DCA0214.1 - LABORATÓRIO DE ESTRUTURAS DE DADOS
Aula 11: Algoritmos em grafos

Prof. Felipe Fernandes

24 e 25 de Maio de 2019

Lembrete: É proibido utilizar qualquer estrutura de dados ou algoritmos pré-fornecidos por alguma biblioteca C/C++. Também é proibido utilizar código obtido na Internet sem a devida referência à fonte.

1. Suponha que nós temos n elementos distintos, numerados de 1 até n. Inicialmente, suponha que cada elemento x constitui um conjunto unitário $\{x\}$. Desejamos definir operações básicas sobre tais conjuntos, como $UNION(x, y)$, que une os conjuntos de x e de y; e $SAME\_SET(x, y)$ que retorna verdadeiro se x e y pertencem ao mesmo conjunto e falso caso contrário. Sua tarefa é implementar tais precedimentos em $O(n)$ e $O(1)$, respectivamente. (Dica: utilize um vetor sequencial).

   (a) Pense rápido: em que este exercício seria útil com respeito aos algoritmos de grafos que estudamos até agora?

2. Seja $G(V, E)$ um grafo conexo e não direcionado, ponderado com valores escalares sobre as arestas. Implemente os algoritmos de Prim e Kruskal com complexidade $O(|E|log|V|)$ para encontrar a árvore geradora mínima de G. Quais estruturas de dados e procedimentos auxiliares você utilizou? Em seguida, modifique ambos os algoritmos a fim de encontrar a árvore geradora máxima de G.

3. Sejam $G(V, E)$ um grafo e $\mathcal{T}$ o conjunto de árvores geradoras de G. Uma partição P de $\mathcal{T}$ classifica as arestas de E como proibidas (arestas que não podem figurar em nenhuma árvore geradora em P), obrigatórias (arestas de E que devem figurar em todas as árvores geradoras em P) e opcionais (arestas que podem ou não figurar em uma árvore geradora em P). A Figura 1 representa uma partição, onde as arestas azuis, vermelhas e pretas são, respectivamente, obrigatórias, proibidas e opcionais. Elabore um algoritmo $O(|E|log|V|)$ que, dada uma partição P, retorna sua árvore geradora mínima. (Obs.: seu algoritmo deve ser capaz de identificar se é possível obter uma árvore a partir da partição).

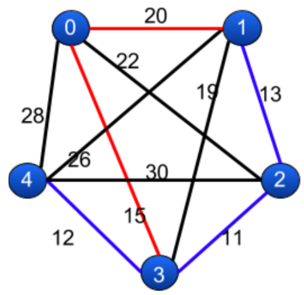

Figure 1: Uma partição

4. Seja $G(V, E)$, $|V| = n$ e $|E| = m$, um grafo conexo e não direcionado, ponderado com valores escalares nas arestas. Seja ainda $T(V, E_T)$ a AGM de G. Suponha que uma nova aresta, de custo k, é inserida em G. Elabore um algoritmo iterativo com complexidade $O(n)$ a fim de obter a nova AGM de G. (Dica: utilize duas pilhas).

5. Implemente uma função, com complexidade $O(nlogn)$, que recebe uma árvore com n vértices e retorna o seu código de Prüffer. Qual estrutura de dados você utilizou para representar a árvore e por quê?

6. Implemente uma função, com complexidade $O(nlogn)$, que recebe uma sequência de código de Prüffer com tamanho $n - 2$ e retorna a árvore correspondente com n vértices.