

# Protocolos punto a punto

Resolución de ejercicios vistos en clase

08.04.2025

## 1. Primer ejercicio

### 1.1. Enunciado

Dado un enlace punto a punto a la luna de 1Mbps con un delay de 1.25 segundos:

- ¿Cuántos bits entran en el enlace?
- Asumiendo que se separan los bits en frames de largo fijo de 1Kb ¿Cuántos frames entran en el enlace?

### 1.2. Resolución

- La ecuación que determina cuantos bits entran en el canal esta determinada por:

$$\blacksquare C_{vol} = V_{tx} * Delay.$$

Dados entonces  $V_{tx} = 1Mbps$  y  $Delay = 1,25$  segundos tenemos que entran 1,25Mb en el canal.

- Para frames de largo fijo 1Kb tenemos que entran:  $\frac{1,25Mb}{1Kb} = 1250$  frames.

## 2. Segundo ejercicio

### 2.1. Enunciado

Calcule la eficiencia del frame tomando en cuenta solo el overhead impuesto por las siguientes técnicas de framing:

- Largo fijo.
- Campo de 16 bits en el encabezado indicando el largo del frame.
- Delimitadores de 8 bits usando bit-stuffing.

### 2.2. Resolución

- Siendo  $\eta_{frame} = \frac{\text{largo de los datos}}{\text{largo total del frame}}$ , en este caso tenemos que:  $\eta_{frame} = \frac{\text{largo de los datos}}{\text{largo de los datos}} = 1$

- Para campos en el encabezado tenemos que:  $\eta_{frame} = \frac{\text{largo del frame} - |\text{campo}_{size}|}{\text{largo del frame}} = \frac{\text{largo del frame} - 16}{\text{largo del frame}}$

- c. Este último caso es variable ya que depende de la cantidad de bits que haya que incorporar al frame debido al stuffing. Siendo  $n$  la cantidad de bits, nos queda

$$\eta_{frame} = \frac{\text{largo del frame} - 8 - n}{\text{largo del frame}}$$

### 3. Tercer ejercicio

#### 3.1. Enunciado

Diseñe posibles conjuntos de frames para los siguientes tipos de protocolos, asumiendo que se detectan errores usando CRC (No hace falta aclarar el largo de los campos):

- Stop & Wait
- Sliding Window con GoBackN usando *Piggybacking*
- Sliding Window con ACK Selectivo

#### 3.2. Resolución

- a. Para el primer caso podríamos definir un frame para el emisor y otro para el receptor de la siguiente forma:

Emisor: #SEQ(1bit); Datos; Checksum  
Receptor: #ACK(1bit); Checksum

- b. Para el segundo caso, al haber *Piggybacking* todos los frames pueden llegar a tener que cumplir simultáneamente los roles de emisión y recepción. Entonces proponemos un único frame como el siguiente:

#SEQ; #ACK; Datos; Checksum

- c. Para el último caso, a diferencia del punto anterior tenemos reconocimiento selectivo. Para estos casos se recomienda, por un tema de eficiencia, que el frame receptor reconozca los frames tanto acumulativa como individualmente. Entonces nos quedarían dos frames como los siguientes:

Emisor: #SEQ; Datos; Checksum  
Receptor: #ACK; #SACK; Checksum

### 4. Cuarto ejercicio

#### 4.1. Enunciado

Un protocolo sobre un enlace punto a punto de 1Mbps y 0.25 segundos de delay, trabaja con ventana deslizante con GoBackN usando frames de largo fijo 2Kb y un CRC de 16bits para detectar errores.

- Calcule cuáles son los tamaños de ventana de emisión y recepción óptimos.
- ¿Cuántos bits hacen falta para secuenciar los frames?
- Calcule cuánto tiempo es necesario para transmitir 20Mb de datos asumiendo que no hay errores.

## 4.2. Resolución

a. Es un protocolo de ventana deslizante GoBackN. Esto nos indica que por definición  $RWS = 1$ . Para el tamaño de la ventana de emisión tenemos:  $SWS = \frac{V_{tx} * 2 * Delay(Frame)}{|Frame|} = \frac{1Mbps * 2 * 0,25s}{2Kb} = 250$  frames.

b.  $SWS + RWS \leq \#Frames \Rightarrow \#SEQ \geq 251 \Rightarrow$  con 8 bits alcanza para secuenciar los frames.

c. Veamos que nos piden transmitir 20Mb. Pero además sabemos que **no hay errores** en la transmisión. Entonces, como ya calculamos el tamaño de ventana que maximiza la eficiencia del protocolo podemos asumir que la eficiencia se aproxima a 1. Con todos estos datos, podemos asumir que podemos transmitir todo el tiempo, que no vamos a bloquearnos esperando reconocimientos y que no va a haber errores. Además, tenemos que tener en cuenta la eficiencia del frame, debido a que no todos los bits que componen el frame son utilizados para los datos. La cantidad de bits del campo de datos en cada frame sería:  $|Datos| = |Frame| - \#Seq - |CRC| = 2000 - 8 - 16 = 1976$ .

Ahora bien, si para transmitir 1976 bits necesitamos un frame, entonces para transmitir 20Mb, necesitaríamos:  $20000000/1976 = 10121,45$ . Esto hay que redondearlo para arriba, debido a que los frames son de largo fijo, entonces necesitamos 10122 frames para el envío de 20Mb.

Finalmente, bajo estas condiciones, transmitir 20Mb no es otra cosa que el  $Delay(10122 Frames)$  (+  $Delay(2Kb)$  para esperar el último ACK):

$Delay(10122 Frames) = T_{tx}(10122 Frames) + T_{prop} = \frac{2000 * 10122 \text{ bits}}{1000000 \text{ bps}} + 0,25s = 20,244s + 0,25s = 20,494$ . Si a esto le sumamos el delay del último ACK, la respuesta final es 20,744s.

## 5. Quinto ejercicio

### 5.1. Enunciado

Un protocolo confiable punto a punto que usa sliding window, opera sobre un canal de 10 Mbps, usa SACK y un frame emisor de 5Kb como el siguiente:

#SEQ (8bits) ; Datos ; Checksum

- Proponga un frame para el receptor.
- ¿Cuál es el valor de delay para el cual el protocolo presenta un 100 % de eficiencia?
- Si el delay fuera de 1 seg ¿Cuántos bits deberían ocupar los números de secuencia de manera de maximizar la eficiencia?

### 5.2. Resolución

a. Al no presentarse requerimientos que ameriten el uso de piggybacking, no vamos a usar esta técnica para el framing. El protocolo va a ser asimétrico, en el sentido de que el frame emisor y el receptor van a ser distintos.

El número de secuencia del frame emisor ocupa 8 bits, esto implica que tanto el campo ACK del receptor como el campo SACK deben ocupar la misma cantidad de bits: 8 bits para ACK y 8 bits para SACK. Esto es porque es necesario poder referenciar en los reconocimientos (ACKs) a cada frame que envía el emisor. **Nota:** no es necesario para la correctitud del algoritmo de ventana deslizante usar SACK y ACK cuando hay reconocimientos selectivos. Sin embargo se estila usar de este modo porque mejora el desempeño del protocolo.

El frame receptor propuesto sería entonces:

#ACK(8bits) ; #SACK(8bits) ; Checksum (16bits)

- b. Para encontrar el valor de delay óptimo hay que tratar de llenar el canal y que no se desperdicie tiempo estando bloqueado sin poder transmitir. Veamos que datos tenemos:

- Tengo 256 frames posibles porque el campo de #SEQ es de 8 bits.
- Pero, para evitar el problema de las reencarnaciones, solo 128 frames podrán ser enviados pues se usa SACK  $\Rightarrow SWS = 128 = RWS$ .

Luego, usamos los datos que tenemos en la fórmula de Eficiencia para el caso en cuál es óptima.

$$1 = \frac{Ttx(SWS)}{RTT(F)} = \frac{Ttx(F)*SWS}{Delay(Fe)+Delay(Fr)} = \frac{(5Kb/10Mbps)*128}{Delay(5Kb)+Delay(32bits)} = 1 \Rightarrow$$

Resolviendo se obtiene que 0.031 segundos aprox. es el valor del delay que maximiza la eficiencia del protocolo.

- c. Para resolver este ejercicio se recalcula el tamaño de la ventana de emisión para ajustarla al delay requerido:

$$SWS = \frac{V_{tx} * 2 * Delay}{|Frame|}$$

Reemplazando los datos y resolviendo se obtiene que  $SWS = 4000$ . Luego,  $\#Frames \geq 8000$  y tomando el techo del  $\log_2(8000)$  se obtiene que se necesitan 13 bits.