USP-ICMC-BInfo

Introdução à Ciência de Computação II - SCC-0501

Prof. João Luís - 16/11/2011

Lista de Exercícios 6

1. Escreva uma versão de um algoritmo que retorna a soma de todos os elementos de um vetor de inteiros usando a abordagem "divisão e conquista".

2. Construa um algoritmo recursivo para encontrar o maior elemento das entradas $A[1]$, $A[2]$, ..., $A[n]$ de um arranjo A. O procedimento deve dividir o arranjo em duas partes de tamanhos aproximadamente iguais.

   (a) Derive uma relação de recorrência para a função de complexidade f, onde $f(n)$ é definida como o número de comparações entre os elementos de A e n ser considerada uma potência de 2.

   (b) Resolva a relação de recorrência.

3. O que aconteceria com a eficiência dos algoritmos baseados em divisão e conquista se recorrêssemos no máximo r vezes, para uma dada constante r, e depois utilizássemos o subalgoritmo básico, em vez de um limite para decidir quando reverter para o subalgoritmo?

4. Apresente um esboço do esquema geral de algoritmos de *backtracking*.

5. O algoritmo de ordenação por seleção pode ser considerado um algoritmo guloso? Se sim, quais são as várias funções envolvidas (a função de viabilidade, a função de seleção, etc.)?

6. Escrever um procedimento que gere todas as $n!$ permutações dos n elementos a_1, ..., a_n, sem a criação de outro vetor. Ao gerar a próxima permutação, um procedimento Q paramétrico é chamado, podendo, por exemplo, gerar como saídas as permutações efetuadas. Sugestão: Considere a tarefa de gerar todas as permutações dos elementos a_1, ..., a_m como consistindo de m subtarefas que gerem todas as permutações de a_1, ..., a_{m-1}, seguido por a_m, onde, na i-ésima tarefa, dos dois elementos a_i e a_m tenham sido inicialmente trocados.

7. Somente 12 das 92 soluções do problema das oito rainhas são essencialmente diferentes. As outras podem ser obtidas como imagens especulares relativas aos eixos e por simetria em relação ao ponto central. Projete um programa que determine as 12 soluções principais. Note-se, por exemplo, que a busca na coluna 1 pode ser restrita às posições 1-4.

8. No problema da seleção de atividades, suponha que, em vez de selecionar sempre a atividade que termina primeiro, selecionamos a última atividade a começar que seja compatível com as atividades selecionadas anteriormente. Descreva como essa abordagem é um algoritmo guloso. Escreva o código (C) de um programa que a implemente.

9. Vamos supor que temos um conjunto de atividades para programar entre um grande número de salas de conferência. Desejamos programar todas as atividades usando o mínimo possível de salas de conferências. Forneça um algoritmo guloso eficiente para determinar que atividade deve usar cada sala de conferência.

10. O professor Midas dirige um automóvel do Rio para São Paulo pela Via Dutra. O tanque de combustível de seu carro, quando completo, contém gasolina suficiente para viajar n quilômetros, e o mapa do professor fornece as distâncias entre os postos de gasolina em sua rota. O professor deseja fazer o mínimo possível de paradas para abastecimento ao longo do caminho. Forneça um método eficiente pelo qual o professor Midas possa determinar em quais postos de gasolina deve parar, e mostre que sua estratégia produz uma solução ótima.

11. Desenvolva um método recursivo (e programe-o) para calcular o número de diferentes modos nos quais um inteiro k pode ser escrito como uma soma, sendo cada um dos operandos menor que n.

12. Um vetor *maze* de 0s e 1s, de 10 x 10, representa um labirinto no qual um viajante precisa encontrar um caminho de *maze*[0][0] a *maze*[9][9]. O viajante pode passar de um quadrado para qualquer outro adjacente na mesma fileira ou coluna, mas não pode saltar quadrados nem se movimentar na diagonal. Além disso, o viajante não pode entrar num quadrado contendo um 1. *maze*[0][0] e *maze*[9][9] contêm 0s. Escreva uma rotina que aceite este labirinto *maze* e imprima uma mensagem informando a inexistência de um caminho através do labirinto, ou que imprima uma lista de posições representando um caminho de [0][0] a [9][9].

Referências

[1] Cormen, T. H., Leiserson, C. E., Rivest, R. L., Stein, C., *Algoritmos - Teoria e Prática*. Ed. Campus, Rio de Janeiro, Segunda Edição, 2002.

[2] Oliveira, Maria Cristina Ferreira de, *Sexta Lista de Exercícios - Paradigmas de projeto de algoritmos*, SCE0181 - Introdução à Ciência da Computação II. ICMC-USP, 2008.

[3] Tenenbaum, A. M., Langsam, Y., Augestein, M. J., *Estruturas de Dados Usando C*. Makron Books, 1995.

[4] Wirth, N., *Algoritmos e Estruturas de Dados*. LTC, 1989.

[5] Ziviani, N., *Projeto de Algoritmos*, 2a. Edição. Thomson, 2004.