MCTA003-17 – Análise de Algoritmos
Centro de Matemática, Computação e Cognição
Universidade Federal do ABC
Profa. Carla Negri Lintzmayer

Lista 4

Entrega: até 23h55 do dia 13/08/2018

- Submeta ao tidia um único arquivo PDF com as suas soluções escaneadas.
- Seja o mais **formal** possível em todas as respostas.
- Identifique devidamente cada exercício.
- Identifique devidamente os autores da lista (caso você tenha feito em dupla).
- A lista é uma forma de treino para a prova, que não terá consulta. Evite plágio!

1. Modifique o algoritmo de PD para o problema da Mochila (inteira) visto em sala para que ele preencha um vetor S de tamanho $n + 1$ que auxiliará na construção da solução ao final da execução.

2. Mostre como utilizar o algoritmo de PD para o problema do Alinhamento de Sequências visto em sala para resolver o problema da Subsequência Comum mais Longa (LCS). No LCS são dadas duas sequências A e B de tamanhos diferentes sobre um mesmo alfabeto, e queremos encontrar o tamanho da subsequência mais longa que é comum a ambas. Uma subsequência consiste de caracteres que aparecem na mesma ordem relativa, mas não necessariamente de forma contínua. Por exemplo, a LCS entre A = abcdgh e B = aedfhr é adh e a LCS entre A = aggtab e B = gxtxayb é gtab.

3. Dada a matriz M totalmente preenchida pelo algoritmo de PD para o problema do Alinhamento de Sequências visto em sala, mostre como construir um alinhamento para as sequências da entrada.

4. Considere o seguinte problema. Seja $G = (V, E)$ um grafo que constitui apenas de um caminho, isto é, $V = \{v_1, v_2, \ldots, v_n\}$ e $E = \{(v_i, v_{i+1}) \colon 1 \leq i < n\}$ e seja $w \colon V \to \mathcal{R}$ uma função de peso sobre os vértices. O problema do Conjunto Independente de Peso Máximo consiste em receber um grafo caminho G e a função w e o objetivo é encontrar um subconjunto S de vértices tal que S é um conjunto independente (isto é, para quaisquer dois vértices $u, v \in S$, não existe a aresta uv em G) e $\sum_{v \in S} w(v)$ é máximo. Analise a estrutura de uma solução ótima, construa uma recorrência e escreva o pseudocódigo de um algoritmo em PD para resolver esse problema.

5. Faça um algoritmo em PD para calcular $\binom{n}{k}$ sem usar a fórmula fechada para isso. Por definição, $\binom{n}{k} = \binom{n-1}{k-1} + \binom{n-1}{k}$ para $1 \leq k < n$ e $\binom{n}{n} = \binom{n}{0} = 1$. Analise a estrutura de uma solução ótima, construa uma recorrência e escreva o pseudocódigo do algoritmo.

6. Defina: algoritmo eficiente, problema de decisão, certificado positivo, certificado negativo, algoritmo verificador, classes P, NP e coNP e problemas NP-completos.

7. Seja A um problema NP-completo. Seja B um outro problema qualquer. Diga o que podemos concluir ao reduzir A para B e ao reduzir B para A.

8. Mostre que se o TSPk pode ser resolvido em tempo polinomial então o TSP também pode. O TSPk é a versão de decisão do TSP.

9. Mostre que o SET-COVER é NP-completo. Dica: use um outro problema de cobertura visto em sala de aula.