Ventanas, Retransmisiones y Congestión

Profesor: Rodrigo Arenas
Auxiliar: Francisco Almeida
Ayudantes: Cristian Romero , Felipe Alfaro

1. P1: Protocolos Clásicos

1.1. Go-Back-N vs Selective Repeat

En general, se considera que Selective Repeat es un mejor algoritmo que Go-Back-N. Sin embargo tiene algunos costos mayores frente al más simple. ¿Existe algún escenario en que Go-Back-N se comporte tan bien como Selective Repeat y por lo tanto no valga la pena usar el algoritmo más complejo? Busque ejemplos y compare.

1.2. Tamaño ventanas en selective repeat

En general, se define que la ventana óptima para selective repeat es que el enviador y el receptor tengan una ventana del tamaño del BDP. Sin embargo, frente a pérdidas, a veces sirve tener ventanas más grandes, mejorando el ancho de banda logrado. Responda las siguientes preguntas:

1.2.1. Ventana envío más grande

Imagine un escenario en que la ventana de envío es grande (digamos, dos veces el BDP) y la ventana de recepción es igual al BDP. ¿Sirve? ¿Por qué?

1.2.2. Ventana recepción más grande

Imagine un escenario en que la ventana de recepción es grande (digamos, dos veces el BDP) y la ventana de envío es igual al BDP. ¿Sirve? ¿Por qué?

1.2.3. Ambas grandes

Imagine un escenario en que las dos ventanas son grandes (digamos, dos veces el BDP) ¿Sirve? ¿Por qué?

1.3. Infinitas

Suponga que alguien inventa una memoria infinita, ahora podemos tener ventanas infinitas de grandes (ignoremos el pequeño problema de tener números de secuencia infinitos). ¿Qué ventajas y desventajas podría tener un sistema así?

2. P2: TCP

2.1. Timeouts

Una caracteristica importante de TCP es su adaptación de los timeouts a RTTs muy variables en el tiempo. Trata de estimar el promedio y la varianza para lograr acertarle a más o menos el 90 % de las veces. Es decir, no retransmitir innecesariamente más del 10 % de los paquetes.

Pero, esto tiende a generar timeouts muy grandes. Una solución es usar Fast Retransmit, que decide que una secuencia de tres ACKs equivocados implica una pérdida y retransmitimos el paquete de la misma forma que si hubiese ocurrido un timeout. Analice los problemas que tiene esto, revise cómo se resuelven y discuta si en algunos casos puede ser mala idea. Dé ejemplos.
HINT: busque qué es "Fast Recovery"

2.2. Congestión

Un problema de la ventana de congestión en enlaces con alto BDP, es que demora mucho en reaccionar, ya que el algoritmo fuerza a que esperemos un RTT completo antes de hacer cualquier cambio. Una posibilidad que se ha sugerido muchas veces es apurar esto un poco, por ejemplo reduciendo la espera RTT/2. Otras personas han sugerido cambiar la fórmula y aumentar la ventana en más de un segmento o reducirla en menos que la mitad. Discuta estas alternativas y analice los riesgos de cada una. ¿Qué sugiere Ud que sería la mejor solución para apurar TCP en estos escenarios?

3. P3: Más Congestión

El algoritmo de la ventana de congestión logra efectivamente controlar el ancho de banda utilizado. Pero, la ventana en realidad fue inventada para otra cosa: para adaptarse al BDP del enlace. Responda las siguientes preguntas:

3.1. Si hay pérdidas por congestión, ¿es como si cambiara el BDP de la conexión y por eso es bueno cambiar la ventana?

3.2. Al achicar la ventana de congestión: ¿se queda bloqueada la conexión esperando ACKs? ¿Es esto lo que se buscaba al achicar la ventana?

3.3. Al achicar la ventana de congestión: ¿estamos modificando el BDP de la conexión?

3.4. Proponiendo una solución alternativa: Podríamos controlar el ancho de banda utilizado (cuando comienzan las pérdidas) agregando un sleep() pequeño y adaptable según la pérdida, después de cada envío de paquete, manteniendo una ventana de tamaño constante. Esto ¿es como si mantuviera el BDP de la conexión? ¿Lograría el objetivo de limitar la congestión?

3.5. Compare esta solución con la ventana de congestión: ¿sería mejor o peor?