Análise de Algoritmos e Estruturas de Dados
Centro de Matemática, Computação e Cognição
Universidade Federal do ABC
Profa. Carla Negri Lintzmayer

Lista 4: Introdução a grafos

Instruções importantes

(1) Em qualquer exercício que peça para você fornecer um algoritmo/solução para um problema, a menos que explicitamente dito o contrário, **você deve**, na seguinte ordem:

- descrever em palavras qual é a ideia do seu algoritmo (**obrigatório**);
- escrever um pseudocódigo (**obrigatório**);
- provar ou fornecer um argumento intuitivo para sua corretude (**obrigatório**), a depender do que o exercício pede;
- analisar o tempo de execução de pior caso do seu algoritmo (**obrigatório**).

O não cumprimento dos itens acima implica que o exercício está incorreto e portanto será desconsiderado.
Na dúvida, procure atendimento!

(2) Você pode utilizar qualquer algoritmo visto em aula sem reescrevê-lo ou provar sua corretude novamente, mesmo que o algoritmo necessite de alguma pequena alteração.

- Descrever clara e sucintamente o que o algoritmo recebe, o que ele devolve e qual o seu consumo de tempo.
- Se houver alterações, descreva-as, indicando, por exemplo, quais linhas estão sendo alteradas/acrescentadas.

1. Dado um grafo G, o *quadrado* de G é o grafo G^2 tal que $V(G^2) = V(G)$ e para cada par de vértices $x, y \in V(G)$, a aresta xy vai existir em $E(G^2)$ se e somente se temos $xy \in E(G)$ ou se existe algum outro vértice $w \in V(G)$ tal que as arestas xw e wy existem em $E(G)$. Em outras palavras, em G^2 existe aresta entre todo par de vértices cuja distância em número de arestas é 1 ou 2 em G.

   Descreva um algoritmo que recebe um grafo G qualquer e devolve G^2. Tenha certeza que o grafo G^2 não possui arcos paralelos. Analise o tempo de execução se o algoritmo for implementado usando listas de adjacências e se for usando matriz de adjacências. Não é preciso apresentar uma demonstração de corretude.

   Dica: Seu pseudocódigo pode começar com os comandos "`crie` G^2" e "`faça` $V(G^2) = V(G)$". Veja que podemos considerar que essas operações levam tempo $\Theta(1)$ porque estamos, na verdade, alocando espaço para a matriz de adjacências ou para a lista de adjacências. O restante do algoritmo deve inserir as arestas em G^2 corretamente.

2. O digrafo transposto de um digrafo D é o digrafo D^T em que $E(D^T) = \{(v, u) \colon (u, v) \in E(D)\}$. Assim, D^T é D com todas as arestas invertidas. Descreva um algoritmo que recebe um digrafo D qualquer e devolve D^T. Analise o tempo de execução se o algoritmo for implementado usando listas de adjacências e se for usando matriz de adjacências. Não é preciso apresentar uma demonstração de corretude.

   Dica: Seu pseudocódigo pode começar com os comandos "`crie` D^T" e "`faça` $V(D^T) = V(D)$". Veja que podemos considerar que essas operações levam tempo $\Theta(1)$ porque estamos, na verdade, alocando espaço para a matriz de adjacências ou para a lista de adjacências. O restante do algoritmo deve inserir os arcos em D^T corretamente.

3. Considere o grafo G definido por $V(G) = \{a, b, c, d, e, f, g, h, i, j, k, \ell\}$ e $E(G) = \{ad, de, ea, ba, bf, fg, gb, cb, hi, ic, jk, k\ell, j\ell\}$. Execute a busca em largura sobre G a partir do vértice e, calculando as distâncias para todos os outros vértices. Apresente os vetores `visitado`, `predecessor` e `distancia`, que são indexados pelos vértices, já preenchidos. Desenhe G e a árvore da busca em largura.

4. Considere o grafo G definido por $V(G) = \{a, b, c, d, e, f, g, h, i, j, k, \ell\}$ e $E(G) = \{ad, de, ea, ba, bf, fg, gb, cb, hi, ic, jk, k\ell, j\ell\}$. Execute a busca em profundidade sobre G a partir do vértice e. Apresente os vetores `visitado` e `predecessor`, que são indexados pelos vértices, já preenchidos. Desenhe G e a árvore da busca em profundidade.

5. Forneça um algoritmo que recebe um grafo G e devolve `sim` caso exista algum ciclo em G, ou devolve `não` caso contrário. Faça isso modificando o algoritmo de busca em profundidade. Analise o tempo de execução se o algoritmo for implementado usando listas de adjacências e se for usando matriz de adjacências. Não é preciso apresentar uma demonstração formal de corretude, apenas acrescente alguma intuição durante a descrição da ideia do algoritmo.

6. Seja D um digrafo. O digrafo de componentes fortemente conexas de D é o grafo D^C tal que existe um vértice em $V(D^C)$ para cada componente fortemente conexa de D e existe um arco $uv \in E(D^C)$ se e somente se existe um arco $xy \in E(D)$ onde x pertence à componente representada por u e y pertence à componente representada por v. Descreva um algoritmo que recebe D e devolve D^C e execute em tempo $O(|V(D)| + |E(D)|)$. Você pode assumir a existência de um algoritmo Compo-

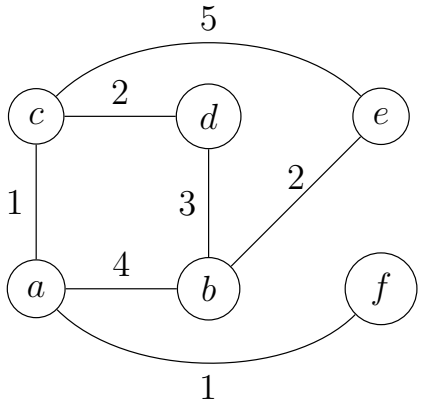
nentesFortementeConexas que recebe D e atualiza os campos $u.\texttt{componente}$ de cada vértice u de forma que, ao final, se $u.\texttt{componente} = v.\texttt{componente}$, então u e v estão na mesma componente fortemente conexa. Tenha certeza que o digrafo D^C não possui arcos paralelos. Não é preciso apresentar uma demonstração de corretude.

7. Escreva um algoritmo que recebe um grafo G e dois vértices s e v e devolve a sequência de vértices de um sv-caminho em tempo $O(|V(G)| + |E(G)|)$. Não é preciso apresentar uma demonstração de corretude.

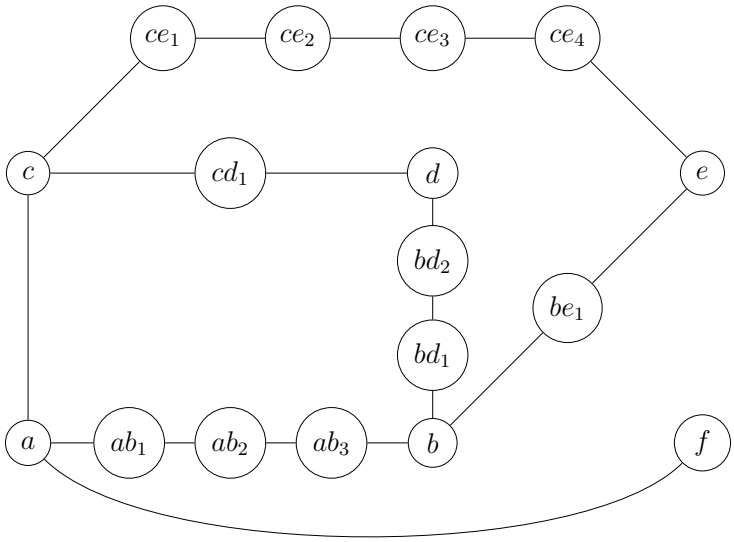
8. Dado um grafo G, o *diâmetro* de G é o valor da maior distância entre quaisquer dois vértices de G. Formalmente, é o número $\max\{dist(u, v) \colon u, v \in V(G)\}$, onde $dist(u, v)$ é a distância entre u e v em G, em número de arestas. Escreva um algoritmo que, dado G, determine o diâmetro de G. Para o tempo de execução, considere as implementações usando listas de adjacências e também matriz de adjacências. Não é preciso apresentar uma demonstração de corretude, mas escreva qual é a invariante do laço principal.

9. Considere a definição de diâmetro dada acima. Escreva um algoritmo que, dada uma árvore T, determine o diâmetro de T. Para o tempo de execução, considere as implementações usando listas de adjacências e também matriz de adjacências.

10. Seja $G = (V, E)$ um grafo e $w \colon E(G) \to \mathbb{Z}_+$ uma função de custo nas arestas tal que $w(e) \geq 1$. Construa o grafo H tal que cada aresta $e \in E(G)$ é substituída por um caminho com $w(e)$ arestas sem custo em H (ou, similarmente, de custo 1). Assim, H possui os mesmos vértices de G e alguns vértices extras. Veja as Figuras 1a e 1b.



(a) Grafo G.



(b) Grafo H.

Figura 1: Exemplos para o exercício 10.

Dado um vértice $s \in V(G)$, qual é o tempo de execução da busca em largura sobre H a partir de s? É possível escrever esse tempo em função de $|V(G)|$ e $|E(G)|$? Para todo $v \in V(G)$, o custo do sv-caminho mínimo em H encontrado pela busca em largura é o mesmo do sv-caminho mínimo em G? Justifique devidamente suas respostas.

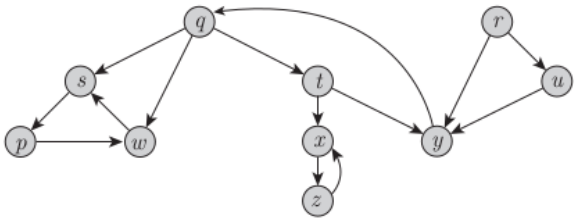
1 Questões retiradas do Enade e Poscomp

QUESTÃO DISCURSIVA 05

A busca primeiro em profundidade é um algoritmo de exploração sistemática em grafos, em que as arestas são exploradas a partir do vértice v mais recentemente descoberto que ainda tem arestas inexploradas saindo dele. Quando todas as arestas de v são exploradas, a busca regressa para explorar as arestas que deixam o vértice a partir do qual v foi descoberto. Esse processo continua até que todos os vértices acessíveis a partir do vértice inicial sejam explorados.

CORMEN, T. H.; LEISERSON, C. E.; RIVEST, R. L.; STEIN, C. **Introduction to algorithms**. 3. ed. Cambridge, Massachusetts: The MIT Press, 2009 (adaptado).

Considere o grafo a seguir.

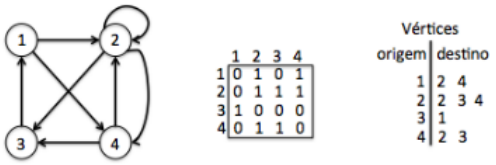


Com base nas informações apresentadas, faça o que se pede nos itens a seguir.

a) Mostre a sequência de vértices descobertos no grafo durante a execução de uma busca em profundidade com controle de estados repetidos. Para isso, utilize o vértice r como inicial. No caso de um vértice explorado ter mais de um vértice adjacente, utilize a ordem alfabética crescente como critério para priorizar a exploração. (valor: 7,0 pontos)

b) Faça uma representação da matriz de adjacências desse grafo, podendo os zeros ser omitidos nessa matriz. (valor: 3,0 pontos)

QUESTÃO 32 – Em relação ao grafo da Figura (a), as Figuras (b) e (c) representam, respectivamente,



(a)

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

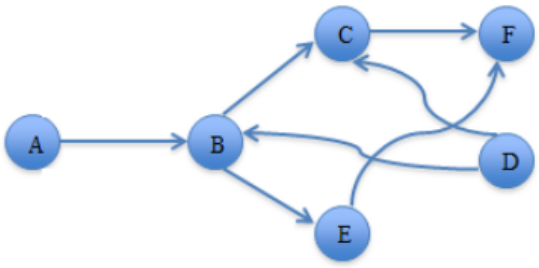
(b)

Vértices	
origem	destino
1	2 4
2	2 3 4
3	1
4	2 3

(c)

A) matriz de arestas e lista de incidências.
B) matriz de adjacências e lista de adjacências.
C) matriz de conexões e lista de arestas.
D) matriz de incidências e lista de vértices.
E) matriz de vértices e lista de conexões.

QUESTÃO 17 – Considere o grafo G abaixo e as afirmações feitas sobre G:



I. O grafo é planar.
II. O menor caminho direcionado medido em número de arcos entre os nós D e F tem comprimento 2.
III. DABCEF representa uma ordenação topológica válida dos nós do grafo.
IV. Existe algum caminho direcionado entre D e todos os outros nós do grafo.
V. O maior componente fortemente conexo de G é composto por um único nó, ou seja, não existe em G um par de nós distintos x e y que tenha um caminho direcionado entre x e y e um caminho direcionado entre y e x.

Quais estão corretas?

A) Apenas II e III.
B) Apenas I, II e IV.
C) Apenas I, III e V.
D) Apenas I, II, III e V.
E) I, II, III, IV e V.

QUESTÃO 36 – As arestas são exploradas a partir do vértice *v* mais recentemente descoberto que ainda possui arestas não exploradas saindo dele. Quando todas as arestas adjacentes a *v* tiverem sido exploradas, a busca anda para trás para explorar vértices que saem do vértice do qual *v* foi descoberto. O processo continua até que sejam descobertos todos os vértices alcançáveis a partir do vértice original. Qual algoritmo de grafos possui a estratégia descrita acima?

A) Ordenação topológica.
B) Busca em profundidade.
C) Componentes fortemente conectados.
D) Árvore geradora mínima.
E) Busca em largura.

QUESTÃO 37 – Sobre ordenação topológica em grafos, é correto afirmar que:

A) A busca em largura é utilizada para obter a ordenação topológica de um grafo direcionado acíclico.
B) A ordenação topológica de um grafo pode ser vista como uma ordenação de suas arestas ao longo de uma linha horizontal, de tal forma que todos os vértices estão classificados em ordem crescente.
C) A ordenação topológica de um grafo direcionado acíclico *G=(V,A)* é uma ordenação linear de todos os seus vértices tal que G contém uma aresta *(u, v)*, então *u* aparece antes de v.
D) A busca binária é utilizada para obter a ordenação topológica de um grafo cíclico não direcionado.
E) O algoritmo para obter a ordenação topológica de um grafo direcionado usa o transposto do grafo que consiste de todas as arestas com as suas direções invertidas.

QUESTÃO 37 – Dado um grafo G e um vértice de origem, qual é o algoritmo de busca que descobre todos os vértices a uma distância K do vértice origem, antes de descobrir qualquer vértice a uma distância K+1?

A) Pré-ordem.
B) Largura.
C) Pós-ordem.
D) Profundidade.
E) Simétrica.