SCC0502 - Algoritmos e Estruturas de Dados I

Prof. Diego Furtado Silva
Departamento de Ciências de Computação (SCC)
Instituto de Ciências Matemáticas e de Computação (ICMC)
Universidade de São Paulo

Lista de exercícios 02

1. Considerando uma árvore binária de busca com n nós, qual é a relação entre o número de comparações (entre a chave procurada e chaves em nós) e a altura da árvore?

2. Implemente uma operação ApagaArvore, para apagar uma árvore binária, liberando a memória alocada para cada nó. Qual o tipo de percurso que deve ser utilizado?

3. Considere que após uma inserção em uma árvore AVL, foi detectado que a subárvore enraizada pelo nó A mudou seu fator de balanceamento para -2. Considere também que seu filho à direita possui, agora, fator de balanceamento +1. Qual é a rotação necessária para que essa subárvore volte a ser balanceada? Demonstre que tal operação garante o balanceamento, independentemente das subárvores descendentes de A.

4. Uma busca inexata é aquela que, quando a chave procurada não existe, retorna a chave mais próxima existente. Discuta como se pode implementar uma busca inexata em uma árvore binária de busca. Implemente o algoritmo e apresente a sua complexidade.

5. O TAD dicionário (especialização de uma tabela de símbolos) possui as seguintes operações principais:
   - Insere(k, D). Insere um elemento de chave igual a k (string) no dicionário D;
   - Remove(k, D). Remove o elemento de chave igual a k do dicionário D;
   - Pesquisa(k, D). Retorna o elemento de chave igual a k do dicionário.

   Assumindo que a chave k é única, discuta os prós e contras e as complexidades de tempo e memória desse TAD utilizando as seguintes implementações:
   a. Uma lista estática não ordenada;

b. Uma lista estática (vetor) ordenada pela chave k;
c. Uma lista dinâmica (lista ligada) ordenada pela chave k;
d. Uma árvore binária de busca;
e. Uma árvore AVL;
f. Uma hash table.

6. Implemente uma operação de busca sobre árvores binárias de busca que retorna o ranque de uma dada chave. O ranque da menor chave é 0, da segunda menor chave é 1, e assim por diante. A operação deve receber a chave e um ponteiro para a raiz da árvore e deve retornar o ranque da chave ou -1 se a chave não existir. A operação deve requerer O(1) de memória. Analise a complexidade dessa operação.

7. Suponha que você esteja projetando uma tabela hash. Nesse contexto, responda às seguintes perguntas:
   a. Qual é a função hash que você escolheu e por quê?
   b. Como escolher o tamanho da tabela hash? Qual é a vantagem (e a desvantagem) de escolher um valor grande para esse parâmetro?
   c. Qual estratégia você escolheria para tratar colisões? Quais as vantagens ou desvantagens dela?
   d. Dado o seu projeto de hash, mostre pelo menos 10 inserções, com pelo menos 3 colisões. Além disso, comente como tratar a remoção no seu caso.

   p.s. Repita este exercício para diferentes tipos de chaves (inteiras, com pontos flutuantes, strings curtas, strings longas, etc).