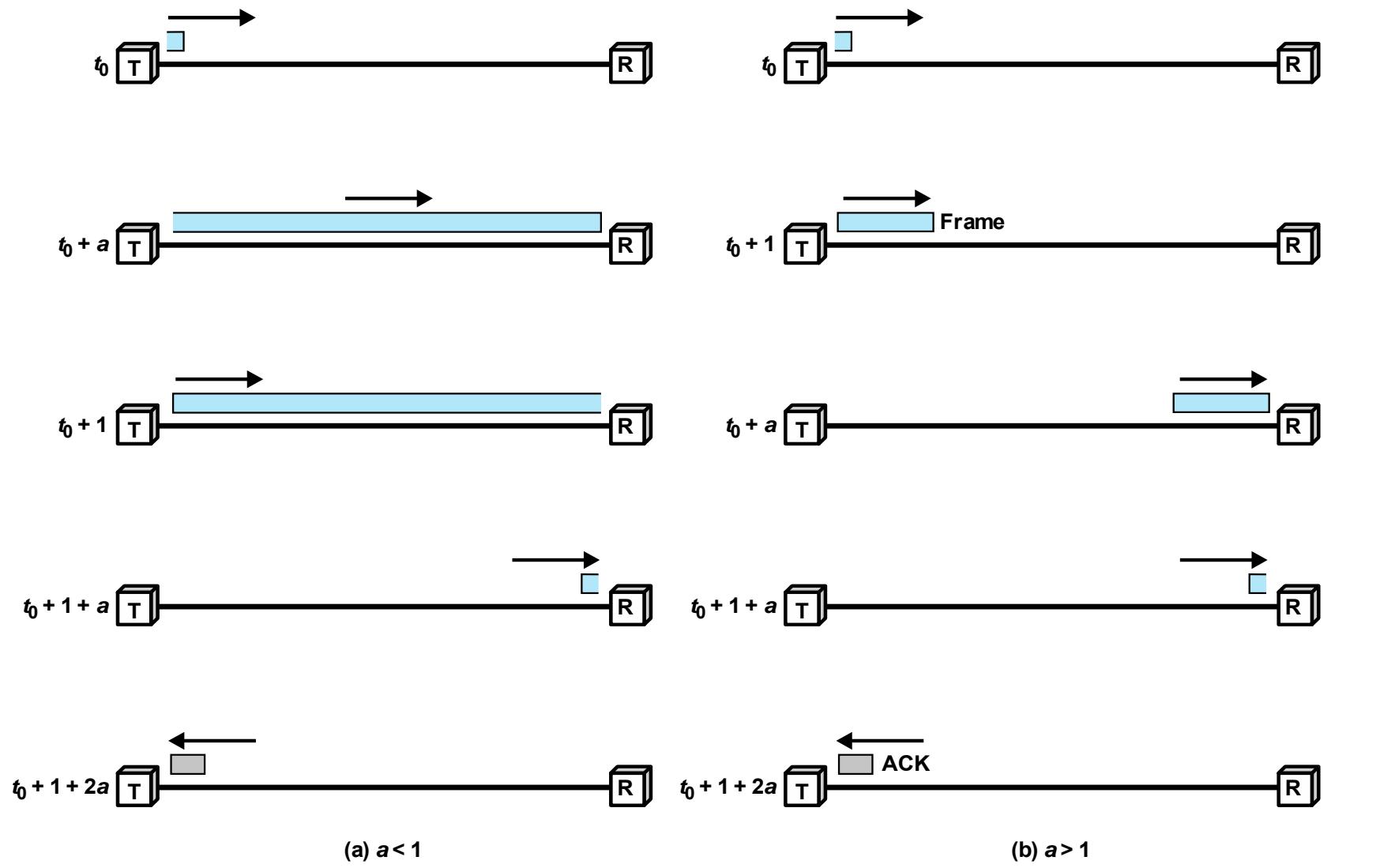


Figure 7.2 Stop-and-Wait Link Utilization (transmission time = 1; propagation time =  $a$ )

$$d = \frac{B}{L}$$

B = bits del Link

L = bits de la trama

$$\frac{V_p}{V_t} = \frac{\text{Kbytes}}{b/s} = \frac{Km}{b} \uparrow \frac{b}{Km} = \frac{V_t}{V_p}$$

# Sliding Windows Flow Control

- Allows multiple numbered frames to be in transit
  - Receiver has buffer W long
  - Transmitter sends up to W frames without ACK
  - ACK includes number of next frame expected
  - Sequence number is bounded by size of field (k)
    - *Frames are numbered modulo  $2^k$*
    - *Giving max window size of up to  $2^k - 1$*
  - Receiver can ACK frames without permitting further transmission (Receive Not Ready)
  - Must send a normal acknowledge to resume
- If have full-duplex link, can piggyback ACKs

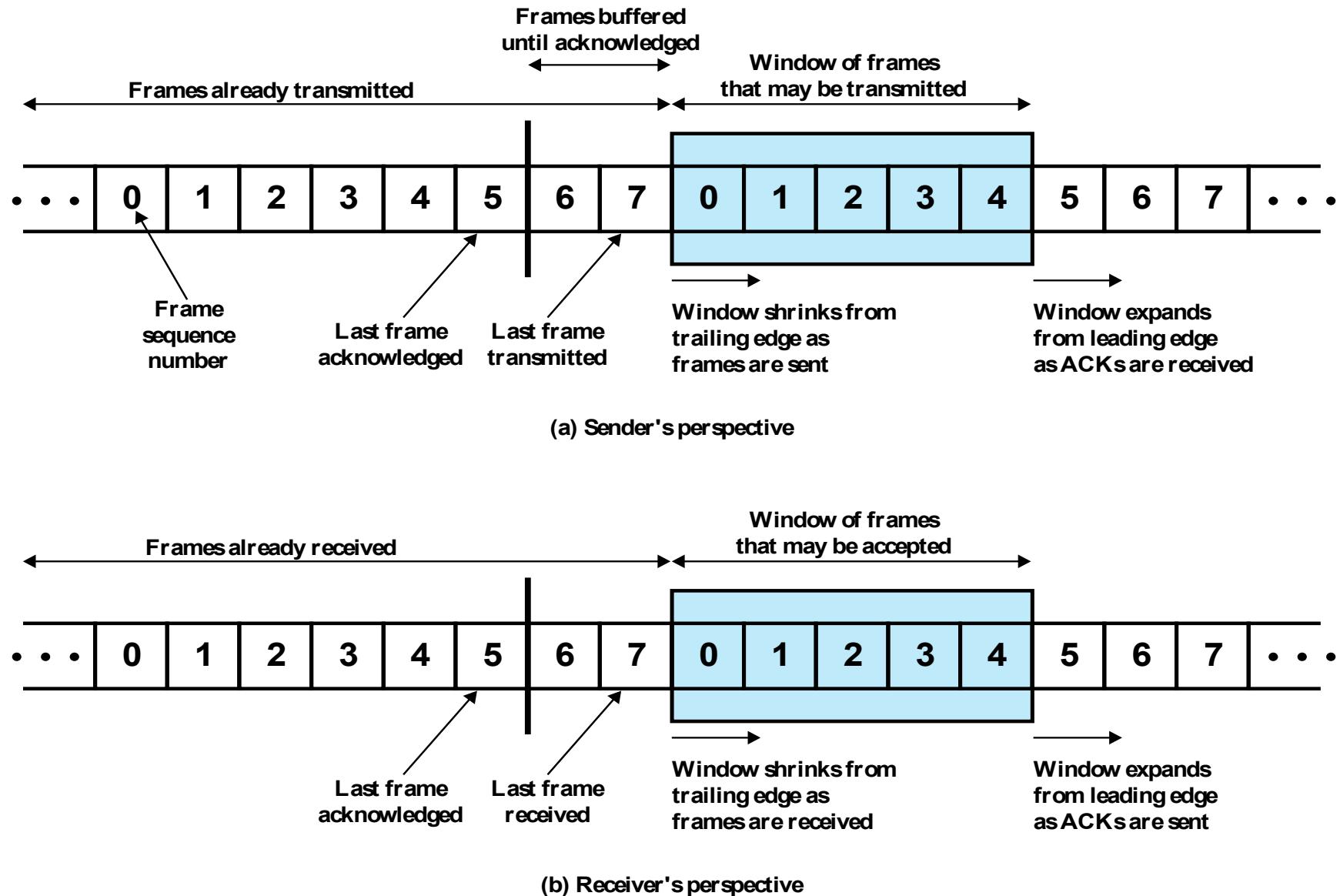
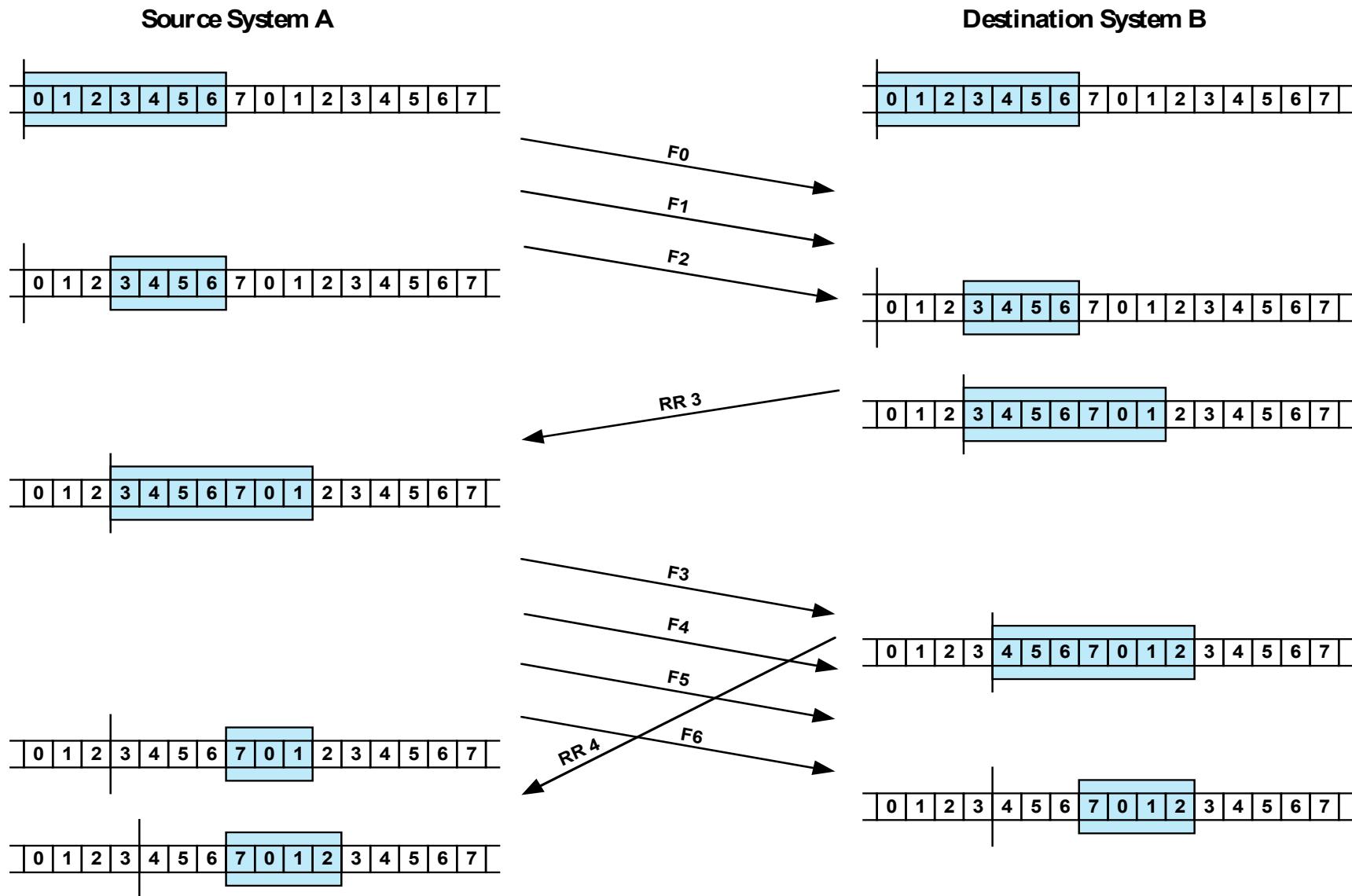
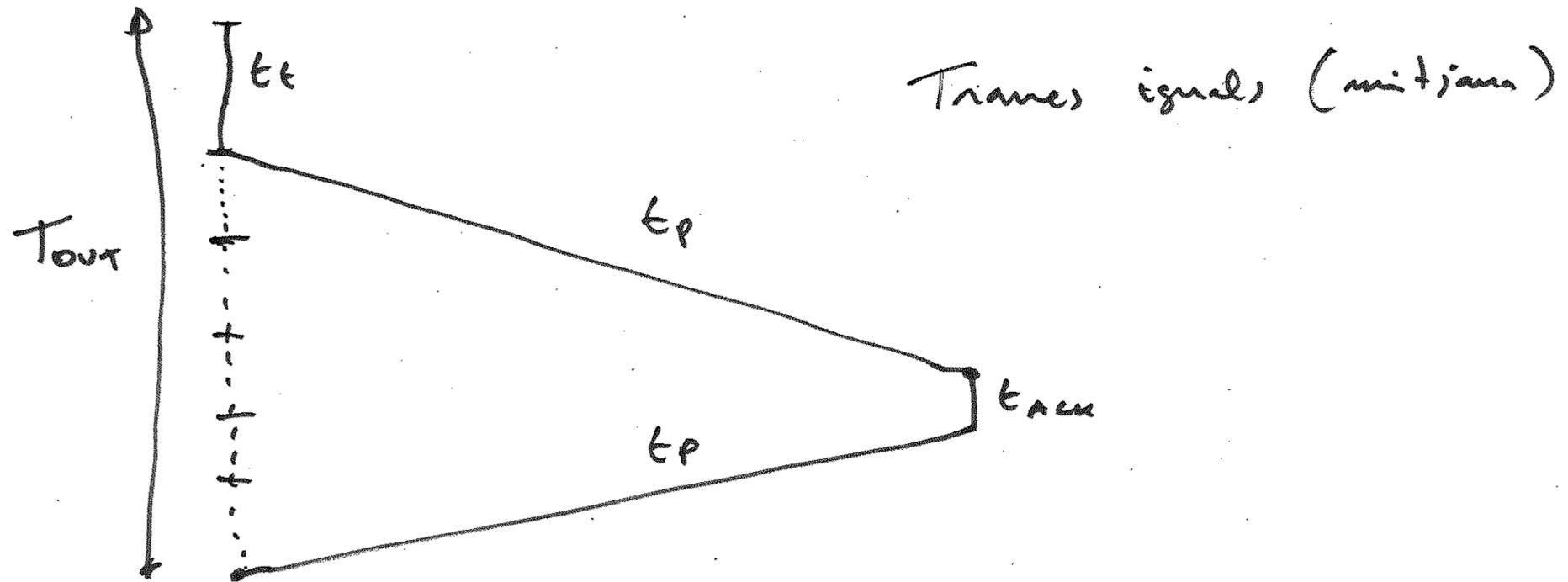


Figure 7.3 Sliding-Window Depiction



**Figure 7.4 Example of a Sliding-Window Protocol**

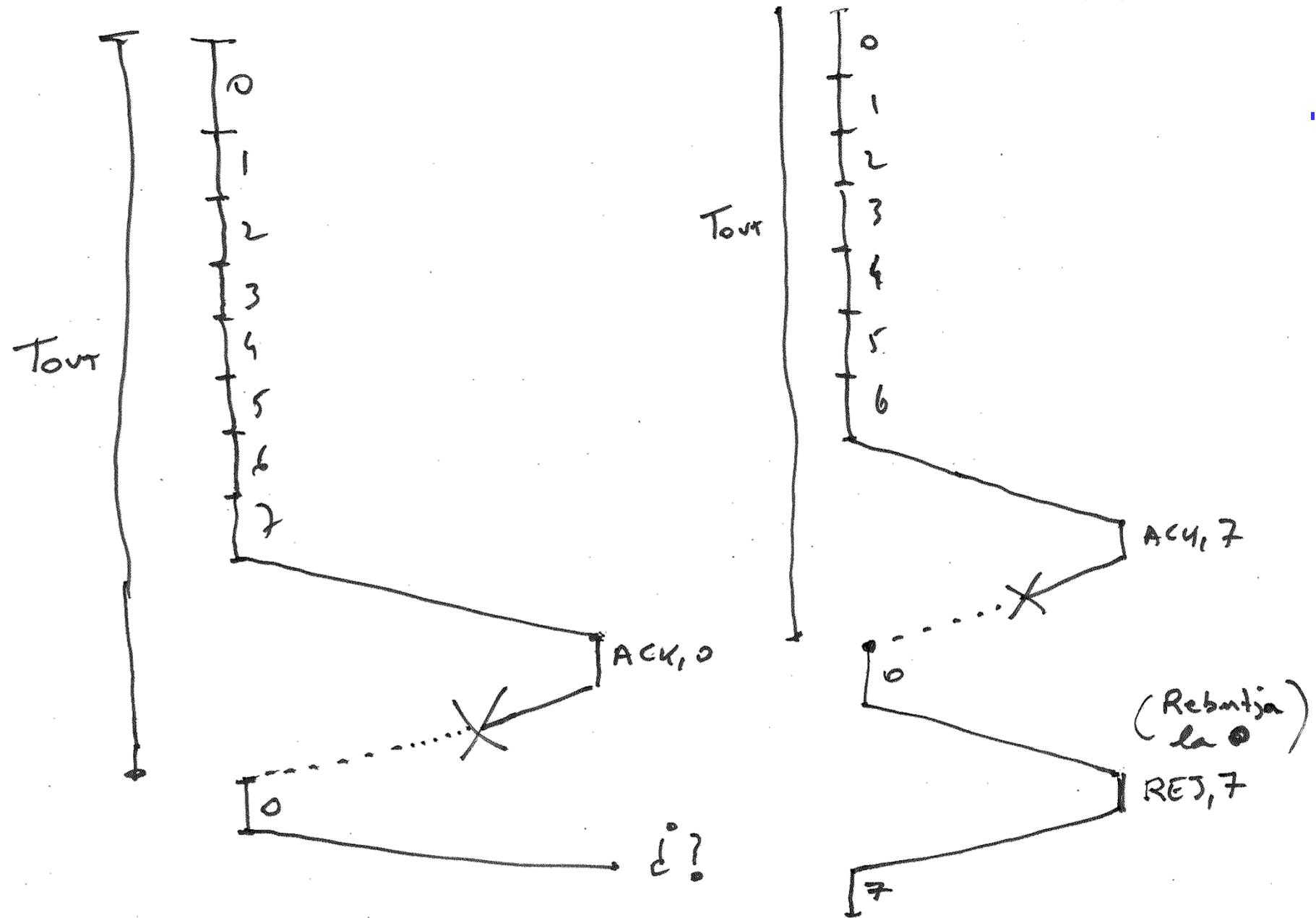
## FINESTRA



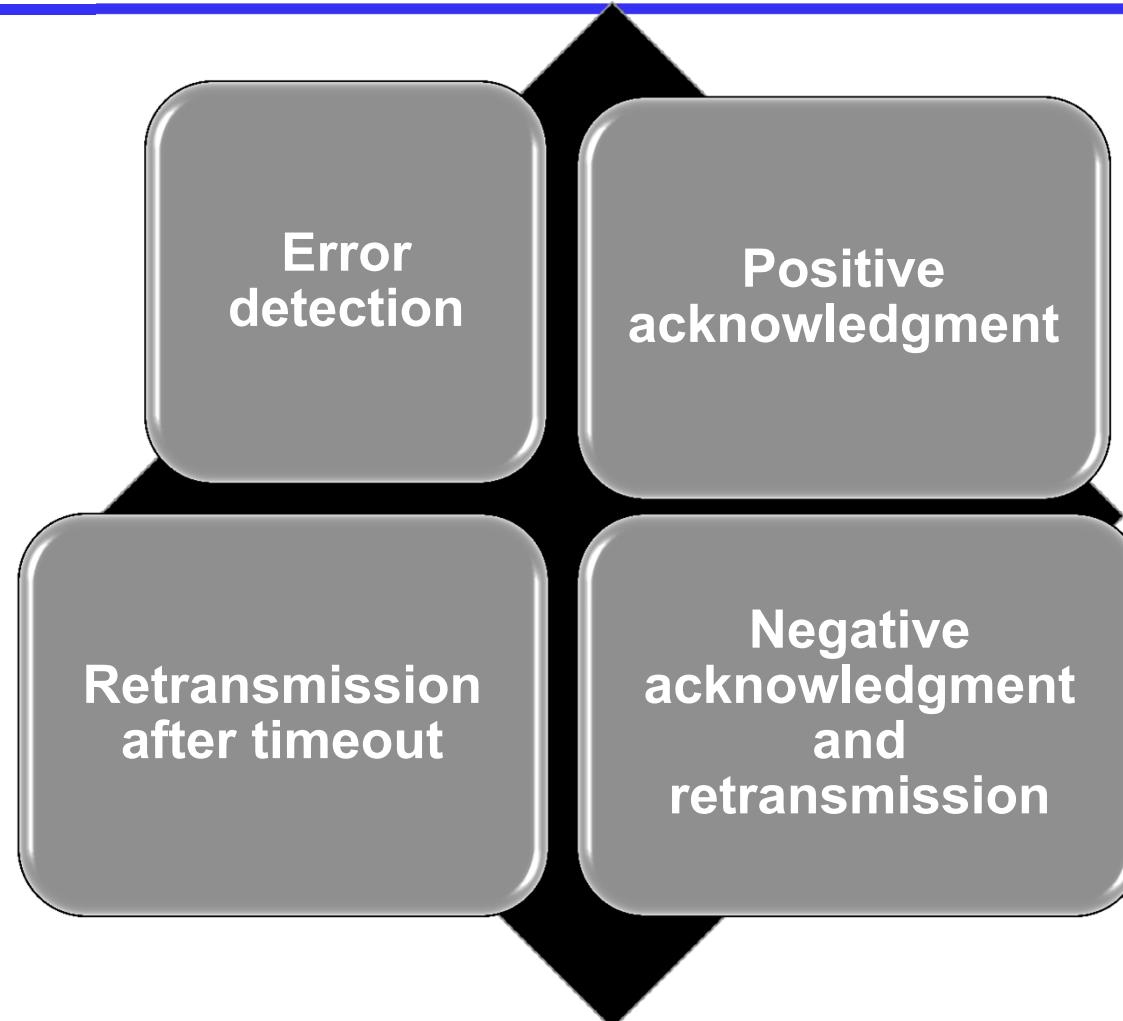
$$T_{out \text{ minimum}} = t_e + t_p + t_{acu} + t_p$$

$$\# \text{ FINESTRA optim} = \frac{T_{out}}{t_e}$$

Finestra: # tramas pendientes de confirmar Max  $2^k - 1$  (Go-Break-N)



# Error Control Techniques



## Lost frames

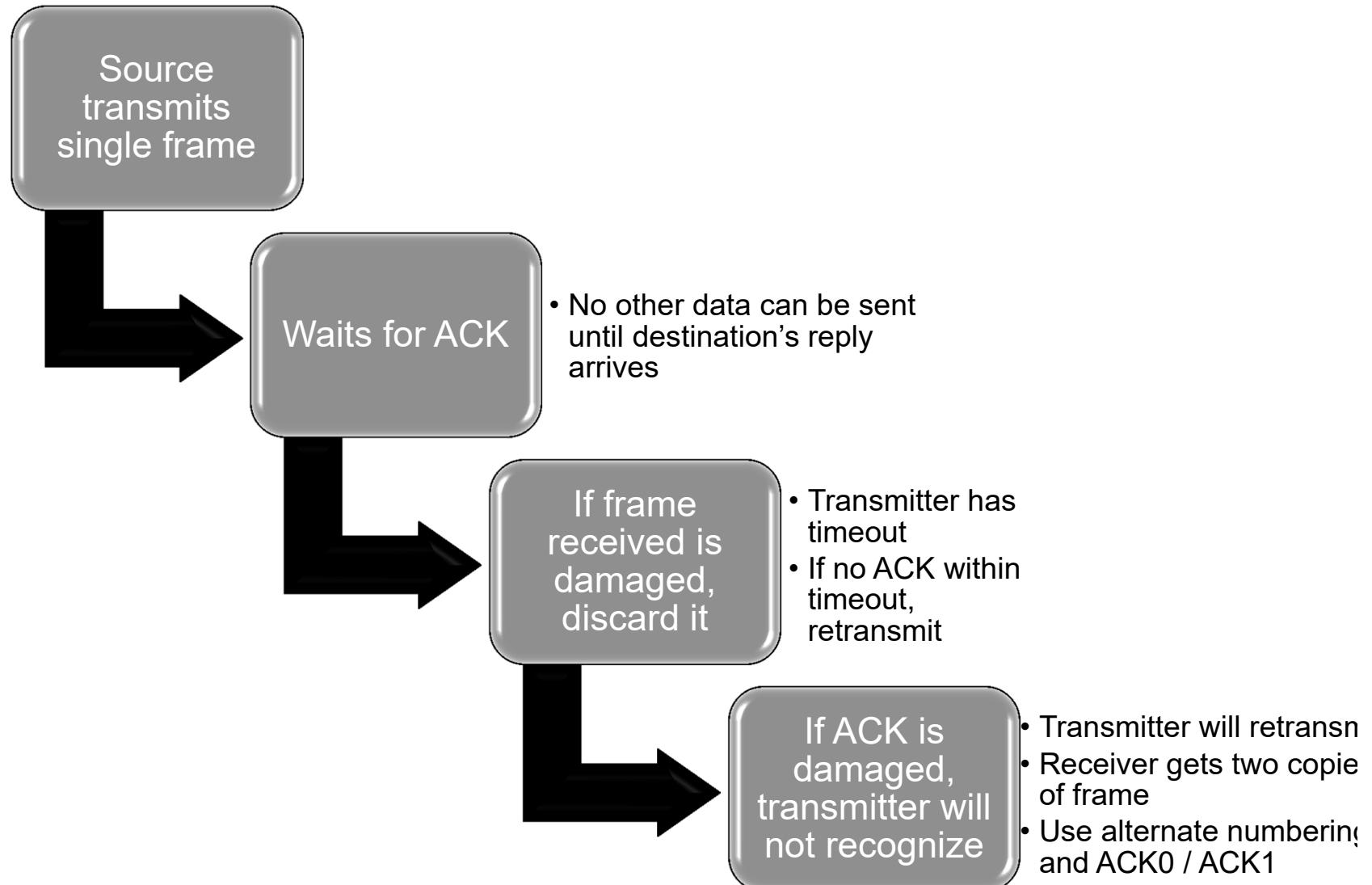
- a frame fails

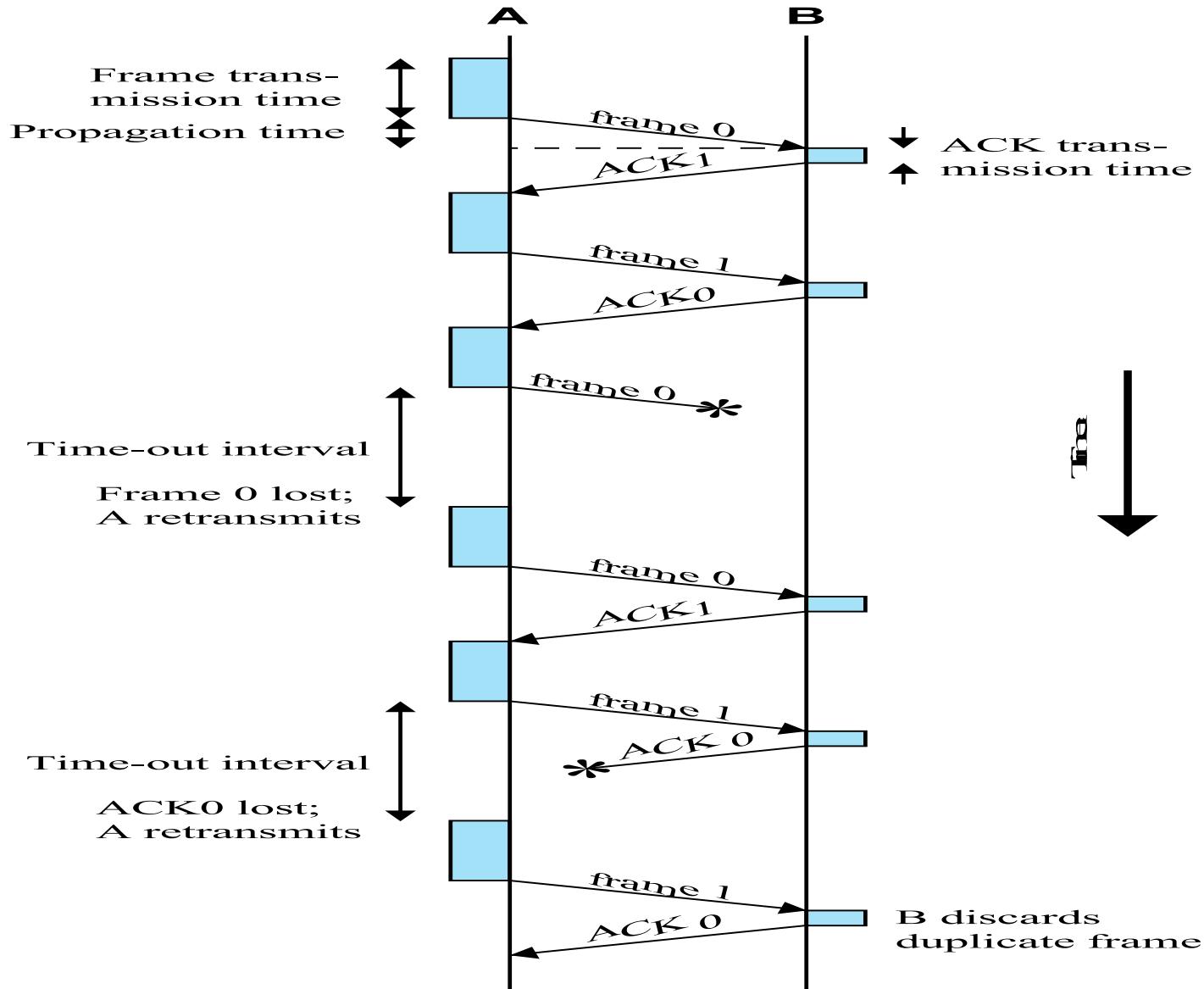
to arrive at the other side

## Damaged frames

- frame arrives but some of the bits are in error

# Stop and Wait ARQ





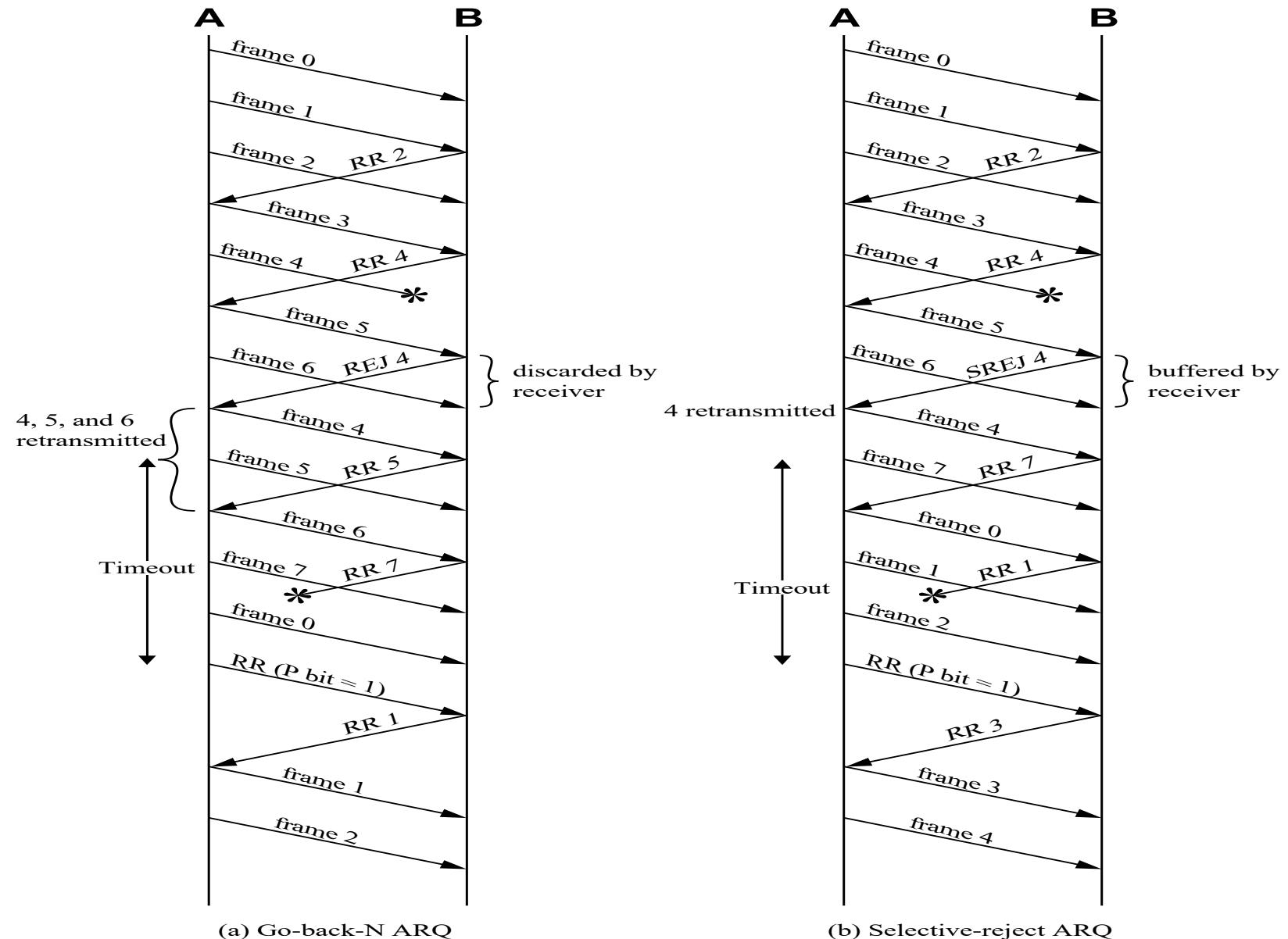
**Figure 7.5 Stop-and-Wait ARQ**

# Go-Back-N ARQ

- Most commonly used error control
- Based on sliding-window
- Use window size to control number of outstanding frames
- While no errors occur, the destination will acknowledge incoming frames as usual
  - RR=receive ready, or piggybacked acknowledgment
- If the destination station detects an error in a frame, it may send a negative acknowledgment
  - REJ=reject
  - Destination will discard that frame and all future frames until the frame in error is received correctly
  - Transmitter must go back and retransmit that frame and all subsequent frames

# Selective-Reject (ARQ)

- Also called selective retransmission
- Only rejected frames are retransmitted
- Subsequent frames are accepted by the receiver and buffered
- Minimizes retransmission
- Receiver must maintain large enough buffer
- More complex logic in transmitter
  - Less widely used
- Useful for satellite links with long propagation delays


**Figure 7.6 Sliding-Window ARQ Protocols**

# **Logical Link Control (LLC)**

---

- Transmission of link level PDUs between stations
- Must support multi-access, shared medium
- Relieved of some details of link access by the MAC layer
- Addressing involves specifying source and destination LLC users
  - Referred to as service access points (SAPs)
  - Used in Access networks

# LLC Services

## Unacknowledged connectionless service

- Data-gram style service
- Delivery of data is not guaranteed

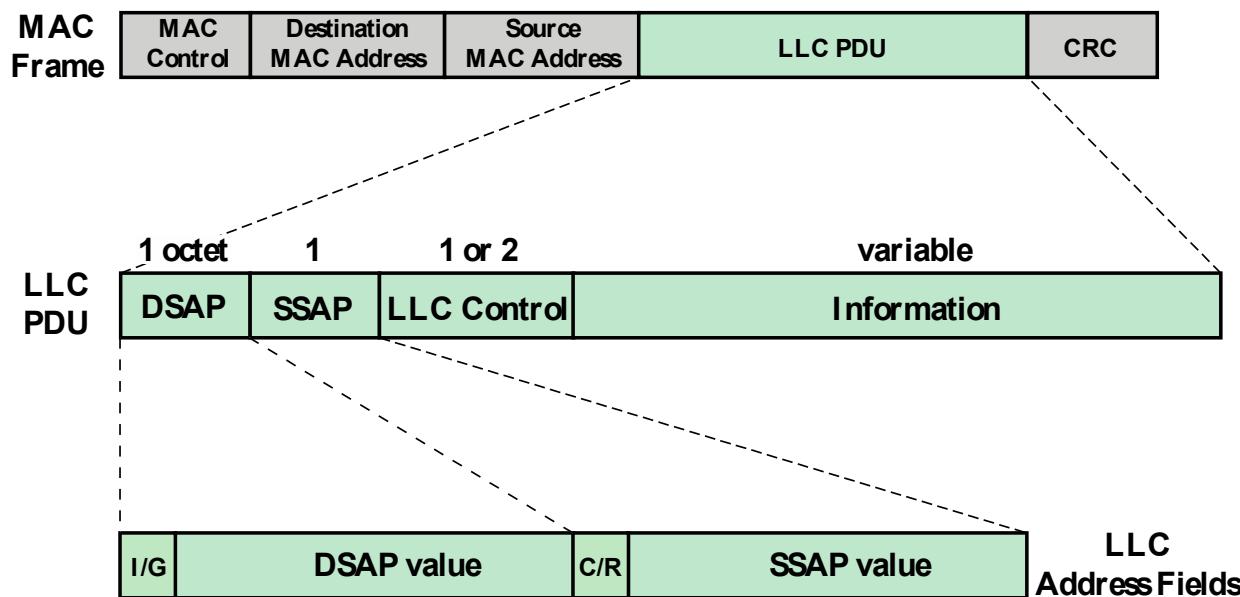
## Connection-mode service

- Logical connection is set up between two users
- Flow and error control are provided

## Acknowledged connectionless service

- Datagrams are to be acknowledged, but no logical connection is set up

# LLC in MAC Protocol



I/G = Individual/Group  
C/R = Command/Response

Figure 11.5 LLC PDU in a Generic MAC Frame Format

# LLC Control

- LLC takes advantage of the control field of the HDLC protocol
- Use of poll/final (P/F) bit depends on context
- In command frames P bit is set to 1 to solicit (poll) a response from the peer LLC entity
- In response frames F bit is set to 1 to indicate the response frame transmitted as a result of a soliciting command
- The basic control field for S- and I-frames uses 3 bit sequence numbers
  - An extended control field can be used that employs 7-bit sequence numbers
- U-frames always contain an 8-bit control field

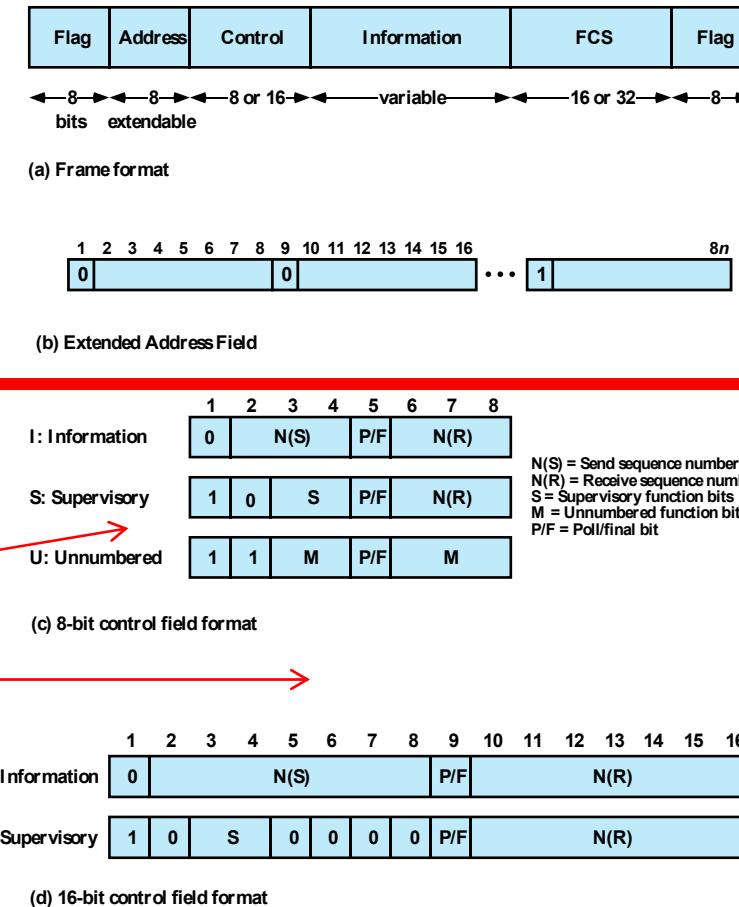


Figure 7.7 HDLC Frame Structure

Name	Command/ Response	Description
<b>Information (I)</b>	C/R	Exchange user data
<b>Supervisory (S)</b>		
Receive ready (RR)	C/R	Positive acknowledgment; ready to receive I-frame
Receive not ready (RNR)	C/R	Positive acknowledgment; not ready to receive
Reject (REJ)	C/R	Negative acknowledgment; go back N
Selective reject (SREJ)	C/R	Negative acknowledgment; selective reject
<b>Unnumbered (U)</b>		
Set normal response/extended mode (SNRM/SNRME)	C	Set mode; extended = 7-bit sequence numbers
Set asynchronous response/extended mode (SARM/SARME)	C	Set mode; extended = 7-bit sequence numbers
Set asynchronous balanced/extended mode (SABM, SABME)	C	Set mode; extended = 7-bit sequence numbers
Set initialization mode (SIM)	C	Initialize link control functions in addressed station
Disconnect (DISC)	C	Terminate logical link connection
Unnumbered Acknowledgment (UA)	R	Acknowledge acceptance of one of the set-mode commands
Disconnected mode (DM)	R	Responder is in disconnected mode
Request disconnect (RD)	R	Request for DISC command
Request initialization mode (RIM)	R	Initialization needed; request for SIM command
Unnumbered information (UI)	C/R	Used to exchange control information
Unnumbered poll (UP)	C	Used to solicit control information
Reset (RSET)	C	Used for recovery; resets N(R), N(S)
Exchange identification (XID)	C/R	Used to request/report status
Test (TEST)	C/R	Exchange identical information fields for testing
Frame reject (FRMR)	R	Report receipt of unacceptable frame

# Commands and Responses

(Table can be found on page 230 in the textbook)

# Exemples

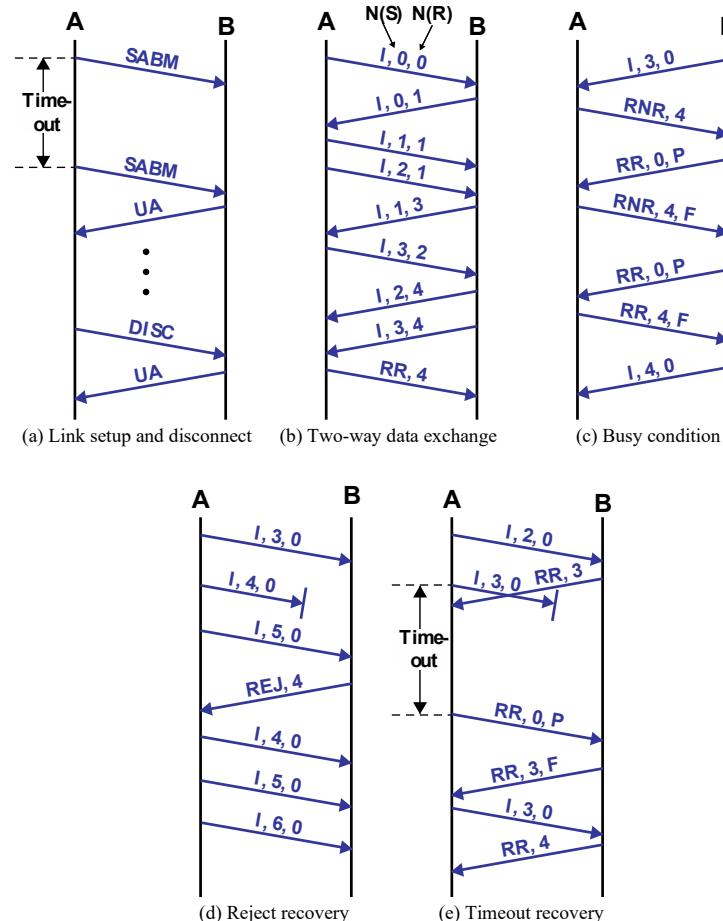


Figure 7.9 Examples of HDLC Operation

# Ethernet MAC Frame format.

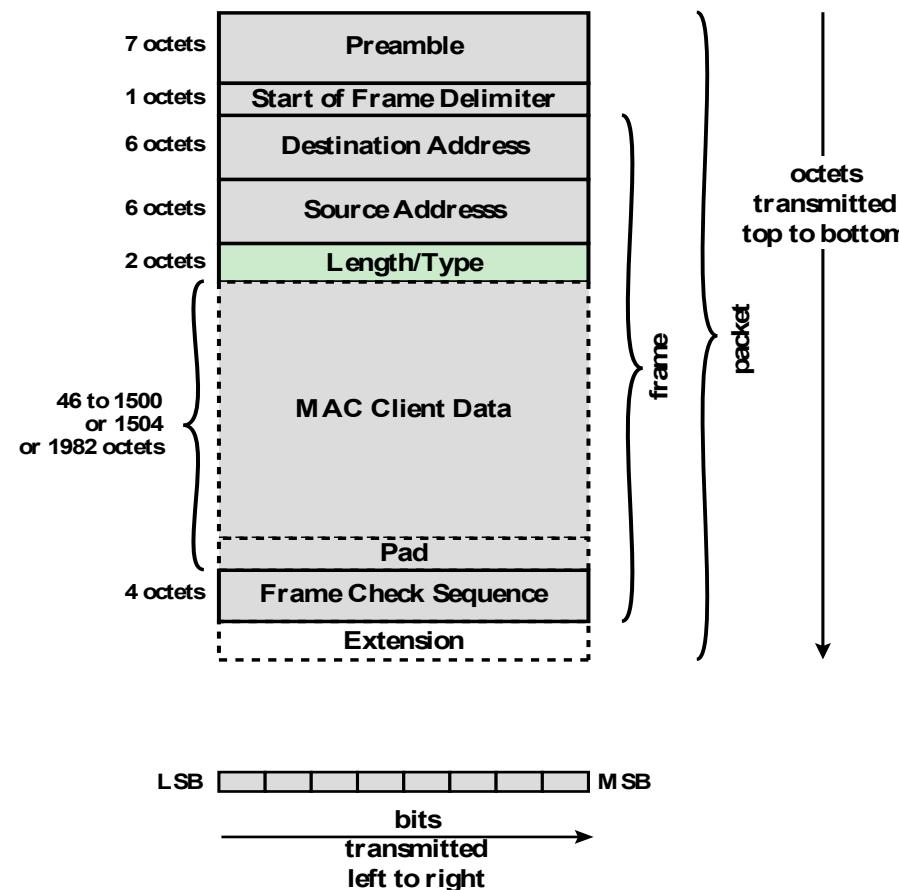


Figure 12.4 IEEE 802.3 MAC Frame Format

# Gigabit Ethernet Medium Options

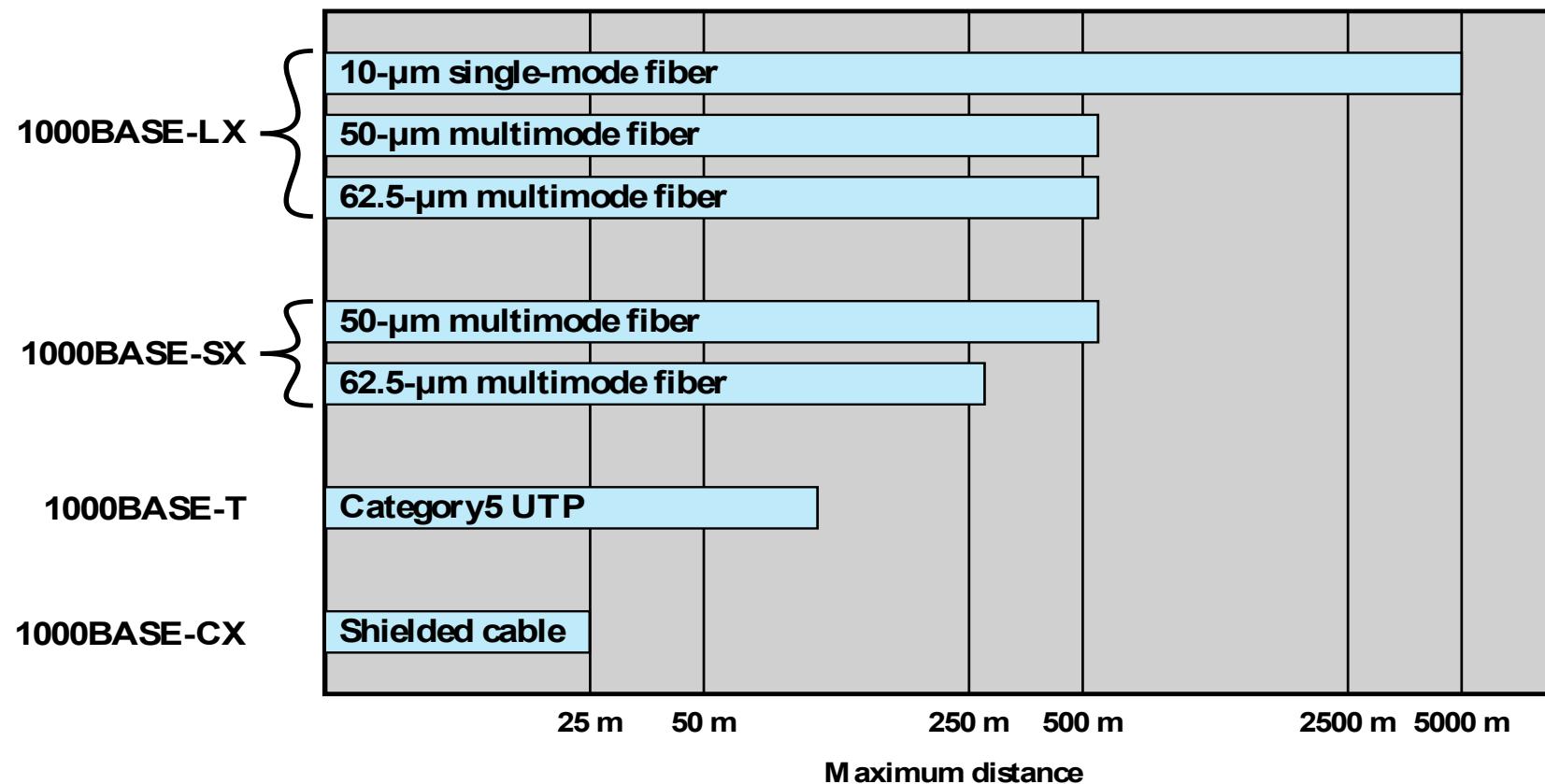


Figure 12.5 Gigabit Ethernet Medium Options (log scale)

# 10 Gbps Ethernet Distance Options

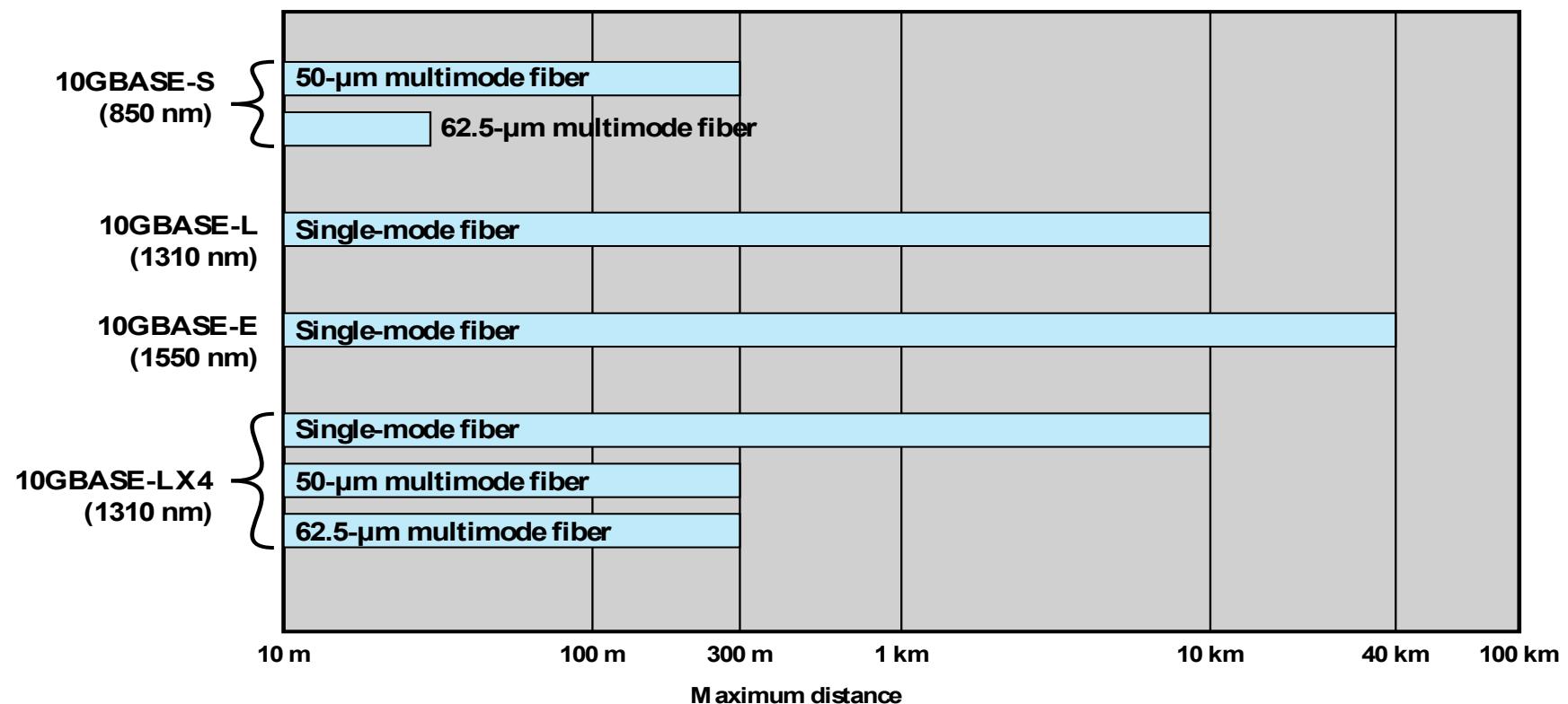


Figure 12.7 10-Gbps Ethernet Distance Options (log scale)

# Media Options for 40-Gbps and 100-Gbps Ethernet

	40 Gbps	100 Gbps
1m backplane	40GBASE-KR4	
10 m copper	40GBASE-CR4	1000GBASE-CR10
100 m multimode fiber	40GBASE-SR4	1000GBASE-SR10
10 km single mode fiber	40GBASE-LR4	1000GBASE-LR4
40 km single mode fiber		1000GBASE-ER4

Naming nomenclature:

Copper: K = backplane; C = cable assembly

Optical: S = short reach (100m); L - long reach (10 km); E = extended long reach (40 km)

Coding scheme: R = 64B/66B block coding

Final number: number of lanes (copper wires or fiber wavelengths)

# Tagged IEEE 802.3 MAC Frame Format

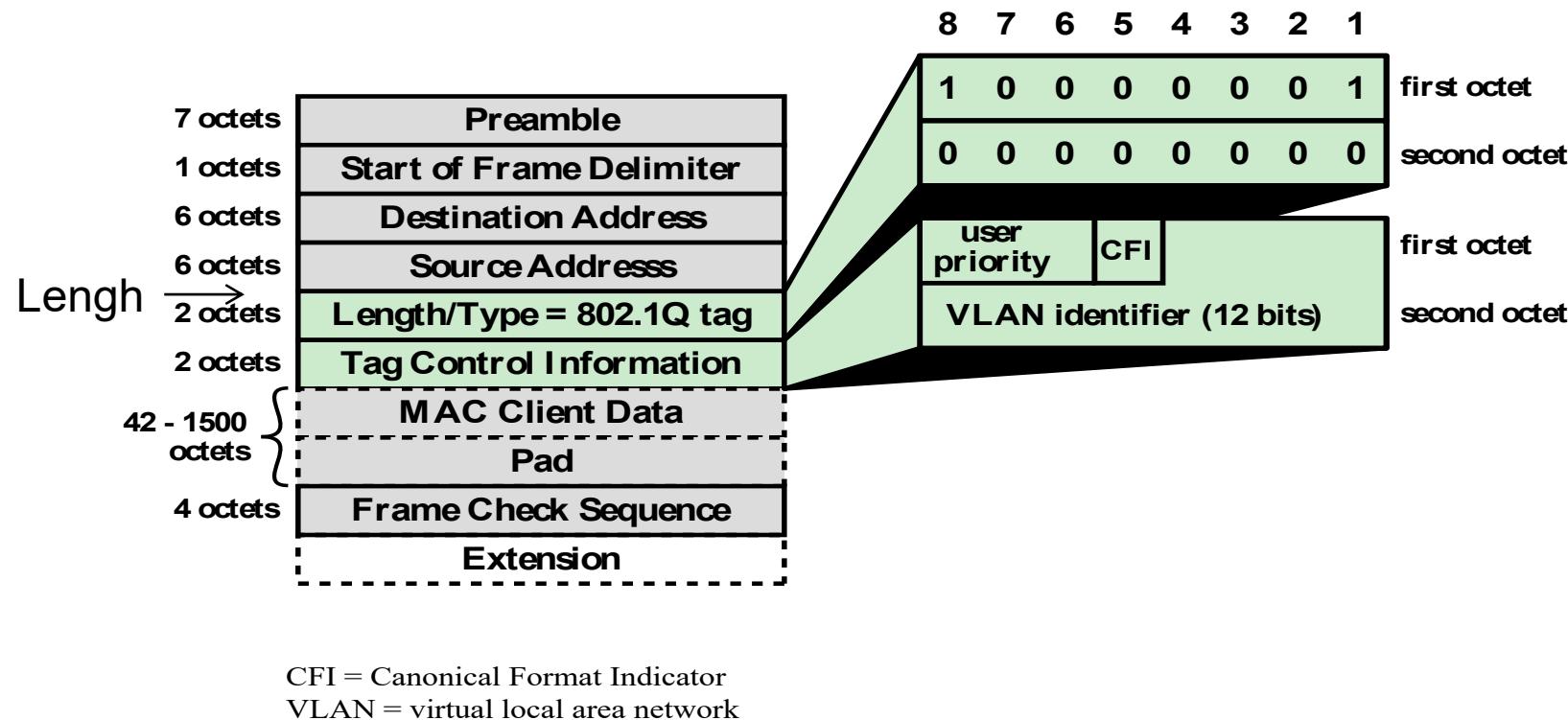
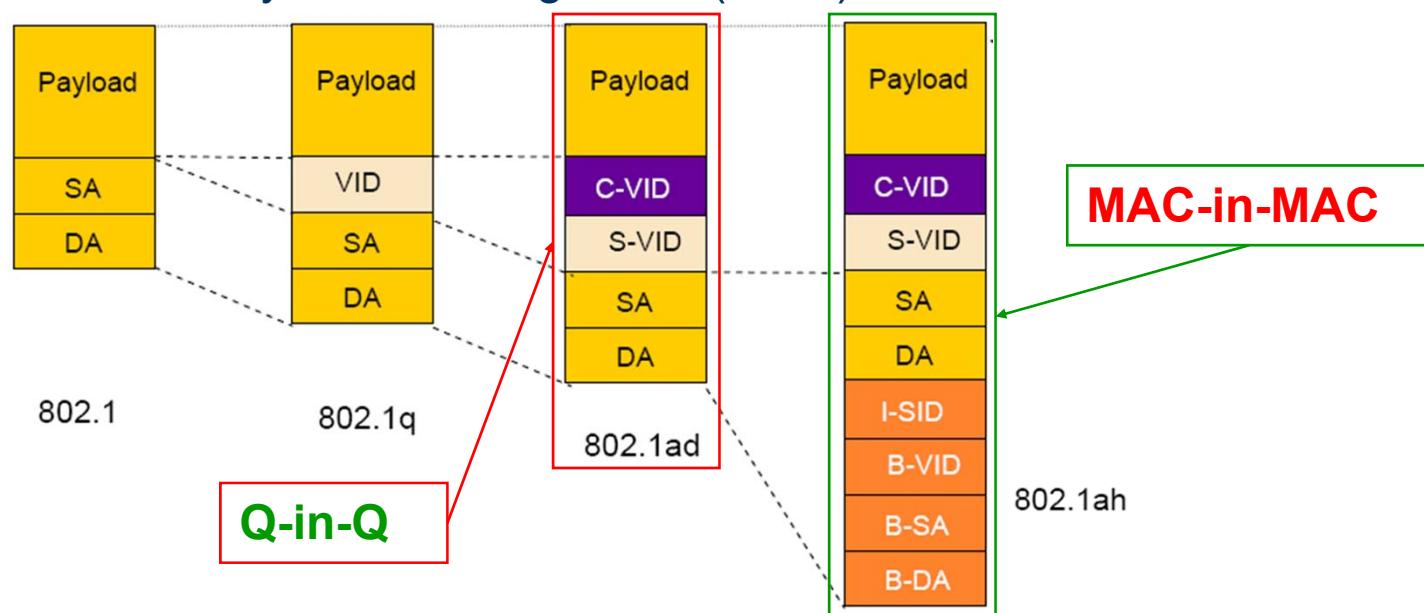


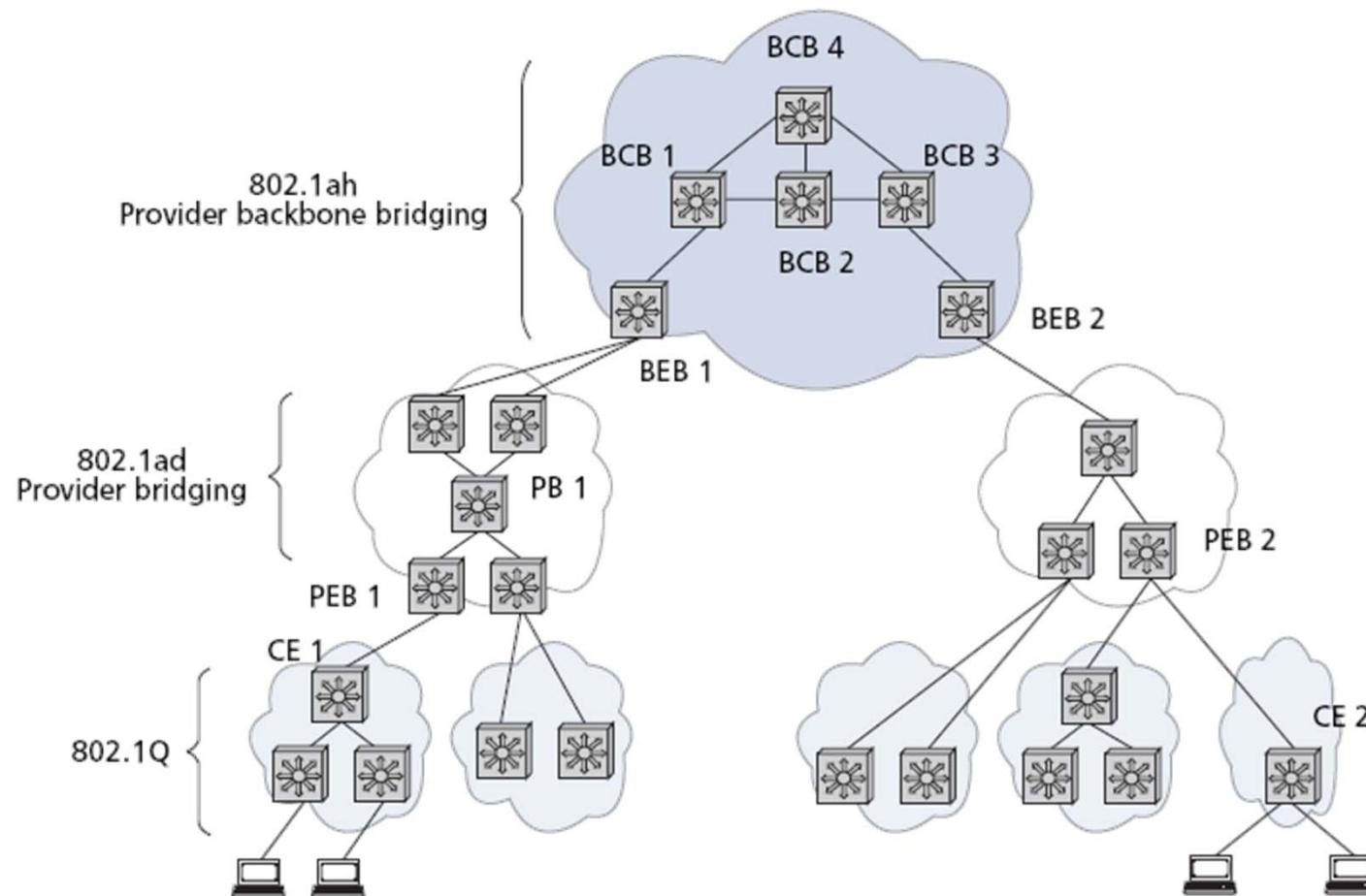
Figure 12.10 Tagged IEEE 802.3 MAC Frame Format

# Provider Backbone Bridge Traffic Engineering (PBB-TE)

- IEEE has developed a number of standards providing enhancements to the original Ethernet standards. PBB-TE adapts Ethernet technology to carrier class transport networks
  - 802.1Q: Virtual LAN
  - 802.1ad: Provider Bridging
  - 802.1ah: Provider Backbone Bridging
  - 802.1ag: Connectivity Fault Management (OAM)



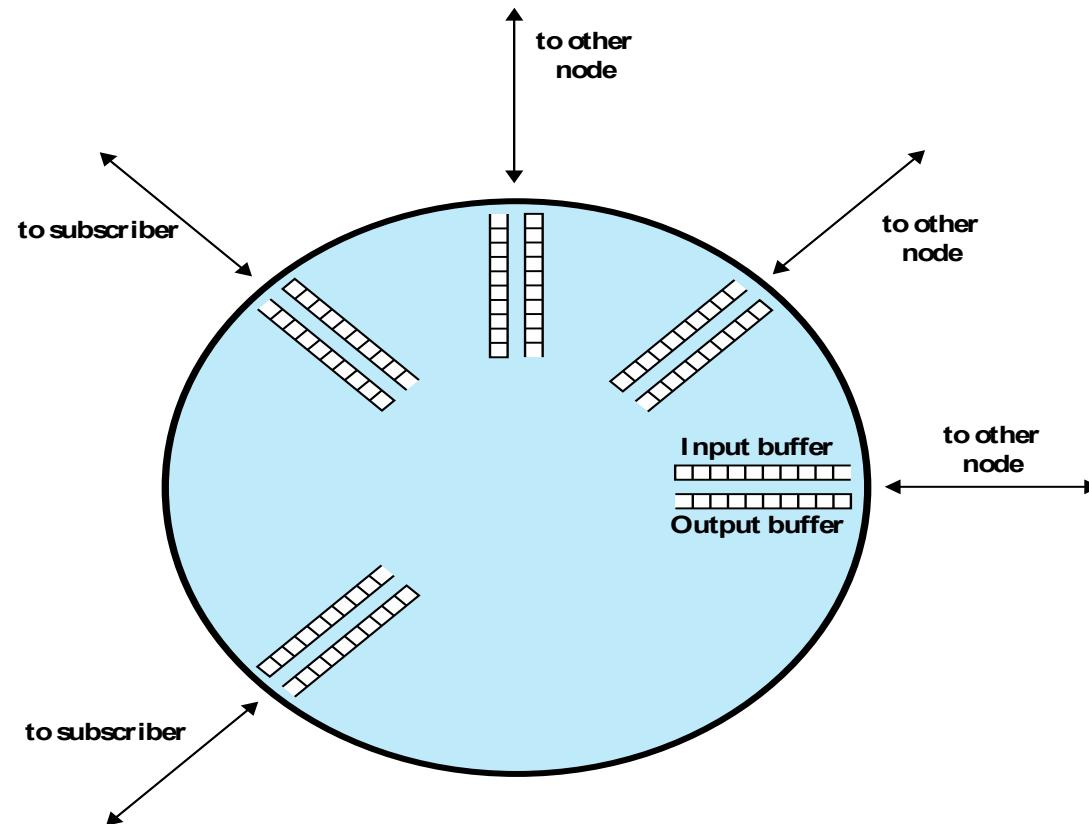
# The way to PBB-TE





## **3.2 Control de la congestió i QoS**

# Queues at Node



**Figure 20.1 Input and Output Queues at Node**

# *Interaction of Queues in a data network*

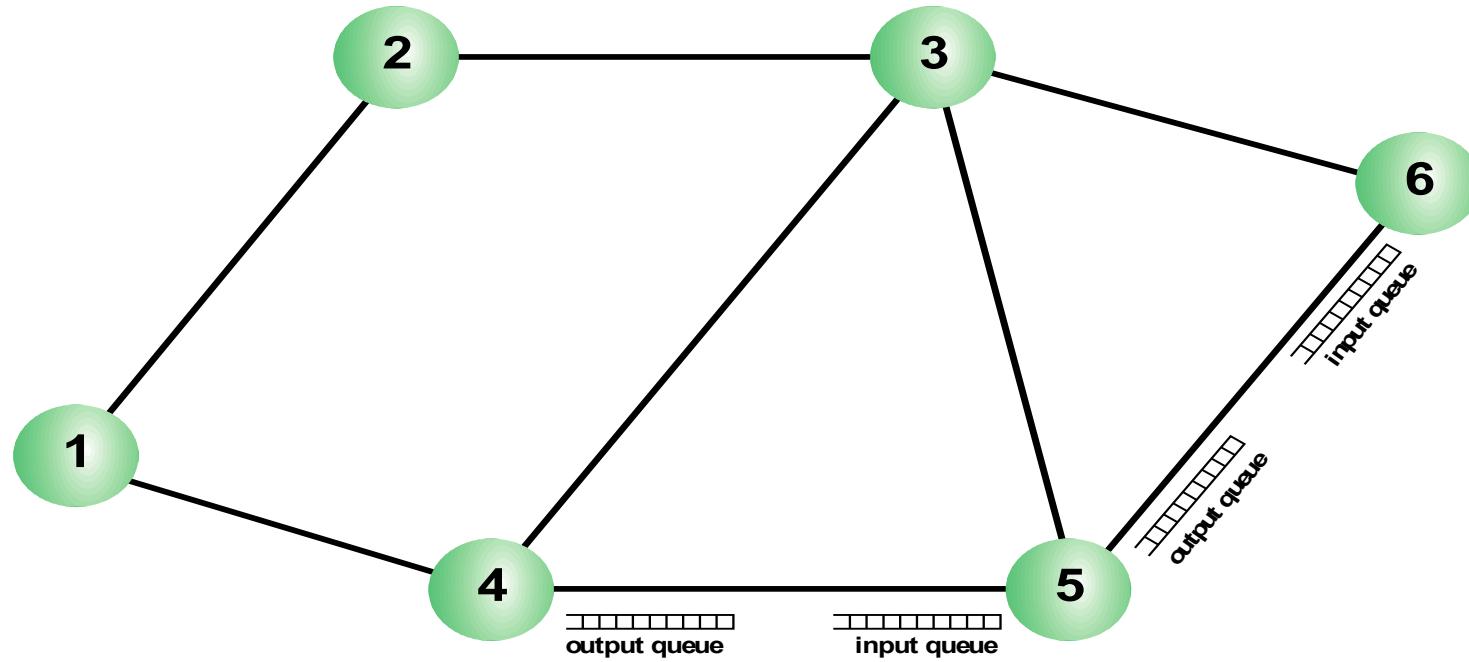
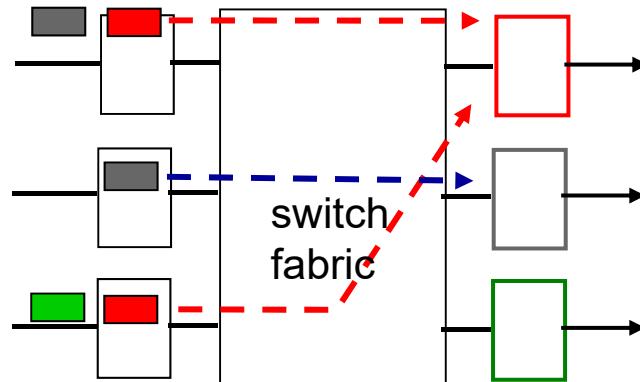


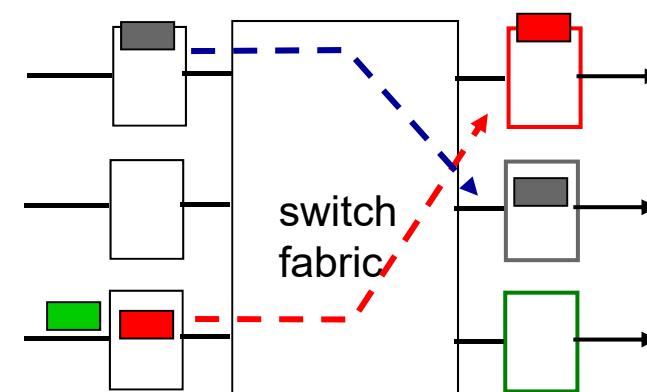
Figure 20.2 Interaction of Queues in a Data Network

# Input port queuing

- fabric slower than input ports combined -> queueing may occur at input queues
  - *queueing delay and loss due to input buffer overflow!*
- Head-of-the-Line (HOL) blocking: queued datagram at front of queue prevents others in queue from moving forward

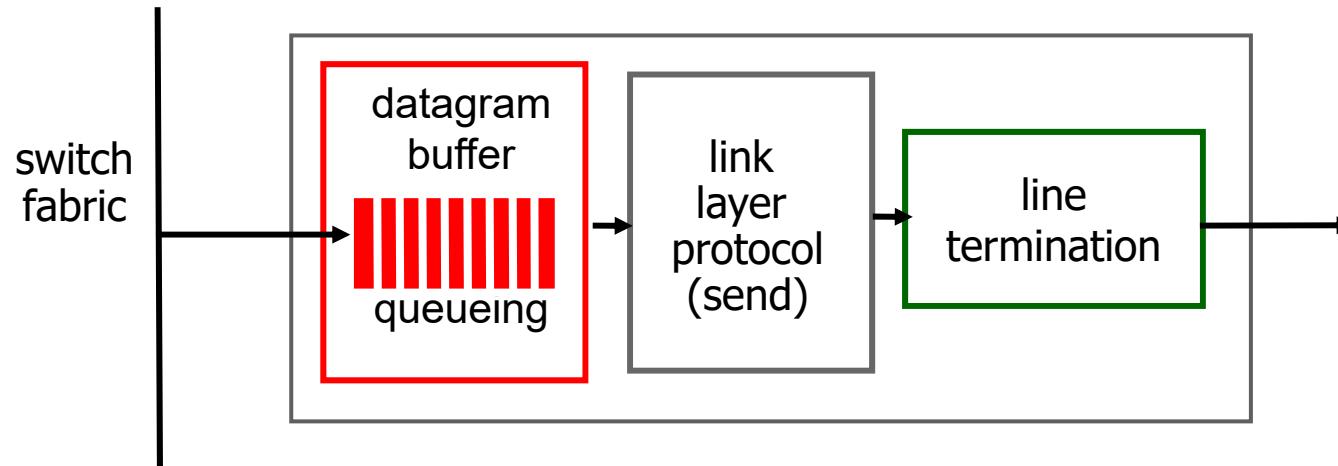


output port contention:  
only one red datagram can be  
transferred.  
*lower red packet is blocked*



one packet time later:  
green packet  
experiences HOL  
blocking

# Output ports



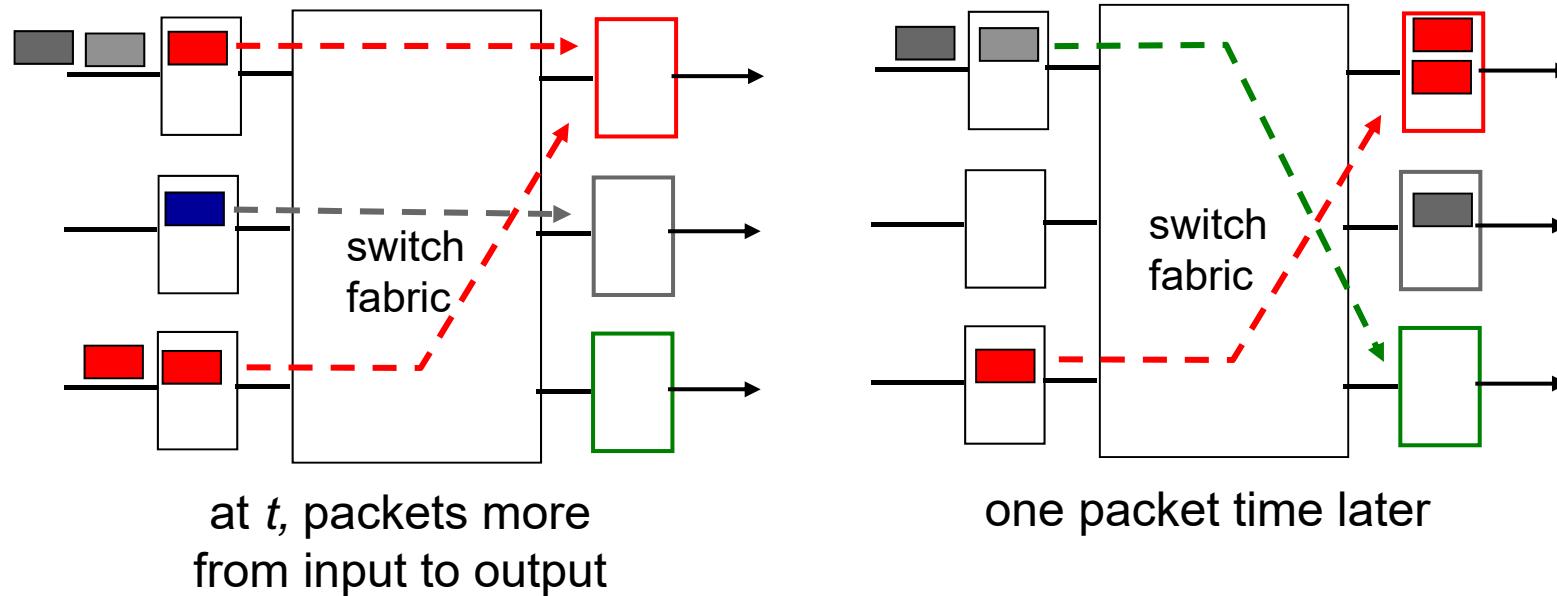
- *buffering* required when datagrams arrive from fabric faster than the transmission rate

Datagram (packets) can be lost due to congestion, lack of buffers

- *scheduling discipline* chooses among queued datagrams for transmission

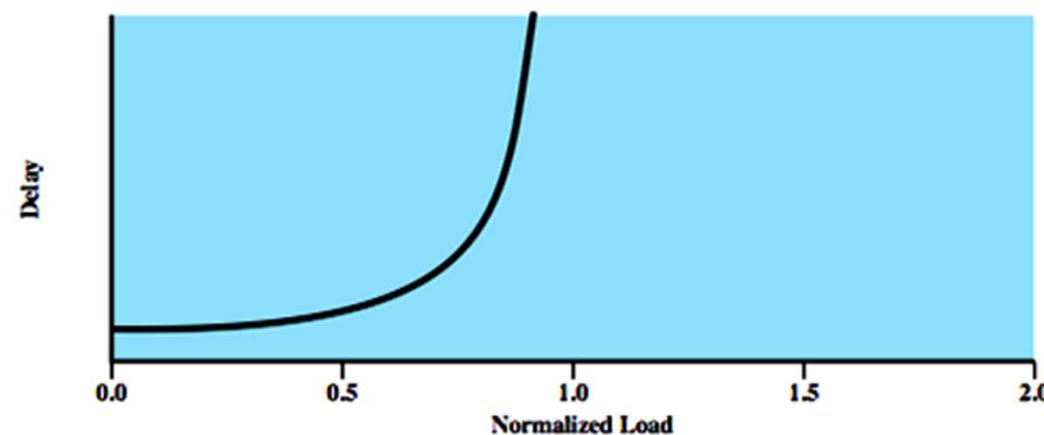
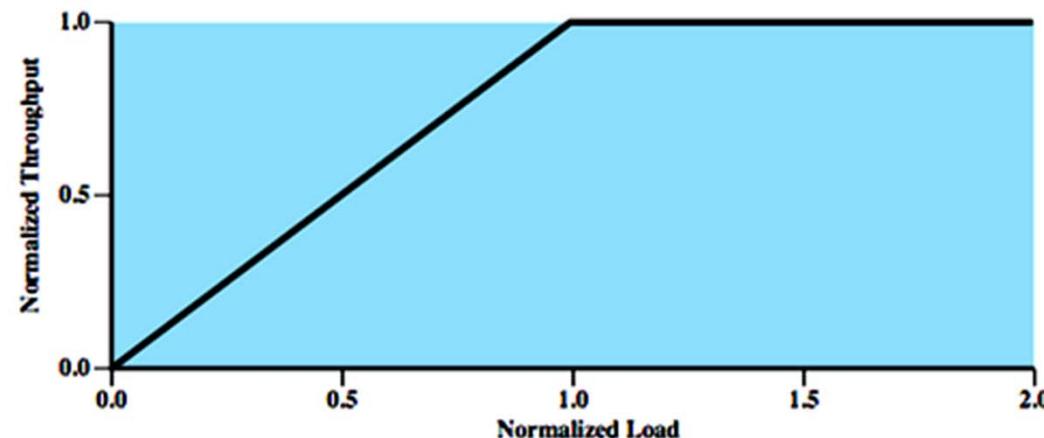
Priority scheduling – who gets best performance, network neutrality

# Output port queueing

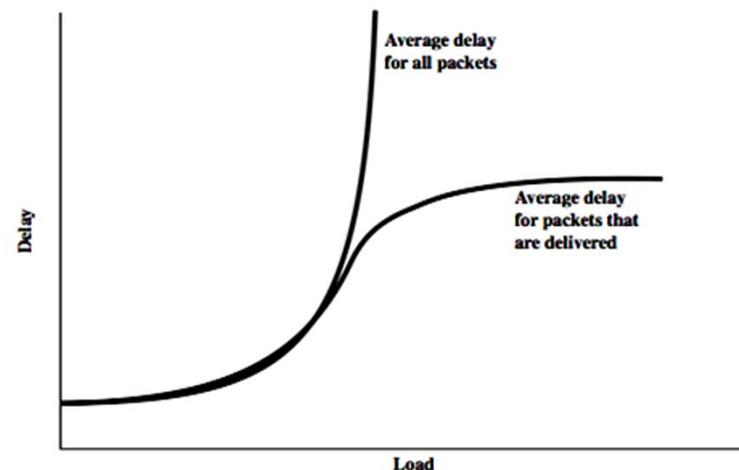
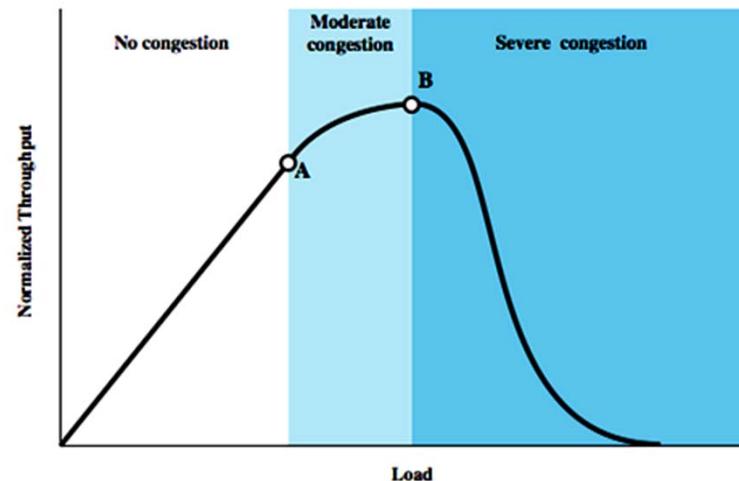


- buffering when arrival rate via switch exceeds output line speed
- *queueing (delay) and loss due to output port buffer overflow!*

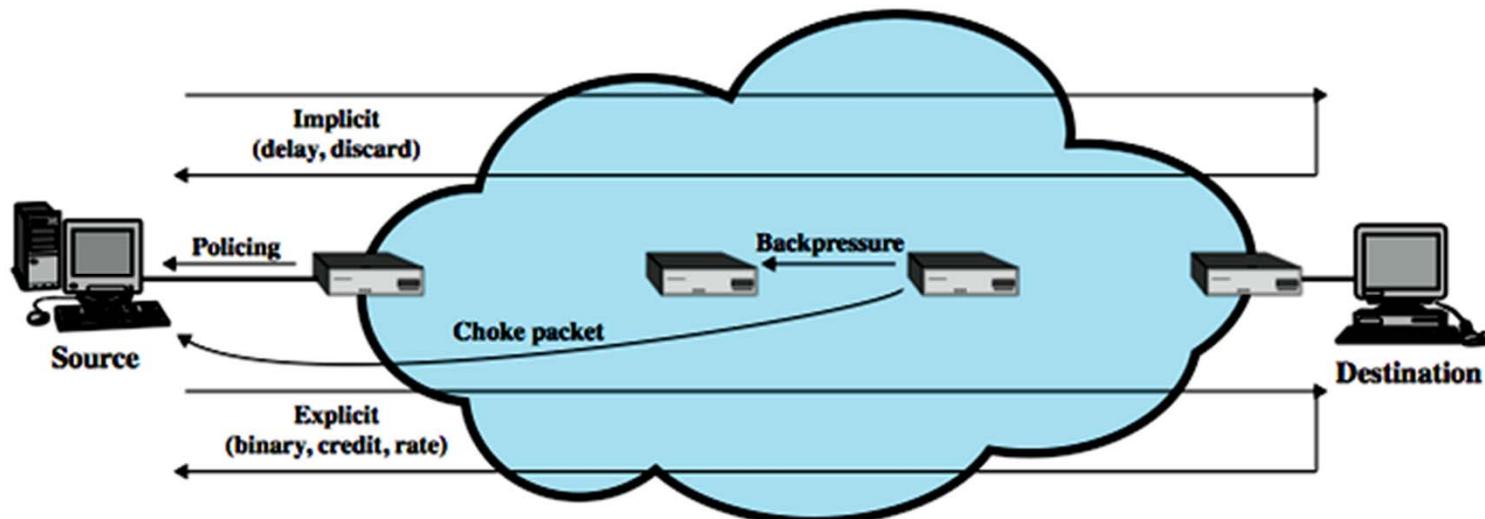
# Ideal Network Utilization



# Effects of Congestion - No Control



# Mechanisms for Congestion Control



# Traffic Management

## Fairness

- Provide equal treatment of various flows

## Quality of service

- Different treatment for different connections

## Reservations

- Traffic contract between user and network
- Excess traffic discarded or handled on a best-effort basis

# ***Type of applications***

---

- Interactive communications
  - Audio/video conferencing
- Information retrieval
  - Download of audio/video streams
- File transfer (data retrieval)
  - Access to databases and/or repositories
  - e-mail

# *Phenomena impacting in these applications*

---

- Lost of data
- End-to-end delay
- Delay jitter
- Throughput (Bandwidth)
- Network failures

# What is QoS?

---

- QoS is usually expressed as the combination of network-imposed features such as delay, delay jitter, bandwidth (throughput), packet loss, and reliability affecting the quality of the transmission
- Basically, there are two categories of QoS parameters
  - Technology based
    - *System parameters*
  - User Perception based,
    - *Also known as Quality of Experience (QoE)*

# QoS Parameters

- Required QoS can be defined by several parameters
  - **Delay:** how long it takes for a packet to traverse the network?
  - **Jitter:** what is the variance in the delay?
  - **Loss:** how often packets get lost in the network and never show up at the destination?
- Are the value of these parameters affordable?

# Service Categories

Real time - limit amount/variation of delay

- Constant bit rate (CBR)
- Real time variable bit rate (rt-VBR)

Non-real time - for bursty traffic

- Non-real time variable bit rate (nrt-VBR)
- Unspecified bit rate (UBR)

# Traffic Shaping/Traffic Policing

- Two important tools in network management:
  - Traffic shaping
    - Concerned with traffic leaving the switch
    - Reduces packet clumping
    - Produces an output packet stream that is less bursty and with a more regular flow of packets
  - Traffic policing
    - Concerned with traffic entering the switch
    - Packets that don't conform may be treated in one of the following ways:
      - Give the packet lower priority compared to packets in other output queues
      - Label the packet as nonconforming by setting the appropriate bits in a header
      - Discard the packet



# ***Token Bucket / Leaky Bucket***

---

- Widely used traffic management tool
- Advantages:
  - Many traffic sources can be defined easily and accurately
  - Provides a concise description of the load to be imposed by a flow, enabling the service to determine easily the resource requirement
  - Provides the input parameters to a policing function

# Token bucket

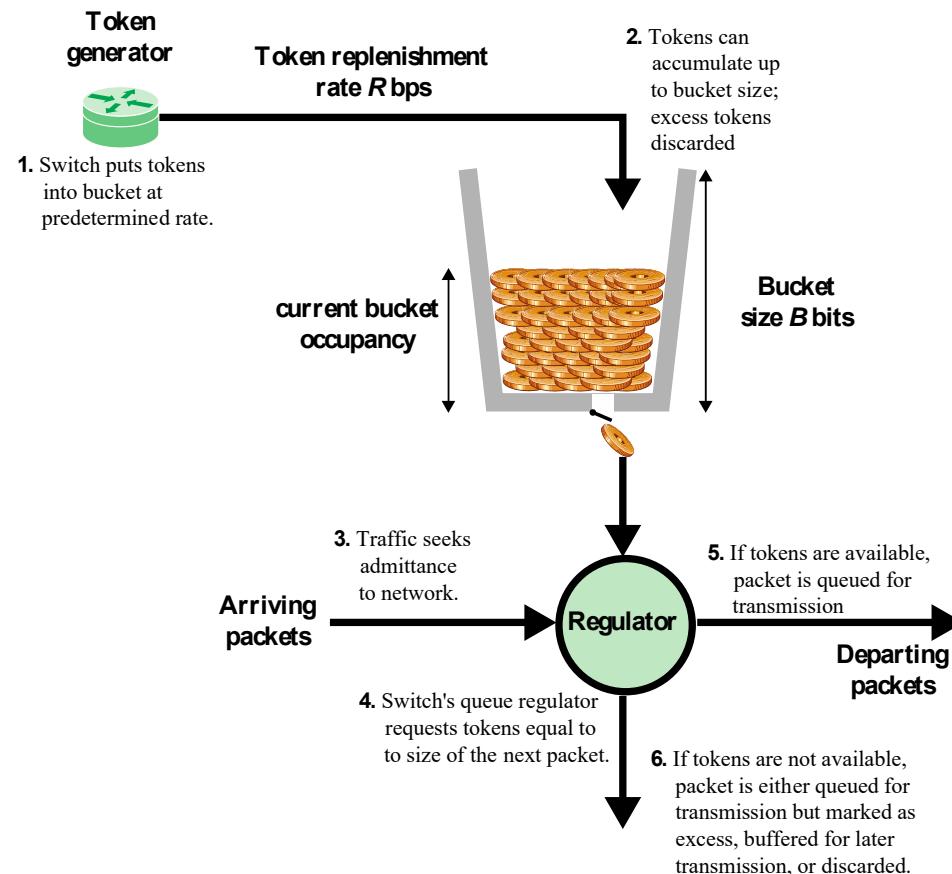


Figure 20.6 Token Bucket Scheme

# Leaky bucket

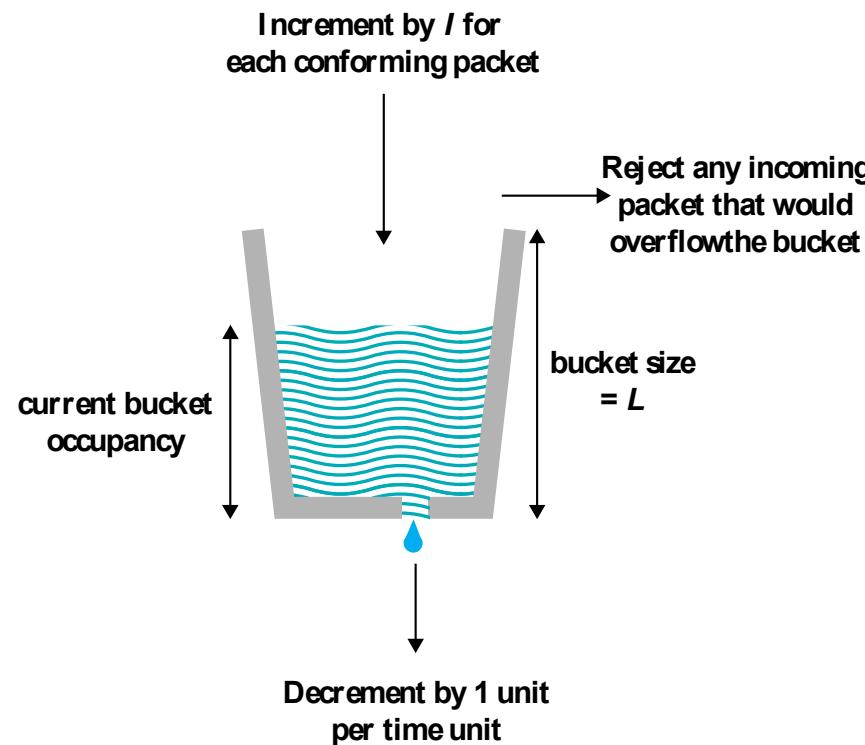
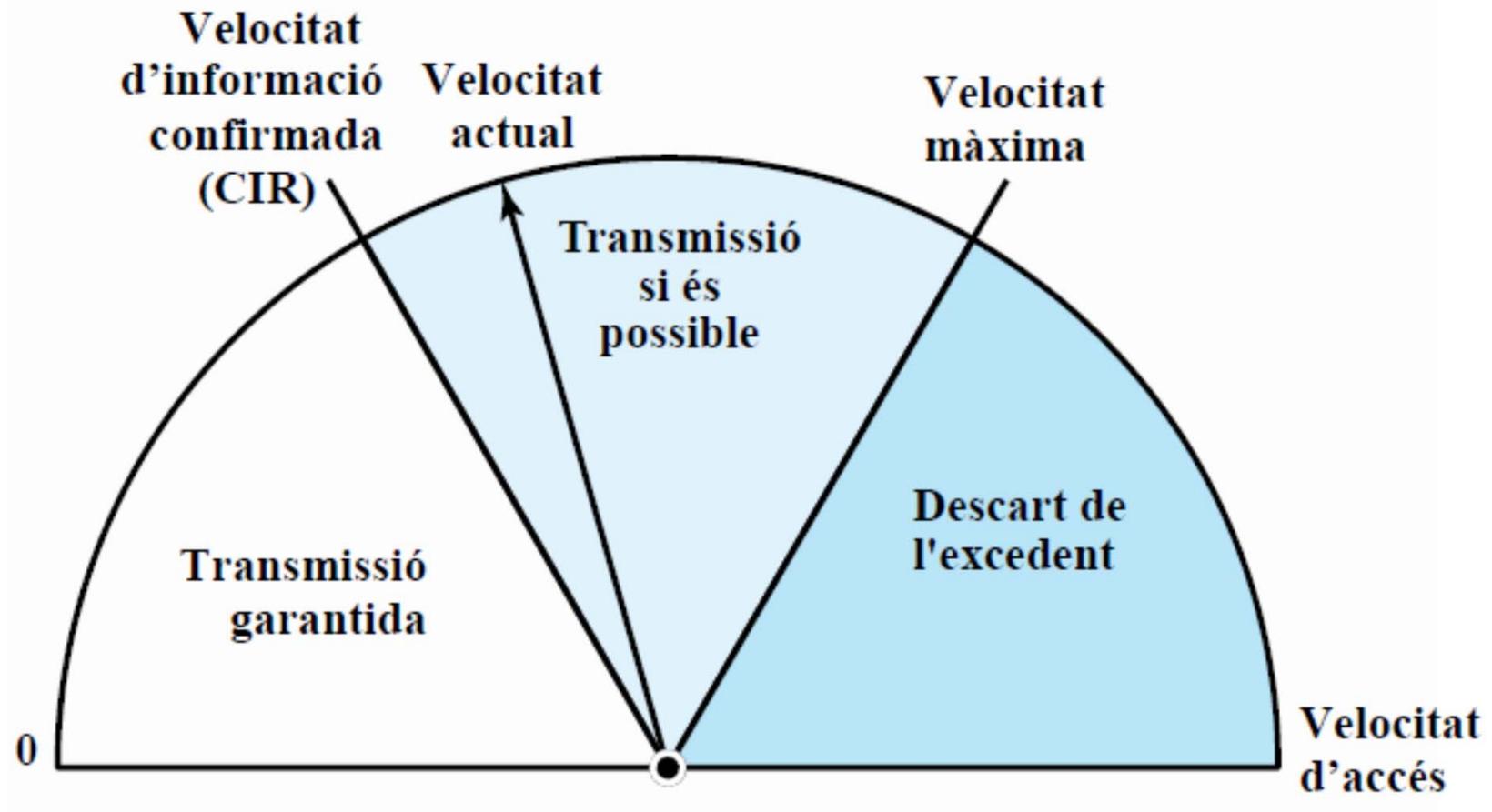
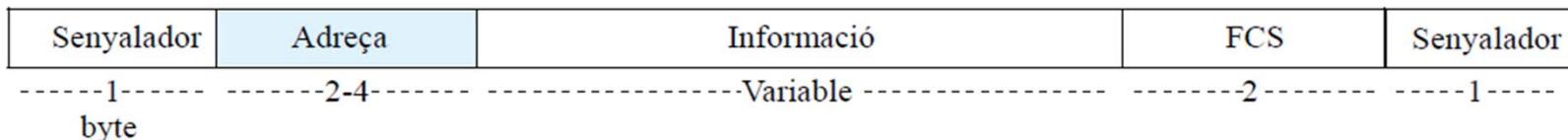


Figure 20.7 Leaky Bucket Algorithm

# CIR strategy



# Level 2. FR: Frame Relay: LAPF Core



(a) Format de trama

8	7	6	5	4	3	2	1
DLCI superior				C/R	EA 0		
DLCI inferior		FECN	BECN	DE	EA 1		

(b) Camp d'adreça: 2 bytes (per defecte)

8	7	6	5	4	3	2	1
DLCI superior				C/R	EA 0		
DLCI		FECN	BECN	DE	EA 0		
DLCI						EA 0	
Control DLCI inferior o DL-CORE				D/C	EA 1		

(d) Camp d'adreça: 4 bytes

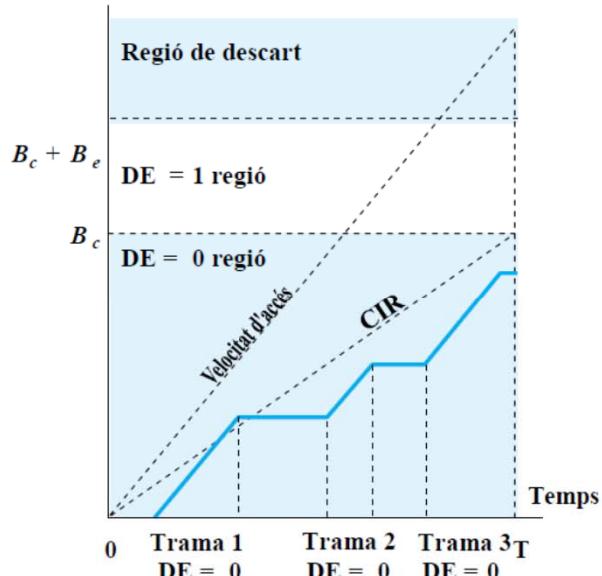
8	7	6	5	4	3	2	1
DLCI superior				C/R	EA 0		
DLCI		FECN BECN		DE	EA 0		
Control DLCI interior o DL-CORE				D/C	EA 1		

(c) Camp d'adreça: 3 bytes

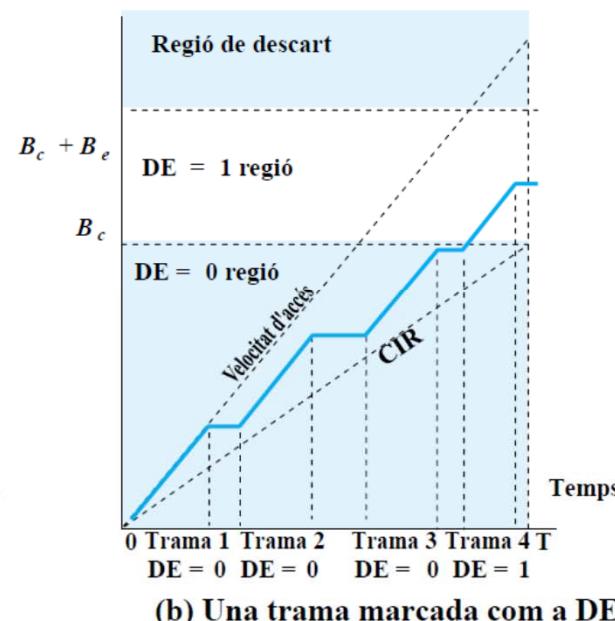
- EA Bit d'extensió de camp d'adreça
- C/R Bit d'ordre/resposta
- FECN Notificació de congestió explícita cap endavant
- BECN Notificació de congestió explícita cap enrere
- DLCI Identificador de connexió d'enllaç de dades
- D/C Indicador de control DLCI o DL-CORE
- DE Idoneïtat del descart

# Exemple Congestion control

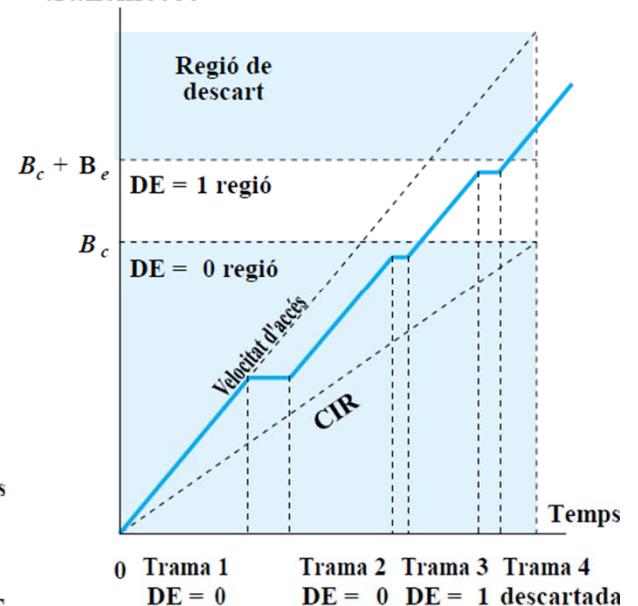
Nombre de bits transmesos



Nombre de bits transmesos



Nombre de bits transmesos



$$T = \frac{B_c}{CIR}$$



## ***3.3 Commutació d'etiquetes MPLS***

# Network-layer functions

*Recall: two network-layer functions:*

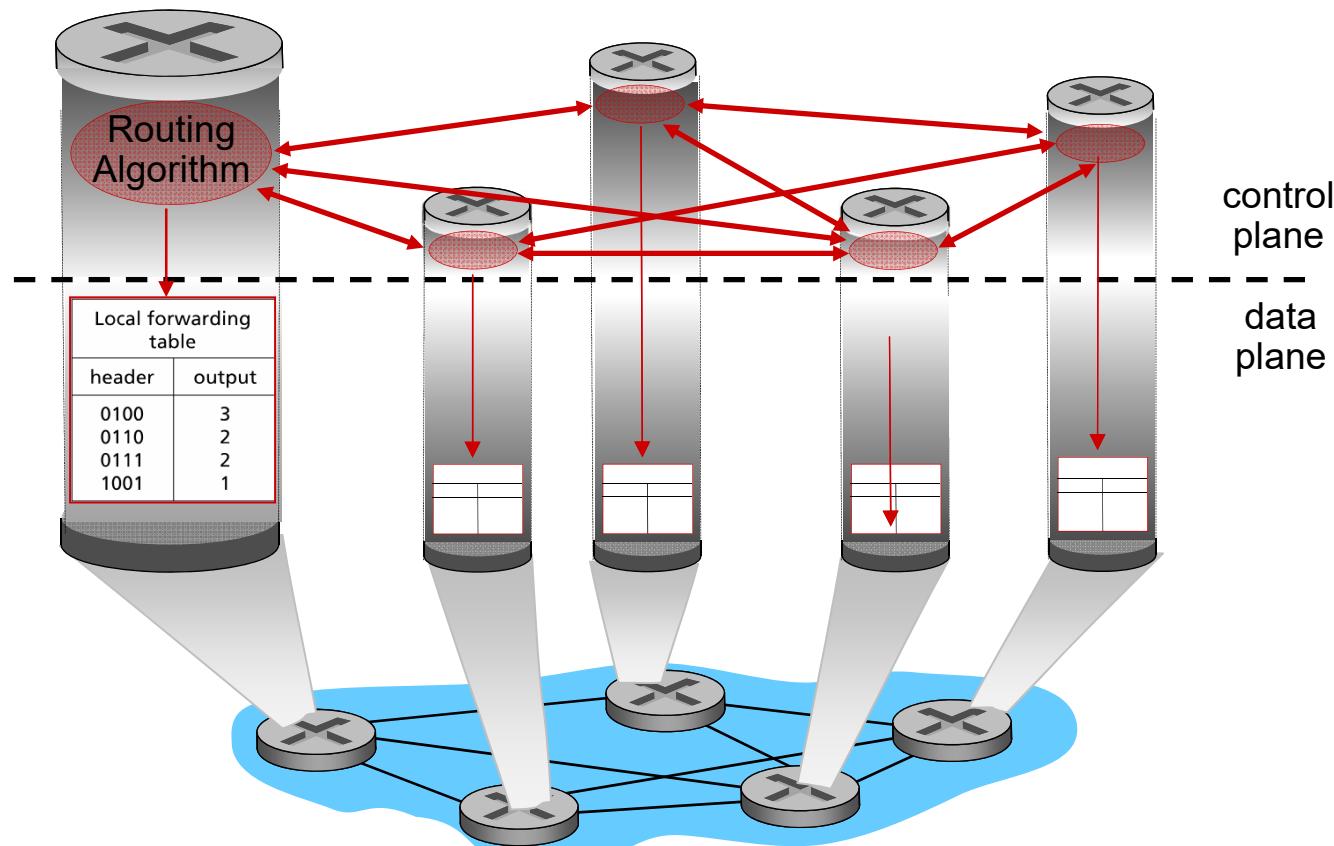
- *forwarding*: move packets from router's input to appropriate router output      ***data plane***
- *routing*: determine route taken by packets from source to destination      ***control plane***

*Two approaches to structuring network control plane:*

- per-router control (traditional)
- logically centralized control (software defined networking)

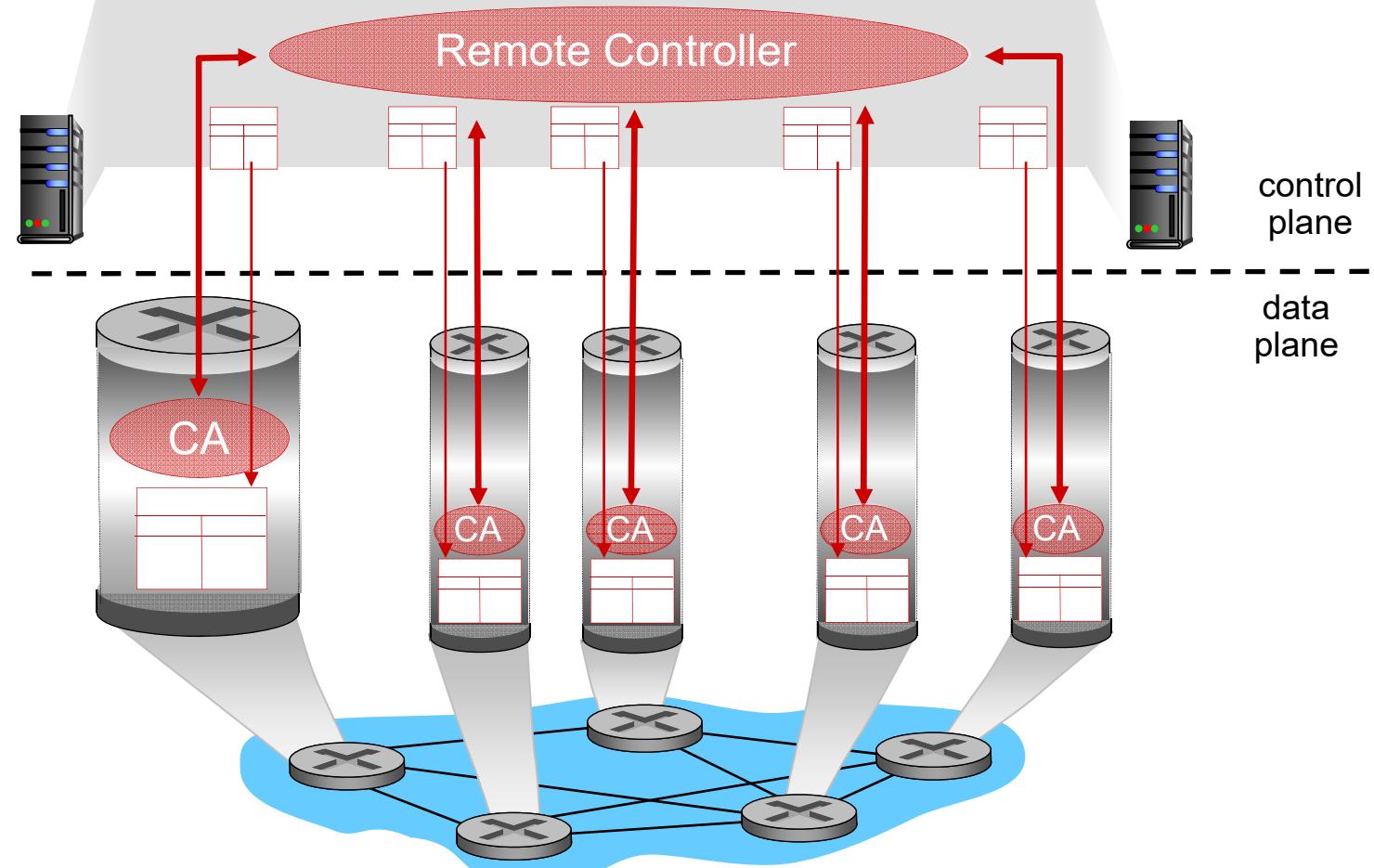
# Per-router control plane

Individual routing algorithm components *in each and every router* interact with each other in control plane to compute forwarding tables



# ***Logically centralized control plane***

A distinct (typically remote) controller interacts with local control agents (CAs) in routers to compute forwarding tables



# Role of MPLS

- Efficient technique for forwarding and routing packets
- Designed with IP networks in mind
  - *Can be used with any link-level protocol*
- Fixed-length label encapsulates an IP packet or a data link frame
- MPLS label contains all information needed to perform routing, delivery, QoS, and traffic management functions
- Is connection oriented

# Traffic Engineering

- Ability to define routes dynamically, plan resource commitments on the basis of known demand, and optimize network utilization
- Effective use can substantially increase usable network capacity
- ATM provided strong traffic engineering capabilities prior to MPLS
- With basic IP there is a primitive form

## MPLS

- Is aware of flows with QoS requirements
- Possible to set up routes on the basis of flows
- Paths can be rerouted intelligently

# MPLS Operation

---

- Label switching routers (LSRs)
  - *Nodes capable of switching and routing packets on the basis of label*
- Labels define a flow of packets between two endpoints
- Assignment of a particular packet is done when the packet enters the network of MPLS routers
- Connection-oriented technology

# MPLS Operation

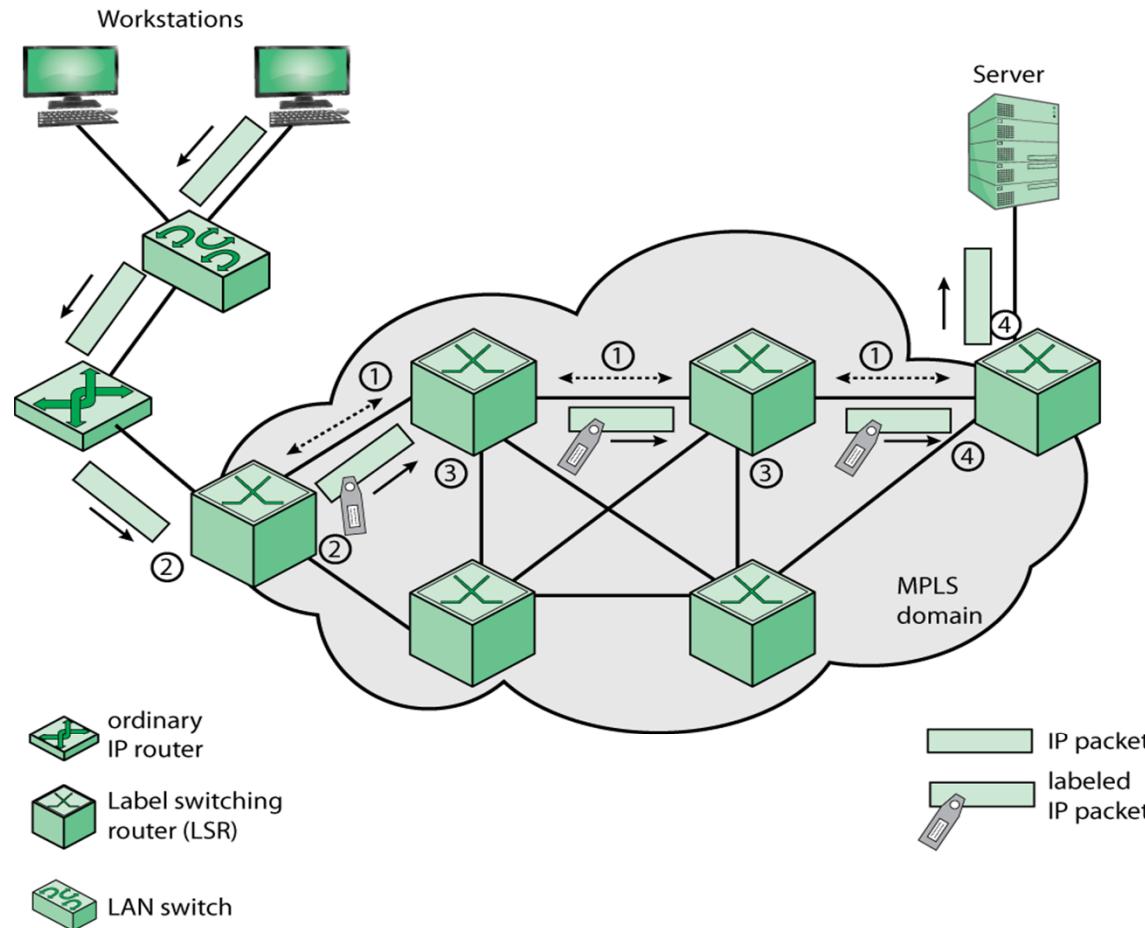


Figure 21.1 MPLS Operation

# MPLS Packet Forwarding

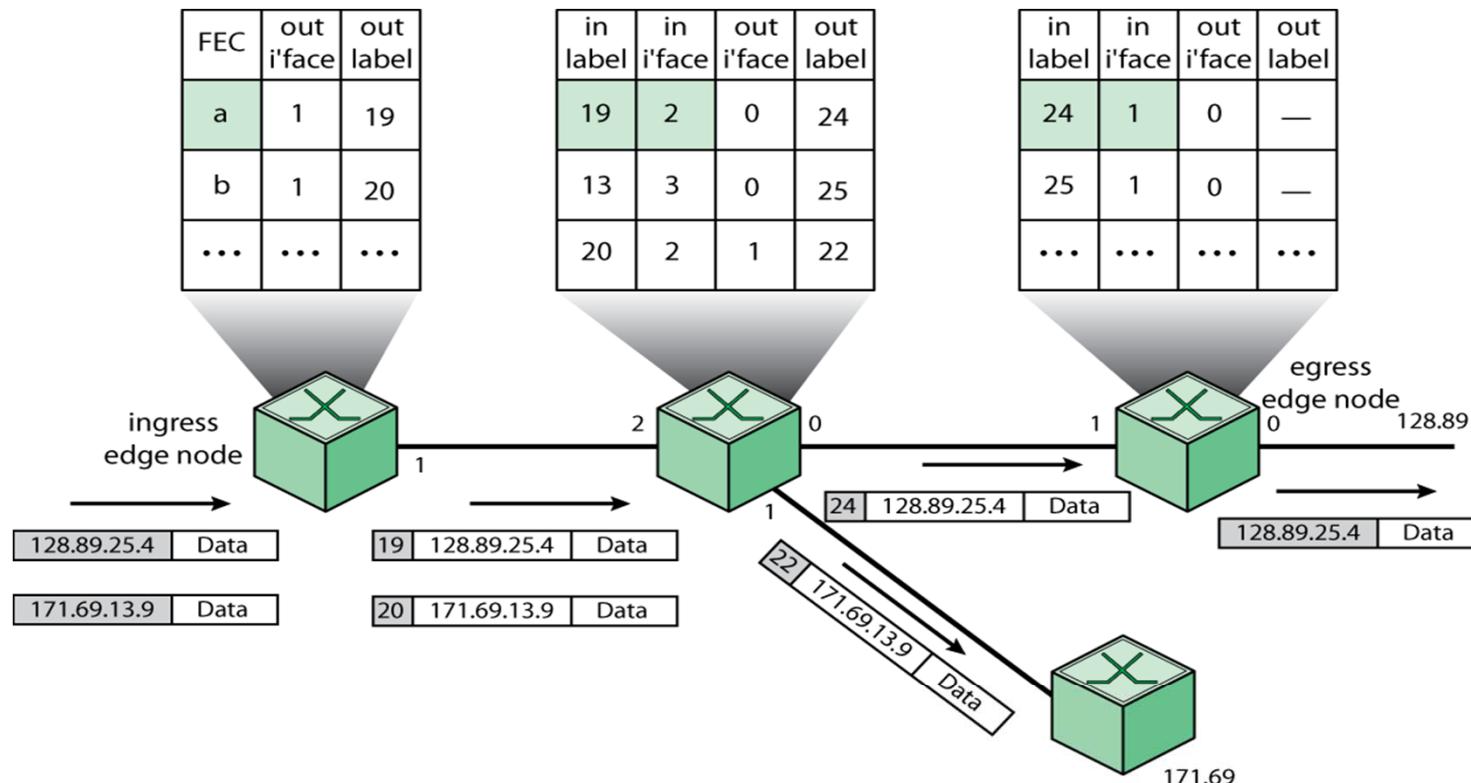


Figure 21.2 MPLS Packet Forwarding

# LSP Creation and Packet Forwarding

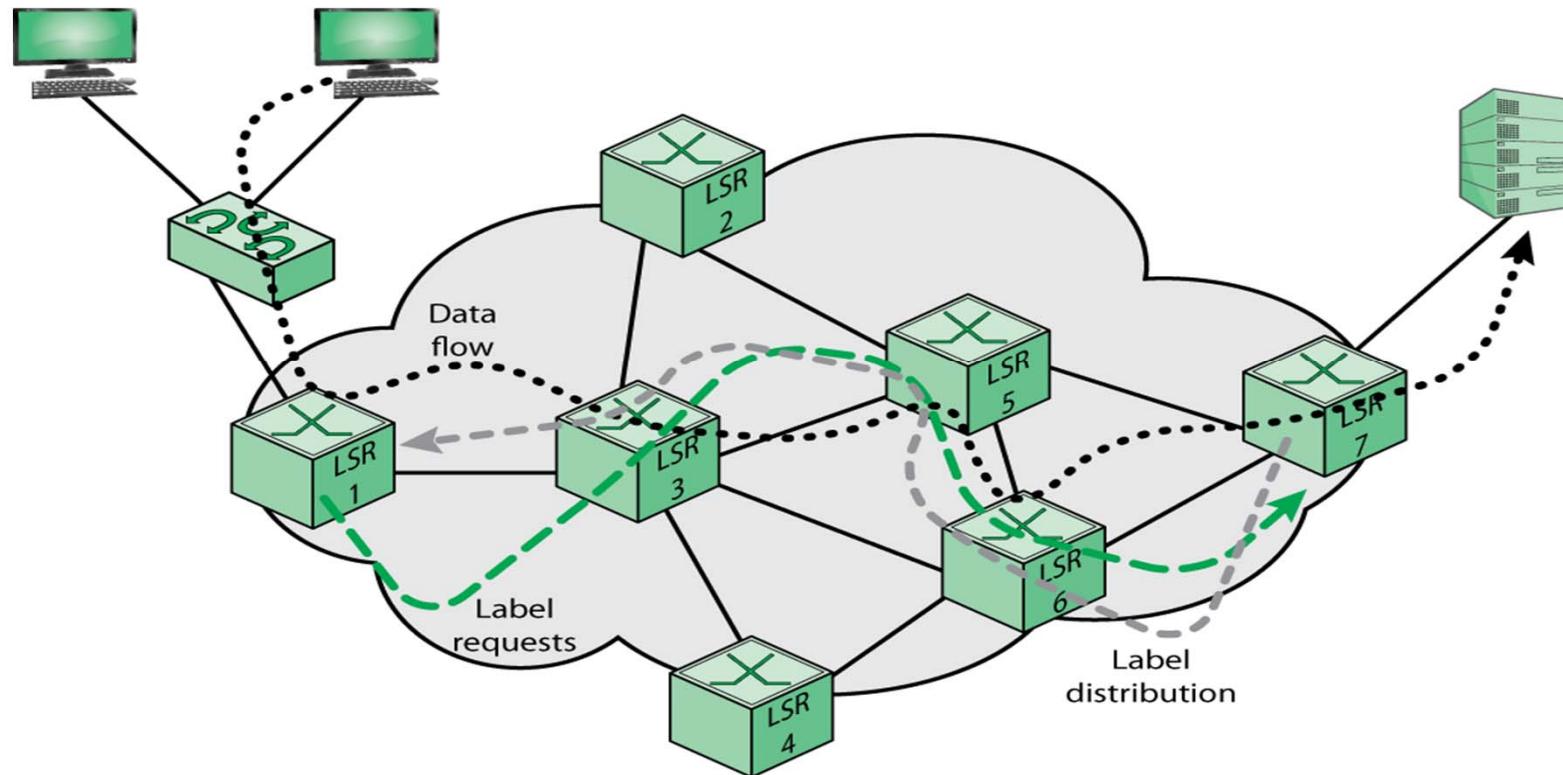


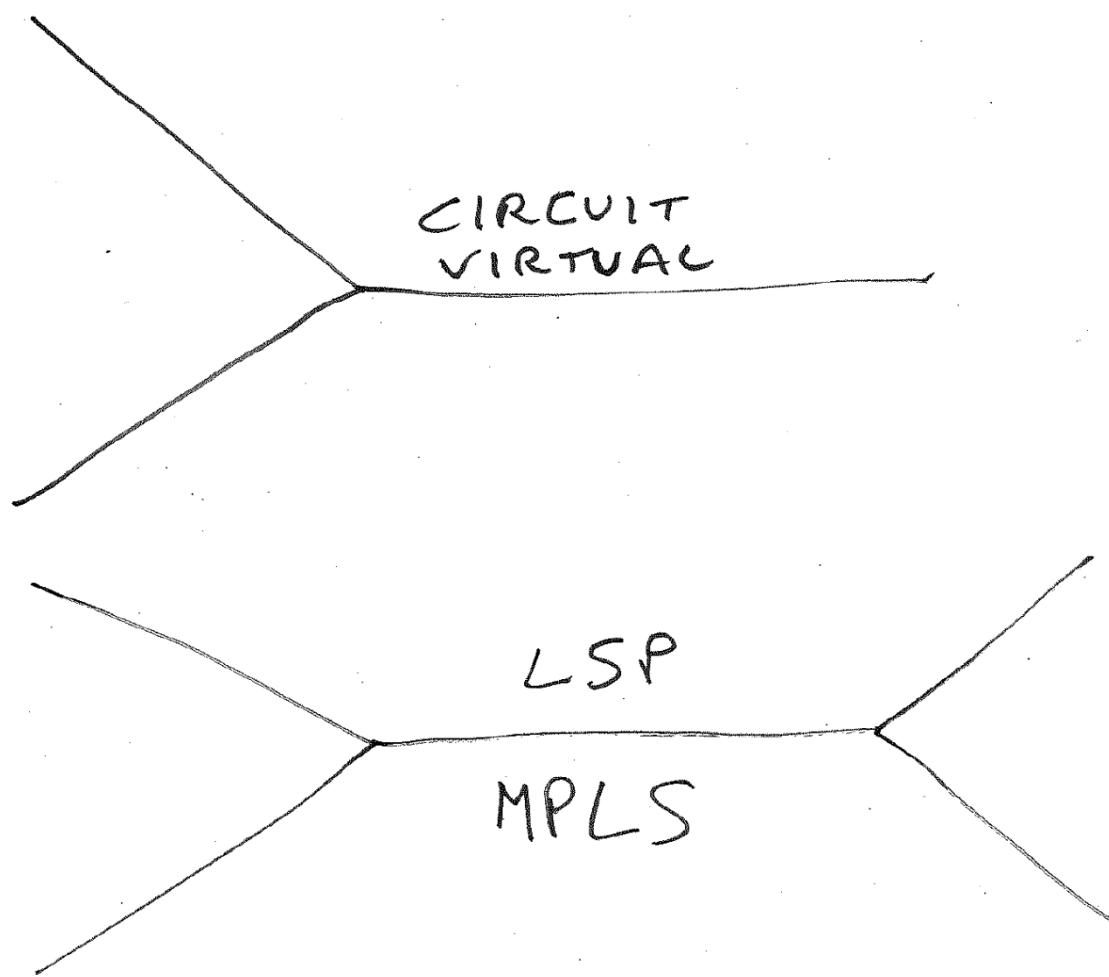
Figure 21.3 LSP Creation and Packet Forwarding through an MPLS Domain

# Label Stacking

- One of the most powerful features of MPLS
  - Processing is always based on the top label
  - At any LSR a label may be removed or added
- Allows creation of tunnels
  - Tunnel refers to traffic routing being determined by labels
- Provides considerable flexibility
- Unlimited stacking

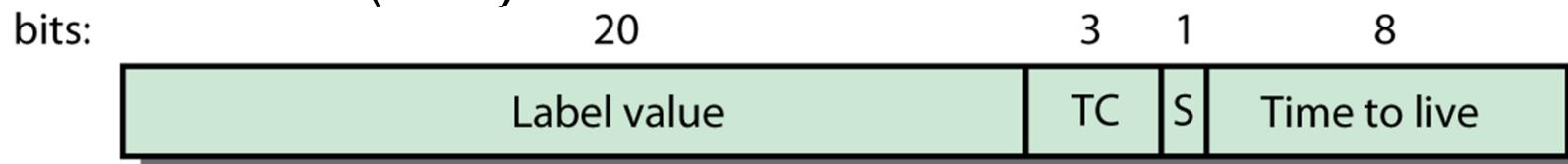


# **LSP with label Stacking**



# Label Format

- defined in RFC 3032
  - 32-bit field consisting of:
    - Label value
    - Traffic class (TC)
    - S
    - Time to live (TTL)



TC = traffic class

S = bottom of stack bit

# Label Placement

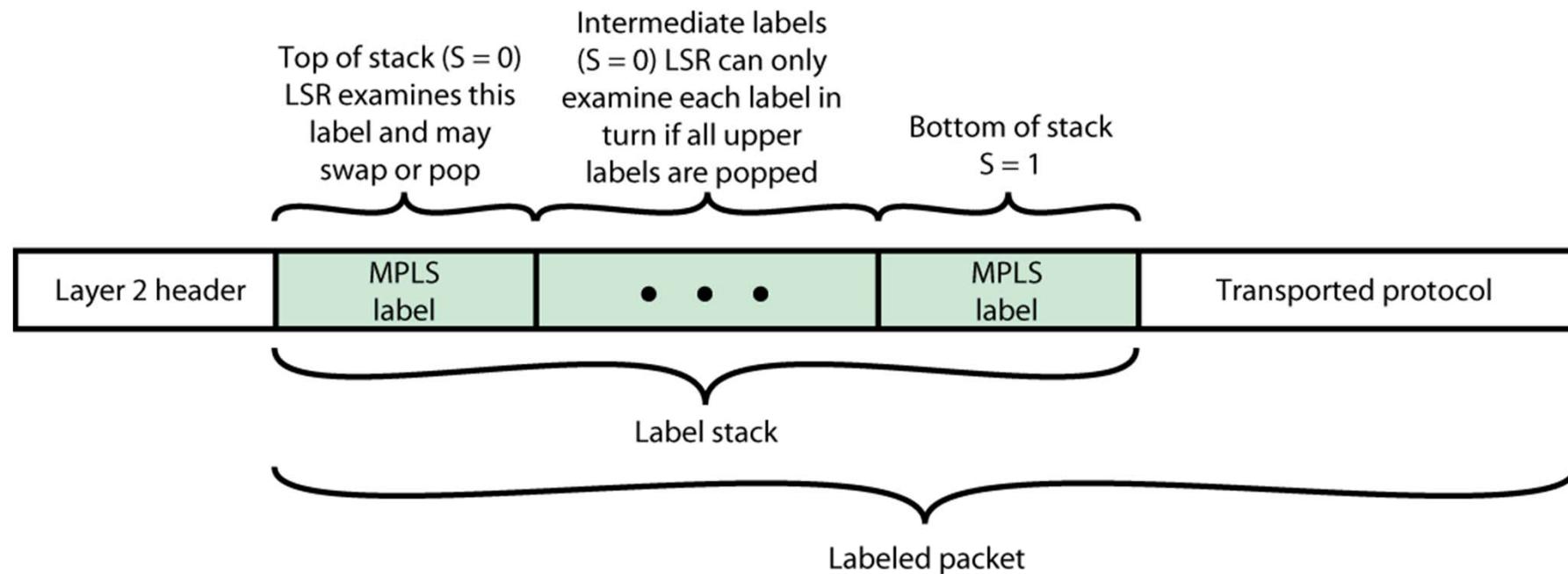
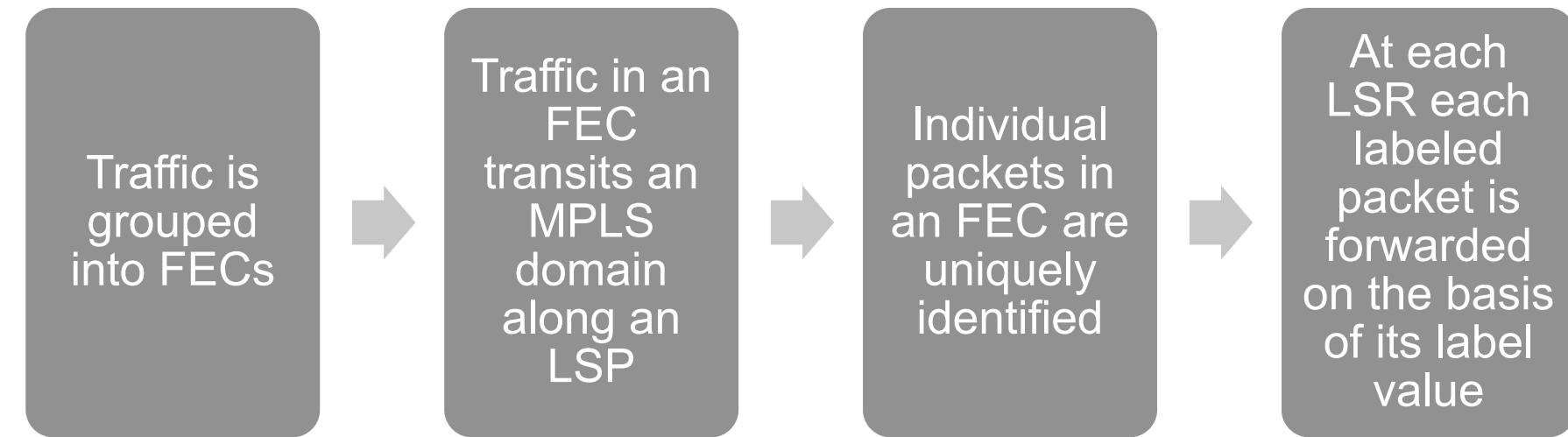


Figure 21.5 Encapsulation for Labeled Packet

# FECs, LSPs, and Labels



# ***Traffic Engineering***

---

- RFC 2702
- Allocate traffic to the network to maximize utilization of the network capacity
- Ensure the most desirable route through the network while meeting QoS requirements

# Example of Traffic Engineering

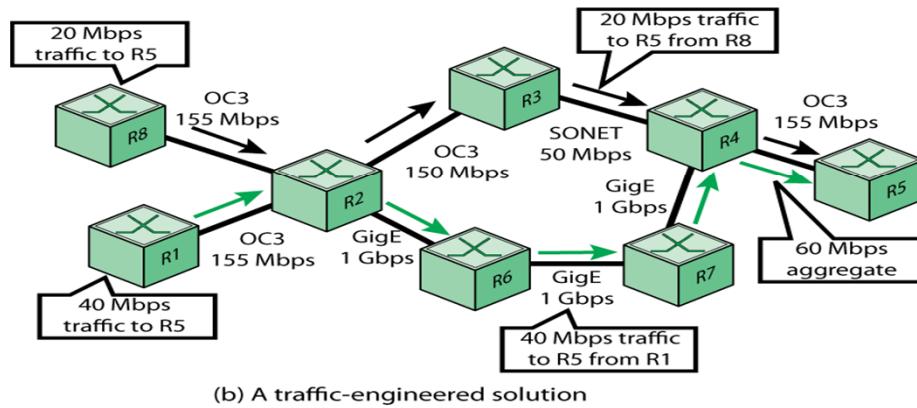
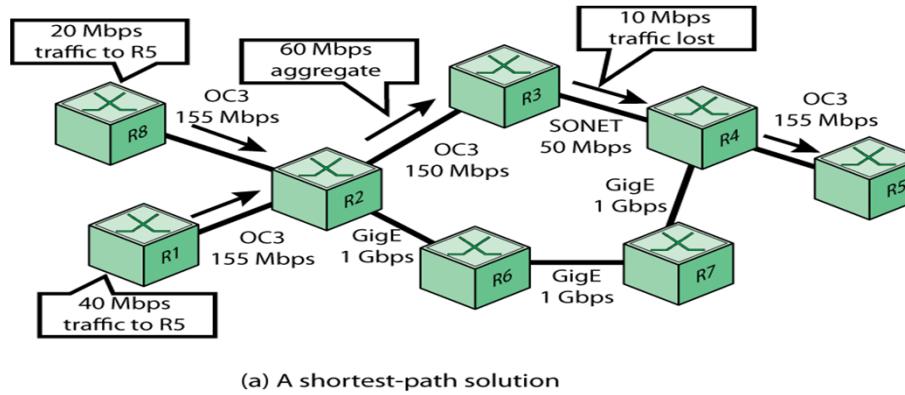


Figure 21.9 Traffic Engineering Example

# **Elements of MPLS Traffic Engineering (MPLS TE)**

- Information distribution
  - A link state protocol is necessary to discover the topology of the network
- Path calculation
  - Shortest path through a network that meets the resource requirements of the traffic flow
- Path setup
  - Signaling protocol to reserve the resources for a traffic flow and to establish the LSP
- Traffic forwarding
  - Accomplished with MPLS using the LSP

# RSVP – TE Operation

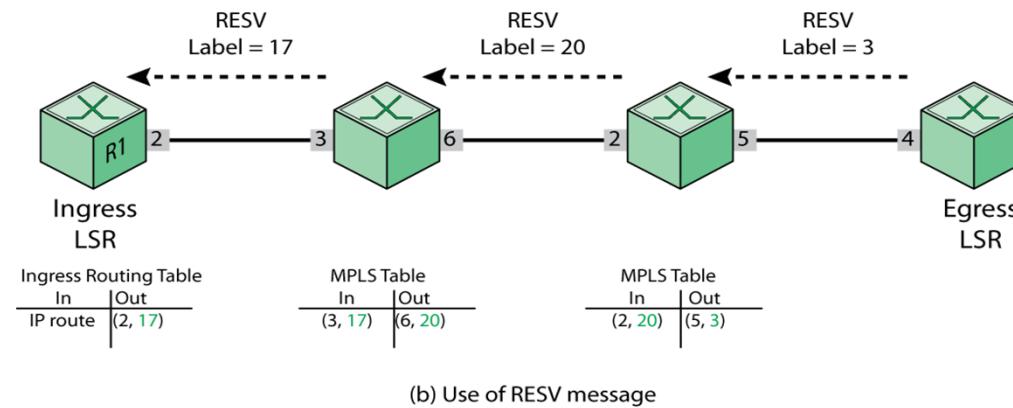
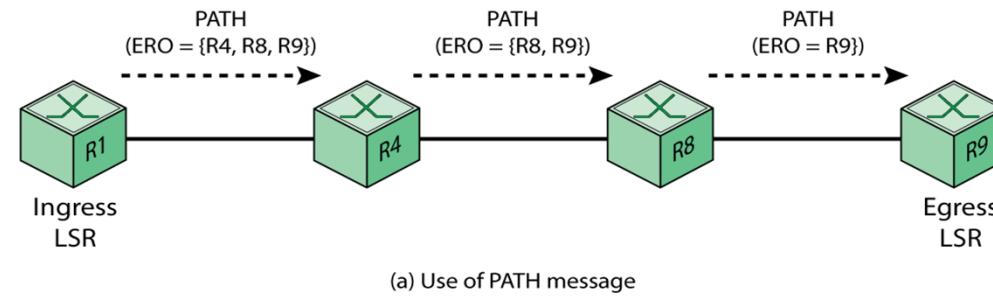


Figure 21.11 RSVP-TE Operation

# ***Virtual Private Network (VPN)***

- Private network configured within a public network in order to take advantage of management facilities of larger networks
- Traffic designated as VPN traffic can only go from a VPN source to a destination in the same VPN

Widely used by enterprises to:

- Create wide area networks (WANs)
- Provide site-to-site communications to branch offices
- Allow mobile user to dial up their company LANs



# Layer 2 VPN

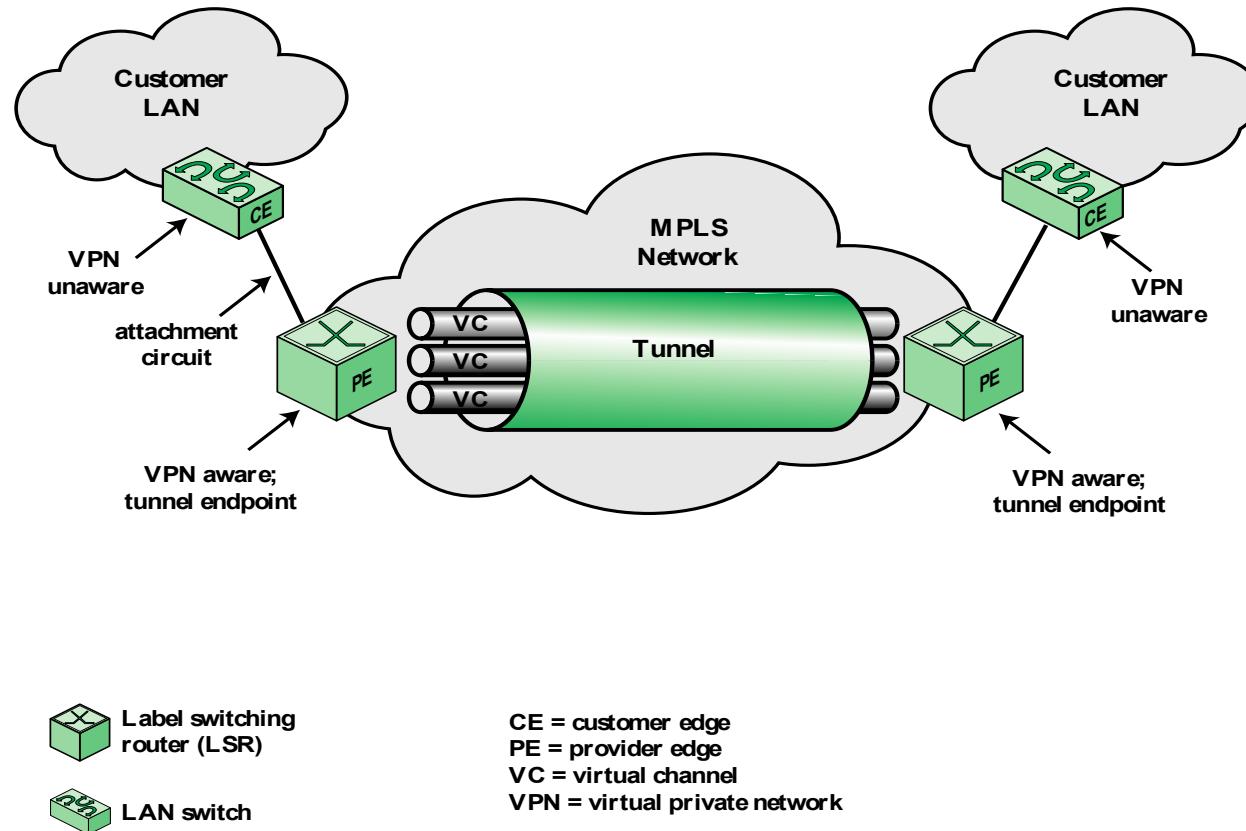


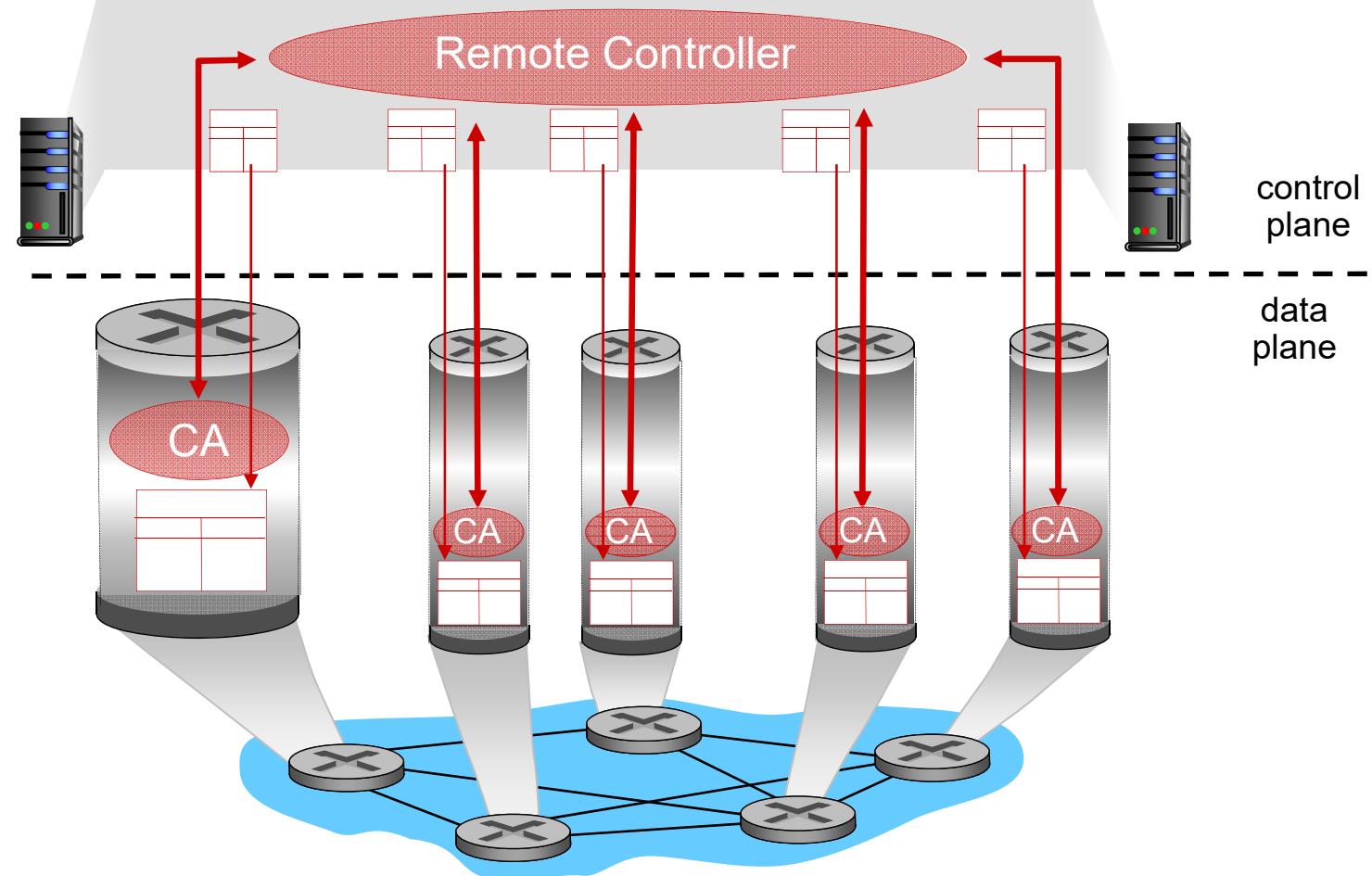
Figure 23.12 Layer 2 VPN Concepts



## ***3.4 Software Defined Networking (SDN)***

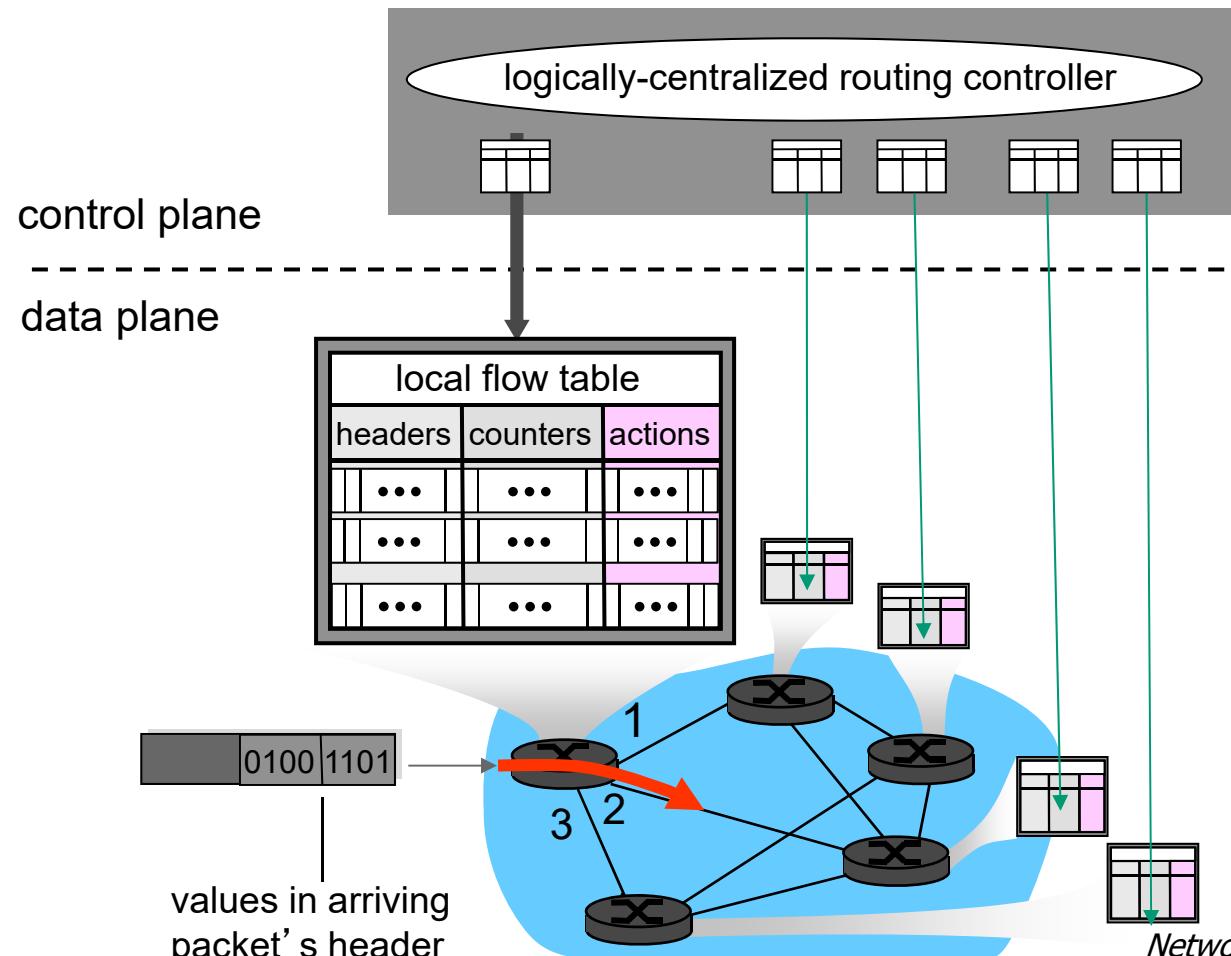
# ***Logically centralized control plane***

A distinct (typically remote) controller interacts with local control agents (CAs) in routers to compute forwarding tables



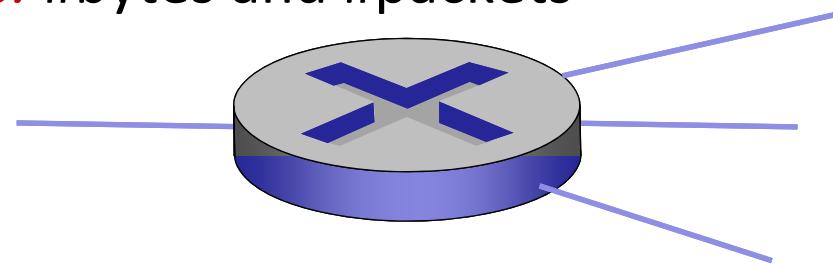
# Generalized Forwarding and SDN

Each router contains a *flow table* that is computed and distributed by a *logically centralized* routing controller



# *Open Flow data plane abstraction*

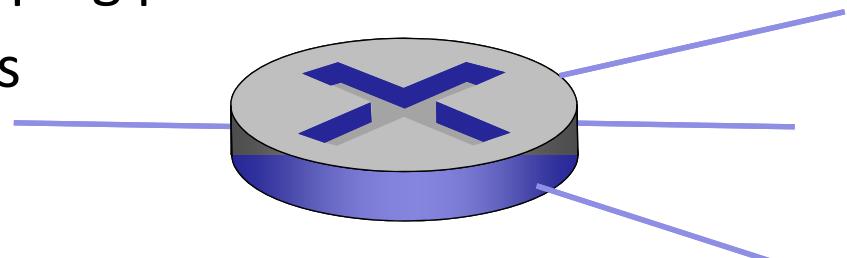
- *flow*: defined by header fields
- generalized forwarding: simple packet-handling rules
  - *Pattern*: match values in packet header fields
  - *Actions: for matched packet*: drop, forward, modify, matched packet or send matched packet to controller
  - *Priority*: disambiguate overlapping patterns
  - *Counters*: #bytes and #packets



*Flow table in a router (computed and distributed by controller) define router's match+action rules*

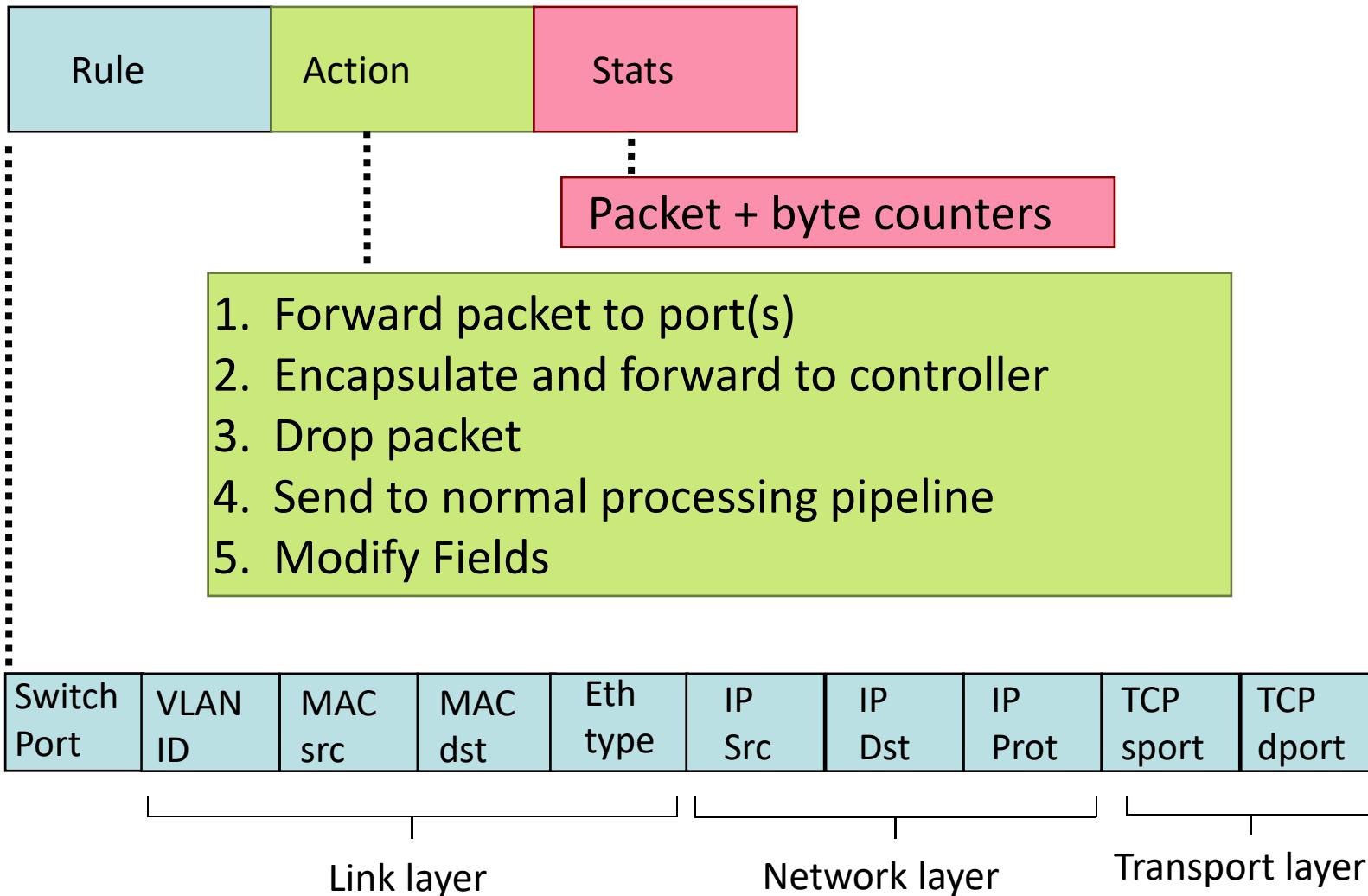
# Open Flow data plane abstraction

- *flow*: defined by header fields
- generalized forwarding: simple packet-handling rules
  - *Pattern*: match values in packet header fields
  - *Actions*: for matched packet: drop, forward, modify, matched packet or send matched packet to controller
  - *Priority*: disambiguate overlapping patterns
  - *Counters*: #bytes and #packets



1.  $\text{src}=1.2.*.*$ ,  $\text{dest}=3.4.5.* \rightarrow \text{drop}$
2.  $\text{src} = *.*.*.*$ ,  $\text{dest}=3.4.*.* \rightarrow \text{forward}(2)$
3.  $\text{src}=10.1.2.3$ ,  $\text{dest} = *.*.*.* \rightarrow \text{send to controller}$

# Open Flow: Flow Table Entries



# Examples (1)

## Destination-based forwarding:

Switch Port	MAC src	MAC dst	Eth type	VLAN ID	IP Src	IP Dst	IP Prot	TCP sport	TCP dport	Action
*	*	*	*	*	*	51.6.0.8	*	*	*	port6

*IP datagrams destined to IP address 51.6.0.8 should be forwarded to router output port 6*

## Firewall:

Switch Port	MAC src	MAC dst	Eth type	VLAN ID	IP Src	IP Dst	IP Prot	TCP sport	TCP dport	Forward
*	*	*	*	*	*	*	*	*	22	drop

*do not forward (block) all datagrams destined to TCP port 22*

Switch Port	MAC src	MAC dst	Eth type	VLAN ID	IP Src	IP Dst	IP Prot	TCP sport	TCP dport	Forward
*	*	*	*	*	128.119.1.1	*	*	*	*	drop

*do not forward (block) all datagrams sent by host 128.119.1.1*

# Examples (2)

Destination-based layer 2 (switch) forwarding:

Switch Port	MAC src	MAC dst	Eth type	VLAN ID	IP Src	IP Dst	IP Prot	TCP sport	TCP dport	Action
*	22:A7:23: 11:E1:02	*	*	*	*	*	*	*	*	port3

*layer 2 frames from MAC address 22:A7:23:11:E1:02  
should be forwarded to output port 6*

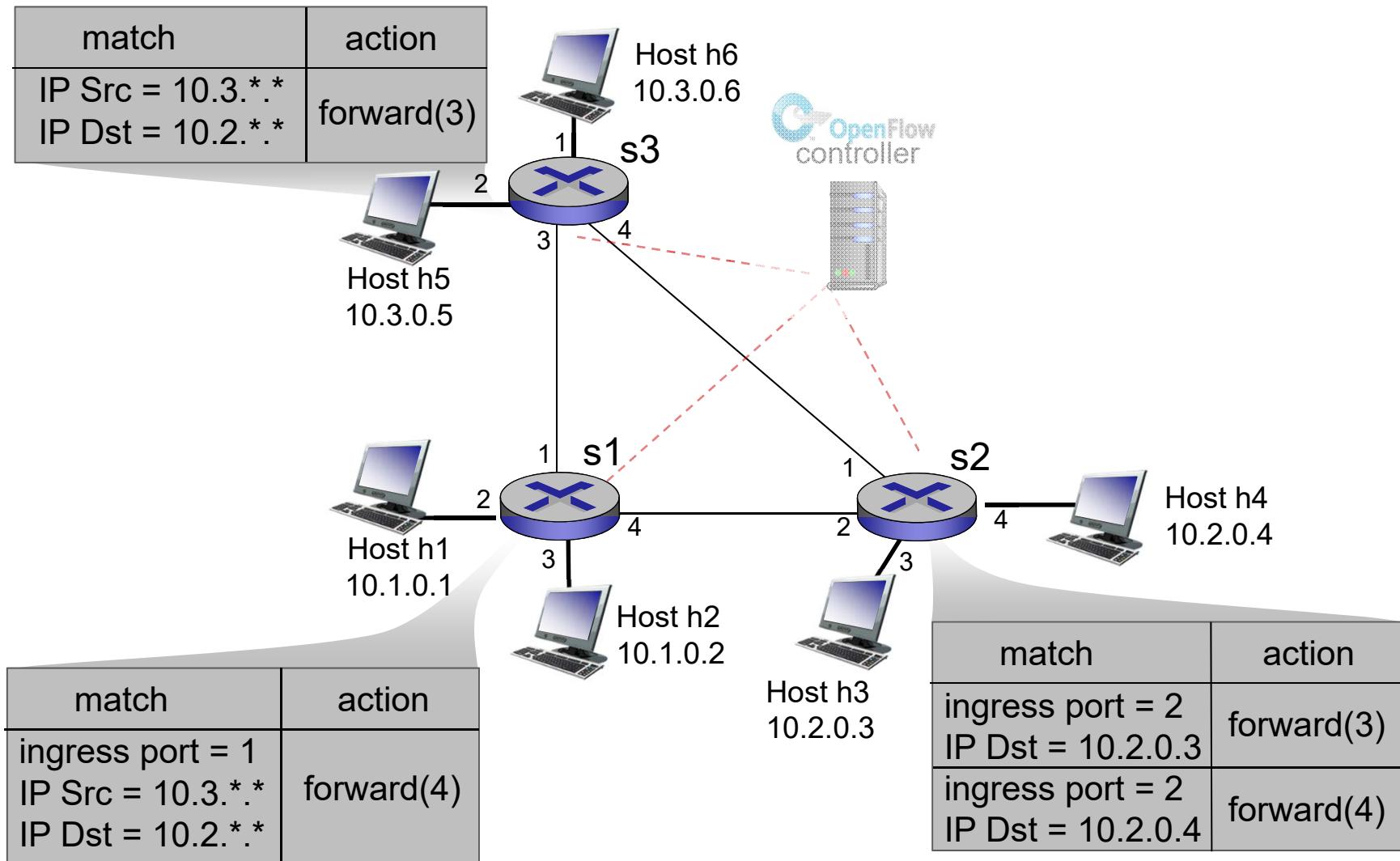
# *Open Flow abstraction*

---

- *match+action*: unifies different kinds of devices
- Router
  - *match*: longest destination IP prefix
  - *action*: forward out a link
- Switch
  - *match*: destination MAC address
  - *action*: forward or flood
- Firewall
  - *match*: IP addresses and TCP/UDP port numbers
  - *action*: permit or deny
- NAT
  - *match*: IP address and port
  - *action*: rewrite address and port

# OpenFlow example

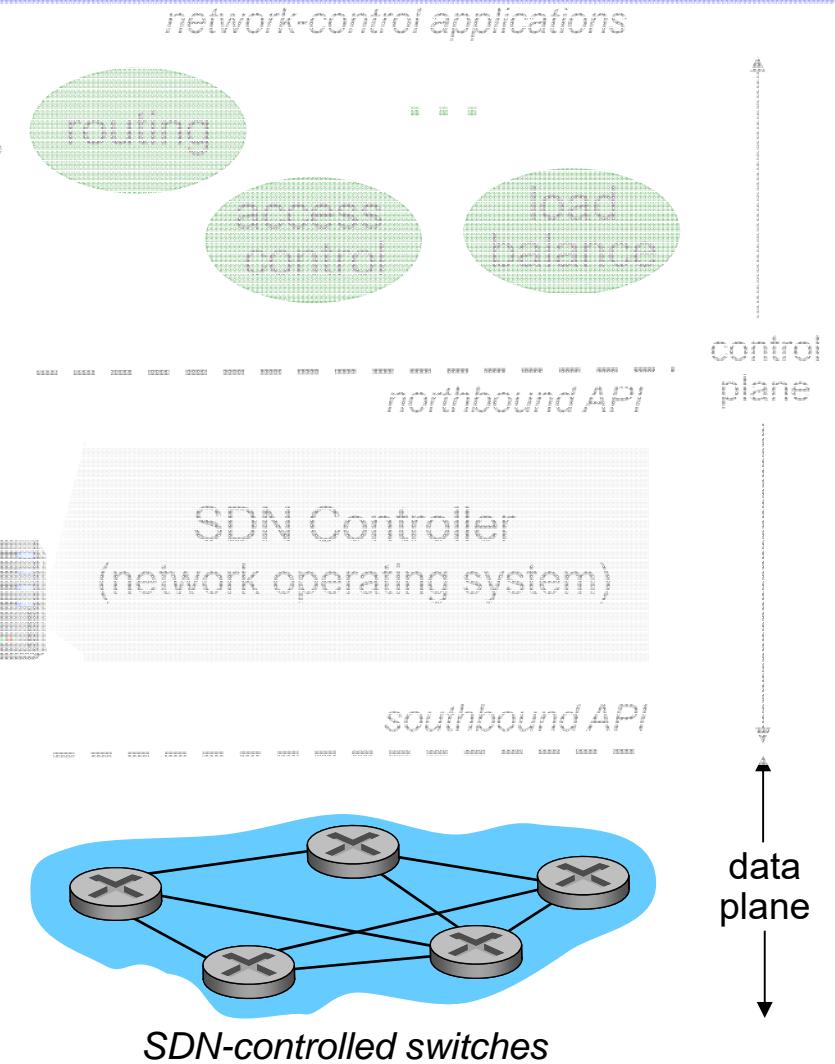
*Example:* datagrams from hosts h5 and h6 should be sent to h3 or h4, via s1 and from there to s2



# SDN perspective: data plane switches

## Data plane switches

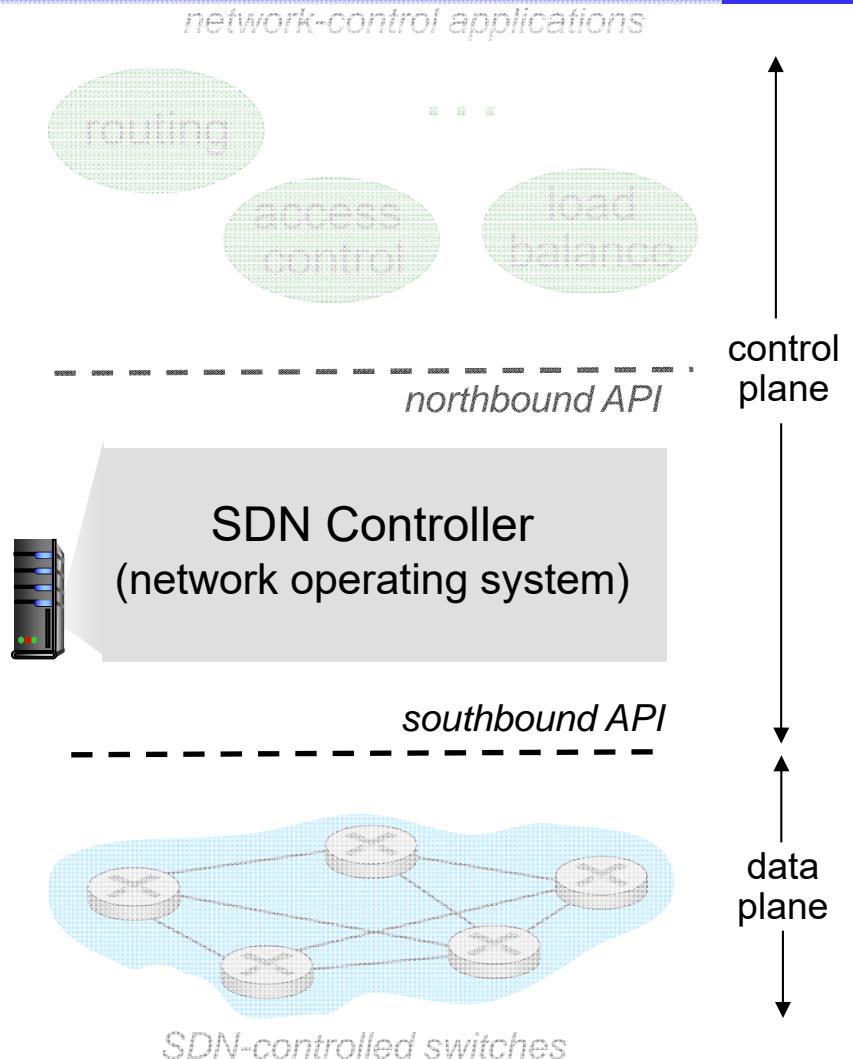
- fast, simple, commodity switches implementing generalized data-plane forwarding (Section 4.4) in hardware
- switch flow table computed, installed by controller
- API for table-based switch control (e.g., OpenFlow)
  - defines what is controllable and what is not
- protocol for communicating with controller (e.g., OpenFlow)



# SDN perspective: SDN controller

## SDN controller (network OS):

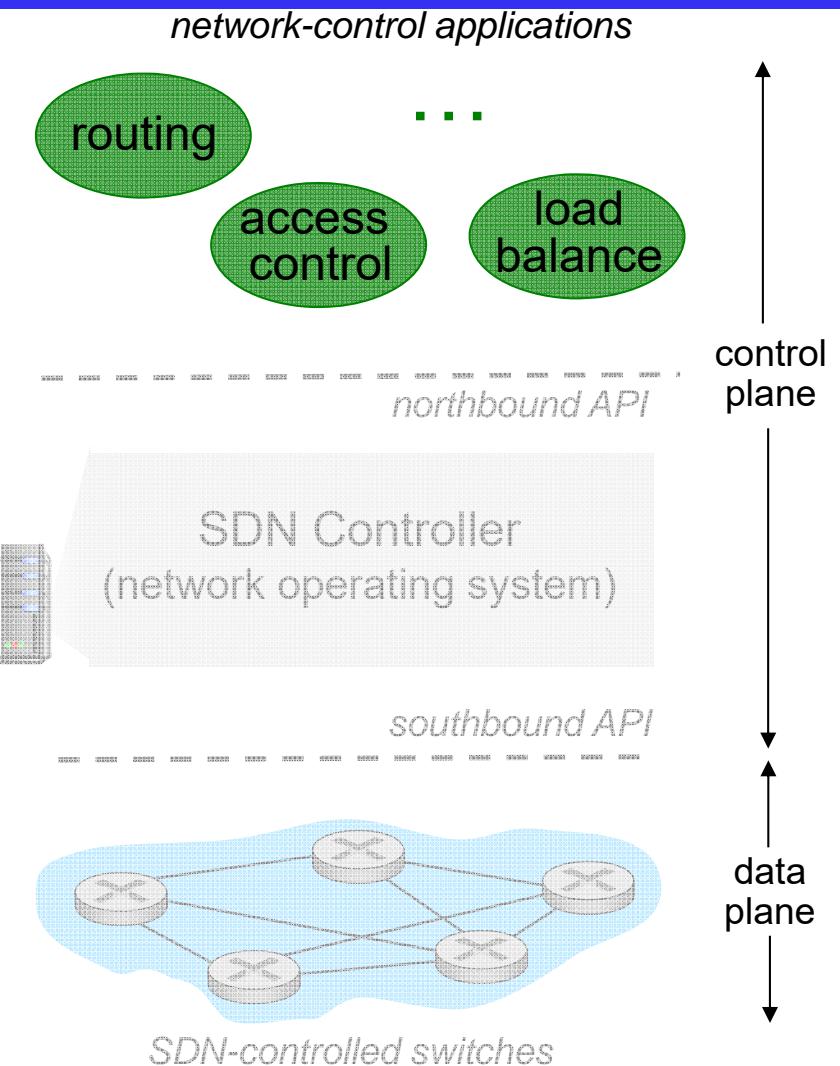
- maintain network state information
- interacts with network control applications “above” via northbound API
- interacts with network switches “below” via southbound API
- implemented as distributed system for performance, scalability, fault-tolerance, robustness



# SDN perspective: control applications

## network-control apps:

- “brains” of control: implement control functions using lower-level services, API provided by SDN controller
- *unbundled*: can be provided by 3<sup>rd</sup> party: distinct from routing vendor, or SDN controller

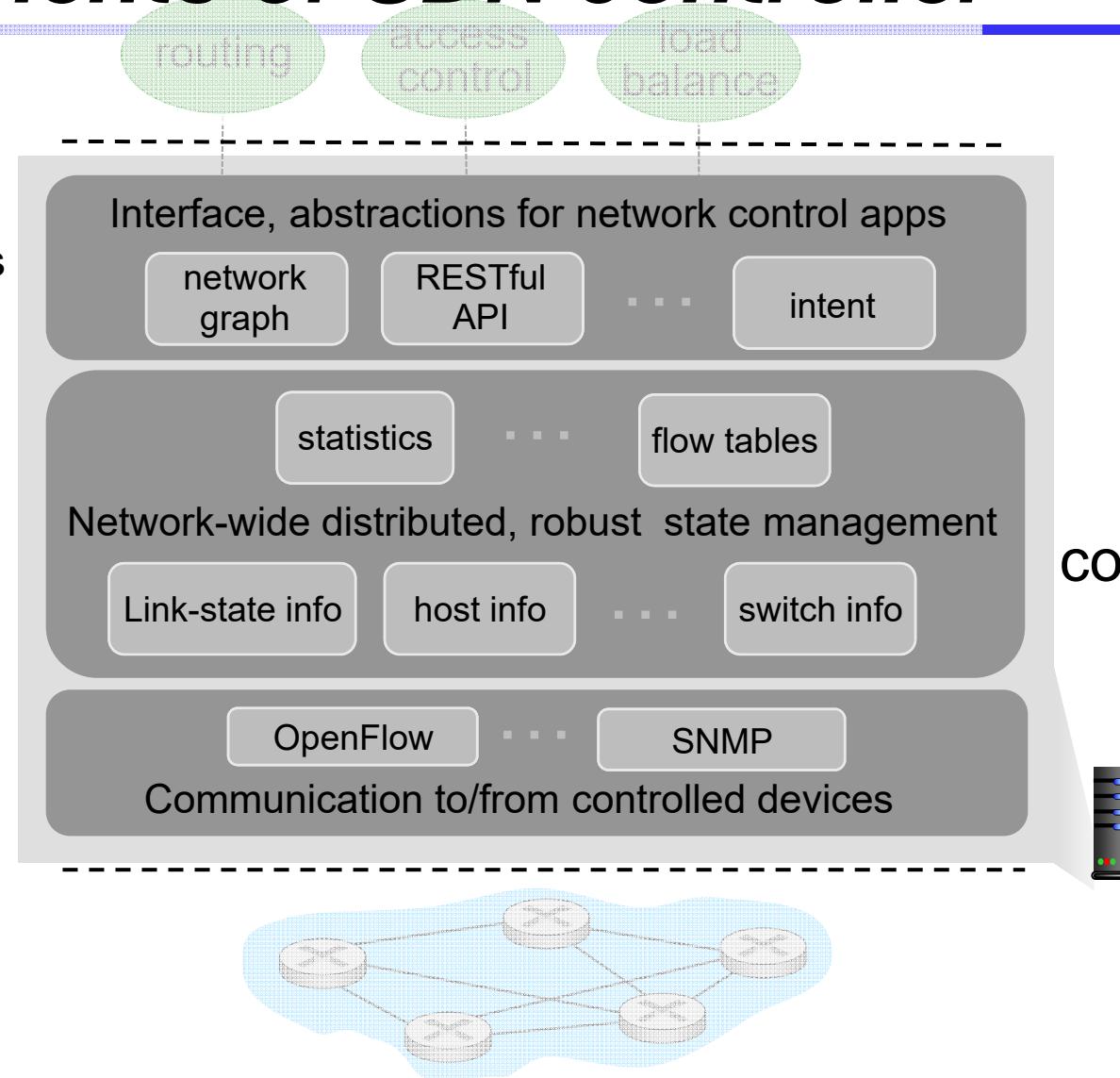


# Components of SDN controller

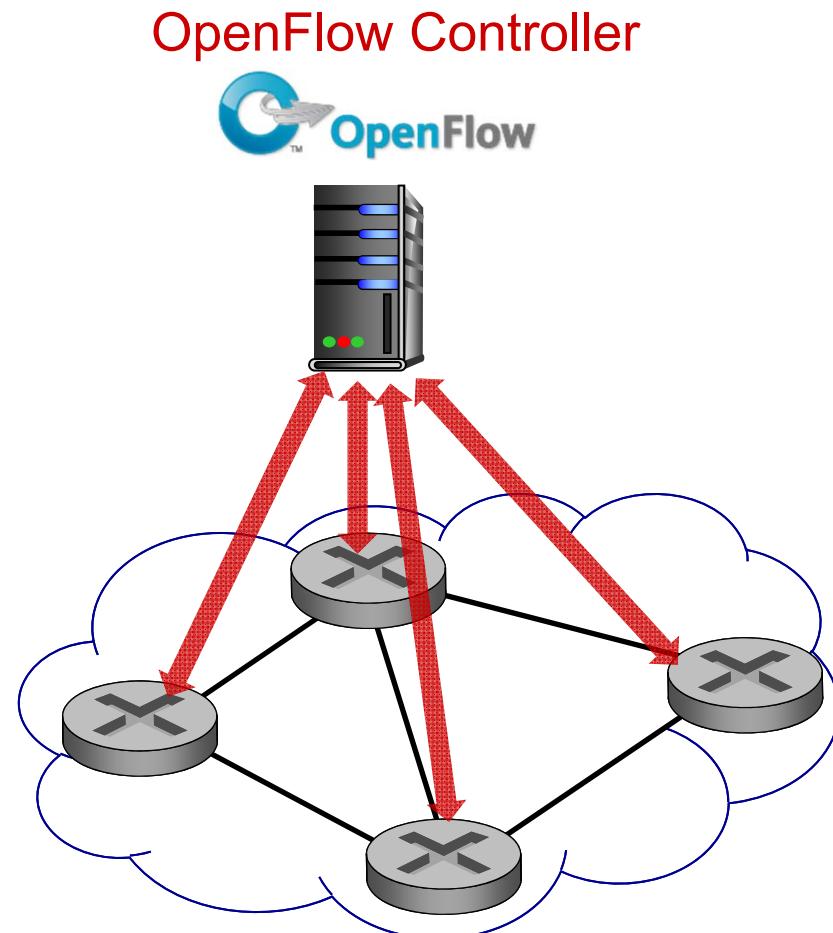
Interface layer to network control apps: abstractions API

Network-wide state management layer: state of networks links, switches, services: a distributed database communication layer:

communicate between SDN controller and controlled switches



# *OpenFlow protocol*

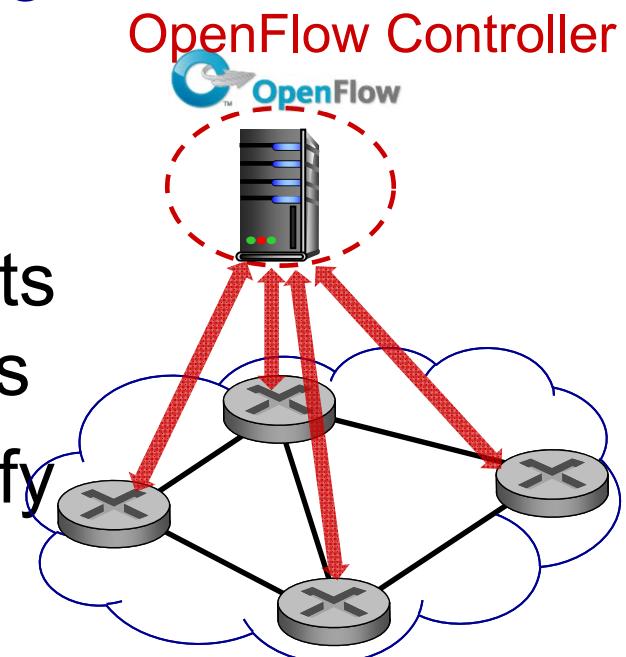


- operates between controller, switch
- TCP used to exchange messages
  - optional encryption
- three classes of OpenFlow messages:
  - controller-to-switch
  - asynchronous (switch to controller)
  - symmetric (misc)

# ***OpenFlow: controller-to-switch messages***

## *Key controller-to-switch messages*

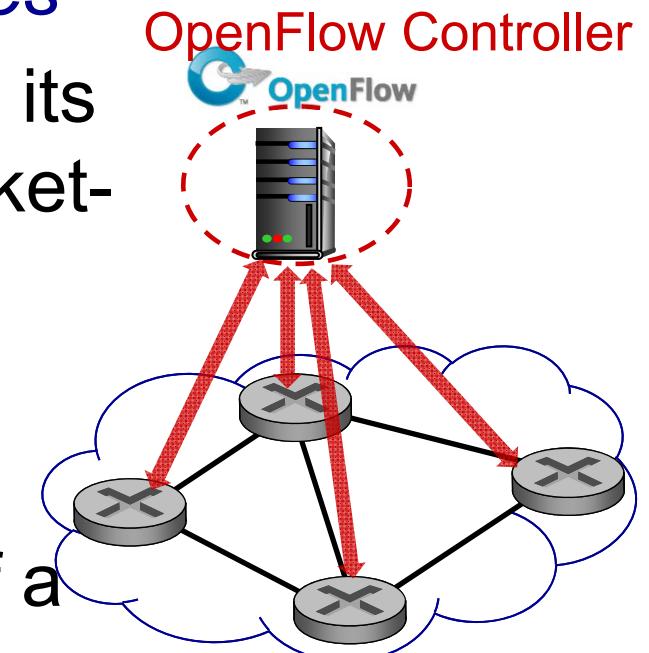
- ***features***: controller queries switch features, switch replies
- ***configure***: controller queries/sets switch configuration parameters
- ***modify-state***: add, delete, modify flow entries in the OpenFlow tables
- ***packet-out***: controller can send this packet out of specific switch port



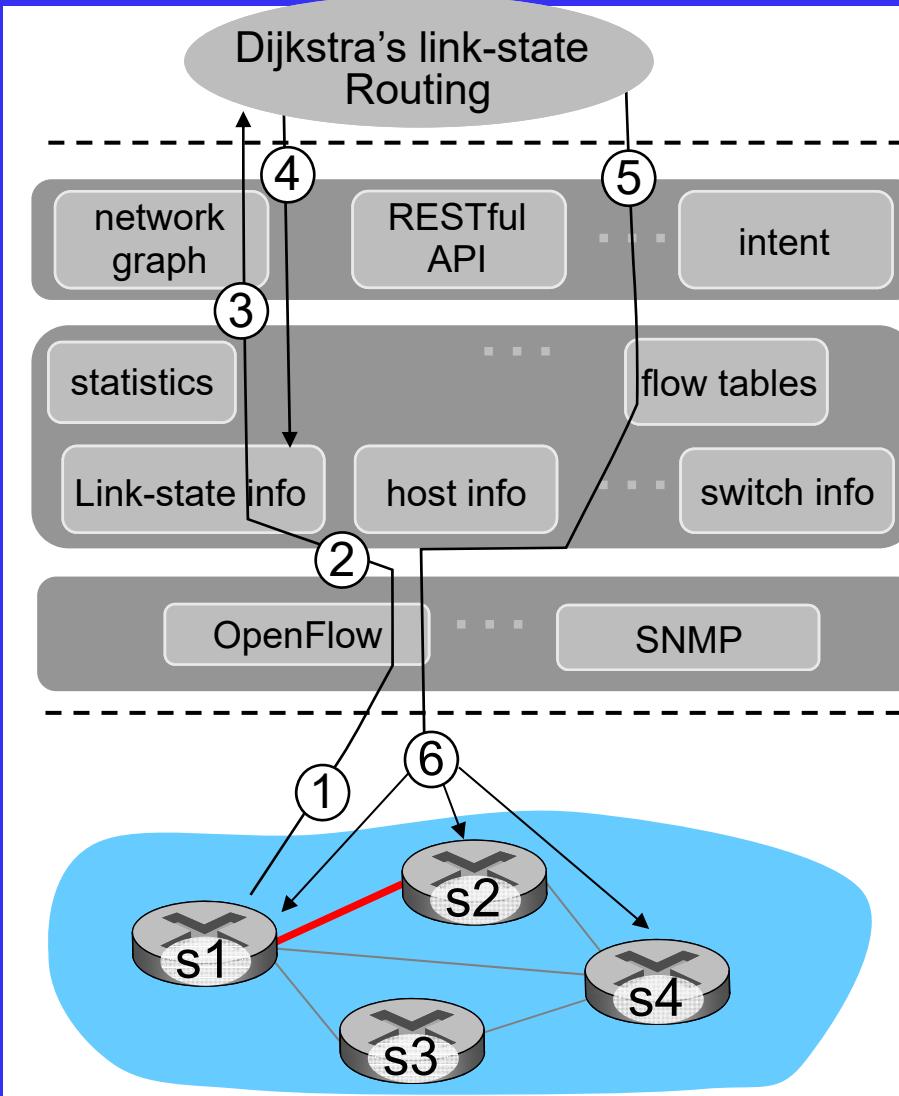
# **OpenFlow: switch-to-controller messages**

## *Key switch-to-controller messages*

- **packet-in:** transfer packet (and its control) to controller. See packet-out message from controller
- **flow-removed:** flow table entry deleted at switch
- **port status:** inform controller of a change on a port.

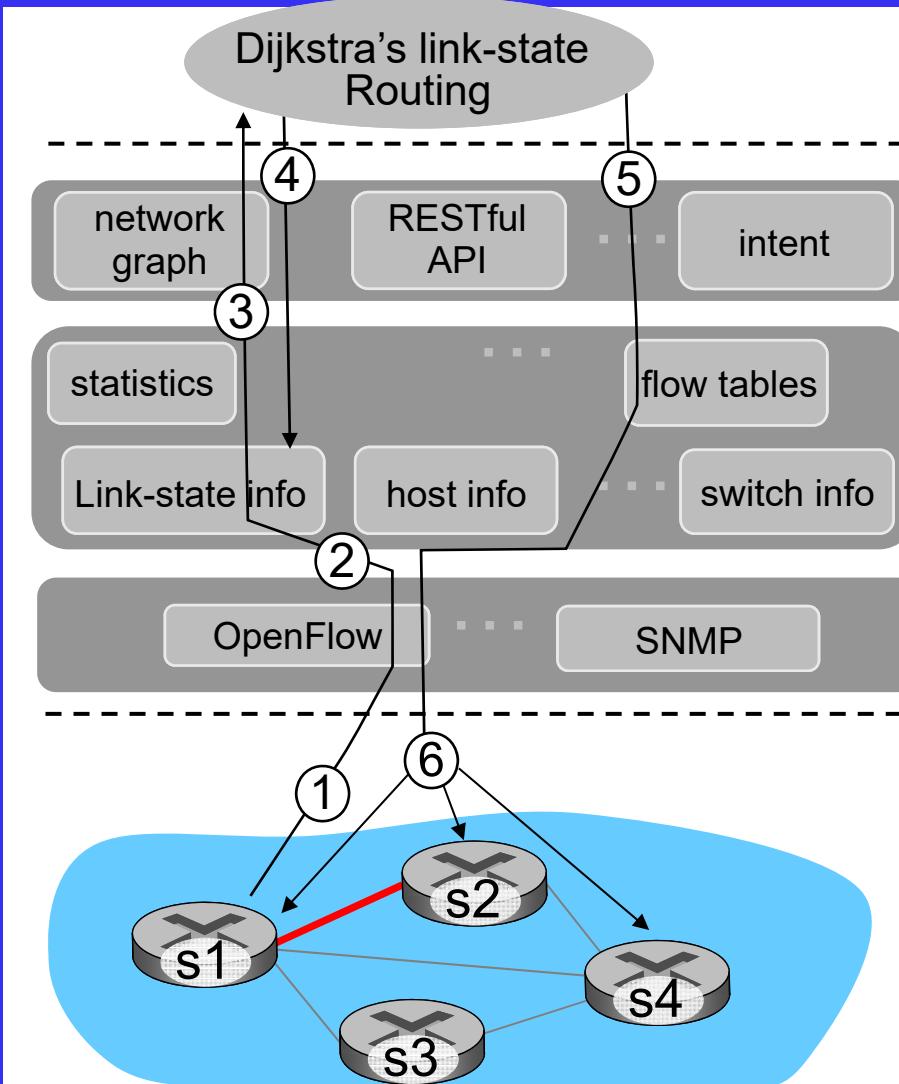


# SDN: control/data plane interaction example



- ① S1, experiencing link failure using OpenFlow port status message to notify controller
- ② SDN controller receives OpenFlow message, updates link status info
- ③ Dijkstra's routing algorithm application has previously registered to be called when ever link status changes. It is called.
- ④ Dijkstra's routing algorithm access network graph info, link state info in controller, computes new routes

# SDN: control/data plane interaction example



- ⑤ link state routing app interacts with flow-table-computation component in SDN controller, which computes new flow tables needed
- ⑥ Controller uses OpenFlow to install new tables in switches that need updating