Tecnológico de Monterrey

Act 4.3 - Actividad integral de Grafos (Evidencia competencia)

Reflexión

Alan Paul García Rosales A01639631

Estefanía Pérez Yeo A01639270

Viernes, 26 de noviembre del 2021

Programación de estructuras de datos y algoritmos fundamentales

TC1031 Gpo. 12

Dr. Eduardo Arturo Rodríguez Tello

Tecnológico de Monterrey

Alan Paul García Rosales A01639631
Estefanía Pérez Yeo A01639270
TC1031 Dr. Eduardo Arturo Rodríguez Tello

Introducción

La situación a resolver para esta actividad es algo distinta, en comparación a las actividades anteriormente realizadas, en esta ocasión se nos entrega un archivo de texto que contiene más del doble de registros, pero para esta nueva entrega se solicita que se organicen mediante el uso de un grafo, el cual nos ayudará a mostrar las conexiones entre las ips y de este modo, intentar averiguar dónde se encuentra el boot master.

Respecto al algoritmo realizado

Al igual que la vez anterior, a mi parecer, la mejor manera de abordar la problemática era crear una clase donde pudiera guardar los registros, para luego poder usar algún algoritmo de ordenamiento que fuera lo suficientemente eficiente como para ordenar más de 100,000 registros en total, con un tiempo relativamente corto utilizando un archivo de texto que nos fue proporcionado.

Posterior a la realización de nuestra clase registro y generar un grafo dirigido que contuviera todos los registros de la bitácora que se nos entregó, procedimos a meter los datos de mayor a menor basándonos en el grado de las ips en un archivo de texto como nos fue solicitado, gracias al uso de la estructura de datos heap tree esto fue muy sencillo, bastó con hacer un top y un pop en un ciclo y de ese modo se comienza a llenar el archivo de texto.

Finalmente, se nos solicitó identificar las 5 IP´s con más conexiones, para esto creamos una clase de tipo IP, que me permitiera almacenar una ip acompañada de su grado, contabilizamos los grados de cada ip, y con ayuda de un heap tree, el cual compararía el número de conexiones. Para pasar esto a un archivo, bastó con hacer lo mismo que con el archivo principal, un top y posteriormente un pop, esto cinco veces para obtener las 5 IP con más conexiones.

Tecnológico de Monterrey

Alan Paul García Rosales A01639631
Estefanía Pérez Yeo A01639270
TC1031 Dr. Eduardo Arturo Rodríguez Tello

Respecto a la importancia y eficiencia del uso de Grafos

Subjetivamente, por el tipo de situación que nos fue encomendada para esta actividad integral, el uso de Grafos fue la mejor elección, ya que nos permite visualizar de cierto modo las conexiones de todas las IPs y ver cuáles cuentan con mayor número de conexiones.

Puesto que el archivo utilizado com bitácora dentro de la evidencia, es relativamente larga en cuestión de número de IPs, es importante encontrar y manejar de la forma más reducida posible los time complexities utilizados en las clases tanto en Grafos como en MaxHeap con ayuda de la clase IP; con esto al momento de ejecución el código correrá de la forma más óptima posible.

Complejidad de los métodos utilizados

A continuación se muestra la complejidad de los métodos necesarios:

CLASE	MÉTODO[1]	COMPLEJIDAD
Graph.cpp	loadGraphList( );	O(n^2)
	split( );	O(n)
	splitString( );	O(n)
	quickSort( )	o(nlogn)
	swap( );	O(1)
	binarySearch( );	O(n)
	partition( )	O(n)
	Graph( )	O(1)
	~Graph(	O(1)
	printGraph( )	O(n^2)

[1] Los métodos descritos en la tabla no contienen los argumentos necesarios, solo son para visualizar su complejidad.

IP.cpp	sobrecarga de operadores >, <, ==, !=, <=, >=	O(1)
	IP( )	O(1)
	getIP( )	O(1)
	getSimpleIP( )	O(1)
	getIpInt( )	O(1)
	getDegree( )	O(1)
	setDegree( )	O(1)
	addToDegree( )	O(1)
MaxHeap.h	MaxHeap( )	O(n)
	~MaxHeap( )	O(1)
	isEmpty( )	O(1)
	getSize( )	O(1)
	getCapacity( )	O(1)
	printMaxHeap( )	O(1)
	push( )	O(n)
	createDegreesBinnacle( )	O(n)
	connections( )	O(logn)
	heaptify( )	O(logn)
	heapSort( )	O(logn)
	moveDown( )	O(logn)
	pop( )	O(logn)
	top( )	O(logn)

Tecnológico de Monterrey

Alan Paul García Rosales A01639631
Estefanía Pérez Yeo A01639270
TC1031 Dr. Eduardo Arturo Rodríguez Tello

Conclusión

Al igual que en la actividad integral anterior, puedo decir que la programación brinda una gran variedad de alternativas para poder dar solución a problemas de la vida cotidiana, aunque en realidad al visualizar las conexiones de más de 105,280 registros resulta complejo este se logra de manera factible gracias a la programación.

Ahora, centrándonos más en la parte del código, respecto a los grafos, como ya comentamos en el párrafo anterior, nos son de mucha utilidad para poder visualizar las relaciones que hay entre distintos objetos dentro del mismo, además, obtener conclusiones de estas conexiones que pueden ser de mucha utilidad.

También consideramos que algoritmos como el Heap sort son de bastante utilidad, conociéndolos y sabiéndolos aplicar, podríamos ordenar cualquier registro con datos comparables como fechas, cantidades, costos, entre otras cosas, estos son aplicables a infinidad de ámbitos, simplemente haciendo la correcta sobrecarga de los operadores.

Para esta cuarta entrega, puedo añadir que, las estructuras de datos, como lo son los Grafos, nos facilitaron de manera sustancial el trabajo, a comparación de otras estructuras de datos, el objetivo de comparación de conexiones de IPs se pudo realizar con relativa facilidad y rapidez. Con esto se concluye que, el uso de las estructuras de datos que aprendimos este tercer periodo, nos facilita el trabajo, si es oportuno utilizarlas.

Se cierra comentando que la presente actividad integradora nos ayudó a darnos cuenta de la gran utilidad de las estructuras de datos que hemos estado aprendiendo a lo largo del tercer periodo y una vez más, ver que el alcance de la programación no tiene límites, los límites los ponemos nosotros.

Extra

Link a replit

Bibliografía[2]

GeeksforGeeks. (2021, septiembre 15). *HeapSort*. Recuperado 6 de noviembre de 2021, de https://www.geeksforgeeks.org/heap-sort/

[2] Las siguientes citas no solo nos fueron informativas, también fueron de ayuda a la hora de implementar código y de algunas de ellas se referenció código directamente y se adaptó a la evidencia.