Tecnológico de Monterrey

REFLEXIÓN - ACT 5.2 INTEGRAL

DANTE HERNÁNDEZ RAMÍREZ - A01668070

Programación de estructuras de datos y algoritmos fundamentales

(Gpo 573)

4 de Febrero del 2026

Reflexión

La ciberseguridad y el análisis de redes requieren herramientas capaces de procesar grandes volúmenes de información de manera eficiente y estructurada. En este contexto, el uso de estructuras de datos adecuadas se vuelve fundamental para representar relaciones complejas entre distintos elementos, como ocurre con los accesos entre direcciones IP dentro de un sistema. Una implementación incorrecta o poco eficiente puede dificultar la detección de patrones relevantes o generar resultados poco confiables.

En la presente actividad, el sistema desarrollado tiene como objetivo analizar los accesos registrados entre direcciones IPv4 mediante el uso de un grafo dirigido y ponderado, permitiendo identificar relaciones entre nodos y obtener resúmenes de accesos a partir de una IP específica. La solución se basa en la correcta lectura de los registros, su almacenamiento estructurado y el uso de recorridos controlados para extraer información útil del sistema.

De manera general, el funcionamiento del programa se apoya en los siguientes pasos principales:

- Lectura de los registros de entrada.
- Construcción del grafo dirigido con listas de adyacencia.
- Almacenamiento de las relaciones entre nodos mediante estructuras lineales.
- Recorrido de las listas de adyacencia para obtener información específica sobre los accesos.

ESTRUCTURAS DE DATOS UTILIZADAS

Para la representación del problema se seleccionó el uso de un grafo dirigido, implementado mediante un arreglo de listas ligadas, lo cual permite manejar eficientemente las relaciones entre nodos sin desperdiciar memoria en matrices densas. Cada nodo del grafo representa una dirección IP, mientras que las aristas indican accesos entre dichas direcciones.

Se emplearon listas ligadas para almacenar las relaciones de adyacencia, ya que permiten una inserción dinámica de elementos y un recorrido secuencial adecuado para el análisis

solicitado. El uso de una estructura de tipo Stack (pila) complementa el diseño al facilitar recorridos controlados del grafo, manteniendo un manejo ordenado de los nodos visitados.

La correcta encapsulación de las estructuras, como el uso de nodos privados y métodos de acceso (getters), garantiza la integridad de los datos y evita manipulaciones indebidas desde otras partes del programa, reforzando las buenas prácticas de diseño orientado a objetos.

ANÁLISIS DE COMPLEJIDAD

La solución implementada puede considerarse eficiente para el tipo de problema abordado. Las principales operaciones del sistema presentan las siguientes complejidades temporales:

- Lectura de los registros de entrada: O(n)
- Construcción del grafo (inserción de aristas): O(n)
- Búsqueda del índice de una IP en el grafo: O(n)
- Recorrido de la lista de adyacencia de un nodo: O(k), donde k es el número de conexiones del nodo
- Obtención del resumen de accesos: O(k)

De manera general, la complejidad total del sistema puede aproximarse a O(n) para la mayoría de los casos prácticos, lo cual resulta adecuado para conjuntos de datos de tamaño mediano o grande.

CONCLUSIÓN

La implementación de un grafo dirigido con listas ligadas demostró ser una solución eficiente y flexible para el análisis de accesos entre direcciones IP. Asimismo, el manejo correcto de la encapsulación y el diseño modular del código contribuyen a la mantenibilidad y claridad del sistema.

REFERENCIAS

Cormen, T. H., Leiserson, C. E., Rivest, R. L., & Stein, C. (2022). Introducción a los algoritmos (4th ed.). MIT Press.

https://mitpress.mit.edu/9780262046305/introduction-to-algorithms/

GeeksforGeeks. (s. f.). Graph data structure.
https://www.geeksforgeeks.org/graph-data-structure-and-algorithms/

GeeksforGeeks. (s. f.). Adjacency list representation of graph.
https://www.geeksforgeeks.org/adjacency-list-representation-of-graph/