ESTRUCTURA DE DATOS 2
Código ST0247

Laboratorio Nro. 3
Vuelta atrás (Backtracking)

Simón E. Correa Henao
Universidad Eafit
Medellín, Colombia
scorreah@eafit.edu.co

David Gómez Correa
Universidad Eafit
Medellín, Colombia
dgomezc10@eafit.edu.co

3) Simulacro de preguntas de sustentación de Proyectos

3.1 Para resolver el problema del camino más corto en un grafo, existen otras técnicas computacionales como lo son: algoritmo Dijkstra (que trabaja con la idea de explorar todos los caminos más cortos que parten del vértice origen y que llevan a todos los demás vértices), y Algoritmo de búsqueda A* (algoritmo de búsqueda en amplitud que trabaja como el algoritmo Dijkstra).

3.2 En caso de que quisiéramos enumerar todos los caminos que hay en un grafo dirigido completo, ¡seria n!, siendo n la cantidad de nodos que tiene el grafo; esto puesto que cada uno de los nodos va a tener conexión con cualquier otro nodo del grafo, al ir recorriendo nodo por nodo la cantidad de caminos se va reduciendo en 1, ¡lo que nos lleva a la conclusión de que la cantidad de caminos que se pueden encontrar (no el más corto) puede ser de n!

Valor de N	Tiempo (Segundos)
4	0
5	0
6	0.010
7	0,040
8	0.092
9	0.35
10	0.724
11	0.020
12	0.11
13	0.621
14	3.9
15	26
16	181
17	2700

3.3

3.4 El algoritmo de DFS puede ser utilizado preferiblemente en problemas relacionados con encontrar el camino más corto de un grafo, puesto que se enfoca en recorrer el grafo sin

detenerse en un nodo en particular sino hasta llegar a un destino o haberlo recorrido todo. Por otro lado, el algoritmo BFS puede utilizarse en problemas relacionados con el coloreado de grafos, puesto que terminará recorriendo de forma ordenada los nodos adyacentes al nodo visitado.

3.5 Para resolver el problema 2.1 se utilizó la estructura conocida como Pair (Pareja) que puede almacenar parejas de cualquieras dos estructuras de datos. En el caso particular del problema 2.1 se utilizó un Pair para almacenar un int (número entero) que representa el costo del camino más corto, y un int[] (arreglo de enteros) que representa el camino más corto. Además de esta estructura, también se utilizó un boolean[] (arreglo de booleanos) del tamaño del grafo, que se encargaba de llevar el registro de los nodos visitados.

El programa recorre recursivamente todo el grafo nodo por nodo, buscando entre los sucesores del nodo que se encuentra visitando, revisando que el siguiente nodo a visitar no se encuentre entre los nodos anteriormente visitados. Recursivamente retorna el camino cuyo costo sea el mínimo, y se pone las condiciones de parada de: que, si ya llegó al nodo destino, entonces retorne el camino; y si el costo acumulado que lleva en ese recorrido es mayor al mínimo costo encontrado, entonces que retorne infinito (para que no se fije en ese camino).

3.6 La complejidad del ejercicio 2.1 es O(n!)

3.7 La variable 'n' representa la cantidad de nodos del grafo

3.8 Para el código del ejercicio 1.1 se utilizó un arreglo de booleanos, que sirvió para llevar registro de los nodos ya visitados.

En cuanto a cómo funciona el programa, básicamente recorre recursivamente todo el grafo, nodo por nodo, visitando todos los nodos sucesores al nodo actual, sin tener en cuenta aquellos ya visitados, es decir, sin repetir nodo. Esto se hace recursivamente buscando el subcamino o camino que genere el menor costo. Para conseguirlo se contó con 2 condiciones de parada que fueron: si ya llegó al nodo destino, entonces retorna el costo total de ese camino; y si ya sobrepasó el costo del camino más corto, entonces retorna infinito, para no seguir buscando por ese camino.

4) Simulacro de Parcial

4.1 *4.1.1 Solucionar(n-a,a,b,c)*

4.1.2 Math.max(res, Solucionar(n-b,a,b,c) +1)
4.1.3 Math.max(res, Solucionar(n-c,a,b,c) +1)

4.2 *4.2.1*
4.2.2 sePuede(v, graph, path, pos)
4.2.3 cicloHamilAux(graph, path, pos+1)

4.3 -

4.4 -

4.5 *4.5.1 Respuesta: 1*
4.5.2 Math.max(ni, nj)
4.5.3 T(n) = T(n-1) + T(n-1)

PhD. Mauricio Toro Bermúdez
Docente | Escuela de Ingeniería | Informática y Sistemas
Correo: mtorobe@eafit.edu.co | Oficina: Bloque 19 – 627
Tel: (+57) (4) 261 95 00 Ext. 9473

4.6 *-*
4.7 *4.7.1 (r == N)*
4.7.2 i
4.7.3 r+1

PhD. Mauricio Toro Bermúdez
Docente | Escuela de Ingeniería | Informática y Sistemas
Correo: mtorobe@eafit.edu.co | Oficina: Bloque 19 – 627
Tel: (+57) (4) 261 95 00 Ext. 9473