Tecnológico de Monterrey

Act 3.4 - Actividad integral de BST (Evidencia competencia)

Reflexión

Alan Paul García Rosales A01639631

Viernes, 5 de noviembre del 2021

Programación de estructuras de datos y algoritmos fundamentales

TC1031 Gpo. 12

Dr. Eduardo Arturo Rodríguez Tello

Alan Paul García Rosales A01639631
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 una gran cantidad de registros, pero para esta nueva entrega se solicita que se organicen mediante la comparación de IP's además de identificar las 5 que más se repiten con ayuda de árboles.

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 16,000 registros en un tiempo relativamente corto comparando las IP´s proporcionadas.

Posterior a la realización de mi clase registro y generar un heap tree que contuviera todos los registros de la bitácora que se nos entregó, procedí a meter los datos de mayor a menor en un archivo de texto como nos fue solicitado, gracias al uso de la estructura de datos 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 accesos, para esto creé una clase de tipo IP, que me permitiera almacenar una ip acompañada de sus accesos, contabilicé los accesos de cada ip, y metí ambos datos (ip y accesos), en un arreglo de tipo IP, posteriormente metí dicho arreglo a un heap tree, el cual compararía el número de accesos. 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 más atacadas.

Tecnológico de Monterrey

Alan Paul García Rosales A01639631
TC1031 Dr. Eduardo Arturo Rodríguez Tello

Respecto a la importancia y eficiencia del uso de algoritmos de tipo BST

Desde mi punto de vista, por el tipo de situación que nos fue encomendada para esta actividad integral, el uso del algoritmo Heap fue la mejor elección, ya que hace una comparación, a mi parecer, rápida y eficiente, además de que haciendo la correcta sobrecarga de operadores, podemos hacerlo funcionar con cualquiera de las clases que creemos. Si no se hubiera usado este tipo de algoritmo, tendríamos que haber hecho la comparación de cada uno de los datos uno con otro, lo que haría un algoritmo demasiado lento e ineficiente, ahí radica la importancia del uso de este tipo de estructura de datos cada que sea oportuno.

Complejidad de los métodos utilizados

A continuación muestro la complejidad de los métodos que necesité:

CLASE	MÉTODO	COMPLEJIDAD
Bitacora.h	leerDatos();	O(n)
	sort();	O(n log n)
	crearBitacoraOrdenada();	O(n)
	ipsMayorAcceso();	O(n)
MaxHeap.h	push(T key);	O(n)
	top();	O(1)
	pop();	O(n)
Registro.h	sobrecarga de operadores >,<,==,!=	O(1)
IP.h	sobrecarga de operadores >,<,==,!=	O(1)

¿Cómo podríamos saber si una red está infectada?

Tecnológico de Monterrey

Alan Paul García Rosales A01639631
TC1031 Dr. Eduardo Arturo Rodríguez Tello

Después de hacer una investigación respecto a los tipos de ataques a redes que podemos encontrar, podría decir que, depende, sin embargo, suponiendo que se tratara en este caso de un ataque de tipo denegación de servicio (DoS), en el que el atacante se dispone a sobrecargar una red inundándola con información para volverla lenta, o en su defecto, hacer que deje de funcionar. Podríamos decir que una red se encuentra infectada si hay múltiples ataques a una misma dirección ip.

Ahora, también existe la posibilidad de que no se identifique un ataque y ese sea verdaderamente efectivo y nos prive del uso de la red o peor aún, nos roben información.

Conclusión

En conclusión, al igual que en la actividad integral anterior, puedo decir que la programación nos brinda una gran variedad de alternativas para poder dar solución a problemas de la vida cotidiana y lo que parece difícil, como ordenar más de 16,000 registros, puede resultar fácil gracias a la programación.

Ahora, centrándonos más en la parte del código, considero 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, son aplicables a infinidad de ámbitos, simplemente haciendo la correcta sobrecarga de los operadores.

Para esta tercera entrega, puedo añadir que, las estructuras de datos, como lo es el heap tree, me facilitaron de manera sustancial el trabajo, a comparación de otras estructuras de datos, el objetivo de comparación de IP’s se pudo realizar con relativa facilidad y rapidez. Con esto puedo concluir que, el uso de las estructuras de datos que aprendimos el segundo periodo, nos facilita el trabajo, si es oportuno utilizarlas.

Puedo cerrar diciendo que la presente actividad integradora me ayudó a darme cuenta de la gran utilidad de las estructuras de datos que aprendimos a lo largo del segundo

periodo y una vez más, darme cuenta que el alcance de la programación no tiene límites, los límites los ponemos nosotros.

Extra

[Link a replit]

Bibliografía

Aclaro, las siguientes citas no solo me fueron informativas, también me fueron de ayuda a la hora de implementar código y de algunas de ellas saqué código directamente y solamente lo adapté al mío.

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

R. (2021, 6 noviembre). *25 Tipos de ataques informáticos y cómo prevenirlos*. CIBERSEGURIDAD. Recuperado 6 de noviembre de 2021, de https://ciberseguridad.blog/25-tipos-de-ataques-informaticos-y-como-prevenirlos/