ESTRUCTURA DE DATOS 2
Código ST0247

Laboratorio Nro. 1
Implementación de Grafos

Tomas David Navarro Munera
Universidad Eafit
Medellín, Colombia
tdnavarrom@eafit.edu.co

Pablo Correa Morales
Universidad Eafit
Medellín, Colombia
pcorream2@eafit.edu.co

3) Simulacro de preguntas de sustentación de Proyectos

3.1 El programa funciona de la siguiente manera: Comenzamos leyendo el archivo, creando un arreglo de tipo cadena, al separar la linea que se esta leyendo por un espacio. Luego, se convierten los valores que hay en cada posición del arreglo a los deseados, para poder crear un objeto Vertice, agregarlo al hashmap y luego repetir el mismo procedimiento de lectura, pero creando un objeto Arco, que crea un linkedlist donde tiene almacenado las posiciones del Vertice en el hashmap.
3.2 Ocuparia un total de 11.25 GB. Si tenemos en cuenta que cada Vertice ocupa 1 byte de memoria.
3.3 Al asignar los valores en un hashmap, no tuvimos problema alguno con los identificadores.
3.4 El algoritmo que usamos para determinar si un grafo es bipartito o no, por medio de este poder coloreablo es basado en el algoritmo BFS (Breadth First Search)que se utiliza con regularidad a la hora de realizar busquedas en arboles o en grafos, la cual comienza desde la raiz del arbol o grafo y va a cada uno de los nodos adyacentes. La logica que concluimos a partir del algoritmo saber si un grafo es bipartito es que: Si un nodo adyacente a la raiz (o cualquier nodo arbitrario) es a su vez adyacente a otro nodo adyacente a la raiz el garfo dejara de ser bipartito. Ademas si es un grafo cerrado para que sea bipartito su numero de vertices debe ser par.Cabe resaltar que nos basamos del algoritmo recomendado por el docente de la pagina "GeeksforGeeks" https://www.geeksforgeeks.org/bipartite-graph/
3.5 Esta respuesta va de la mano con la del numeral anterior, puesto que en lo que se basa el BFS es que se para en cada nodo y revisa si tiene adyacencias, pero esto lo hace por niveles (recorrido por niveles), por lo que resulta que el orden del algoritmo no sera de O(V) sino de O(V^m).Puesto que tiene que recorrer primero el nivel entero antes de poder pasar a las adyacencias del siguiente nivel.

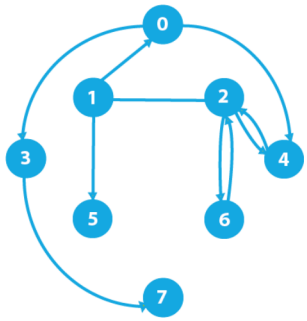
PhD. Mauricio Toro Bermúdez
Docente | Escuela de Ingeniería | Informática y Sistemas
Correo: mtorobe@eafit.edu.co | Oficina: Bloque 19 – 627
Tel: (+57) (4) 261 95 00 Ext. 9473

3.6

"v es el vertice en el que estamos parado"
"u es el vertice adyacente con el que estamos comparando v"
"colum es el numero de columnas"
"row es el numero de filas"
"V es el numero de vertices"
"E es el numero de arcos"
"m es el numero de niveles del grafo"

4) Simulacro de Parcial

4.1 (OPC)



	0	1	2	3	4	5	6	7
0				1	1			
1	1		1			1		
2		1			1		1	
3								1
4			1					
5								
6			1					
7								

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

4.2

0 ->[3,4]
1 ->[0,2,5]
2 ->[1,4,6]
3 ->[7]
4 ->[2]
5 ->N/A
6 -> [2]
7 -> N/A

4.3 ***O(n^2)***

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