Applications of DFS

Following are the problems that use DFS

Problems

4 Path Finding

5 To test if a graph is bipartite

6 Solving puzzles with only one solution

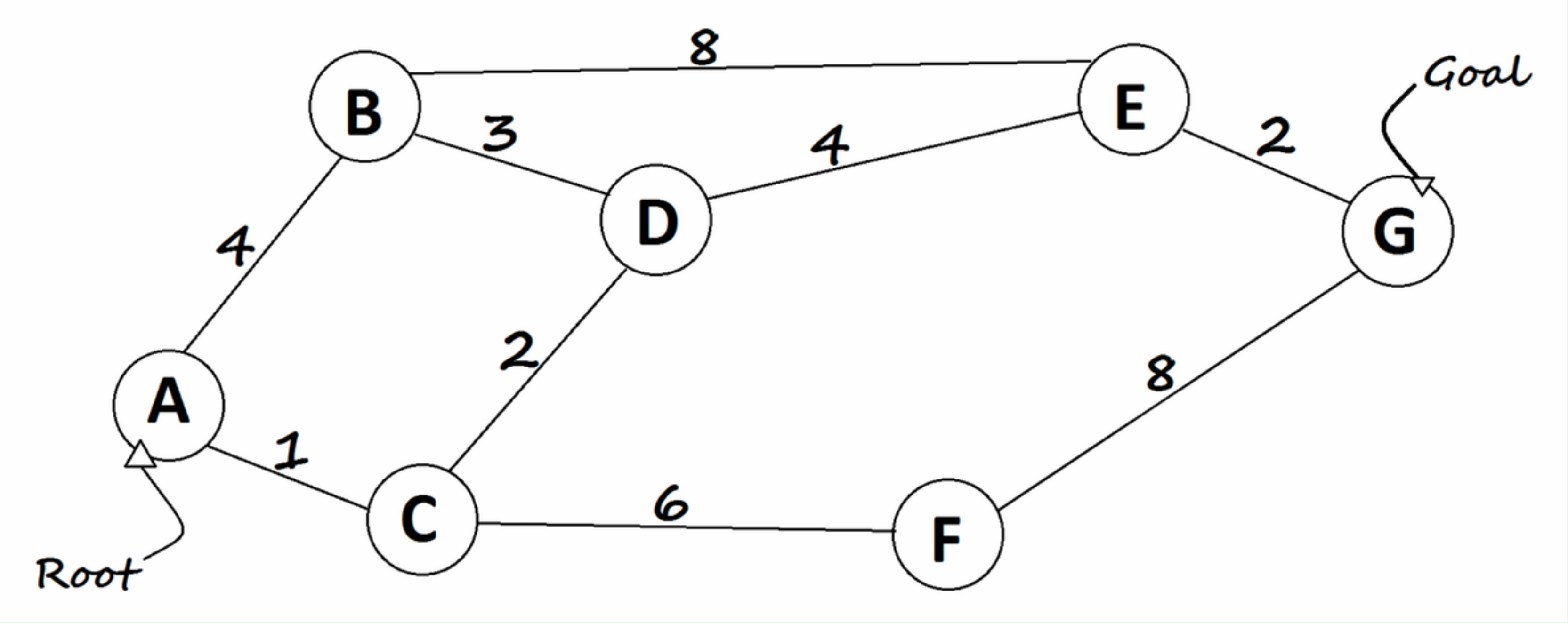
Let's Begin!

Are you ready?

STOP START KEEP GOING

Path Finding

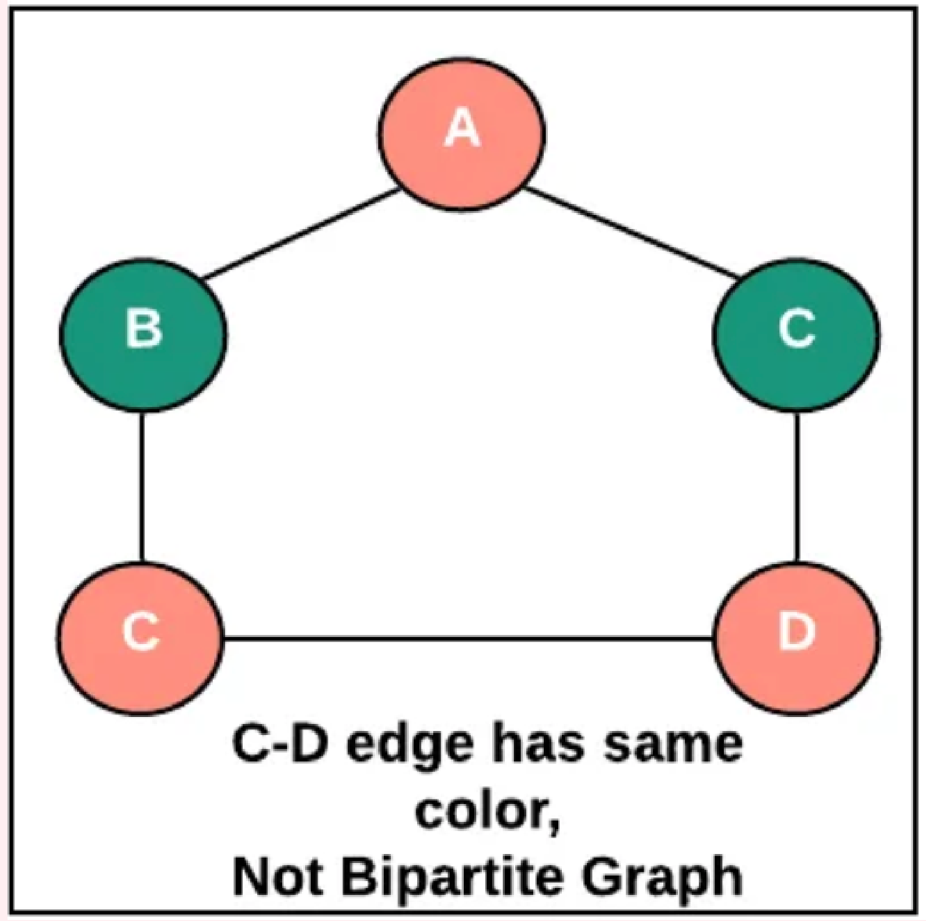
We can specialize the DFS algorithm to find a path between two given vertices u and z.
i) Call DFS(G, u) with u as the start vertex.
ii) Use a stack S to keep track of the path between the start vertex and the current vertex.
iii) As soon as destination vertex z is encountered, return the path as the contents of the stack



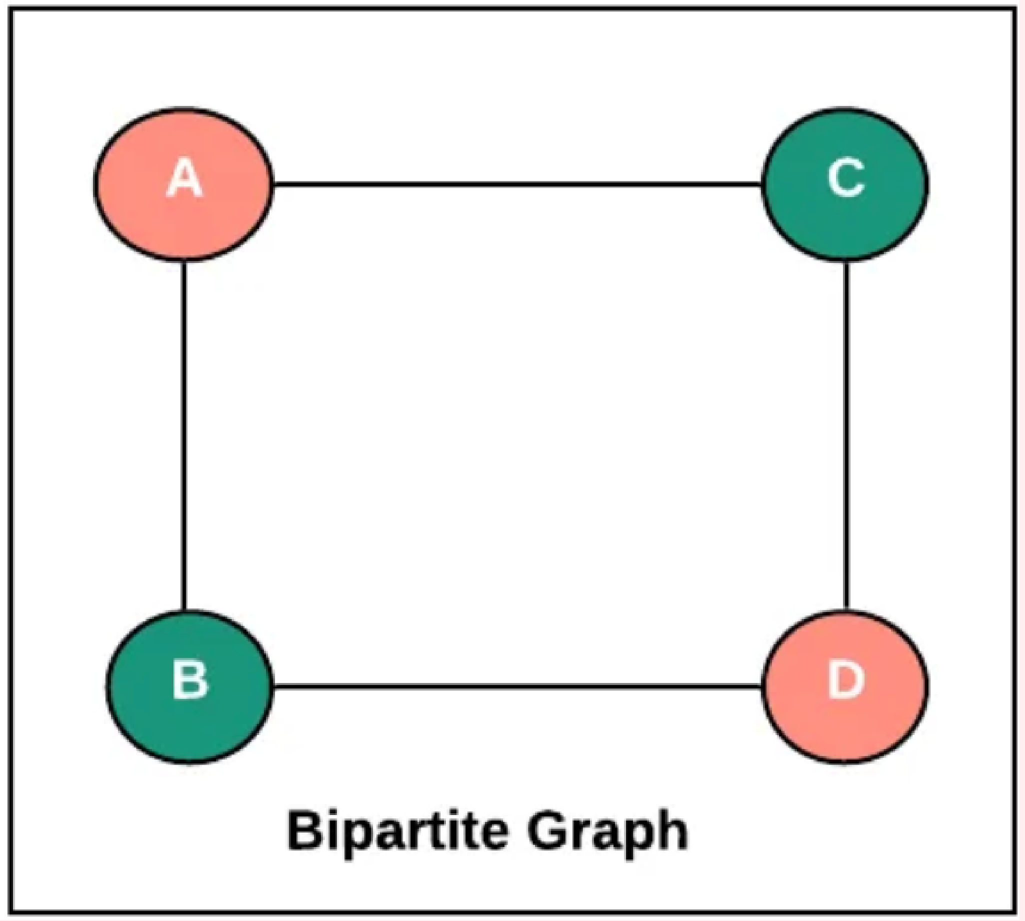
STOP START KEEP GOING

To test if a graph is bipartite

We can augment either BFS or DFS when we first discover a new vertex, color it oppposited its parents, and for each other edge, check it doesn’t link two vertices of the same color. The first vertex in any connected component can be red or black!



C-D edge has same color,
Not Bipartite Graph



Bipartite Graph

STOP START KEEP GOING

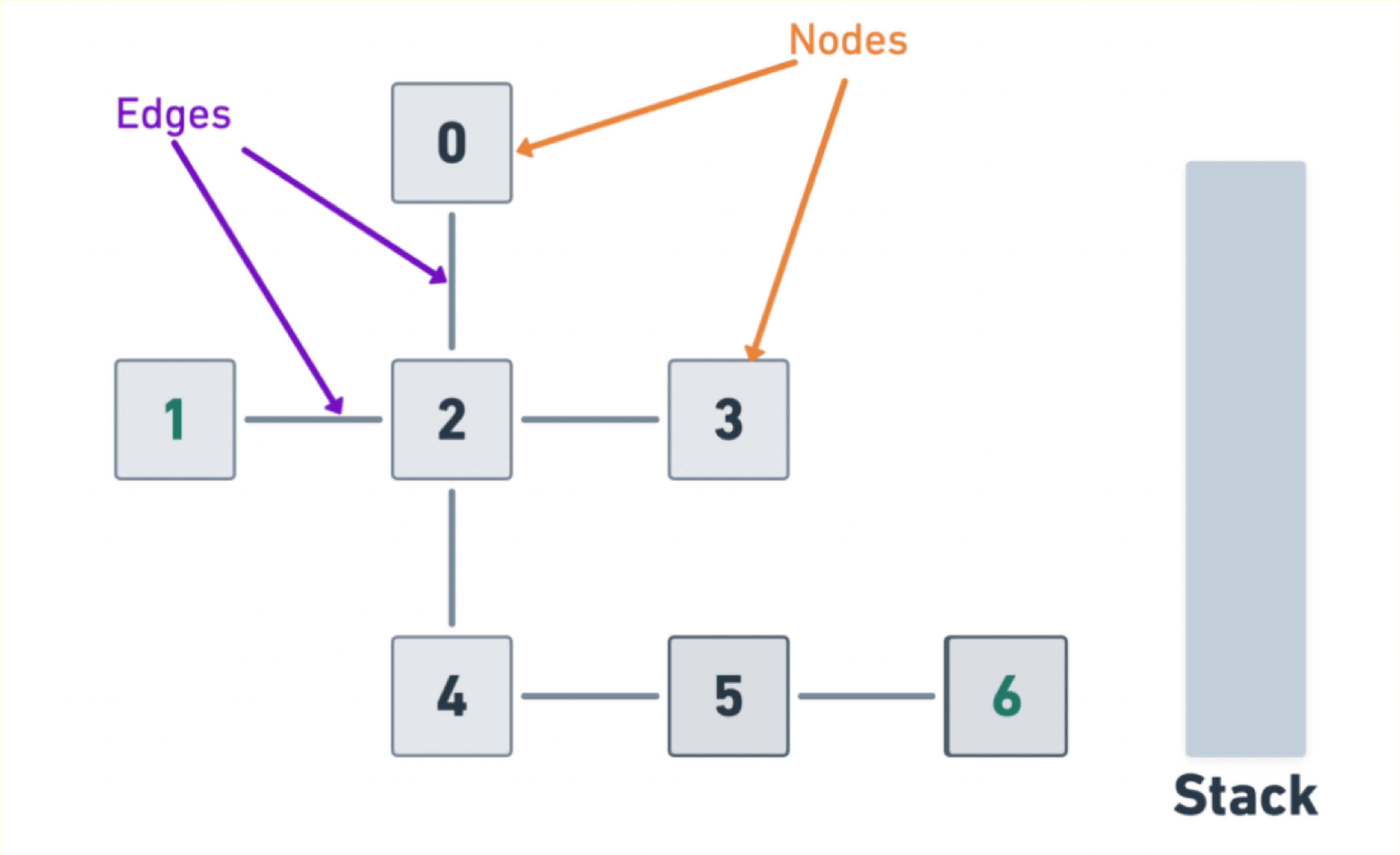
Solving puzzles with only one solution

In order to figure out how to traverse a maze through code, we first need to understand what depth-first search is. Depth-first search (sometimes referred to in this article as DFS) is a graph/tree traversal algorithm that follows a path as far as it can until it either, reaches the goal or has nowhere else to go. It’s almost like running as far as you can in one direction until you hit a wall. Hopefully that analogy helps clear up any lingering confusion. You might even be starting to see how we can use Depth-First Search to solve a maze!

Before we dive into the algorithm itself there are a few things we need to understand about depth-first search first. There are many approaches and styles to implementing a depth-first search algorithm and a lot of those implementation choices will entirely depend on the problem you’re trying to solve. So first let's define some assumptions. These assumptions may seem really straight forward but they’re really important in helping us come up with a solution to solve our maze.

STOP START KEEP GOING

CONTINUE



DFS CAN BE ADAPTED TO FIND ALL SOLUTIONS TO A MAZE BY ONLY INCLUDING NODES ON THE CURRENT PATH IN THE VISITED SET fancy group

Thank you

Have a great day ahead.