O estudo utilizando apenas este material **não é suficiente** para o entendimento do conteúdo. Recomendamos a leitura das referências no final deste material e a resolução (por parte do aluno) de todos os exercícios indicados.

Grafos

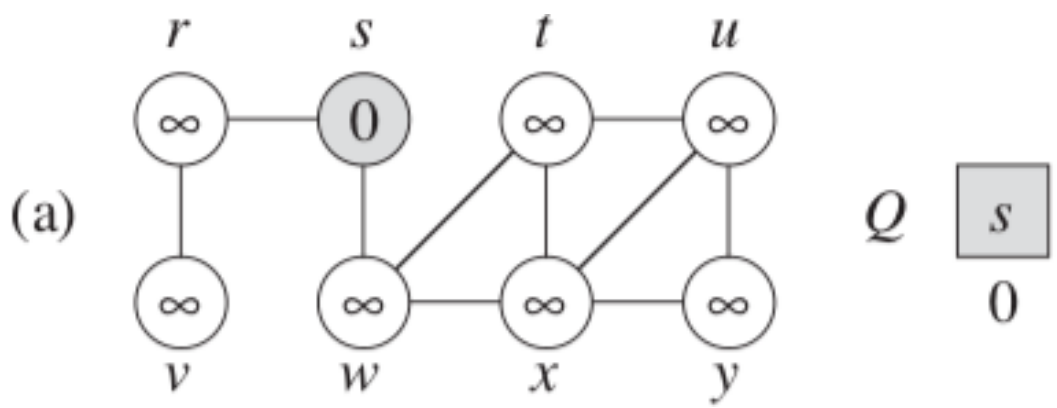
Busca em largura

Conteúdo

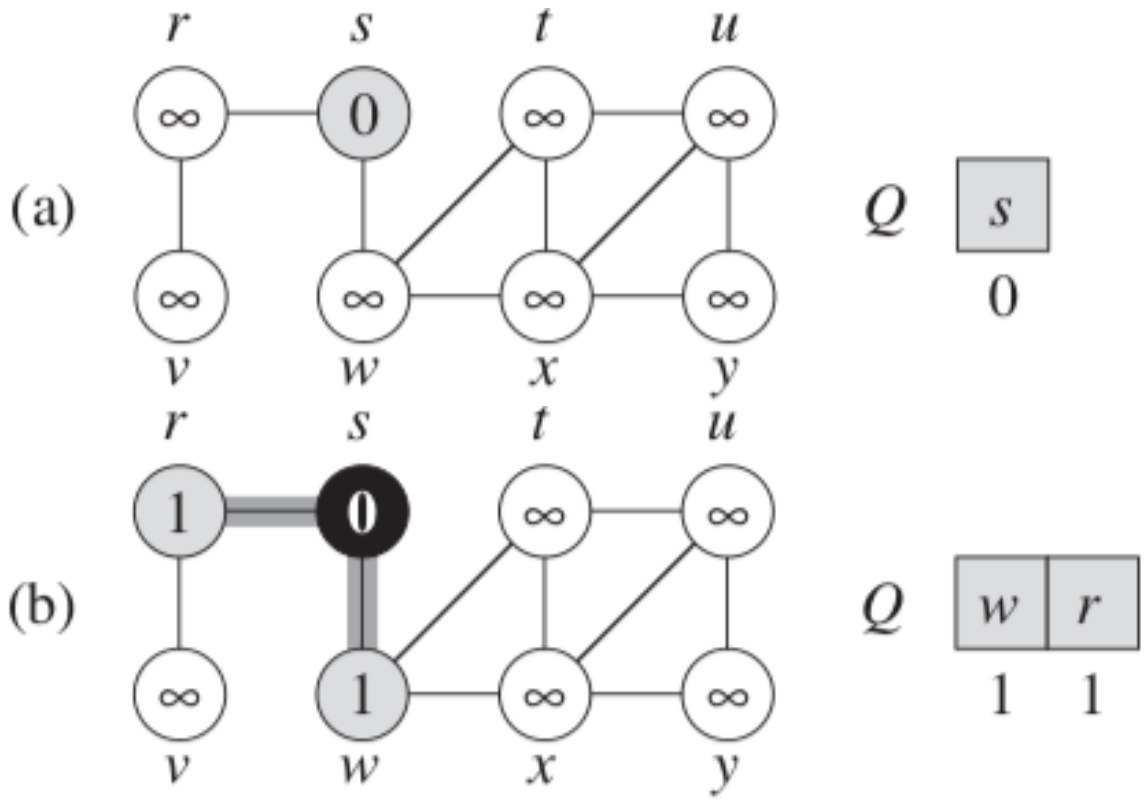
Introdução

- Dados um $G = (V, E)$ e um vértice de origem s, a busca em largura explora sistematicamente as arestas de G até descobrir cada vértice acessível a partir de s
- O algoritmo calcula a distância (menor número de arestas) deste s até todos os vértices acessíveis a partir de s
- O algoritmo produz uma árvore primeiro em extensão
- Recebe este nome porque expande a fronteira entre vértices descobertos e não descobertos uniformemente ao longo da extensão da fronteira. Descobre todos os vértices de distância k de s antes de descobrir quaisquer vértices de distância $k + 1$

Exemplo de execução

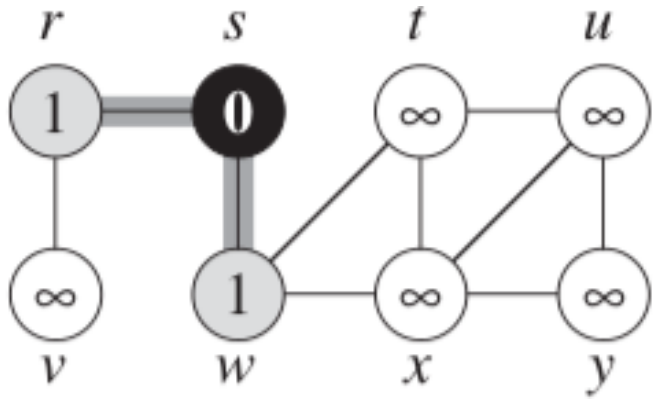


Exemplo de execução



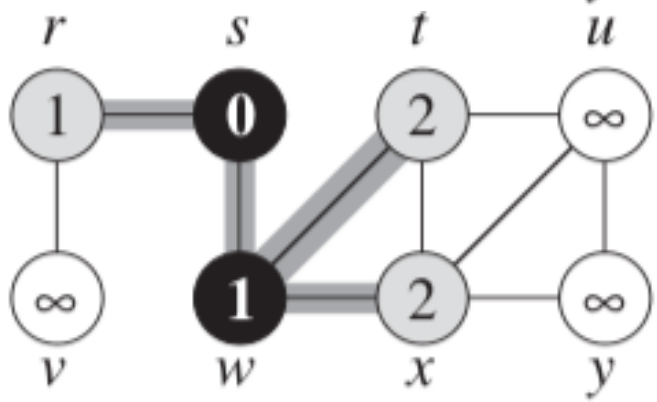
Exemplo de execução

(b)



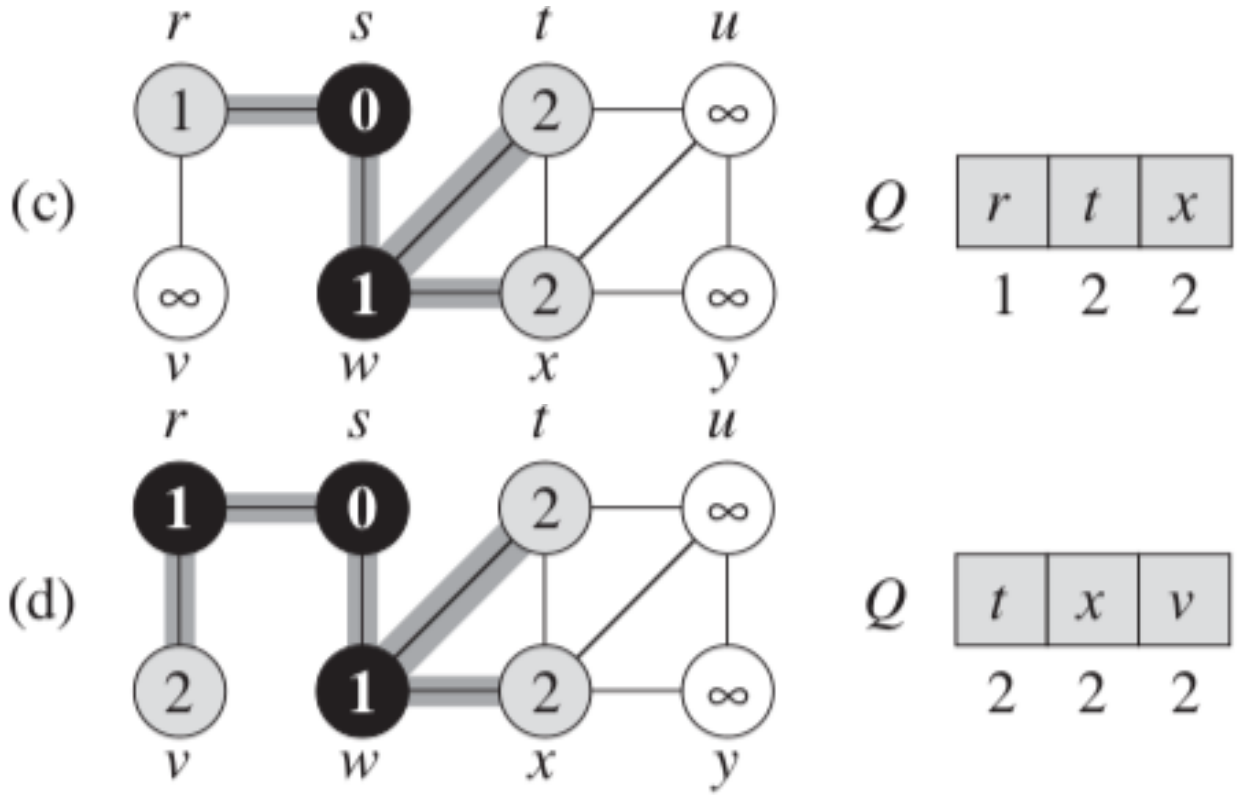
Q | w | r |
1 1

(c)

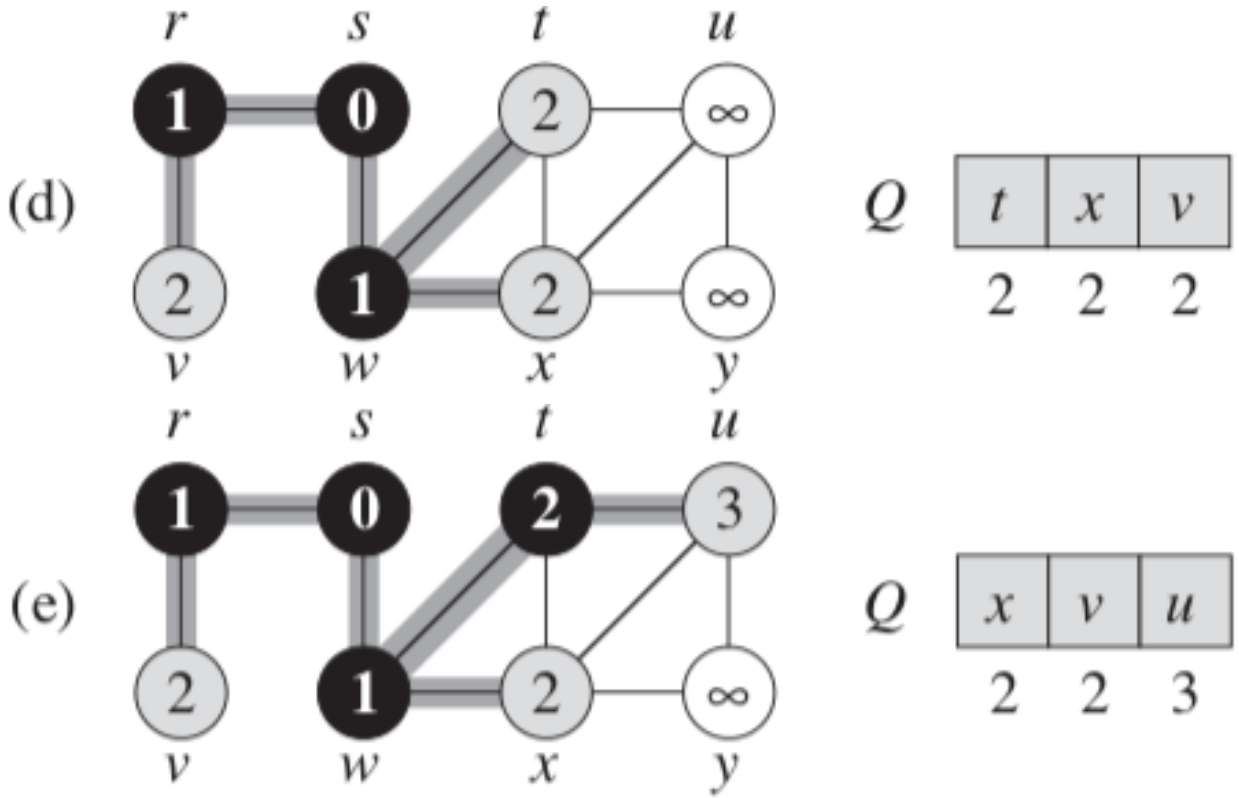


Q | r | t | x |
1 2 2

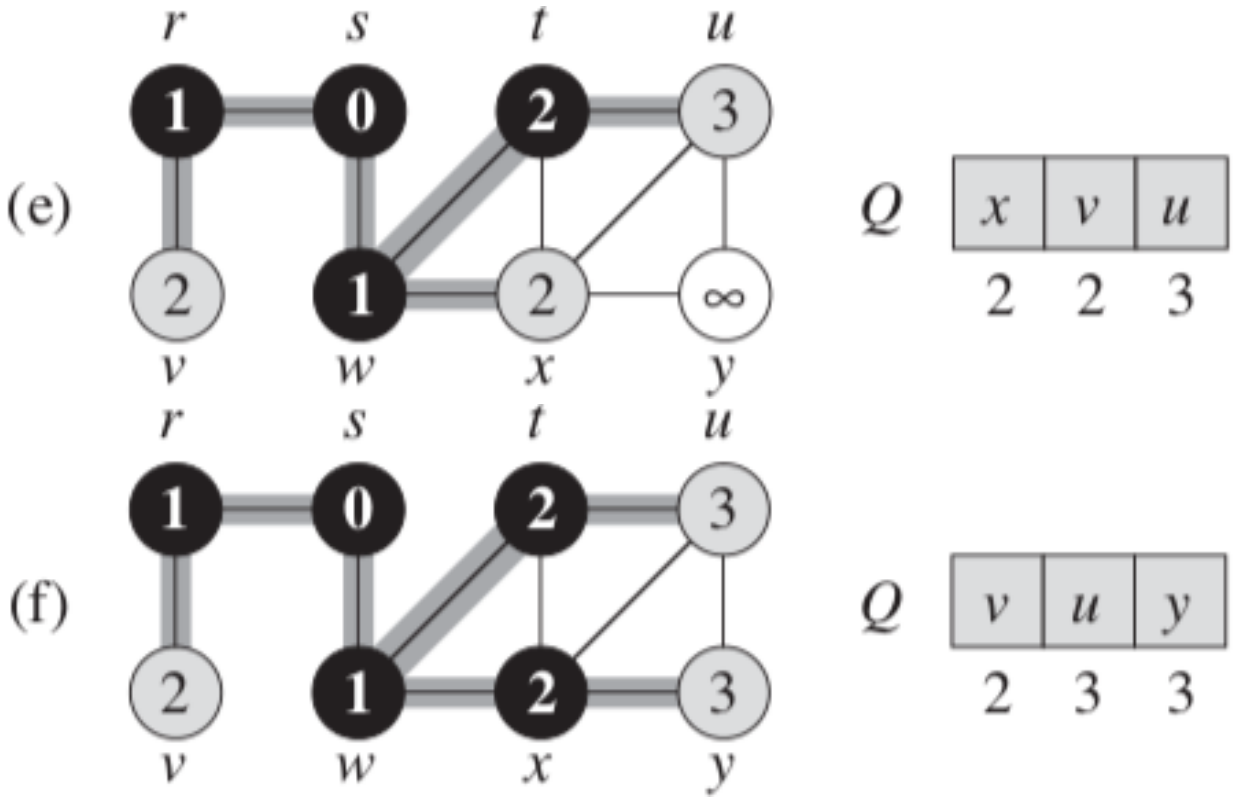
Exemplo de execução



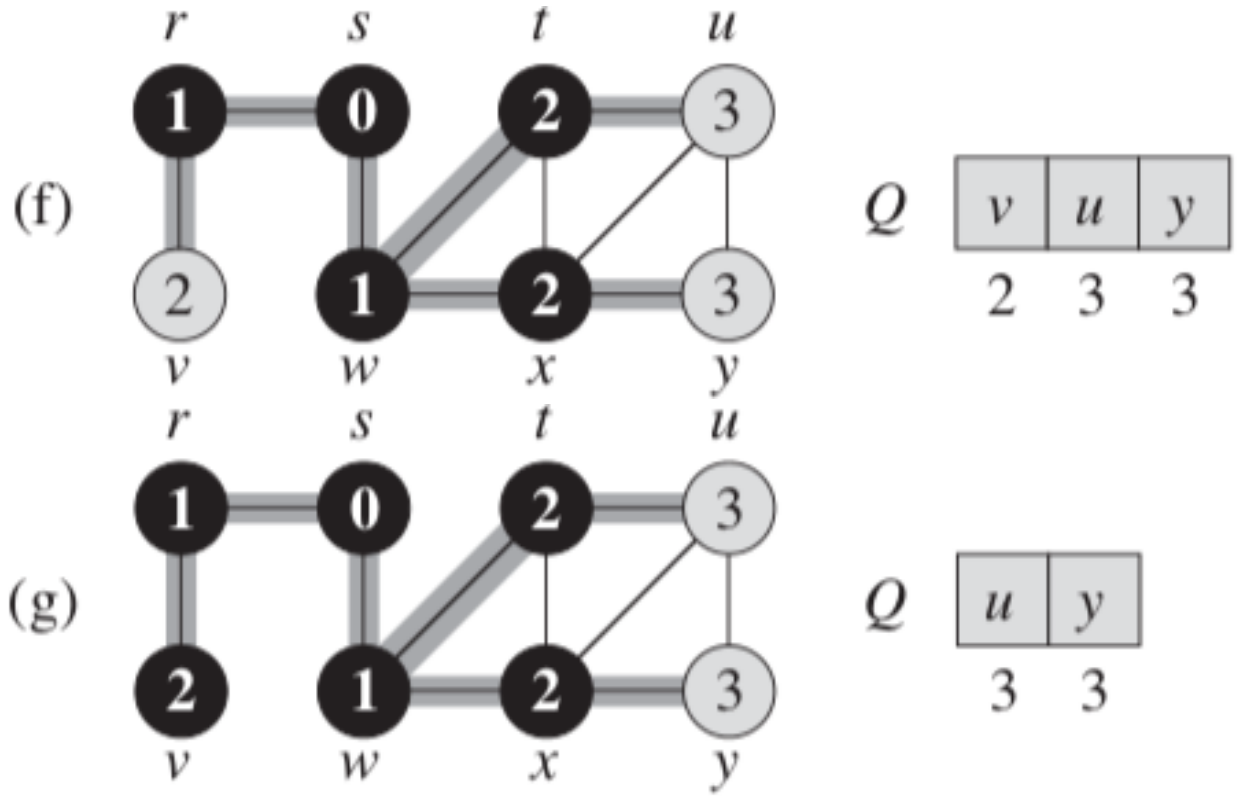
Exemplo de execução



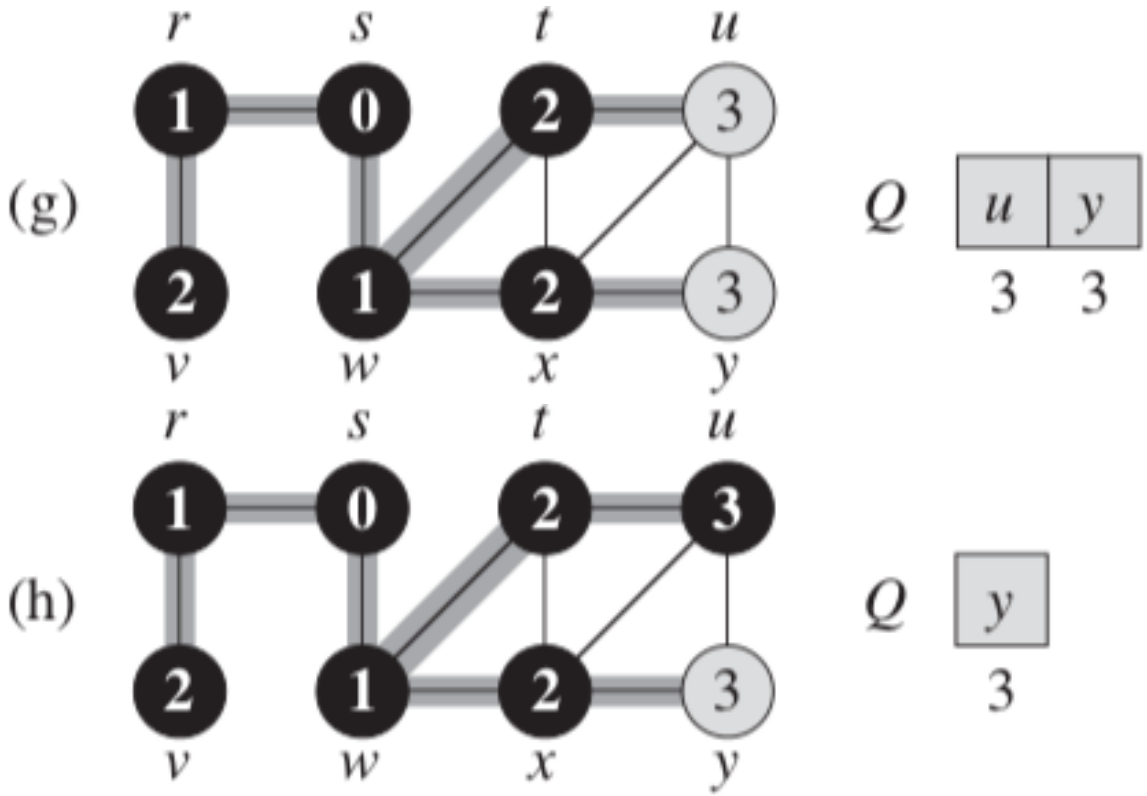
Exemplo de execução



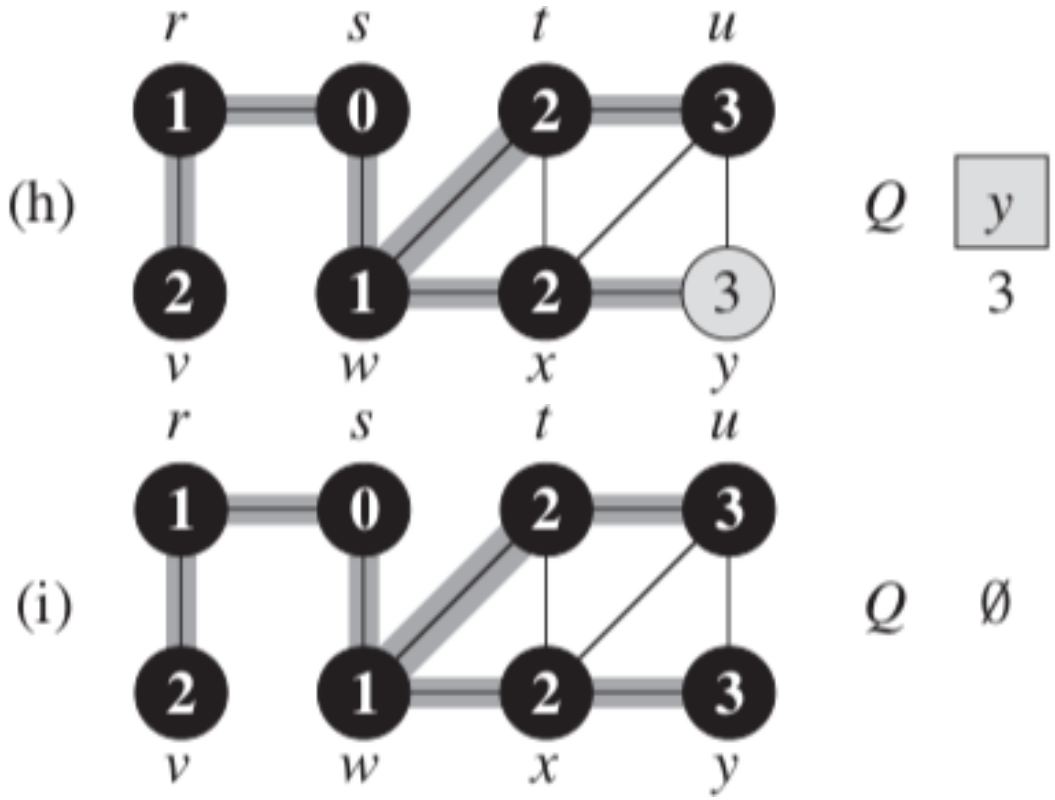
Exemplo de execução



Exemplo de execução



Exemplo de execução



Procedimento bfs

```
bfs(G, s)
 1 for cada vértice u em G.V - {s}
 2   u.d = infinito
 3   u.pai = nil
 4   u.cor = branco
 5 s.d = 0
 6 s.pai = nil
 7 s.cor = cinza
 8 Q = {}
 9 Q.add(s)
10 while Q != {}
11   u = Q.remove()
12   for cada vértice v em G.adj[u]
13     if v.cor == branco
14       v.d = u.d + 1
15       v.pai = u
16       v.cor = cinza
17       Q.add(v)
18   u.cor = preto
```

Análise do tempo de execução do `bfs`

- Análise agregada
- O teste da linha 13 garante que cada vértice é colocado na fila no máximo uma vez, e portanto, é retirado da fila no máximo uma vez
- As operações de colocar e retirar da fila demoram $O(1)$, portanto, o tempo total das operações com filas é $O(V)$
- A lista de adjacência de cada vértice é examinada apenas quando o vértice é retirado da fila, portanto, no máximo uma vez
- Como a soma dos comprimentos das listas de adjacências é $\Theta(E)$, o tempo para percorrer todas as listas é o máximo $O(E)$
- O tempo de inicialização é $O(V)$
- Tempo total de execução do `bfs` é $O(V + E)$

Árvore primeiro na extensão

- `bfs` constrói uma árvore primeiro na extensão
- A árvore é definida pelo campo pai (π) em cada vértice
- Para um grafo $G = (V, E)$ e um vértice de origem s, definimos o **subgrafo predecessor** de G como $G_\pi = (V_\pi, E_\pi)$ onde
  - $V_\pi = \{v \in V : v.\pi \neq \text{NIL}\} \cup \{s\}$
  - $E_\pi = \{(v.\pi, v) : v \in V_\pi - \{s\}\}$
- O subgrafo predecessor G_π é uma árvore primeiro na extensão
  - V_π consiste nos vértices acessíveis a partir de s
  - Para todo $v \in V_\pi$, existe um caminho único simples desde s até v em G_π, que também é o caminho mais curto de s até v em G
- Uma árvore primeiro na extensão é de fato uma árvore, pois é conexa e $|E_\pi| = |V_\pi| - 1$

Árvores primeiro na extensão

```
imprimir-caminho(G, s, v)
1  if v == s
2    imprimir s
3  else if v.pai == nil
4    imprimir "nenhum caminho existente de" s "para" v
5  else
6    imprimir-caminho(G, s, v.pai)
7    imprimir v
```

- Executado em tempo linear no número de vértices no caminho impresso, pois cada chamada recursiva é feita para um caminho com um vértice menor que o atual

Exercícios

- Os números dos exercícios referem-se a 3° edição
- Entre parênteses está o número do exercício correspondente na 2° edição
- Se não existe informação entre parênteses, significa que o exercício é o mesmo nas duas edições
- 22.2-1, 22.2-2, 22.2-3 (não presente), 22.2-4 (22.2-3), 22.2-5 (22.2-4), 22.2-6 (22.2-5), 22.2-7 (22.2-6), 22.2-9 (22.2-8)

Exercícios

22.2-1 Mostre os valores de d e π que resultam da execução da busca em largura sobre o grafo orientado da figura 22.2(a), usando o vértice 3 como origem.

22.2-2 Mostre os valores de d e π que resultam da execução da busca em largura sobre o grafo não-orientado da figura 22.3, usando o vértice u como origem.

22.2-3 Mostre que usar apenas um bit para armazenar a cor de cada vértice é suficiente argumento que o procedimento `bfs` irá produzir o mesmo resultado se as linhas 5 e 14 forem removidas.

22.2-4 Qual o tempo de execução de `bfs` se o grafo de entrada é representado pos uma matriz de adjacências e o algoritmo é modificado para manipular essa forma de entrada?

Exercícios

22.2-5 Argumente que na busca em largura, o valor `u.d` atribuído ao vértice `u` é independente da ordem que os vértices aparecem na lista de adjacência. Usando a figura 22.3 como exemplo, mostre que a busca em largura computada pelo `bfs` pode depender da ordem dos vértices na lista de adjacências.

22.2-6 Forneça um exemplo de um grafo orientado $G = (V, E)$, um vértice de origem $s \in V$ e um conjunto de arestas de árvore $E_\pi \subseteq E$ tal que, para cada vértice $v \in V$, o caminho único em (V, E_π) de s até v é um caminho mais curto em G, ainda que o conjunto de arestas E_π não possa ser produzido pela execução de `bfs` sobre G, não importando o modo como os vértices estão ordenados em cada lista de adjacências.

Exercícios

22.2-7 Existem dois tipos de lutadores profissionais: "bons sujeitos" e "maus sujeitos". Entre qualquer par de lutadores profissionais pode ou não haver uma rivalidade. Suponha que temos n lutadores profissionais e temos uma lista de r pares de lutadores para os quais existem rivalidades. Dê um algoritmo de tempo $O(n + r)$ que determine se é possível designar alguns dos lutadores como bons sujeitos e os restantes como maus sujeitos, de tal forma que a rivalidade ocorra em cada caso entre um bom sujeito e um mau sujeito. Se for possível realizar tal designação, seu algoritmo deve produzi-la.

Exercícios

22.2-9 Seja $G = (V, E)$ um grafo conexo não orientado. Forneça um algoritmo de tempo $O(V + E)$ para calcular um caminho em G que percorra cada aresta de E exatamente uma vez em cada sentido. Descreva como você pode encontrar a saída de uma labirinto se receber uma grande provisão de moedas de centavos.

Referências

- Thomas H. Cormen et al. Introduction to Algorithms. 3rd edition. Capítulo 22.2.