

Gráficas

Emmanuel Cruz Hernández
emmanuel_cruzh@ciencias.unam.mx

10 de Agosto de 2021

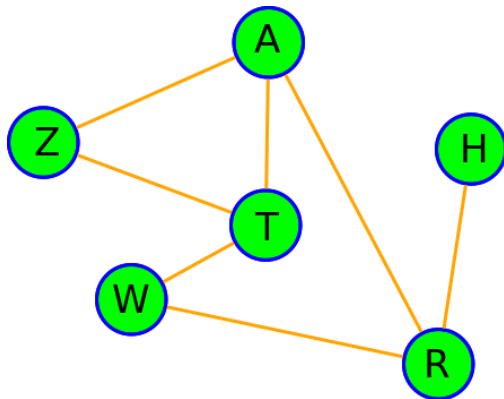
Contenido

- 1 Introducción
- 2 Matriz de adyacencias
- 3 Listas de adyacencias
- 4 Recorridos fundamentales
 - DFS
 - BFS
- 5 Bibliografía

Una gráfica es una forma de representar relaciones entre pares de objetos.

Visto de forma abstracta, una gráfica G es un conjunto V de vértices y una colección E de pares de vértices de V , llamadas aristas.

Representación gráfica



- `addEdge(A , B)`: agrega una arista entre el elemento A y el elemento B . Ocurre un error si alguno de los dos elementos no existe.
- `removeEdge(A , B)`: elimina una arista entre el elemento A y el elemento B . Ocurre un error si alguno de los dos elementos no existe.
- `setValue(A , B)`: reemplaza el valor del vértice que contiene a A por B . Ocurre un error si A no existe.
- `vertex()`: regresa una lista con los vértices de la gráfica.

Una gráfica se puede implementar con apoyo de otras estructuras.

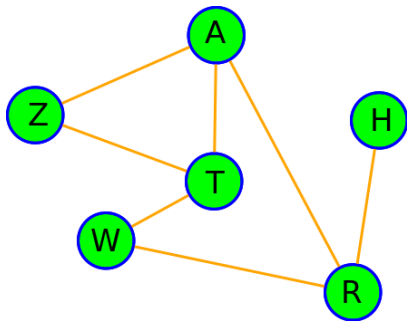
- Implementación con listas.
- Implementación con matrices.

Matriz de adyacencias

Una matriz de adyacencias es un arreglo bidimensional que almacena la representación de una gráfica.

Los índices i y j del arreglo representan los identificadores de los elementos de la gráfica, mientras que las entradas almacenan un identificador para saber si existe adyacencia.

Representación con matrices de adyacencias



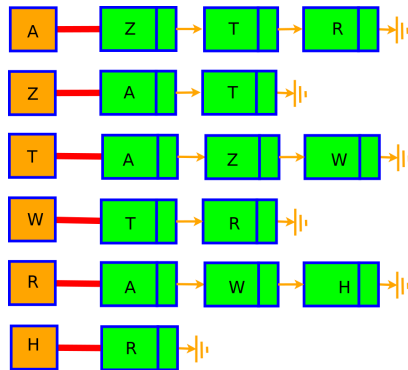
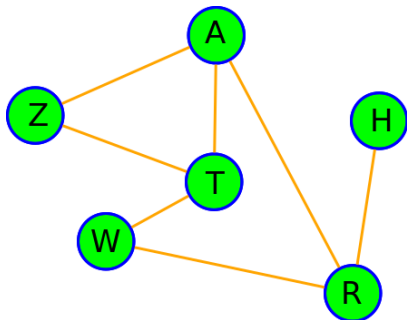
	A	Z	T	W	R	H
A	0	1	1	0	1	0
Z	1	0	1	0	0	0
T	1	1	0	1	0	0
W	0	0	1	0	1	0
R	1	0	0	1	0	1
H	0	0	0	0	1	0

Lista de adyacencias

Una lista de adyacencias es una representación de gráficas usando una lista como base.

A un vértice de la gráfica se le asocia una lista que almacena todos aquellos vértices que son adyacentes al vértice asociado a la lista.

Representación con listas de adyacencias



Usando listas de adyacencias

Es importante mencionar que no es necesario usar listas simplemente ligadas para la implementación de gráficas.

Hay dos algoritmos de recorrido que son fundamentales en las gráficas.

- DFS
- BFS

La sigla DFS significa *Depth-First Search*. También conocida como *búsqueda por profundidad*.

Es un algoritmo de búsqueda que recorre todos los vértices de una gráfica.

El algoritmo comienza fijando un vértice de la gráfica.

Si llamamos a nuestro vértice actual u , atravesamos G considerando una arista arbitraria (u, v) incidente al vértice actual u .

- Si la arista (u, v) nos lleva a un vértice v que ya está visitado, ignoramos esa arista.
- Si (u, v) conduce a un vértice v no visitado, nos movemos a v . Pintamos v como visitado y lo convertimos en el vértice actual, repitiendo el cálculo anterior recursivamente.

Pseudocódigo DFS

DFS(G, u):

Se marca u como visitado

for cada adyacencia de u , $e = (u, v)$ **do**

if v no ha sido visitado **then**

 Registre la arista e como la arista de descubrimiento

 Llamar recursivamente DFS(G, v)

end if

end for

La sigla BFS significa *Breadth-First Search*. También conocida como *búsqueda en anchura*.

Es un algoritmo de búsqueda que recorre todos los vértices de una gráfica.

El algoritmo comienza fijando un vértice de la gráfica.

Este recorrido se basa en recorrer los vértices por capas o niveles. En el nivel 0, sólo está el vértice inicial que está en el parámetro. En el nivel 1, se encuentran todos aquellos vértices que están a distancia 1 del vértice inicial y así sucesivamente.

Para esta implementación se usa una cola como estructura auxiliar.

Pseudocódigo BFS

BFS(G, u):

Cola Q

Marcar u como visitado

$Q.insert(u)$

while not $Q.isEmpty$ **do**

$v = Q.delete()$

for todo w en hijos de v **do**

if w no está visitado **then**




 Se marca como visitado w

$Q.insert(w)$

end if

end for

end while

-  GOODRICH, M.T., TAMASSIA, R. Y GOLDWASSER, M.H., *Data Structures and Algorithms in Java*, Wiley, Sexta Edición, 2014.
-  WEISS, M. A., *Data Structures and Algorithm Analysis in Java*, Pearson, Tercera Edición, 2012.
-  M. ZERÓN, C., *Notas de Estructuras de Datos 2018-1*