Instituto Tecnológico y de Estudios Superiores de Monterrey
Programación de estructuras de datos y algoritmos fundamentales (Gpo 13)
Profesor: Luis Humberto González Guerra
Nombre: Sofía Gutiérrez | Matrícula: A00827191
Act 4.3 - Actividad Integral de Grafos (Evidencia Competencia)

Un grafo es una estructura de datos no lineal que puede ser definida como un conjunto de nodos, que también son conocidos como vértices, y arcos, los cuales pueden como puentes o conexiones entre dos o más nodos, por lo que estas estructuras de datos representan relaciones n:m (de muchos a muchos).

Para esta evidencia, recibimos una bitácora en la que los primeros 13,370 datos a recibir son direcciones ip únicas – nodos; la primera línea del archivo .txt indica la cantidad de nodos seguido por la cantidad de arcos, estos debían ser calculados hasta después pero el formato de las direcciones ip era diferente, fue necesario quitarles el puerto y ya después almacenarlos en la lista de adyacencias.
Para todo el desarrollo de la evidencia, además del uso de grafos, utilizamos también los unorder maps, en el cual almacenamos las direcciones ip como string junto con su posición y outdegree. A decir verdad, el uso de este contenedor nos facilitó demasiado el trabajo, de haberlo hecho con struct o vectores, hubiésemos tenido que hacer el enlace externo entre cada uno de los elementos; con el unorder map, se tuvo la ventaja que al ingresar a cualquier dato, teníamos acceso directo a sus demás componentes.

Durante el desarrollo del programa, se tuvo muy en mente el cómo programar de la manera más eficiente posible, pues la cantidad de datos que tendríamos que analizar era muy grande por lo que el tiempo de ejecución era crucial. Lo que decidimos fue almacenar los datos únicos en el unorder map e inicializar sus outdegree en 0's; después, al momento de recibir los nodos con sus arcos, les quitamos el puente y se le aumentó el outdegree dependiendo de la ip que se tratase, y en ese mismo instante se revisaba si el outdegree registrado más recientemente era mayor que el anterior, de manera que ya se iba registrando el outdegree mayor para obtener el botmaster.

Como mencioné, el proceso que seguimos mi equipo y yo para desarrollar esta evidencia, es muy importante que al desarrollar un código, se tenga muy en mente la eficiencia del mismo. Gracias a que nosotros tomamos en cuenta eso, fue posible analizar más de 13 mil datos, crear sus arcos analizando más de 91 mil líneas de texto, obtener sus outdegrees y encontrar al botmaster en un poco más de medio segundo