Algoritmos elementares em grafos

Prof. Dr. Wesin Ribeiro

Neste capítulo

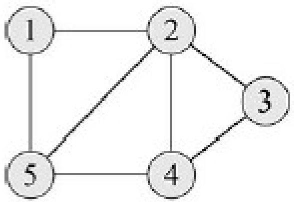
- Introdução
- Representações de grafos
- Busca em largura
- Busca em profundidade
- Ordenação topológica
- Componentes fortemente conexas
- Revisão

Introdução

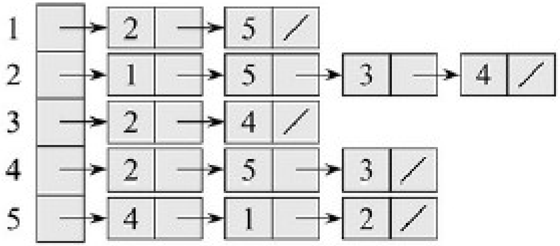
Nossa caixa de ferramentas está cheia de opções que nos ajudam a solucionar problemas no mundo da computação. Este capítulo apresenta as várias peculiaridades de uma das ferramentas mais utilizadas: os grafos. Um típico exemplo de grafo é uma rede de rotas e cidades de um país. Entretanto, as vezes é um grafo está oculto na definição do problema e pode ser difícil detectá-lo.

Grafos não dirigidos

Se G é um grafo não dirigido, a soma dos comprimentos de todas as listas de adjacências é 2|E|, já que, se (u, v) é uma aresta não dirigida, então u aparece na lista de adjacências de v e vice-versa.



(a)



(b)

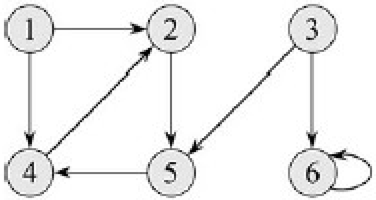
	1	2	3	4	5
1	0	1	0	0	1
2	1	0	1	1	1
3	0	1	0	1	0
4	0	1	1	0	1
5	1	1	0	1	0

(c)

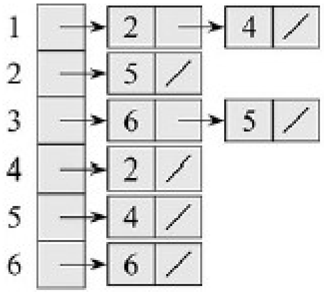
listas de adjacências *matriz de adjacências*

Grafos dirigidos

Se *G é um grafo dirigido, a soma dos* comprimentos de todas as listas de adjacências é |*E*|*, já que a aresta (u, v) pertence a Adj[u].*



(a)



(b)

	1	2	3	4	5	6
1	0	1	0	1	0	0
2	0	0	0	0	1	0
3	0	0	0	0	1	1
4	0	1	0	0	0	0
5	0	0	0	1	0	0
6	0	0	0	0	0	1

(c)

$$a_{ij} = \begin{cases} 1 & \text{if } (i, j) \in E , \\ 0 & \text{otherwise} . \end{cases}$$

Busca em largura (BFS)

A busca em largura é um dos algoritmos mais simples para executar busca em um grafo e é o arquétipo de muitos algoritmos de grafos importantes.

- Partindo de um vértice inicial, explora todos os vértices adjacentes.
- Para cada vértice vizinho, repete o processo para os vértices inexplorados.
- Utiliza a estrutura fila para rastrear os vértices a serem visitados.
- Achar componentes conectados
- Achar menor caminho entre dois vértices
- Testar bipartição dos grafos

Busca em largura

Dado um grafo G = (V, E) e um vértice fonte s, a busca em largura explora sistematicamente as arestas de G para “descobrir” cada vértice que pode ser alcançado a partir de s.

- Vértices brancos, cinzentos e pretos
- Vértices descobertos não são brancos
- Todos os vértices adjacentes a um vértice preto são descobertos
- Vértices cinzentos podem ter alguns vizinhos brancos.

Ilustração do algoritmo BFS

Detalhe na utilização da fila Q

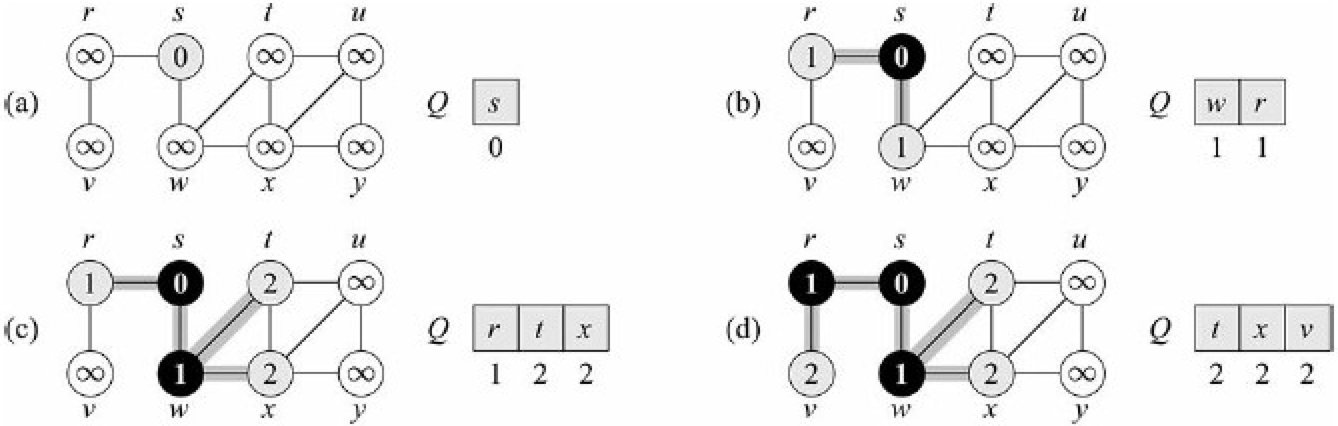
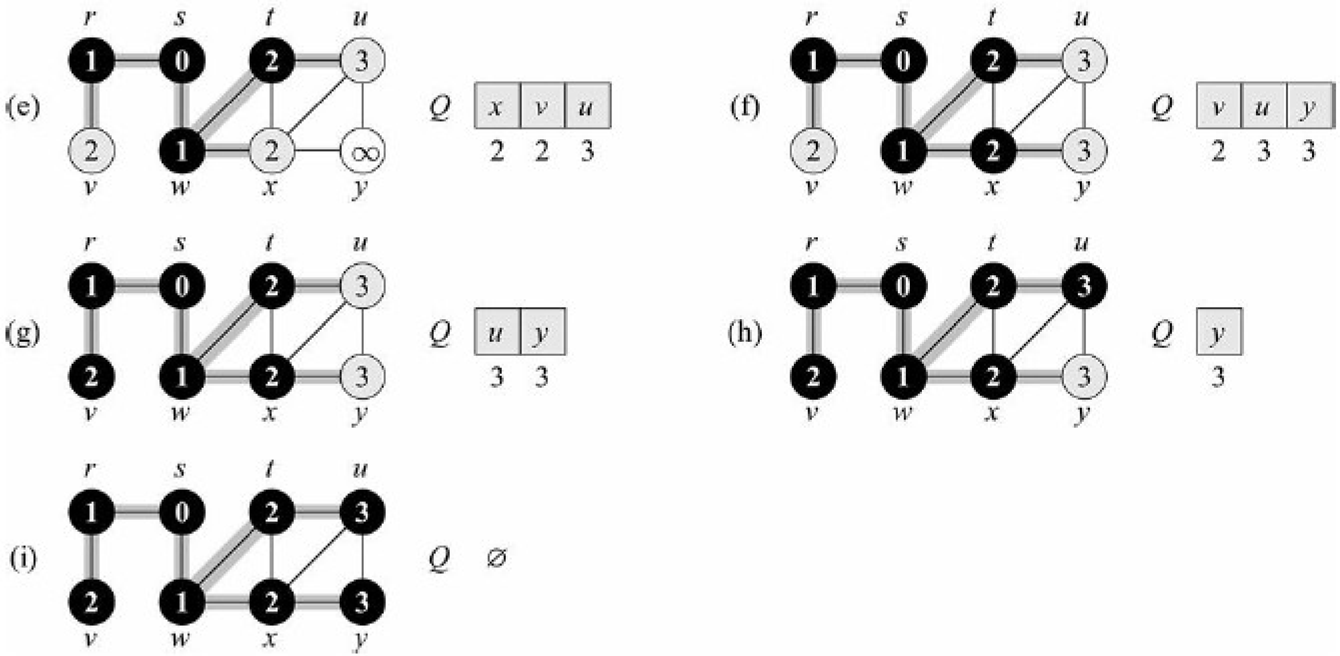


Ilustração do algoritmo BFS

Quando todos os vértices são pretos, não há mais elementos em Q.



Algoritmo BFS

```
BFS(G, s)
for each vertex u ∈ G.V − {s}
    u.color = WHITE
    u.d = ∞
    u.π = NIL
s.color = GRAY
s.d = 0
s.π = NIL
Q = ∅
ENQUEUE(Q, s)
while Q ≠ ∅
    u = DEQUEUE(Q)
    for each v ∈ G.Adj[u]
        if v.color == WHITE
            v.color = GRAY
            v.d = u.d + 1
            v.π = u
            ENQUEUE(Q, v)
    u.color = BLACK
```

Busca em profundidade (DFS)

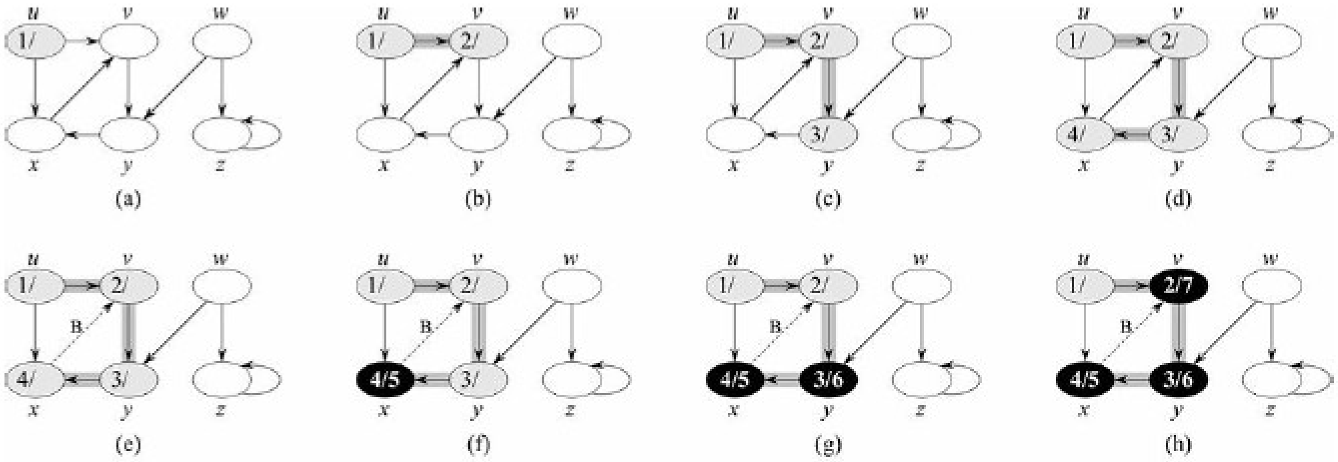
A estratégia seguida pela busca em profundidade é, como seu nome sugere, buscar “mais fundo” no grafo, sempre que possível.

- Partindo de um vértice inicial v
- Explora o máximo possível cada um dos ramos antes de retroceder
- Utiliza recursão para rastrear os vértices a serem visitados
- Encontrar componentes conectados e fortemente conectados
- Ordenação topológica de um grafo
- Resolver quebra-cabeças (labirinto)

Ilustração do algoritmo DFS

Detalhe na cor do vértice e na marca temporal descoberta/término

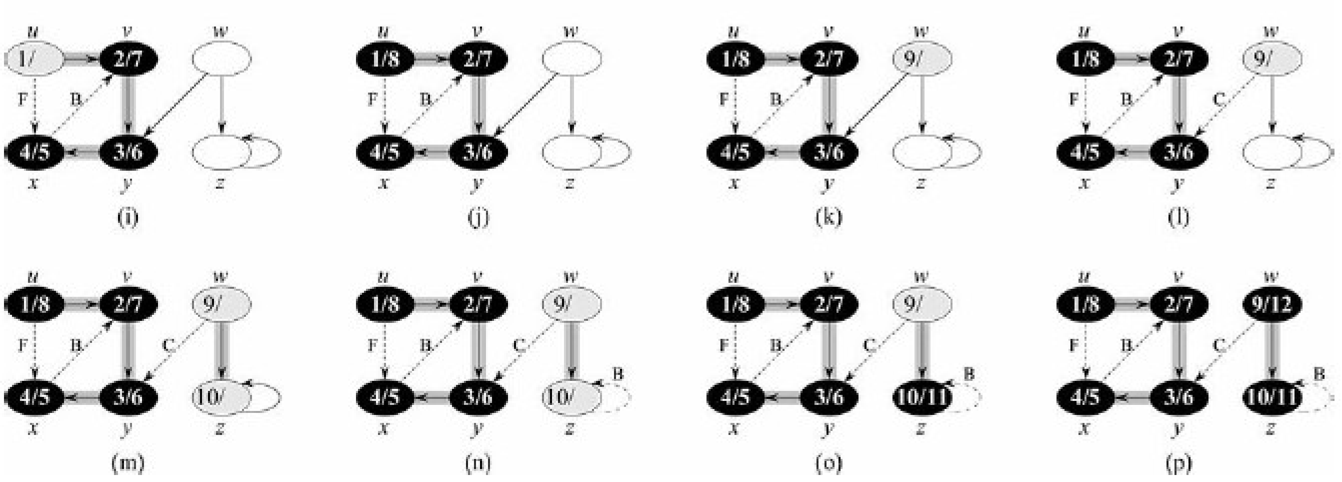
A medida que as arestas são exploradas, elas são marcadas como arestas de árvores(sombreadas) ou não (pontilhadas)



Tipos de arestas

um grafo dirigido é acíclico se e somente se uma busca em profundidade não produz nenhuma aresta B

As arestas pontilhadas podem ser de retorno (B), cruzadas (C) e diretas (F). Os números representam o tempo de descoberta/tempo de término.



Algoritmo DFS

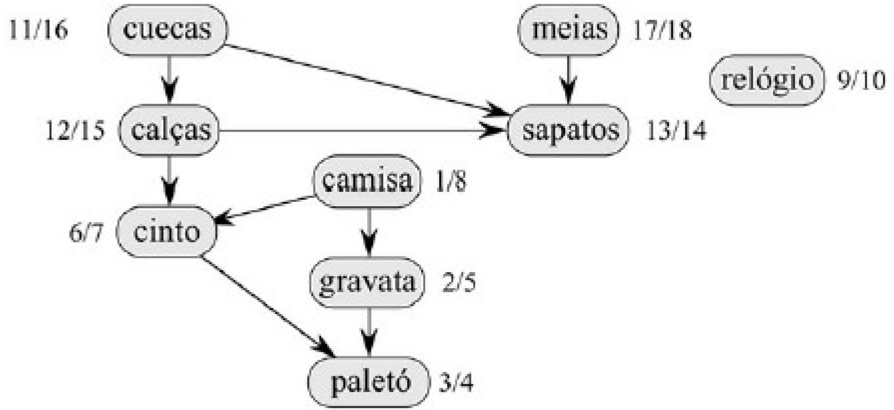
```
DFS(G)
for each vertex u ∈ G.V
    u.color = WHITE
    u.π = NIL
time = 0
for each vertex u ∈ G.V
    if u.color == WHITE
        DFS-VISIT(G, u)
```

```
DFS-VISIT(G, u)
time = time + 1
u.d = time
u.color = GRAY
for each v ∈ G.Adj[u]
    if v.color == WHITE
        v.π = u
        DFS-VISIT(G, v)
u.color = BLACK
time = time + 1
u.f = time
```

Ordenação topológica

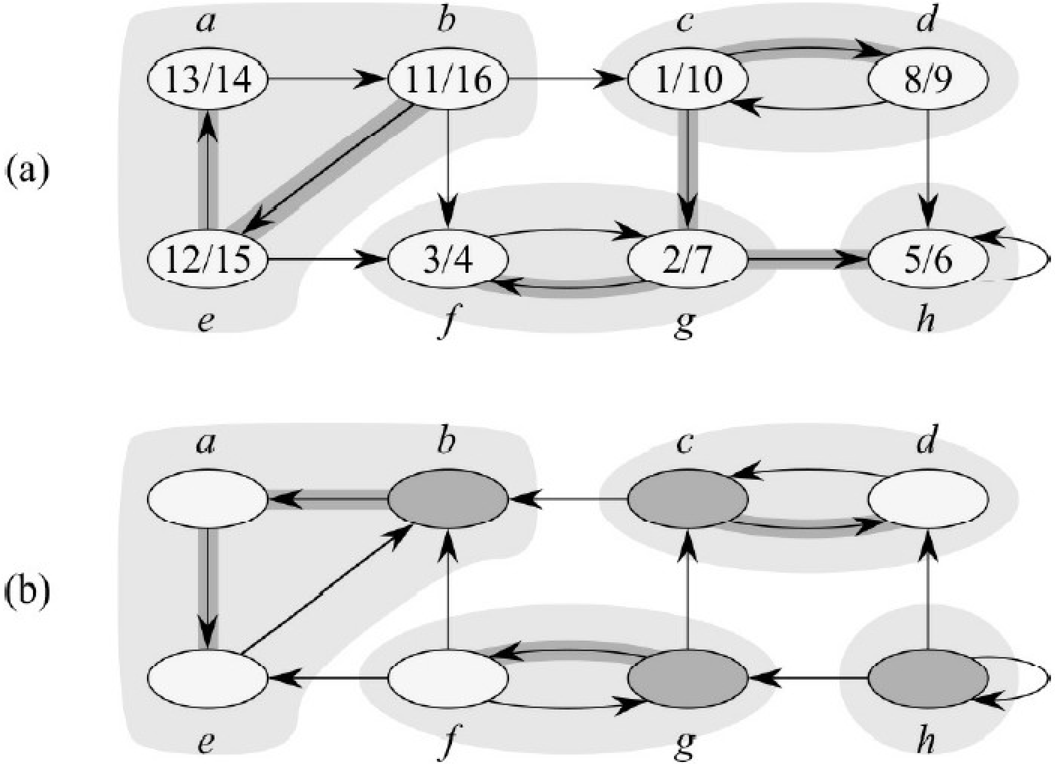
A busca em profundidade pode ser usada para encontrar a ordenação topológica em grafos acíclicos dirigidos (gad).

Podemos ver uma ordenação topológica de um grafo como uma ordenação de seus vértices ao longo de uma linha horizontal de modo tal que todas as arestas dirigidas vão da esquerda para a direita



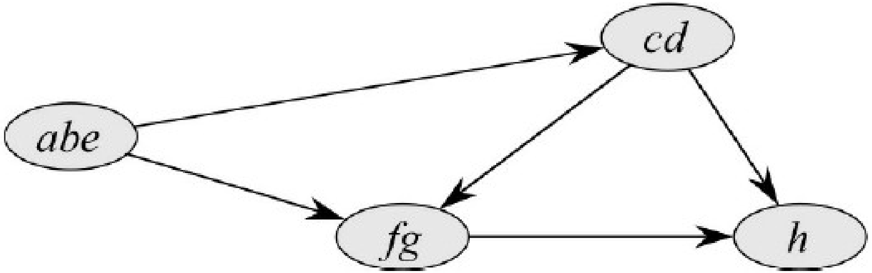
Componentes fortemente conexas

Uma aplicação clássica da busca em profundidade é a decomposição do grafo em suas componentes fortemente conexas. uma componente fortemente conexa de um grafo dirigido G = (V, E) é um conjunto máximo de vértices C ⊆ V tal que, para todo par de vértices u e v em C, temos u v e v u; isto é, u pode ser alcançado a partir do vértice v e vice-versa.



Componentes fortemente conexas

O grafo acíclico de componentes G obtido pela contração de cada componente fortemente conexa de G, de modo que apenas um único vértice permaneça em cada componente.



Exercícios

Mostre os valores de d e p que resultam da execução da busca em largura no grafo dirigido da Figura 2, usando o vértice 3 como fonte.

Mostre os valores d e p que resultam da execução da busca em largura no grafo não dirigido da Figura 3, usando o vértice u como fonte.

Reescreva o procedimento DFS utilizando uma pilha para eliminar recursão.

Revisão

Atenção, chegou a hora da revisão.

- Grafos são estruturas compostas por vértices e arestas. Podem ser dirigidos ou não dirigidos, cíclicos ou não cíclicos
- Grafos podem ser representados por uma lista ou matriz de adjacências
- Busca em largura utiliza um fila para percorrer os vértices mais próximos de um vértice inicial antes de retroceder
- Busca em profundidade utiliza recursão ou uma pilha para percorrer toda uma ramificação de vértices antes de retroceder
- Ordenação topológica é um grafo dirigido organizado na ordem decrescente da marca temporal obtida após uma busca em profundidade
- Componentes fortemente conexos são conjuntos de vértices interligados de tal maneira que podem ser substituídos por um só vértice que representa a conexão dos vértices envolvidos.