NP-completude

Classes de problemas e problemas polinomiais

Questão 1. (CLRS) Exercícios: 34.1-4

Questão 2. O objetivo deste exercício é entender porque alguma vezes é fácil reduzir (Turing) um problema de otimização para sua versão de decisão. Considere o Problema do Caixeiro Viajante (TSP): Dado um grafo completo $G = (V, E)$ com custos $w : V \to \mathbb{N}$, encontre um caminho hamiltoniano peso mínimo de G.

(a) Defina a versão de decisão do problema, TSP-Dec com parâmetro k.
(b) Suponha que você tem à disposição um algoritmo polinomial A que dados (G, w) e k decide TSP-dec, i.e., responde sim se e somente se há uma solução de custo até k. Mostre, usando A, como determinar o peso do caminho hamiltoniano mínimo mínima de G em tempo polinomial.
(c) Mostre, usando A e o peso do caminho hamiltoniano mínimo determinado no item anterior, como **encontrar** um do caminho hamiltoniano mínimo em G em tempo polinomial.
(d) Conclua que TSP $\preccurlyeq_p$ TSP-dec.

Problemas verificáveis em tempo polinomial

Questão 3. Se existe problema $L \in \mathsf{NP}$ tal que $L \notin \mathsf{NP}$-completo, então para todo $L' \in \mathsf{NP}$-completo, não existe redução $L' \preccurlyeq_p L$.

Questão 4. (CLRS) (34.2-1) Considere o problema Isomorfismo de Grafos (IG). Mostre que IG é NP apresentando um algoritmo verificador polinomial.

Questão 5. (CLRS) (34.2-5) Mostre que qualquer linguagem pertencente a NP pode ser decidida por um algoritmo em tempo $2^{O(n^k)}$, para alguma constante k.

Questão 6. (CLRS) (34.2-6) Um caminho hamiltoniano em um grafo não direcionado, é um caminho que passa exatamente uma vez por todos os vértices do grafo. O problema do caminho hamiltoniano (HAM-PATH) consiste em decidir se um grafo possui ou não um caminho hamiltoniano. Apresente um algoritmo verificador polinomial para a linguagem HAM-PATH e conclua que HAM-PATH é NP.

Questão 7. (CLRS) (34.2-7)Mostre que o problema do caminho hamiltoniano pode ser resolvido em tempo polinomial para grafos direcionados acíclicos. Dê um algoritmo eficiente para o problema.

Questão 8. (CLRS) (34.2-9) Mostre que $\mathsf{P} \subseteq \mathsf{NP}$ e $\mathsf{P} \subseteq \text{co-}\mathsf{NP}$.

Questão 9. (CLRS) (34.2-10) Mostre que se $\mathsf{NP} \neq \text{co-}\mathsf{NP}$ então $\mathsf{P} \neq \mathsf{NP}$.

Questão 10. Considere o problema para decidir se um número p é primo. Mostre que esse problema está em co-NP. Atente-se para o tamanho da codificação de um número p.

Demonstrações de NP-completude

Questão 11. (CLRS) (34.3-2) Mostre que se $L_1 \preccurlyeq_p L_2$ e $L_2 \preccurlyeq_p L_3$ então $L_1 \preccurlyeq_p L_3$.

Questão 12. Mostre que $L \preccurlyeq_p \overline{L}$ se e somente se $\overline{L} \preccurlyeq_p L$.

Questão 13. Dado um conjunto de elementos $E = \{1, 2, ..., n\}$ e uma família de conjuntos $\mathcal{S} = \{S_1, S_2, ..., S_m\}$, onde $S_i \subseteq E$, uma cobertura de conjuntos é uma subfamília $F \subseteq \mathcal{S}$ tal que $\bigcup_{S \in F} S = E$. O problema da cobertura por conjuntos é: dados E, $\mathcal{S}$ e k, existe uma cobertura de conjuntos F de tamanho $|F| \leq k$? Mostre que cobertura por conjuntos é NP-completo. Dica: você pode supor que cobertura por vértices é NP-completo.

Questão 14. (CLRS) (34.5-3) The integer linear-programming problem is like the 0-1 integer-programming problem given in Exercise 34.5-2, except that the values of the vector x may be any integers rather than just 0 or 1. Assuming that the 0-1 integer-programming problem is NP-hard, show that the integer linear-programming problem is NP-complete.

Questão 15. (CLRS) (34.5-4) Show how to solve the subset-sum problem in polynomial time if the target value t is expressed in unary.

Questão 16. (CLRS) (34.5-5) The set-partition problem takes as input a set S of numbers. The question is whether the numbers can be partitioned into two sets A and $\bar{A} = S - A$ such that $\sum_{x \in A} x = \sum_{x \in \bar{A}} x$. Show that the set-partition problem is NP-complete.

Questão 17. (CLRS) (34.5-7) The longest-simple-cycle problem is the problem of determining a simple cycle (no repeated vertices) of maximum length in a graph. Formulate a related decision problem, and show that the decision problem is NP-complete.

Questão 18. Considere o problema do empacotamento (BIN). Dado um conjunto finito de n objetos com pesos $w_1, w_2, \ldots, w_n$ inteiros positivos e dois valores inteiros positivos W e k, é possível colocar todos os objetos em k caixas cujo limite máximo de peso é W? Mostre que este problema é NP-completo.

Questão 19. Prove que o problema a seguir é NP-completo: dado um grafo $G = (V, E)$ e um inteiro positivo k, determine se G contém uma árvore geradora T tal que todo vértice em T tenha grau no máximo k.

Questão 20. Mostre que o seguinte problema é NP-completo: dado um grafo $G = (V, E)$ e um inteiro positivo k, determine se G contém uma árvore geradora T tal que todo vértice em T tem grau no máximo k.

Questão 21. (CLRS) Problema: 34-1 (a-b), 34-4 (d-f)

Questão 22. (CLRS) Resolva o Problema 34-3 (somente itens d,e,f); seja bastante conciso. Não plagie.

Mapmakers try to use as few colors as possible when coloring countries on a map, as long as no two countries that share a border have the same color. We can model this problem with an undirected graph $G = (V, E)$ in which each vertex represents a country and vertices whose respective countries share a border are adjacent. Then, a k**-coloring** is a function $c : V \to \{1, 2, \ldots, k\}$ such that $c(u) \neq c(v)$ for every edge $(u, v) \in E$. In other words, the numbers $\{1, 2, \ldots, k\}$ represent the k colors, and adjacent vertices must have different colors. The graph-coloring problem is to determine the minimum number of colors needed to color a given graph. Let 3-COLOR be the problem to decide whether a graph has a 3-coloring. Show that 3-COLOR is NP-complete.