

Lista de Exercícios 4: Algoritmos de Ordenação I

Professor: Moacir Pereira Ponti Jr.

PAE(s): Pâmela/Paulo

1. Modifique o programa do *quicksort* de modo que, se um subsetor for pequeno, a classificação por bolha seja usada.
2. Explique como seria possível melhorar o método *bubblesort*, armazenando não apenas a informação da troca, mas também a posição do vetor onde ocorreu a troca. Implemente essa modificação.
3. No método *insertsort*, a cada passo, o menor elemento é procurado para que seja inserido na sequência já ordenada. Essa procura pode ser realizada sequencialmente ou por busca binária. Analise o desempenho de ambas as abordagens.
4. Discuta como a escolha do pivô pode influenciar no desempenho do método *quicksort*. Proponha estratégias para a escolha do pivô, visando melhorar seu desempenho.
5. Faça um teste de mesa com cada método de ordenação estudado até o momento, utilizando as seguintes sequências de dados de entrada:

(i) $S_1 = \{2, 4, 6, 8, 10, 12\}$

(ii) $S_2 = \{11, 9, 7, 5, 3, 1\}$

(iii) $S_3 = \{5, 7, 2, 8, 1, 6\}$

(iv) $S_4 = \{2, 4, 6, 8, 10, 12, 11, 9, 7, 5, 3, 1\}$

(v) $S_5 = \{2, 4, 6, 8, 10, 12, 1, 3, 5, 7, 9, 11\}$

(vi) $S_6 = \{8, 9, 7, 9, 3, 2, 3, 8, 4, 6\}$

(vii) $S_7 = \{89, 79, 32, 38, 46, 26, 43, 38, 32, 79\}$

Em cada caso, mostre o número de comparações e trocas que realizam na ordenação de sequências.

6. Dos algoritmos estudados, quais são estáveis? Utilize os itens (vi) e (vii) do exercício anterior para apoiar sua resposta.
7. Considere a ordenação de n números armazenados no arranjo A , localizando primeiro o menor elemento de A e permutando esse elemento contido em $A[1]$. Em seguida, encontre o segundo menor elemento de A e o troque pelo elemento $A[2]$. Continue dessa maneira para os primeiros $n - 1$ elementos de A . Escreva o pseudocódigo para esse algoritmo conhecido como *ordenação por seleção*. Qual invariante do laço esse algoritmo mantém? Por que ele só precisa ser executado para os primeiros $n - 1$ elementos, e não para todos os elementos? Forneça os tempos de execução do melhor caso e do pior caso da ordenação por seleção em notação \mathcal{O} .

8. Crie um algoritmo chamado *quickfind* baseado no *quicksort* para que, em vez de ordenar uma sequência de números inteiros, ele nos retorne o k -ésimo menor elemento dessa sequência. Por exemplo: Suponha que os elementos $S = \{7, 1, 3, 10, 17, 2, 21, 9\}$ estejam armazenados nessa ordem em um vetor e que desejamos obter o quinto maior elemento dessa sequência. Então, uma chamada como `quickfind(S,0,7,5)`, deverá retornar o número 9, onde **S** é o nome do vetor, 0 e 7 são, respectivamente, a menor e a maior posição do vetor e 5 indica que desejamos o quinto menor elemento.
- Obs.:** Você não deve ordenar a sequência e depois tomar o k -ésimo elemento.

Referências

- [1] Parte deste material foi adaptada das listas de exercícios do Prof. Luis Gustavo Nonato, ICMC/USP.
- [2] Preiss, B. R., *Data Structures and Algorithms with Object-Oriented Design Patterns in C++*. 1997.
- [3] Parte deste material foi adaptada das listas de exercícios do Prof. João Luís Garcia Rosa, ICMC/USP.