Redes de Computadoras: Capa de Enlace

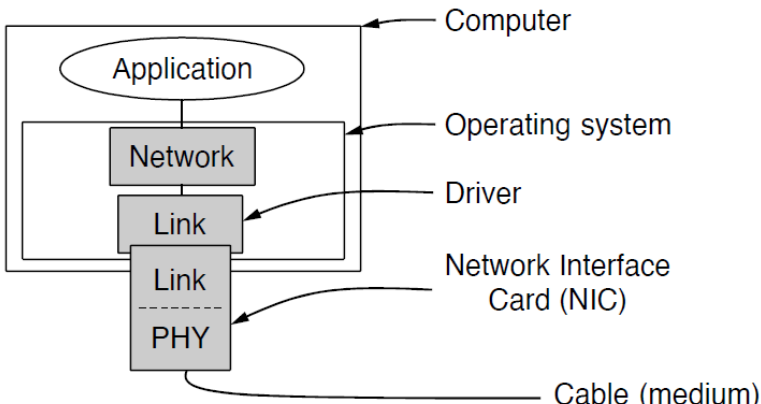
Alejandro Beltrán Varela [1] Christopher Guerra Herrero [1] Roberto Marti Cedeño [1]

[1]Facultad de Matemática y Computación. Universidad de La Habana

Temáticas:

- Características generales
- Servicios provistos a la capa superior
- Estrategias de delimitacion de frames
- Detección y corrección de errores
- Algoritmos elementales de transmisión de información

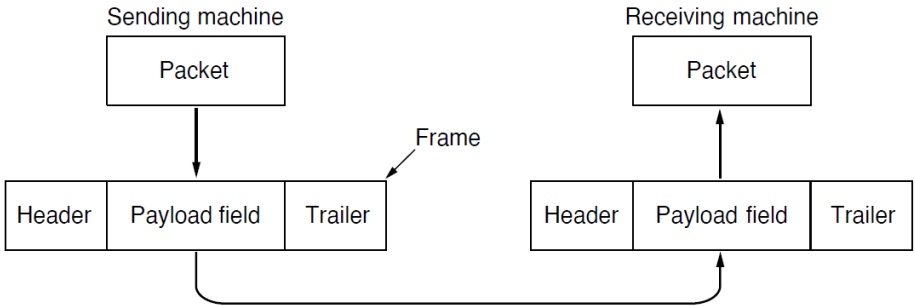
Arquitectura de las capas de red



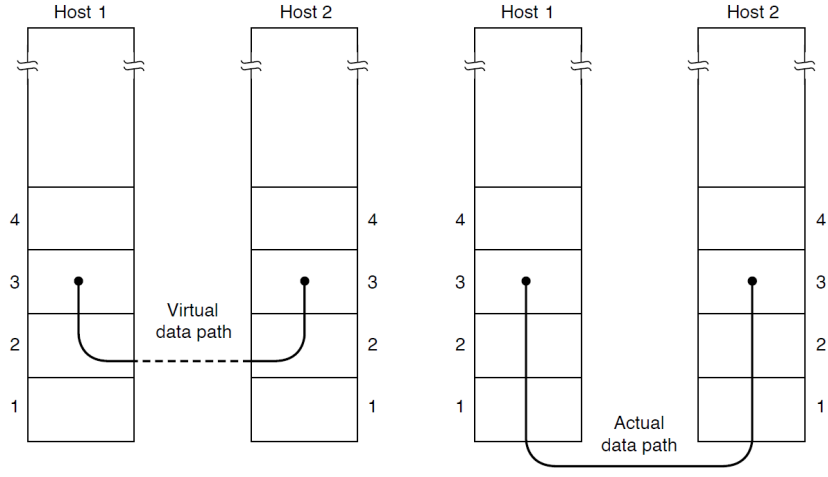
Objetivos de diseño de la capa de enlace

- Proveer de una interfaz bien definida de servicios para la capa de red.
- Delimitar la secuencias de bytes en frames bien definidos.
- Detectar y corregir errores de transmisión.
- Regular en flujo de los datos de forma tal que remitentes rápidos no ahogen a receptores lentos.

Estructura de un frame



Comunicación



Servicios provistos

- Servicio sin retroalimentación no oriendato a conexiones
- Servicio con retroalimentación no orientado a conexiones
- Servicio con retroalimentación orientado a conexiones

Servicio sin retroalimentación no orientado a conexiones

- Medio confiable con bajo índice de errores
- No se comunican los frames recibidos
- No se establece algún tipo de comunicación lógica
- Tráfico en tiempo real. Voz y video.

Servicio con retroalimentación no orientado a conexiones

- Medio no confiable con alto índice de errores
- Se comunican los frames recibidos
- No se establece algún tipo de comunicación lógica
- Genera un costo adicional

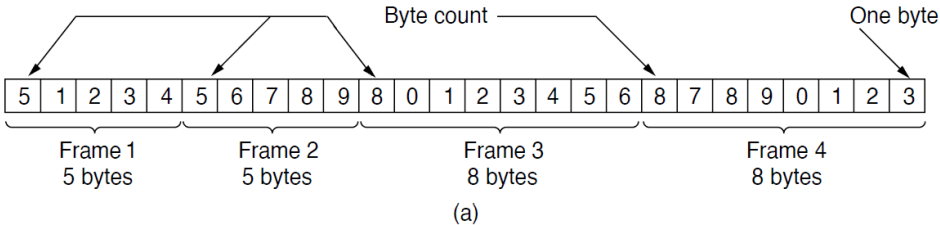
Servicio con retroalimentación orientado a conexiones

- Cualquier Medio
- Se comunican los frames recibidos
- Se establece una comunicación antes de realizar el envío
- Cada frame enviado se verifica
- Cada frame se recive una vez y en el orden correcto

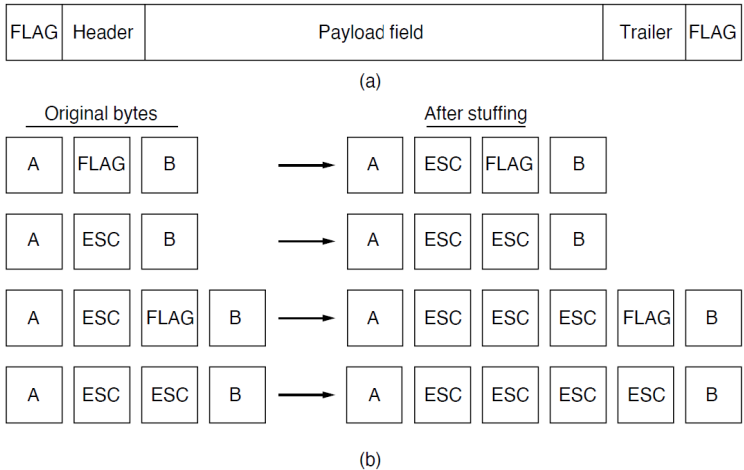
Estrategias de delimitacion de frames

- Conteo de bytes
- Relleno mediante flag bytes
- Relleno mediante flag bits
- Incumplimiento de la codificación de la capa física

Conteo de bytes



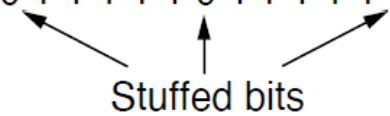
Relleno mediante flag bytes



Relleno mediante flag bits

(a) 0 1 1 0 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 0 0 1 0

(b) 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



(c) 0 1 1 0 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 0 0 1 0

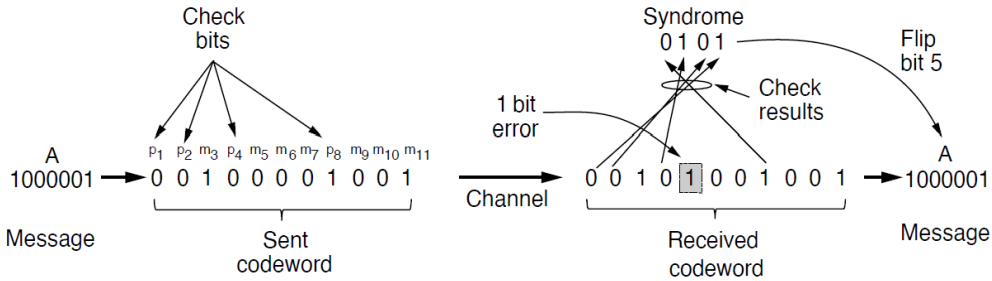
Incumplimiento de la codificación de la capa física

Data (4B)	Codeword (5B)	Data (4B)	Codeword (5B)
0000	11110	1000	10010
0001	01001	1001	10011
0010	10100	1010	10110
0011	10101	1011	10111
0100	01010	1100	11010
0101	01011	1101	11011
0110	01110	1110	11100
0111	01111	1111	11101

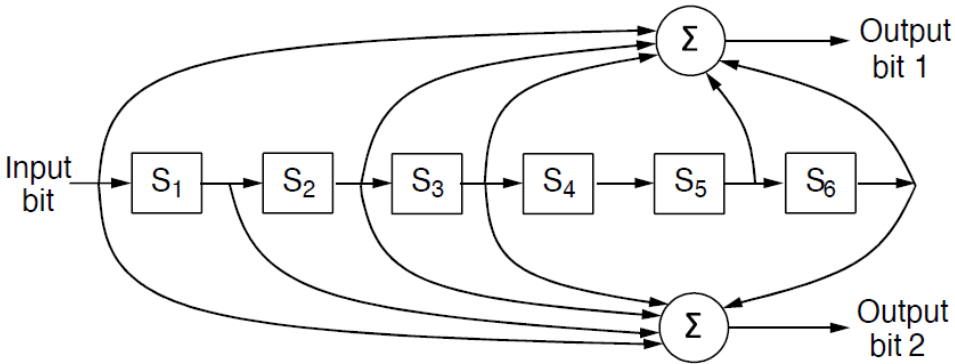
Detección y corección de errores

- Hamming Codes
- Códigos convolucionales binarios
- Códigos Reed-Solomon
- Códigos de baja densidad con respecto a la paridad

Códigos de Hamming



Códigos convolucionales binarios



Códigos Reed-Solomon

- Un polinomio de grado n esta determinado por n + 1 puntos
- símbolos de tamaño m bits.
- m = 8. 256 bytes de longitud.
- (255, 233). 255 Símbolos, 22 redundantes 233 de datos.

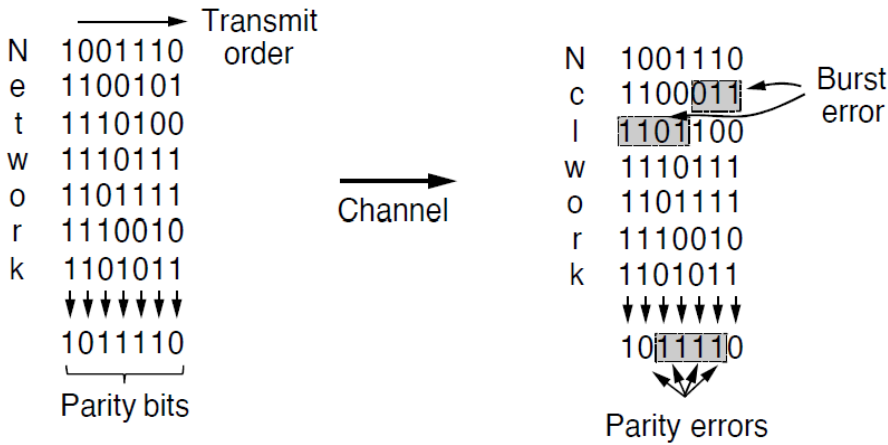
LDPC

- Cada bit esta determinado unívocamente por un set pequeño de los bits de la entrada
- Representación en matriz con baja densidad de 1
- Algoritmo de aproximación que devuelve los datos recibidos aproximados a una palabra válida.
- Empleado en 10 Gbps Ethernet, redes sobre cableado eléctrico, 802.11

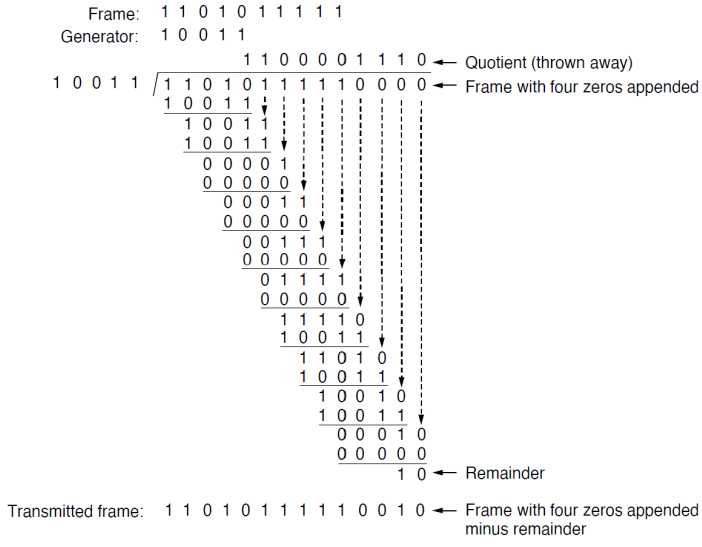
Detección de errores

- Paridad
- Checksums
- CRC

Paridad


Transmit order
N 1001110
e 1100101
t 1110100
w 1110111
o 1101111
r 1110010
k 1101011
1011110
Parity bits
Channel
N 1001110
c 1100011
l 1101100
w 1110111
o 1101111
r 1110010
k 1101011
10 1111 0
Burst error
Parity errors

CRC


Frame: 1 1 0 1 0 1 1 1 1 1
Generator: 1 0 0 1 1
1 1 0 0 0 0 1 1 1 0 ← Quotient (thrown away)
1 0 0 1 1 | 1 1 0 1 0 1 1 1 1 1 0 0 0 0 ← Frame with four zeros appended
1 0 0 1 1
1 0 0 1 1
1 0 0 1 1
0 0 0 0 1
0 0 0 0 0
0 0 0 1 1
0 0 0 0 0 0
0 0 1 1 1 1
0 0 0 0 0 0
0 1 1 1 1 1
0 0 0 0 0 0
1 1 1 1 1 0
1 0 0 1 1 1
1 1 0 1 0 1
1 0 0 1 1 1
1 0 0 1 0 1
1 0 0 1 1 1
0 0 0 1 0 1
0 0 0 0 0 0
1 0 ← Remainder
Transmitted frame: 1 1 0 1 0 1 1 1 1 1 0 0 1 0 ← Frame with four zeros appended minus remainder

CRC-32

$$x^{32} + x^{26} + x^{23} + x^{22} + x^{16} + x^{12} + x^{11} + x^{10} + x^{8} + x^{7} + x^{5} + x^{4} + x^{2} + x^{1} + 1$$

Algoritmos elementales de capa de enlace

- Sin control de flujo ni corrección de errores
- Control de flujo sin corrección de errores
- Control de flujo, números de secuencia y ARQ
- Transmisión bidireccional
- Ventanas desplazantes

Asunciones iniciales

- Procesos independientes
- Comunicación Unidireccional
- Equipos y medios confiables

Algoritmo básico

```
void sender1(void)
{
  frame s;
  packet buffer;
  while (true) {
        from_network_layer(&buffer);
        s.info = buffer;
        to_physical_layer(&s);
  }



}
```

```
void receiver1(void)
{
  frame r;
  event_type event;
  while (true) {
        wait_for_event(&event);
        from_physical_layer(&r);
        to_network_layer(&r.info);
  }
}
```

Algoritmo con control de flujo

```
void sender2(void)
{
  frame s;
  packet buffer;
  event_type event;
  while (true) {
        from_network_layer(&buffer);
        s.info = buffer;
        to_physical_layer(&s);
        wait_for_event(&event);
  }
}
```

```
void receiver2(void)
{
  frame r, s;
  event_type event;
  while (true) {
        wait_for_event(&event);
        from_physical_layer(&r);
        to_network_layer(&r.info);
        to_physical_layer(&s);
  }
}
```

Algoritmo con número de secuencia y ARQ

```
void sender3(void)
{
  seq_nr next_frame_to_send;
  frame s;
  packet buffer;
  event_type event;

  next_frame_to_send = 0;
  from_network_layer(&buffer);
  while (true) {
        s.info = buffer;
        s.seq = next_frame_to_send;
        to_physical_layer(&s);
        start_timer(s.seq);
        wait_for_event(&event);
        if (event == frame_arrival) {
              from_physical_layer(&s);
              if (s.ack == next_frame_to_send) {
                    stop_timer(s.ack);
                    from_network_layer(&buffer);
                    inc(next_frame_to_send);
              }
        }
  }
}
```

```
void receiver3(void)
{
  seq_nr frame_expected;
  frame r, s;
  event_type event;

  frame_expected = 0;
  while (true) {
        wait_for_event(&event);
        if (event == frame_arrival) {
              from_physical_layer(&r);
              if (r.seq == frame_expected) {
                    to_network_layer(&r.info);
                    inc(frame_expected);
              }
              s.ack = 1 – frame_expected;
              to_physical_layer(&s);
        }
  }
}
```

Mejorando la comunicación

- Piggybacking
- Ventanas desplazantes
  - 1-bit
  - Go back n
  - Repetición selectiva

Ventanas desplazantes