Lista 2 de Complexidade de Algoritmos - 2021.01 (COS700/MAB703)

Professores: Celina Miraglia, Fábio Botler e Franklin Marquezino
Monitores: Mariana Martins e Rafael Schneider
Data de entrega: 17/01/2022

Observacão. A resolução de cada questão deve ser iniciada em uma nova folha de papel. Além disso, antes do início de cada questão, deve-se incluir o número da questão e o nome completo do aluno.

Para resolver as questões abaixo considere as seguintes definições:

- Um subgrafo C de um grafo G é um **clique** em G, se para todo par de vértices $x, y \in V(C)$, temos $xy \in E(G)$;
- Um conjunto $I \subseteq V(G)$ é **independente** se para todo par de vértices $x, y \in I$, temos $xy \notin E(G)$;
- Um conjunto $S \subseteq V(G)$ é uma **cobertura por vértices** de G, se para toda aresta $xy \in E(G)$, temos $x \in S$ ou $y \in S$.

Considere então os seguintes problemas:

Clique Máxima

I **Problema**: MaxClique
Dados: um grafo G
Objetivo: encontrar um maior clique $C \subseteq V(G)$ em G.

I' **Problema**: k-Clique
Dados: um grafo G e um inteiro k
Objetivo: decidir se há clique $C \subseteq V(G)$ em G tal que $|V(C)| \geq k$.

Independente Máximo

II **Problema**: MaxStableSet
Dados: um grafo G
Objetivo: encontrar um maior conjunto independente em G.

II' **Problema**: k-StableSet
Dados: um grafo G e um inteiro k
Objetivo: decidir se há conjunto independente de vértice $I \subseteq V(G)$ tal que $|I| \geq k$.

Cobertura Mínima por Vértices

III **Problema**: MinVertexCover
Dados: um grafo G
Objetivo: encontrar uma menor cobertura por vértices $S \subseteq V(G)$ em G.

III' **Problema**: k-VertexCover
Dados: um grafo G e um inteiro k
Objetivo: decidir se há cobertura por vértices $S \subseteq V(G)$ tal que $|S| \leq k$.

1. Escolha i em $\{I, II, III\}$, considere o par de problemas i, i' e mostre que:

- Podemos resolver i' com uma única execução de um algoritmo que resolva i.
- Podemos resolver i com um número polinomial (no tamanho de uma instância) de execuções de um algoritmo que resolva i'.

2. Suponha que exista um algoritmo eficiente $A_{I'}$ para resolver o problema I'. Construa algoritmos eficientes para resolver os problemas II' e III'.

3. Mostre que o problema I' é $\mathcal{NP}$-completo.

4. Seja $A[1 \ldots n]$ um array de n números distintos. Se $i < j$ e $A[i] > A[j]$, então o par (i, j) é denominado uma inversão de A. Suponha que os elementos de A formem uma permutação aleatória uniforme de $\langle 1, 2, \ldots, n \rangle$, isto é, cada uma das permutações possíveis aparecem com igual probabilidade. Use variáveis aleatórias indicadoras para calcular o número esperado de inversões.