CI-0117 Laboratorio 8 Paralela y Concurrente
Rodrigo Vilchez Ulloa | B78292

---

1. Estrategia Personalizada

• La estrategia personalizada consiste en un mapeo similar al que se realiza en el mapeo cíclico, solo que en este caso las líneas se asignan de forma descendente, por lo que la distribución de la carga sí estaría balanceada pues solo se asigna una línea más a ciertos trabajadores en relación con los otros cuando el número de lineas no es divisible entre el número de trabajadores.

2. Clase FileReader

• La clase utiliza dos métodos esenciales, además del destructor y constructor, que son **hasNext()** y **getNext()**, además de seis atributos los cuales son: el nombre del archivo HTML, un puntero a un *std::mutex*, la cantidad de trabajadores para ese archivo, la estrategia a utilizar para distribuir las líneas por trabajador, el total de líneas que posee el archivo HTML, un puntero a un arreglo que es utilizado por tres de las cuatro estrategias para determinar cuál es la línea que le corresponde leer a trabajador en particular en un momento determinado del programa, y la cantidad de líneas leídas (esta variable se utiliza para el mapeo dinámico). El constructor de la clase recibe cinco parámetros, los cuales son el nombre del archivo, número de trabajadores, estrategia, el total de líneas y un puntero a un mutex. Dentro del constructor, se asigna espacio a un array a partir de la cantidad de trabajadores que se haya seleccionado. Este array almacenará el número de línea que le corresponde leer a un trabajador cuando este solicite una, por lo que inicialmente, para el mapeo por bloques, cíclico y personalizado, el constructor deberá darle valores iniciales a este array para que en un primer momento, cada trabajador lea una línea diferente. El número de la primera línea que deba leer cada trabajador dependerá de la estrategia utilizada. Para asignar el número de la primera línea en el caso de **mapeo por bloques**, se utiliza la siguiente fórmula: $techo(total\_lineas/cantidad\_trabajadores) * i + 1$, donde i corresponde a un iterador que va desde 0 hasta $cantidad\_trabajadores - 1$, techo significa que redondea hacia arriba. Para el **mapeo cíclico**, se utiliza la fórmula $i + 1$ donde i es el iterador que se comporta de la misma forma que en el caso anterior, por lo que la línea 1 se la asgina al primer trabajador, la línea 2 al segundo y así sucesivamente. Para el **mapeo personalizado**, se utilizó $total\_lineas - i$, de forma que al primer trabajador le corresponde inicar en la última línea, al segundo trabajador en la penúltima línea y así sucesivamente. En el caso del destructor, este se encarga de liberar el espacio de memoria asignado a este array. El total de líneas que posee el archivo HTML se calcula utilizando un *getline()* que va incrementando en uno a un contador hasta que ya no hayan más líneas en el archivo. El método **hasNext()** se comporta de manera distinta según la estrategia utilizada. En el caso de **mapeo por bloques**, a un trabajador le restan líneas por leer si el número de línea que le toca leer es menor que el número total de líneas y si esa línea es menor que $techo(total\_lineas/cantidad\_trabajadores) * (num\_trabajador + 1) + 1$, esto es así pues en el mapeo por bloques le asigna n líneas a un trabajador, siendo estas líneas consecutivas, al segundo trabajador se le asigna n líneas a partir de las líneas del trabajador anterior, de esta forma una línea no le es enviada a dos trabajadores a la vez. Para el **mapeo ciclico**, un trabajador sabe que todavía tiene líneas por leer si la línea que le toca leer es menor que la cantidad total de líneas. En el caso de **mapeo dinámico**, un trabajador sabe que puede seguir leyendo líneas cuando la cantidad de líneas leídas por todos los trabajadores es menor que la cantidad total de líneas. En el caso de **mapeo personalizado**, el trabajador puede seguir leyendo líneas si el número de línea que le toca leer es mayor que cero. Para cada estrategia, si su condición se cumple devuelve un

true, de lo contrario un false. La función **getNext()** le otorga un *std::string*, que es una línea en particular, a cada trabajador cuando este lo solicita. Este método recibe el número de trabajador que está solicitando la línea y la línea otorgada depende de lo que indique el array que posee la línea que le toca leer a cada trabajador. Para cada estrategia, una vez que se obtuvo la línea que le corresponde al trabajador, su celda del array se actualiza al número de línea que le corresponda leer en una siguiente solicitud.

3. Pruebas

Para compilar el programa, se utiliza el comando **make**, y para correr el programa se utiliza la siguiente sintaxis de un comando:

```
$ ./lector Nombre_Archivo Cantidad_Trabajadores Estrategia $
```

Para definir el tipo de estrategia, 1 corresponde a mapeo por bloques, 2 corresponde a mapeo cíclico, 3 a mapeo dinámico y 4 a mapeo personalizado.
Para las pruebas se utilizará un archivo con el siguiente contenido:

```
A
B
C
D
E
F
G
H
I
J
K
L
M
N
```

• Especificaciones de las pruebas: para las pruebas nada más se imprime la línea que el trabajador está recibiendo en cierto momento para demostrar la funcionalidad de la asignación de líneas a los trabajadores, el conteo de etiquetas HTML se implementará para la tarea programada. En el caso de **mapeo dinámico**, algunos trabajadores imprimen una línea vacía, esto sucede porque al momento de llamar a **hasNext()**, este devuelve que todavía ese trabajador puede seguir leyendo líneas porque la cantidad de líneas leídas no ha excedido el total de líneas del archivo, sin embargo, al ser un problama ejecutado con hilos, este valor puede cambiar después de que este método devolviera True, y cuando el trabajador vaya a solicitar una línea, sucede que ese número de línea no existe en el archivo porque ya se excedió el total, por lo que nada más envía un *std::string* sin ningún contenido, esto pensando que para la tarea programada, la funcionalidad de contar etiquetas no se verá afectada pues es una línea vacía.

- Mapeo por Bloques, 4 trabajadores

```
rigovil@rodrigo:~/Escritorio/UCR/I Semestre 2020/CI-0117/Laboratorios/Semana 8$ ./lector prueba.txt 4 1
Trabajador #0: A
Trabajador #1: E
Trabajador #2: I
Trabajador #0: B
Trabajador #2: J
Trabajador #1: F
Trabajador #0: C
Trabajador #2: K
Trabajador #0: D
Trabajador #1: G
Trabajador #2: L
Trabajador #1: H
Trabajador #3: M
Trabajador #3: N
rigovil@rodrigo:~/Escritorio/UCR/I Semestre 2020/CI-0117/Laboratorios/Semana 8$
```

- Mapeo por Bloques, 2 trabajadores

```
rigovil@rodrigo:~/Escritorio/UCR/I Semestre 2020/CI-0117/Laboratorios/Semana 8$ ./lector prueba.txt 2 1
Trabajador #0: A
Trabajador #1: H
Trabajador #0: B
Trabajador #1: I
Trabajador #0: C
Trabajador #1: J
Trabajador #0: D
Trabajador #1: K
Trabajador #0: E
Trabajador #1: L
Trabajador #0: F
Trabajador #1: M
Trabajador #0: G
Trabajador #1: N
rigovil@rodrigo:~/Escritorio/UCR/I Semestre 2020/CI-0117/Laboratorios/Semana 8$
```

- Mapeo Cíclico, 4 trabajadores

```
rigovil@rodrigo:~/Escritorio/UCR/I Semestre 2020/CI-0117/Laboratorios/Semana 8$ ./lector prueba.txt 4 2
Trabajador #0: A
Trabajador #0: E
Trabajador #1: B
Trabajador #0: I
Trabajador #1: F
Trabajador #0: M
Trabajador #2: C
Trabajador #3: D
Trabajador #1: J
Trabajador #3: H
Trabajador #1: N
Trabajador #2: G
Trabajador #3: L
Trabajador #2: K
rigovil@rodrigo:~/Escritorio/UCR/I Semestre 2020/CI-0117/Laboratorios/Semana 8$
```

- Mapeo Cíclico, 7 trabajadores

```
rigovil@rodrigo:~/Escritorio/UCR/I Semestre 2020/CI-0117/Laboratorios/Semana 8$ ./lector prueba.txt 7 2
Trabajador #0: A
Trabajador #0: H
Trabajador #1: B
Trabajador #3: D
Trabajador #1: I
Trabajador #3: K
Trabajador #2: C
Trabajador #4: E
Trabajador #2: J
Trabajador #4: L
Trabajador #5: F
Trabajador #5: M
Trabajador #6: G
Trabajador #6: N
rigovil@rodrigo:~/Escritorio/UCR/I Semestre 2020/CI-0117/Laboratorios/Semana 8$
```

- Mapeo Dinámico, 5 trabajadores

```
rigovil@rodrigo:~/Escritorio/UCR/I Semestre 2020/CI-0117/Laboratorios/Semana 8$ ./lector prueba.txt 5 3
Trabajador #0: A
Trabajador #1: B
Trabajador #2: C
Trabajador #3: F
Trabajador #2: G
Trabajador #1: E
Trabajador #3: I
Trabajador #0: D
Trabajador #2: J
Trabajador #4: H
Trabajador #3: K
Trabajador #4:
Trabajador #1: L
Trabajador #2: N
Trabajador #0: M
rigovil@rodrigo:~/Escritorio/UCR/I Semestre 2020/CI-0117/Laboratorios/Semana 8$
```

- Mapeo Dinámico, 4 trabajadores

```
rigovil@rodrigo:~/Escritorio/UCR/I Semestre 2020/CI-0117/Laboratorios/Semana 8$ ./lector prueba.txt 4 3
Trabajador #0: A
Trabajador #0: C
Trabajador #2: D
Trabajador #0: E
Trabajador #1: B
Trabajador #0: G
Trabajador #2: F
Trabajador #1: H
Trabajador #2: K
Trabajador #1: L
Trabajador #2: M
Trabajador #0: J
Trabajador #3: I
Trabajador #1:
Trabajador #2: N
rigovil@rodrigo:~/Escritorio/UCR/I Semestre 2020/CI-0117/Laboratorios/Semana 8$
```

- Mapeo Personalizado, 3 trabajadores

```
rigovil@rodrigo:~/Escritorio/UCR/I Semestre 2020/CI-0117/Laboratorios/Semana 8$ ./lector prueba.txt 3 4
Trabajador #0: N
Trabajador #1: M
Trabajador #2: L
Trabajador #1: J
Trabajador #2: I
Trabajador #1: G
Trabajador #2: F
Trabajador #1: D
Trabajador #2: C
Trabajador #1: A
Trabajador #0: K
Trabajador #0: H
Trabajador #0: E
Trabajador #0: B
rigovil@rodrigo:~/Escritorio/UCR/I Semestre 2020/CI-0117/Laboratorios/Semana 8$
```

- Mapeo Personalizado, 4 trabajadores

```
rigovil@rodrigo:~/Escritorio/UCR/I Semestre 2020/CI-0117/Laboratorios/Semana 8$ ./lector prueba.txt 4 4
Trabajador #0: N
Trabajador #0: J
Trabajador #0: F
Trabajador #1: M
Trabajador #2: L
Trabajador #1: I
Trabajador #0: B
Trabajador #1: E
Trabajador #2: H
Trabajador #1: A
Trabajador #2: D
Trabajador #3: K
Trabajador #3: G
Trabajador #3: C
rigovil@rodrigo:~/Escritorio/UCR/I Semestre 2020/CI-0117/Laboratorios/Semana 8$
```

- Mapeo Dinámico, 3 trabajadores pero solo 2 procesan el archivo, el trabajador #2 recibe líneas vacías.

```
rigovil@rodrigo:~/Escritorio/UCR/I Semestre 2020/CI-0117/Laboratorios/Semana 8$ ./lector prueba.txt 3 3
Trabajador #0: A
Trabajador #0: B
Trabajador #0: C
Trabajador #2:
Trabajador #1: D
Trabajador #2:
Trabajador #0: E
Trabajador #1: F
Trabajador #2:
Trabajador #1: H
Trabajador #2:
Trabajador #0: G
Trabajador #1: I
Trabajador #0: J
Trabajador #1: K
Trabajador #0: L
Trabajador #1: M
Trabajador #2:
Trabajador #1:
Trabajador #0: N
rigovil@rodrigo:~/Escritorio/UCR/I Semestre 2020/CI-0117/Laboratorios/Semana 8$
```