TECNOLÓGICO DE MONTERREY®

TC2038 – ANÁLISIS Y DISEÑO DE ALGORITMOS AVANZADOS
GRUPO 603

Actividad Integradora 2

Eunice Santos Galindo – A00831991
Brenda Elena Saucedo González – A00829855
Edgar Alexandro Castillo Palacios – A00830568

3 de diciembre de 2022

Introducción

Durante el año 2020 todo el mundo se vio afectado por un evento que nadie esperaba: la pandemia ocasionada por el COVID-19. En todos los países del planeta se tomaron medidas sanitarias para intentar contener la pandemia. Una de estas medidas fue el mandar a toda la población a sus casas, moviendo gran parte de las actividades presenciales a un modelo remoto en el que las empresas proveedoras de servicios de Internet (ISP, por sus siglas en inglés de Internet Service Provider) tomaron un papel más que protagónico. Mucha gente se movió a la modalidad de trabajo remoto, o home-office, también la mayoría de instituciones educativas optaron por continuar sus operaciones bajo un modelo a distancia aumentando de gran forma la transmisión de datos en Internet.

El ISP es la empresa que brinda conexión a Internet a sus clientes. Un ISP conecta a sus usuarios a Internet a través de diferentes tecnologías como ADSL, cablemódem, GSM, dial-up, fibra óptica, satélite, streaming, etc.

La presente situación problema consiste en solucionar tres subproblemas, en los cuales tenemos a nuestras manos, el poder mejorar los servicios de Internet en una población pequeña. Con base a esto, se nos asignó el encontrar la manera de cablear los puntos más importantes de dicha población de tal forma que se utilice la menor cantidad de fibra óptica, la forma óptima de visitar todos los puntos de la red y regresar al punto de origen, así como analizar la cantidad máxima de información que puede pasar desde un nodo a otro.

En lo que respecta, nuestro programa presenta diversos algoritmos empleados para la solución de la situación problema planteada, los cuales explicaremos a continuación.

Algoritmos Empleados

Kruskal

La principal problemática está enfocado a las empresas proveedoras de servicios de Internet (ISP), en donde en uno de los subproblemas se planteó que en ocasiones, se cuenta con una nueva vecindad o población a la cual se le desea trazar una ruta óptima de cableado usando la menor cantidad de material posible, sin mencionar que en algunos caminos se deberá de hacer uso de más material que otros, por ser más largos.

Observando la problemática desde una perspectiva más analítica, podemos hacer uso de un grafo para representar las colonias conectadas (vértices) por los caminos (arcos).

Para dar solución a dicho problema, podemos hacer uso de un árbol recubridor para este grafo, el cual sería un subconjunto de estos caminos que no

tenga ciclos pero que mantenga conectadas todas las colonias. Por supuesto puede haber más de un árbol recubridor posible, sin embargo, el árbol recubridor mínimo será el de menos material utilizado.

El algoritmo de Kruskal es un algoritmo de la teoría de grafos para encontrar un árbol recubridor mínimo en un grafo conexo y ponderado. Es decir, busca un subconjunto de aristas que, formando un árbol, incluyen todos los vértices y donde el valor de la suma de todas las aristas del árbol es el mínimo.

La implementación de nuestro algoritmo toma como parámetro una estructura que definimos para la lectura y comprensión del grafo, la cual procesa y analiza los datos para buscar y encontrar la ruta más óptima, en donde se hace uso de otra estructura que almacena los nodos padres y su respectiva clasificación para hacer las debidas comparaciones y evitar hacer de una ruta un ciclo.

Por el tipo de algoritmo y su implementación, su complejidad es de O(E log E) o también podría ser de O(E log V), en donde “E” depende de la cantidad de arcos o conexiones que existen en el grafo y “V” depende de la cantidad de vértices.

Traveling Salesman Problem

Teniendo en cuenta que la empresa apenas está entrando en el mundo tecnológico y aun hacen uso de estados de cuenta físicos así como publicidad, avisos y notificaciones impresas, es necesario calcular una ruta para que un trabajador visite todas las colonias exactamente una vez y vuelva a la colonia de origen al final, tratando de recorrer la menor distancia posible.

La primera parte del problema podría sonar como un Ciclo Hamiltoniano, ya que este recorre todos los vértices de un grafo exactamente una vez y vuelve al nodo de origen al terminar.

La diferencia entre el Ciclo Hamiltoniano y lo que se nos pide, es que en esta situación se tiene que tomar en cuenta el costo que toma cambiar de un nodo a otro. Es por esto que se necesita tomar en cuenta el Problema del Agente Viajero (Traveling Salesman Problem) el cual regresa el Ciclo Hamiltoniano que tiene el menor costo posible.

Existen varias maneras de resolver este problema. Si tomamos en cuenta una manera Naive o a fuerza bruta, podríamos calcular el costo para todas las posibles combinaciones de rutas en un grafo y así ir comparando costos para siempre guardar el mínimo junto con su ruta. Esto podría resultar con una complejidad factorial de O(n!) lo cual solo es recomendado en pequeños casos ya que en casos donde se tenga un gran número de datos a procesar no es muy eficiente.

Debido a esto existen muchas maneras y algoritmos los cuales pueden llegar a reducir la complejidad del problema. Un ejemplo de esto es el algoritmo de

Bellman-Held-Karp que puede reducir la complejidad hasta $O(n^2 \times 2^n)$ resolviendo el problema con recursividad, lo cual lo hace un algoritmo de programación dinámica.

Algoritmo de Ford Fulkerson

Al estar trabajando con ciudades con una gran cantidad de campos electromagnéticos, se pueden generar interferencias. La empresa quiere conocer el flujo máximo de información del nodo inicial al nodo final, para esto se hicieron estimaciones sobre el flujo de un punto a otro representadas en una matriz[i][j] donde cada posición indica la capacidad máxima de flujo entre la colonia i y la colonia j.

El flujo máximo del grafo debe ser el mismo de entrada que de salida, pero se pueden tomar diferentes caminos y estrategias durante.

En términos de grafos, la solución no optimizada sería simplemente buscar un camino posible desde el nodo inicial hasta el final, donde en cada arista, el flujo siempre sea menor a la capacidad de la misma, pero esta es una solución parcial que no siempre dará la respuesta correcta, puesto que no se aprovecha al máximo la capacidad de todas las aristas, al verse limitadas por el máximo de la arista que se recorre previamente.

La solución eficiente viene de plantear el uso de grafos residuales, esto es cuando la capacidad de una arista no ha sido utilizada de manera eficiente y aún es posible pasar flujo a través de ella (de ahí el nombre, residuales). De manera que el algoritmo va aumentando progresivamente el valor del flujo máximo cada vez que encuentra otro camino posible por el que puede pasar flujo desde el inicio hasta el final, utilizando la capacidad residual que queda en el grafo cada vez que se usan las aristas como parte de un camino posible. Para hacer el recorrido se hace un recorrido con técnica de amplitud, por lo que la implementación requiere el uso de filas (queues), además se necesita guardar un registro del camino recorrido para hacer la actualización de las capacidades residuales cuando se encuentre un nuevo flujo para agregar al máximo almacenado.

Finalmente, la complejidad del algoritmo planteado es de O(flujoMaximo * numAristas), pero por la implementación de recorrido estilo amplitud, la complejidad es de $O(EV^3)$, ya que los caminos que encuentra siempre tienen el mínimo posible de aristas. Una posible mejora es representar el grafo con lista de adyacencias, ya que en matriz la complejidad de BFS es de $O(V^2)$, pero con esa mejor podría reducirse el tiempo total de Ford Fulkerson a $O(VE^2)$, donde V = vértices y E = aristas. Aunque se optó por dejar la representación proporcionada en esta actividad, debido a que en cualquier representación, sigue siendo óptimo usar este algoritmo.

Conclusión

En conclusión, comprendemos que existe una gran diversidad de algoritmos que manejan y manipulan estructuras de datos como lo son los grafos dirigidos y no dirigidos, de los cuales algunos podrían ser más eficientes que otros, sin embargo, el desarrollo e implementación de los algoritmos de los que como equipo hicimos uso, la comprensión de su lógica fue adquirida de otro tipo de algoritmos similares que hemos estado analizando en el curso actual, los cuales nos ayudaron a figurar el desarrollo de esta misma, de manera que realizamos adaptaciones para la presente situación problema.

Somos conscientes de que estos tipos de algoritmos pueden ser aplicados en una gran diversidad de contextos, no sólo para encontrar la distancia mínima para cablear una población, sino también se puede aplicar para encontrar la ruta más eficiente para conectar los nodos para lo que se necesite, como el obtener la ruta conectada que contiene el menor precio posible, así como obtener la ruta para poder visitar todos los nodos y obtener la cantidad máxima que puede ser transportado en ciertos caminos, etc.

Referencias

Anónimo. (s.f.). *Situación problema 2*. Noviembre 19, 2022, de ITESM. Sitio web: https://experiencia21.tec.mx/courses/313041/pages/situacion-problema-2

Colaboradores de Wikipedia. (s.f.). *Proveedor de servicios de internet.* Noviembre 19, 2022, de ITESM. Sitio web: https://es.wikipedia.org/wiki/Proveedor_de_servicios_de_internet

Colaboradores de Wikipedia. (s.f.). Kruskal's Minimum Spanning Tree Algorithm | Greedy Algo-2. Noviembre 19, 2022, de ITESM. Sitio web: https://www.geeksforgeeks.org/kruskals-minimum-spanning-tree-algorithm-greedy-algo-2/

Colaboradores de Wikipedia. (s.f.). Algoritmo de Kruskal. Noviembre 20, 2022, de ITESM. Sitio web: https://es.wikipedia.org/wiki/Algoritmo_de_Kruskal

Colaboradores de Wikipedia. (s.f.). Árbol recubridor mínimo. Noviembre 20, 2022, de ITESM. Sitio web: https://es.wikipedia.org/wiki/%C3%81rbol_recubridor_m%C3%ADnimo

Chumbley A., Moore K., Hor T., Khim J., Ross E. (s.f). Ford-Fulkerson Algorithm. Brilliant. Sitio web: https://brilliant.org/wiki/ford-fulkerson-algorithm/

Travelling Salesman Problem C++ With Code Examples. (s. f.). https://www.folkstalk.com/tech/travelling-salesman-problem-c-with-code-examples/

Nhat Nguyen, Q. (2020, 10 mayo). Travelling Salesman Problem and Bellman-Held-Karp Algorithm. Graduate School of Mathematics, Nagoya University. http://www.math.nagoya-u.ac.jp/~richard/teaching/s2020/Quang1.pdf