Os Caminhos da Busca em Largura (*The Breadth-First Search and Its Paths in PICAT, in Portugese*)

Claudio Cesar de Sá

2025, February

Motivação ...

- Num problema do AoC-2024, Problema 16, precisei recuperar o menor caminho entre dois pontos.
- O grafo vinha de um labirinto BI-DIRECIONAL
- Claro que um A* neste caso seria o mais indicado
- Mas não era a questão. Era um Djkstra ou um *planning* que resolvia aquele problema.

Agradecimentos ...

- Aos vários autores de figuras que tomei emprestado na WEB e aqui usadas
- Sem referência mas no Google images
- Hakan, Neng-Fa, comunidade PICAT
- https://screen-recorder.com (Free online screen recorder)

Roteiro

- ~~A linguagem PICAT~~
- ~~O extensão para PICAT se executar sob o ambiente Jupyter~~
- ~~Facilidades e desvantagens~~
- Grafo → um problema
- ~~Representações de grafos em PICAT~~
- “Breadth First Search” (BFS)
- Implementações (2): uma clássica introdutória; uma com a geração de todos caminhos (motivação do título)
- Claro → explosão combinatorial!

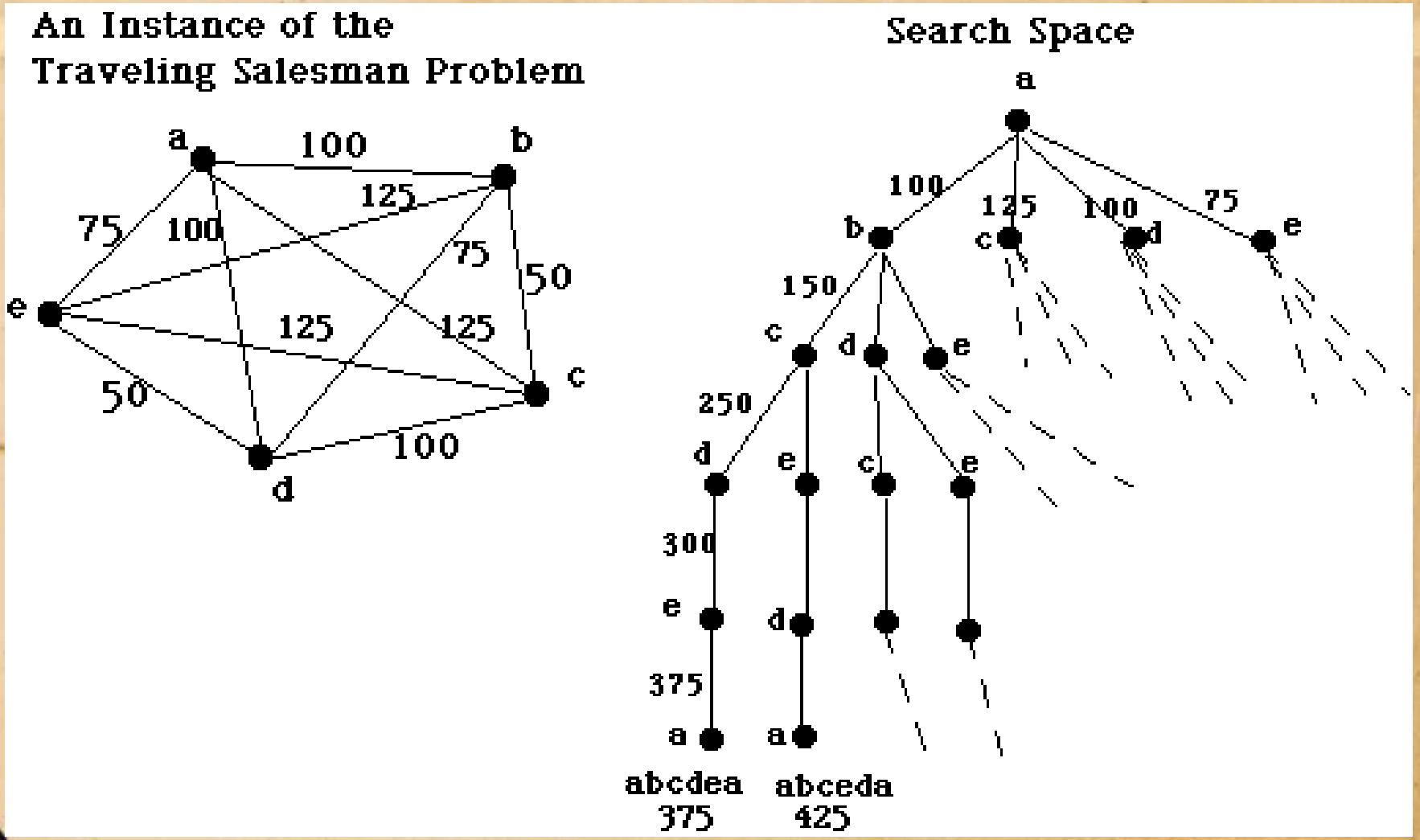
Pré-requisitos: ficam a seu critério

- Buscas em grafos: BFS, DFS, etc
- Quanto ao percurso que um BFS executa (*transversal* ou visitações) há muitos vídeos em detalhes
- Conhecimento de PICAT – nível intermediário – avançamos aqui.
- Estrutura de dados
- Buscas em grafos: BFS, DFS, etc
- Caso esteja ausente dos requisitos … faça uma ‘backtracking’ de aprendizagem

O que tem de legal nestes códigos:

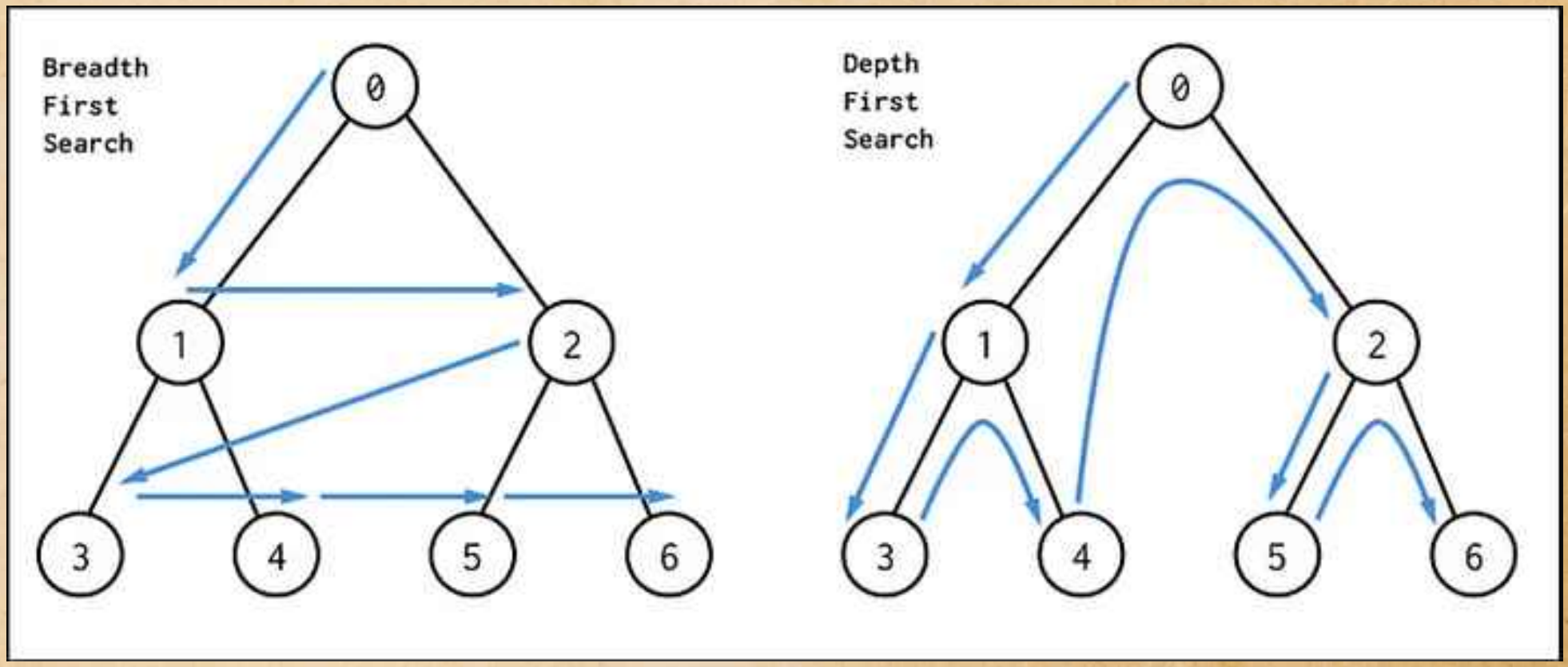
- Casamento de padrões (unificação)
- Códigos legíveis (sem otimizações). Feitos para entender.
- Várias cláusulas escritas em estilos: **predicativo** (*Prolog-like*) e **funcional** (*Haskell-like*)
- Códigos apresentados no meu GitHub claudiosa/CCS/picat/YOUTUBE_bfs_picat/

Idéia: um problema, uma representação em grafos, uma busca: conceito onipresente!

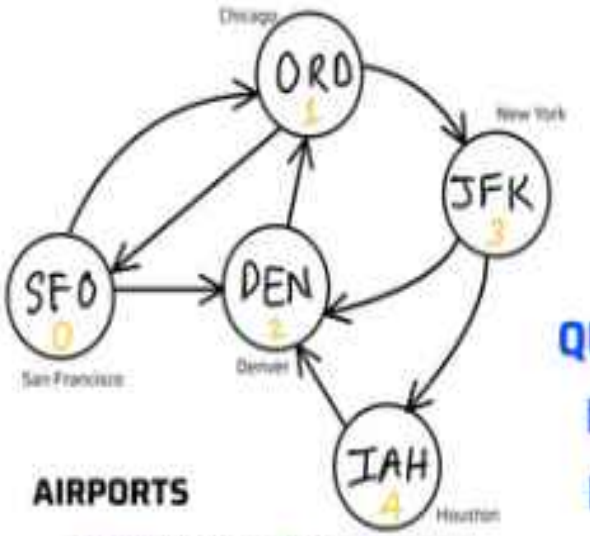


Aqui se deseja voltar ao ponto inicial … um ciclo.
Mas, o que é um ciclo?

BFS (Breadth-First Search) X DFS (Depth-First Search)

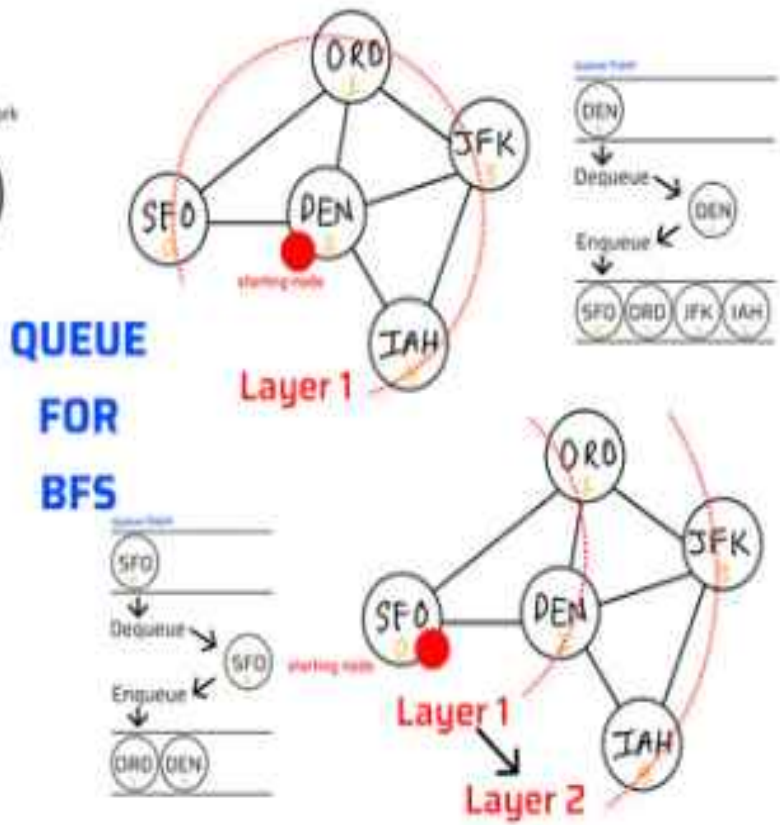


Nomenclatura e uso:

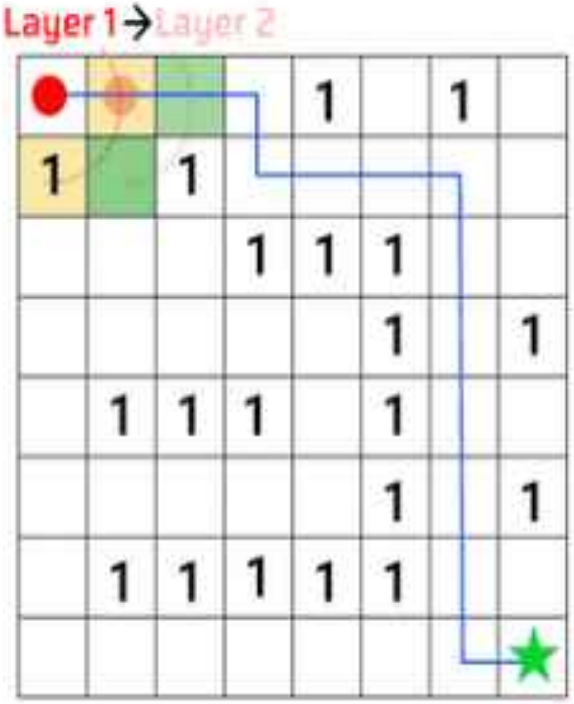



```
const graph1 = [
  [1, 2], // SFO, index 0
  [0, 3], // ORD, index 1
  [1], // DEN, index 2
  [2, 4], // JFK, index 3
  [2], // IAH, index 4
]
```

Review Graph



BFS for graphs



BFS for matrix

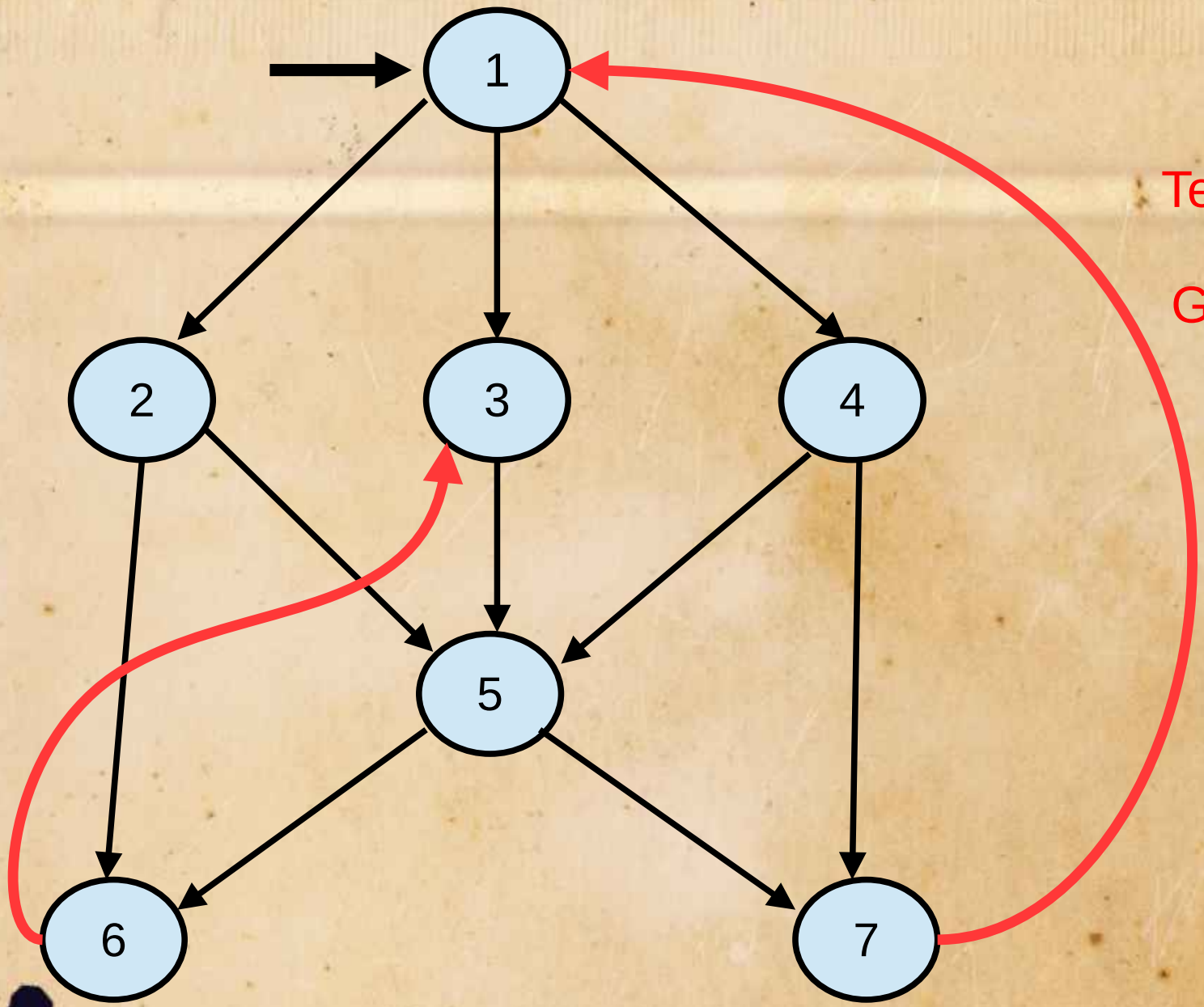
Complexidade de alguns algoritmos: ….
b: branch, d: depth

A comparison table between DFS, BFS and IDDFS

	Time Complexity	Space Complexity	When to Use ?
DFS	$O(b^d)$	O (d)	=> Don't care if the answer is closest to the starting vertex/root. => When graph/tree is not very big/infinite.
BFS	$O(b^d)$	$O(b^d)$	=> When space is not an issue => When we do care/want the closest answer to the root.
IDDFS	$O(b^d)$	O(bd)	=> You want a BFS, you don't have enough memory, and somewhat slower performance is accepted. In short, you want a BFS + DFS.

Sobre os grafos apresentado:

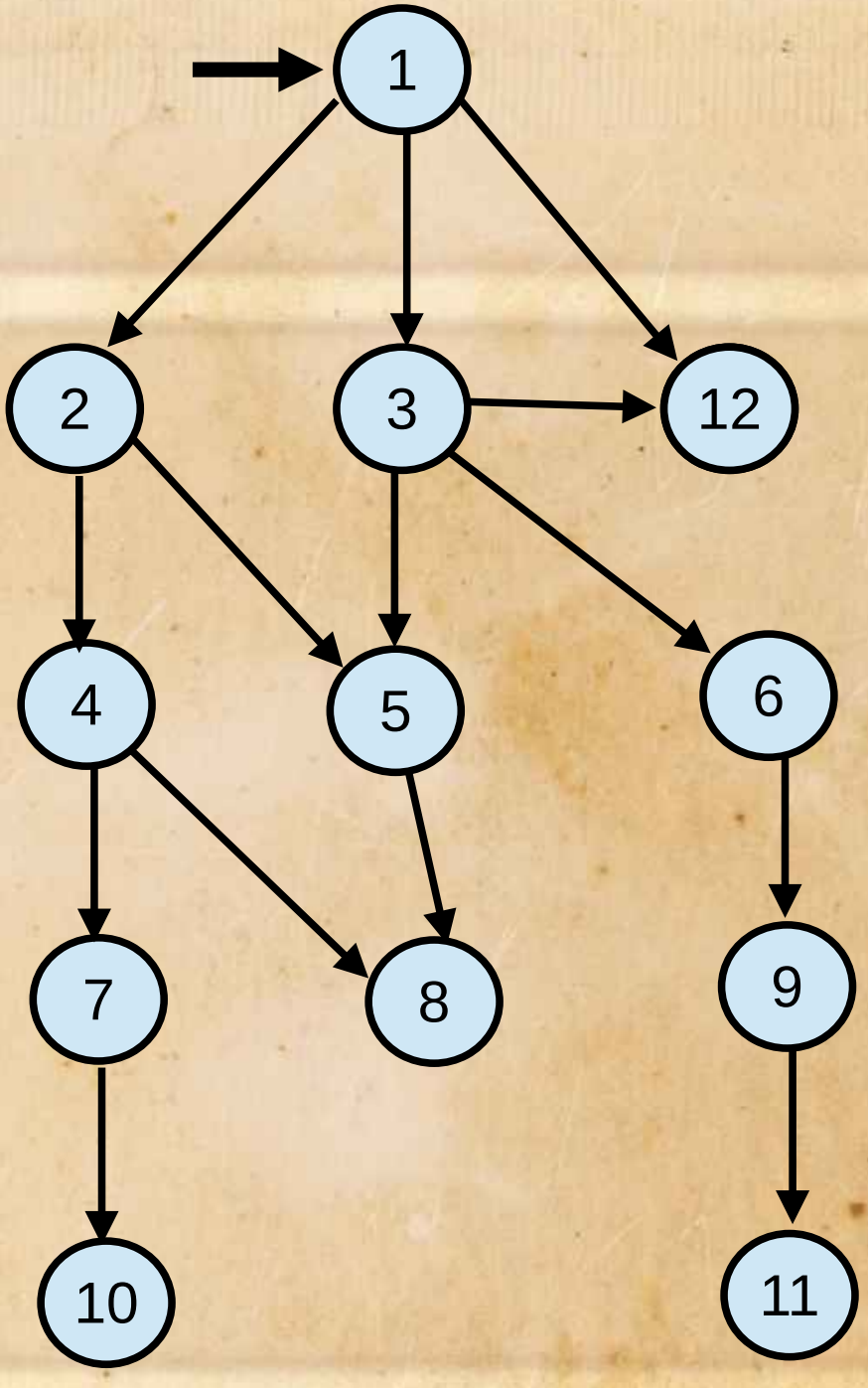
- Foram criados por mim
- As arestas não tem peso (valor, rótulo), mas pouca alteração teria que ser feita neste modelo
- O objetivo é explicar mais sobre código PICAT e detalhar o BFS para **recuperar caminhos**.
- Estes armazenamento de caminhos leva a complexidade deste algoritmo … ver tabela da página anterior.



Tem ciclos, **não** é
um
Grafo Orientado!

Grafo 01

Um DAG ...



Grafo 02

BFS – Visão Clássica

- Uso do BFS: caminho ótimo pois é o mais curto entre dois pontos
- Eficiência: há um preço a pagar. Isso que veremos.
- Legibilidade: boa
- Representação do grafo: Lista de listas como matriz de adjacência
- Grafo é a estrutura algébrica completa, a matriz de adjacências é como os nós se relacionam
- Vá ao código

BFS – salvando todos caminhos

- Eficiência: igual
- Legibilidade: um pouco mais complexa
- Fácil de modificar o código
- Ótima representação do grafo: lista de lista como nós de adjacência
- Vá ao código

Conclusões:

- O núcleo de um BFS é uma fila
- Duas variações do BFS
- Uma versão que fornece apenas um caminho visitado até um nó fim
- Na segunda versão, a cada passo TODAS as trilhas/caminhos possíveis são salvos até se encontrar o nó fim
-
-

Conclusões ...

- Em vários momentos, você precisa recuperar dados, como coordenadas, caminhos, etc,
- BFS é exponencial, logo, precisa apurar e cuidar seu uso
- Usar quando você está próximo, porém um DFS pode ser muito custoso

Obrigado ...

- Todo *feedback* é bem-vindo
- ccs1664@gmail.com ou ccs1664@yahoo.com