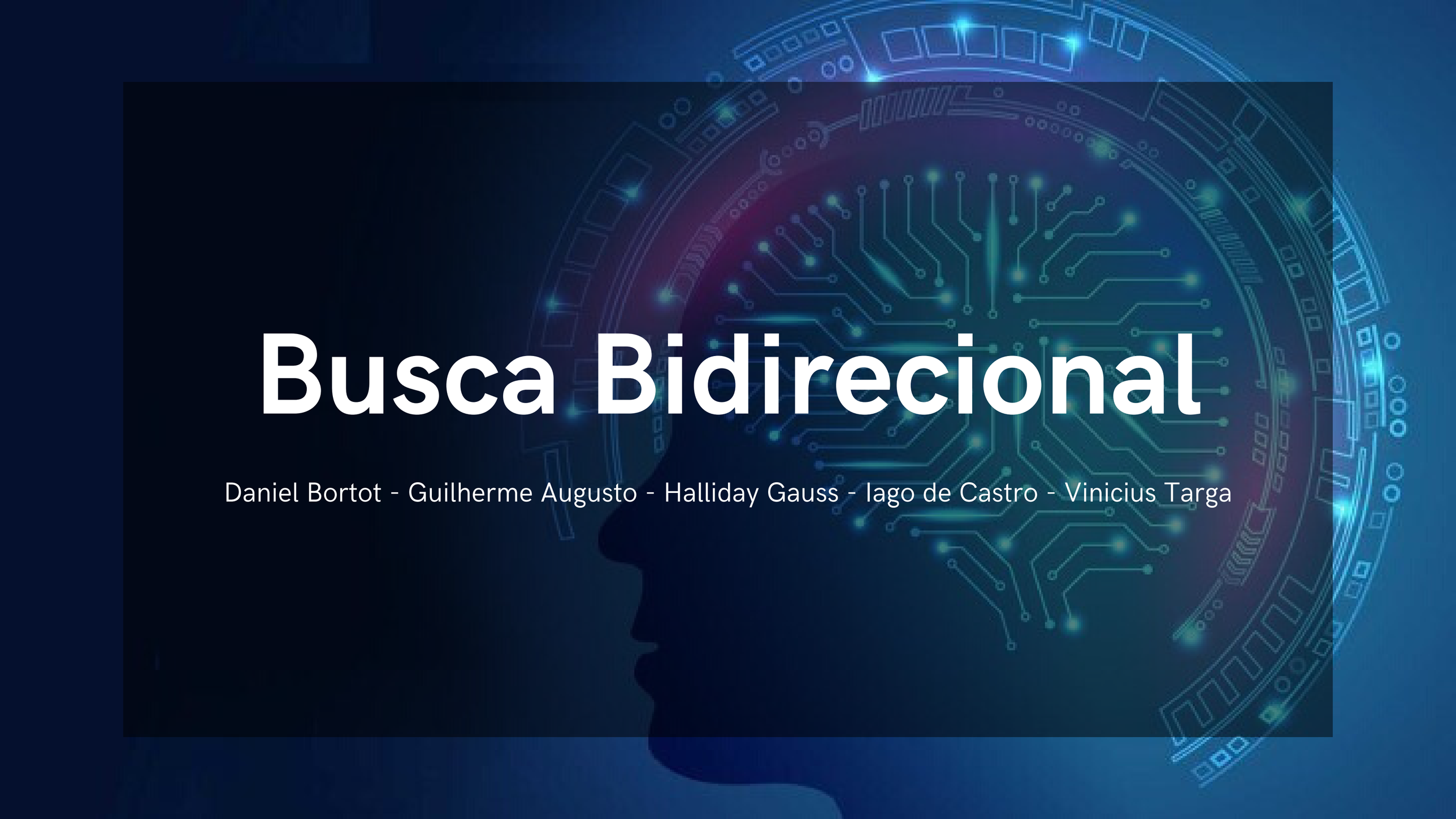


Busca Bidirecional

Daniel Bortot - Guilherme Augusto - Halliday Gauss - Iago de Castro - Vinicius Targa

Descrição

- A pesquisa Bidirecional é um algoritmo de busca em grafos que tem como objetivo encontrar o caminho mais curto entre um ponto inicial e um objetivo.

- Consiste em uma estratégia de busca cega, ou seja, é um tipo de busca que não possui nenhuma informação adicional sobre estados, além daquelas fornecidas na definição do problema.

Características

- A busca bidirecional só pode ser aplicada quando conhecemos o ponto de partida e o ponto final do problema a ser modelado.

- A partir dos dados fornecidos, é aplicado o algoritmo de busca em largura nos pontos de inicio e fim simultaneamente até que os dois se encontram em um determinado vértice e sendo assim encontrando o menor caminho entre esses pontos.

Características

Nem todo o problema é adequado para aplicar a busca bidirecional.

Exemplo:

- Caminho entre duas cidades. Sabe-se a cidade de partida e de chegada, deseja-se o caminho entre elas.
- O problema das n rainhas, onde cada casa do tabuleiro é um vértice, não existe vértice de chegada.

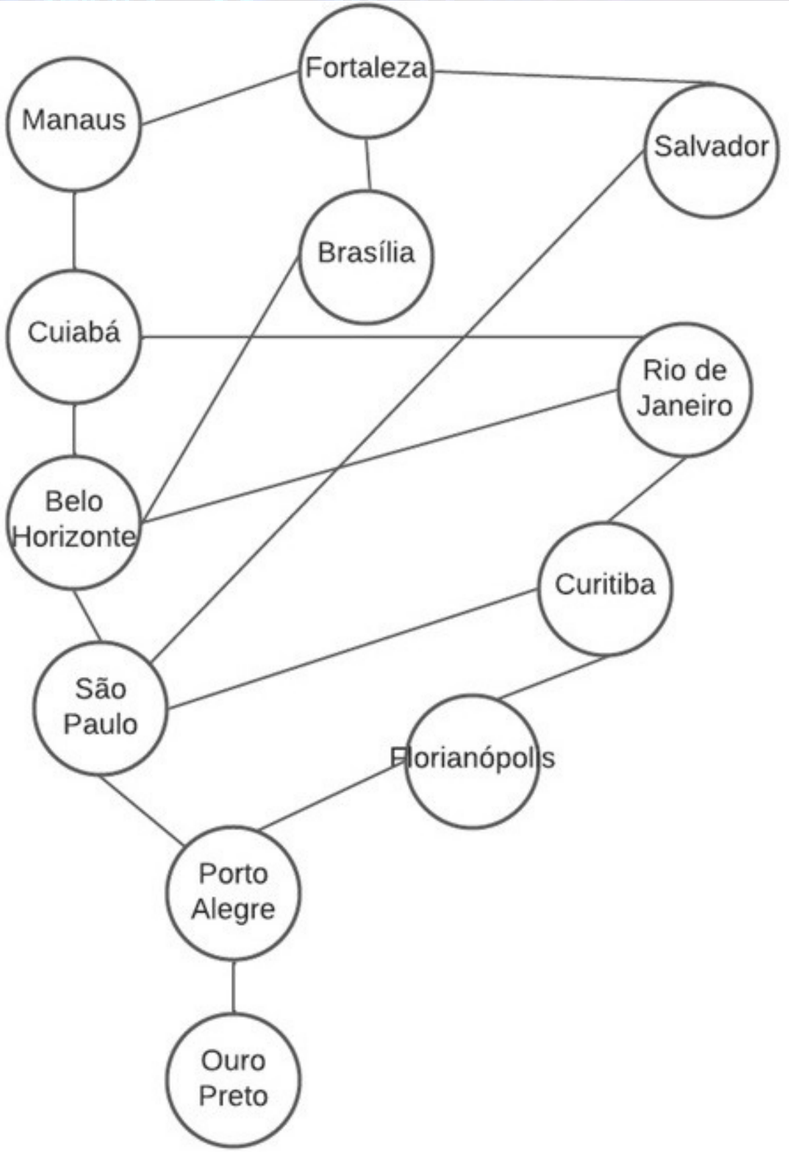
Algoritmo - Pseudocódigo

Entrada: Um grafo, nó inicial e nó final

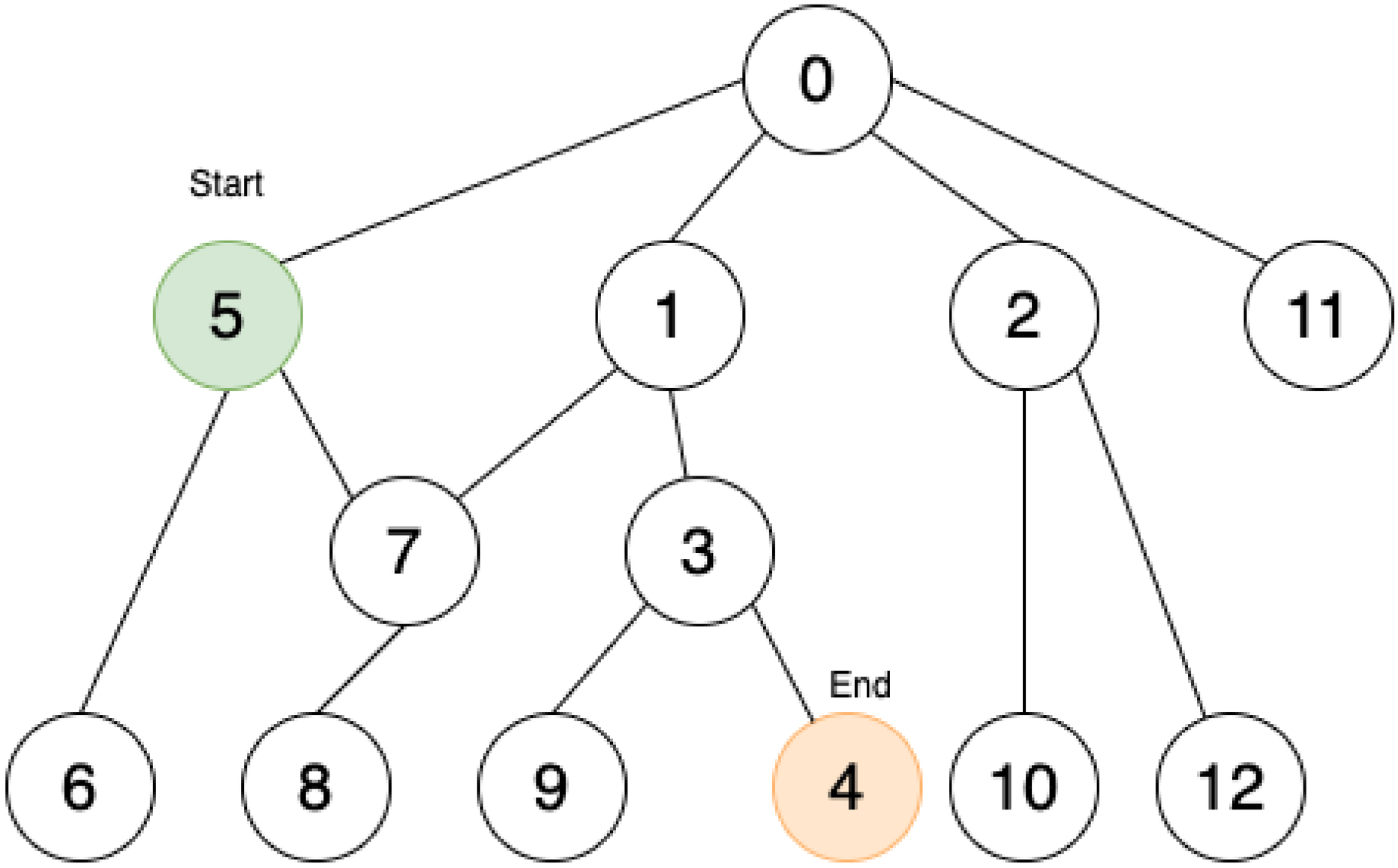
fronteira inicial = [nó inicial]
fronteira final = [no final]

Enquanto nenhuma das fronteiras for vazia:
    Busca em largura: fronteira inicial
    Busca em largura: fronteira final
        Se um nó repete em ambas as fronteiras:
            Retorna caminho encontrado
Fim Enquanto

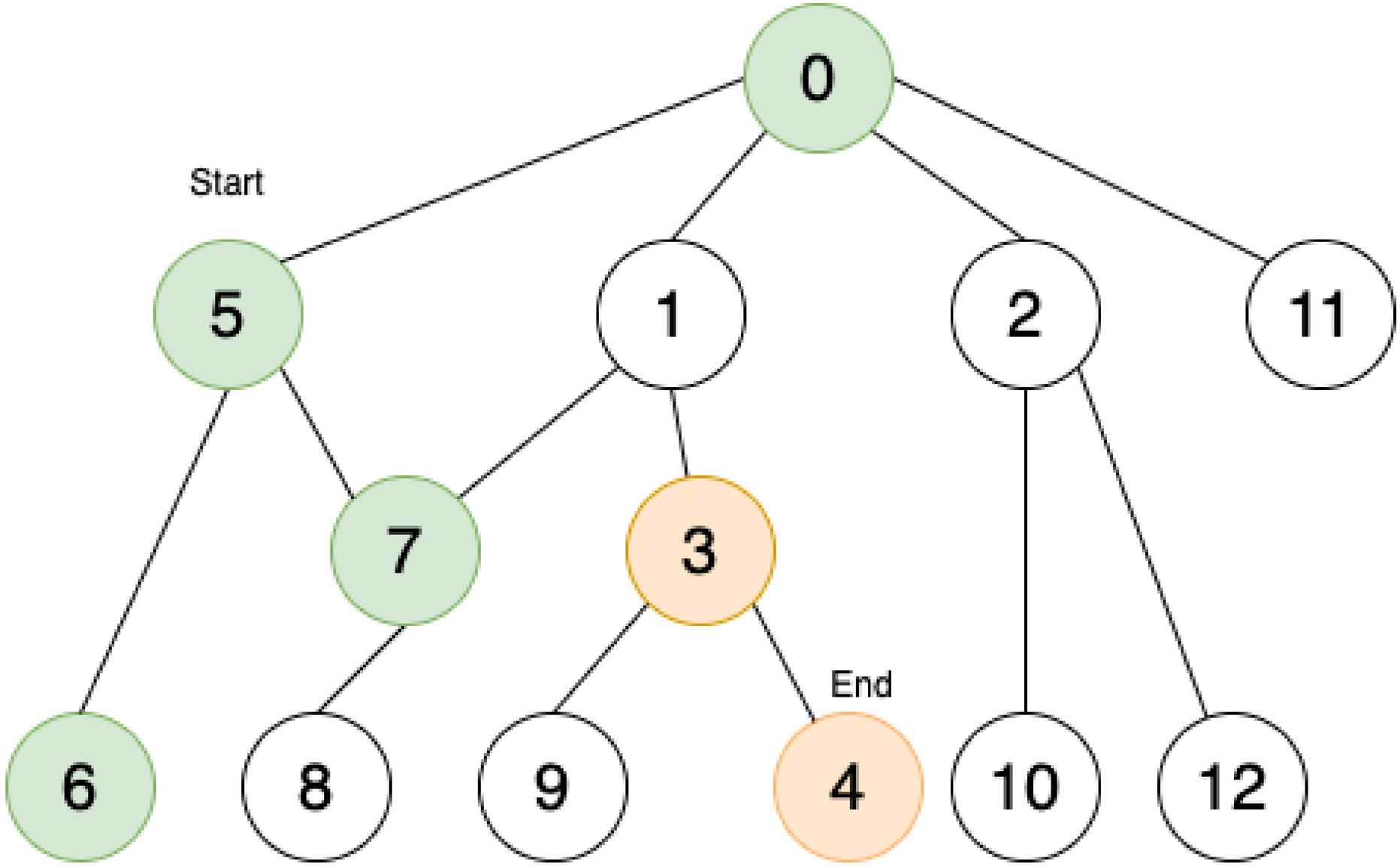
Retorna caminho inexistente



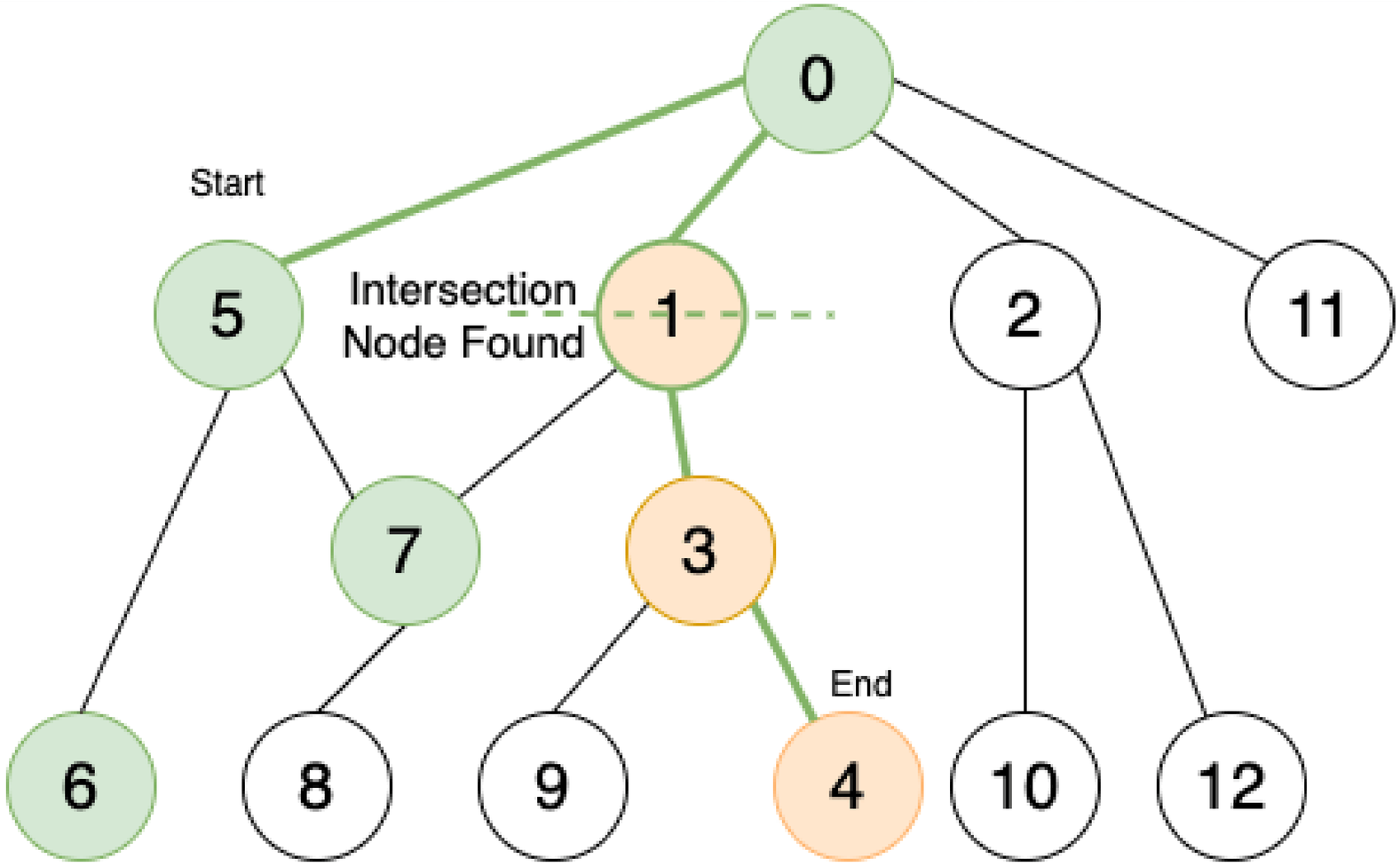
Busca Bidirecional



Busca Bidirecional



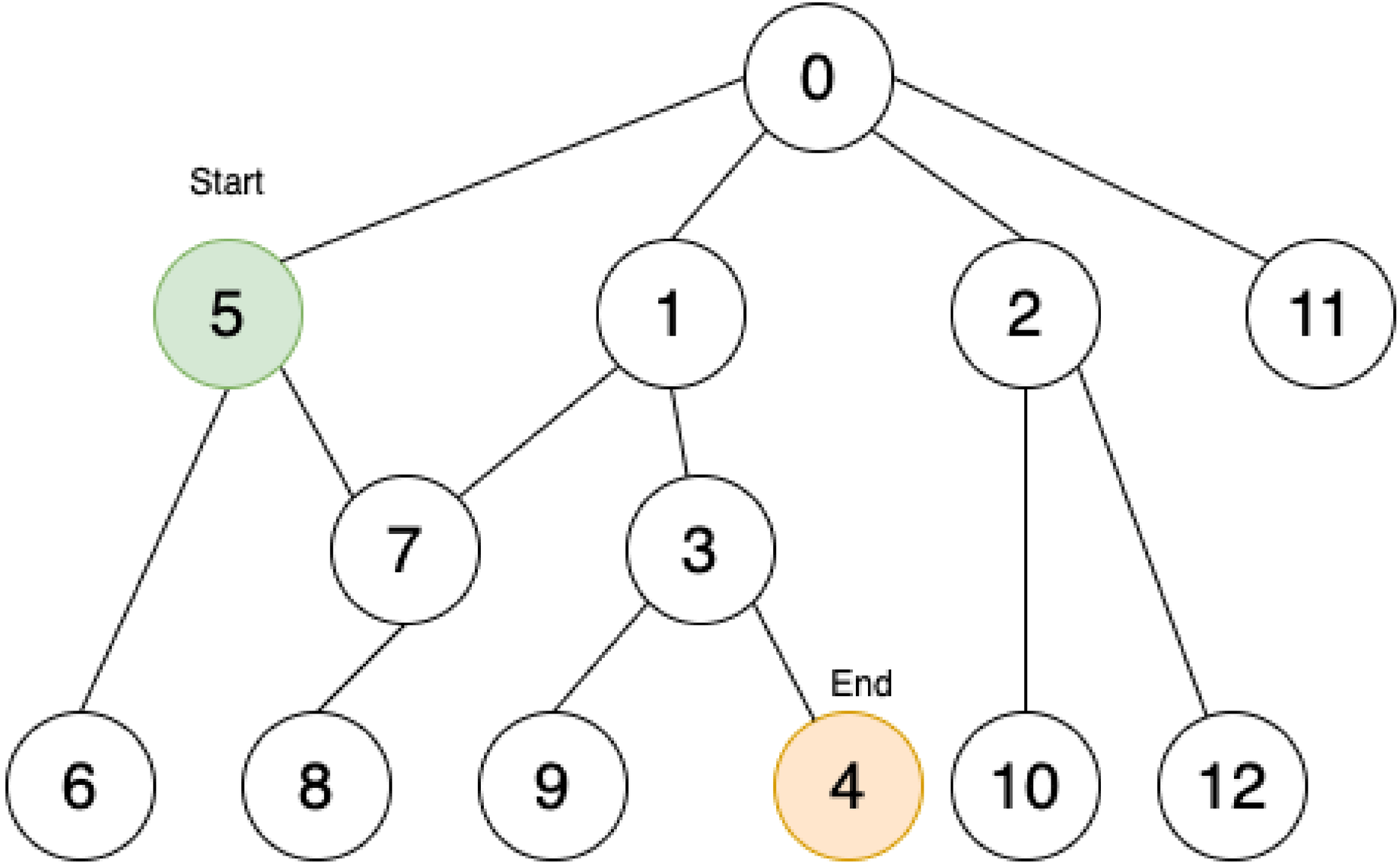
Busca Bidirecional



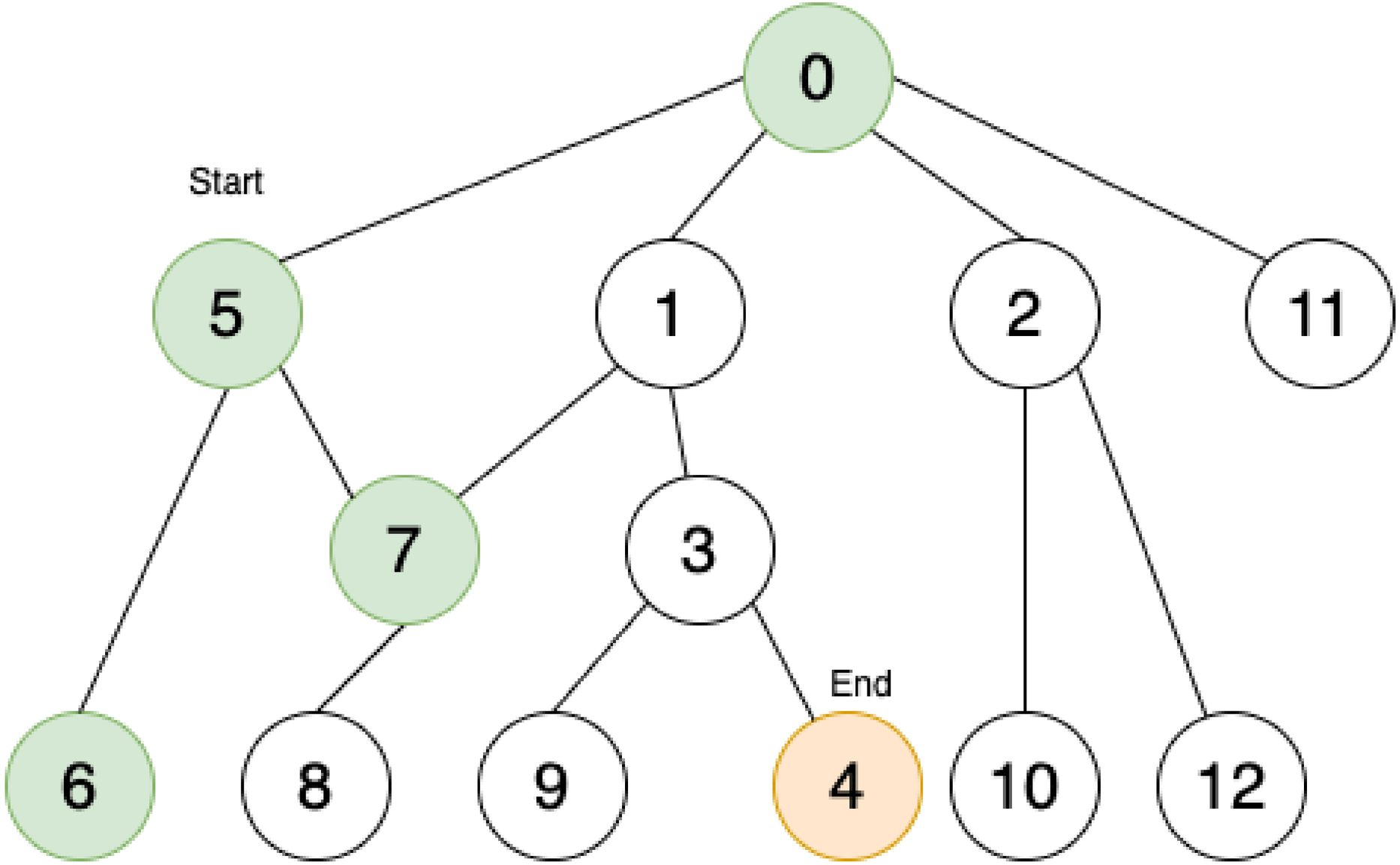
Path found!

5 → 0 → 1 → 3 → 4

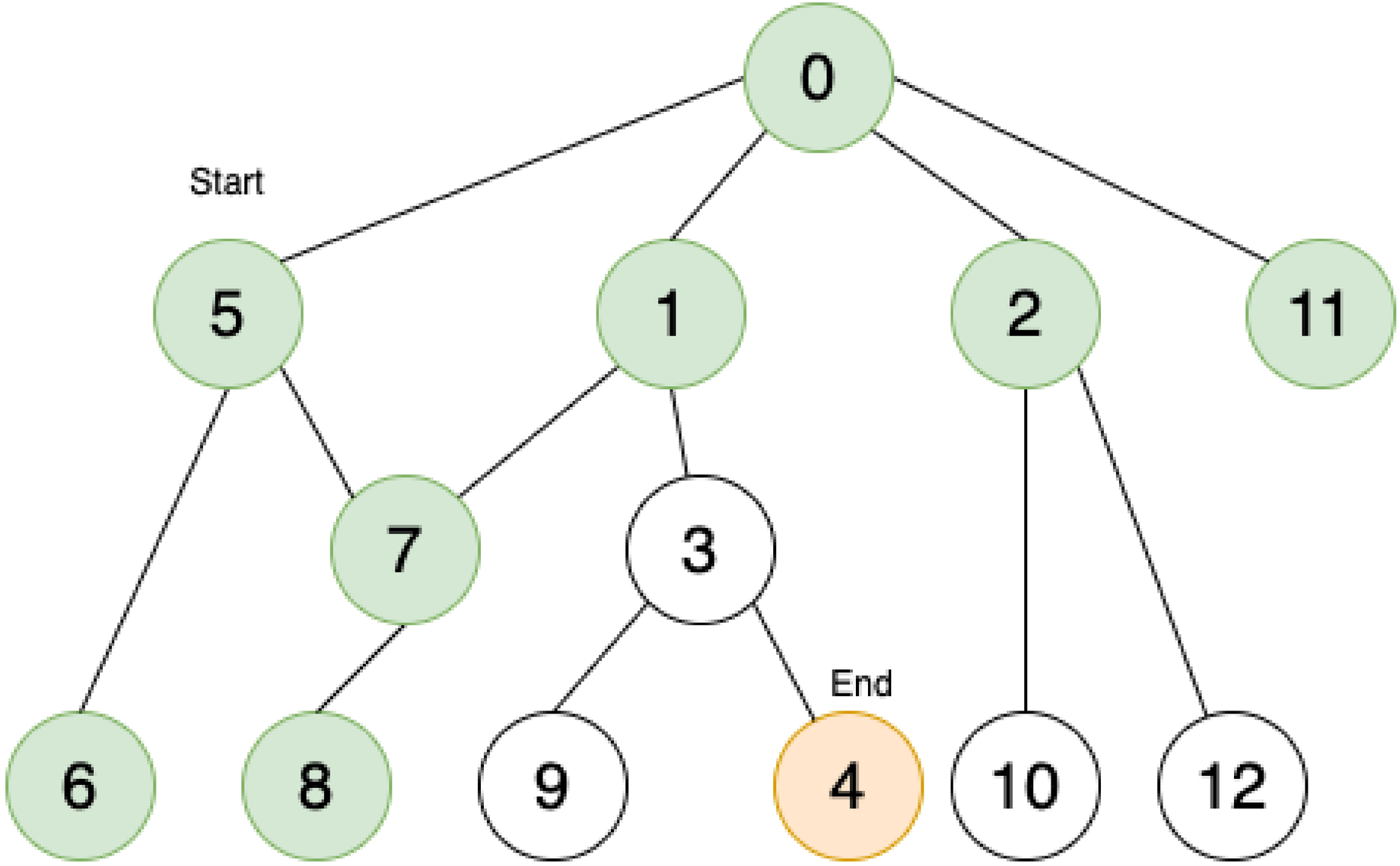
BFS



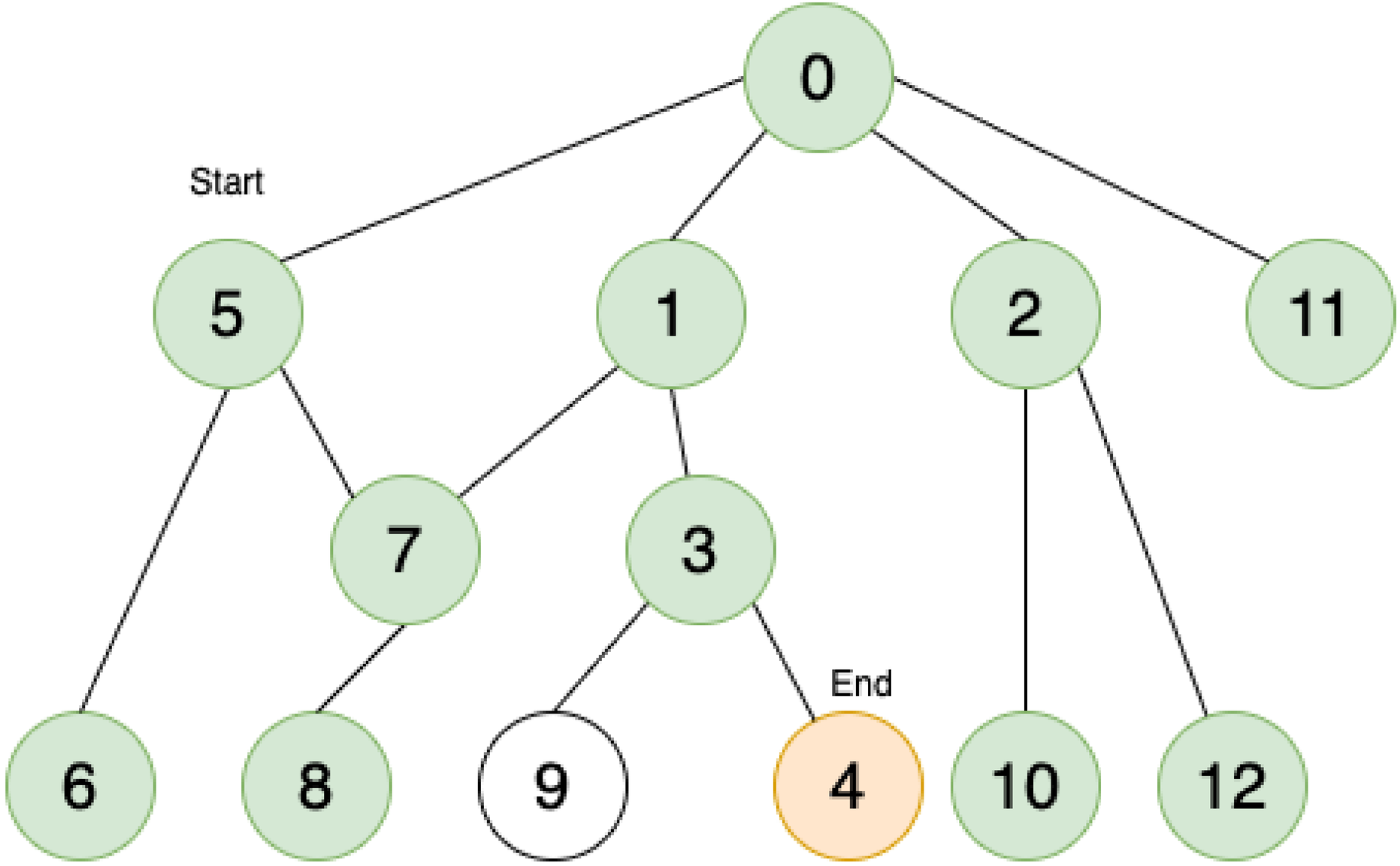
BFS



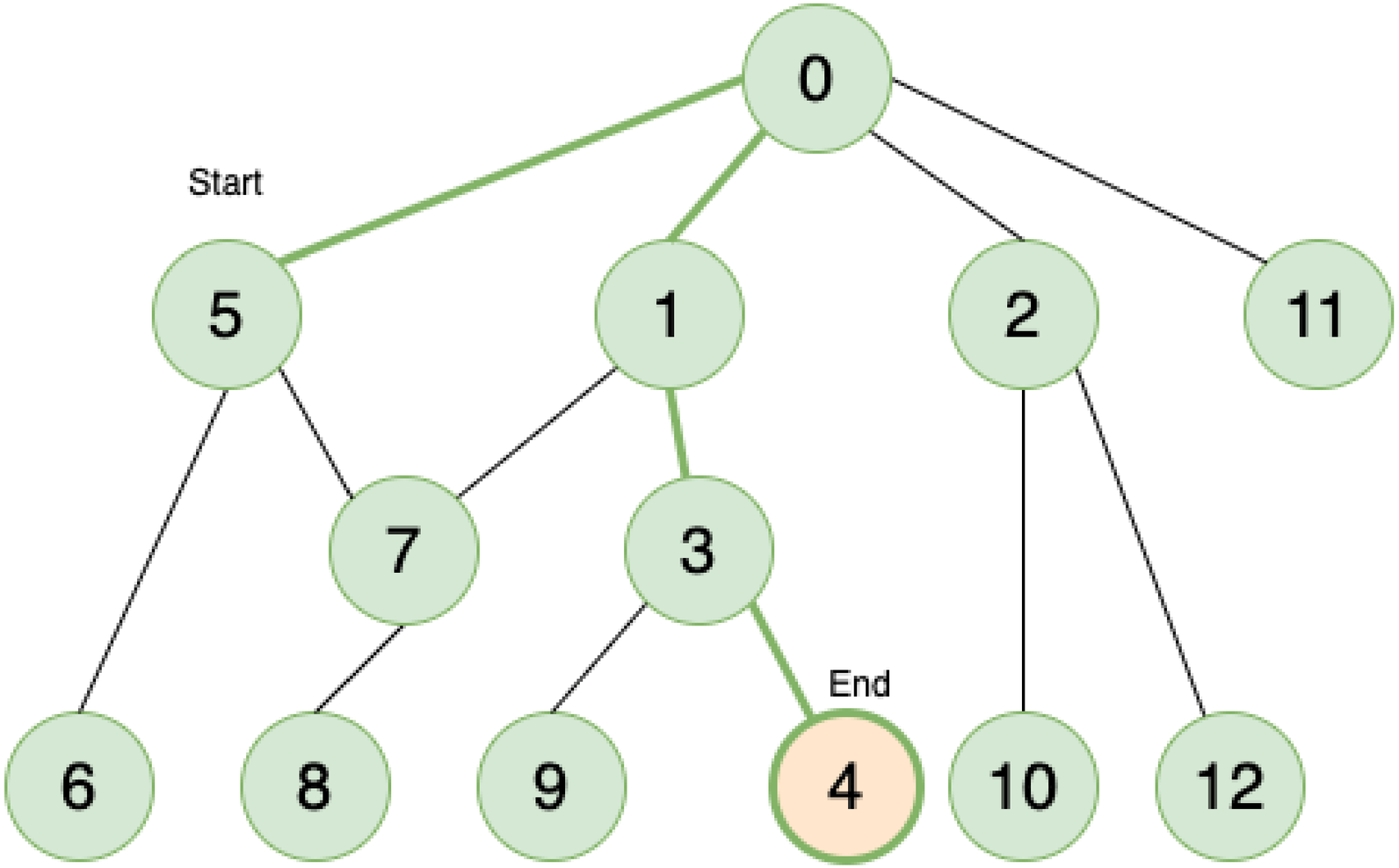
BFS



BFS



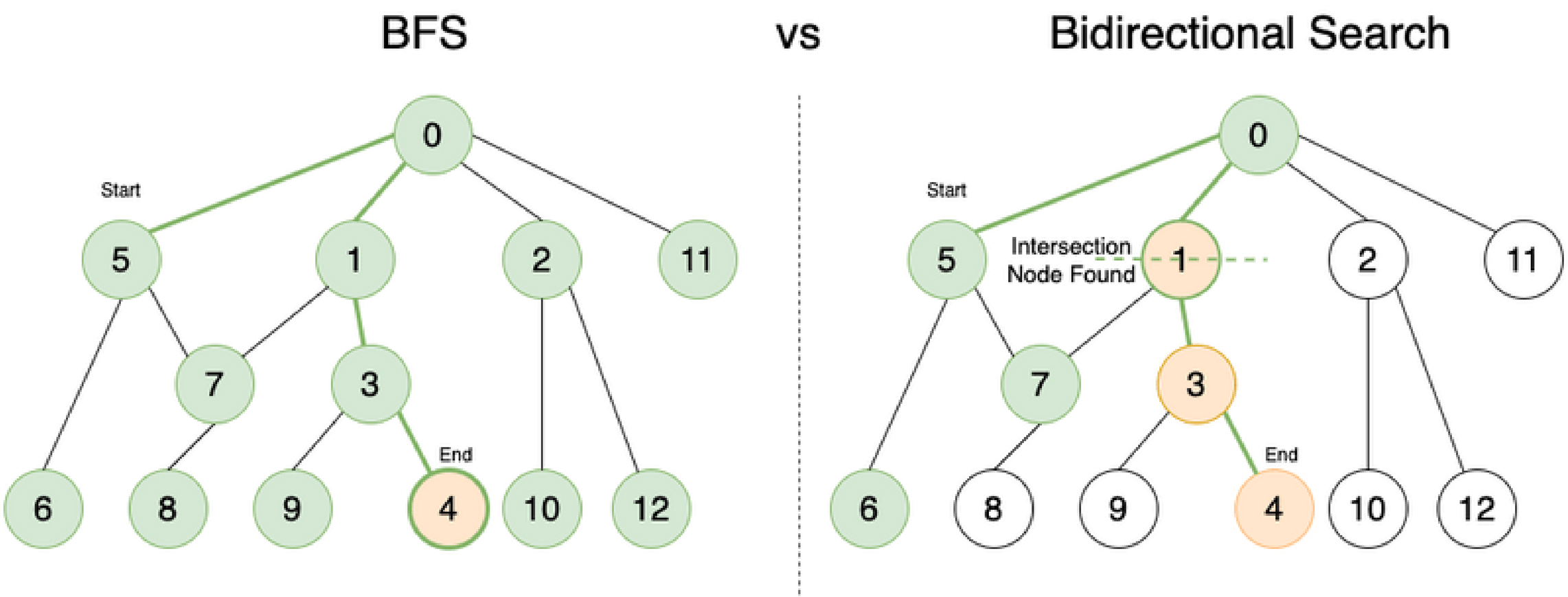
BFS



Path found!

5 → 0 → 1 → 3 → 4

Busca Bidirecional x BFS



Algoritmo - Implementação

Google Colab

Utilizando python no Google Colab

Complexidade

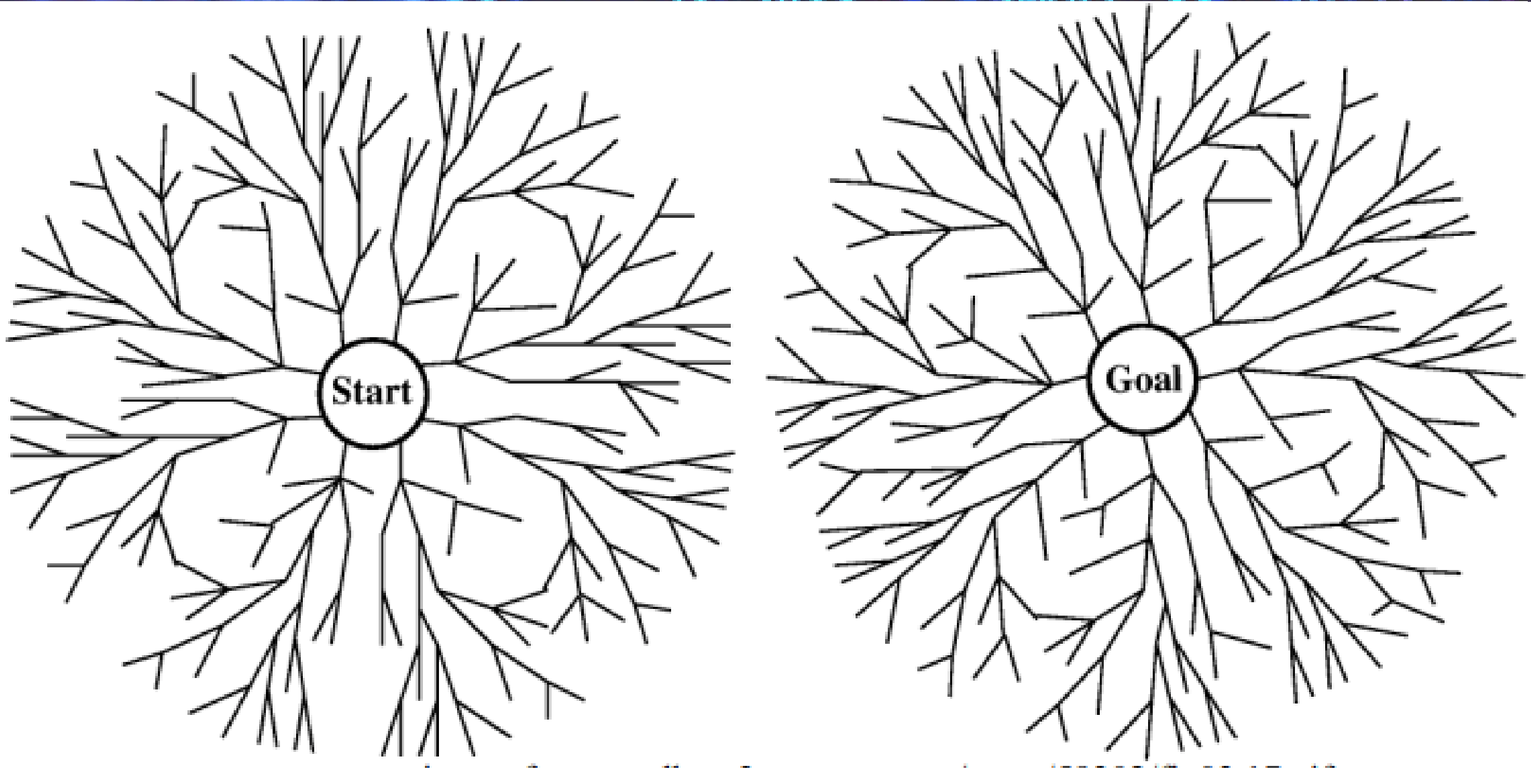
Complexidade de Tempo

O(b^(d/2)). Tendo 'b' como fator de ramificação e 'd' de profundidade.

Complexidade de Espaço

O(b^(d/2)). Tendo 'b' como fator de ramificação e 'd' de profundidade.

Análise de Complexidade



BFS:

$$b + b^2 + b^3 + \ldots + b^d = O(b^d)$$

Bidirectional search:

$$b + b^2 + b^3 + \ldots + b^{d/2} = O(b^{d/2})$$

Aplicações

Existência de um caminho entre dois vértices em um grafo:

- Resolver labirintos e encontrar um caminho entre duas cidades (superior a BFS).

Caminho mais curto em grafos não ponderados ou ponderados de peso único.

Otimizar algoritmos como o Djsktra (Djsktra Birecional) e caminho mais curto (em grafos de peso múltiplo)

- Distância entre cidades

Ações de um agente inteligente para alcançar uma meta.

.

Bibliografia

- https://iq.opengenus.org/bidirectional-search/#:~:text=Bidirectional%20Search%20is%20Graph%20Search,for%20other%20applications%20as%20well.

- Pavlik, J.A., Sewell, E.C. & Jacobson, S.H. Two new bidirectional search algorithms. Comput Optim Appl 80, 377–409 (2021). https://doi.org/10.1007/s10589-021-00303-5

- H. Ma and R. Liang, "Using Bidirectional Search to Compute Optimal Shortest Paths over Multi-weight Graphs," 2013 International Conference on Information Science and Cloud Computing Companion, 2013, pp. 66-71, doi: 10.1109/ISCC-C.2013.150.

- Aplicação de Busca Bidirecional em Planejamento como Verificação Simbólica de Modelos: https://repositorio.ufc.br/handle/riufc/39485

- https://pt.coursera.org/lecture/algorithms-on-graphs/bidirectional-search-t6k1V

Obrigado!

Sinta-se à vontade para perguntar se tiver alguma dúvida.