Facultad de
Ciencias Exactas
Físicas y Naturales

Algoritmos y Estructura de Datos

Final 2019: “Simulación de tráfico de Internet”

Profesor:
Wolffman, Gustavo.
Aime, Ruben.

Alumnos:
Lujan, Martin - 39448179

- 2020-

Introducción

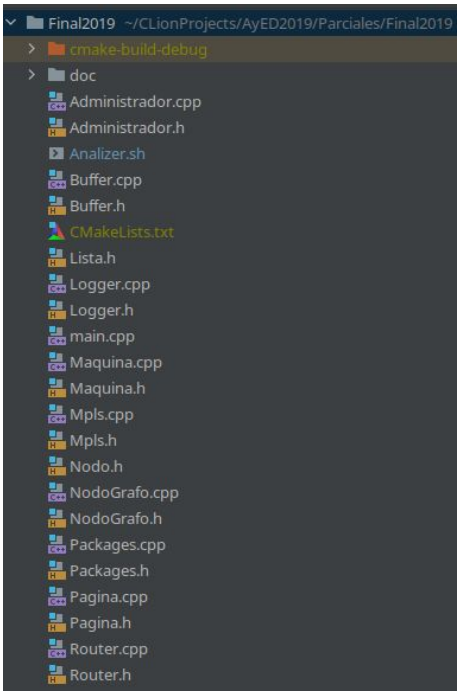
En el siguiente proyecto se deberá desarrollar un programa que realice una simulación del tráfico de datos, al “estilo” de internet. Esto implica que se deberá simular computadoras que generan datos, para enviarlo a routers, los cuales se encargaran de dividir el dato recibido en paquetes y redireccionarlo hacia el buffer de salida correspondiente. En caso de que el router sea intermediario entre el router de origen y el de destino (donde se encuentran conectadas las máquinas) este moverá los paquetes a los buffers correspondientes. Pero en el caso de que el router sea el final, este deberá esperar a que reciba todos los paquetes necesarios para formar el dato y entregarlo a la máquina destino. A todo esto se suma un “administrador” el cual periódicamente observará los routers para determinar el tamaño de sus buffers y así calcular el peso del enlace, mediante Dijkstra. Será desarrollado en el lenguaje C++, con el entorno CLion y funcionando en Archlinux.

Desarrollo

Al igual que los anteriores proyectos, el problema se desglosó en partes para segmentar el código en partes, mejorando su sintaxis. Estas son:

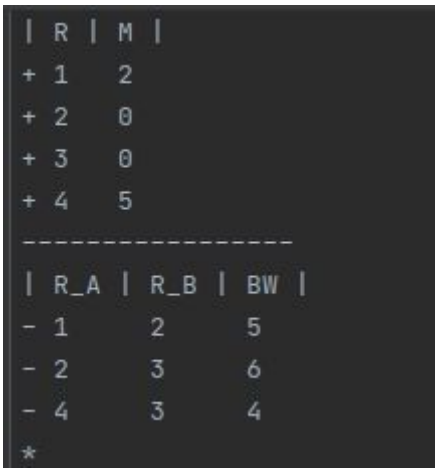
- Administrador
- Buffer
- Lista
- Logger
- Máquina
- Mpls
- Nodo
- Packages
- Página
- Router

Como se observa en la siguiente imagen:



El proyecto es grande, por lo que se necesitaron de varias clases pero con un rol específico. Se dará una breve descripción de cada clase y cómo aporta cada una al proyecto.
Comenzando por la clase de **Página** es aquella que es generada por las máquinas para ser enviadas en a los routers, este objeto contiene los campos de dato (string), los IP de origen/destino y el ID de cada página.
La clase **Packages** contiene los campos necesarios para dividir las páginas generadas por las máquinas en paquetes más pequeños capaces de ser transmitidos por la red. Los campos de la clase son, *letral* (char), *origen, destino, fram, frametotal* (es la cantidad de paquetes en los que se divide una página).
Continuando con la clase **Buffer** como bien indica su nombre, forman parte del buffer de los routers. Estos contienen los campos de *cola* (lista de paquetes) y ID, el cual indica la salida al router al que se encuentra conectado.
Router es una de las clases mas complejas, debido a que forma un papel importante en la red. Contiene los campos de *IPRouter* y 6 listas entre las cuales se encuentra *Input, Rvecinos, BufferSalida, Máquinas, etc.* Por último queda el numero de router *N_R*.
Máquina simula las computadoras conectadas a la red mediante los routers. Estas máquinas generan datos aleatorios que serán enviadas a otras máquinas que pueden encontrarse en la red local u en otra. Alguno de los campos de máquina son, *IPMaquina, PaginasEnviadas,* dos listas, una para las páginas enviadas y otra para las recibidas, entre otras.
NodoGrafo es la clase utilizada para la generación del grafo de la red, el cual contiene información como *router, peso* (del enlace) y *ancho de banda disponible en la conexión.*
Toda esta información es utilizada por la clase **Mpls** (En honor al protocolo) que es la encargada de dictaminar cuál es el mejor camino en la red. Esta clase utiliza el algoritmo de Dijkstra, por lo que implementa todos los métodos y campos que este necesita. Esta clase es creada cada vez que se debe enrutar un paquete, se encarga de en base a los pesos de los enlaces, devolver el mejor camino para dicho paquete.
La clase más importante es **Administrador** la cual es la encargada de instancias las demás y hacer uso de cada una de ellas. Esta orquesta el programa, ya que lee el archivo parametrizador, crea el gráfico y contiene el método "*Simulate*" el cual pondrá en marcha la simulación de internet. Este se ejecuta hasta que las máquinas reciban todas las páginas que se enviaron en total, aquí finaliza el programa.
Se utiliza la clase **Logger** para escribir en un txt el LOG del programa, cada evento que se produce en el programa es ingresado en este txt para su posterior análisis.
Las clases **Lista** y **Nodo** se escribieron con *templates* la cual nos permite generar instancias de estas clases con distintos tipos de dato, ya sea otro objeto u del tipo definido como int, string, etc. Esto nos permite ahorrar código y tener una Lista "genérica" que nos permite utilizarla tantas veces como necesitemos.
Por último se debió implementar un script en bash, para poder analizar el LOG con más facilidad dado a que este tiene mucho flujo de información. Este simple script nos permite filtrar un paquete específico enviado por la red, para ver todos los saltos que tuvo a lo largo de esta, y verificar que llegó a destino con éxito. Además nos permite variar entre los datos de una misma palabra, para chequear que los paquetes toman distintos caminos pero llegan al mismo destino.

La estructura del archivo parametrizador debe ser la siguiente:



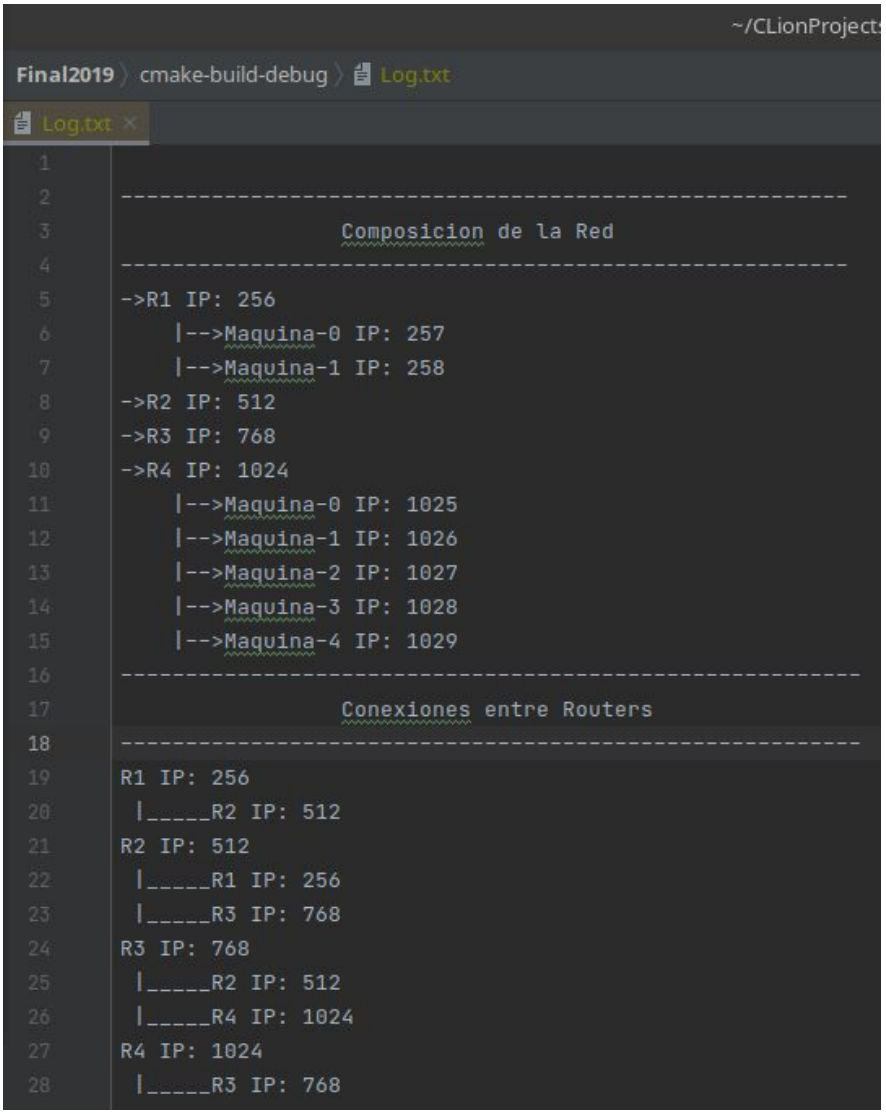
```
| R | M |
+ 1   2
+ 2   0
+ 3   0
+ 4   5
-----------------
| R_A | R_B | BW |
- 1     2     5
- 2     3     6
- 4     3     4
*
```

Esta estructura se encuentra dividida en 2 secciones, la primera es la creación de los routers con sus respectivas máquinas conectadas a estos. El programa acepta que un router tenga “0” (cero) máquinas conectadas, es decir que sea solo para ruteo de paquetes. La segunda seccion claramente divida, es la encargada de configurar las conexiones entre todos los routers de la red. Cada columna indica:

- **R_A**: Router A
- **R_B:** Router B
- **BW:** Ancho de Banda entre ellos

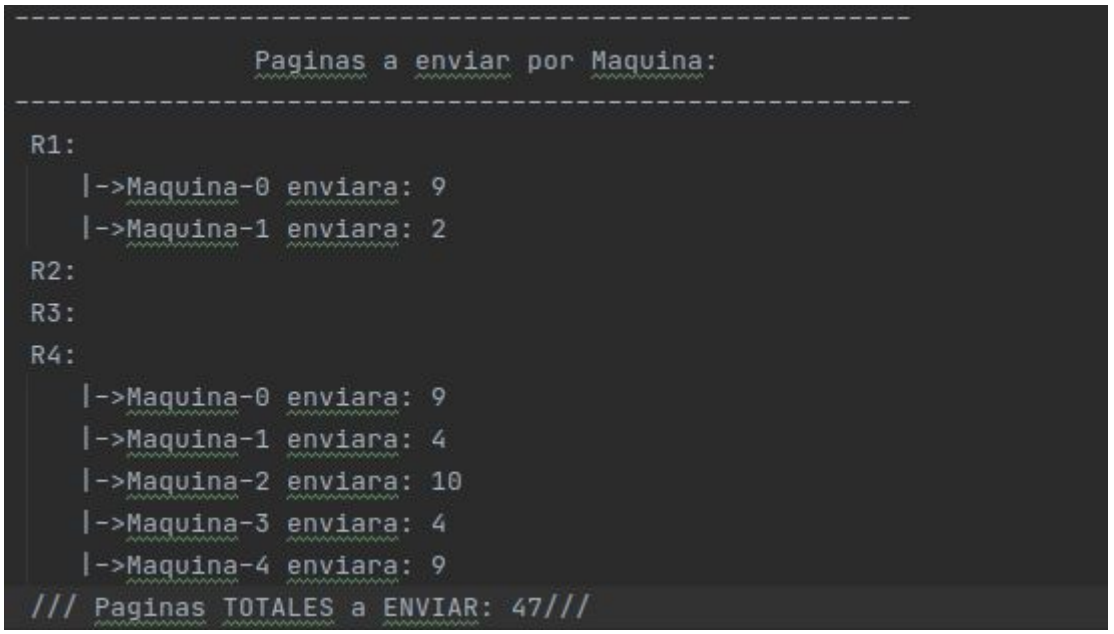
Solo se debe ingresar una vez la conexión entre el R_A y el R_B, ya que esta es bidireccional a la hora de crear el grafo. Como carácter final nos encontramos el asterisco (*) indicando el final del archivo de configuración de la red.

Como punto final, se explora el formato del LOG que genera como salida. Se debe aclarar que tanto los mensajes que salen por pantalla como los del archivo de texto, son los mismos y en el mismo orden.
Lo primero que se muestra por pantalla y se imprime en LOG es estructura del gráfico:



La primera sección es la *Composición de la red*, la cual lista la cantidad de routers, su dirección IP más todas las máquinas conectadas que tienen estos, con su respectiva IP también. *Conexiones entre Routers* se lista la topología de la red entre los routers. El formato se comprende por el número de router con su IP, más todos los routers conectados a este.

En la segunda sección, se muestran la cantidad de páginas a enviar por maquina y el total que se enviará.



A continuación se describe cada uno de los eventos que se registran en el log, acompañado de ejemplos. El primero que nos encontramos es el evento "*SendPage Num"* el cual es la creación de una página de cada máquina con la información que viaja en la misma. Por ejemplo:

```
SendPag Num: 1 | MaquinaIp: 257 | Restan: 8 | Palabra: jiaqktytym | Origen: 257 | Destino: 1028 | Largo: 10
SendPag Num: 1 | MaquinaIp: 258 | Restan: 1 | Palabra: jclaunxrcz | Origen: 258 | Destino: 1026 | Largo: 10
SendPag Num: 1 | MaquinaIp: 1025 | Restan: 8 | Palabra: wl | Origen: 1025 | Destino: 258 | Largo: 2
SendPag Num: 1 | MaquinaIp: 1026 | Restan: 3 | Palabra: ruyneciw | Origen: 1026 | Destino: 257 | Largo: 8
SendPag Num: 1 | MaquinaIp: 1027 | Restan: 9 | Palabra: nyhtjassky | Origen: 1027 | Destino: 257 | Largo: 10
SendPag Num: 1 | MaquinaIp: 1028 | Restan: 3 | Palabra: fy | Origen: 1028 | Destino: 257 | Largo: 2
SendPag Num: 1 | MaquinaIp: 1029 | Restan: 8 | Palabra: euodwpyhuy | Origen: 1029 | Destino: 258 | Largo: 10
```

El proximo evento es *"Input-Output"* que se refiere a tomar las paginas que tenemos en la cola de entrada del router, fraccionarla en paquetes y determinar a donde viaja, para enviarlo al buffer correspondiente. Por ejemplo:

```
Input-Output | Turno: 1 | R4 | N°Paquete 0 Pag N°1 | Dato: r | Origen: 1026 | Destino: 257 | Buffer a R3 | Camino a recorrer: 3 2 1
Input-Output | Turno: 1 | R4 | N°Paquete 1 Pag N°1 | Dato: u | Origen: 1026 | Destino: 257 | Buffer a R3 | Camino a recorrer: 3 2 1
Input-Output | Turno: 1 | R4 | N°Paquete 2 Pag N°1 | Dato: y | Origen: 1026 | Destino: 257 | Buffer a R3 | Camino a recorrer: 3 2 1
Input-Output | Turno: 1 | R4 | N°Paquete 3 Pag N°1 | Dato: n | Origen: 1026 | Destino: 257 | Buffer a R3 | Camino a recorrer: 3 2 1
Input-Output | Turno: 1 | R4 | N°Paquete 4 Pag N°1 | Dato: e | Origen: 1026 | Destino: 257 | Buffer a R3 | Camino a recorrer: 3 2 1
Input-Output | Turno: 1 | R4 | N°Paquete 5 Pag N°1 | Dato: c | Origen: 1026 | Destino: 257 | Buffer a R3 | Camino a recorrer: 3 2 1
Input-Output | Turno: 1 | R4 | N°Paquete 6 Pag N°1 | Dato: i | Origen: 1026 | Destino: 257 | Buffer a R3 | Camino a recorrer: 3 2 1
Input-Output | Turno: 1 | R4 | N°Paquete 7 Pag N°1 | Dato: w | Origen: 1026 | Destino: 257 | Buffer a R3 | Camino a recorrer: 3 2 1
```

Aquí observamos que la página Número 1 de la máquina con IP 1025 es dividida en 8 paquetes más pequeños. Notar que este número coincide con el "largo" del dato informado en el evento anterior. Finalmente se determina el mejor camino para cada uno. Se debe tener en cuenta de que este camino puede variar dependiendo de la congestión de la red.

Continuando con los eventos, tenemos *"Ruteo"* el cual como bien indica la palabra, se produce el salto de los paquetes de un router a otro.

```
Ruteo | R1 | N°Paquete 0 Pag N°1 | IPOrigen 257 | IPDestino 1028 | Out RouterIP: 256 to RouterIP: 512 | BW enlace: 5 | BW restante: 4
Ruteo | R1 | N°Paquete 0 Pag N°1 | IPOrigen 258 | IPDestino 1026 | Out RouterIP: 256 to RouterIP: 512 | BW enlace: 5 | BW restante: 3
Ruteo | R1 | N°Paquete 1 Pag N°1 | IPOrigen 257 | IPDestino 1028 | Out RouterIP: 256 to RouterIP: 512 | BW enlace: 5 | BW restante: 2
Ruteo | R1 | N°Paquete 1 Pag N°1 | IPOrigen 258 | IPDestino 1026 | Out RouterIP: 256 to RouterIP: 512 | BW enlace: 5 | BW restante: 1
Ruteo | R1 | N°Paquete 2 Pag N°1 | IPOrigen 257 | IPDestino 1028 | Out RouterIP: 256 to RouterIP: 512 | BW enlace: 5 | BW restante: 0
Ruteo | R4 | N°Paquete 0 Pag N°1 | IPOrigen 1025 | IPDestino 258 | Out RouterIP: 1024 to RouterIP: 768 | BW enlace: 4 | BW restante: 3
Ruteo | R4 | N°Paquete 0 Pag N°1 | IPOrigen 1026 | IPDestino 257 | Out RouterIP: 1024 to RouterIP: 768 | BW enlace: 4 | BW restante: 2
Ruteo | R4 | N°Paquete 1 Pag N°1 | IPOrigen 1025 | IPDestino 258 | Out RouterIP: 1024 to RouterIP: 768 | BW enlace: 4 | BW restante: 1
Ruteo | R4 | N°Paquete 1 Pag N°1 | IPOrigen 1026 | IPDestino 257 | Out RouterIP: 1024 to RouterIP: 768 | BW enlace: 4 | BW restante: 0
```

Aquí se envían los paquetes a los routers correspondientes, pero no se envían todos juntos, porque se estaría dando prioridad a algunos paquetes sobre otros, por lo que se envía un paquete de cada dato encolado mientras haya ancho de banda disponible en el enlace (BW). De hecho, se informa el ancho de banda restante en cada envío.

Después del evento "*Ruteo*" nos podemos encontrar con el evento "Llegada" el cual nos indica que el paquete antes ruteado, llego al router destino. Este evento informa el numero de router, el numero de paquete, el numero de pagina al que corresponde, el dato contenido y su destino final. Por ejemplo:

```
Ruteo | R2 | N°Paquete 9 Pag N°9 | IPOrigen 1029 | IPDestino 258 | Out RouterIP: 512 to RouterIP: 256 | BW enlace: 5 | BW restante: 2
Llegada | R1 | N°Paquete: 9 Pag N°9 | Dato: h | Destino: 258
```

Debemos saber cuando se produce el cálculo de los pesos en los enlaces, esto está definido que cada 2 turnos. Para esto se sumó el evento *"Recomputan su peso"*, nos muestra la actualización del estado de la red. Por ejemplo:

```
Recomputan su peso | R1 IP: 256 y R2 IP: 512 | Peso actualizado: 7
Recomputan su peso | R2 IP: 512 y R1 IP: 256 | Peso actualizado: 1
Recomputan su peso | R2 IP: 512 y R3 IP: 768 | Peso actualizado: 1
Recomputan su peso | R3 IP: 768 y R2 IP: 512 | Peso actualizado: 1
Recomputan su peso | R3 IP: 768 y R4 IP: 1024 | Peso actualizado: 2
Recomputan su peso | R4 IP: 1024 y R3 IP: 768 | Peso actualizado: 23
```

Para el cálculo de los pesos se utiliza el grafo, tomamos las conexiones entre los routers, uno como "padre" y el segundo como "hijo". Tomamos el largo de la cola del padre, respecto a su hijo y lo dividimos por el ancho de banda, obteniendo el peso del router 1, respecto al 2 por ejemplo. Esto se repite para todos las conexiones, debido a que la congestión entre el R1 y R2 puede no ser la misma entre R2 y R1.

Por último y no menos importante, tenemos el evento de *"Recepción"* el cual como bien indica, se produce cuando todos los paquetes se encuentran en el router destino, este los ensambla, formando la página y se la entrega a la máquina correspondiente. Por ejemplo:

```
Recepcion | MaquinaIP: 258 | N° Pag 1 | Origen 1025 | Destino 258 | Dato: wl
Recepcion | MaquinaIP: 1025 | N° Pag 3 | Origen 1029 | Destino 1025 | Dato: trtesumvpy
Recepcion | MaquinaIP: 1025 | N° Pag 3 | Origen 1026 | Destino 1025 | Dato: dwiidmw
```

En este turno se puede ver que solo se recibieron 3 Páginas, que no son consecutivas verificando así que unos paquetes pueden llegar en cualquier orden, dada la congestión de la red.

Conclusión

El programa se pudo concluir con éxito y satisfecho con los resultados. No fue tarea fácil y se produjeron bastantes problemas en el camino, pero cada uno fue resuelto con los conocimientos obtenidos en la materia. Al principio se dificulto el planteamiento del problema, ya que la abstracción y la falta de otros conocimientos de red pueden generar complicaciones.

Este trabajo nos da una idea del potencial de los grafos para simular estructuras de red o todo aquello que represente un “camino”. Además es observable el potencial de algoritmo de Dijkstra, que nos permite en base a complejos cálculos matemáticos la determinación del mejor camino, siempre.

Puede ser discutible si la implementación de listas para los buffers, y demás elementos fue el mejor camino, pero a mi criterio fue la manera más simple de resolverlo.

Sin dudas lo que no es discutible es el potencial que existe en la programación y me incentiva a seguir aprendiendo en esta fantástica profesión que elegí para mi futuro.