2018 /1

# Projeto e Análise de Algoritmos

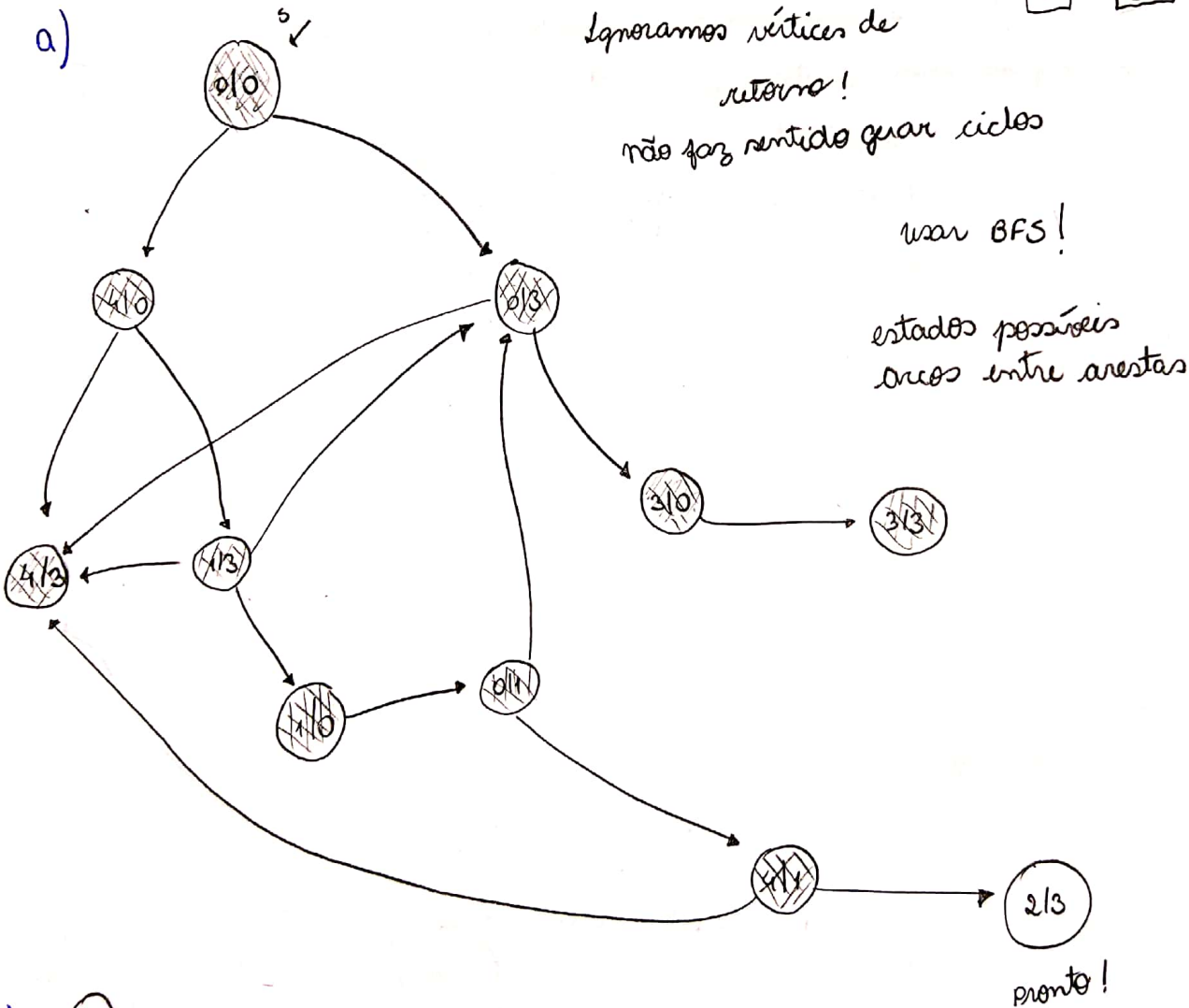
## Teoria dos Grafos

## Prova 1 (12 pontos)

**Questão 1 (5 pontos):**
Você tem duas garrafas, uma de 4 litros de capacidade e outra de 3 litros de capacidade. Também tem uma fonte infinita de agua. Como as garrafas não tem nenhuma marcação a única forma de saber quanta agua é transferida a uma garrafa é encher ou esvaziar uma garrafa. Pretende-se colocar 2 litros de agua na garrafa de 4 litros.
a) (4 pontos) Modele o problema usando um grafo e indique o algoritmo que deve ser executado para resolver o problema.
b) (1 ponto) Resolva o problema.

4L 3L

a)




Ignoramos vértices de retorno!
não faz sentido gerar ciclos

usar BFS!

estados possíveis
arcos entre arestas

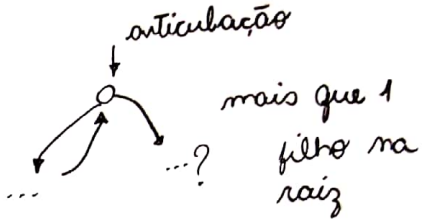
pronto!

b) s: (0,0)
Q = S
lista prioridade = (0,0), (0,3), (4,0), (4,3), (3,0), (1,3), (3,3), (1,0), (0,1), (4,1), (2,3) vazio
lista explorados (0,0), (0,3), (4,0), (4,3), (3,0), (1,3), (3,3), (1,0), (0,1), (4,1), (2,3) (11 iterações)

**Questão 2 (5 pontos):**
Um ponto de articulação é um vértice cuja remoção aumenta o número de componentes conexas do grafo. Desenvolva um algoritmo linear (de complexidade não superior a O(n+m)) para determinar todos os pontos de articulação de um grafo.
Dica: Pode apenas indicar modificações aos algoritmos vistos em sala de aula.

Usar DFS:

Se retornar para raiz e entrgar novo vértice
raiz = vertice de articulação

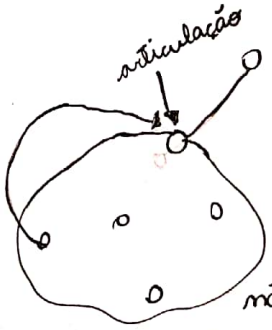



Para cada descendente:

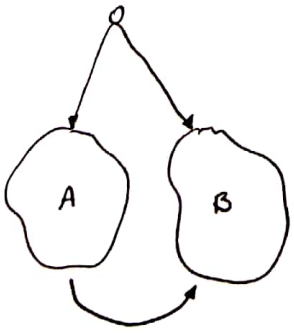
Devolver o menor tempo de abertura de um vértice visto pelos descendentes

Se o vértice não é folha e nenhum descendente vê algum vértice com tempo de abertura menor do que ele, o vértice é vértice de articulação.
(a menos que seja raiz).

Ponto de articulação = DFS




não emerge nenhum outro cinza com tempo de abertura menor que ele




A não pode ter aresta para o pois o DFS entraria em contradição

**Questão 3 (4 pontos):**
Pretende-se computar caminhos mínimos desde um determinado vértice a todos os outros vértices de um grafo orientado. Em cada contexto abaixo decida entre o algoritmo de Dijkstra e Bellman-Ford. Em caso de que os dois algoritmos se apliquem escolha aquele de menor complexidade considerando a implementação de Dijkstra com heap binário e Bellman-Ford com as melhoras que ache razoáveis. Justifique.

Não assuma nada a mais do que está indicado em cada contexto.

a) arcos com custos positivos.
b) alguns arcos podem ter custo negativo.
c) grafo esparso com |E| limitado a 4x|V| e custos positivos nos arcos
d) arcos com custos positivos e grafo denso no qual todos os caminhos mínimos existem e tem no máximo 5 arcos.

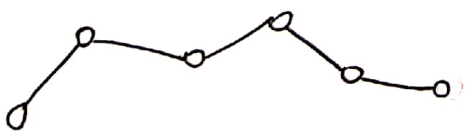
a) Dijkstra com heap normal fica $O(\log v)$ já que é o menor e o grafo tem arcos positivos.

b) Bellman-Ford devido aos arcos negativos

c) Dijkstra melhor! (Dijkstra $V \log V$, Bellman Ford $V^2$)

d) Bellman-Ford com 5 iterações. não será necessário usar n-1 vezes e a complexidade baixa para $O(n)$ ou $O(V)$.
Quando chegamos a uma iteração que não modifica mais o grafo, as próximas também não farão nada. $O(5E)$

na 6ª iteração nada mudaria se houvesse arco negativo
seria 6E $O(E)$



6V
→ 5A (V-1)
em todos os caminhos