By: TatoNaranjo :)

Depth First Search

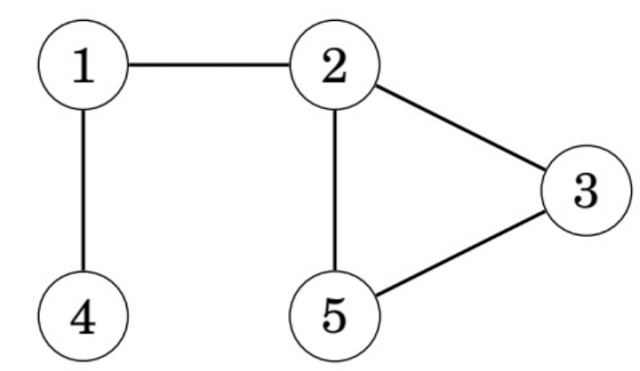
DFS es una técnica directa de recorrido de grafos. El algoritmo comienza en un nodo inicial, y procede a los otros nodos que son alcanzables desde el nodo inicial usando los vértices del grafo.

Depth First Search siempre sigue un camino simple en el grafo siempre y cuando encuentre nodos nuevos. Luego se devuelve a los nodos previos y comienza a explorar otras partes del grafo.

El algoritmo lleva un registro de los nodos visitados así que solo procesa el nodo una vez.

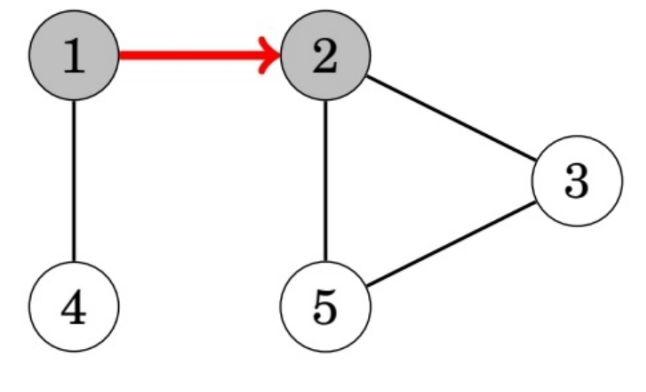
Ejemplo:

Veamos como el algoritmo de DFS procesa el siguiente grafo:

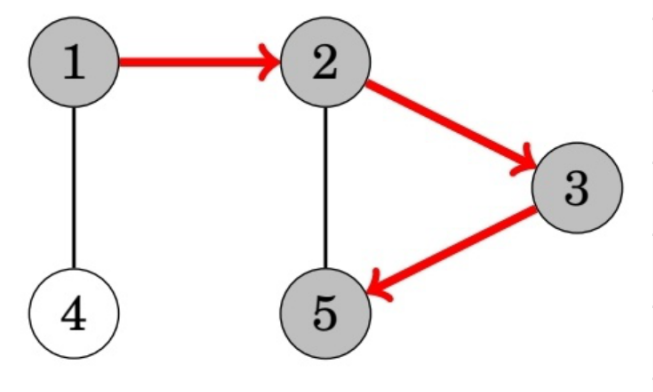


Podemos iniciar la búsqueda desde cualquier nodo del grafo, primero empezaremos la búsqueda desde el nodo 1.

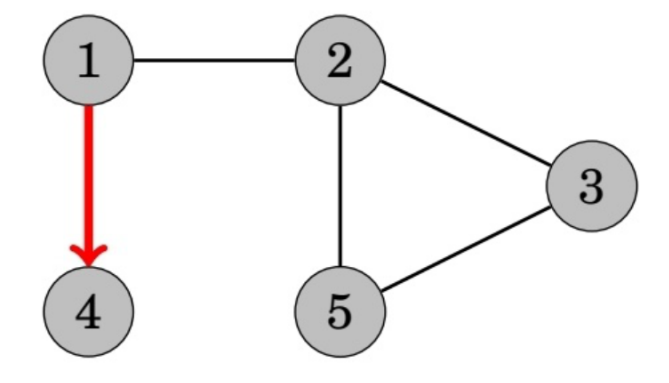
La búsqueda avanza primero al nodo 2:



Luego, se visitan los nodos 3 y 5:



Los vecinos del nodo 5 son 2 y 3, pero la búsqueda ya los ha visitado a ambos, por lo que nuestro siguiente movimiento será del nodo 1 al nodo 4.



Luego, la búsqueda termina ya que se han visitado todos los nodos.

La complejidad algorítmica de DFS es $O(n+m)$ en donde n es el numero de nodos y m es el número de vértices, porque el algoritmo procesa a cada nodo y vértice exactamente una vez.

Implementación

Depth First Search puede ser implementado de una forma conveniente usando recursividad. La siguiente función dfs comienza una Depth First Search en un nodo dado. La función asume que el grafo está almacenado en listas de adyacencia en un arreglo:

```
vector<int> adj[N];
```

y también maneja un arreglo:

```
bool visited[N];
```

que lleva el registro de todos los nodos visitados.

Inicialmente cada valor del arreglo es Falso y cuando la busqueda llega al nodo s, el valor de visited[s] se vuelve verdadero. La función se puede implementar de la siguiente forma:

```
void dfs(int s) {
    if (visited[s]) return;
    visited[s] = true;
    // process node s
    for (auto u: adj[s]) {
        dfs(u);
    }
}
```