OBSERVACIONES DEL RETO 4

Martín Rincón Cod 201914114
Mariana Ruiz Cod 202011140

	Máquina 1	Máquina 2
Procesadores	Intel(R) Core(TM) i5-9300H CPU @ 2.40GHz 2.40 GHz	Intel(R) Core(TM) i3-1005G1 CPU @ 1.20GHz 1.19 GHz
Memoria RAM (GB)	16GB	8GB
Sistema Operativo	Windows 10 64-bits	Windows 10 Pro 64 bits

Tabla 1. Especificaciones de las máquinas para ejecutar las pruebas de rendimiento.

Carga del catálogo

Carga del catálogo

Máquina	Consumo de Datos [kB]	Tiempo de Ejecución [ms]
1	4338.4	2233.92
2	4973.4	2408.4

Tabla 2. Comparación de consumo de datos y tiempo de ejecución para carga de catálogo para ambas máquinas

Requerimiento 1 (Grupal)

Análisis de complejidad: Este requerimiento se soluciona implementando el algoritmo de Kosaraju sobre el grafo que representa las conexiones entre landing points. Por lo anterior, la complejidad del requerimiento será la misma del algoritmo, O(V+E).

En este caso tenemos 4928 conexiones y 1444 landing points, de modo que el algoritmo tendrá que realizar, a lo sumo, 6372 operaciones; se trata de un número relativamente pequeño de operaciones, por lo que es consistente con los resultados obtenidos en la toma de datos.

El algoritmo de Kosaraju se basa en DFS, el cual es un algoritmo recursivo. Los algoritmos recursivos, por su naturaleza, tienen un alto consumo de memoria, lo cual se muestra en la memoria registrada por las tomas.

Requerimiento 1

Máquina	Consumo de Datos [kB]	Tiempo de Ejecución [ms]
1	136.41	467.22
2	135.102	453.79

Tabla 3. Comparación de consumo de datos y tiempo de ejecución del requerimiento 1 para ambas máquinas

Requerimiento 2 (Grupal)

Análisis de complejidad: El requerimiento se desarrolla implementando el algoritmo de Dijkstra. Se sabe que este algoritmo tiene un consumo en memoria proporcional a V (1444), un valor casi igual a los KB consumidos en la toma de datos. En cuanto a tiempo de ejecución, el grafo usa como estructura de datos la lista de adyacencia, por lo que la complejidad de Dijkstra es de O(E*log(V)); esto es consistente con el tiempo de ejecución obtenido en las pruebas.

Requerimiento 2

Máquina	Consumo de Datos [kB]	Tiempo de Ejecución [ms]
1	1505.57	439.1
2	1552.31	388.548

Tabla 4. Comparación de consumo de datos y tiempo de ejecución del requerimiento 2 para ambas máquinas

Requerimiento 3 (Grupal)

Análisis de complejidad: La primera parte de este requerimiento utiliza el algoritmo de Prim, por lo que el requerimiento tendrá la misma complejidad temporal del algoritmo al utilizar la estructura de datos de lista de adyacencia. Esta complejidad temporal corresponde a O(E*log(V)). En cuanto a la complejidad espacial, esta es proporcional a V.

Requerimiento 3 / Parte 1: MST

Máquina	Consumo de Datos [kB]	Tiempo de Ejecución [ms]
1	2558.56	582
2	2645.27	493.3

Tabla 5. Comparación de consumo de datos y tiempo de ejecución del requerimiento 3 para ambas máquinas

Análisis de complejidad: La segunda parte, además de utilizar el algoritmo de Prim, hace 2 recorridos por los 1375 arcos del MST. En uno de estos recorridos, se realizan otros recorridos parciales internos para buscar la rama más larga. El tiempo de ejecución de esta parte aumenta debido al uso de la función lt.isPresent() que se utiliza en cada una de las iteraciones del ciclo parcial interno. Aun así, este ciclo interno no representa una carga muy grande en cuanto a tiempo, por lo que podemos decir que la complejidad temporal es de O(V). En cuanto a complejidad espacial, este algoritmo no requiere mayor espacio, pues crea arreglos de pocos elementos, por lo que se puede afirmar que la complejidad espacial es constante.

Requerimiento 3 / Parte 2: Búsqueda de la rama más larga

Máquina	Consumo de Datos [kB]	Tiempo de Ejecución [ms]
1	19.61	4082.3
2	22.35	3705.02

Tabla 5. Comparación de consumo de datos y tiempo de ejecución del requerimiento 3 para ambas máquinas