Exercícios de Projeto e Análise de Algoritmos*

Flávio K. Miyazawa

1. Desenvolva um algoritmo para detectar a existência de circuito em grafo não direcionado. Seu algoritmo deve rodar em tempo $O(|V|)$, independentemente de $O(|E|)$. Considere que o grafo é dado por listas de adjacência. Justifique.

2. Desenvolva um algoritmo linear para detectar a existência de circuito ímpar em grafo não direcionado. Um circuito é ímpar se o número de arestas no circuito é ímpar.

3. Um grafo conexo é Euleriano se saindo de um vértice, é possível caminhar por cada aresta, exatamente uma vez, e voltar no vértice inicial. Descreva um algoritmo com complexidade de tempo linear que descobre se um grafo é ou não euleriano. Em caso positivo, o algoritmo deve devolver a sequência de vértices e arestas do ciclo euleriano. Dica: Use o resultado: Um grafo conexo é euleriano se dado um circuito $C = (V_c, E_c)$ de $G = (V, E)$, as componentes conexas de $G - E_c$ são também eulerianas. Observação: Um vértice isolado é um grafo euleriano.

4. Prove que um grafo G é euleriano se e somente se G é conexo e todos os vértices de G possuem grau par.

5. A transposta de um grafo dirigido $G = (V, E)$ é o grafo $G^T = (V, E^T)$, onde $E^T = \{(v, u) \in V \times V : (u, v) \in E\}$. Assim, G^T é o grafo G com todas suas arestas invertidas. Descreva algoritmos eficientes para computar G^T a partir de G, quando G é dado usando listas de adjacência e matriz de adjacência.

6. Qual a complexidade de tempo do algoritmo de busca em largura (BFS) se o grafo de entrada for representado por uma matriz de adjacência ?

7. Considere o algoritmo de busca em largura (BFS) dado em sala. Neste algoritmo usamos uma estrutura de FILA (primeiro a entrar, primeiro a sair). Considere o mesmo algoritmo, trocando a estrutura de FILA por uma estrutura de PILHA (primeiro a entrar, último a sair). I.e., cada vez que um elemento é colocado na PILHA, ele é colocado no topo da pilha e um elemento é retirado sempre do topo. Mostre que com apenas esta modificação, o algoritmo se torna uma busca em profundidade.

8. Uma das importâncias de busca em profundidade é sua relação com algoritmos recursivos. Isto os torna fáceis de implementar. Descreva um algoritmo recursivo da busca em profundidade.

9. Dê um contraexemplo para a afirmação seguinte: "Em um grafo direcionado G, se existe um caminho direcionado de u a v e, numa busca em profundidade u é descoberto antes de v, então v é descendente de u na árvore de busca em profundidade".

*Exercícios sobre busca em grafos, ordenação topológica, árvores geradoras e caminhos mínimos. Peço que envie sugestões, adições e correções para `fkm@ic.unicamp.br`.

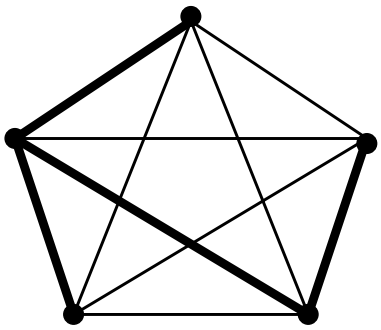
10. Escreva um algoritmo para determinar se um grafo direcionado é enraizado (com a melhor complexidade de tempo que voce conseguir). Um grafo é enraizado se existe um vertice $r \in V$ e existe pelo menos um caminho de r a qualquer outro vértice de G.

11. Um vértice em um grafo dirigido é dito ser *inicial* se a partir dele é possível atingir qualquer outro vértice por um caminho dirigido. Descreva um algoritmo, com complexidade de tempo linear, que dado um grafo dirigido por listas de adjacência, devolve o conjunto de vértices iniciais (que pode ser $\emptyset$).

12. Dado um grafo não direcionado conexo G, projete e analise um algoritmo de tempo polinomial que encontra uma árvore geradora (ou espalhada) de G com altura mínima. O algoritmo também deve apresentar o vértice que é a raiz dessa árvore. **OBS:** A *altura* de uma árvore é a maior distância da raiz até uma folha. **Dicas:** 1) o grafo não é ponderado; não confunda este problema com o problema da árvore geradora de peso mínimo. 2) Não gaste seu tempo tentando obter um algoritmo super-eficiente; está sendo pedido um algoritmo de tempo polinomial apenas

13. Dado um grafo não direcionado G, proponha um algoritmo de tempo polinomial para determinar se G possui um triângulo, isto é, 3 vértices mutualmente adjacentes.

14. Veja o grafo G na figura abaixo. As arestas grossas mostram um subgrafo de G que é uma árvore. Determine se essa árvore pode ser obtida por busca em largura **ou** por busca em profundidade a partir do vértice r. Se possível, diga qual o tipo de busca empregado e a ordem de visitação dos vértices; se não for possível, justifique.



15. Uma forma de fazer uma ordenação topológica sobre um grafo direcionado, é repetidamente encontrar um vértice de grau de entrada 0; imprima este vértice; remova este vértice (juntamente com os arcos saindo dele); e repita o processo. Descreva um algoritmo que usa esta ideia e que execute em tempo $O(|V| + |E|)$.

16. Considerando a questão anterior, o que acontece se existe um circuito? É possível construir um algoritmo para detectar circuitos a partir do algoritmo da questão anterior ?

17. Seja G um grafo direcionado acíclico, cujos vértices, numerados de 1 a n, já estão ordenados topologicamente. Descreva um algoritmo para encontrar um caminho de comprimento máximo em tempo linear no tamanho do grafo (isto é,, $O(|V| + |E|)$). Demonstre a corretude do algoritmo.

18. Uma caixa d-dimensional $c' = (x_1, \ldots, x_d)$, onde x_i é o tamanho da caixa na dimensão i, cabe numa caixa $c'' = (y_1, \ldots, y_d)$ se existe uma permutação π de $1, \ldots, d$ tal que

$$x_{\pi_1} < y_1, \ \cdots, \ x_{\pi_d} < y_d.$$

Neste caso, dizemos que c' cabe em c''.

- Dê um algoritmo eficiente para determinar se $(x_1, \ldots, x_d)$ cabe em $(y_1, \ldots, y_d)$.
- Dado um conjunto de caixas S, apresente um algoritmo eficiente para encontrar uma maior sequência de caixas $c_1, \ldots, c_k$, onde $c_i \in S$ e c_i cabe em c_{i+1}. Apresente um algoritmo com a melhor complexidade de tempo que conseguir.

Prove que este algoritmo está correto.

19. Seja $G = (V, E)$ um grafo direcionado aciclico e k o número máximo de arcos de qualquer caminho em G. Descreva em alto nivel um algoritmo que particiona os vertices do grafo em $k + 1$ subconjuntos tal que para cada dois vertices v e w de um mesmo subconjunto, nao existe caminho de v para w e não existe caminho de w para v. Seu algoritmo deve ter complexidade de tempo $O(V + E)$.

20. Um *caminho hamiltoniano* é um caminho que inclui todos os vértices do grafo. Projete um algoritmo para determinar se um dado grafo direcionado acíclico $G = (V, E)$ contém um caminho hamiltoniano. Seu algoritmo deve executar em tempo linear.

21. *Paul Erdös* é considerado o mais prolífico dos matemáticos, com diversas contribuições em mais de 1500 artigos publicados, muitos de grande importância que pavimentaram subáreas da matemática e teoria da computação. Diante da enorme quantidade de artigos e colaboradores com quem publicou, surgiu uma homenagem a ele através do termo *Número de Erdös*[*], dado a um pesquisador. O número de Erdös é calculado da seguinte maneira:

- Paul Erdös possui número de Erdös igual a 0.
- O número de Erdös de um pesquisador p, que não é o próprio Paul Erdös, é igual à soma de 1 com o menor número de Erdös de um dos seus co-autores; ou ∞ caso nenhum dos co-autores tenha número de Erdös finito.

Considere que você possui um grafo de colaborações $G = (V, E)$, onde os vértices são pesquisadores e $(u, v) \in E$ se o pesquisador u teve um artigo publicado com o pesquisador v. Faça um algoritmo com complexidade de tempo linear para calcular o número de Erdös para todos os pesquisadores do grafo. Suponha que o grafo é dado por listas de adjacências.

22. O *Mathematics Genealogy Project*[†] é uma base de dados contendo dados sobre a orientação (de doutorado) entre matemáticos. Quando disponível, a base apresenta informações de seu doutorado (como o título, ano da tese de doutorado e seu(s) orientador(es)) e uma lista com os pesquisadores que fizeram doutorado sob sua orientação. Suponha que não ocorrem ciclos direcionados no grafo subjacente. Faça algoritmos com complexidade de tempo linear para os seguintes pontos:

- Encontre os pesquisadores que não possuem orientadores.
- Uma sequência de orientação $P = (p_1, p_2, \ldots, p_k)$ é uma sequência de pesquisadores onde p_i orientou p_{i+1}, para $i = 1, \ldots, k-1$. Encontre uma maior sequência de orientação em número de pesquisadores.
- Encontre uma sequência de orientação $P = (p_1, \ldots, p_k)$, onde a distância em anos da obtenção do doutorado de p_1 até o ano da obtenção do doutorado de p_k é máxima.
- Dados dois pesquisadores p e q, verifique se p é um ancestral acadêmico de q. Em caso positivo, apresente a sequência de orientação de p até q.
- Dado um pesquisador p, construa a árvore acadêmica genealógica de p.

[*] `https://oakland.edu/enp/`

[†] `www.genealogy.ams.org/`

23. Seja G um grafo não direcionado, com pesos nas arestas. Demonstre que se todos os custos são distintos, então existe exatamente uma única árvore geradora de peso mínimo.

24. Seja $G = (V, E)$ um grafo não direcionado com custos associados às arestas e T uma árvore geradora de custo mínimo (AGM) em G. Suponha que o custo de uma única aresta $e \in E$ qualquer de G é modificada. Diga quais as condições nas quais T deixa de ser uma AGM de G. Caso T não seja mais AGM do grafo com o custo de e modificado, diga se é possível obter uma nova AGM com complexidade de tempo assintoticamente melhor que os algoritmos de PRIM e KRUSKAL.

25. Dado grafo $G = (V, E)$, com custo c_e em cada aresta $e \in E$, vimos como encontrar uma árvore T de G que é geradora e de custo mínimo. Agora, dado T, faça um algoritmo (com a melhor complexidade de tempo que puder) que encontra a próxima árvore geradora de custo mínimo T', com $T \neq T'$.

26. Seja $G = (V, E)$ um grafo não direcionado com pesos nas arestas e seja F um subgrafo de G que é uma floresta (i.e., F não contém circuitos). Projete um algoritmo eficiente para encontrar uma árvore geradora em G que contém todas as arestas de F, e tem um custo mínimo dentre todas as árvores geradoras que contêm F.

27. Considere o algoritmo de caminho mínimo a partir de um vértice: apresentado em sala. Demonstre que os caminhos mínimos de um vértice para os demais vértices, formam uma árvore geradora (*árvore de caminhos mínimos*).

28. O algoritmo para encontrar caminhos mínimos de Dijkstra decide de forma arbitrária os empates. Descreva como modificar o algoritmo tal que, se existem vários caminhos mais curtos diferentes de mesmo comprimento, então o novo algoritmo escolhe o que contém o menor número de arestas. Você pode usar espaço $O(|E|)$ adicional.

29. Dê um exemplo de grafo ponderado acíclico e direcionado com UMA aresta de peso negativo onde o algoritmo de Dijkstra falha.