Redes de datos 1

Capa de enlace

Facultad de Ingeniería – Universidad de la República

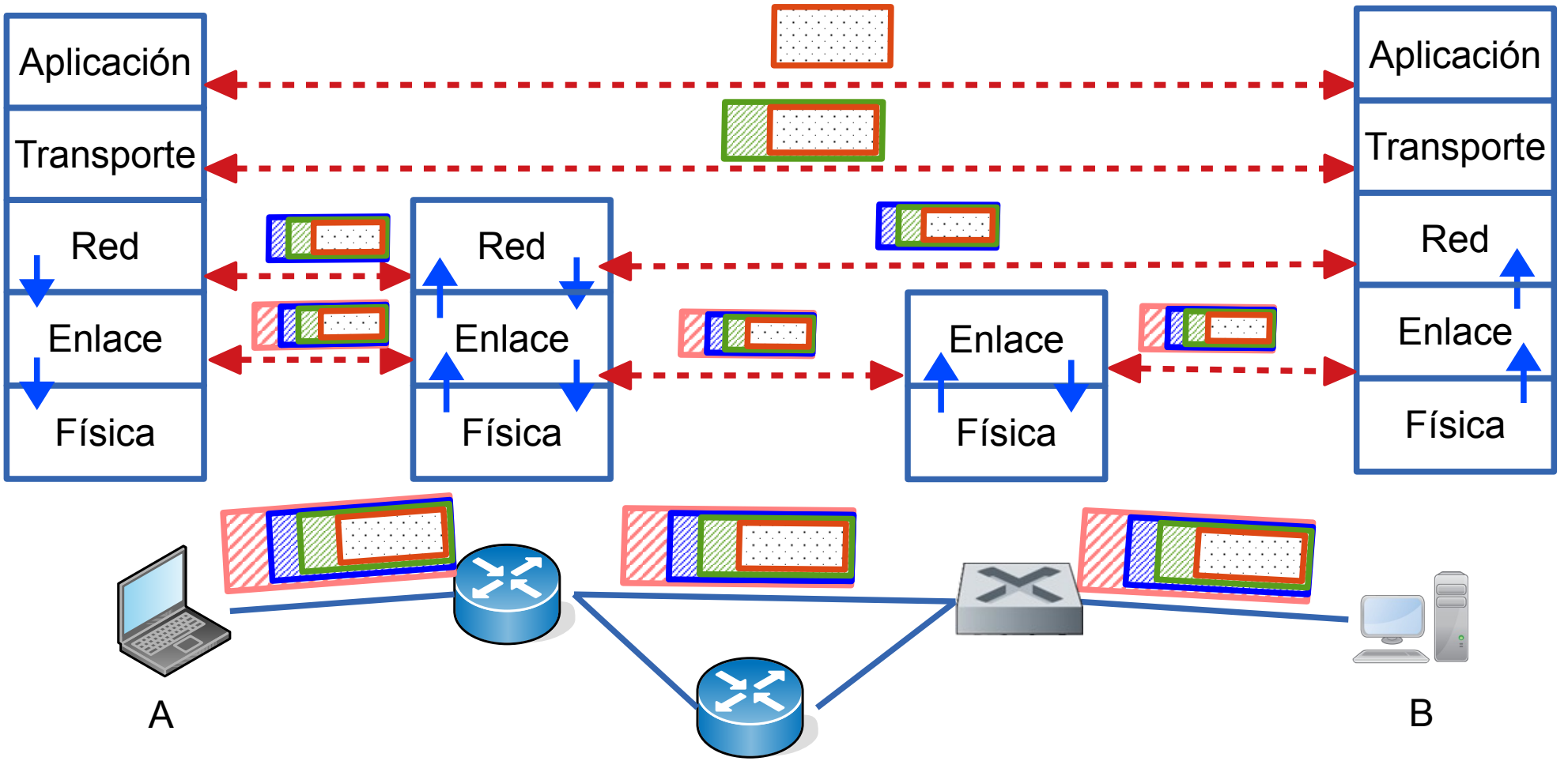
Instituto de Ingeniería Eléctrica

Agenda

- Conceptos de capa de enlace
  - Objetivo y funciones de la capa de enlace
  - Entramado
  - Detección y corrección de errores
  - Servicios confiables
- Ejemplos de protocolos punto a punto
- Protocolos de acceso a medios compartidos
  - Protocolos para compartir el canal
- Redes de área local cableadas
  - 802.3 y su evolución
  - Redes con switches
  - Vlans
- Networking en el datacenter
- Redes de área local inalámbricas

Relación con otras capas

- Brinda servicio a la capa de red
  - Le permite conectarse a través de un link a un nodo adyacente
- Utiliza los servicios de capa física para enviar los bits de la trama



Capa de Enlace

- Se encarga de controlar la comunicación entre dos máquinas “adyacentes”
- Adyacentes significa conectadas mediante un “cable” o algo que actúa como tal
- Los principales problemas a resolver son:
  - Entramado (detección de frontera entre tramas)
  - errores del medio físico
  - retardo de los canales
  - control de acceso al medio físico

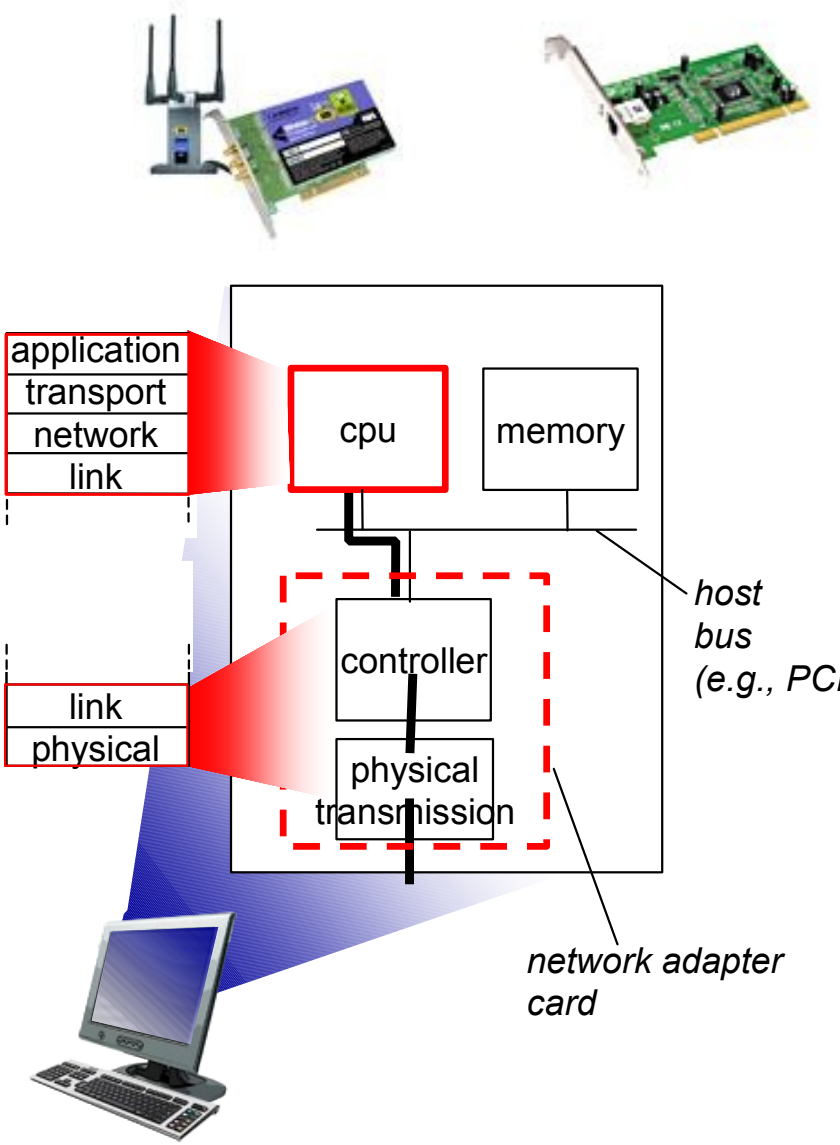
Enlaces punto a punto vs medios compartidos

- Enlaces punto a punto: exactamente 2 equipos, interconectados mediante algún medio físico
  - Todo lo que envíe será recibido por el otro nodo
  - No preciso direccionamiento ni control de acceso al canal
  - Half duplex (un solo medio) o full duplex (un canal de ida y uno de vuelta)
  - Ejemplo: enlace serial directo entre 2 equipos
- Medios compartidos: múltiples equipos pueden acceder al mismo medio
  - Ejemplo: red de área local inalámbrica (WiFi)
  - Se requiere direccionamiento para elegir el destino
  - Se requieren protocolos para ordenar el acceso al medio compartido
  - Surge una sub-capa: MAC (Medium Access Control)

Principales funciones de la capa de enlace

- Entramado
  - Permitir al receptor detectar dónde comienza y termina una trama
  - Agrupar los datos en unidades de tamaño adecuado
- Control de errores
  - Identificar si lo que recibe el receptor es lo enviado. En algunos protocolos también corrección de errores
- Control de flujo
  - Evitar que se sature un receptor lento o muy ocupado
- Control de acceso a un medio compartido
  - Permitir que múltiples equipos compartan un medio de forma ordenada y equitativa
- Direccionamiento
  - En medios compartidos, identificar el destinatario de los mensajes
- Entrega confiable
  - En caso de requerirlo

¿Dónde está implementada la capa de enlace?



- Varía de protocolo en protocolo
- Es común que varias funciones se implementen en el adaptador
  - Tarjeta independiente o embebido
  - Ejemplos: Ethernet, WiFi
- Parte se implementa en el “driver” del adaptador de red a nivel de un módulo del sistema operativo

Entramado (framing)

- La capa 2 para dar el servicio a la capa de red debe valerse de la capa física
- Como hay errores en la capa física hay que detectar y eventualmente corregir errores
- División en tramas y hacer un control de error en cada trama
- La división en tramas no es tan sencilla
- Múltiples métodos de entramado. Algunos ejemplos:
  - Conteo de caracteres
  - Caracteres de delimitación
  - Banderas de delimitación
  - Violaciones del código de línea de la capa física
  - Otros
- La capa de enlace le agrega a la trama un encabezado y posiblemente un trailer (bits al final del mensaje), y le aplica algún método de entramado

Conteo de caracteres

- Se indica el largo de cada trama
- Para saber dónde comienza la siguiente trama, se cuentan los caracteres

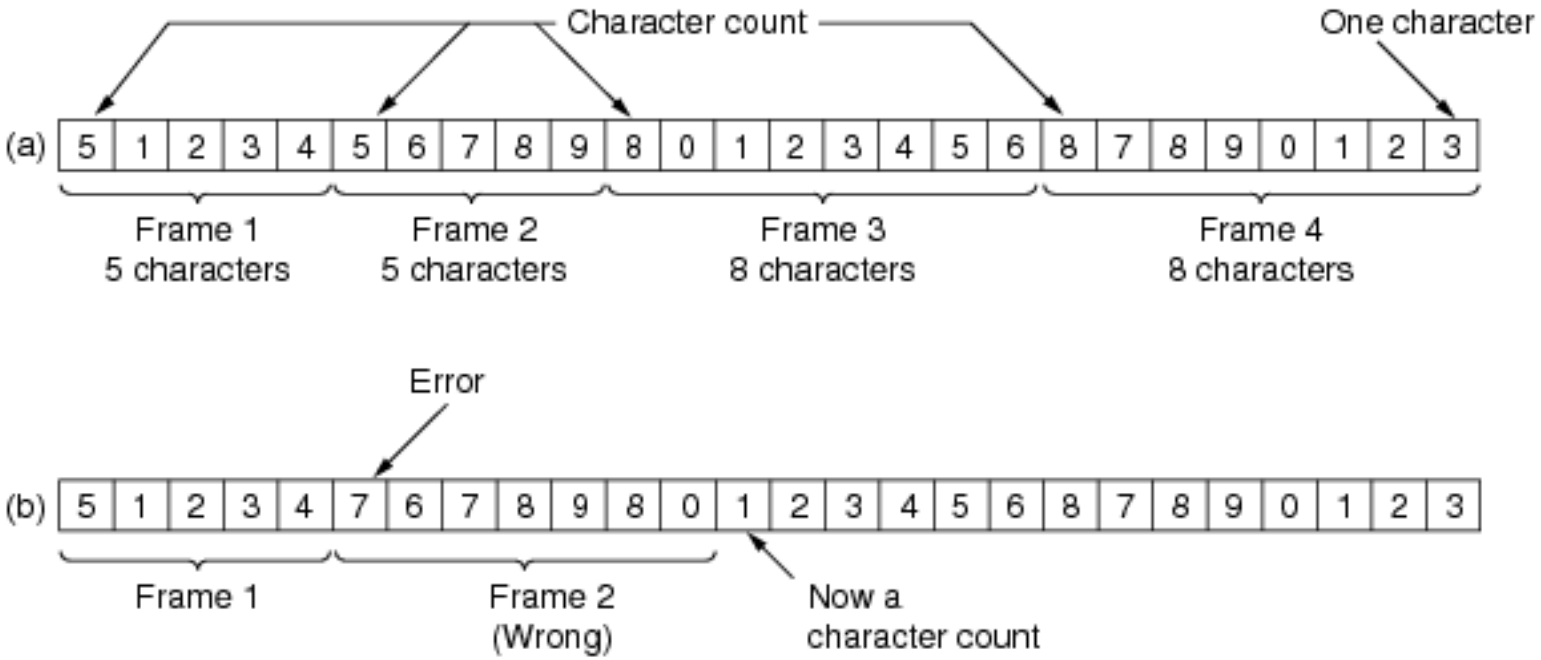


Fig. 3-4. A character stream. (a) Without errors. (b) With one error.

Conteo de caracteres (cont.)

- Problema: ¿qué pasa si se recibe con error el campo de “largo” de la trama?
  - Si se corrompe el encabezado de la trama se pierden las fronteras de tramas
  - Difícil resincronizar
    - Por ejemplo, se agregan secuencias de bits predecibles en el encabezado, para poder buscarlas en caso de requerir resincronizar
  - Poco usado como mecanismo único (si en conjunto con otros mecanismos)

Caracteres de principio y fin

- Pensado en la época en que se transmitía mayoritariamente texto ASCII
- Insertar banderas para delimitar el comienzo y fin de la trama
- Comúnmente se delimita con caracteres especiales (de control)

  Por ejemplo en ASCII hay caracteres reservados para esto:

  - DLE STX = bandera de comienzo
  - DLE ETX = bandera de fin
- Se transmite:

- El receptor detecta el principio y fin de trama buscando esa secuencia de caracteres
- ¿Qué pasa si aparece DLE STX (o DLE ETX) entre los datos de capa de red?

** DLE: Data Link Escape. STX: Start Transmission. ETX: End Transmission

Inserción de caracteres para lograr transparencia

- Recibido de la capa de red en el Tx:

- Inserción de caracteres:

- Inserción de banderas (lo que se transmite):

- En el receptor se procesa de modo que:
  - DLE STX = bandera de comienzo
  - DLE DLE = se saca un DLE
  - DLE ETX = bandera de fin
- Se entrega a la capa de red en el Rx:

Inserción de caracteres. Problemas

- Muy atado a transmisión de caracteres de 8 bits
- Optimizado para contenido ASCII
- Alto overhead si aparece DLE múltiples veces

Banderas e inserción de bits

- Misma idea que inserción de caracteres pero con banderas definidas como patrones de bits
- Bandera típica 01111110 = 7E hexadecimal
- Se debe evitar que la bandera aparezca en los datos
- Si la bandera se da en los datos, se insertan bits de relleno
- En transmisión:
  - Para asegurar que no se transmiten seis bits con valor “1” seguidos, se inserta un “0” siempre luego que aparezcan 5 “1” seguidos
    - Sin importar lo que venga después!!!
  - Se insertan las banderas y se envía al receptor
- En recepción:
  - Se reconocen las banderas 01111110
  - Si se ven 5 “1” y un “0” se saca el “0”
    - Sin importar lo que venga después!!!

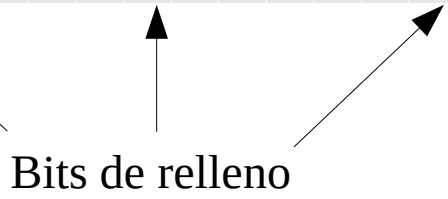
Banderas e inserción de bits

- Datos de capa 3:

0 1 1 0 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 0 0 1 0

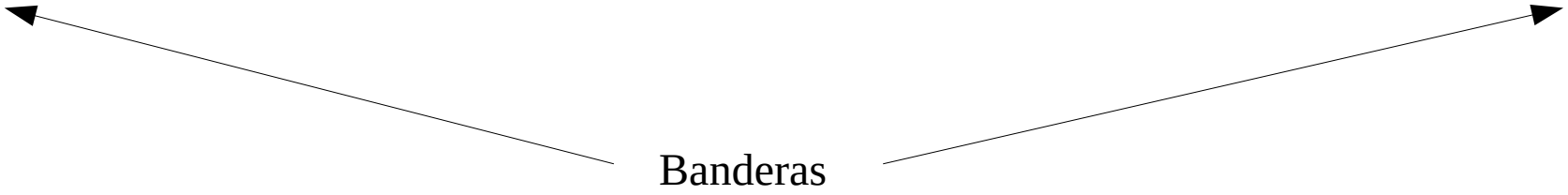
- Datos a enviar:

0 1 1 0 1 1 1 1 1 **0** 1 1 1 1 1 **0** 1 1 1 1 1 **0** 1 0 0 1 0



- A esta secuencia se agregan antes de enviar las banderas de comienzo y fin

0 1 1 1 1 1 1 0 0 1 1 0 1 1 1 1 1 **0** 1 1 1 1 1 **0** 1 1 1 1 1 **0** 1 0 0 1 0 0 1 1 1 1 1 1 0

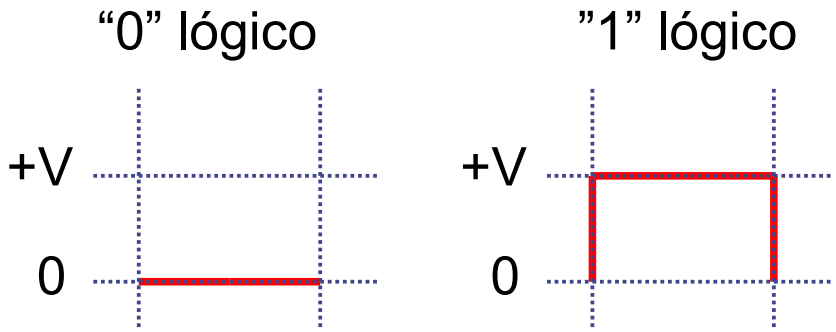


Violaciones del código de línea

- En capa física se utilizan símbolos (típicamente combinaciones de amplitud/frecuencia/fase u otras magnitudes) para representar uno o más bits
- Podemos reservar algún símbolo de capa física para representar el inicio y/o fin de trama
- También podemos utilizar alguna combinación inválida de magnitudes con esta misma función
- Problema: viola la independencia de capas

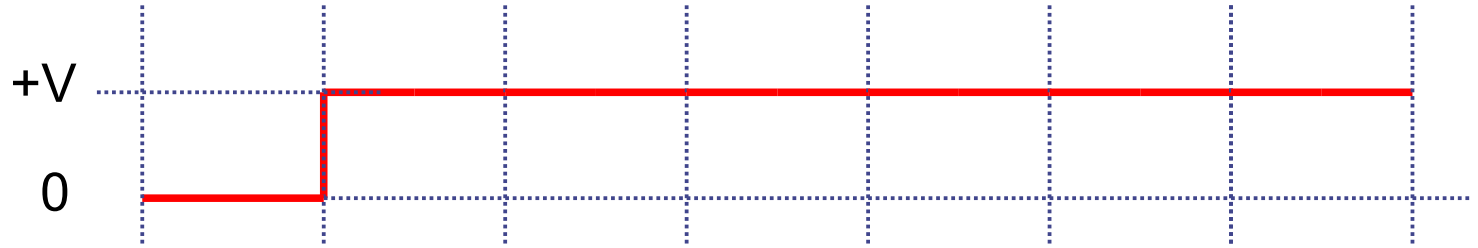
Código de línea

- Ejemplo:



- Secuencias de muchos “1” o muchos “0” seguidos

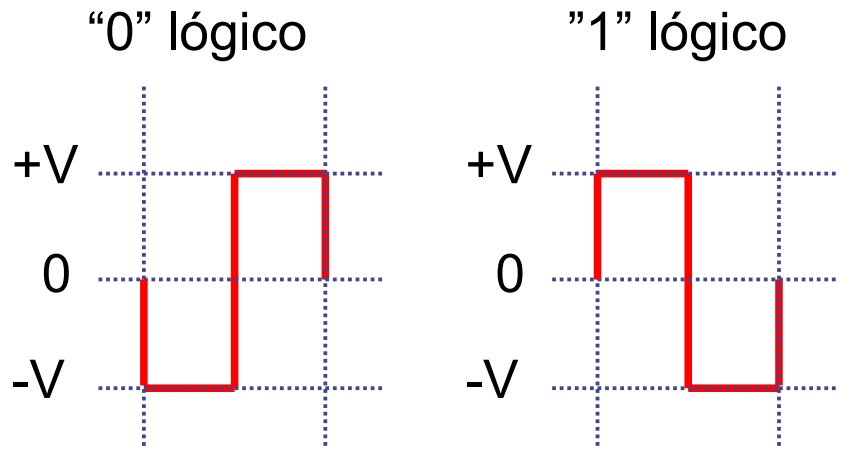
0	1	1	1	1	1	1



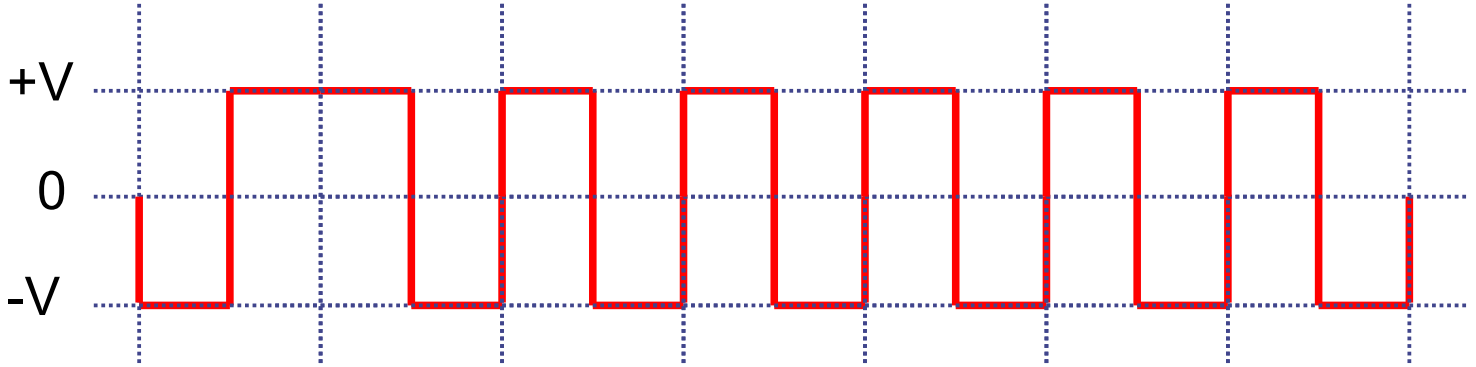
- Generan señales:
  - que no tienen nivel de continua 0
  - en las que no se puede recuperar el reloj (las fronteras de bits)

Código de línea

- Pero si uso por ejemplo (código Manchester):



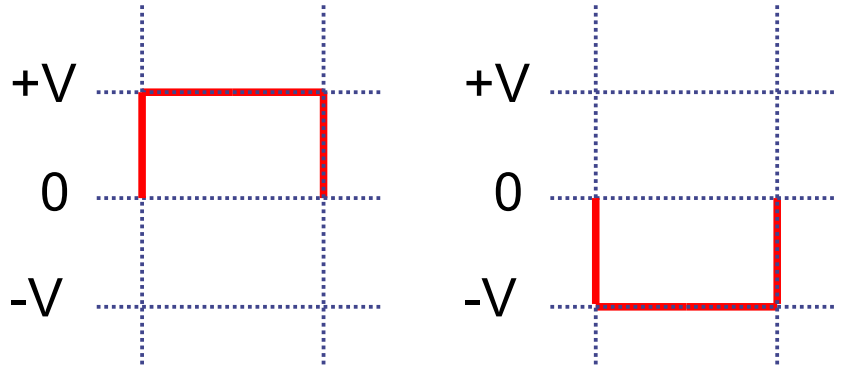
- La misma secuencia queda: 0 1 1 1 1 1 1



- Que tiene nivel de continua 0 y flancos de reloj que permiten sincronizarse

Violaciones del código de línea

- Pero aparecen dos combinaciones que no representan ni un “0” ni un “1” y podrían usarse para delimitar tramas:



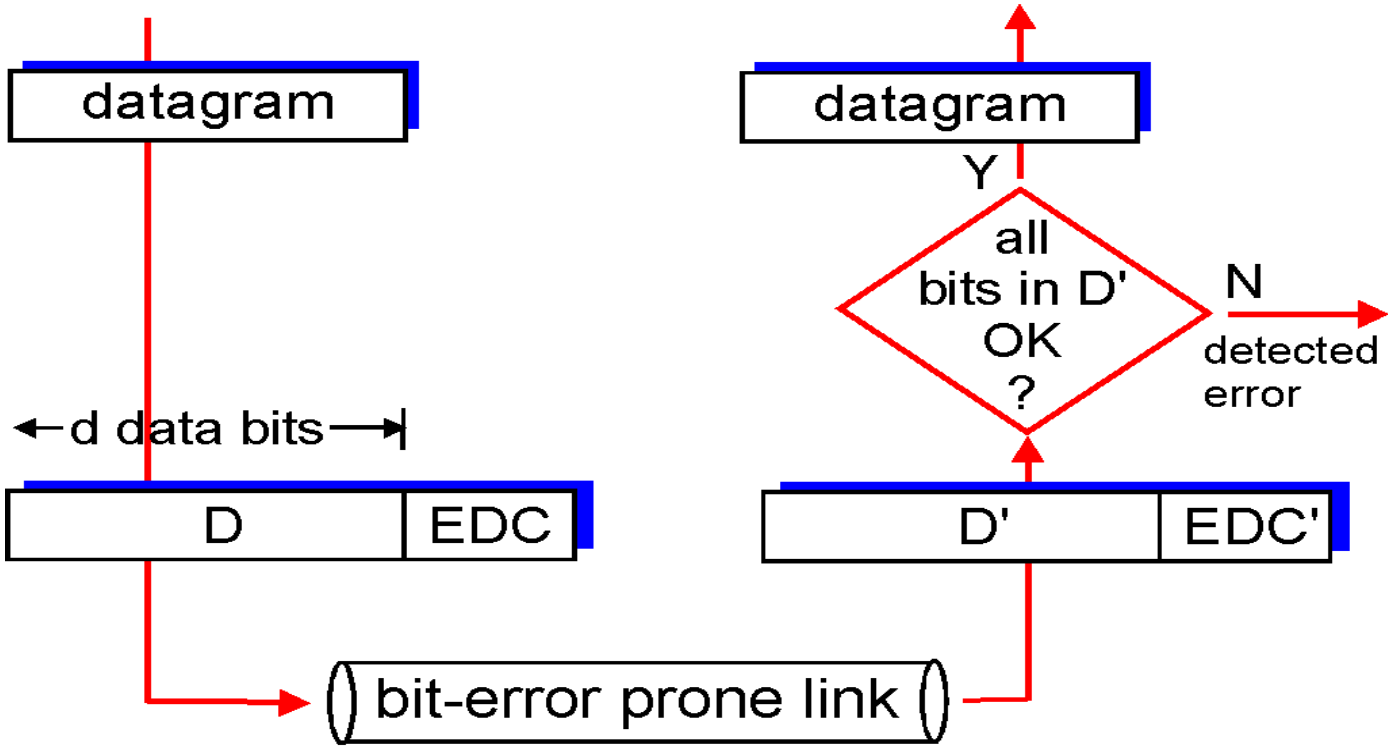
- Por ejemplo Ethernet a 10 Mbps utiliza código manchester, y un mecanismo de este estilo para determinar el inicio y fin de la trama

Detección y corrección de errores

- ¿Qué son errores?
  - La señal que representa los bits de la trama, al ser transmitida por la capa física, sufre diversas modificaciones (atenuación, deformación, interferencia, ruido, etc)
  - El receptor puede interpretar erróneamente uno o mas bits del mensaje
- Distintas capas físicas tienen distintas probabilidades de error
- También pueden variar las características de los errores (por ejemplo errores en ráfagas vs. errores independientes)
- Se agrega redundancia para intentar detectar (o eventualmente corregir) estos errores
- Ningún mecanismo puede detectar el 100% de los errores

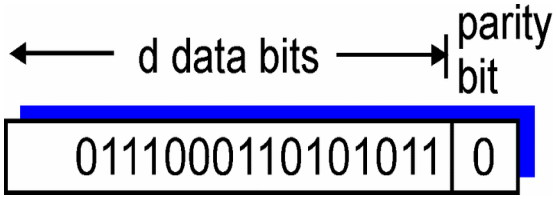
Detección de errores

- Dado un mensaje de largo *d* bits, se le agregan *r* bits de redundancia o chequeo. Estos *n=d+r* bits son los que se transmiten
- El receptor recibe los *n* bits, y verifica la redundancia
- Si detecta un error descarta el mensaje, en caso contrario pasa la carga útil a la capa superior.



Ejemplo: bit de paridad

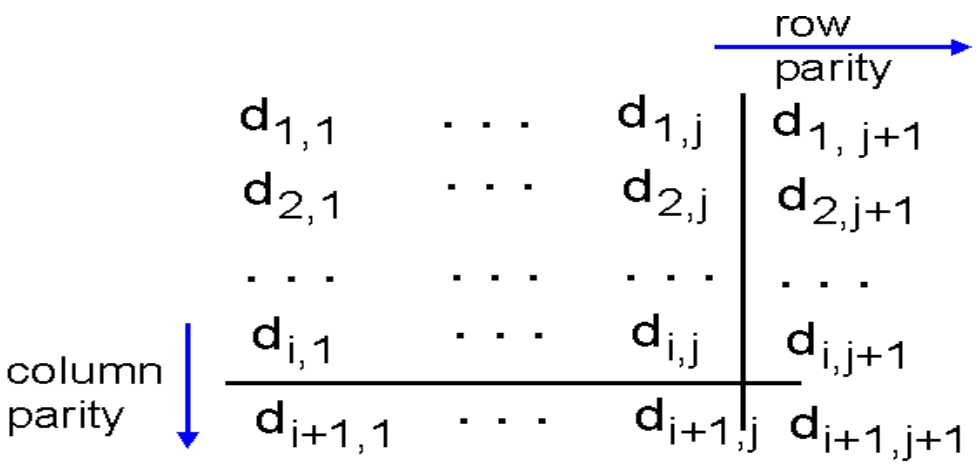
- Se agrega 1 bit (paridad)
- Paridad par: transmisor elige el bit de paridad de forma que la cantidad de bits en 1 sea par (idem paridad impar)
- En el receptor se verifica la paridad
  - En caso de error se descarta
- Detecta todos los errores de 1 bit
- Observar que detecta todos casos con un número impar de bits errados
  - Y ninguno con número par

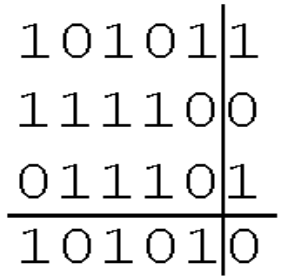


Paridad impar

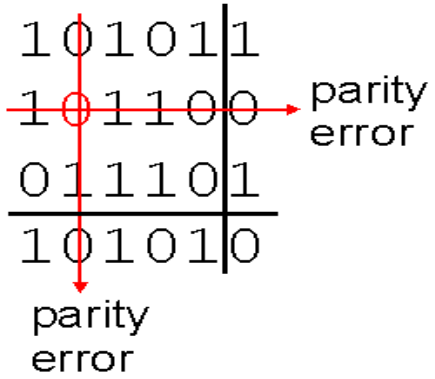
Ejemplo: generalización de paridad en forma matricial

- Bits se organizan en una matriz
- Se calcula bit de paridad para cada fila y columna
- Si asumimos un solo bit errado, podemos determinar qué bit es el errado y corregirlo
  - FEC: Forward Error Correction





no errors



correctable single bit error

Sumas de comprobación

- Se tratan los datos a enviar como enteros de k bits
- Se realiza una cuenta sobre estos enteros
- Ejemplo: suma de comprobación en TCP
  - Se trata los datos como una sucesión de enteros de 16 bits
  - Se realiza la suma
  - Se envía el complemento a 1 de dicha suma
  - En el receptor se verifica
- Fácil de calcular en software
- Bajo overhead
- Protección débil contra errores

Códigos de Redundancia Cíclica (Códigos polinomiales)

- CRC: Cyclic Redundancy check Code (código de redundancia cíclica)
  - También llamados códigos polinomiales
- Muy utilizado en capa 2 (ejemplos: Ethernet, WiFi)
- Mayor capacidad de detección de errores, especialmente errores en ráfagas
- El cálculo es más complejo en software que una suma de comprobación
  - Pero no es un problema si se implementa en hardware
  - Ni es un problema en procesadores modernos

Códigos polinómicos (CRC)

- Se tratan los bits del mensaje como coeficientes de un polinomio
- Si tengo un mensaje de k bits, $c_{k-1}c_{k-2}...c_0$ lo puedo ver como un polinomio de grado k-1
  - $c_{k-1}x^{k-1} + c_{k-2}x^{k-2} + ... + c_0x^0$
- Ej: 110001 se representa como $x^5+x^4+x^0$
- La aritmética se hace en módulo 2, no hay acarreos y tanto la suma como la resta son idénticas al XOR (or exclusivo)
- El transmisor y receptor deben ponerse de acuerdo en el uso del llamado polinomio generador G(x) (usualmente determinado por el protocolo)
- Los coeficientes más y menos significativos de G(x) deben ser 1
- El mensaje de m bits se representa como D(x), m debe ser mayor que el largo de G(x)
- La idea es agregar una suma de comprobación al final de la trama de modo tal que el polinomio representado por el conjunto sea divisible entre G(x)
- El receptor divide lo que recibe entre G(x), si el resto es 0 no hay errores. Si es distinto de 0 es porque hubo errores en la transmisión

Algoritmo para cálculo del CRC de forma polinomial

- Si r es el grado de G(x), agrego r bits en 0 en la parte menos significativa de la trama. Lo que tengo entonces es la representación de

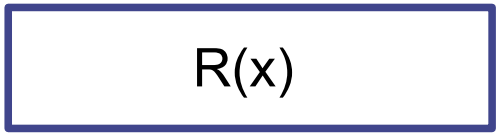
  $x^r\ D(x)$

- Ejemplo:
  - Si tengo la secuencia **110001** que es el polinomio $D(x) = x^5+x^4+1$
  - Y agrego 3 ceros a la derecha, queda **110001000**
  - Quedaría el polinomio: $\mathbf{1}.x^8+\mathbf{1}.x^7+\mathbf{0}.x^6+\mathbf{0}.x^5+\mathbf{0}.x^4+\mathbf{1}.x^3+\mathbf{0}.x^2+\mathbf{0}.x^1+\mathbf{0}.x^0$
  - O sea: $x^8+x^7+x^3$
  - O sea: $(x^5+x^4+1) x^3$
  - O sea: $x^3\ D(x)$
- Divido $x^r\ D(x)$ entre G(x) con aritmética módulo 2 y obtengo un resto R(x) (con menos de r bits)
- Resto R(x) a $x^r\ D(x)$ obteniendo $M(x)= x^r\ D(x)-R(x)$ que obviamente es divisible entre G(x)
- Se envía un mensaje cuyos bits son los coeficientes de M(x)

Detalles de implentación del CRC

- Observar que $x^r D(x)$ tendrá esta forma:

- Y el resto R(x) tendrá la siguiente forma:



- Por lo que la resta para obtener $M(x)= x^r D(x)-R(x)$ se implementa simplemente concatenando los bits de D(x) con los bits de R(x)

Procesamiento en el receptor

- El receptor recibe un mensaje, M'(x) = M(x) + E(x)
- E(x) representa los errores introducidos en el canal. Tenemos un bit 1 en la posición de cada bit que se haya invertido
- El receptor divide (M(x)+E(x))/G(x) y calcula el resto de la división
- Como:

  Resto[(M(x)+E(x))/G(x)] = Resto[M(x)/G(x)] + Resto[E(x)/G(x)]

  y Resto[M(x)/G(x)] = 0

  => el resultado es Resto[E(x)/G(x)]

- El receptor aceptará como válidos los mensajes que cumplan que el resto calculado es cero (es decir, que son divisibles entre G(x))
- Solo se escapan los patrones de error que correspondan a un polinomio divisible entre G(x)

Códigos polinómicos

- El problema pasa a ser entonces elegir adecuadamente el polinomio G(x)
- Se hace en base a propiedades de los polinomios, por ejemplo:
  - Para detectar error simple: $E(x)=x^i$ (*)
    - se necesita que G(x) tenga al menos dos términos
  - Para detectar dos errores: $E(x)=x^i+x^j=x^j(x^{i-j}+1)$
    - Si G(x) no es divisible por x (condición *)
    - se necesita que G(x) no divida a x^{k+1} para cualquier k<i-j (largo de la trama)
  - etc.

Polinomios generadores estandarizados

- Múltiples polinomios generadores estandarizados. Algunos ejemplos:
  - CRC-16 (16 bits) $x^{16}+x^{15}+x^{2}+1$
  - CRC-CCITT (16 bits) $x^{16}+x^{12}+x^{5}+1$
  - IEEE 802 CRC-32 (32 bits)
    $x^{32}+x^{26}+x^{23}+x^{22}+x^{16}+x^{12}+x^{11}+x^{10}+x^{8}+x^{7}+x^{5}+x^{4}+x^{2}+x^{1}+1$
- Los de 16 bits detectan:
  - 100% simples y dobles
  - 100% los de número impar de bits
  - 100% de ráfagas de largo 16 o menos
  - 99.99% de ráfagas de 17 bits o más bits
- El de 32 bits:
  - 100% simples, dobles y triples
  - 100% los de número impar de bits
  - 100% de ráfagas de largo 32 o menos
  - 99,99999998% de los errores de 4 bits, y otras propiedades

Corrección de errores

- La capacidad de detectar y **corregir** errores en el receptor se denomina “Forward Error Correction” (FEC)
- Requieren un overhead sustancialmente mayor que los códigos detectores de error
- No estudiaremos ninguno de ellos en particular.
- Se utilizan:
  - En medios con alta tasa de error, donde se justifica el overhead para evitar una retransmisión
  - En enlaces con un delay muy grande que hace impráctica la retransmisión
  - Otras aplicaciones, como el almacenamiento de datos en discos magnéticos, CDs y DVDs, donde la “retransmisión” no es posible

Entrega confiable

- Algunos protocolos de capa de enlace ofrecen el servicio de entrega confiable
- ¿Por qué tener entrega confiable en capa de enlace, si la tenemos en capa de transporte?
  - Evitar la retransmisión end to end para links con alta probabilidad de errores
  - Capas de transporte sin entrega confiable
  - Etc.
- Se utilizan los mismos principios que vimos en capa de transporte
  - Ventanas deslizantes, reconocimientos
  - Repetición selectiva o go back N
- Muchos protocolos sobre enlaces con baja tasa de errores (fibra, par trenzado) no implementan servicios de entrega confiable
- Algunos protocolos lo implementan parcialmente: se implementan reconocimientos (garantiza entrega) pero sin ventanas deslizantes (pueden generar duplicados por pérdida del reconocimiento)

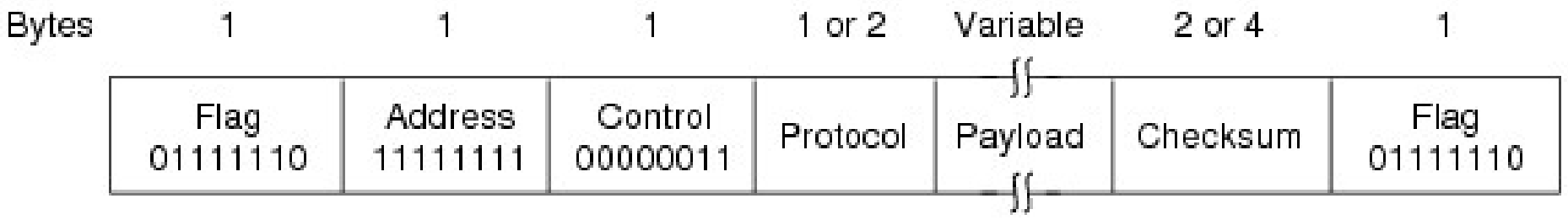
Control de flujo

- El control de flujo en capa de enlace cumple la misma función que en capa 4
  - Pero entre equipos adyacentes, no extremo a extremo
- Se implementa de la misma manera que en capa de transporte, utilizando ventanas deslizantes
- En los protocolos de redes de área local (Ethernet, WiFi) no se suele utilizar control de flujo
  - En versiones de Ethernet de velocidades de 1 Gbps o superiores, existe una opción de “solicitud de pausa” para cumplir esta función

Ejemplo de protocolo de capa de enlace: PPP

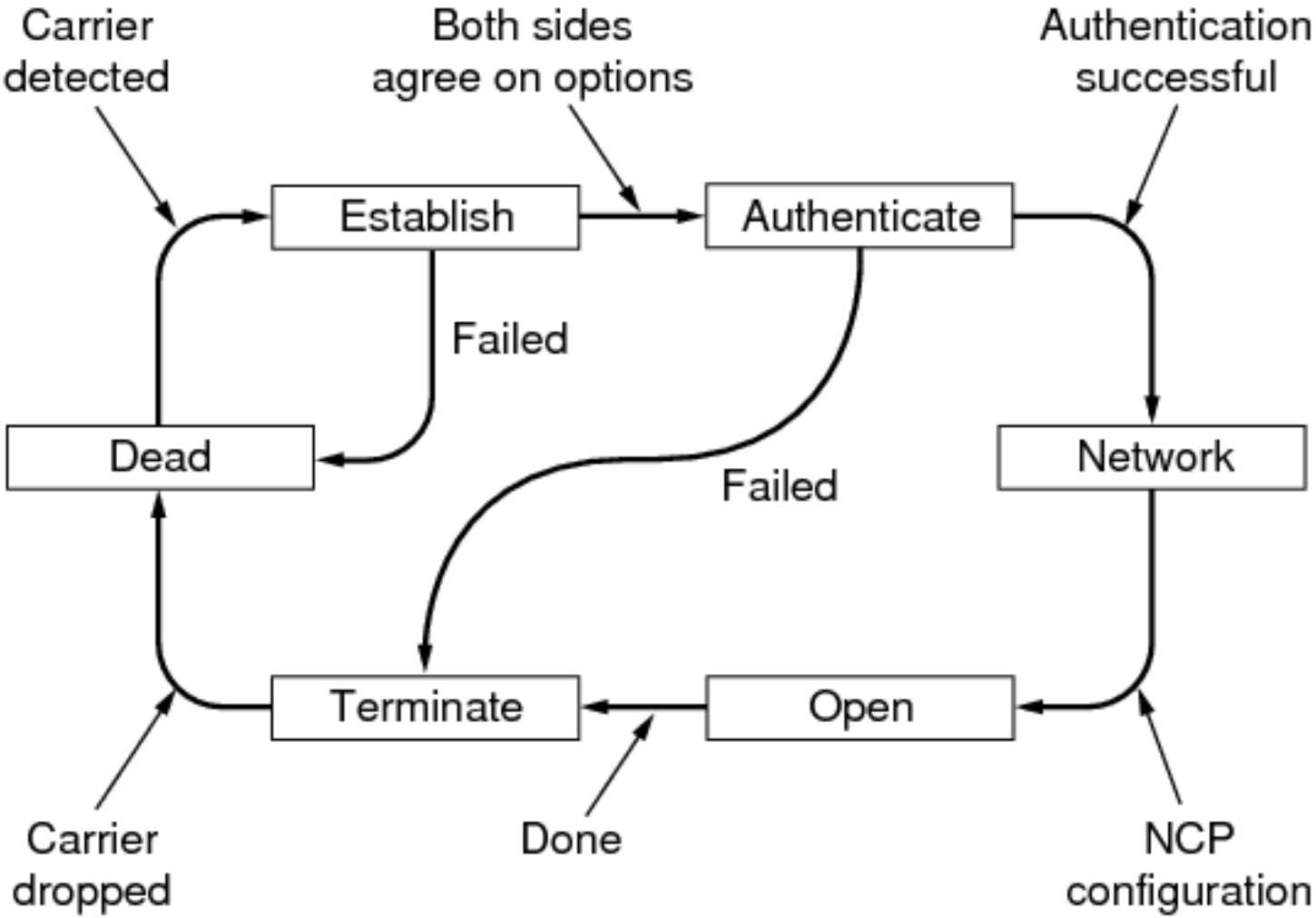
- PPP (Point to Point Protocol): protocolo para enlaces punto a punto
- Provee:
  - Entramado (framing) con detección de error
  - LCP (Link Control Protocol): para conectarse, testear la línea, negociar opciones y terminar una conexión
  - NCP (Network Control Protocol): una manera de negociar opciones a nivel de capa de red (por ej. la dirección de capa 3)

- Formato de trama:



- Utiliza relleno de bits si aparecen secuencias de más de 5 “1” en el payload

PPP: Diagrama de estados simplificado



Capa de enlace
Sub capa de acceso al medio

Facultad de Ingeniería – Universidad de la República

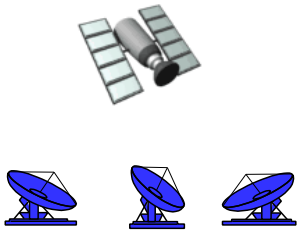
Instituto de Ingeniería Eléctrica

Sub Capa MAC (Medium Access Control)

- Aparece cuando tenemos un medio donde múltiples estaciones pueden transmitir (y recibir) tramas
  - También se les suele llamar redes broadcast (una trama enviada llega a todos los receptores)
- Nuevos problemas a resolver:
  - Coordinar el acceso al medio
  - Seleccionar el destino
- Algunos ejemplos:

Cable compartido (ej., Ethernet tradicional)

RF compartido (ej., 802.11 WiFi)



RF compartido (satélite)

Personas en una fiesta (aire compartido, acústico)

Sub capa MAC

- Objetivo (original): compartir un mismo canal entre múltiples estaciones
  - Eficientemente
  - De forma sencilla
  - “económicamente”
- Requeriremos:
  - Direccionamiento de las estaciones
  - Mecanismo para compartir el canal
  - En general detección de errores

Direccionamiento en capa MAC

- Necesidad: separar el tráfico de diferentes estaciones
  - Solamente tomar “mis” tramas
  - La tarjeta descartará el tráfico no dirigido a mi
  - Ahorro procesamiento en la CPU
- Los switches podrán enviar la trama al puerto adecuado
- Dirección asociada a la interfaz
- Solo significado local al link

- Distintos protocolos elegirán distintas direcciones
- En general, división en direcciones unicast (una estación), multicast (un grupo) y broadcast (todas las estaciones)
- En los protocolos LAN desarrollados por la IEEE, direcciones MAC de 48 bits
  - Las veremos luego

¿Por qué otras direcciones?

- ¿Por qué no compartir las direcciones con la capa de red?
- Independencia de capas
  - ¿Las direcciones de cual protocolo de red?
  - ¿Cómo manejo 2 protocolos de red sobre la misma capa MAC?
- Direccionamiento global (capa 3) versus local (capa 2)
  - Distintos objetivos requieren soluciones distintas

2 tipos de medios broadcast

- Canales compartidos físicamente:
  - Estaciones utilizan el mismo medio físico
  - Posibilidad de “interferencia” (señales superpuestas)
  - Ejemplo: redes inalámbricas, 802.3 original
- Canales compartidos lógicamente:
  - A nivel físico las estaciones tienen enlaces punto a punto con un concentrador o switch
    - No hay posibilidad de “interferencia”
  - Pero a nivel lógico cada estación puede enviar tramas a cualquier otra (o a todas) las que comparten el dominio de broadcast
  - Ejemplo: ethernet switcheada (la veremos luego)
  - En este caso también tenemos una capa MAC

Características deseables en un protocolo MAC

- Dado un canal de R bits por segundo:
  - Cuando un solo nodo quiere transmitir, puede utilizar toda la capacidad de R bps
  - Si M nodos quieren transmitir, cada uno puede hacerlo a una velocidad promedio de R/M bps
  - El protocolo es descentralizado
    - No requiere un nodo especial para coordinar las transmisiones
    - No requiere sincronización de relojes ni de timeslots
  - Simple

Taxonomía de los protocolos de capa MAC

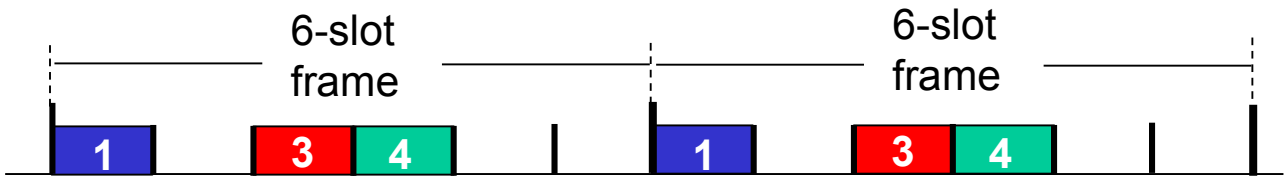
- Hay 3 grandes clases:
  - Particionado del canal:
    - Se divide el canal en fracciones (frecuencia/tiempo/código)
    - Se asigna una fracción a cada nodo para su uso exclusivo
  - Acceso aleatorio:
    - Canal no se divide
    - Se permiten colisiones
    - Debemos poder “recuperarnos” de las colisiones
  - “Nodos se turnan”
    - Los nodos toman turnos para transmitir
    - Los nodos que tienen más para transmitir pueden tomar turnos más largos
- Además de las clases anteriores, podemos considerar una “sub-clase” donde los nodos se conectan a un dispositivo, “switch”, que se encarga de reenviar las tramas entre segmentos.

Colisiones

- Cuando dos tramas se superponen en el tiempo (en el mismo canal), se dice que hay una colisión
- El receptor recibe la superposición de varias señales, que no podrá distinguir
- Las tramas que colisionan son irrecuperables. No es posible determinar las tramas originales observando el canal
- El receptor se da cuenta ya sea porque a nivel de capa física/MAC se violan las restricciones del protocolo, o porque no verifica la suma de comprobación (CRC)

Protocolos MAC particionando el canal: TDMA

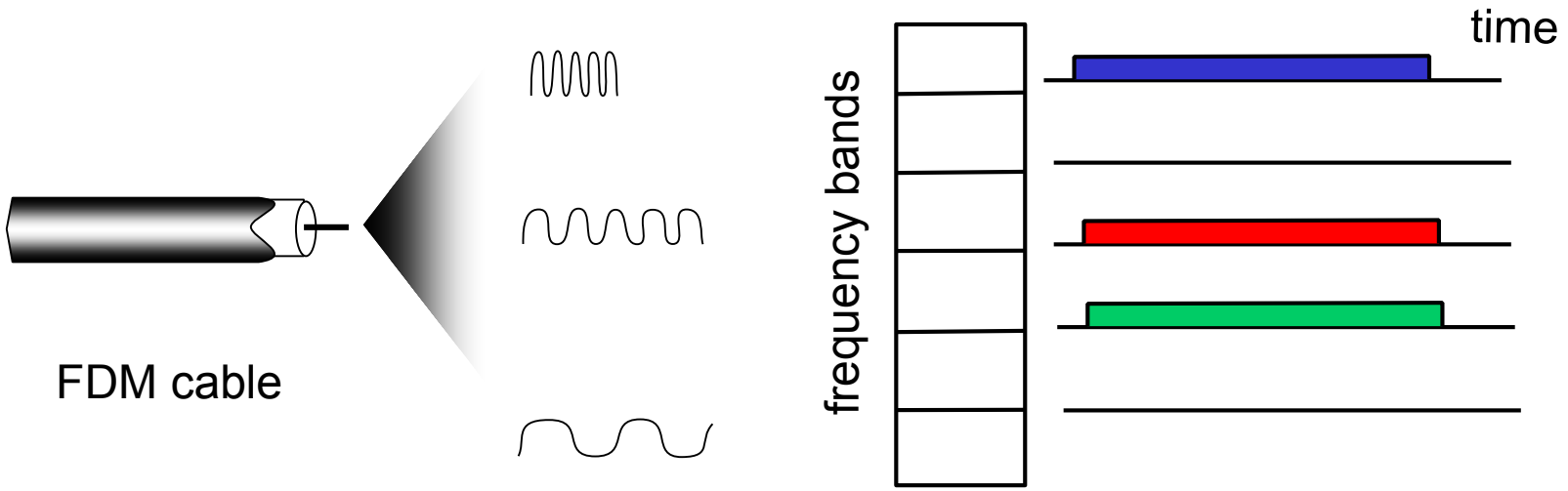
- TDMA: Time Division Multiple Access
  - Acceso al canal en “rondas”
  - Cada estación recibe un slot de tiempo fijo en cada ronda (por ejemplo el tiempo necesario para enviar una trama de tamaño máximo)
  - Si una estación no tiene datos para transmitir su slot se desperdicia
  - Ejemplo, 6 estaciones:



- Defecto: si muchas estaciones tienen tráfico esporádico, se desperdicia mucha capacidad
- Ventaja: no hay colisiones
- Posible mejora: asignación dinámica de slots
  - Pero requiere overhead para la asignación y hace complejo el protocolo

Protocolos MAC particionando el canal: FDMA

- FDMA: Frequency Division Multiple Access
- Se divide el espectro disponible en múltiples bandas, asignando una a cada estación
- Al igual que en TDMA, se desperdicia la capacidad correspondiente a las estaciones que no tienen datos para transferir



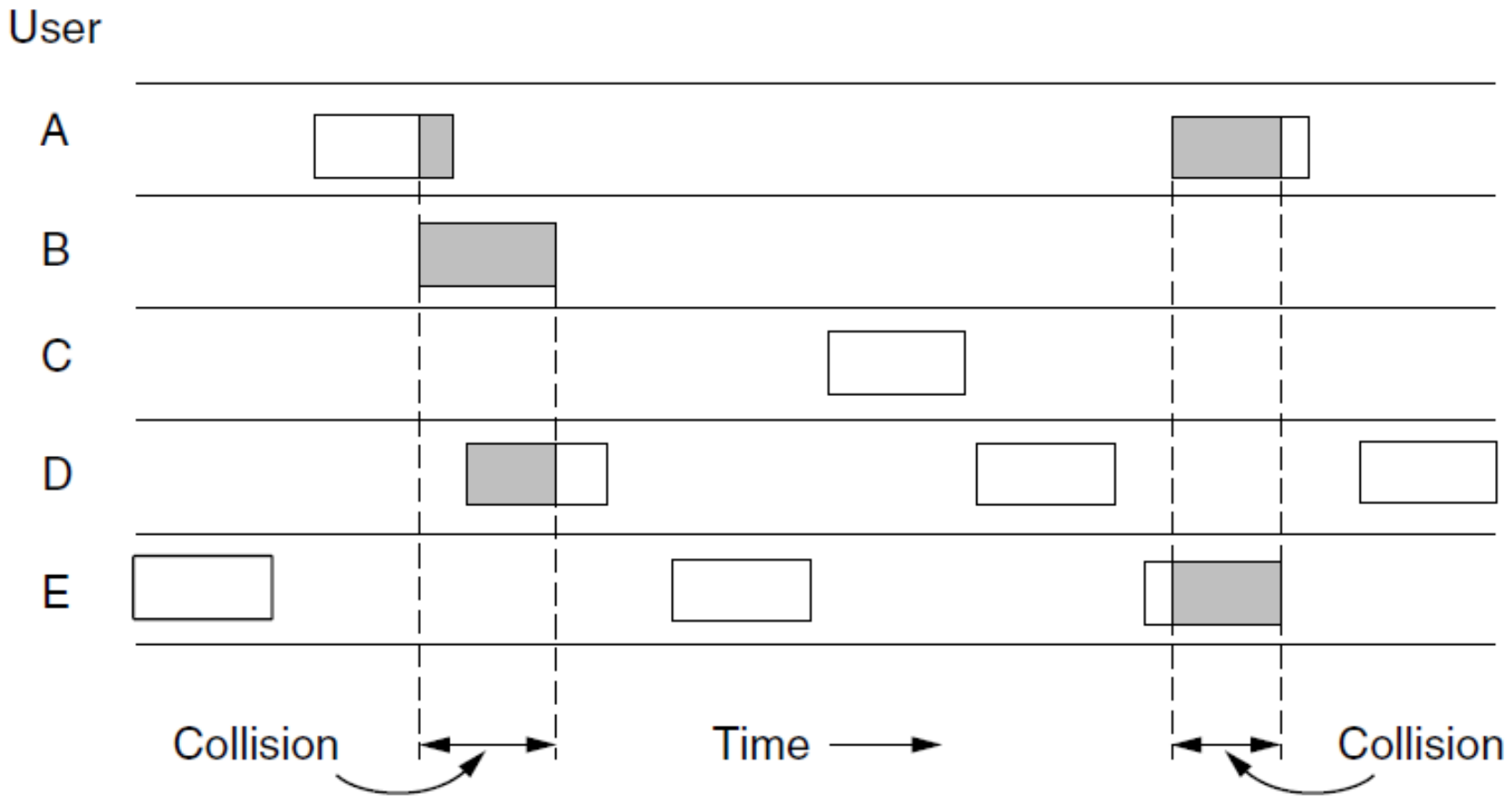
Link Layer and LANs 6-22

Protocolos con acceso aleatorio

- Cuando un nodo tiene algo para transmitir:
  - Transmite usando la capacidad completa del canal R (cumpliendo ciertas reglas)
  - No hay una coordinación a-priori entre los nodos
- Si dos o más nodos transmiten, hay una colisión
  - Precisamos poder detectar las colisiones
  - Precisamos poder recuperarnos de las colisiones

Un poco de historia: ALOHA

- 1970 radio bases en Hawaii (Abramson)
- Estaciones transmiten cuando tienen datos
  - Sin ninguna coordinación

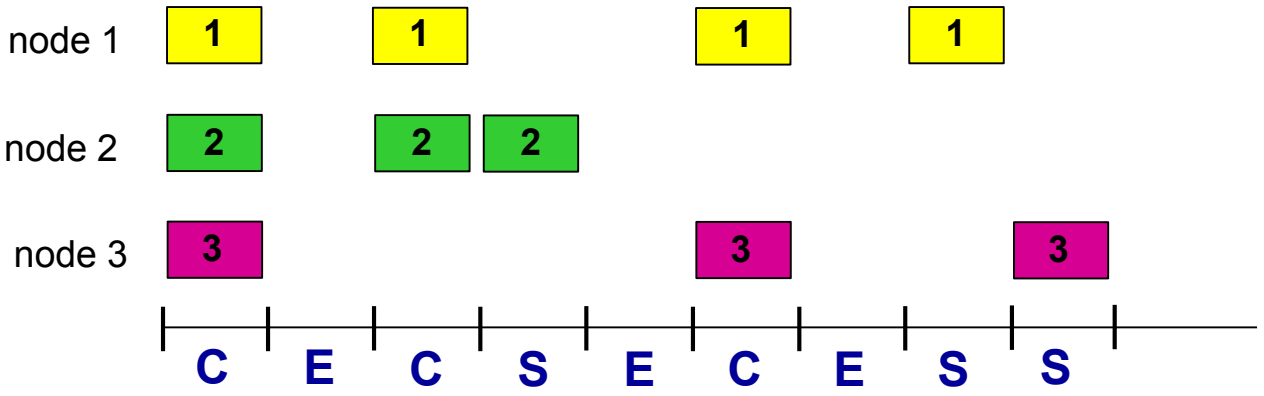


ALOHA (2)

- se detecta si hubo colisión
  - Variantes: inmediatamente o con retardo
  - En el original, observo si la base retransmite mi trama
- En caso de colisión, se debe esperar un tiempo aleatorio y retransmitir
  - **Debe ser aleatorio para evitar la sincronización entre las estaciones**
- Eficiencia teórica máxima posible: 18%

ALOHA ranurado

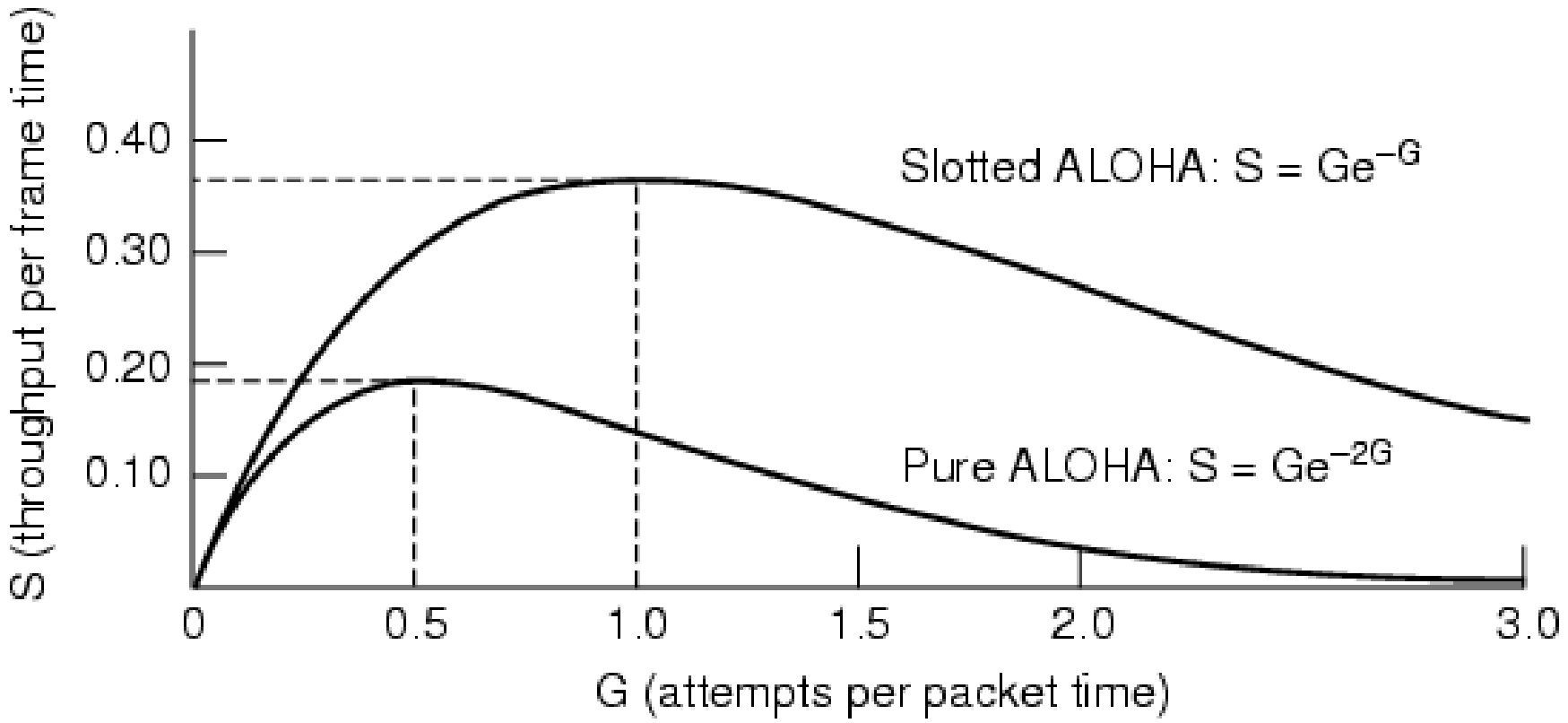
- 1972: se discretiza el tiempo (slots o ranuras de tiempo de duración necesaria para transmitir una trama)
- Solo puedo transmitir al comienzo de un slot
- Problema: preciso sincronización entre estaciones
- Si 2 o más estaciones transmiten en el mismo slot, todas las estaciones se dan cuenta de la colisión
- Si hay una colisión, esperan un tiempo aleatorio



C: collision E: Empty S: Success

Eficiencia (modelo simplificado)

- Pueden ver el cálculo en el libro (Kurose o Tanenbaum)
- En la gráfica se observa el promedio de utilización del canal (proporción de slots con tramas útiles) en función de la carga ofrecida (G)



Protocolos con detección de portadora (CSMA)

- CSMA: Carrier Sense Multiple Access (Acceso Múltiple con detección de portadora)
- Mejora: Antes de transmitir, detectar si otro equipo está utilizando el canal (“portadora”)
- Igual hay colisiones por retardos de propagación en el canal
  - Escucho libre el canal aún cuando otra estación comenzó a transmitir
- Solo utilizables en medios con bajo retardo, donde las estaciones puedan “escucharse entre sí en tiempo real”

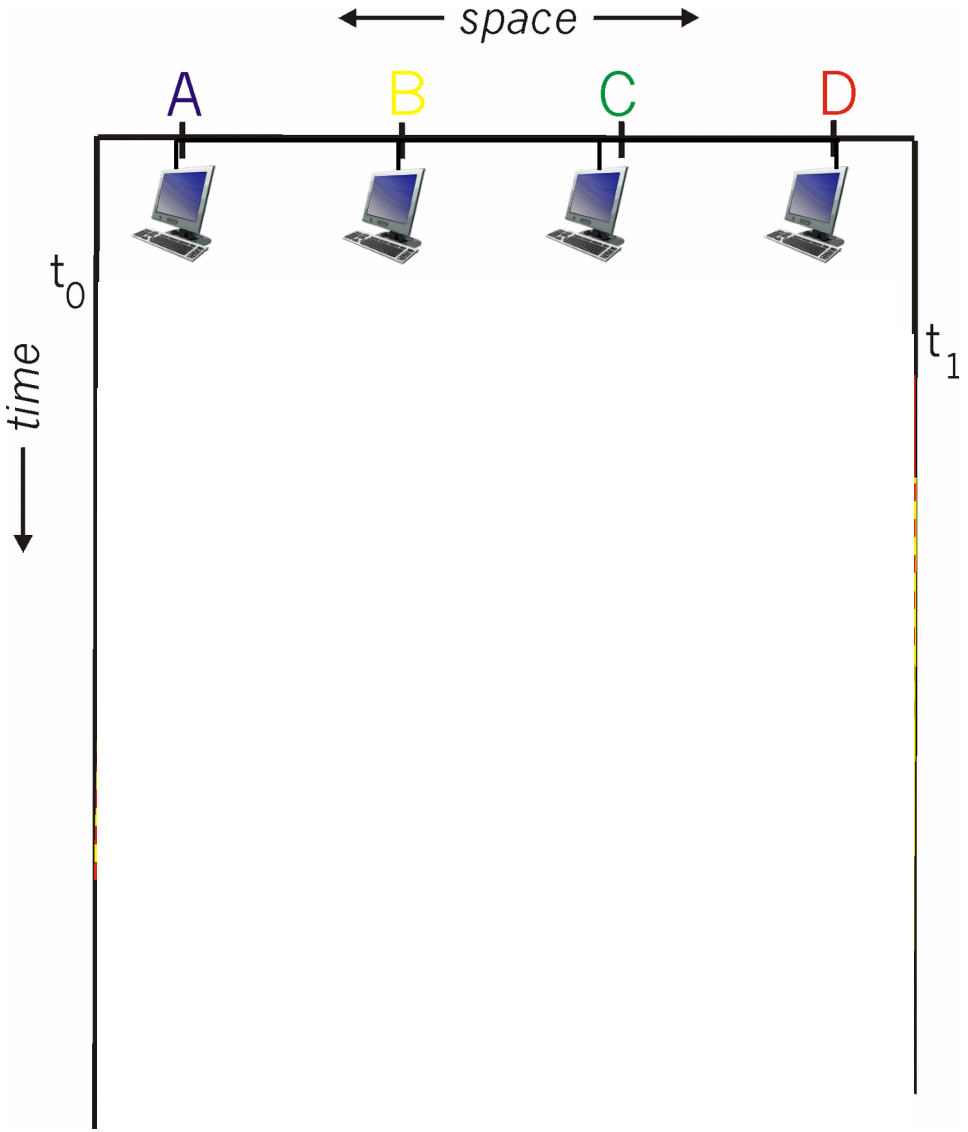
Colisiones en CSMA

- Pueden ocurrir colisiones

  El retardo de propagación hace que los nodos puedan no escucharse a tiempo

- En caso de colisión, se desperdicia todo el tiempo correspondiente a la transmisión de una trama

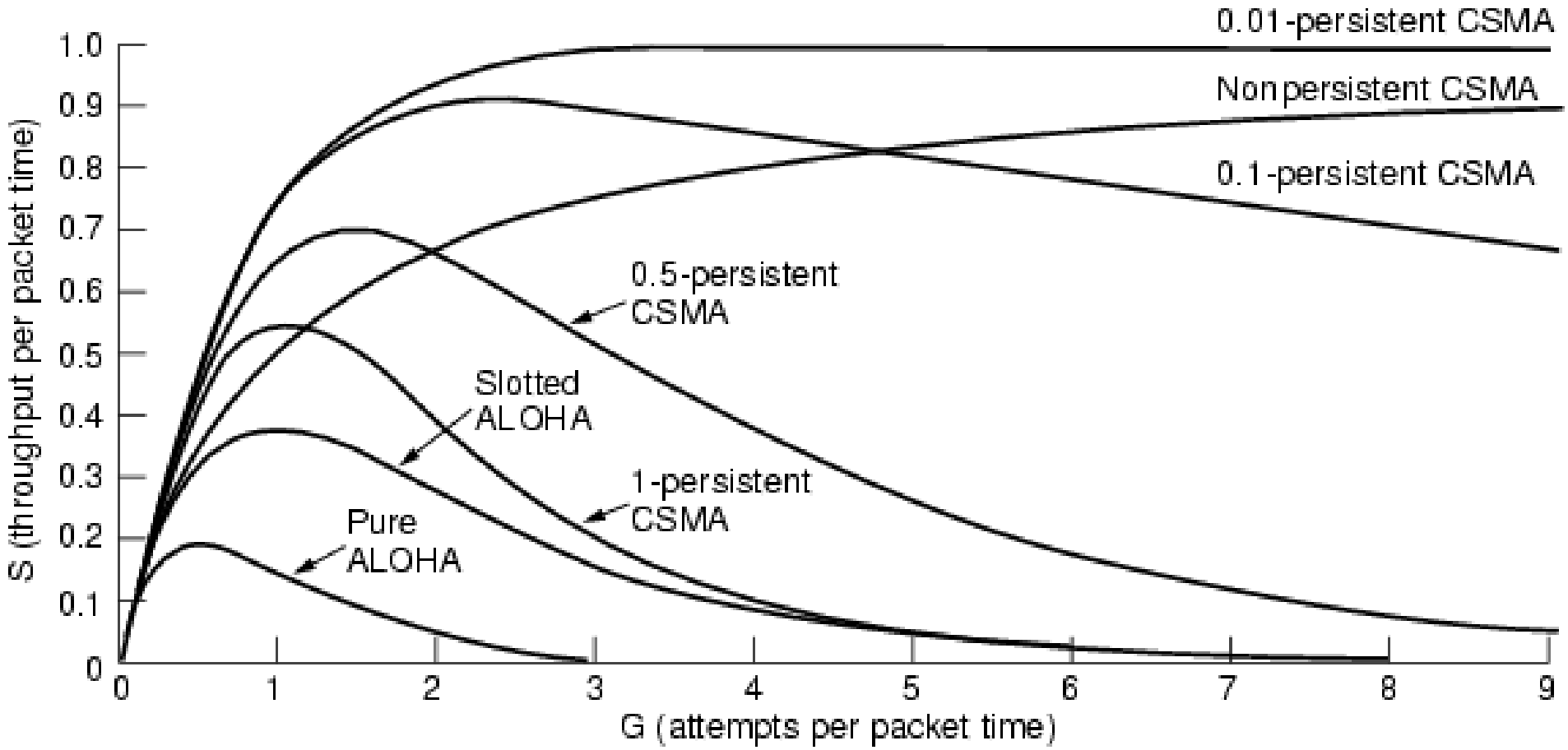
- La probabilidad de colisión aumenta con la distancia entre estaciones



Variantes de CSMA

- CSMA persistente y no persistente
  - persistente o 1-persistente
    - Si el canal esta libre se transmite
    - Si está ocupado, se transmite tan pronto se libere
  - no-persistente
    - Si el canal esta libre se transmite
    - Si el canal está ocupado, se espera un tiempo aleatorio antes de sensar nuevamente el canal
- CSMA p-persistente
  - canales en tiempo ranurado (estaciones sincronizadas)
  - Si el canal está ocupado, se mira el canal en la siguiente ranura
  - Si el canal está libre, transmite con probabilidad p y espera hasta la siguiente ranura con probabilidad 1-p
  - Si el canal se ocupa, se espera un tiempo aleatorio y se comienza nuevamente

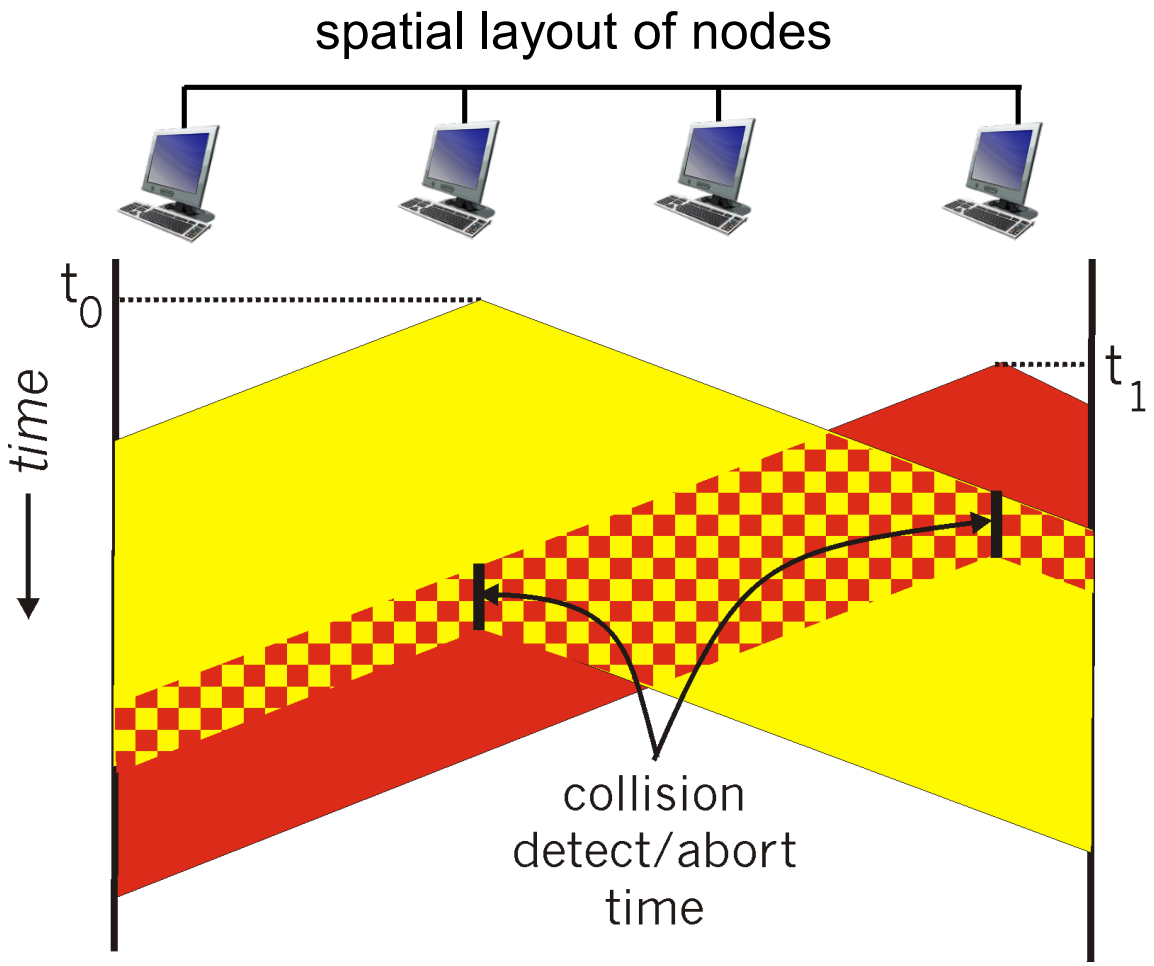
Comparación de performance (modelos simplificados)



CSMA/CD (CSMA con detección de colisión)

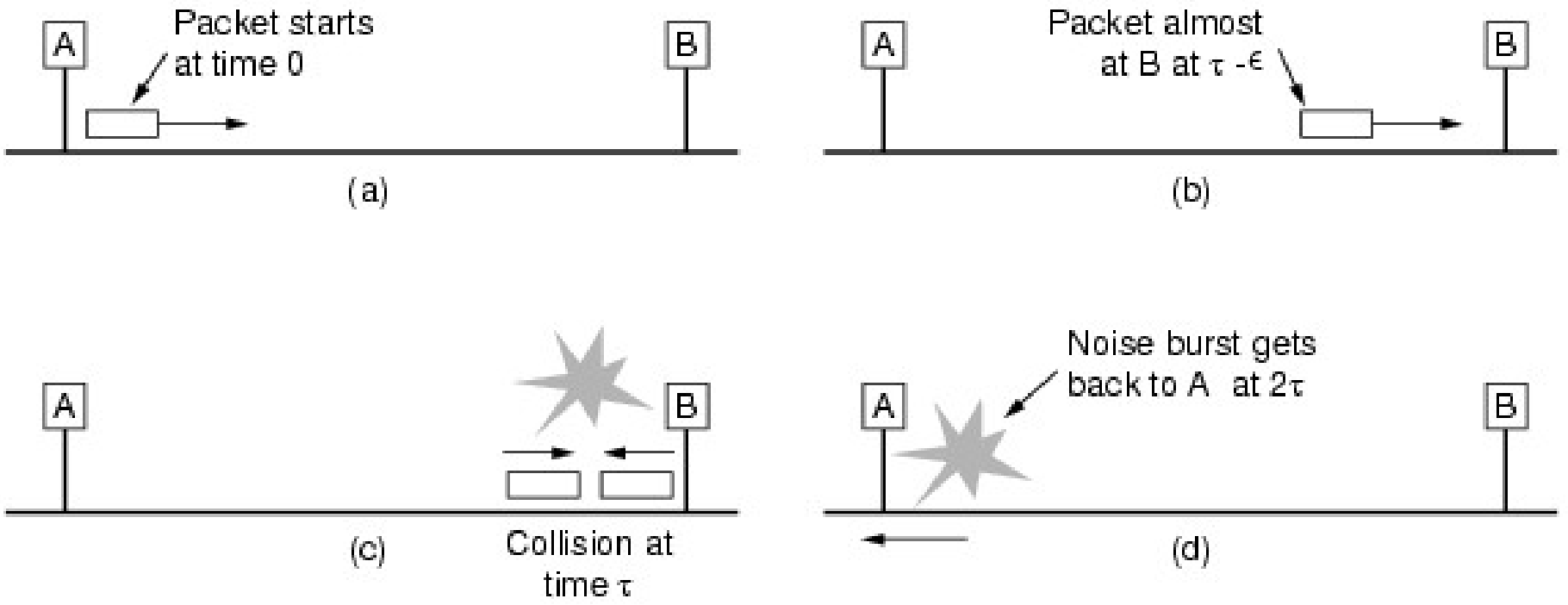
- CD: Collision detection
- Sensar el canal mientras se transmite (escuchar mientras se habla)
- Detener la transmisión cuando detecta colisión
- Detección de colisiones: analógico
  - Se ve si en el canal se lee algo distinto a lo que se escribió
- Ejemplo: Ethernet (802.3) a bajas velocidades (10-100 Mbps) usa CSMA/CD
- No se evitan colisiones, pero se disminuye el tiempo que el canal está ocupado con la colisión

Diagrama espacio/tiempo



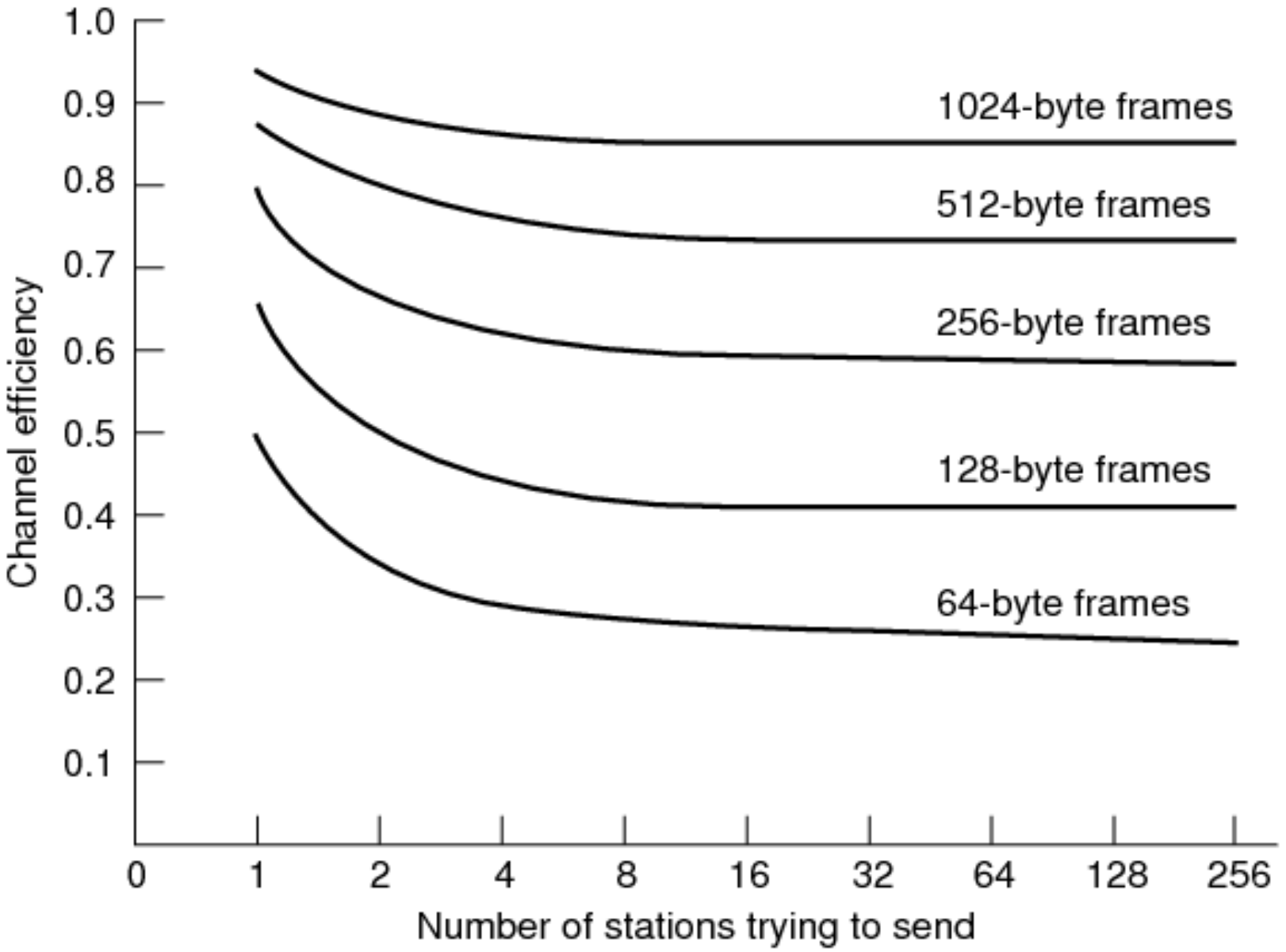
Largo mínimo de trama en CSMA/CD

- Queremos asegurarnos que todos los equipos detectan la colisión



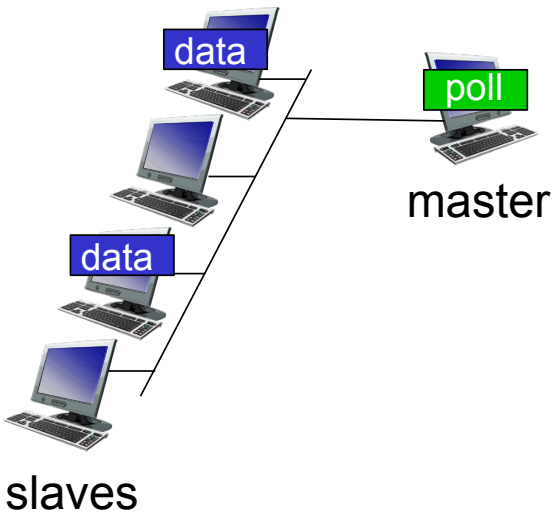
- Si antes que llegue la señal desde B, A deja de transmitir, no tiene forma de saber si la colisión fue con su trama
- Tiempo mínimo de transmisión: 2τ (tiempo de ida y vuelta)
- Se impone como mínimo tiempo de transmisión el tiempo de ida y vuelta entre las estaciones más lejanas (precisamos un máximo de distancia permitido). Nos fija el largo mínimo de trama

Eficiencia (modelo simplificado)



Protocolos “por turnos”

- “polling” (encuesta)
  - Nodo maestro “invita” a cada esclavo a transmitir por turnos
  - Típicamente esclavos “tontos”
  - Problemas: punto de falla (maestro), overhead debido al polling, latencia
  - Ej.: Bluetooth



- Pasaje de token (ficha)
  - Ficha de control se pasa de una estación a la siguiente secuencialmente
  - Solo se envía si se tiene el token
  - Problemas: recuperación de la ficha si se corrompe, latencia, overhead
  - Redes históricas: Token ring, token bus