Resumos RCOM

MIEIC

Janeiro 2019

Conteúdo

1 Physical Layer

1.1 Transmitting information

$$C = 2B \log_2 M \tag{1}$$

C

channel capacity

B

bandwith

2B

baudrate in symbol/s or baud

M
: levels used to encode information

1.2 Types of modulations

- Binary signal
- Amplitude modulation
- Frequency modulation
- Phase modulation
- Quadrature Amplitude Modulation (M - QAM)

1.3 Shannon's Law

The maximum theoretical capacity of a channel (bit/s) is given by the following expressions:

$$SNR = \frac{P_r}{N_0 B_c} \tag{2}$$

$$C = B_c \log_2(1 + SNR) \tag{3}$$

SNR
: signal to noise ratio

B_c
: bandwidth of the channel (Hz)

P_r
: signal power as seen by receiver (W)

N_0
: White noise; noise power per unit bandwidth (W/Hz)

1.4 Free space loss

$$\frac{P_t}{P_r} = \frac{(4\lambda f d)^2}{c^2} \tag{4}$$

P_t
: signal power at transmitting antenna

P_r
: signal power at receiving antenna

λ
: carrier wavelength

d
: propagation distance between antennas

c
: speed of light $3 * 10^8$ m/s

1.5 Solved Exam Problems

2018R - 1

- 16 QAM
- bitrate (C) = 8kbit/s
- baudrate (2B) = ?

$$C = 2B \log_2 M$$
$$8 = 2B \log_2 16$$
$$8 = 2B * 4$$
$$2B = 2$$

2018N - 1

- 8PSK
- baudrate (2B) = 250 kbaud
- bitrate (C) = ?

$$C = 2B \log_2 M$$
$$C = 250 \log_2 8$$
$$C = 250 * 3$$
$$C = 750$$

2017N - 2

- baudrate (2B) = 100 kbaud
- bitrate (C) = 300 kbit/s
- phase modulation
- n^{o} of phases = ?

$$C = 2B \log_2 M$$
$$300 = 100 \log_2 M$$
$$3 = log_2 M$$
$$M = 2^3 = 8$$

2017N - 3

$$\frac{P_t}{P_r} = \frac{(4\lambda f d)^2}{c^2}$$

$$P_r = \frac{P_t}{\frac{(4\lambda f d)^2}{c^2}}$$

$$P_r = \frac{P_t c^2}{(4\lambda f d)^2}$$

$$SNR = \frac{P_r}{N_0 B}$$

Quanto maior d, menor P_r, e quanto menor P_r, menor SNR. Quanto maior B, menor SNR, logo menor a eficiência.

2016R - 1

- baudrate (2B) = 80 kbaud
- bitrate (C) = 320 kbit/s
- phase modulation
- nº of phases = ?

$$C = 2B \log_2 M$$

$$320 = 80 \log_2 M$$

$$4 = log_2 M$$

$$M = 2^4 = 16$$

2016N - 2

- 16 QAM
- bitrate (C) = 100kbit/s
- baudrate (2B) = ?

$$C = 2B \log_2 M$$

$$100 = 2B \log_2 16$$

$$100 = 2B * 4$$

$$2B = 25$$

2016N - 3 Canal rádio com propagação em espaço livre.

$$\frac{P_t}{P_r} = \frac{(4\lambda f d)^2}{c^2}$$

$$P_r = \frac{P_t}{\frac{(4\lambda f d)^2}{c^2}}$$

$$P_r = \frac{P_t c^2}{(4\lambda f d)^2}$$

$$SNR = \frac{P_r}{N_0 B}$$

$$C = B_c \log_2(1 + SNR)$$

Quanto menor a distância e frequência, maior P_r. Quanto maior P_r, maior SNR, logo maior a capacidade.

2015 - 2

- 2 ligações sem fios
- Pt1 = Pt2 (potência transmitida pelo emissor)
- B1 = B2 (largura de banda do canal)
- d1 < d2 (distância entre o emissor e o recetor)
- relação entre P e C das ligações?

$$\frac{P_t}{P_r} = \frac{(4\lambda f d)^2}{c^2}$$

$$P_t = P_r \frac{(4\lambda f d)^2}{c^2}$$

De Pt1 = Pt2,

$$P_r 1 * (4\lambda f d_1)^2 = P_r 2 * (4\lambda f d_2)^2$$

Como d1 < d2, P_r1 > P_r2, então C1 > C2

$$C_1 = B_1 \log_2(1 + \frac{P_r 1}{N_0 B_1})$$

$$C_2 = B_1 \log_2(1 + \frac{P_r 2}{N_0 B_1})$$

2014N - 1

- baudrate (2B) = 8 kbaud
- bitrate (C) = 32 kbit/s
- bandwith (B) = 4 kHz
- M = ?

$$C = 2B \log_2 M$$

$$32 = 8 \log_2 M$$

$$4 = \log_2 M$$

$$M = 16$$

2013N - 1

- bandwith (B) = 1 MHz
- baudrate (2B) = 2 MHz
- SNR = 40 dB
- 8 level impulses => M = 8
- bitrate (C) = ?

$$C = 2B \log_2 M$$

$$C = 2 \log_2 8$$

$$C = 2 * 3 = 6$$

2012N - 1

- 4 QAM
- baudrate (2B) = 100 kbaud
- bitrate (C) = ?

$$C = 100 \log_2 4$$

$$C = 100 * 2$$

$$C = 200$$

2011N - 2 Num canal sem fios, potência recebida é tanto maior quanto menor for a distância emissor-recetor e o comprimento de onda da portadora.

$$\frac{P_t}{P_r} = \frac{(4\lambda f d)^2}{c^2}$$

$$P_r = \frac{P_t}{\frac{(4\lambda f d)^2}{c^2}}$$

$$P_r = \frac{P_t c^2}{(4\lambda f d)^2}$$

2 Data Link Layer

2.1 Data Link layer functions and services

2.1.1 Main functions

- Provide service interface to the network layer.
- Eliminate/reduce transmission errors.
- Regulate data flow: Slow receivers not swamped by fast senders.

2.1.2 Services provided

Principal service: Transfer data from the network layer on the source machine to the network layer on the destination machine.

There are three reasonable possibilities that we will consider:

- **Unacknowledged connectionless service:**
  - No logical connection is established beforehand or released afterwards.
  - Transmitter sends independent frames without having the destination machine acknowledge them.
  - If a frame is lost due to noise on the line, no attempt is made to detect or recover from that loss.
  - Appropriate when the error rate is very low and for real-time traffic.
- **Acknowledged connectionless service:**
  - No logical connections used.
  - Each frame sent is individually acknowledged so the sender knows if a frame arrived correctly or has been lost.
  - If it has not arrived within a specified time interval, it can be sent again.

  - This service is useful over unreliable channels, such as wireless systems.(i.e. Wi-Fi).

- **Acknowledged connection-oriented service:**
  - The source and destination machines establish a connection before any data are transferred.
  - Each frame is numbered, and the data link layer guarantees that each frame sent is indeed received.
  - Guarantees that each frame is received exactly once and that all frames are received in the right order.
  - Appropriate over long, unreliable links (satellite channel, long-distance telephone circuit).
  - Divided in 3 phases:
    * **First phase:** The connection is established (initialize variables and counters needed to keep track of which frames have been received and which ones have not).
    * **Second phase:** One or more frames are actually transmitted.
    * **Third phase:** The connection is released (free the variables, buffers, and other resources used to maintain the connection).

2.2 Framing

Breaking up the bit stream into discrete frames, computing a short token called a checksum for each frame, and including the checksum in the frame when it is transmitted. When a frame arrives at the destination,the checksum is recomputed. If it is different from the one contained in the frame, the data link layer knows that an error occurred.

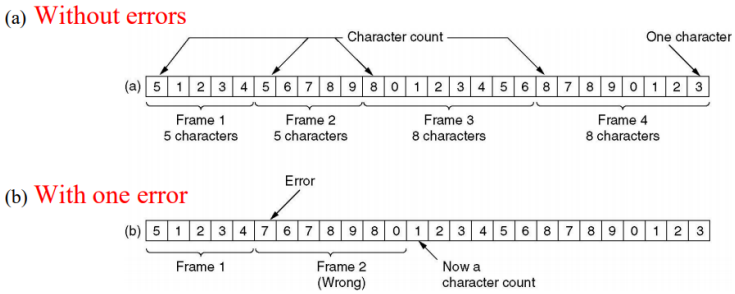
A good design must make it easy for a receiver to find the start of new frames while using little of the channel bandwidth. We will look at three methods:

2.2.1 Byte count

Uses a field in the header to specify the number of bytes in the frame. When the data link layer at the destination sees the byte count, it knows how many bytes follow and hence where the end of the frame is.

Issues

- The count can be garbled by a transmission error.
- A single bit flip, may trigger the destination to get out of synchronization.
- If an out-of-sync occurs it is unable to locate the correct start of the next frame.



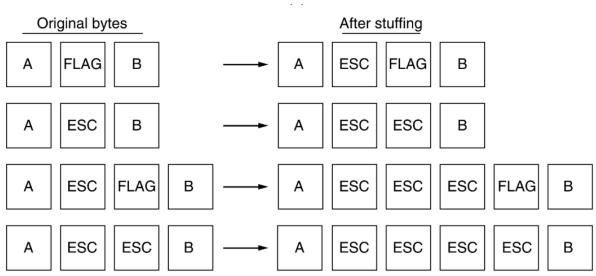
2.2.2 Flag bytes with byte stuffing

This method gets around the problem of resynchronization by having each frame start and end with special bytes (flag bytes).

However, this flag may occur in the middle of the data and induce the receiver in error by thinking the end of the frame was reached. This issue can be solved using **byte stuffing**.

Byte Stuffing

- Inserting a special escape byte (ESC) before each flag byte in the data.
- Makes framing flag bytes distinguishable from the ones in the data.
- Escape bytes present in the data also need to be escaped.



2.2.3 Flag bits with bit stuffing

- Each frame begins and ends with a special bit pattern (01111110 / 0x7E).
- When the sender finds five consecutive 1 bits in the data, it stuffs a 0 bit into the outgoing bit stream.
- When the receiver finds five consecutive incoming 1 bits, followed by a 0 bit, it destuffs the 0 bit.

Advantages:

- The boundary between two frames is unambiguously recognized by the flag pattern (flag sequences can only occur at frame boundaries and never within the data).
- Frames can contain an arbitrary number of bits made up of units of any size.
- Ensures a minimum density of transitions that help the physical layer maintain synchronization.

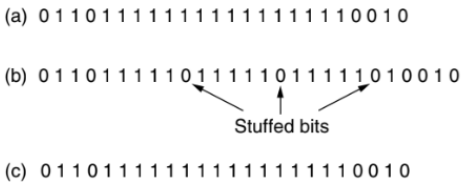


Figure 5. Bit stuffing. (a) The original data. (b) The data as they appear on the line. (c) The data as they are stored in the receiver's memory after destuffing.

Both byte and bit stuffing:

- Are completely transparent to the network layer in both computers.
- Have a frame length that depends on the contents of the data.

Many data link protocols use a combination of these three methods for safety.

2.3 Error detection

2.3.1 Types of Errors

- **Simple Error:** Random and independent from previous error.
- **Errors in burst:**
  - Not independent.
  - Affect neighbour bits.
  - Burst lenght defined by the first and last bits in error.

2.3.2 Counting Errors

Frame Error Probability(FER):

$$FER = 1 - (1 - BER)^n \tag{5}$$

BER = Bit Error Ratio
n = frame length

No Error Probability: $P = (1-p)^n$
Error Probability: $P = 1 - (1-p)^n$
i Error Probability: $P = \binom{n}{i} p^i (1-p)^{n-i}$

p = bit error probability
n = frame length

2.3.3 Error Detection Techniques

Used by the receiver to determine if a packet contains errors. If a packet is found to contain errors, the receiver may request the transmitter to re-send the packet.

2.3.4 Parity Check

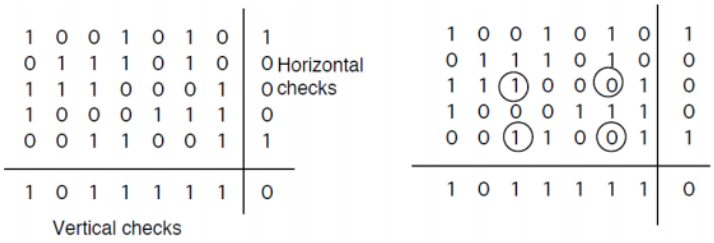
Simple Parity Check: One parity bit added to every k information bits so that:

- The total number of bits 1 even (even parity).
- The total number of bits 1 odd (odd parity).

Allows the detection of simple errors and any number of odd errors in a block of k + 1 bits. However, does not detect even number of errors in a block of k + 1 bits.

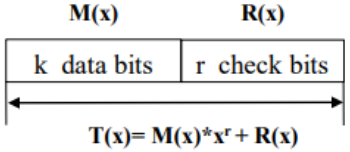
Bi-dimensional Parity

- Parity per row.
- Parity per column.
- Minimum code distance = 4.



2.3.5 Cyclic Redundancy Check (CRC)

A fixed number of check bits are appended to the message to be transmitted. Data receivers check on the check value attached by finding the remainder of the polynomial division of the contents transmitted. If it seems that an error has occurred, a negative acknowledgement is transmitted asking for data retransmission.



The bit string is represented as a polynomium (i.e. $110011 \Rightarrow x^5+x^4+x+1$)

How to compute the check bits: R(x)?

- Choose a generator string G(x) of length r+1 bits.
- Choose R(x) such that T(x) is a multiple of $G(x) : T(x) = A \times G(x)$.

Generating R(x):

$$R(x) = M(x)x^r \textbf{ mod } G(x) \tag{6}$$

Choice of G(x) is very important! ($G(x) = x^r + \ldots + 1$)

Generating R(x) example:

Assume for example:

- r=3.
- $G(x) = x^3 + 1 \Rightarrow 1001$.
- $M(x) = x^5 + x^4 + x^2 + 1 \Rightarrow 110101$.

Then:

- $x^r = x^3$.
- $M(x) \times x^3 = x^8 + x^7 + x^5 + x^3 \Rightarrow 110101000$.
- $R(x) = M(x)x^3$ **mod** $G(x) = 110101000$ **mod** $1001 = 011 = x^1 + 1$

Checking at the Receiver

- Divide T(x) by G(x):
  - If the remainder $R(x) = 0 \Rightarrow$ no errors.
  - If the remainder $R(x)! = 0 \Rightarrow$ errors have occurred.

Performance:

For r check bits per frame the following can be detected

- All patterns of 1, 2, or 3 errors (d $>$ 3).
- All bursts of errors of r or fewer bits.
- All errors consisting of an odd number of inverted bits.

2.4 Automatic Repeat reQuest (ARQ)

An error-control method for data transmission that uses acknowledgements (messages indicating whether or not the message has been correctly received) and timeouts to achieve reliable data transmission over an unreliable service. This mechanisms automatically request the retransmission of:

- Missing packets.
- Packets with errors.

There are three common ARQ schemes:

2.4.1 Stop and Wait

- Sender transmits information frame I and waits for positive confirmation ACK from receiver.
- Receiver receives I frame:
  - If I frame has no error sends ACK.
  - If I frame has error sends NACK.
- Sender receives I frame:

  - If ACK, proceeds and transmits new frame.
  - If NACK, retransmits frame I.

- If I, ACK or NACK is lost a timeout is required!

Issue: If the transmitter times-out and sends a packet twice, the receiver cannot tell whether the second frame is a retransmission or a new frame transmission.
Solution: Define a 1 bit sequence number in the header of the frame.

This sequence number alternates (from 0 to 1) in subsequent frames. The transmitter sends a frame with a sequence number attached to it so the receiver can check if it matches the expected. When the receiver sends an ACK, it includes the sequence number of the next packet it expects. This way, the receiver can detect duplicated frames by checking if the frame sequence numbers alternate.

Efficiency(S):

$$S = \frac{T_f}{T_f + 2 \times T_{prop}} = \frac{1}{1 + 2a} \tag{7}$$

where:

T_f = Data transmission time
T_{prop} = Propagation Delay

Probability of k Attempts required to transmit a frame with success

$$P[A = k] = {p_e}^{k-1}(1 - p_e) \tag{8}$$

where:

p_e = frame error probability(FER)

Expected number of Attempts to transmit a frame with success

$$E[A] = \frac{1}{1 - p_e} \tag{9}$$

where:

p_e = frame error probability(FER)

Efficiency with Errors

$$S = \frac{T_f}{E[A](T_f + 2 \times T_{prop})} = \frac{1 - p_e}{1 + 2a} \tag{10}$$

where:

p_e = frame error probability(FER)

2.4.2 Go Back N

Allows the transmission of new packets before earlier ones are acknowledged.

Sender:

- May transmit up to W frames without receiving RR(Receiver Ready = ACK).
- I frames are numbered sequentially I(NS): I(0), I(1), I(2), etc.
- Cannot send I(NS=i+W) until it has received the RR(NR=i).

Receiver:

- Does not accept frames out of sequence.
- Sends RR(NR) to sender indicating:
  - That all the packets up to NR-1 have been received in sequence.
  - The sequence number, NR, of the next expected frame.

Behaviour under Errors

- Frames with errors are silently discarded by the Receiver.
- If Receiver receives Data frame out of sequence:
  - First out-of-sequence-frame: Receiver sends REJ(NR) where NR = next in-sequence frame expected.
  - Following out-of sequence-frames: Receiver discards them; no REJ sent.
- When Sender receives REJ(NR=x), the Sender:
  - Goes-Back and retransmits I(x), I(x+1), etc.
  - Continues using Sliding Window mechanism.
- If timeout occurs, the Sender:
  - Requests the Receiver to send a RR message.
  - Sends a special message (RR command message).

Maximum Window Size(W):

$$W = M - 1 = 2^k - 1 \tag{11}$$

where:

M = Number of sequence numbers
k = Number of bits used to code sequence numbers

Efficiency:

- If $W \geq 1 + 2a \Rightarrow S = 1$.
- If $W < 1 + 2a \Rightarrow S = \frac{W}{1+2a}$.

Efficiency with Errors:

$$S = \begin{cases} \dfrac{1 - p_e}{1 + 2ap_e} & , W \geq 1 + 2a \\ \dfrac{W(1 - p_e)}{(1 + 2a)(1 - p_e + Wp_e)} & , W < 1 + 2a \end{cases}$$

Figura 1: pe - frame error probability (ratio, FER)

2.4.3 Selective Repeat

Similar to **Go Back N**, however it does not discard successful frames when errors occur.

Receiver:

- Accepts out of sequence frames.
- Confirms negatively, SREJ, a frame not arrived.
- Uses RR to confirm blocks of frames arrived in sequence.

Sender: Retransmits only the frames signaled by SREJ.

Maximum window size(W):

$$W = \frac{M}{2} = 2^{k-1} \tag{12}$$

where:

M = Number of sequence numbers
k = Number of bits used to code sequence numbers

Efficiency:

$$S = \begin{cases} 1 - p_e & ,W \geq 1 + 2a \\ \dfrac{W(1-p_e)}{1+2a} & ,W < 1 + 2a \end{cases}$$

Figura 2: pe - frame error probability (ratio, FER)

2.4.4 Useful Formulas for All Methods

Data transmission time (T_f):

$$T_f = \frac{L}{R} \tag{13}$$

where:

L = Frame Size
R = Data Rate

Propagation Delay(T_{prop}):

$$T_{prop} = \frac{d}{V} \tag{14}$$

where:

d = Distance between sender and receiver
V = Propagation Velocity

SUMETHIN(a)

$$a = \frac{T_{prop}}{T_f} \tag{15}$$

where:

T_{prop} = Propagation Delay
T_f = Data transmission time

Maximum Rate(R_{max}):

$$R_{max} = S \times R \tag{16}$$

where:

S = Efficiency
R = Data rate

Round Trip Time(RTT - Time of transmission and acknowledgement of a frame):

$$RTT = 2 \times T_{prop} + T_f \tag{17}$$

where:

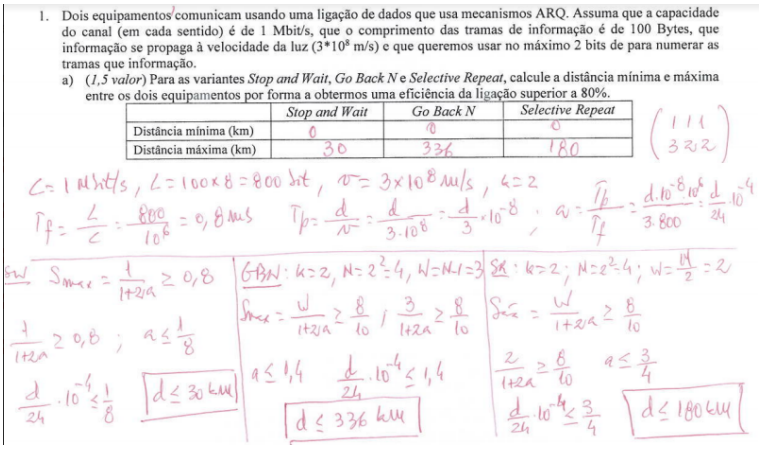
T_{prop} = Propagation Delay
T_f = Data transmission time

2.5 Data Link Layer Some Exam Exercises - Part 2 Only

Note: In some exams, some question may need information given in previous questions.

2.5.1 2016/17 Exam

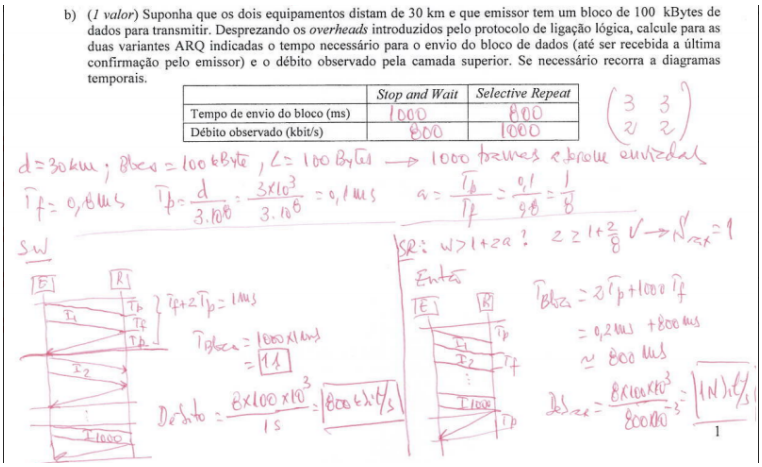
Minimum and Maximum Distances between two machines, having an efficiency above X



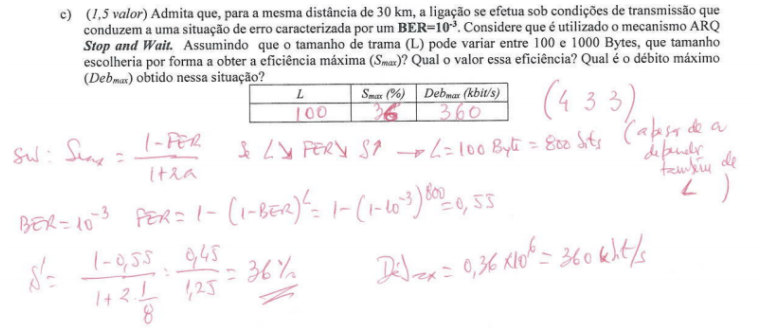
1. Dois equipamentos comunicam usando uma ligação de dados que usa mecanismos ARQ. Assuma que a capacidade do canal (em cada sentido) é de 1 Mbit/s, que o comprimento das tramas de informação é de 100 Bytes, que informação se propaga à velocidade da luz ($3*10^8$ m/s) e que queremos usar no máximo 2 bits de para numerar as tramas que informação.
   a) (*1,5 valor*) Para as variantes *Stop and Wait*, *Go Back N* e *Selective Repeat*, calcule a distância mínima e máxima entre os dois equipamentos por forma a obtermos uma eficiência da ligação superior a 80%.

	Stop and Wait	*Go Back N*	*Selective Repeat*
Distância mínima (km)	0	0	0
Distância máxima (km)	30	336	180

Block of Data Send Time and Observed Rate



Choose Block of Data Size and calculate Efficiency and Maximum Rate



2.5.2 2015/16 Exam

Window Size, Efficiency and Maximum Rate

1. Duas estações separadas por uma distância de 2000 km comunicam usando um protocolo de ligação de dados do tipo ARQ. O atraso de propagação da informação é de 5 μs/km e a capacidade do canal é 1024 kbit/s (em cada sentido). Admita que as tramas de Informação usam 3 bits para numeração, têm um tamanho típico de 2048 bits e são imediatamente confirmadas por tramas de Supervisão em sentido oposto. Despreze o tamanho das tramas de Supervisão.

a) *(1,5 valor)* Para as variantes *Go-Back-N* e *Selective Repeat*, calcule a janela de transmissão, a eficiência máxima do protocolo e os débitos máximos.

	Go-Back-N	*Selective Repeat*
Janela de transmissão, W	7	4
Eficiência máxima, S (%)	63,6	36,1
Débito Máximo (kbit/s)	651	370

$\begin{pmatrix} 2 & 2 \\ 2 & 2 \\ 1 & 1 \end{pmatrix}$

$d = 2000\ km$; $T_p = 5\ \frac{\mu s}{km} \times 2000\ km = 10\ ms$; $T_f = \frac{L}{C} = \frac{2048\ bit}{1024 \times 10^3\ bit/s} = 2\ ms$

$a = T_p / T_f = \frac{10}{2} = 5$; $1 + 2a = 11$; $M = 2^k = 2^3 = 8$

GBN: $W = M - 1 = 8 - 1 = 7$

$W < 1 + 2a \Rightarrow S_{GBN} = \frac{W}{1+2a} = \frac{7}{11} = 63,6\%$

$Deb_{GBN} = S_{GBN} \times C = 0,636 \times 1024 = 651\ kbit/s$

SR: $W = \frac{M}{2} = \frac{8}{2} = 4$

$S_{SR} = \frac{W}{1+2a} = \frac{4}{11} = 36,1\%$

$Deb_{SR} = S_{SR} \times 1024 = 370\ kbit/s$

Efficiency using two different frame sizes

b) *(1 valor)* Pretende-se analisar o efeito dos erros de transmissão e do tamanho das tramas de Informação. Considere tramas com tamanhos 1024 e 2048 bits e uma situação de ruído caracterizada por **BER=10^{-3}**. Calcule a eficiência máxima dos dois mecanismos para estes 2 casos e discuta o comportamento destes mecanismos em relação ao tamanho das tramas

S_{max} (%)	*Go-Back-N*	*Selective Repeat*
L=2048	1,3	4,7
L=1024	2,5	6,9

$\begin{pmatrix} 3 & 2 \\ 2 & 3 \end{pmatrix}$

$L_1 = 2048$: $FER_1 = 1 - (1 - BER)^{L_1} = 1 - (1 - 10^{-3})^{2048} = 0,87$; $1 - FER_1 = 0,13$

GBN: $7 < 11 \Rightarrow S = \frac{W(1-FER)}{(1+2a)(1-FER+WFER)} = \frac{7 \times 0,13}{11 \times (0,13 + 7 \times 0,87)} = \frac{0,91}{68,42} = 1,3\%$

SR: $4 < 11 \Rightarrow S = \frac{W(1-FER)}{1+2a} = \frac{4 \times 0,13}{11} = 4,7\%$

$L_2 = 1024$; $T_f = \frac{L}{C} = \frac{1024}{1024 \times 10^3} = 1\ ms$; $a = \frac{T_p}{T_f} = \frac{10}{1} = 10$; $1 + 2a = 21$

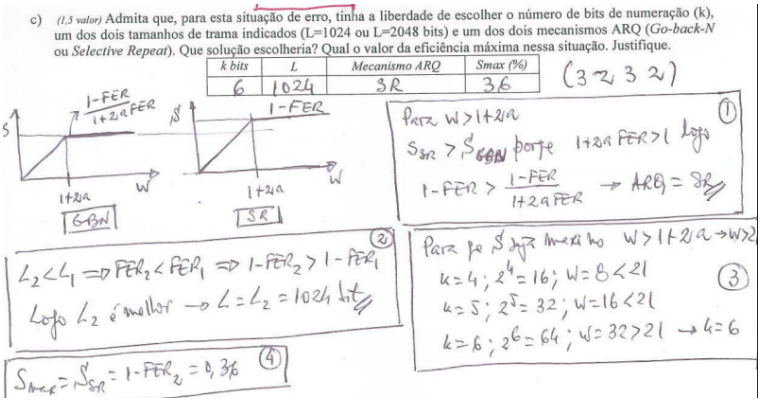
$FER_2 = 1 - (1 - BER)^{L_2} = 1 - (1 - 10^{-3})^{1024} = 0,64$; $1 - FER_2 = 0,36$

GBN: $7 < 21 \Rightarrow S = \frac{W(1-FER)}{(1+2a)(1-FER+WFER)} = \frac{7 \times 0,36}{21(0,36 + 7 \times 0,64)} = \frac{2,36}{101,64} = 2,5\%$

SR: $4 < 21 \Rightarrow S = \frac{W(1-FER)}{1+2a} = \frac{4 \times 0,36}{21} = 6,9\%$

1

Number of bits for sequence number, Frame Size, Mechanism ARQ, Efficiency



3 Delay Models

3.1 Communication Link

- Bit pipe with a given capacity C (bit/s)
- Link capacity -> rate at which bits are transmitted to the link
- Link may transport multiplexed traffic streams

Important Variables and Expressions

C
: channel capacity (total capacity)

3.2 Multiplexing Strategies

- Statistical Multiplexing
- Frequency Division Multiplexing
- Time Division Multiplexing

3.3 Statistical Multiplexing

- Packets of all traffic streams merged in a single quele (first-come, first-served)

Important Variables and Expressions

L
: Length of packet

T_{frame}
: time of transmition

$$T_{frame} = L/C \tag{18}$$

3.4 Frequency Division Multiplexing

- Link capacity C subdivided into m portions
- Channel bandwidth W subdivided into m channels of W/m Hz
- Capacity of each channel = C/m

Important Variables and Expressions

L
: Length of packet

T_{frame}
: time of transmition

m
: number of divisions

W
: channel bandwidth

$$T_{frame} = Lm/C \tag{19}$$

3.5 Time Division Multiplexing

- Time axis divided into m slots of fixed length
- Communication -> m channels with capacity C/m

Important Variables and Expressions

L
: Length of packet

T_{frame}
: time of transmition

m
: number of divisions

$$T_{frame} = Lm/C \tag{20}$$

3.6 Queue Models

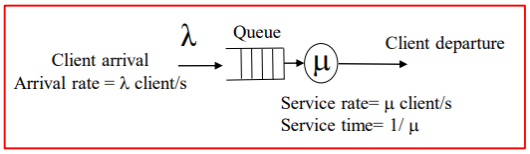


Figura 3: Depiction of a queue model

- Characterization of Delay - Important performance parameter in computer networks
- Customers (packet to be transmitted through a link) arrive at random times to obtain service (transmit a packet)

Important Variables and Expressions

λ
: arrival rate

μ
: service rate

N
: Average number of customers/packets in the network

T
: Average delay per packet -> waiting plus service times

ρ
: traffic intensity (occupation of the server)

$T_{pac(frame)}$
: Service time = packet transmission time

$$T_{pac(frame)} = L/C = 1/\mu \tag{21}$$

$$\rho = \lambda/\mu \tag{22}$$

3.6.1 M/M/1 Queue

Important Variables and Expressions

T_W
: average waiting time

N_W
: average number of clients waiting

$$N = \rho/1 - \rho = \lambda/\mu - \lambda \tag{23}$$

$$T = 1/\mu - \lambda \tag{24}$$

$$T_W = T - T_S = 1/\mu - \lambda - 1/\mu = \rho/\mu(1 - \rho) \tag{25}$$

$$N_W = T_w\lambda = \lambda/\mu - \lambda - \lambda/\mu = N - \rho \tag{26}$$

3.6.2 M/D/1 Queue

$$T_W = \lambda E(T^2_{pac(frame)})/2(1 - \rho) \tag{27}$$

4 MAC Sublayer

Hello, here is some text without a meaning. This...

5 Network Layer

5.1 Network

:Network label - camada responsável pela transferência de pacotes

5.2 Overview

- Camada de Network (Network layer)
  - Transporta os pacotes(datagrams)
  - "from sending host to receiving host"
  - funções localizadas em todos os hosts e routers
- Transmissor(Sender):
  - Encapsula a informação em pacotes
  - Cria os pacotes
- Receptor(Receiver):
  - Recebe os pacotes
  - Envia a informação para o transport layer
- Router:

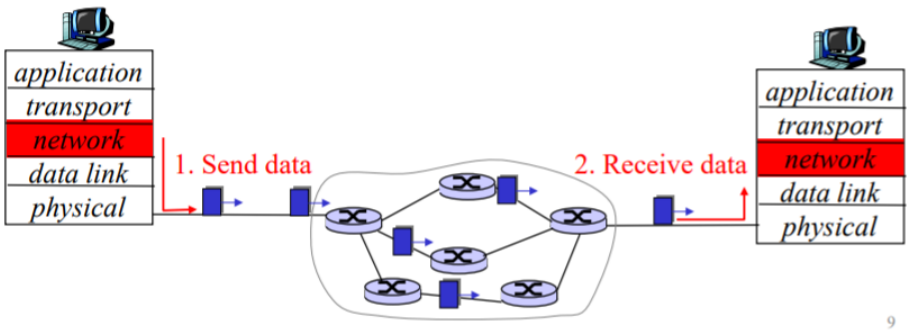
- Recebe os pacotes pela linha de input
- Examina o cabeçalho dos pacotes
- Reencaminha os pacotes para o sítio certo
- Tem de saber o caminho mais curto para determinar o caminho

5.3 Funções principais da camada de rede

- Forwarding
  - router trata de enviar o pacote desde a porta de entrada(input) até à porta de saída(outpu)
- Routing
  - determina a rota definida pelos packets
  - algoritmos, caminho mais curto

5.4 Rede de datagramas

- Serviço não orientado à ligação
- Não há o conceito de circuito
- Os pacotes são redirecionados de acordo com a fonte e o destino
- Pacotes com o mesmo par fonte-destino podem seguir caminhos diferentes



Destination Address Range	Output Link Interface
11001000 00010111 00010000 00000000 through 11001000 00010111 00010111 11111111	0
11001000 00010111 00011000 00000000 through 11001000 00010111 00011000 11111111	1
11001000 00010111 00011001 00000000 through 11001000 00010111 00011111 11111111	2
otherwise	3

2^{32} possible entries in IPv4

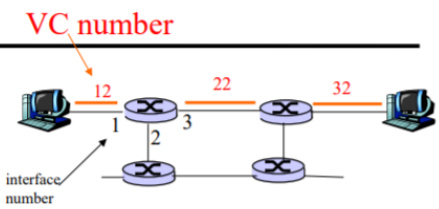
5.5 Circuitos Virtuais

- Serviço orientado à ligação
- Fases:
  1. Estabelecer o circuito
  2. Transferência de dados
  3. Terminação do circuito
- Cada pacote carrega um identificador do circuito virtual
- Caminho da fonte ao destino -> sequência de identificadores virtuais, um para cada ligação
- Estado de cada circuito mantido pelo router, que pode alocar recursos (bandwidth, buffers) por circuito virtual

5.5.1 Forwarding Table

Contém prefixos e a respetiva porta de saída <Endereço/Mask, port>

VC - Forwarding Table



Forwarding table in northwest router:

Incoming interface	Incoming VC #	Outgoing interface	Outgoing VC #
1	12	3	22
2	63	1	18
3	7	2	17
1	97	3	87
…	…	…	…

Routers maintain connection state information!

8

5.5.2 Ex: Maior correspondência de prefixo

Prefix Match	Link Interface
11001000 00010111 00010	0
11001000 00010111 00011000	1
11001000 00010111 00011	2
otherwise	3

Examples. Which Interface?

DA: 11001000 00010111 00010110 10100001 → 0

DA: 11001000 00010111 00011000 10101010 → 1,2 → 1

longest prefix

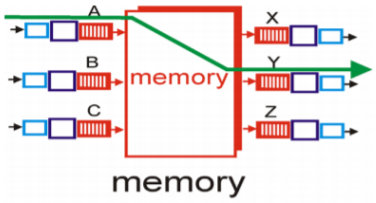
5.6 Circuitos Virtuais versus Rede de Datagramas

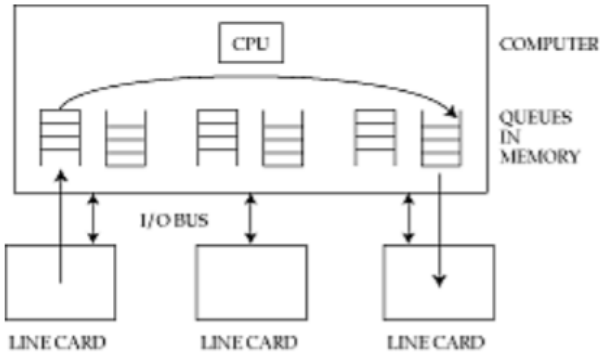
Issue	Datagram subnet	Virtual-circuit subnet
Circuit setup	Not needed	Required
Addressing	Each packet contains the full source and destination address	Each packet contains a short VC number
State information	Routers do not hold state information about connections	Each VC requires router table space per connection
Routing	Each packet is routed independently	Route chosen when VC is set up; all packets follow it
Effect of router failures	None, except for packets lost during the crash	All VCs that passed through the failed router are terminated
Quality of service	Difficult	Easy if enough resources can be allocated in advance for each VC
Congestion control	Difficult	Easy if enough resources can be allocated in advance for each VC

5.7 Arquitetura do router

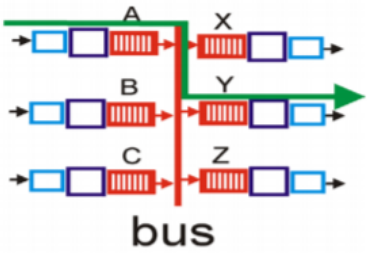
- Funções principais:
  - Correr algoritmos de roteamento e protocolos (RIP, OSPF, BGP)
  - Reencaminhar pacotes
- Componentes principais:
  - Input Port
    - Physical Layer (bit-level)
    - Data Link Layer (e.g., Ethernet)
    - Queuing (se os pacotes chegarem rápido demais)
    - Lookup + Forwarding (faz algum reencaminhamento imediatamente)
  - Output Port
    - Buffering (quando é excedida a velocidade de saída)
    - Queuing (com disciplina de agendamento)(Queuing perda e espera - devido ao overflow do buffer da porta de input)
    - Data Link Layer (protocol, desencaplsulação)
    - Physical Layer (linha de terminação)
  - Switching Fabric
    - Controla o reencaminhamento (fisicamente ou através dum CPU)

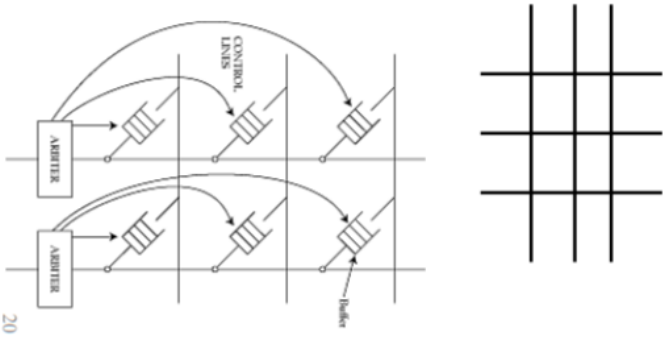
* Switching Via Memória do Computador
  - Router de primeira geração
  - Em computadores tradicionais, switching é controlado pelo CPU
  - Cada pacote é copiado para a memória do sistema e transferida duas vezes pelo bus



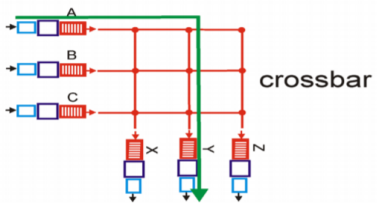


* Switching via a Bus
  - Os pacotes sao processados por um bus partilhado
  - A transferência dos pacotes desde a linha de input e output é realizada de foram direta
  - A taxa da conecção do bus é limitada pela bus bandwidth



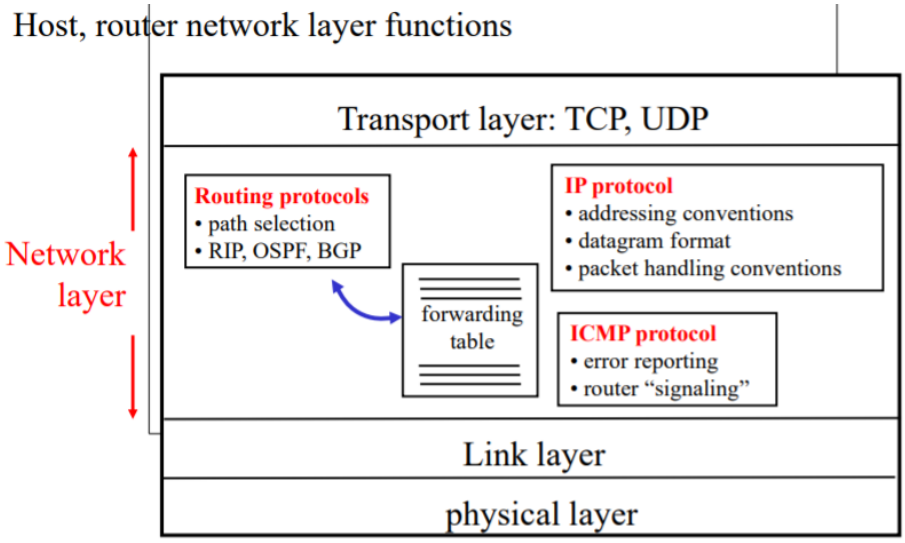


* Switching via a Crossbar
  - 2N buses
  - Possibilita transferências simultaneas de pacotes
  - a cross bar pode conter buffers intermos
  - Ultrapassa os limites da bus bandwidth

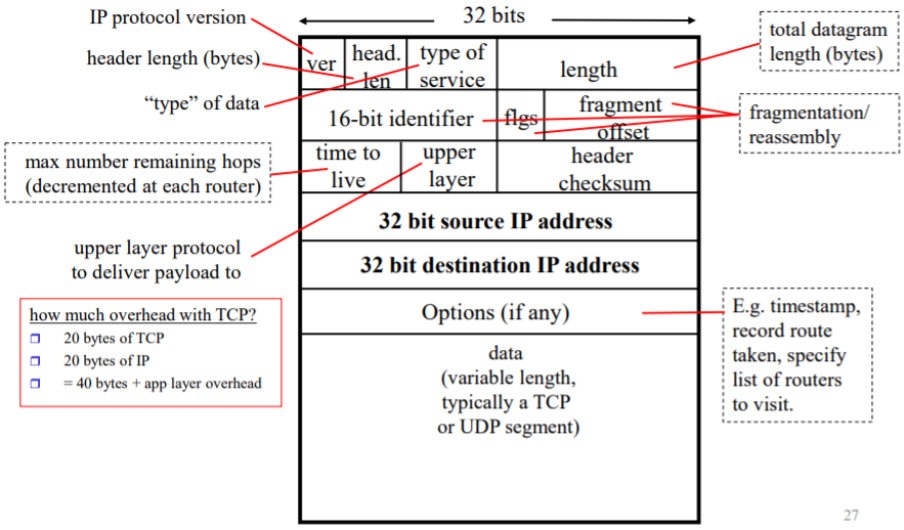


5.8 Protocolo Internet

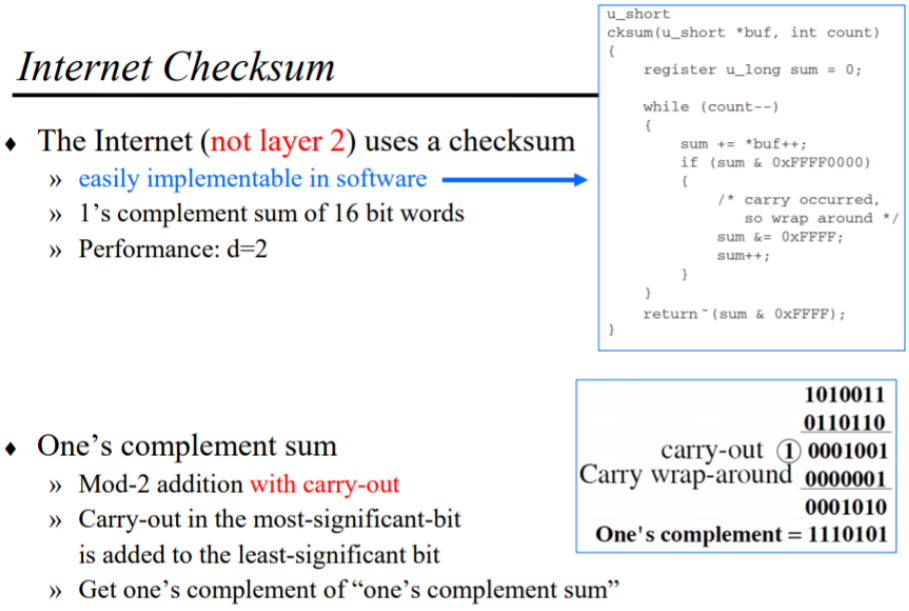
1. Camada de rede Internet



2. Formato datagrama IP



3. Internet Checksum



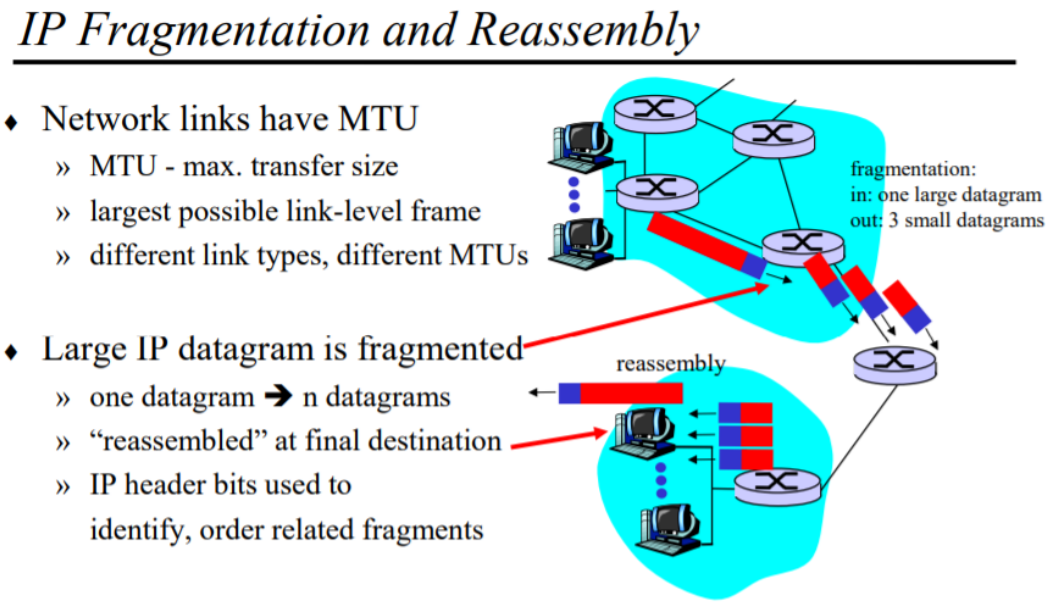
5.8.1 Cada pacote contém:

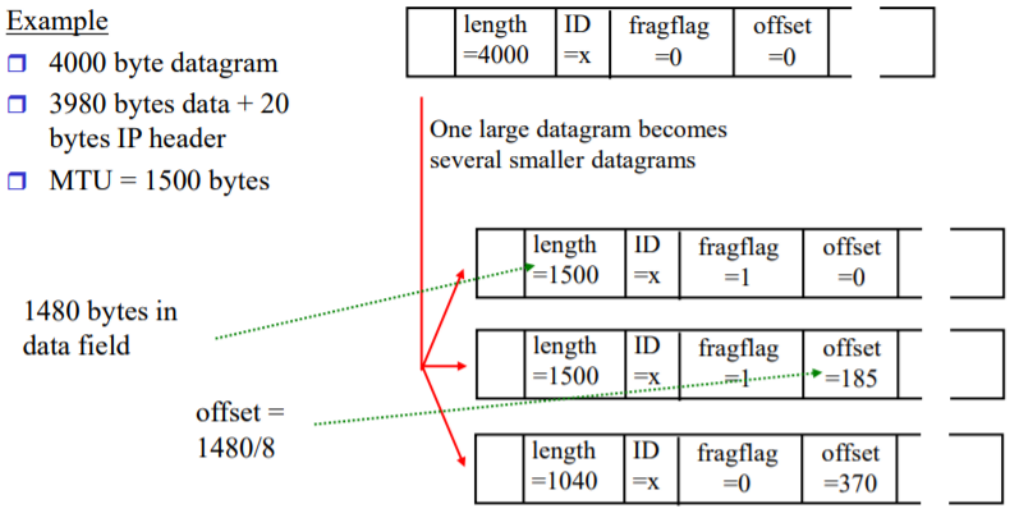
- Versão do protocolo IP

- Tamanho do Header
- Tipo de serviço
- Tamanho da informação
- Identificador + Flags + Offset de Fragmento (Permite fragmentar mensagens em vários pacotes)
- Time To Live (para os pacotes não ficarem indefinidamente perdidos na rede)
- Upper Layer Protocol
- Checksum do Header
- IP de Origem
- IP de Destino
- Opções (opcional)
- Informação (Normalmente pacote TCP ou UDP)

5.8.2 Fragmentação IP e Reassembly

- Identificador <- Identifica o pacote
- fragflag <- 1 se houver mais informação, 0 se for o último fragmento
- Offset <- Offset do fragmento em bytes / 8

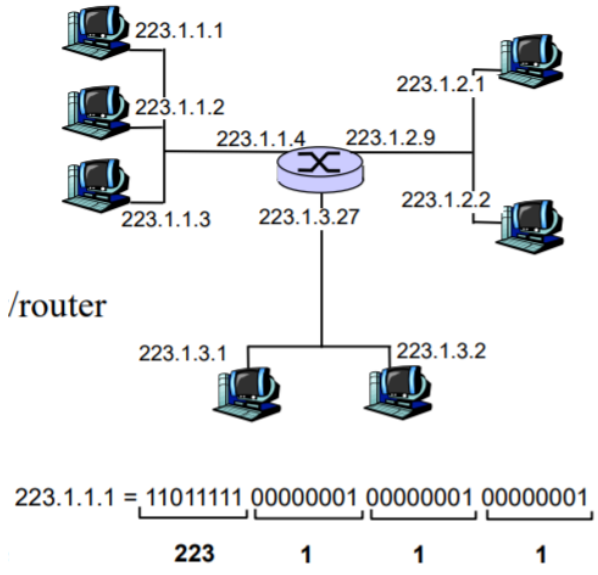




5.8.3 Endereço IP

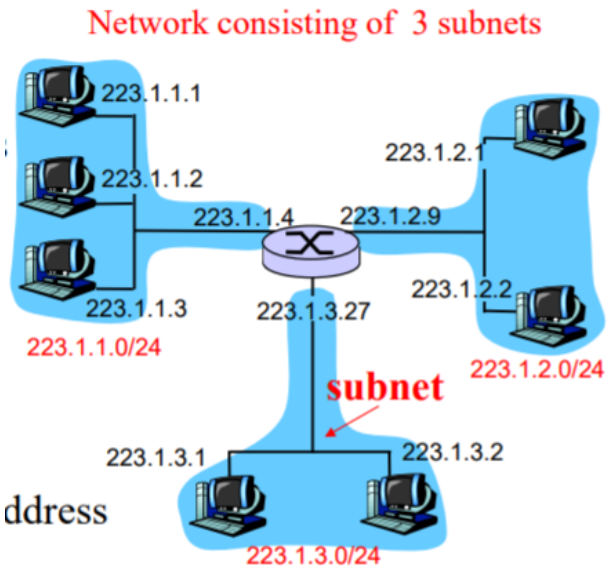
Endereço IP - é formado por um identificador de 32-bit para uma interface host/router Interface possuem:

- conecção entre host/router e link físico(physical link)
- routers com multiplas interfaces
- endereços IP associados com as interfaces



5.8.4 Subnets

- Parte mais significativa do IP: Subnet parte
- Parte menos significativa: host(interface) parte
- Subnet é um set de interfaces
- cada um tem a subnet parte do IP igual para comunicação
- Cada computador consegue aceder a outro sem intervenção do router

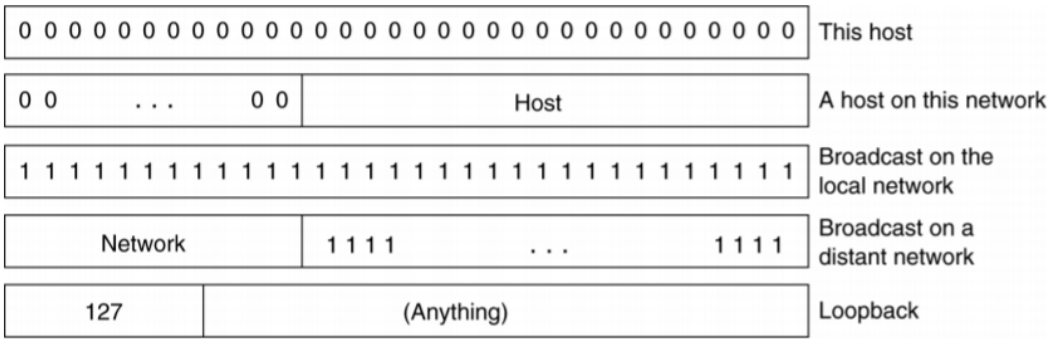


CIDR - Classless InterDomain Routing

- a porção de bits do endereço subnet tem tamanho arbitrário
- formato -> a.b.c.d/x, em que x é o número de bits na porção do endereço subnet

200.23.16.0/23

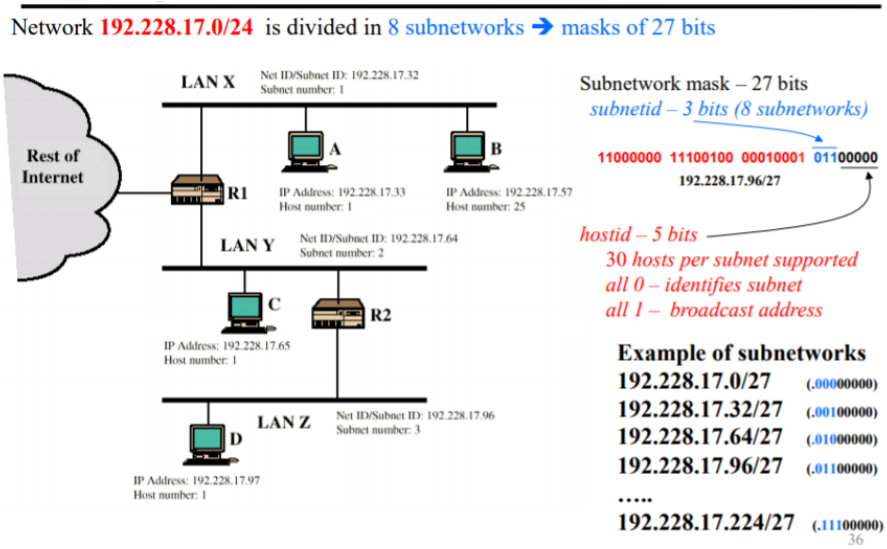
5.8.5 Endereços especiais



- 0.0.0.0 - este host
- 127.0.0.0 - loopback
- 255.255.255.255 - broadcast
- x.x.255.255 - broadcast na subnet x.x.0.0/16
- x.x.0.0 - subnet x.x.0.0/16

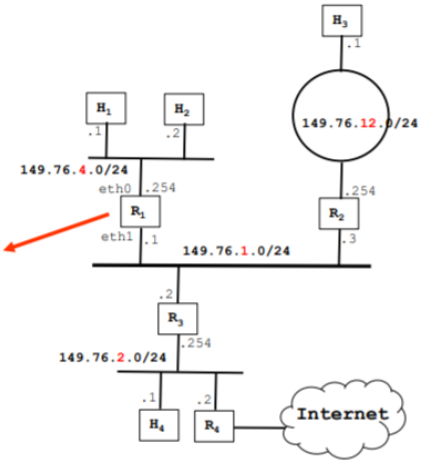
De Notar: - Uma subrede xx.xx.xx.0/24 suporta 255 endereços, no entanto, dois já estão reservados (xx.xx.xx.0 e xx.xx.xx.255), logo só suporta 253 máquinas.

Forming Sub-Networks (importante)



Criar table em R1 (importante)

Destination	Gateway	Interface
149.76.1.0/24	-	eth1
149.76.2.0/24	149.76.1.2	eth1
149.76.4.0/24	-	eth0
149.76.12.0/24	149.76.1.3	eth1
0/0	149.76.1.2	eth1



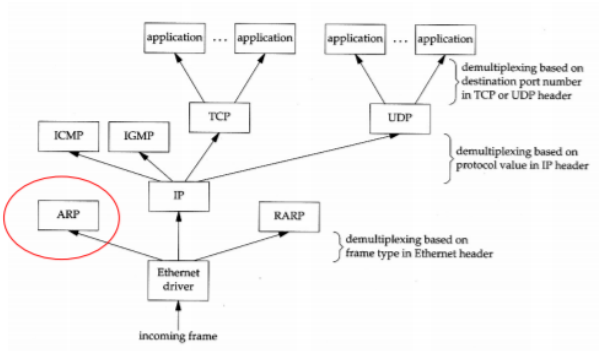
função IP forwarding (importante)

- Forwarding table has entries in format
  `<networkAddress/mask, port>`
- Forwarding function
  - » When a datagram arrives with destination address A, then
    - For each entry of the forwarding table
      - `val= A & mask*` (e.g., mask=8, mask*=255.0.0.0)
      - if `( val == networkAddress` & mask*)
        - add corresponding output port to the set of candidate ports
    - Select the port with the largest mask → most specific route
  - » Example
    - `frdTbl={<128.32.1.5/16,1>, <128.32.225.0/18,3>, <128.0.0.0/8,5>}`
    - Datagram with destination address `A=128.32.195.1`
    - Set of candidate output ports → `{1,3,5}`.
    - Selected port → 3 ← largest mask, 18 bits

5.9 Address Resolution Protocol APR

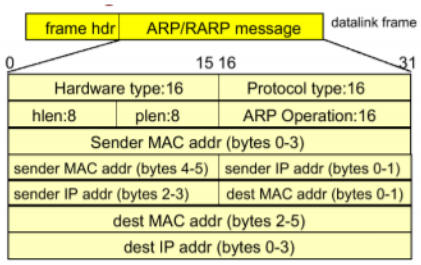
Demultiplexing

- Ethernet header (type)
  - » IP - 0x0800
  - » ARP - 0x0806
  - » RARP - 0x8035
  - » IPX- 0x8037
  - » IPv6 - 0x86DD
  - » MPLS - 0x8847
- IP header (protocol)
  - » ICMP - 1
  - » IGMP - 2
  - » TCP - 6
  - » UDP – 17
- TCP/UDP header (port)
  - » FTP – 21
  - » Telnet - 23
  - » HTTP - 80
  - » SMTP - 25

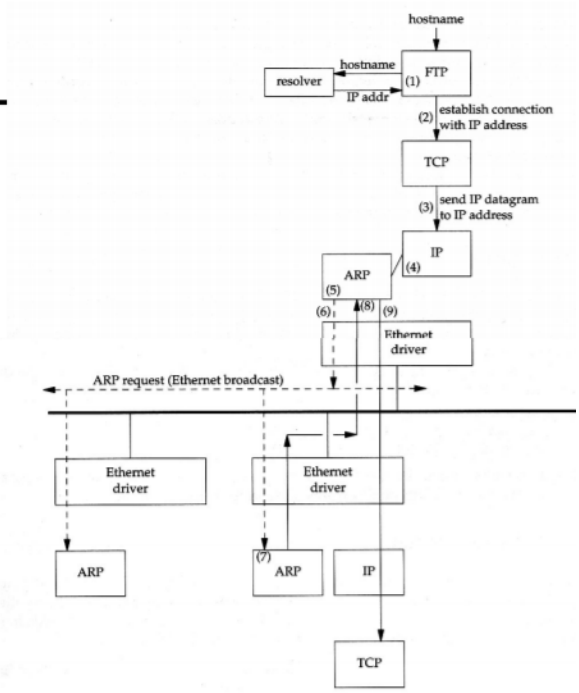


- Uma interface de rede tem 1 endereço MAC e 1 (ou mais) endereços IP
- ARP - protocolo usado para obeter o endereço MAC associado a um endereço IP dado
- RARP - reverso de ARP - protocolo usado para obter o endereço IP associado ao endereço MAC

ARP Example



- **hardware type : Ethernet=1 ARCNET=7, localtalk=11**
- **protocol type : IP=0x800**
- **hlen : length of hardware address, Ethernet=6 bytes**
- **plen : length of protocol address, IP=4 bytes**
- **ARP operation : ARP request = 1, ARP reply = 2**
  RARP request = 3, RARP reply = 4



5.10 Obter endereço IP

- Parte do endereço da subnet é definido pelo ISP

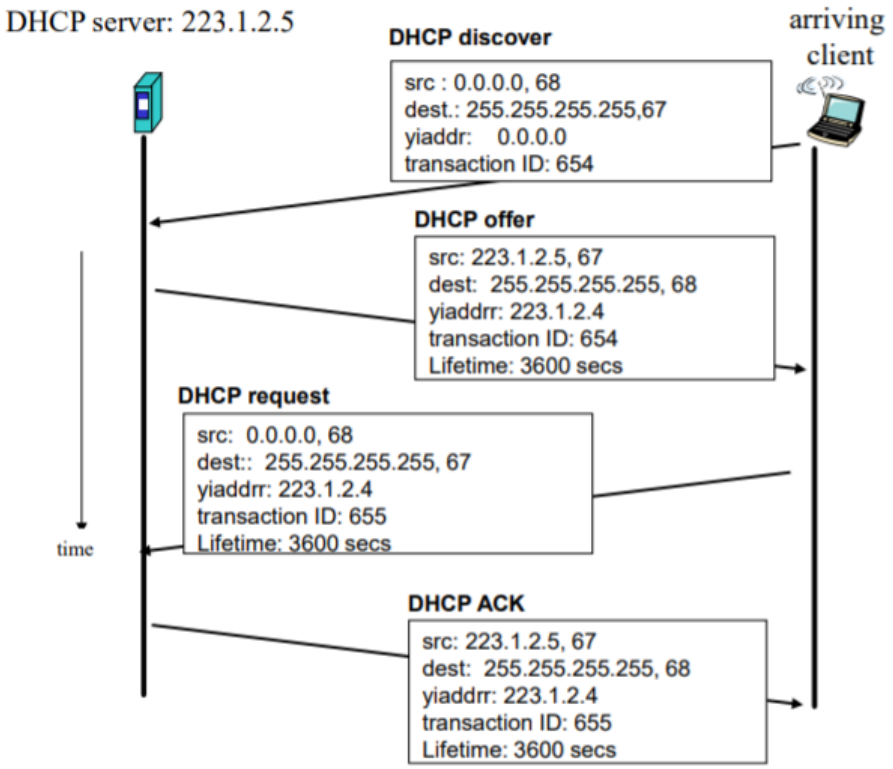
ISP's block	11001000 00010111 00010000	00000000	200.23.16.0/20
Organization 0	11001000 00010111 00010000	00000000	200.23.16.0/23
Organization 1	11001000 00010111 00010010	00000000	200.23.18.0/23
Organization 2	11001000 00010111 00010100	00000000	200.23.20.0/23
...	.....	....	....
Organization 7	11001000 00010111 00011110	00000000	200.23.30.0/23

- endereçamento hierárquico permite eficiência da informação do router
- O ISP depois trata internamente das suas subredes
- O ISP obtém endereços pela ICANN
- ICANN: Internet Corporation for Assigned Names and Numbers
  - aloca endereços

  - controla o Domain Name Service (DNS)
  - associa os nomes do domínio
  - resolve conflitos

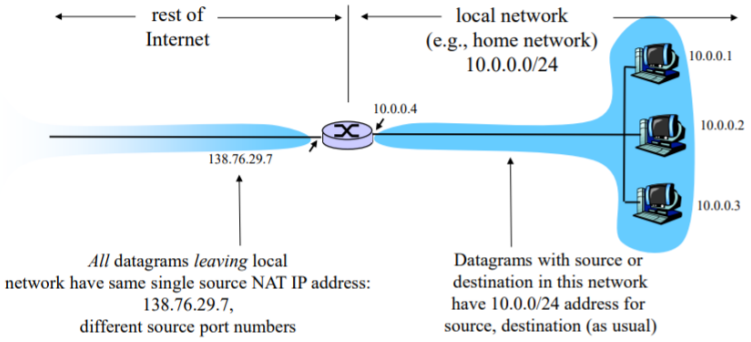
- o host obtém endereços IP de forma hard-coded pelo sistema admin num ficheiro ou pelo DHCP

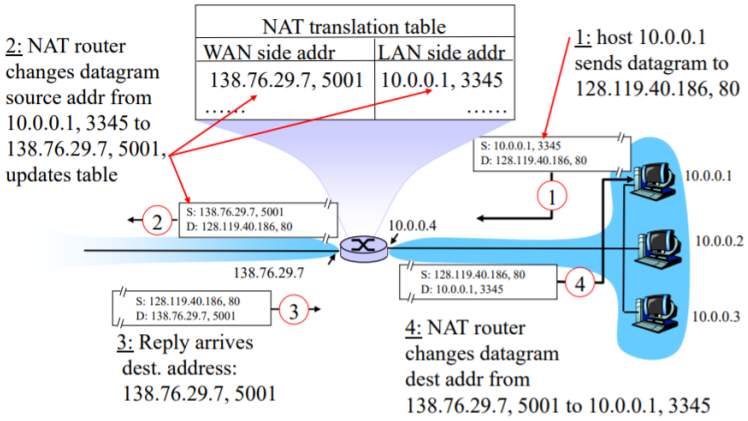
- DHCP: Dynamic Host Configuration Protocol
  - Dinamicamente recebe endereços do servidor
  - "plug-and-play"
  - permite descobrir e obter endereços da rede do servidor
  - reusa os endereços
  - Overview:
    * O host faz broadcast de "DHCP discover"(msg)
    * O servidor DHCP oferece um endereço, enviando em broadcast "DHCP offer"(msg)
    * O host pede esse endereço enviando em broadcast "DHCP request"(msg)
    * Se tudo estiver em ordem, o DHCP responde em broadcast com um "DHCP ACK"(msg)
    * Todas as mensagens entre o host e o DHCP possuem um id de transação



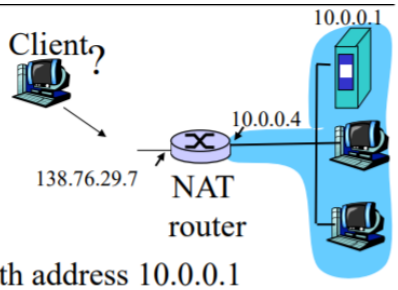
5.11 NAT - Network Address Translation

- Permite que cada computador tenha um IP interno numa rede, sendo o IP externo diferente
- Para isso, possui uma hash table a que associa um IP interno e uma porta a um número, que será a porta de saída
- Caso um cliente se queira ligar a um servidor dentro de uma rede com NAT, é necessário configurar o port forwarding





5.11.1 NAT Transversal



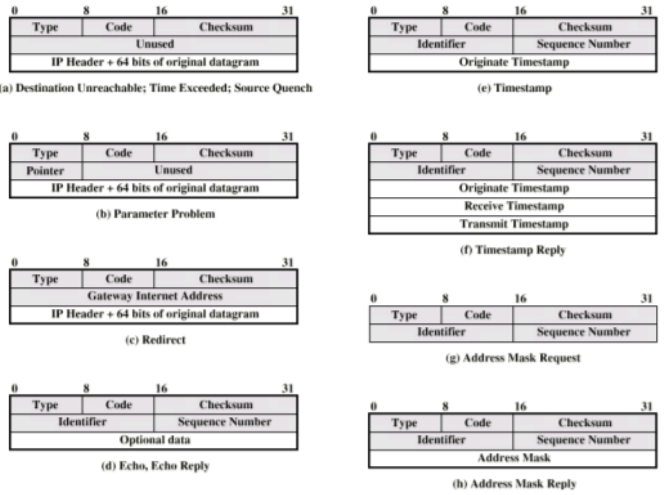
- Client wants to connect to server with address 10.0.0.1
  - » but server address 10.0.0.1 is private
  - » only one externally visible NATed address: 138.76.29.7
- Possible solution – Port forwarding
  - » statically configure NAT
    to forward incoming connection requests at given port to server
  - » e.g., (138.76.29.7, port 2500) always forwarded to 10.0.0.1 port 25000

5.12 ICMP - Internet Message Control Protocol

- Usado pelo router ou host para mandar mensagens de erro ou de controlo (como o ping)

5.12.1 IP datagramas info:

- Carried in IP datagrams



Type	Code	Description
0	0	echo reply (ping)
3	0	dest. network unreachable
3	1	dest host unreachable
3	2	dest protocol unreachable
3	3	dest port unreachable
3	6	dest network unknown
3	7	dest host unknown
4	0	source quench (congestion control - not used)
5		Redirect
8	0	echo request (ping)
9	0	route advertisement
10	0	router discovery
11	0	TTL expired
12	0	bad IP header

62

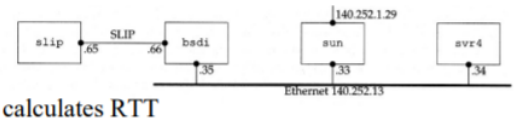
5.12.2 Tracerout and ICMP

- Permite fazer traceroute enviando mensagens com TTL=1,2,3... e esperando respostas de erro "TTL expired"até receber um "Host unreachable"

- Source sends series of UDP segments to destination
  - » first segment has TTL =1
  - » second segment has TTL=2, …
  - » unlikely port number
- When nth datagram arrives to nth router
  - » router discards datagram
  - » sends to source: ICMP TTL expired
  - » message includes router name & IP address

```
svr4% traceroute slip
traceroute to slip (140.252.13.65), 30 hops max. 40 byte packets
1 bsdi (140.252.13.35) 20 ms 10 ms 10 ms
2 slip (140.252.13.65) 120 ms 120 ms 120 ms

slip% traceroute svr4
traceroute to svr4 (140.252.13.34), 30 hops max, 40 byte packets
1 bsdi (140.252.13.66) 110 ms 110 ms 110 ms
2 svr4 (140.252.13.34) 110 ms 120 ms 110 ms
```



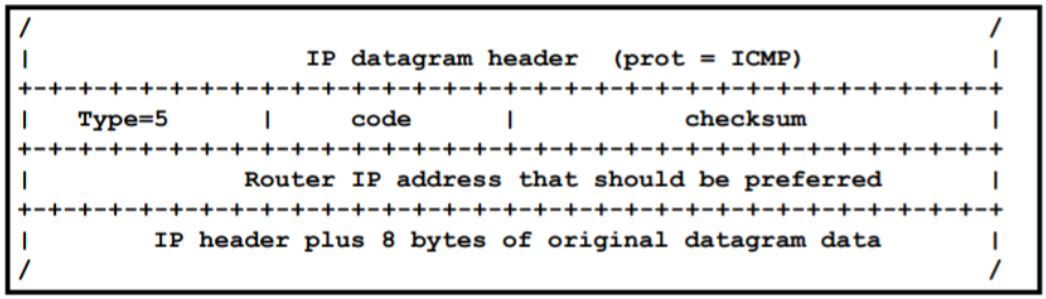
- When ICMP message arrives, source calculates RTT
- Traceroute does this 3 times for each TTL
- Stop criterion
  - » UDP segment eventually arrives at destination host
  - » Destination returns ICMP “dest port unreachable” packet
  - » source stops

5.12.3 ICMP Redirect

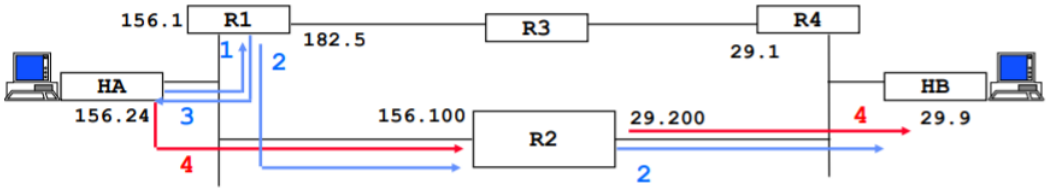
- ICMP Redirect - Permite informar outros hosts do caminho mais rápido para determinado destino

- General routing principle of the TCP/IP architecture
  - » routers have extensive knowledge of routes
  - » hosts have minimal routing information ➔ learn routes also from ICMP redirects
- ICMP redirect message
  - » Sent by router R1 to source host A
    - – when R1 receives a packet from A with destination = B, and R1
      - finds that the next hop is R2 and
      - A is on-link with R2
  - » R1 sends ICMP redirect to A saying next hop for destination B is R2
  - » A updates its forwarding table with a host route

ICMP Redirect Format



ICMP Redirect Example



	dest IP addr	srce IP addr	prot	data part
1:	193.154.29.9	193.154.156.24	udp	xxxxxxx
2:	193.154.29.9	193.154.156.24	udp	xxxxxxx
3:	193.154.156.24	193.154.156.1	icmp	type=redir code=host cksum 193.154.156.100 xxxxxxx (28 bytes of 1)
4:	193.154.29.9	193.154.156.24	udp	.....

After 4

```
HA$ netstat -nr
Routing Table:
  Destination           Gateway           Flags  Interface
-------------------- -------------------- ----- ---------
127.0.0.1            127.0.0.1             UH      lo0
193.154.29.9         193.154.156.100       UGH     eth0
193.154.156.0        193.154.156.24        U       eth0
224.0.0.0            193.154.156.24        U       eth0
default              193.154.156.1         UG      eth0
```

Flags:
U - *route Up*
G - *route to a Gateway (next hop router)*
H - *route to a Host*

5.13 IPv6

- IPv4
  - espaço reduzido de endereçamento (32 bits)
  - uso não continuo
  - o uso de algumas soluções como private networks (NAT) e classless networks (CDIR) superava os problemas acima
- IETF developed new IP version: IPv6
  - Uso dos mesmos princípios do IPv4
  - muitas melhorias
  - Header foi redefinido

5.13.1 IPv6 - Melhorias

- Endereços 128 bits (16 octets, 8 shorts). No classes
- Melhor QoS suporte (native flow level)
- funções nativas de segurança (autenticação, data encriptação)
- Autoconfiguração (Plug-n-play)
- Routing
- Multicast

5.13.2 Representação dos endereços

- 8 x 16 bit, hexadecimal, separados por:
  47CD : 1234 : 3200 : 0000 : 0000 : 4325 : B792 : 0428
- formato comprimido:
  FF01:0:0:0:0:0:0:43 -> FF01::43
- compatibilidade com IPv4:
  0:0:0:0:0:0:13.1.68.3 or ::13.1.68.3
- Loopback endereço:
  ::1
- Prefixo de rede "/", igual ao IPv4:
  FEDC:BA98:7600::/40 -> network prefix = 40 bits

5.13.3 Endereços Reservados

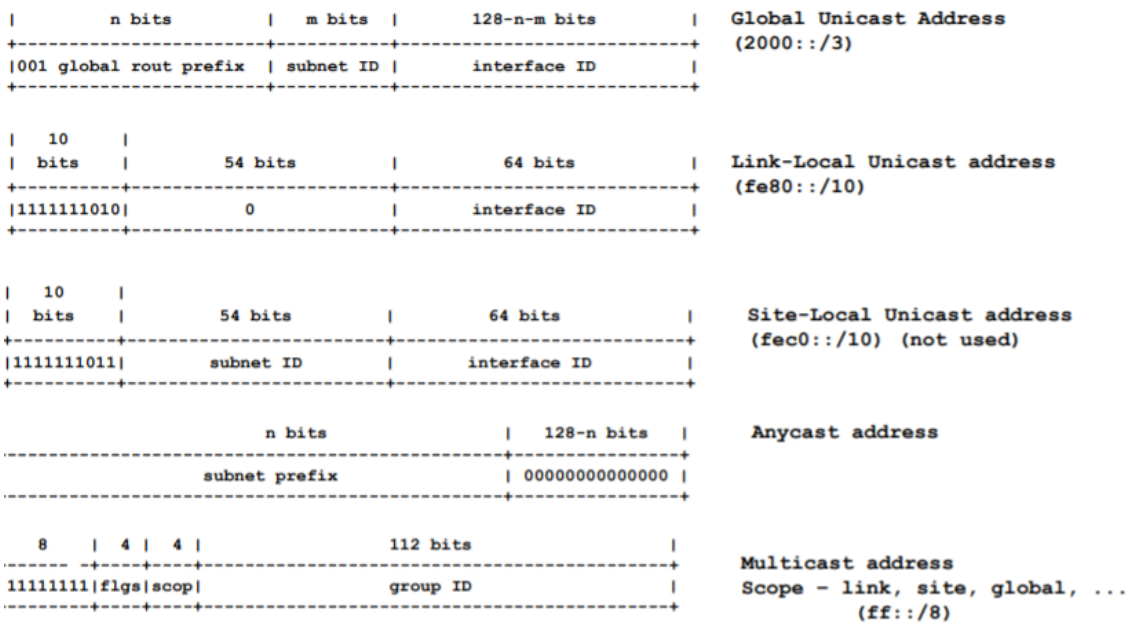
Allocation	Prefix (binary)	Fraction of Address Space
Unassigned	0000 0000	1/256
Unassigned	0000 0001	1/256
Reserved for NSAP Allocation	0000 001	1/128
Unassigned	0000 01	1/64
Unassigned	0000 1	1/32
Unassigned	0001	1/16
Global Unicast	001	1/8
Unassigned	010	1/8
Unassigned	011	1/8
Unassigned	100	1/8
Unassigned	101	1/8
Unassigned	110	1/8
Unassigned	1110	1/16
Unassigned	1111 0	1/32
Unassigned	1111 10	1/64
Unassigned	1111 110	1/128
Unassigned	1111 1110 0	1/512
Link-Local Unicast Addresses	1111 1110 10	1/1024
Site-Local Unicast Addresses	1111 1110 11	1/1024
Multicast Addresses	1111 1111	1/256

5.13.4 Tipo de Endereços

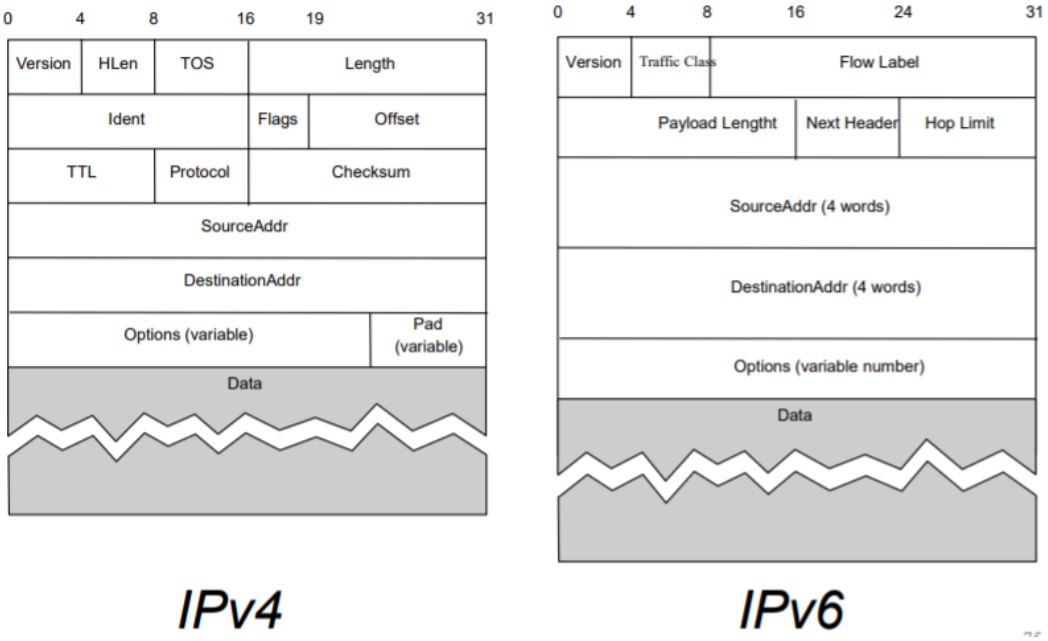
- Link-Local
  - Usado para a comunicação entre hosts na mesma LAN/link
  - Endereço criado pelo endereço MAC
  - Routers nao enviam pacotes tendo endereços de destino Link-Local
- Global Unicast

  - Endereços globais
  - Endereços: prefixo de rede + identificador do computador
  - Prefixos estruturados: Agregação de rede; menos entradas nas router forwarding tables
- Anycast
  - Endereços de grupo
  - Um pacote é recebido por um e um só membro do grupo
- Multicast
  - Endereços de grupo
  - Um pacote pode ser recebido por vários membros do grupo

5.13.5 Formato dos Endereços



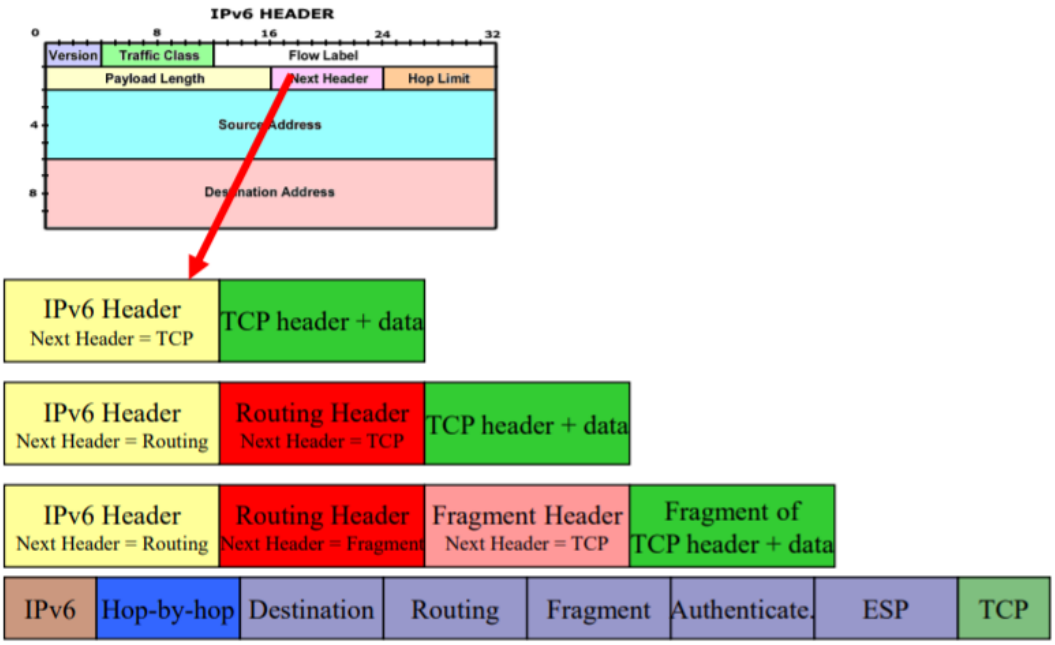
5.13.6 Headers IPv4 e IPv6



5.13.7 IPv6 Header

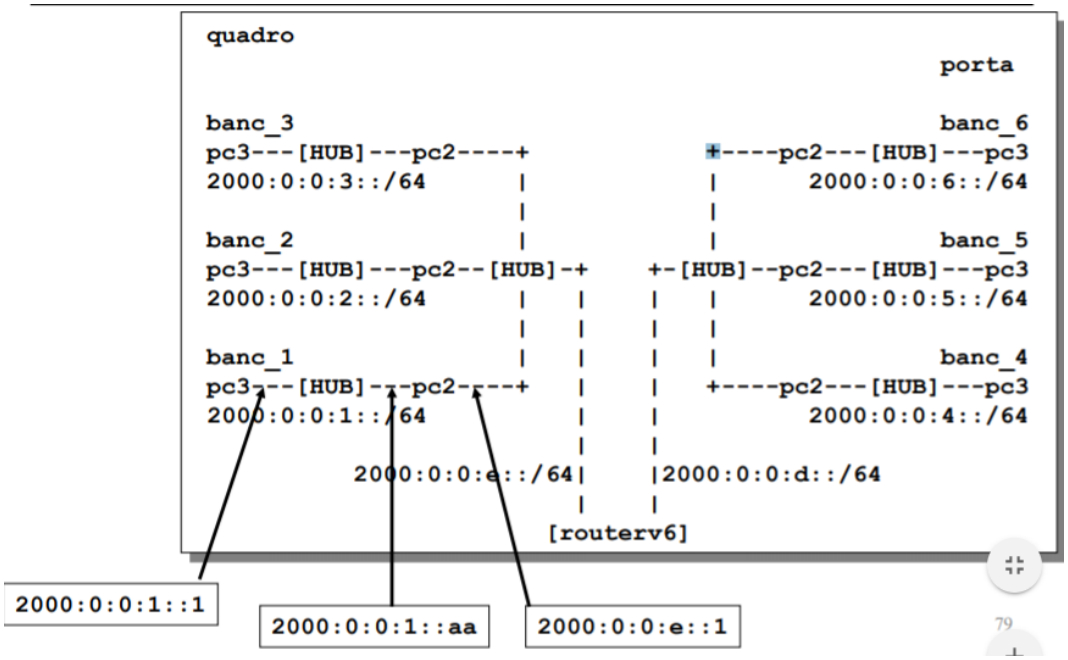
- Flow label - identifica o fluxo do pacote
  - QoS, ressalva de recursos
  - Pacotes recebem o mesmo serviço
- Payload lenght - Header não incluído
- Next header - identifica o próximo header/extensão
- Options - incluída nas extensões dos headers

5.13.8 Extension Headers



- Hop-by-Hop: inspeciona todos os nodes atravessados pelo pacote
- Destination: informação do node de destino
- Routing: Lista dos nodes para serem visitados pelo pacote
- Fragmentation: feito pelo source, deve encontrar MPU
- Authentication: autenticação (assinatura) do header do pacote
- ESP: encriptação da informação(data)

5.13.9 Exemplo da Rede do Laboratório



5.13.10 Protocol Neighbor Discovery (ND)

IPv6 node usa ND para:

- Encontrar outros nodes no mesmo link/LAN
- Encontrar o node do endereço MAC (ND substitui ARP)
- Encontrar routers na sua rede
- Manter/Segurar a informação sobre os nodes vizinhos

ND similar às funções IPv4:

- ARP IPv4
- ICMP Router Discovery
- ICMP Redirect

5.13.11 ND Mensagens

- ICMP mensagens (over IP), Uso de endereços Link Local
- **Neighbor Solicitation:** Enviado pelo host para obter o endereço MAC de um vizinho/para verificar a sua presença

- **Neighbor Advertisement:** resposta ao pedido
- **Router Advertisement:** Informação sobre o prefixo da rede, periodica ou abaixo do pedido. Enviado pelo router para o endereço IP do Link Local multicast
- **Router Solicitation:** Hosts solicitam do router uma mensagem Router Advertisment
- **Redirect:** Usado pelo router para informar o host acerca da melhor rota para o destino

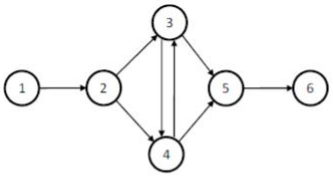
6 Transport Layer

Hello, here is some text without a meaning. This...

7 Routing

7.0.1 Graphs

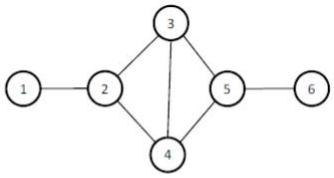
Directed:



a) Directed graph

$$
\begin{aligned}
G = \ & (V, E) \\
V = \ & \{v_1, v_2, v_3, v_4, v_5, v_6\}, && |V| = 6 \\
E = \ & \{(v_1, v_2), (v_2, v_3), (v_2, v_4), (v_3, v_4), \\
& (v_4, v_3), (v_3, v_5), (v_4, v_5), (v_5, v_6)\}, && |E| = 8
\end{aligned}
$$

Undirected:

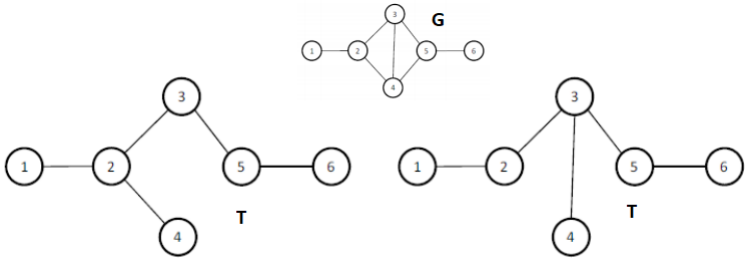


b) Undirected graph

$$
\begin{aligned}
G = \ & (V, E) \\
V = \ & \{v_1, v_2, v_3, v_4, v_5, v_6\}, && |V| = 6 \\
E = \ & \{(v_1, v_2), (v_2, v_3), (v_2, v_4), (v_3, v_4), \\
& (v_3, v_5), (v_4, v_5), (v_5, v_6)\}, && |E| = 7
\end{aligned}
$$

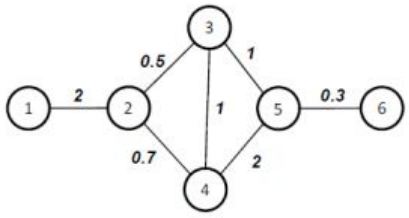
7.0.2 Tree

- Tree T = (V,E)
  - Graph with no cycles
  - |E| = |V| - 1
  - Any two V connected by only one E
- A tree T spans a graph G = (V,E) (spanning tree) if
  - T = (V,E') & E ⊆ E' (T must have the same vertices and a subset of the graph edges)



7.0.3 Shortest Path Trees

- Graphs and Trees can be weighted
  - G=(V,E,W)
  - T=(V,E',W)



- Total cost of a tree T

$$C_{total}(T) = \sum_{i=1}^{|E|}$$

(sum of all tree edges weight)

- Minimum spanning tree T^*

$$C_{total}(T^*) = min(C_{total}(T))$$

  - algorithms used to compute MST: Prism, Kruskal

- **Shortest Path Tree (SPT) rooted at vertex** s
  - tree composed by the **union of the shortest paths between** s **and each vertex of** G
  - algorithms used to compute SPT: **Dijkstra**, **Bellman-Ford**
- Computer networks use **Shortest Path Trees**

7.1 Routing in Layer 3 Networks

7.1.1 Forwarding, Routing

- **Forwarding** → data plane
  - directing packet from input to output link
  - using a forwarding table
- **Routing** → control plane
  - computing paths the packets will follow
  - routers exchange messages
  - each router creates its forwarding table

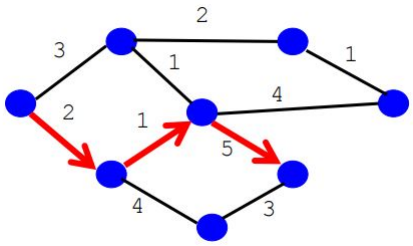
7.1.2 Importance of Routing

- End-to-end performance
  - path affects quality of service
  - delay, throughput, packet loss
- Use of network resources
  - balance traffic over routers and links
  - avoiding congestion by directing traffic to less-loaded links
- Transient disruptions
  - failures, maintenance
  - limiting packet loss and delay during changes

7.1.3 Shortest-Path Routing

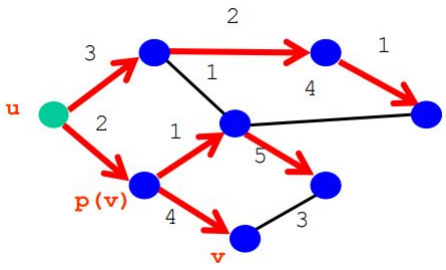
Path-selection model

- Destination-based
- Load-insensitive (ex: static link weights)
- Minimum hop count or minimum sum of link weights



7.1.4 Shortest-Path Problem

- Given a network topology with link costs
  - **c(x,y)** - link cost from node x to node y
  - ∞ if x and y are not direct neighbors
- Compute the least-cost paths from source **u** to all nodes
  - **p(v)** - node predecessor of node v in the path from u



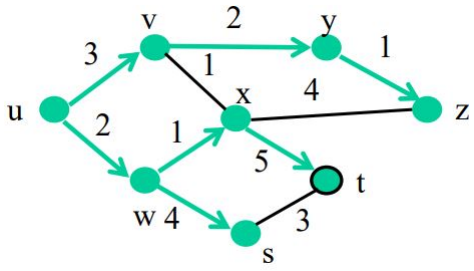
7.1.5 Dijkstra's Shortest-Path Algorithm

- Iterative algorithm
  - After k iterations → known least-cost paths to k nodes
- **S** → set of nodes for which least-cost path is known
  - Initially, **S**=**{u}**, where **u** is the source node
  - Add one node to **S** in each iteration
- **D(v)** → current cost of path from source to node **v**
  - Initially
    - **D(v)**=**c(u,v)** for all nodes adjacent to **u**
    - **D(v)**=∞ for all other nodes **v**
  - Continually update **D(v)** when shorter paths are learned

```
1  Initialization:
2    S = {u}
3    for all nodes v
4      if v adjacent to u {
5          D(v) = c(u,v) }
6      else D(v) = ∞
7
8  Loop
9     find node w not in S with the smallest D(w)
10    add w to S
11    update D(v) for all v adjacent to w and not in S:
12       D(v) = min{D(v), D(w) + c(w,v)}
13 until all nodes in S
```

7.1.6 Shortest-Path Tree

- Shortest-path tree from u



- Forwarding table at u

	link
v	(u,v)
w	(u,w)
x	(u,w)
y	(u,v)
z	(u,v)
s	(u,w)
t	(u,w)

7.1.7 Link-State Routing

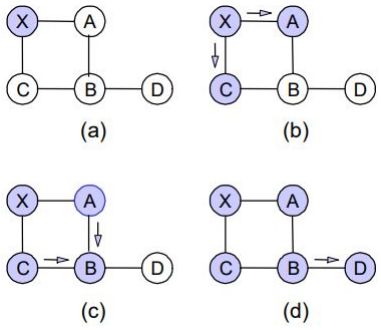
- Each router keeps track of its incident links
  - link up, link down
  - cost on the link
- Each router broadcasts link state
  - every router gets a complete view of the graph
- **Each router runs Dijkstra's algorithm**, to
  - compute the shortest paths
  - construct the forwarding table

7.1.8 Detection of Topology Changes

- Beacons generated by routers on links
  - periodic "hello" messages in both directions
  - few missed "hellos" → link failure

7.1.9 Broadcasting the Link State

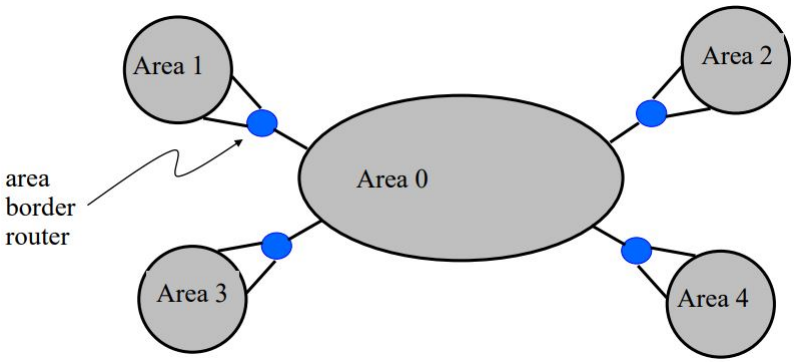
- How to Flood the link state?
  - every node sends link-state information through adjacent links
  - next nodes forward that info to all links except the one where the information arrived



- When to initiate flooding?
  - Topology change
    - link or node failure/recovery
    - link cost change
  - Periodically
    - refresh link-state information
    - typically 30 minutes

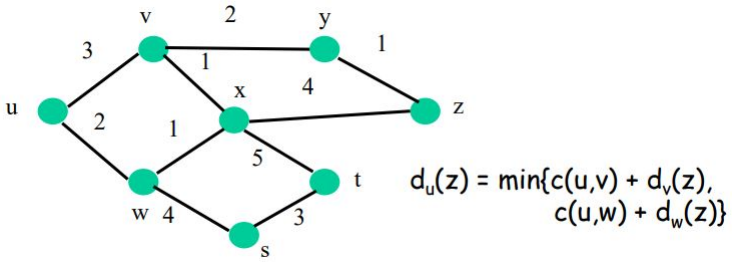
7.1.10 Scaling Link-State Routing

- Overhead of link-state routing
  - flooding link-state packets throughout the network
  - running Dijkstra's shortest-path algorithm
- Introducing hierarchy through "areas"



7.1.11 Bellman-Ford Algorithm

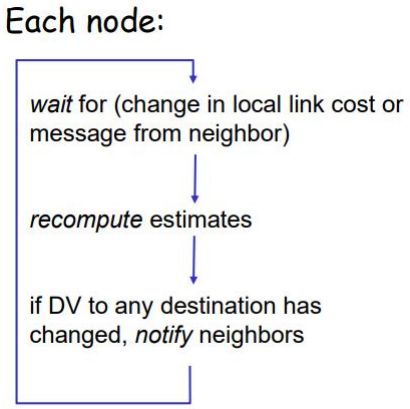
- Define distances at each node x
  - $d_x(y)$ = cost of least-cost path from x to y
- Update distances based on neighbors
  - $d_x(y)$ = min $\{c(x,v) + d_v(y)\}$ over all neighbors v



7.1.12 Distance Vector Algorithm

- c(x,v) = cost for direct link from x to v
  - node x maintains costs of direct links **c(x,v)**
- $D_x(y)$ = estimate of least cost from x to y
  - node x maintains distance vector $\mathbf{D}_x = [\mathbf{D}_x(\mathbf{y}): \mathbf{y} \in \mathbf{N}]$
- Node x maintains also its neighbors' distance vectors
  - for each neighbor v, x maintains $\mathbf{D}_v = [\mathbf{D}_v(\mathbf{y}): \mathbf{y} \in \mathbf{N}]$
- Each node v periodically sends D_v to its neighbors
  - and neighbors update their own distance vectors
  - $D_x(y) \leftarrow \min_v\{c(x,v) + D_v(y)\}$ for each node $y \in N$

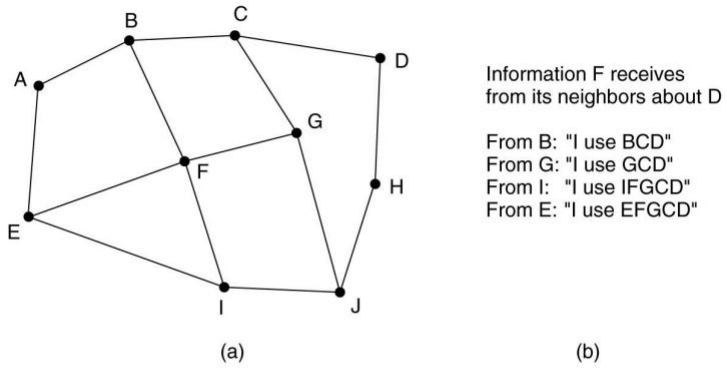
- Over time, the distance vector D_x converges
- Iterative, asynchronous, each local iteration caused by:
  - local link cost change
  - distance vector update message from neighbor
- Distributed
  - node notifies neighbors only when its DV changes
- Neighbors then notify their neighbors, if necessary



7.1.13 Routing Information Protocol (RIP)

- Distance vector protocol
  - nodes send distance vectors every 30 seconds
  - or when an update causes a change in routing
- RIP is limited to small networks

7.1.14 BGP – The Exterior Gateway Routing Protocol

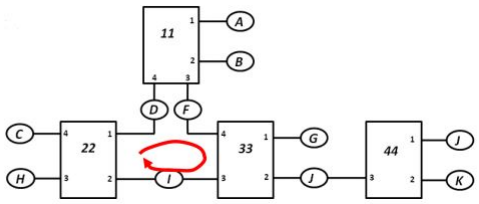


(a) A set of BGP routers. (b) Information sent to F

7.2 Unique Spanning Tree in Ethernet Networks

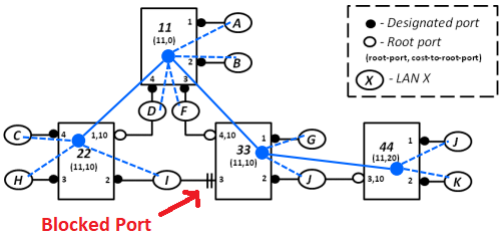
7.2.1 L2 Networking - Single Tree Required

- Ethernet frame
  - No hop-count
  - Could loop forever
  - broadcast frame, mis-configuration



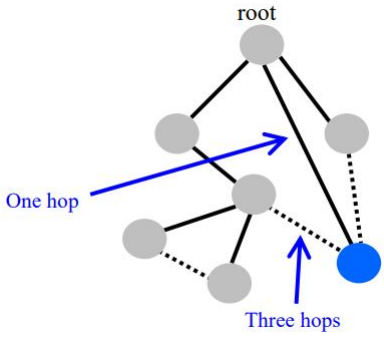
- Layer 2 network
  - **Required to have tree topology**
  - Single path between every pair of stations
- Spanning Tree Protocol (STP)
  - Running in bridges
  - Helps building the spanning tree

– Blocks ports



7.2.2 Constructing a Spanning Tree

- Distributed algorithm
  - switches need to elect a "root"
    - the switch with the smallest identifier
  - each switch identifies if its interface is on **the shortest path from the root**
  - messages(Y,d,X)
    - from node X
    - claiming Y is the root
    - and the distance is d



7.2.3 Steps in Spanning Tree Algorithm

- Initially, each switch thinks it is the root
  - switch sends a message out every interface
  - identifying itself as the root with distance 0
- Other switches update their view of the root
  - upon receiving a message, check the root id

- if the new id is smaller, start viewing that switch as root

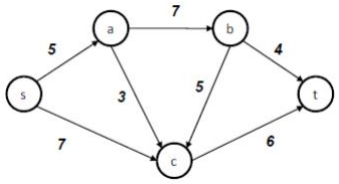
- Switches compute their distance from the root
  - add 1 to the distance received from a neighbor
  - identify interfaces not on a shortest path to the root and exclude them from the spanning tree

7.3 Maximum Flow of a Network

7.3.1 Flow Network Model

- **Flow network**
  - source s
  - sink t
  - nodes a, b and c

- Edges are labeled with **capacities (ex: bit/s)**



- Communication networks are not flow networks
  - they are queue networks
  - flow networks enable to determine limit values

7.3.2 Maximum Capacity of a Flow Network

- Max-flow min-cut theorem
  - maximum amount of flow transferable through a network
  - equals minimum value among all simple cuts of the network

- Cut → split of the nodes V into two disjoint sets S and T
  - $S \cup T = V$
  - there are $2^{|V|-2}$ possible cuts

- Capacity of cut (S,T):

$$c(S,T) = \sum_{(u,v)|u\in S, v\in T, (u,v)\in E} c(u,v)$$

  - (sum of the cost of all edges from S to T)