Universidade Federal do Pará
Instituto de Ciências Exatas e Naturais
Faculdade de Computação
Análise de Algoritmos

Trabalho sobre Algoritmos de Ordenação

Valor do trabalho 4.0 pts

Equipes de no máximo 4 alunos

Prazo de entrega 21/12/2017

Questão 1. Implemente em uma linguagem de programação a sua escolha o algoritmo de ordenação QuickSort com o pivô sendo o elemento do meio. Em seguida, trabalhe os itens abaixo.

a. Use uma sequência aleatória de números inteiros entre 1 e 100 como entrada do algoritmo e apresente o *print* da execução.

b. Qual é a complexidade no tempo do algoritmo? Explique.

c. Compare o tempo de execução do algoritmo quando o mesmo recebe como entrada sequências ordenadas (de forma crescente e decrescente) e desordenadas. Caso mais números fossem inseridos nas sequências, como o desempenho do algoritmo seria afetado? Usando dados reais, ilustre graficamente suas conclusões.

d. Modifique o algoritmo para que a escolha do pivô seja feita de forma aleatória. Em seguida, refaça os itens (a), (b) e (c) usando o algoritmo modificado.

e. Faça um estudo comparativo dos dois algoritmos (pivô no meio e aleatório) nas situações apresentadas.

Questão 2. Implemente em uma linguagem de programação a sua escolha o algoritmo de ordenação HeapSort. Em seguida, trabalhe os itens abaixo.

a. Use uma sequência aleatória de números inteiros entre 1 e 100 como entrada do algoritmo e apresente o *print* da execução.

b. Qual é a complexidade no tempo do algoritmo? Explique.

c. Compare o tempo de execução do algoritmo quando o mesmo recebe como entrada sequências ordenadas (em ordem crescente e decrescente) e desordenadas. Caso mais números fossem inseridos nas sequências, como o desempenho do algoritmo seria afetado? Usando dados reais, ilustre graficamente suas conclusões.

d. Faça um estudo comparativo do algoritmo HeapSort com os algoritmos QuickSort, implementados na Questão 1, nas situações apresentadas.

Questão 3. Implemente em uma linguagem de programação a sua escolha os algoritmos de ordenação linear CountingSor e BucketSort. Em seguida, trabalhe os itens abaixo.

a. Analise o comportamento do tempo de execução do algoritmo CountingSort quando o número de elementos da sequência de entrada é aumentado gradativamente. Durante a análise, comente a relação que existe entre o tamanho da sequência de entrada e o elemento de maior valor nessa sequência. Usando dados reais, ilustre graficamente suas conclusões.

b. Compare o tempo de execução do algoritmo BucketSort quando o mesmo recebe como entrada sequências uniformemente distribuídas e sem essa propriedade. Caso mais números fossem inseridos nas sequências, como o desempenho do algoritmo seria afetado? Usando dados reais, ilustre graficamente suas conclusões.

Questão 4. Compare o tempo de execução dos algoritmos CountingSort e QuickSort em situações reais de uso. Explique qual algoritmo é mais eficiente e por que.