Estrutura de Dados (CCA410)

Aula 10 – Busca em Grafos

Prof. Luciano Rossi
Prof. Leonardo Anjoletto Ferreira
Prof. Flavio Tonidandel
Prof. Fabio Suim

Ciência da Computação
Centro Universitário FEI

2º Semestre de 2025

Grafos

Classificação das Estruturas de Dados

Primitivas	Compostas		
	Simples	Lineares	Não-Lineares
Inteiro	Cadeia (string)	Pilha	Árvore
Real	Registro (struct)	Fila	**Grafo**
Lógico	Arranjo (array)	Lista	
Caractere			

Algoritmos de Busca em Grafos

centro universitário FEI

Algoritmos de Busca em Grafos

Definição

Algoritmos de Busca em Grafos

São métodos sistemáticos para explorar e percorrer as estruturas de grafos, visitando seus vértices (nós) e arestas de forma ordenada. Eles são fundamentais para resolver problemas de navegação, conectividade e análise de redes.

centro universitário FEI

Objetivos

- Encontrar caminhos entre dois vértices
- Verificar conectividade
- Detectar ciclos
- Explorar toda a estrutura do grafo

Busca em Largura (*Breadth First Search (BFS)*)

Algoritmos de Busca em Grafos

Busca em Largura (*Breadth First Search (BFS)*)

Características

- Explora o grafo "nível por nível"
- Visita primeiro todos os vizinhos diretos, depois os vizinhos dos vizinhos
- Usa uma fila como estrutura auxiliar
- Encontra o caminho mais curto (em número de arestas)

centro universitário FEI

Algoritmos de Busca em Grafos

Busca em Largura (*Breadth First Search (BFS)*) - Simulação

- $BFS(G, s)$:

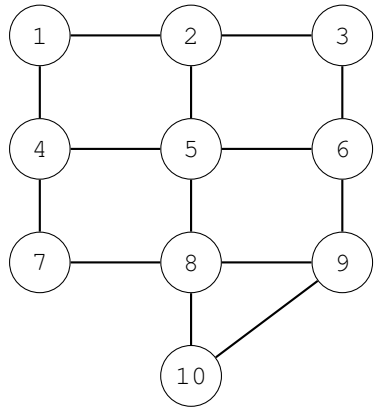
```
1  para cada vértice v_i em G.V − {s} faça
2      v_i.d = ∞
3      v_i.cor = BRANCO
4  s.d = 0
5  s.cor = CINZA
6  Q = VAZIO
7  Insere(Q, s)
8  enquanto Q! = VAZIO faça
9      u_i = Remove(Q)
10     para cada v_i em G.Adj[u_i] faça
11         se v_i.cor == BRANCO
12             v_i.d = u_i.d + 1
13             v_i.cor = CINZA
14             Insere(Q, v_i)
15     u_i.cor = PRETO
```

Algoritmos de Busca em Grafos

Busca em Largura (*Breadth First Search (BFS)*) - Simulação

- $BFS(G, s)$:

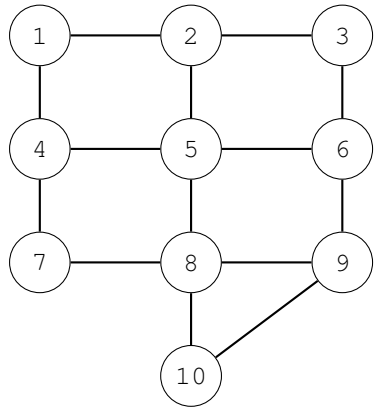
1 para cada vértice v_i em $G.V - \{s\}$ faça
2 $\quad v_i.d = \infty$
3 $\quad v_i.cor = BRANCO$
4 $s.d = 0$
5 $s.cor = CINZA$
6 $Q = VAZIO$
7 $Insere(Q, s)$



Algoritmos de Busca em Grafos

Busca em Largura (*Breadth First Search (BFS)*) - Simulação

```
8  enquanto Q! = VAZIO faça
9      u_i = Remove(Q)
10     para cada v_i em G.Adj[u_i] faça
11         se v_i.cor == BRANCO
12             v_i.d = u_i.d + 1
13             v_i.cor = CINZA
14             Insere(Q, v_i)
15     u_i.cor = PRETO
```



Busca em Profundidade (*Depth First Search (DFS)*)

centro universitário FEI

Algoritmos de Busca em Grafos

Busca em Profundidade (*Depth First Search (DFS)*)

Características

- Explora o grafo "o mais fundo possível" antes de retroceder
- Segue um caminho até não poder mais avançar, depois volta
- Usa uma pilha (ou recursão) como estrutura auxiliar
- Útil para detecção de ciclos e ordenação topológica

centro universitário FEI

Algoritmos de Busca em Grafos

Busca em Profundidade (*Depth First Search (DFS)*) - Simulação

```
DFS(G)
1  para cada vértice u_i em G.V faça
2      u_i.cor = BRANCO
3  tempo = 0
4  para cada vértice u_i em G.V faça
5      se u_i.cor == BRANCO
6          entao VisitaDFS(G, u_i)
```

```
VisitaDFS(G, u_i)
1  tempo = tempo+1
2  u_i.d = tempo
3  u_i.cor = CINZA
4  para cada v_i em Adj[u_i]
5      se v_i.cor == BRANCO
6          VisitaDFS(G, v_i)
7  tempo = tempo+1
8  u_i.f = tempo
9  u_i.cor = PRETO
```

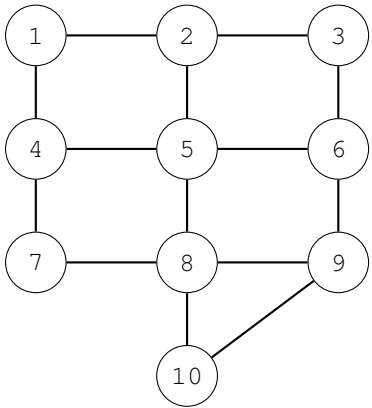
centro universitário FEI

Algoritmos de Busca em Grafos

Busca em Profundidade (*Depth First Search (DFS)*) - Simulação

$DFS(G)$

1 para cada vértice u_i em $G.V$ faça
2 $\quad u_i.cor = BRANCO$
3 tempo $= 0$
4 para cada vértice u_i em $G.V$ faça
5 $\quad$ se $u_i.cor == BRANCO$
6 $\quad\quad$ entao $VisitaDFS(G, u_i)$



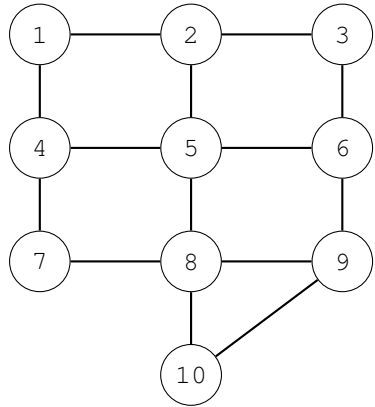
centro universitário FEI

Algoritmos de Busca em Grafos

Busca em Profundidade (*Depth First Search (DFS)*) - Simulação

$VisitaDFS(G, u_i)$

1 tempo = tempo+1
2 $u_i.d =$ tempo
3 $u_i.cor = CINZA$
4 para cada v_i em $Adj[u_i]$
5 se $v_i.cor == BRANCO$
6 $VisitaDFS(G, v_i)$
7 tempo = tempo+1
8 $u_i.f =$ tempo
9 $u_i.cor = PRETO$



Algoritmos de Busca em Grafos

Comparação entre BFS e DFS

BFS (Busca em Largura)

Características

- Explora nível por nível
- Usa fila (FIFO)
- Garante caminho mais curto
- Complexidade: $O(V + E)$

Estrutura de Dados

- Fila para controle de visitação
- Array de distâncias
- Array de predecessores

Aplicações

- Caminho mais curto
- Conectividade
- Análise de níveis
- Redes sociais

DFS (Busca em Profundidade)

Características

- Explora o mais fundo possível
- Usa pilha ou recursão
- Não garante caminho mais curto
- Complexidade: $O(V + E)$

Estrutura de Dados

- Pilha (ou recursão)
- Tempos de descoberta/finalização
- Classificação de arestas

Aplicações

- Detecção de ciclos
- Ordenação topológica
- Componentes conexos
- Labirintos

centro universitário FEI

Algoritmos de Busca em Grafos

Quando Usar Cada Algoritmo

Use BFS quando:

- Precisar do caminho mais curto
- Objetivo está próximo da origem
- Analisar conectividade por níveis
- Graus de separação em redes
- Algoritmos de roteamento

Vantagens

+ Caminho ótimo garantido
+ Exploração sistemática
+ Bom para análise de proximidade

Desvantagens

- Alto uso de memória
- Pode ser lento para objetivos distantes

Use DFS quando:

- Detectar ciclos no grafo
- Fazer ordenação topológica
- Analisar componentes conexos
- Qualquer caminho é suficiente
- Exploração completa necessária

Vantagens

+ Menor uso de memória
+ Informação temporal rica
+ Implementação recursiva natural

Desvantagens

- Caminho pode não ser ótimo
- Risco de stack overflow

centro universitário FEI

Algoritmos de Busca em Grafos

Resumo Comparativo

Critério	BFS	DFS
Estrutura de Dados	Fila	Pilha/Recursão
Caminho Encontrado	Mais Curto	Qualquer
Complexidade Temporal	$O(V+E)$	$O(V+E)$
Complexidade Espacial	$O(V)$	$O(V)$
Uso de Memória	Alto (grafos largos)	Baixo (proporcional à profundidade)
Detecção de Ciclos	Possível	Muito Eficiente
Implementação	Naturalmente iterativa	Naturalmente recursiva
Informação Temporal	Não	Sim (descoberta/finalização)
Classificação de Arestas	Limitada	Completa

Conclusão

Ambos algoritmos têm **mesma complexidade assintótica** mas comportamentos diferentes. A escolha depende do **objetivo específico** e da **estrutura do grafo**.

Estrutura de Dados (CCA410)

Aula 10 – Busca em Grafos

Prof. Luciano Rossi
Prof. Leonardo Anjoletto Ferreira
Prof. Flavio Tonidandel
Prof. Fabio Suim

Ciência da Computação
Centro Universitário FEI

2º Semestre de 2025