TC2038.652

E2. Actividad Integradora 2

Carla Oñate Gardella	A01653555
Manuel Camacho Padilla	A01423135
Octavio Augusto Alemán Esparza	A01660702

Profesor:

José Daniel Azofeifa Ugalde

Introducción

En este documento se encuentra la explicación y una breve conclusión sobre cómo se resolvieron los problemas presentados en esta evidencia 2. Se explicará un poco sobre los algoritmos que se usaron para resolver los 4 problemas. Al final se encuentra una conclusión de cada integrante del equipo donde se explica lo aprendido al hacer esta evidencia.

Ejercicio 1 - Algoritmo de Dijkstra

Complejidad: $O\left(n^2\right)$

El algoritmo de Dijkstra es un algoritmo utilizado para buscar las rutas mínimas en un gráfico con o sin clasificación, cíclico y con pesos no negativos en los arcos. Fue inventado en 1956 por el científico informático holandés Edsger Dijkstra, quien más tarde lo publicó en 1959. Este algoritmo encuentra aplicación en múltiples contextos como la optimización en la construcción de redes, como fue el caso de esta entrega, donde se pidió crear centrales de red para poder subastar a diferentes ciudades.

El funcionamiento del algoritmo de Dijkstra es un gráfico con *n* vértices marcados por enteros {1, 2,. , N}, y que uno de estos nodos es el nodo inicial y el nodo de destino. El peso en el arco que une los nodos j y k se indica con p (j, k). Al final del análisis, se deben adjuntar dos etiquetas a cada nodo, f (i) que indica el peso total de la ruta(la suma de los pesos en las rutas de arco para llegar al nodo i - ésimo) y J (i) que indica el nodo que precede a i en la ruta mínima. Se definen dos conjuntos S y T, que contienen respectivamente los nodos a los que las etiquetas ya se han asignado y los que aún no se han escaneado.

Ejercicio 2 - Algoritmo del Viajero

Complejidad: $O\left(n^2\right)$

Calcular la ruta más eficiente entre varias ciudades es un gran desafío. Hay cientos de opciones posibles que cambian en función de cientos de variables diferentes. Y es increíblemente costoso para cualquier negocio de entrega, servicio o camiones. Para resolver el problema del vendedor ambulante o algoritmo del viajero, se necesitan algoritmos robustos y un poco de poder computacional. Si no desea invertir dinero en un equipo interno de matemáticos e ingenieros expertos, necesita una solución de terceros. Por este algoritmo es importante de conocer como

ingenieros del ahora de tecnologías de la computación. Ya que, una vez que se comienza a planificar rutas en todo un mapa, alcanza un nivel de complejidad casi impensable. Con infinitas permutaciones de rutas potenciales entre diferentes paradas (típicamente llamadas nodos), identificar la ruta más corta y eficiente es un gran desafío. Es por eso que se realizarán varias permutaciones entre las combinaciones de ciudades, sin duda no es eficiente este método, y se tendría que trabajar en una mejor solución para este reto, pero en nuestro caso fue suficiente debido a la baja cantidad de datos, pero aun nivel mayor este algoritmo no sea factible, al menos como se desarrolló en este proyecto.

Ejercicio 3 - Algoritmo de Flujo Máximo

Complejidad: $O(E * f)$

En el tercer problema de la evidencia se pide obtener el flujo máximo de transmisión entre las colonias. Para hacer esto se usó un algoritmo para obtener el flujo máximo de la red el cual se llama Ford-Fulkerson. Este algoritmo lo que hace es buscar caminos del nodo inicial y el nodo final y encontrar cual es el flujo máximo dentro de las aristas de ese camino. Al obtenerlo lo resta de las capacidades actuales y crea un grafo residual el cual tiene la carga o flujo actual que puede pasar. Al hacer esto se hace un cálculo de cuántos caminos llegan del nodo inicial al nodo final y cuánta carga logra pasar tomando en cuenta la capacidad entre cada nodo.

Para hacer esto primero se usa el Breadth First Search para encontrar un camino del nodo inicial y nodo final, al mismo tiempo que se hace esto se consigue el flujo máximo que puede pasar por ese camino, el cual es el mínimo de todos los flujos encontrados. Al ya tener esto se actualiza el grafo residual restando y agregando el valor inicial en el flujo de ida y de regreso respectivamente. Ya que no se tengan más caminos con capacidad se regresa el total que se consiguió el cual es el flujo máximo de la red.

Ejercicio 4 - Geometría Computacional

Complejidad: $O\left(n^2\right)$

En el cuarto problema se nos solicitaba ubicar la central más cercana geográficamente a una serie de colonias en las que se busca hacer nuevas

contrataciones. Para ello, el programa recibe una serie de coordenadas de forma (X, Y), primeramente de las centrales y posteriormente de las ciudades.

Para dar solución a este problema, se realizó una adaptación a la fórmula de distancia Euclidiana, la cuál devuelve el valor de una distancia vectorial dada a partir de dos puntos en un plano. Dicha fórmula se puede expresar de la siguiente forma:

$$d_{P_1P_2} = \left|\left(\frac{x_2 - x_1}{y_2 - y_1}\right)\right| = \sqrt{\left(x_2 - x_1\right)^2 + \left(y_2 - y_1\right)^2}$$

Esta fórmula fue implementada mediante una función, la cual es llamada recibiendo como parámetros las coordenadas (X, Y) de una central así como las coordenadas (X, Y) de una colonia. Para encontrar la central más cercana a cada colonia, se hace uso de dos ciclos *for* anidados, el primero ciclando a través de un vector de coordenadas de las colonias y el segundo ciclando a través de un vector de coordenadas de las centrales. En este punto es importante mencionar que se asume que las coordenadas de entrada representan distancias en kilómetros dentro de un plano. El programa comienza asumiendo que la primera coincidencia entre una Colonia (n) y la Central (A) es la menor, de manera que guarda dichos valores en un par de variables. Si en algún momento se encuentra una central con una distancia menor, se reajustan los valores de dichas variables y se guarda su valor para efectos de comparación con las coordenadas restantes dentro del vector.

Al terminar cada ciclo tras haber comparado una colonia con las coordenadas de todas las centrales, se imprime la menor distancia encontrada. Por lo anterior, esta solución presenta una complejidad de $O\left(n^2\right)$, dado que se deben comparar todas las entradas una a una de un vector de colonias con todas las entradas una a una de un vector de centrales.

Cabe mencionar que el programa requiere de tener al menos una entrada en ambos vectores y, dado que no se indica un valor de distancia máxima, siempre se encontrará una coincidencia para cada colonia, así esta sea muy pequeña o muy grande. La única excepción es si se encuentra una colonia con coordenadas iguales a las de una central, en cuyo caso se indicará la coincidencia en consola.

Asimismo, se validan las entradas mediante un regex que checa el formato de coordenada (x,y).

Conclusión

- Carla Oñate: Durante este entregable pude aprender diferentes técnicas que resuelven problemas importantes que son más complejos a los vistos anteriormente. Durante este periodo hemos podido implementar soluciones a ejercicios que nos retan a tener un mejor entendimiento de las estructuras de datos que se pueden usar para resolverlos. Ya que estas pueden dar una ventaja sobre el tiempo de ejecución así como de la facilidad de resolver el problema. Más específicamente esta evidencia pone a prueba el entendimiento de los algoritmos vistos en clase en una situación realista la cual puede ser replicada en muchas otras situaciones. Al igual esta retador poder validar la información que recibe el sistema ya que en este problema todos los algoritmos usan las coordenadas de ciudades que no son diferentes entre uno y otro. En general todo lo que se hizo nos ayudó a profundizar lo aprendido durante estas 5 semanas de una manera retadora.

- Manuel Camacho:

  Este entregable ha sido mucho más fácil que el anterior, pero también más interesante por el uso de grafos y geometría computacional. Son temas que son de suma importancia en la industria, por ejemplo, en los mapas o servicios de entrega. Creo que es importante que como ingenieros podamos entender bien el concepto y las bases de estos algoritmos.

  Me siento orgulloso de mi equipo porque pudimos organizarnos bien y atender los temas de una manera en la que todos supiéramos lo que estamos haciendo. Además, cuando había dudas para resolver un problema, pudimos asistir a asesorías con el profesor y en manera grupal entender los detalles que nos hacían falta conocer. Por ende, puedo decir que ha sido un buen trabajo, y a su vez, muy interesantes conceptos.

  José Daniel Azofeifa Ugalde, profe, gracias por dejarnos hacer validaciones, ahora que lo pienso, nos está acostumbrando a siempre mantener la seguridad.

- Augusto Alemán:

  Puedo decir que esta evidencia ha servido no solo la manera de interconectar muchos de los temas vistos en clase, sino también para demostrar su valía y

aplicaciones en contextos de problemáticas reales. En varias ocasiones hemos visto temáticas y ejercicios sin duda muy retadores e interesantes, no obstante, viendo cada tema como un fragmento individual resulta bastante difícil aplicar dichos contenidos en problemáticas reales.

En este caso, al estar combinando varios de los temas aprendidos durante clase, así como algunos paradigmas de programación, pudimos crear una solución a la problemática planteada que realmente satisfaga todas sus implicaciones, pudiendo ser capaces además de llevar esta solución a un escenario real en un futuro en caso de ser necesario.