Investigación de Operaciones
I-Ciclo, 2015
Prof. Ileana Alpízar

UNIVERSIDAD DE COSTA RICA

José Slon Baltodano
Emmanuel Arias Soto

Proyecto de simulación de eventos discretos: Protocolo para envío de mensajes en una red de dos computadoras

Manual de instalación

Este proyecto se implementó utilizando Java, por lo tanto para poder correrlo es necesario tener Java correctamente instalado.

Para correr este programa basta con ejecutar el archivo ejecutable. Para ello se debe ejecutar el comando *java -jar simulacion.jar* estando en la carpeta del proyecto.

```
JoseMBP:Simulación JoseSlon$ pwd
/Users/JoseSlon/Dropbox/UCR/2015/I_Semestre/Investigación de Operaciones/Proyecto/Simulación
JoseMBP:Simulación JoseSlon$ ls
Archivo.java            Frame.java              Mensaje.java            Timer.java              manifest.txt            registros.zip
Estadistica.java        Main.java               Simulacion.java         doc                     registros               simulacion.jar
JoseMBP:Simulación JoseSlon$ java -jar simulacion.jar
```

Descripción del problema a simular

Suponga que se tiene una LAN (Local Area Network) compuesta solo por dos computadoras A y B. En esta red hay una única línea de comunicación desde A hasta B y otra desde B hacia A. Además la transferencia y la propagación es bit por bit.

Estas dos computadoras (A y B) utilizan el siguiente protocolo para el envío de mensajes desde A hacia B (es una sobresimplificación del protocolo “GO BACK N”):

La computadora A recibe los mensajes para ser enviados a la computadora B con una distribución para su tiempo entre arribos normal, con una media de 25 segundos y una varianza de 1 seg cuadrado o sea es n(μ=25,σ2=1). Todos los mensajes son de igual longitud y se van colocando en una cola, la cual para efectos prácticos es de tamaño infinito. El proceso que coloca los mensajes en cola no tiene relación con el que se encarga de hacer los envíos a B, por lo que no se va a tomar en cuenta y se asumirá entonces que el tiempo para colocar un mensaje en cola es 0.

El proceso en A que hace los envíos maneja una ventana de tamaño 8 (el N = 8). El tamaño de la ventana indica que este proceso en A para comenzar a enviar información a B la primera vez, debe esperar a haber recibido 8 mensajes. A inicia su trabajo y comienza a enviar uno por uno estos 8 mensajes. Para enviar un mensaje, el proceso de A debe realizar 2 tareas:

- Primero convierte el mensaje en un "frame", esta es una estructura de datos que contiene el número de identificación del frame, el mensaje, y cierta información para que B pueda revisar si el mensaje llega o no con errores. Para nuestro caso los números de identificación irán desde 0 en adelante, lo cual probablemente obligue a manejar valores muy altos si se corre por mucho tiempo la simulación. El tiempo que el proceso de A tarda para convertir un mensaje a un "frame" tiene distribución exponencial con una media de 2 segundos.

- Luego se coloca el “frame” en la línea bit por bit (tiempo de transferencia a la línea), como todos los “frames” son de igual longitud este tiempo es fijo de 1 segundo por “frame” (en resumen, este proceso tarda en total 1 + X segundos, con X exponencial con una media de 2 segundos, preparando y transfiriendo a la línea un mensaje).

Note que este proceso de A se mantiene ocupado mientras prepara el frame y lo pone en la línea. Cuando termina de poner en la línea el último bit del “frame”, se encuentra “libre” para poder preparar y enviar otro “frame”, independientemente de si el “frame” anterior aún fuera o no de camino hacia B.

El "frame" que A acaba de transferir a la línea, llega a B cuando llegue el último bit. El tiempo que tarda este último bit en llegar se llama tiempo de propagación y es fijo de 1 segundo.
La computadora A envía sus "frames" en secuencia según su número de identificación, y la computadora B espera recibir los "frames" en esa misma secuencia.

Cuando se envía un “frame” a B hay una probabilidad de 0.1 de que llegue con error y probabilidad 0.05 de que se pierda.

En B hay un proceso que se encarga de recibir los "frames" que van llegando desde A, si el proceso encargado de revisar los "frames" está ocupado, entonces solo coloca en una cola el recién recibido tardando 0 segundos. La revisión de un "frame" recibido consiste en verificar si llegó en la secuencia esperada y no lleva error, entonces crea y envía a A un número de confirmación o “ACK” (acknowledge) el cual tiene como valor el número del “frame” que B espera como siguiente. Por ejemplo, si B acaba de recibir correctamente el ”frame” con número de secuencia 4, entonces le enviará a A un ACK = 5.

El proceso de B que revisa "frames" y crea y envía ACK´s tarda los siguientes tiempos con sus distribuciones:

- Tiempo de revisión del "frame" para detectar si viene en secuencia y si llegó o no con errores, así como la creación del ACK si se debe hacer, sigue la siguiente distribución de probabilidad:

$$f(x) = 2x/5 \quad 2 <= x <= 3 \text{ segundos}$$

Si B recibe un “frame”, pero detecta que viene con errores (utilizando la información que agrega A al mensaje), o si viene mal según la secuencia entonces B no envía el ACK a A (pero el tiempo anterior igual se utiliza).

- Si se debe enviar ACK, este proceso en B debe entonces ponerlo bit por bit en la línea de transmisión. Este tiempo es fijo por ACK y es igual a 0.25 segundos (note que si no hay que enviar ACK, este tiempo no se “gasta”).

El tiempo que tarda en llegar a A el último bit del ACK (tiempo de propagación) es de 1 segundo. Cada ACK enviado tiene una probabilidad 0,15 de que se pierda (suponga que si llega no tiene error).

Si A ha enviado un “frame” a B, pero luego de x segundos no ha recibido de vuelta la confirmación de su llegada, A reenvía a B dicho “frame” y todos los que había enviado luego de éste. (Timer de x segundos) Note que esto puede ocurrir por tres razones: una, que el "frame" de camino a B se perdiera, otra que el "frame" llegó a B pero con errores, y la última, que el ACK se perdió en su camino a A .

A puede enviar todos los "frames" de su ventana (hasta 8), uno detrás de otro, sin esperar “ACK” (acknowledge). Si no ha recibido ningún ACK luego de enviar la ventana, A espera sin enviar más. Apenas A recibe un ACK correcto corre la ventana, descartando los "frames" que ya sabe llegaron bien a B, y envía los nuevos frames que ingresaron a la ventana (aunque no completen 8). Cuando se espera el siguiente ACK, si este no llega en el tiempo esperado (x segundos) reenvía a B todos los "frames" de la ventana. Note que A puede recibir un ACK con una valor mayor al esperado, pero esto solo significa que B recibió correctamente los "frames" pero que los ACK´s enviados se perdieron en su camino a A. Si A recibe un ACK, pero no hay mensajes que enviar, entonces ahora se espera de nuevo a completar 8 en su ventana para realizar el siguiente envío. A la ventana ingresan "frames" solo cuando hay corrimiento por un ACK recibido, por lo que pueda que en cierto momento lleguen mensajes que se quedan en cola aunque haya campo en la ventana.

Suponga que el proceso en A que se encarga de recibir los ACK´s no es el mismo que hace los envíos a B, ni el mismo que recibe los mensajes en A. De esta manera no se pregunta si está ocupado o no, se supondrá que al igual que el que recibe los mensajes que van llegando a A, siempre puede recibir y procesar un ACK. Suponga además que el tiempo de análisis del ACK recibido y el tiempo de lo que se deba hacer a consecuencia de este análisis (correr la ventana descartando mensajes que ya B recibió etc) gasta 0 tiempo.
De igual manera si se vence el timer de un "frame" enviado, el proceso encargado de reaccionar ante este evento es otro específico para esto (no es el que recibe mensajes a A, ni el que recibe ACK´s desde B, ni el que envía frames a B) y a este es el que le toca hacer las modificaciones (corrimiento de ventana en la dirección opuesta, y etc etc, gastando tiempo 0).

Diagramas

Los diagramas se encuentran adjuntos en la carpeta *diagramas*.

Estadísticas Obtenidas

Las estadísticas se encuentran adjuntas en la carpeta *registros*.

Análisis del Sistema

El análisis del sistema se hará basado en la implementación del protocolo con las distribuciones de tiempos aleatorios dadas por el enunciado y con un timer de 12 segundos. En cuanto a la longitud promedio de la cola, esta se encuentra muchas veces vacía. Tomando en cuenta que los mensajes se quedan aproximadamente 7.5 segundos en el sistema y sus tiempo de arribo de aproximadamente 25 minutos hacen que los mensajes no esperen tanto tiempo en la cola de A. Cabe destacar que la razón que provoca la espera de un mensaje en el sistema es el hecho de que el frame se pierda o se dañe y también que se llegue a perder el ACK.

El tiempo promedio de transmisión es de aproximadamente 2.6 segundos lo que deja un tiempo de servicio de 4.90 segundos. En este caso la eficiencia queda en un 0.5 aproximadamente. Si bien una eficiencia de 0.5 no es mala, esta podría mejorarse.

Sobre el protocolo en general creemos que el sistema de comunicación no es el más eficiente. El hecho de enviar uno a uno los paquetes ya denota un aumento en las probabilidades de que algo salga mal y la comunicación se retrase.