Jan Marco Muñoz Pineda

Actividad de grafos

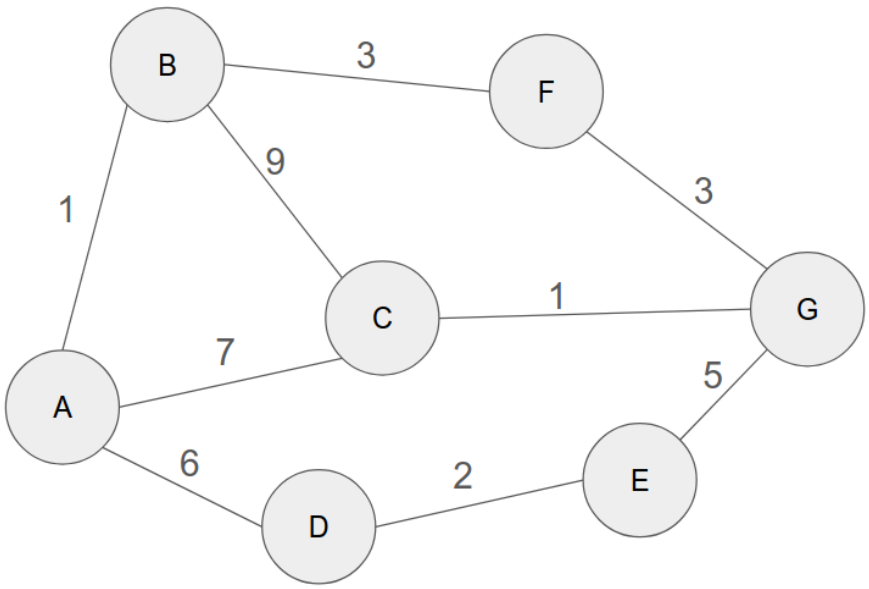
El algoritmo de Dijkstra consiste en encontrar el camino más corto desde un punto dado a otro.

El primer paso será definir la matriz de adyacencia:

	A	B	C	D	E	F	G
A		1	7	6			
B	1		9			3	
C	7	9					1
D	6				2		
E				2			5
F		3					3
G			1		5	3	

Matriz en la cual los espacios vacíos son equivalentes a 0.

Una vez definida la matriz de adyacencia se podrá implementar el algoritmo de Dijkstra, junto con la matriz de adyacencia se podra realizar una evaluación y comparar los caminos para hayar el camino más corto desde el nodo A hasta el nodo G.



Como se observa en el grafo se observan 5 caminos posibles para llegar desde el nodo A hasta el nodo G.

- A – B – F – G
- A – B – C – G
- A – C – B – F – G
- A – C – G
- A – D – E – G

Como se puede observar obtenemos que el camino más corto es A – C – G por lo que en la implementación nos debería dar este camino como salida del algoritmo.

Explicación código C++

Variables de la clase:

• m_Vertices: Es un vector que contiene los vértices del grafo, donde cada vértice tiene un índice que corresponde a su posición en el vector.

• m_Matrix: Es una matriz de adyacencia que utiliza un vector de vectores. m_Matrix[i][j] contiene el valor de la arista entre los vértices i y j. En este caso, un valor diferente de 0 indica que existe una conexión entre los vértices i y j, y ese valor puede representar el costo o peso de la arista.

2. Funcionamiento del método Levels:

• El método implementa una búsqueda en anchura (BFS) comenzando desde un nodo inicial (seed). Este enfoque es adecuado para explorar el grafo en niveles, lo que significa que primero se exploran todos los nodos a una distancia de un salto del nodo inicial, luego todos los nodos a una distancia de dos saltos, y así sucesivamente.

• Se utiliza una cola (queue<int> q) para gestionar el orden en que se visitan los nodos. Al comenzar, el nodo seed es el primer elemento en la cola.

• La variable marks es un vector booleano utilizado para marcar los nodos ya visitados, evitando ciclos y repeticiones de nodos en la lista de resultados.

• Cada vez que se procesa un nodo n (es decir, se saca de la cola), se verifica si ha sido visitado. Si no lo ha sido, se marca como visitado y se añade a res, que es la lista de resultados.

- Luego, para cada vecino i del nodo n, si existe una arista (m_Matrix[n][i] != 0) y el vecino aún no ha sido visitado, se añade a la cola para procesarlo en el siguiente nivel de BFS.

3. Retorno del método:

- Al final, el método devuelve res, que es una lista de vértices en el orden en que fueron visitados por el recorrido BFS, siguiendo los niveles desde el nodo seed.