UNIVERSIDAD NACIONAL DE CÓRDOBA

FACULTAD DE CIENCIAS EXACTAS, FÍSICAS Y NATURALES



CÁTEDRA DE ALGORITMOS Y ESTRUCTURAS DE DATOS

Simulación del Tráfico de Paquetes en Internet

Antoñana, Rodrigo………………………………………………………………33242359

Diez de Medina Quintar, Lucas Leandro……..……………………….34188759

Pinzani, Paulo Augusto………………….…………………………………….34767625

Saravia Rajal, Fernando…………………………………………………….…35194107

Docente: Gustavo Wolfman - Ruben Ayme

-2010-

**Introducción:**

Se nos solicitó programar una simulación de una red de routers, con máquinas conectadas a cada uno de ellos. Dichas máquinas deben enviar páginas entre ellas, siendo tarea de los routers la distribución de las páginas y la forma en que estas se envían a su destino. Para ello, se plantea la situación real de que no todas las páginas tienen el mismo tamaño, y por ello, para evitar que las colas de envío se sobrecarguen, los routers tienen que dividir las páginas que las máquinas quieren enviar, en paquetes, todos del mismo tamaño.

Las conexiones entre los routers y las máquinas tienen ancho de banda infinito, por lo que la página puede enviarse directamente desde la máquina al router. Las conexiones entre los routers tienen un ancho de banda determinado, que indica la cantidad de paquetes que pueden mandarse por cada ciclo. De allí que un ciclo consiste en que cada router envía tantos paquetes como el ancho de banda de la conexión con el router adyacente le permita.

Las direcciones de las máquinas son tipo IP simplificadas, cuyo primer número indica el número de router al que está conectada, y el segundo número indica la máquina. Los routers tienen sólo un número que indica su dirección. Basándonos en esto, puede haber 256 routers en la red, y 65536 máquinas como máximo.

Las máquinas deben generar páginas con información, tamaño y origen pseudoaleatorios. Los routers, a su vez, deberán elegir el camino más corto para que los paquetes lleguen a su destino en el menor tiempo posible. Para ello, se solicitó que los caminos más cortos sean recalculados cada 30 ciclos. Este cálculo es realizado por un administrador del sistema, que conoce cómo son las conexiones y los anchos de banda de cada una de ellas. Para determinar el camino óptimo el administrador recibe el tamaño de las colas de envío de los routers, y utiliza una relación entre el tamaño de la cola y el ancho de banda de la conexión para determinar cuántos ciclos debe esperar un paquete para ser enviado. Debido a que cuando el paquete llega al router, el mismo tiene que esperar al próximo ciclo para ser enviado, la espera mínima es de 1 ciclo.

**Marco Teórico:**

**Estructura de Datos**: es una forma de organizar datos de tipo genérico, que permite abordar el manejo de los datos de manera global, abstrayéndose del problema particular que se plantee. A continuación exponemos una lista y una breve explicación de las estructuras de datos que se utilizaron en el proyecto.

**Cola (Queue):** Es una estructura de tipo FIFO (**F**irst**I**n, **F**irst**O**ut), donde los datos se almacenan en orden de llegada, hasta el momento en que se solicitan. Los datos se entregan en orden de llegada. Esta estructura puede compararse con la cola de un banco o de un supermercado.

**Lista (List):** Es una estructura de tipo LIFO (**L**ast**I**n, **F**irst**O**ut). Los datos que llegan se almacenan hasta que son solicitados. El primer dato que se entrega cuando se solicita información a la lista, es el último que se incorporó. Puede utilizarse cuando no es necesario mantener un orden entre los datos almacenados.

**Set:** Es la implementación de un conjunto matemático. La estructura set almacena datos no repetidos, de igual manera que lo hace un conjunto de números, por ejemplo. Las operaciones que pueden realizarse en una estructura Set, son las mismas que pueden realizarse con conjuntos matemáticos (intersección, unión, diferencia, etc.).

**Grafo:** Es una estructura que almacena los datos en vértices, que se conectan con otros datos mediante arcos. Cada conexión puede tener información asignada (etiqueta), en cuyo caso el grafo se llama grafo etiquetado. A su vez, un grafo es dirigido si las conexiones parten de un vértice y llegan a otro (es decir tienen origen y destino unívoco), y es no dirigido si las conexiones son hacia ambos lados entre dos vértices.

**Algoritmo**: Es una sucesión ordenada de pasos que nos permite obtener un resultado deseado a partir de información inicial. A continuación nombramos algunos algoritmos utilizados en el proyecto.

**Recursión:** Este algoritmo permite realizar la misma operación de manera recursiva, ingresando a una función o método varias veces hasta obtener el resultado deseado. Se utiliza entre otras cosas para recorrer listas o colas. Tiene como desventaja que al ingresar varias veces al método durante la ejecución del mismo método, las llamadas anteriores van quedando guardadas en memoria, junto con todas las variables declaradas antes de la recursión, por lo que es un método muy ineficiente para cómputos de varios datos (listas largas, por ejemplo).

**Algoritmo de Dijkstra:**Es un algoritmo desarrollado por EdsgerWybeDijkstra cuyo objetivo es encontrar el camino más corto que conecte dos vértices de un grafo. Lo importante de este algoritmo, es la tabla de precedencias que se construye al ejecutarlo, ya que en ella se detalla cuál es el vértice desde el que se debe llegar a un destino, partiendo de un determinado origen. Por lo tanto, una vez ejecutado el algoritmo, el camino se construye partiendo desde el destino, y volviendo hacia el origen, utilizando esta tabla.

**Clase**: Es la declaración explícita del comportamiento que tendrá en el programa, un determinado tipo de objetos. En una clase se indican los campos, se programan los métodos y se establece el estado inicial que un objeto tendrá al ser creado en el programa.

**Objeto**: Es una instancia de una clase. Cuando se define una clase, se indica cómo funcionarán los objetos de esa clase, pero por única vez. Luego, en el programa, pueden crearse tantos objetos de una clase como sean necesarios.

**Desarrollo:**

Para el desarrollo del sistema se tomó un tiempo de diseño para consensuar la implementación de la simulación. Se planteó el desarrollo de las siguientes clases con los comportamientos indicados a continuación:

**Clase Pagina:**

Las páginas contienen información formada por un número aleatorio de caracteres alfanuméricos elegidos al azar. Están identificadas por un número entero. Además, poseen información del origen y el destino de la página, y un campo que almacena el tamaño de la página (la cantidad de caracteres que forman la información). Todo esto es utilizado por el primer router al cual llegan, para que este la divida en paquetes.

**Clase Paquete:**

Las páginas que llegan a los routers son divididas en paquetes de igual tamaño. Luego, estos paquetes son enviados a través de la red, hasta llegar a su destino, donde son juntados, y forman nuevamente la página que se entrega a la máquina de destino. Por ello los paquetes solo almacenan información útil para los routers, como ser:

* IP de origen y de destino
* Nombre y tamaño de la página de la que forman parte
* Ubicación del paquete en la página (para saber dónde reubicarlo)
* Parte de la información de la página (en nuestro caso, un caracter)

**Clase Maquina:**

Las máquinas son objetos que representan las terminales de la red. Como tales, poseen una estructura IP que contiene dos campos enteros: router, y maquina. Estos campos guardarán la dirección IP de la máquina. La clase Maquina cuenta con un método para generar un objeto página con información y tamaño aleatorios, y asignarle un destino también aleatorio.

**Clase Router:**

La clase Router es una de las más complejas. En primer lugar, posee un campo que almacena la dirección del router. Luego contiene dos arreglos de 256 elementos que indican al router las máquinas y los routers que tiene conectados. También tiene dos conjuntos (Set) de enteros, que contienen los nombres de las páginas que deben ser enviadas a los routers vecinos, y los nombres de las páginas que deben ser enviadas a las máquinas conectadas directamente a este router.

Otro de los campos importantes es un arreglo de 256 números enteros, en el que cada índice representa a un router de la red. Esta es llamada tabla de precedencias, y será utilizada para indicarle al router, cuál es el próximo nodo de un paquete, si este quiere ser enviado a un destino específico.

Por último, el router cuenta con tres Listas de Colas de Paquetes. *La primera* es una lista de colas que tiene una cola por cada router que haya conectado. En estas colas se irán encolando los paquetes de acuerdo a cuál sea el router siguiente. *La segunda* es una lista que tiene en cada cola, los paquetes correspondientes a una página que tiene como destino final el router en cuestión. Una vez que la cola se complete con todos los paquetes de la página, la misma se rearma y se entrega a la máquina correspondiente. *La tercera* es una lista que posee una cola por cada nombre de página que llegue al router. En estas colas se agrupan los paquetes que formen parte de la misma página. Cada vez que un paquete llega al router, es organizado en esta lista de colas, y luego se toma de a un paquete por cola, y se los va distribuyendo en las colas que correspondan, dirigidas a los siguientes routers. De esta manera, los paquetes quedan siempre entrelazados y el servidor no se satura si hay una página muy pesada.

Los métodos más importantes de esta clase son:

* **nextRouter**: Devuelve el próximo router al que hay que mandar un paquete para que llegue a su destino por el camino más corto.
* **recibirPaquete**: Recibe un paquete y lo encola en la cola pagesToRout y al final de cada ciclo es distribuido en las colas que corresponda para ser enviado.
* **repartirPaquetes**: Reparte los paquetes que están en la cola pagesToRout hacia las colas de cada router o de cada máquina, según corresponda.
* **recibirPagina**: Recibe una página de una maquina y la desarma en paquetes para ser encolados en su correspondiente cola.

**Clase Admin:**

La clase admin tiene dos campos, uno es una matriz de 256x256 elementos que almacena los anchos de banda de las conexiones entre los routers. El otro campo es un grafo (no dirigido) cuyos vértices son números enteros (direcciones de los routers) y cuyas etiquetas son calculadas de la siguiente manera:

(tamaño de cola)/(ancho de banda) +1.

El método más importante de esta clase es “encontrarCMC”, el cual cumple la función de modificar la tabla de precedencias del router y así poder encontrar el camino más corto.

**Clase Principal:**

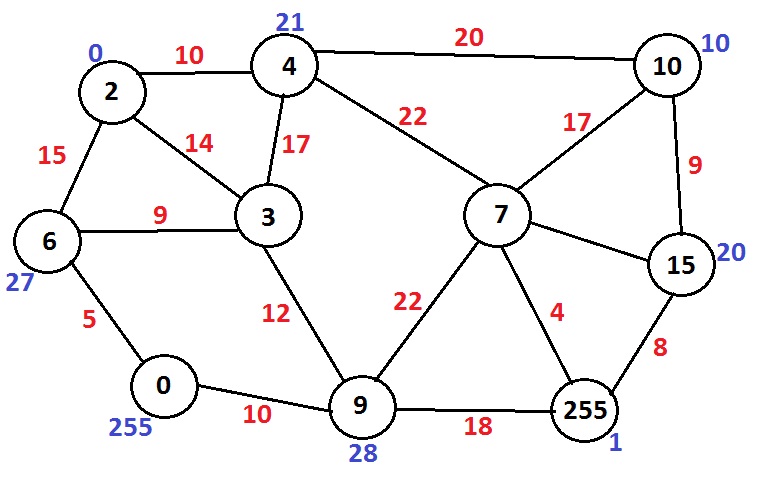
En esta clase se crea un administrador, un arreglo de 256 punteros a routers, una matriz de 256x256 punteros a maquinas, una matriz de 256x256 enteros que almacenan los anchos de banda entre las conexiones.

Esta clase tiene los siguientes métodos:

* **leerArchivo**: este método lee el archivo de parametrizacion y configura las conexiones y los anchos de banda entre las mismas.
* **getMaquinaAlAzar**: este método devuelve una maquina al azar, verificando que la misma exista, para luego operar con ella.
* **calcularCaminosMasCortos**: este método recorre todos los routers que existen en la red y calcula la tabla de precedencias de todos ellos.
* **enviarPaginas**: Cada 30 ciclos, se envían 43 páginas aleatoriamente de cualquier origen a cualquier destino.
* **operarRouters**: este método cumple 3 funciones:
  + - * + Se mandan los paquetes correspondientes de cada router a sus routers vecino de acuerdo al ancho de banda de la conexión entre los routers.
        + Luego se reparten los paquetes que llegaron a cada router en las colas correspondientes dentro del mismo.
        + Se verifican si hay páginas completas para entregar en cada router, en caso afirmativo se procede a entregarlas.
  + **recalcularPesos**: Con este método vuelven a calcularse los pesos de la conexiones entre los routers con la formula:

(tamaño de cola)/(ancho de banda) +1.

Funcionamiento:



En el programa se configuro la red de la figura anterior, a la cual se la hizo operar de la siguiente manera:

1. Se calculan los caminos más cortos para todas las conexiones teniendo en cuenta solo el ancho de banda.
2. Se crea un ciclo que hace operar los routers con el método **operarRouters** (explicado anteriormente).
3. Inicialmente y cada 30 de estos ciclos se obtienen maquinas al azar, las cuales generan paginas aleatorias que son enviadas a maquinas destino. También cada 30 ciclos se re calculan los caminos más cortos y estos se les informan a los routers.

Ejemplo de recorrido de una página:

Supongamos que se quiere enviar 2 páginas desde la maquina 9.28 a la maquina 4.21, entonces:

Estas 2 páginas son enviadas desde la maquina 28 al router 9, donde aquí son divididas en paquetes y puestos en la lista pageToRout en dos colas separadas, agrupando los paquetes de cada página.

El camino más corto para ir del router 9 al router 4 es pasando por el router 7 debido a que este posee más ancho de banda. Por lo tanto todos los paquetes se encolaran en la cola que va al router 7. Para que queden intercalados los paquetes de cada página se va tomando un paquete de cada cola en pageToRout y se los va encolando de a uno en la cola del router 7.

Una vez en la cola del router 7, se envían 22 paquetes, 11 de una página y 11 de la otra, quedando los restantes para ser enviados en el próximo ciclo.

El router 7 los recibe y los encola en 2 colas separadas (una por cada página) en la lista pageToRout y repite el procedimiento anterior.

Una vez que en el router 4 se encuentran todos los paquetes de las paginas, o al menos de una, el mismo se encarga de armar la pagina y entregársela a la maquina destino, en este caso la número 21.

De esta manera mostramos como se envía una página, siendo este el funcionamiento general del programa para N maquinas.

**Conclusión:**

El trabajo fue complicado, extenso y requirió un diseño previo a la programación, a lo cual no estábamos acostumbrados.

La aplicación de estructuras de datos nos permitió abstraernos de los tipos de datos que manejábamos y utilizarlos para mejorar la organización. De esta forma, pudimos simplificar el problema mayor, en sub-problemas más pequeños.

La utilización de algoritmos vistos en clase, nos facilito el análisis y procedimiento y la operación para cumplir determinados objetivos (como encontrar el camino mas corto).