Problemas NP-Completos

Fábio Botler

Programa de Engenharia de Sistemas e Computação
Universidade Federal do Rio de Janeiro

Aula passada

- Algoritmos eficientes
- Problemas tratáveis e intratáveis
- Problemas algoritmicos
- Codificações
- Tipos de problemas

Aula de hoje

- A Classe P
- Problemas aparentemente difíceis

Definição

Um algoritmo A é dito **eficiente** *se sua complexidade for um polinômio no tamanho da sua entrada.*

Definição

Um algoritmo A é dito **eficiente** *se sua complexidade for um polinômio no tamanho da sua entrada.*

A é eficiente se existe inteiro k tal que para toda instância I, o número de operações que A realiza para resolver I é $O(|I|^k)$.

Definição

Um algoritmo A é dito **eficiente** *se sua complexidade for um polinômio no tamanho da sua entrada.*

A é eficiente se existe inteiro k tal que para toda instância I, o número de operações que A realiza para resolver I é $O(|I|^k)$.

Exemplo

- $O(n^2)$

Definição

Um algoritmo A é dito **eficiente** *se sua complexidade for um polinômio no tamanho da sua entrada.*

A é eficiente se existe inteiro k tal que para toda instância I, o número de operações que A realiza para resolver I é $O(|I|^k)$.

Exemplo

- $O(n^2)$
- $O(n^{1000})$

Definição

Um algoritmo A é dito **eficiente** *se sua complexidade for um polinômio no tamanho da sua entrada.*

A é eficiente se existe inteiro k tal que para toda instância I, o número de operações que A realiza para resolver I é $O(|I|^k)$.

Exemplo

- $O(n^2)$
- $O(n^{1000})$
- $O(n \log \log n)$

Definição

Um algoritmo A é dito **eficiente** *se sua complexidade for um polinômio no tamanho da sua entrada.*

A é eficiente se existe inteiro k tal que para toda instância I, o número de operações que A realiza para resolver I é $O(|I|^k)$.

Exemplo

- $O(n^2)$
- $O(n^{1000})$
- $O(n \log \log n)$

Também usaremos

- ▷ **polinomial** para algoritmos eficientes
- ▷ **exponencial** para algoritmos de complexidade $O(2^n)$

A eficiência do algoritmo representa apenas o caso assintótico.

A eficiência do algoritmo representa apenas o caso assintótico.

Exemplo

Considere dois algoritmos A_1 *e* A_2 *para um mesmo problema, tal que*

$$A_1 = \theta(n^{10}) \text{ e } A_2 = \theta(2^n)$$

Para uma instância de tamanho $n = 2$, A_1 *realiza algo da ordem de* 1024 *operações, enquanto* A_2 *realiza algo da ordem de* 4 *operações.*

A eficiência do algoritmo representa apenas o caso assintótico.

Exemplo

Considere dois algoritmos A_1 e A_2 para um mesmo problema, tal que

$$A_1 = \theta(n^{10}) \text{ e } A_2 = \theta(2^n)$$

Para uma instância de tamanho $n = 2$, A_1 realiza algo da ordem de 1024 *operações, enquanto A_2 realiza algo da ordem de* 4 *operações.*

Um algoritmo de complexidade $\theta(n^{1000})$ é eficiente só em teoria.

A Classe P

$$P = \left\{ \begin{array}{l} \text{problemas de decisão que} \\ \text{admitem algoritmo polinomial} \end{array} \right\}$$

A Classe P

$$P = \left\{ \begin{array}{l} \text{problemas de decisão que} \\ \text{admitem algoritmo polinomial} \end{array} \right\}$$

Exemplo

A Classe P

$$P = \left\{ \begin{array}{l} \text{problemas de decisão que} \\ \text{admitem algoritmo polinomial} \end{array} \right\}$$

Exemplo

- *programação linear;*

A Classe P

$$P = \left\{ \begin{array}{l} \text{problemas de decisão que} \\ \text{admitem algoritmo polinomial} \end{array} \right\}$$

Exemplo

- *programação linear;*
- *máximo divisor comum;*

A Classe P

$$P = \left\{ \begin{array}{l} \text{problemas de decisão que} \\ \text{admitem algoritmo polinomial} \end{array} \right\}$$

Exemplo

- *programação linear;*
- *máximo divisor comum;*
- *emparelhamento máximo;*

A Classe P

$$P = \left\{ \begin{array}{l} \text{problemas de decisão que} \\ \text{admitem algoritmo polinomial} \end{array} \right\}$$

Exemplo

- *programação linear;*
- *máximo divisor comum;*
- *emparelhamento máximo;*
- *decidir se um número é primo;*

A Classe P

$$P = \left\{ \begin{array}{l} \text{problemas de decisão que} \\ \text{admitem algoritmo polinomial} \end{array} \right\}$$

Exemplo

- *programação linear;*
- *máximo divisor comum;*
- *emparelhamento máximo;*
- *decidir se um número é primo;*
- *decidir se um grafo é bipartido, ou se admite uma 2-coloração própria;*

A Classe P

$$P = \left\{ \begin{array}{l} \text{problemas de decisão que} \\ \text{admitem algoritmo polinomial} \end{array} \right\}$$

Exemplo

- *programação linear;*
- *máximo divisor comum;*
- *emparelhamento máximo;*
- *decidir se um número é primo;*
- *decidir se um grafo é bipartido, ou se admite uma 2-coloração própria;*
- *decidir se um grafo é planar.* *linear*

A Classe P

$$P = \left\{ \begin{array}{l} \text{problemas de decisão que} \\ \text{admitem algoritmo polinomial} \end{array} \right\}$$

Exemplo

- *programação linear;*
- *máximo divisor comum;*
- *emparelhamento máximo;*
- *decidir se um número é primo;*
- *decidir se um grafo é bipartido, ou se admite uma 2-coloração própria;*
- *decidir se um grafo é planar.* *linear*

O problema de decidir se um grafo possui um circuito Hamiltoniano está em P?

- ▷ Como não é conhecido nenhum algoritmo polinomial para o Problema de Circuito Hamiltoniano, não podemos concluir que ele está em P.
- ▷ Por outro lado, não há prova de que esse problema não está em P.

Alguns problemas aparentemente difíceis

Satisfatibilidade

DADOS:	Uma expressão booleana E na Forma Normal Conjuntiva (FNC)
OBJETIVO:	E é satisfatível?

Satisfatibilidade

DADOS:	Uma expressão booleana E na Forma Normal Conjuntiva (FNC)
OBJETIVO:	E é satisfatível?

Conjunto de **variáveis booleanas** $x_1, x_2, \ldots$ e suas negações $\overline{x_1}, \overline{x_2}, \ldots$.

Satisfatibilidade

DADOS:	Uma expressão booleana E na Forma Normal Conjuntiva (FNC)
OBJETIVO:	E é satisfatível?

Conjunto de **variáveis booleanas** $x_1, x_2, \ldots$ e suas negações $\overline{x_1}, \overline{x_2}, \ldots$.
Também chamados de **literais**.

Satisfatibilidade

DADOS:	Uma expressão booleana E na Forma Normal Conjuntiva (FNC)
OBJETIVO:	E é satisfatível?

Conjunto de **variáveis booleanas** $x_1, x_2, \ldots$ e suas negações $\overline{x_1}, \overline{x_2}, \ldots$.
Também chamados de **literais**.

x_i pode ser <u>verdadeiro</u> ou <u>falso</u>
x_i é verdadeiro se e somente se $\bar{x}_i$ é falso

$\wedge$ conjunção
$\vee$ disjunção

$\wedge$ conjunção
$\vee$ disjunção

$\wedge$	0	1
0	0	0
1	0	1

$\vee$	0	1
0	0	1
1	1	1

Uma **cláusula** é uma disjunção de literais.

Uma **cláusula** é uma disjunção de literais.

Exemplo

$$\overline{x_1} \vee x_2 \vee \overline{x_3} \vee x_4$$

Uma **cláusula** é uma disjunção de literais.

Exemplo

$$\overline{x_1} \vee x_2 \vee \overline{x_3} \vee x_4$$

Uma expressão booleana é dita na **forma normal conjuntiva (FNC)** quando for uma conjução de cláusulas.

Uma **cláusula** é uma disjunção de literais.

Exemplo

$$\overline{x_1} \vee x_2 \vee \overline{x_3} \vee x_4$$

Uma expressão booleana é dita na **forma normal conjuntiva (FNC)** quando for uma conjução de cláusulas.

Exemplo

$$(x_1 \vee \overline{x_2}) \wedge (\overline{x_1} \vee x_2 \vee \overline{x_3} \vee x_4) \wedge (\overline{x_4})$$

Definição

Uma expressão booleana é dita **satisfatível** *se existe uma atribuição de valores às suas variáveis de tal modo que o valor da expressão seja verdadeiro.*

Exemplo

$$(x_1 \vee \overline{x_2}) \wedge (\overline{x_1} \vee x_2 \vee \overline{x_3} \vee x_4) \wedge (\overline{x_4})$$

Definição

Uma expressão booleana é dita **satisfatível** *se existe uma atribuição de valores às suas variáveis de tal modo que o valor da expressão seja verdadeiro.*

Exemplo

$$(x_1 \vee \overline{x_2}) \wedge (\overline{x_1} \vee x_2 \vee \overline{x_3} \vee x_4) \wedge (\overline{x_4})$$

com $(x_1, x_2, x_3, x_4) = (1, 1, 0, 0)$ *temos*

$$(1 \vee 0) \wedge (0 \vee 1 \vee 1 \vee 1) \wedge (1) = 1$$

Exemplo

$$(x_1 \vee x_2) \wedge (x_1 \vee \overline{x_2}) \wedge (\overline{x_1} \vee x_2) \wedge (\overline{x_1} \vee \overline{x_2})$$

Exemplo

$$(x_1 \vee x_2) \wedge (x_1 \vee \overline{x_2}) \wedge (\overline{x_1} \vee x_2) \wedge (\overline{x_1} \vee \overline{x_2})$$

não é satisfatível.

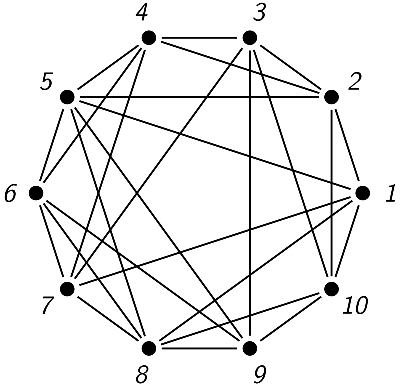
Clique

DADOS:	Um grafo G e um inteiro k
OBJETIVO:	G possui uma clique de tamanho pelo menos k?

Clique

DADOS: Um grafo G e um inteiro k
OBJETIVO: G possui uma clique de tamanho pelo menos k?

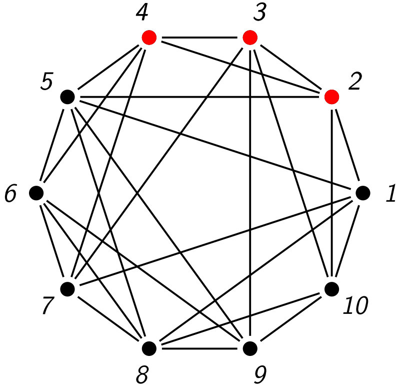
Exemplo



Clique

DADOS: Um grafo G e um inteiro k
OBJETIVO: G possui uma clique de tamanho pelo menos k?

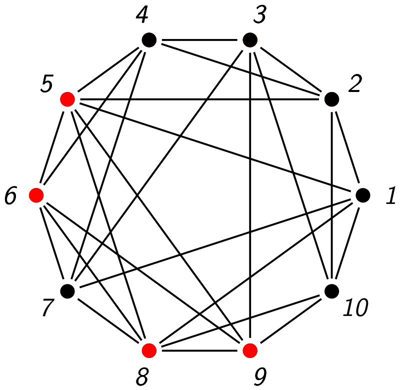
Exemplo



Clique

DADOS: Um grafo G e um inteiro k
OBJETIVO: G possui uma clique de tamanho pelo menos k?

Exemplo



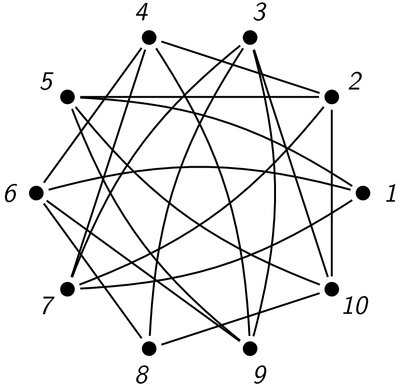
Conjunto Independente de Vértices

DADOS:	Um grafo G e um inteiro k
OBJETIVO:	G possui um conjunto independente de vértices de tamanho pelo menos k?

Conjunto Independente de Vértices

DADOS:	Um grafo G e um inteiro k
OBJETIVO:	G possui um conjunto independente de vértices de tamanho pelo menos k?

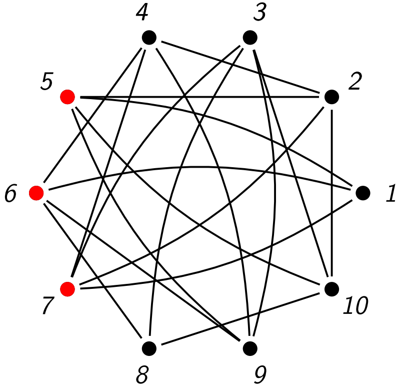
Exemplo



Conjunto Independente de Vértices

DADOS:	Um grafo G e um inteiro k
OBJETIVO:	G possui um conjunto independente de vértices de tamanho pelo menos k?

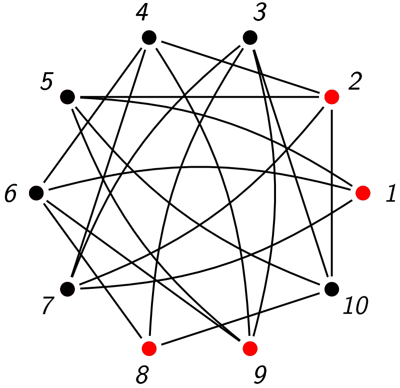
Exemplo



Conjunto Independente de Vértices

DADOS:	Um grafo G e um inteiro k
OBJETIVO:	G possui um conjunto independente de vértices de tamanho pelo menos k?

Exemplo



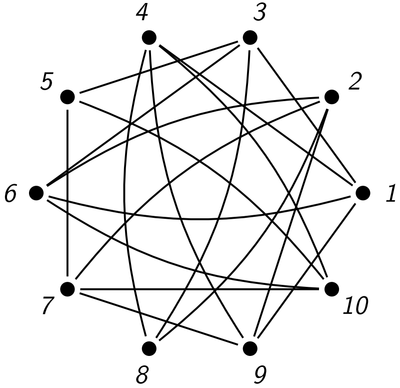
Cobertura por vértices

DADOS:	Um grafo G e um inteiro k
OBJETIVO:	G possui uma cobertura por vértices de tamanho no máximo k?

Cobertura por vértices

DADOS:	Um grafo G e um inteiro k
OBJETIVO:	G possui uma cobertura por vértices de tamanho no máximo k?

Exemplo

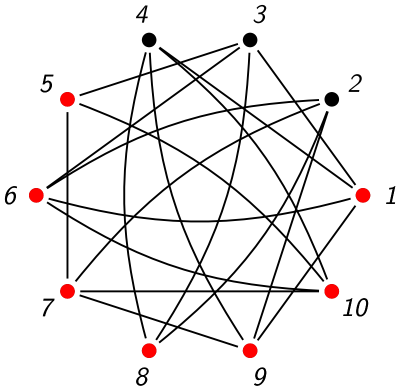


Cobertura por vértices

DADOS: Um grafo G e um inteiro k

OBJETIVO: G possui uma cobertura por vértices de tamanho no máximo k?

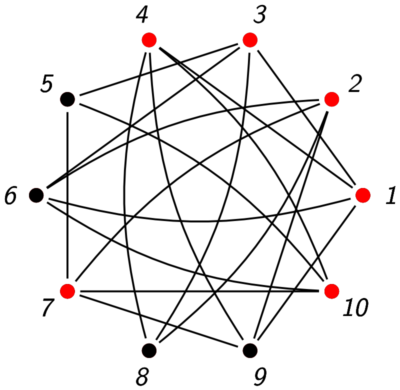
Exemplo



Cobertura por vértices

DADOS:	Um grafo G e um inteiro k
OBJETIVO:	G possui uma cobertura por vértices de tamanho no máximo k?

Exemplo



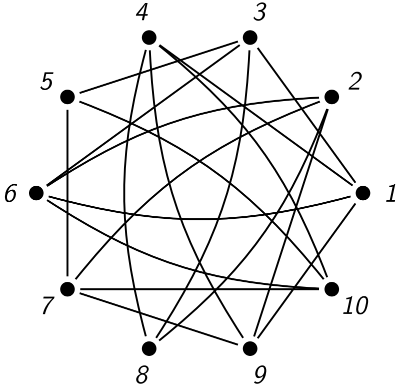
Circuito Hamiltoniano

DADOS:	Um grafo G
OBJETIVO:	G possui um Circuito Hamiltoniano?

Circuito Hamiltoniano

DADOS: Um grafo G
OBJETIVO: G possui um Circuito Hamiltoniano?

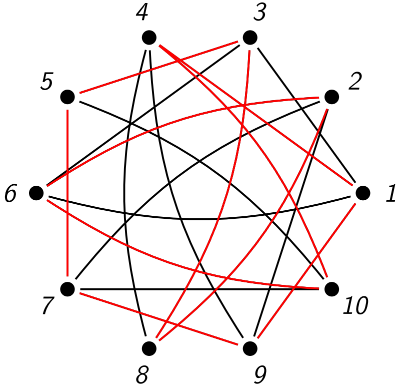
Exemplo



Circuito Hamiltoniano

DADOS: Um grafo G
OBJETIVO: G possui um Circuito Hamiltoniano?

Exemplo



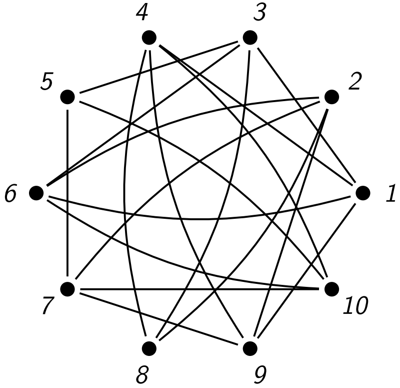
Coloração

DADOS:	Um grafo G e um inteiro k
OBJETIVO:	G admite uma coloração própria com k cores?

Coloração

DADOS:	Um grafo G e um inteiro k
OBJETIVO:	G admite uma coloração própria com k cores?

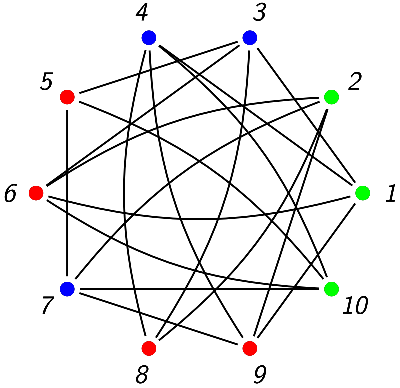
Exemplo



Coloração

DADOS: Um grafo G e um inteiro k
OBJETIVO: G admite uma coloração própria com k cores?

Exemplo



Aresta coloração

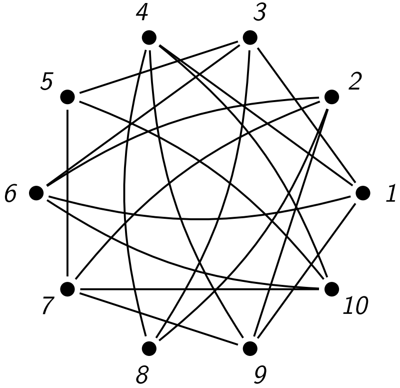
DADOS: Um grafo G

OBJETIVO: G admite uma aresta coloração própria com $\Delta(G)$ cores?

Aresta coloração

DADOS: Um grafo G
OBJETIVO: G admite uma aresta coloração própria com $\Delta(G)$ cores?

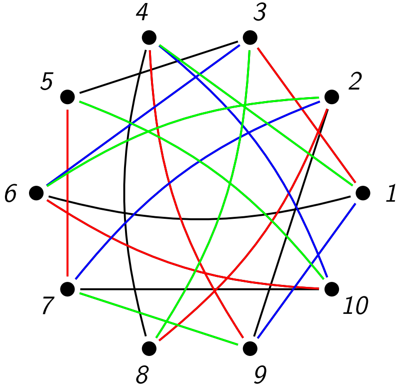
Exemplo



Aresta coloração

DADOS:	Um grafo G
OBJETIVO:	G admite uma aresta coloração própria com $\Delta(G)$ cores?

Exemplo



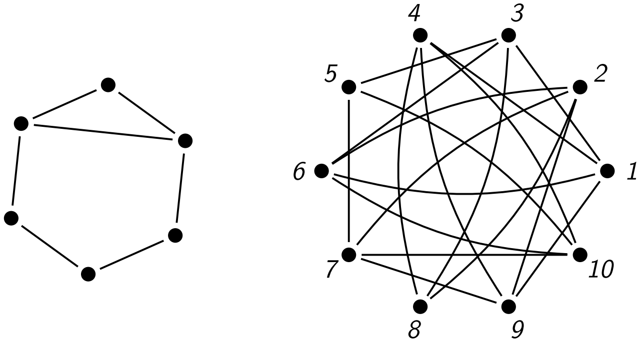
Isomorfismo de subgrafos

DADOS:	Grafo G_1 e G_2
OBJETIVO:	G_1 contém um subgrafo isomorfo a G_2?

Isomorfismo de subgrafos

DADOS: Grafo G_1 e G_2
OBJETIVO: G_1 contém um subgrafo isomorfo a G_2?

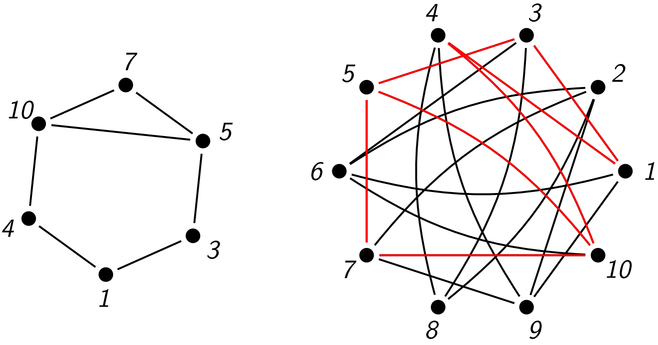
Exemplo



Isomorfismo de subgrafos

DADOS:	Grafo G_1 e G_2
OBJETIVO:	G_1 contém um subgrafo isomorfo a G_2?

Exemplo



Cobertura por conjuntos

DADOS: Um conjunto U, uma família $\mathcal{S}$ de subconjuntos de U, e um inteiro k

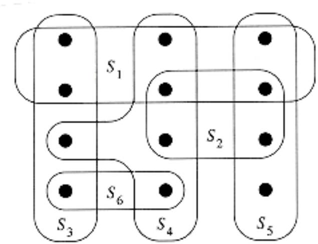
OBJETIVO: existe $\mathcal{S}' \subseteq \mathcal{S}$ tal que $|\mathcal{S}'| \leq k$ e $U \subseteq \displaystyle\bigcup_{S \in \mathcal{S}'} S$?

Cobertura por conjuntos

DADOS: Um conjunto U, uma família $\mathcal{S}$ de subconjuntos de U, e um inteiro k

OBJETIVO: existe $\mathcal{S}' \subseteq \mathcal{S}$ tal que $|\mathcal{S}'| \leq k$ e $U \subseteq \bigcup_{S \in \mathcal{S}'} S$?

Exemplo



Certificados

Um **certificado** para um problema Π é uma justificativa para a resposta SIM.

Um **co-certificado** para um problema Π é uma justificativa para a resposta NÃO.

Fase 1: **exibição** do certificado

Fase 2: **reconhecimento** do certificado

A Classe NP

$$NP = \left\{ \begin{array}{l} \text{problemas de decisão para os quais existe} \\ \text{certificado que pode ser reconhecido por} \\ \text{um algoritmo polinomial} \end{array} \right\}$$

Problemas NP-Completos

Fábio Botler

Programa de Engenharia de Sistemas e Computação
Universidade Federal do Rio de Janeiro