Análise de algoritmos - Lista 2

Professor: Guilherme Oliveira Mota

Entrega: 23h55 de 9/7/2018 (**Somente pelo Tidia**)

(1) Prove que os seguintes itens são válidos:
- $n! = O(n^n)$.
- $n! = \Omega(2^n)$.
- $\log(n!) = O(n \log n)$.
- $\log(n!) = \Omega(n \log n)$.

(2) Utilize o método indicado para obter um bom limitante superior assintótico para as seguintes recorrências.
- $T(n) = 4T(n/16) + 100\sqrt{n}$ (Teorema mestre).
- $T(n) = 4T(n/4) + 10$ (Método iterativo).
- $T(n) = 16T(n/4) + n^3$ (Teorema mestre).
- $T(n) = 16T(n/2) + \sqrt{n}$ (Método da árvore + Método da substituição).

(3) Dado um vetor ordenado $A[1..n]$ onde todos seus elementos são distintos, decidir se existe $1 \leq i \leq n$ tal que $A[i] = i$ em tempo $O(\log n)$.

(4) Escreva um algoritmo chamado HEAP-AUMENTA-VALOR(A, i, k) que recebe um heap máximo A, um índice i de A, e um valor $k \geq A[i]$ e aumenta $A[i]$ para k, consertando a propriedade de heap caso ela seja violada. Prove que seu algoritmo funciona corretamente.

(5) Escreva um algoritmo que recebe um grafo G e dois vértices s e v e retorna um caminho com a menor quantidade de arestas possível entre s e v.

(6) Utilizando a busca em largura, escreva um algoritmo que verifica se um grafo é conexo ou não.

(7) Utilizando a busca em profundidade, dado um grafo $G = (V, E)$, escreva um algoritmo que verifica se existe um ciclo em G em tempo $O(|V| + |E|)$.

(8) Apresente um algoritmo para encontrar componentes fortemente conexas e mostre sua execução em um grafo G com duas componentes fortemente conexas e 3 vértices em cada componente. O grafo G deve ser descrito com uma matriz de adjacências.

Abaixo temos alguns exercícios que não precisam ser entregues com a lista. Caso haja tempo, recomendo resolvê-los para melhorar o aprendizado.

(9) **Opcional** – Dado um vetor $A[1..n]$, uma *inversão* é um par $\{i, j\}$ com $1 \leq i < j \leq n$ tal que $A[i] > A[j]$. Faça um algoritmo que conta a quantidade de inversões em um vetor $A[1..n]$ em tempo $O(n \log n)$.

(10) **Opcional** – Faça um algoritmo para verificar se um dado grafo $G = (V, E)$ é bipartido em tempo $O(|V| + |E|)$.

(11) **Opcional** – Mostre que uma árvore binária tem altura $\Omega(\log n)$.

(12) **Opcional** – Prove que, dado um grafo $G = (V, E)$ e um vértice $s \in V$, o algoritmo de busca em largura calcula corretamente a distância de s aos vértices alcançáveis a partir de s.