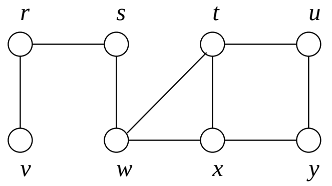
MAC 5711 - Análise de Algoritmos

Departamento de Ciência da Computação

Segundo semestre de 2002

Lista 8

1. **(Exercício 23.2-2 do CLR)** Mostre o resultado da execução do algoritmo de busca em largura no grafo da figura 23.3 (abaixo), usando o vértice u como início.



2. **(Exercício 23.2-6 do CLR)** Descreva um algoritmo eficiente para determinar se um grafo é bipartido. (Um grafo é *bipartido* se não contém nenhum circuito de comprimento ímpar.)

3. **(Problema 23-2 do CLR)** Seja $G = (V, E)$ um grafo conexo não-orientado. Um *vértice de articulação* de G é um vértice cuja remoção desconecta G. Uma *ponte* em G é uma aresta cuja remoção desconecta G. Uma *componente biconexa* de G é um conjunto maximal de arestas tal que quaisquer duas arestas no conjunto estão em um mesmo circuito. Podemos determinar vértices de articulação, pontes e componentes biconexas usando busca em profundidade. Seja G_π uma árvore de busca em profundidade de G.

   (a) Prove que a raiz de G_π é um vértice de articulação de G se e somente se ela tem pelo menos dois filhos em G_π.

   (b) Seja v um vértice de G_π que nao seja a raiz. Prove que v e um vértice de articulação de G se e somente se v tem um filho s tal que nao existe nenhuma aresta de retorno de s ou de qualquer descendente de s para um ancestral proprio de v.

   (c) Uma aresta (u, w) é *de retorno* se w é descentente de u. Seja

   $$l[v] \;=\; \min\{d[v], \{d[w] : (u, w) \text{ é uma aresta de retorno para algum descendente } u \text{ de } v\}\}.$$

   Mostre como calcular $l[v]$ para todos os vértices v em V em tempo $O(|E|)$.

   (d) Mostre como computar todos os pontos de articulação de G em tempo $O(|E|)$.

   (e) Prove que uma aresta de G é uma ponte se e somente se ela não pertence a nenhum circuito.

   (f) Mostre como calcular todas as pontes de G em tempo $O(|E|)$.

   (g) Prove que as componentes biconexas de G particionam as arestas de G que não são pontes.

   (h) Dê um algoritmo que consuma tempo $O(|E|)$ e rotule cada aresta de G com um inteiro positivo $cb[e]$ tal que $cb[e] = cb[e']$ se e somente se e e e' estão na mesma componente biconexa de G.

4. Um grafo orientado é *acíclico* se não tem nenhum circuito orientado. Uma *ordenação topológica* de um grafo orientado acíclico G é uma ordenação $v_1, \ldots, v_n$ de seus vértices em que toda aresta $v_i v_j$ de G é tal que $i < j$ (ou seja, todos os arcos vão da esquerda para a direita). Dê um algoritmo que, em tempo linear, imprima uma ordenação topológica de um grafo dado. Baseie-se em busca em profundidade.

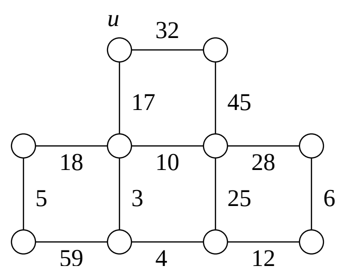
5. Caracterize todos os grafos (não-orientados) que contém um vértice v tal que existe uma árvore DFS enraizada em v e uma árvore BFS enraízada em v que coincidem.

6. **(Exercício 24.1-8 do CLR)** Seja T uma árvore geradora mínima de um grafo G, e seja L uma lista ordenada dos pesos das arestas de T. Mostre que, para qualquer outra árvore geradora mínima T' de G, a lista L é também uma lista ordenada dos pesos das arestas de T'.

7. **(Exercício 24.2-4 do CLR)** Suponha que o peso de cada aresta do grafo é um inteiro no intervalo de 1 a $|V|$. Qual o tempo de execução da implementação mais rápida que você consegue descrever para o algoritmo de Kruskal? E se os pesos estiverem entre 1 e W, para uma constante W?

8. **(Exercício 24.2-5 do CLR)** Suponha que o peso de cada aresta do grafo é um inteiro no intervalo de 1 a $|V|$. Qual o tempo de execução da implementação mais rápida que você consegue descrever para o algoritmo de Prim? E se os pesos estiverem entre 1 e W, para uma constante W?

9. Aplique o algoritmo de Kruskal e o algoritmo de Prim (começando do vértice u) ao grafo abaixo. Mostre o resultado da execução de cada um dos algoritmos após cada iteração.



10. Desenhe um grafo com custos nas arestas que tenha mais de uma árvore geradora mínima. Mostre que, se todos os custos no grafo são distintos, então o grafo contém uma única árvore geradora mínima.

Entregue no dia 18/11 os exercícios 2, 3, 6, 7 e 8.