Teoria dos Grafos
Unidade 3:
Busca em Grafos

Prof. Dr. Paulo César Rodacki Gomes
rodacki@ifc-riodosul.edu.br

INSTITUTO FEDERAL DE
EDUCAÇÃO, CIÊNCIA E TECNOLOGIA
CATARINENSE
Campus Rio do Sul

Blibliografia

- Joan M. Aldous, Robin J. Wilson. **Graphs and Applications**: as introductory approach. Springer. 2001
- Thomas Cormen et al. **Algoritmos**: teoria e prática. Ed. Campus. 2004.

Tópicos

- Motivação
- Caminhos
- Árvores de busca
- Busca em Largura
- Busca em Profundidade
- Aplicações

Busca

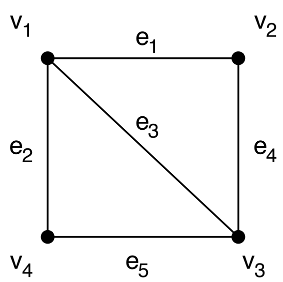
- Pesquisar um grafo: visitar seus vértices passando sistematicamente por suas arestas
- Busca em largura
- Busca em profundidade
- A busca forma caminhos nos grafos

Definições

- Passeio: um passeio é uma sequencia de vértices e arestas de tal forma que o vértico final de uma aresta é igual ao vértice inicial da próxima aresta no passeio
- O comprimento (ou distância) de um passeio é igual ao somatório de vezes que cada aresta é percorrida

Definições

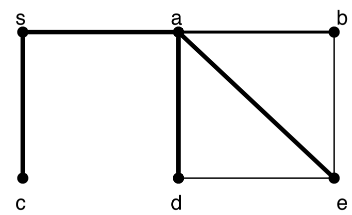
- Trilha: é um passeio sem repetição de arestas
- Caminho: é uma trilha sem repetição de vértices (exceto possivelmente pelos vértices inicial e final

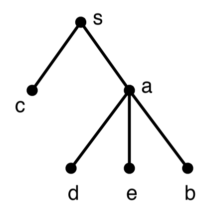


Busca em largura

- explora o grafo a partir de um vértice inicial arbitrário
- vai se alastrando uniformemente pelo grafo
- o conjunto de caminhos do vértice inicial a cada um dos demais vértices gera uma árvore de busca em largura

Árvore de Busca





- Vetor de roteamento

$$\pi(s) = nil,\ \pi(a) = s,\ \pi(b) = a,\ \pi(c) = s,\ \pi(d) = a \text{ e } \pi(e) = a$$

Algoritmo de busca em Largura (BFS)

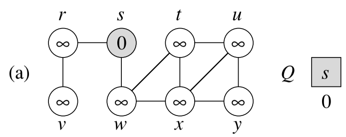
- s: representa o vértice inicial;
- $cor(v)$: cor do vértice v, que pode assumir um dos seguintes valores $\{BRANCO, CINZA, PRETO\}$. O valor BRANCO significa que a busca ainda não atingiu este vértice. O valor CINZA, indica que a busca já atingiu o vértice v, mas ele ainda possui vizinhos (vértices adjacentes) ainda não visitados. O valor PRETO indica que a visitação deste vértice está encerrada, isto é, tanto ele como todos os seus vizinhos já foram visitados pela busca;
- $\pi(v)$: referência ao vértice predecessor do vértice v na busca, representa seu pai ná árvore de busca em largura;
- Q: fila de vértices, que deve implementar as operações INSERE(Q,v), resposável pela inserção do vértice v no final da fila, e REMOVE(Q), que remove e retorna o primeiro vértice da fila Q;
- $d(v)$: atributo numérico que armazena a distância de s até v em quantidade de arestas percorridas.

Algoritmo: BFS(G, s)

para cada $v \in V$ **faça**
  $cor(v) \leftarrow BRANCO$;
  $\pi(v) \leftarrow nil$;
  $d(v) \leftarrow \infty$;

$d(s) \leftarrow 0$;
$cor(s) \leftarrow CINZA$;
$Q \leftarrow \emptyset$;
$INSERE(Q, s)$;

enquanto $Q \neq \emptyset$ **faça**
  $u \leftarrow REMOVE(Q)$;
  para cada $v \in Adj(u)$ **faça**
    se $cor(v) = BRANCO$ **então**
      $INSERE(Q, v)$;
      $cor(v) \leftarrow CINZA$;
      $\pi(v) \leftarrow u$;
      $d(v) \leftarrow d(u) + 1$;
  $cor(u) \leftarrow PRETO$;



Algoritmo: BFS(G, s)

para cada $v \in V$ **faça**
  $cor(v) \leftarrow BRANCO$;
  $\pi(v) \leftarrow nil$;
  $d(v) \leftarrow \infty$;

$d(s) \leftarrow 0$;
$cor(s) \leftarrow CINZA$;
$Q \leftarrow \emptyset$;
$INSERE(Q, s)$;

enquanto $Q \neq \emptyset$ **faça**
  $u \leftarrow REMOVE(Q)$;
  para cada $v \in Adj(u)$ **faça**
    se $cor(v) = BRANCO$ **então**
      $INSERE(Q, v)$;
      $cor(v) \leftarrow CINZA$;
      $\pi(v) \leftarrow u$;
      $d(v) \leftarrow d(u) + 1$;
  $cor(u) \leftarrow PRETO$;

Menor caminho

Definição 1.19 (Menor caminho). *A menor distância de um caminho do vértice s até um vértice v, $\delta(s,v)$, é definida pelo menor número de arestas dentre todos os possíveis caminhos de s para v. Se não existir caminho de s para v, então $\delta(s,v) = \infty$. Um caminho de distância (ou comprimento) $\delta(s,v)$ é chamado menor caminho do vértice s para o vértice v.*

- A árvore de busca em largura contém os menores caminhos de s até cada um dos vértices v atingíveis por s.

Imprimir caminho

Este procedimento imprime, utilizando um vetor de roteamento, o caminho do vértice s para o vértice v (se existir tal caminho)

Algoritmo: ImprimeCaminho(G, s, v)

se $v{=}{=}s$ **então**
  imprime(s);
senão
  se $\pi(v) == nil$ **então**
    imprime("não existe caminho de s para v");
  senão
    ImprimeCaminho$(G, s, \pi(v))$;
    imprime(v);

Busca em profundidade

- $cor(v)$: cor do vértice v, que pode assumir um dos seguintes valores $\{BRANCO, CINZA, PRETO\}$. O valor BRANCO significa que a busca ainda não atingiu este vértice. O valor CINZA, indica que a busca já atingiu o vértice v, mas ele ainda possui vizinhos (vértices adjacentes) ainda não visitados. O valor PRETO indica que a visitação deste vértice está encerrada, isto é, tanto ele como todos os seus vizinhos já foram visitados pela busca;
- $\pi(v)$: referência ao vértice predecessor do vértice v na busca, representa seu pai ná árvore de busca em profundidade;
- *tempo*: relógio lógico, valor inteiro que varia de 0 a $2n$;
- $d(v)$: tempo de descoberta ou abertura do vértice v, representa o valor do tempo lógico quando o vértice é visitado pela primeira vez;
- $f(v)$: tempo de fechamento do vértice v, representa o valor do tempo lógico quando a visitação no vértice é encerrada, isto é, quando sua cor fica preta.

Busca em profundidade

Algoritmo: DFS(G)

para cada $v \in V$ **faça**
  $cor(v) \leftarrow BRANCO$;
  $\pi(v) \leftarrow nil$;
$tempo \leftarrow 0$;
para cada *vértice* $u \in V$ **faça**
  se $cor(u) = BRANCO$ **então**
    $DFS-VISIT(u)$;

Algoritmo: DFS-VISIT(u)

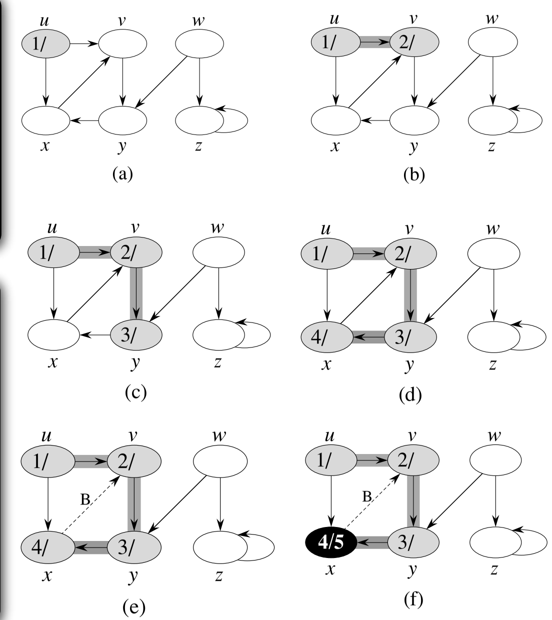
$cor(u) \leftarrow CINZA$;
$tempo \leftarrow tempo + 1$;
$d(u) \leftarrow tempo$;
para cada $v \in Adj(u)$ **faça**
  se $cor(v) = BRANCO$ **então**
    $\pi(v) \leftarrow u$;
    $DFS-VISIT(v)$;
$cor(u) \leftarrow PRETO$;
$tempo \leftarrow tempo + 1$;
$f(u) \leftarrow tempo$;

Algoritmo: DFS(G)

para cada $v \in V$ **faça**
  $cor(v) \leftarrow BRANCO$;
  $\pi(v) \leftarrow nil$;
$tempo \leftarrow 0$;
para cada *vértice* $u \in V$ **faça**
  se $cor(u) = BRANCO$ **então**
    $DFS-VISIT(u)$;

Algoritmo: DFS-VISIT(u)

$cor(u) \leftarrow CINZA$;
$tempo \leftarrow tempo + 1$;
$d(u) \leftarrow tempo$;
para cada $v \in Adj(u)$ **faça**
  se $cor(v) = BRANCO$ **então**
    $\pi(v) \leftarrow u$;
    $DFS-VISIT(v)$;
$cor(u) \leftarrow PRETO$;
$tempo \leftarrow tempo + 1$;
$f(u) \leftarrow tempo$;



Algoritmo: DFS(G)

para cada $v \in V$ **faça**
- $cor(v) \leftarrow BRANCO$;
- $\pi(v) \leftarrow nil$;

$tempo \leftarrow 0$;

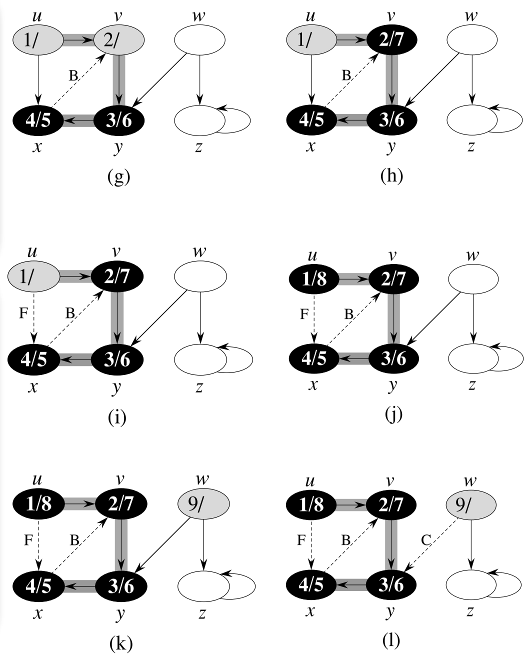
para cada *vértice* $u \in V$ **faça**
- **se** $cor(u) = BRANCO$ **então**
  - $DFS - VISIT(u)$;

Algoritmo: DFS-VISIT(u)

$cor(u) \leftarrow CINZA$;
$tempo \leftarrow tempo + 1$;
$d(u) \leftarrow tempo$;

para cada $v \in Adj(u)$ **faça**
- **se** $cor(v) = BRANCO$ **então**
  - $\pi(v) \leftarrow u$;
  - $DFS - VISIT(v)$;

$cor(u) \leftarrow PRETO$;
$tempo \leftarrow tempo + 1$;
$f(u) \leftarrow tempo$;



Algoritmo: DFS(G)

para cada $v \in V$ **faça**
- $cor(v) \leftarrow BRANCO$;
- $\pi(v) \leftarrow nil$;

$tempo \leftarrow 0$;

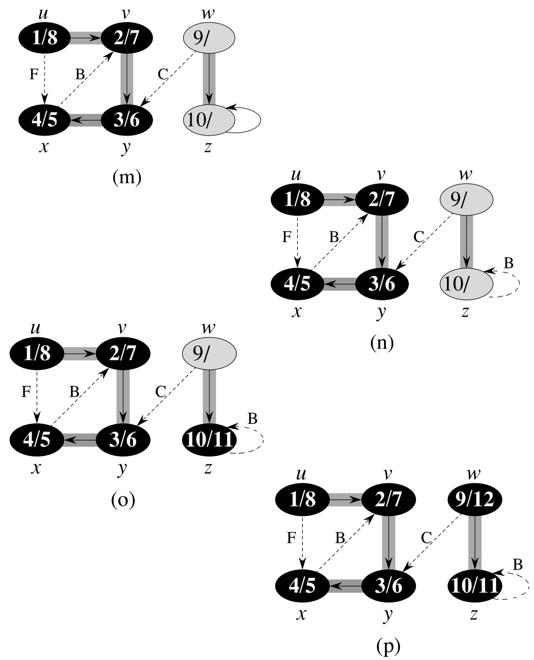
para cada *vértice* $u \in V$ **faça**
- **se** $cor(u) = BRANCO$ **então**
  - $DFS - VISIT(u)$;

Algoritmo: DFS-VISIT(u)

$cor(u) \leftarrow CINZA$;
$tempo \leftarrow tempo + 1$;
$d(u) \leftarrow tempo$;

para cada $v \in Adj(u)$ **faça**
- **se** $cor(v) = BRANCO$ **então**
  - $\pi(v) \leftarrow u$;
  - $DFS - VISIT(v)$;

$cor(u) \leftarrow PRETO$;
$tempo \leftarrow tempo + 1$;
$f(u) \leftarrow tempo$;



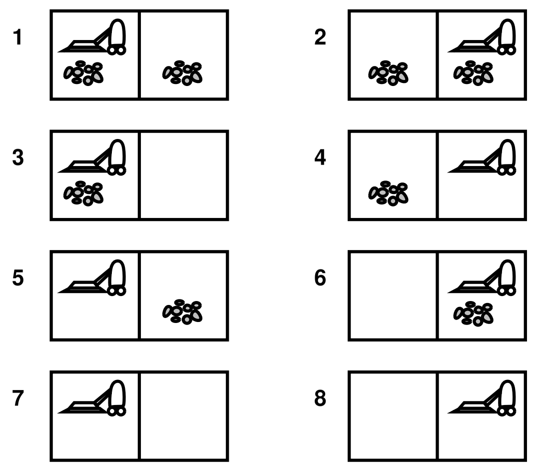
Exemplo de aplicação

- Grafo de estados: é um grafo onde os vértices representam estados de um problema, e as arestas representam possíveis transições entre dois estados

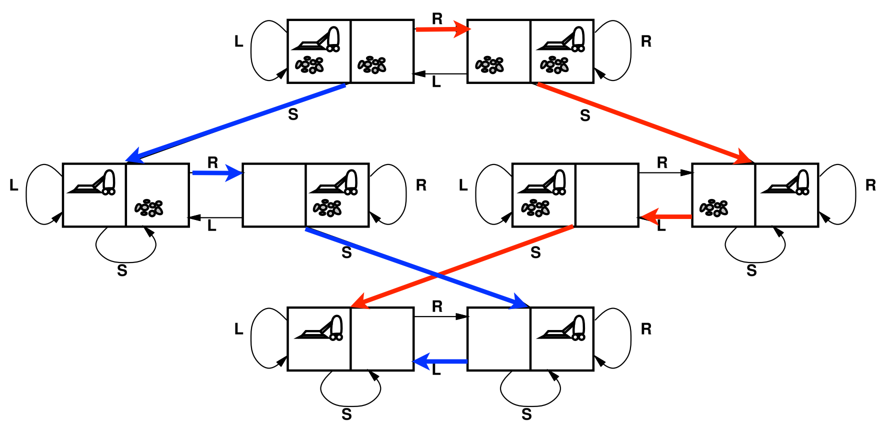
Exemplo: robô aspirador de pó

- Um robô aspirador de pó deve limpar uma casa com duas posições. As operações que ele sabe executar são:
  - sugar
  - ir para a posição da esquerda
  - ir para a posição da direita
- Como o aspirador pode montar um plano para limpar a casa se inicialmente ele esta na posição direita e as duas posições têm sujeira?
  - Quais os estados possíveis do mundo do aspirador e as transições?

- Estados possíveis:

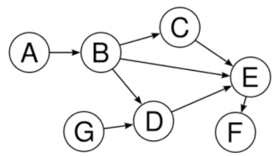


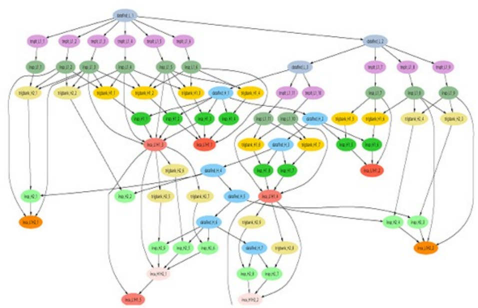
- Grafo de estados representando o espaço de busca



Ordenação topológica

- Consiste em encontrar uma ordem linear para os vértices de Grafos Acíclicos Dirigidos (GADs), respeitando a hierarquia imposta pelas arestas.





Ordenação topológica

Ordenação-Topológica(G)

1. Chame DFS(G) para computar os tempos de fechamento f(v) para cada vértice v;

2. Assim que cada vértice for encerrado, insira-o no início de uma lista encadeada;

3.Retorne a lista de vértices.

Exemplo: prof Paulo vai para a nossa formatura!

- Vamos ajudá-lo a se vestir, para isso, precisamos definir quais peças de roupa ele vai usar:
  - paletó
  - calças
  - cinto
  - cuecas
  - meias
  - camisa
  - gravata
  - relógio
  - sapatos