MC558: Análise de Algoritmos II
Lista de Classes de Complexidade – Parte 1

1. Prove que os seguintes problemas estão em $\mathcal{NP}$: cobertura de vértices, conjunto dominante, partição, problema binário da mochila, ciclo hamiltoniano, caixeiro viajante, 3-coloração. Escreva um algoritmo polinomial não determinístico para cada problema. Use corretamente a função Escolha(S) (ela devolve um elemento de S).

2. Prove que o problema do conjunto independente é $\mathcal{NP}$-completo.

3. Considere o seguinte algoritmo para determinar se um grafo $G = (V, E)$, com $|V| = n$ e $|E| = m$, tem uma clique de tamanho k. Primeiro são gerados todos os subconjuntos contendo exatamente k vértices. Existem $O(n^k)$ subconjuntos deste tipo. Em seguida, verifica-se se algum dos *subgrafos induzidos* por estes subconjuntos é completo, devolvendo `SIM` em caso afirmativo e `NÃO` caso contrário. Supondo que o grafo de entrada é dado pela sua lista de adjacências, determine a complexidade deste algoritmo. Por que isto não é uma prova de que $\mathcal{P} = \mathcal{NP}$?

4. Escreva a instância de entrada do problema `3SAT` obtida através da redução do problema `SAT` vista em aula quando a instância de entrada deste último problema é dada por:

$$F = (x + y + \overline{z} + w + u + \overline{z}) \cdot (\overline{x} + \overline{y} + z + \overline{w} + u + v) \cdot (x + \overline{y} + \overline{z} + w + u + \overline{v}) \cdot (x + \overline{y}).$$

5. Desenhe o grafo obtido da redução vista em aula do problema `SAT` para o problema `CLIQUE` quando a instância de entrada de `SAT` é dada por:

$$F = (x + \overline{y} + z) \cdot (\overline{x} + y + \overline{z}) \cdot (\overline{x} + y + z) \cdot (x + \overline{y} + \overline{z}).$$

6. Desenhe o grafo obtido da redução vista em aula do problema `3SAT` para o problema `3COLOR` quando a instância de entrada de `3SAT` é dada por:

$$F = (x + \overline{y} + z) \cdot (\overline{x} + y + \overline{z}) \cdot (\overline{x} + y + z).$$

7. Usando um dos problemas vistos em aula, prove que o problema a seguir é $\mathcal{NP}$-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 $\leq k$.

8. Mostre que o problema de cobertura de vértices (`CV`) visto em aula permanece $\mathcal{NP}$-completo mesmo se <u>todos</u> os vértices do grafo tiverem grau **par**. *Sugestão*: use o problema `CV` original para fazer esta prova.

   Nota: o número de vértices de grau ímpar em um grafo qualquer é sempre par. Use este fato na sua demonstração de que o problema acima é $\mathcal{NP}$-completo.

9. Seja $G = (V, E)$ um grafo não orientado. Dado um subconjunto de vértices U de G, denota-se por $E(U)$ o conjunto de arestas de E com ambas os extremos em U. O *subgrafo induzido por* U *em* G é o subgrafo $H = (U, E(U))$.

   Considere então o seguinte problema: dado $G = (V, E)$ e dois inteiros não-negativos d e k, ambos menores ou iguais a $|V|$, determine se G contém um subconjunto U com k vértices tal que o grau de todo vértice de U no *subgrafo induzido por* U em G seja $\leq d$.

   Prove que este problema é $\mathcal{NP}$-completo.

10. (*) Seja F_{11} uma fórmula booleana na *forma normal conjuntiva* onde cada variável aparece exatamente uma vez como x e exatamente uma vez como $\overline{x}$. Encontre um algoritmo polinomial que determine se F_{11} pode se tornar verdadeira para alguma atribuição dos valores das variáveis **ou** mostre que este problema é $\mathcal{NP}$-completo.

11. (*) Prove que a seguinte variação do problema `3SAT` chamada de `1-em-3SAT` é um problema $\mathcal{NP}$-completo. A entrada é a mesma do problema `3SAT` usual. O problema é determinar se existe ou não uma atribuição aos valores das variáveis que torne a fórmula verdadeira **mas** de modo que em cada cláusula haja **exatamente** uma única variável verdadeira. A seguir, mostre explicitamente uma redução polinomial de `1-em-3SAT` a `SAT` (sabe-se que existe pois ambos são $\mathcal{NP}$-completos!)

12. Suponha que o problema do caminho hamiltoniano (`CaH`) para grafos não orientados é $\mathcal{NP}$-completo. Prove que o problema do ciclo hamiltoniano para grafos não orientados (`CiH`) é $\mathcal{NP}$-completo.

13. Suponha que o problema do caminho hamiltoniano (`CaH`) para grafos não orientados é $\mathcal{NP}$-completo. Prove que o problema de caminho hamiltoniano para *grafos orientados* (`CaHD`) é $\mathcal{NP}$-completo. Sua redução também funciona para ciclo hamiltoniano ?

14. Considere o seguinte problema: dado um grafo não orientado G e dois vértices especiais s e t em G, determine se G contém um caminho hamiltoniano que comece em s e termine em t. Prove que não existe algoritmo determinístico polinomial para este problema, a menos que $\mathcal{P} = \mathcal{NP}$ (use redução de Turing).

15. Considere a versão de decisão do problema de encontrar um caminho (simples) mínimo `CMD`: dado um grafo orientado G possivelmente com arcos de peso negativo, vértices s e t em G e um inteiro k, decidir se existe um caminho de s a t de peso menor ou igual a k. Mostre que este problema é $\mathcal{NP}$-completo. (Nossa! Eu achava que o algoritmo de Bellman-Ford resolvia esse problema... Ou não?)

16. O problema do caixeiro viajante (`TSP`) tem uma versão não-euclidiana: dados um grafo completo G com pesos nas arestas e um inteiro k decidir se G possui um ciclo hamiltoniano de peso menor ou igual a k.

    Supondo que o problema do ciclo hamiltoniano (`CiH`) para grafos não orientados é $\mathcal{NP}$-completo mostre que `TSP` é $\mathcal{NP}$-completo mesmo quandos todos os pesos das arestas são iguais a 1 ou 2.

17. Prove que (a versão de decisão) o problema de encontrar uma árvore geradora com o **número mínimo de folhas** em um grafo não orientado é $\mathcal{NP}$-completo.

18. Considere o problema (inteiro) da mochila (KP) ou *knapsack problem*: dados (i) itens $1, \ldots, n$ tais que a cada item i está associado: um peso w_i inteiro, um custo c_i e um inteiro m_i indicando o número de cópias do item i, (ii) um inteiro W e (iii) um número real C, decidir se existe uma "mochila" de peso no máximo W e de custo pelo menos C. Nesta versão permite-se tomar mais de uma cópia de um item i, desde que não ultrapasse m_i. Mostre que o problema da mochila é $\mathcal{NP}$-completo.

19. Um problema de Programação Linear Inteira (PLI) consiste em dados uma matriz A e vetores b, c (com dimensões apropriadas), encontrar uma solução ótima de

$$\begin{array}{rl} \min & cx \\ \text{s.a} & Ax \leq b \\ & x \geq 0 \\ & x \text{ inteiro} \end{array}$$

O sistema de restrições *não precisa* estar na forma canônica (basta ser um conjunto de restrições lineares). Além disso, pode-se estabeler um intervalo para x ($l \leq x \leq u$).

Mostre que PLI é $\mathcal{NP}$-difícil modelando o problema Cobertura de Vértices (versão de otimização) como um programa linear inteiro.

20. Modele também os seguintes problemas (versão de otimização) como um PLI: Conjunto Dominante, Mochila Binária, CLIQUE.

21. Mostre que o seguinte problema é $\mathcal{NP}$-difícil. Dada uma coleção de conjuntos $\mathcal{C} = \{S_1, \ldots, S_k\}$ onde cada elemento $e \in S_i$, $0 \leq i \leq k$, possui tamanho $t(e)$ e valor $v(e)$. O problema consiste em escolher **exatamente** um elemento e de cada conjunto S_i, $0 \leq i \leq k$ tal que a soma dos tamanhos destes elementos seja no máximo K e a soma dos valores dos elementos seja maximizada.

22. (*) Mostre que o problema a seguir é $\mathcal{NP}$-completo: dado um grafo não orientado G e um inteiro k determinar se G possui uma clique de tamanho k e um conjunto independente de tamanho k.

23. Mostre que o problema de seqüenciamento com janelas de tempo é $\mathcal{NP}$-completo. Veja os slides para a definição do problema.

24. Sejam G um grafo e $S := \{\{s_1, t_1\}, \{s_2, t2\}, \ldots, \{s_n, t_n\}\}$ um conjunto de pares não-ordenados de vértices de G. Um *multicorte* que separa S é um subconjunto F de arestas de G tal que nenhum componente de $G - F$ contém dois vértices de um mesmo par em S. O problema do multicorte mínimo (`MC`) é o seguinte: dado um grafo G, um conjunto de pares S e um inteiro k decidir se existe um multicorte com no máximo k arestas. Prove que `MC` é $\mathcal{NP}$-completo mesmo se G for uma árvore.

    Dica: faça uma redução do problema de cobertura de vértices.