

# Projeto e Análise de Algoritmos II (MC558)

## Classes de Problemas

Prof. Dr. Ruben Interian

# Resumo

1 Revisão do conteúdo e objetivo

2 Provas de  $\mathcal{NP}$ -completude

3 Classe co- $\mathcal{NP}$

4 Síntese

## Resumo

- 1 Revisão do conteúdo e objetivo
  - 2 Provas de  $\mathcal{NP}$ -completude
  - 3 Classe co- $\mathcal{NP}$
  - 4 Síntese

# Revisão do conteúdo

- Um problema  $A$  é  $\text{NP}$ -difícil se todo problema de  $\text{NP}$  se reduz polinomialmente a  $A$ .
- Um problema  $A$  é  $\text{NP}$ -completo se  $A \in \text{NP}$  e  $A$  é  $\text{NP}$ -difícil.
- **SAT** é  $\text{NP}$ -completo.
- Para provar que outro problema  $A$  é  $\text{NP}$ -completo é necessário:
  - Provar que  $A$  é  $\text{NP}$ , e
  - Encontrar uma redução polinomial de um problema  $\text{NP}$ -completo para  $A$ .

# Objetivo

## Responder à pergunta:

- Como identificar um problema complexo? Como mostrar que ele é complexo?
  - “Eu não consigo resolver” não é suficiente!

# Resumo

1 Revisão do conteúdo e objetivo

2 Provas de  $\mathcal{NP}$ -completude

3 Classe co- $\mathcal{NP}$

4 Síntese

## Provas de $\mathcal{NP}$ -completude

$\mathcal{NP}$ -difícil

**A** é um problema  $\mathcal{NP}$ -difícil se todo problema de  $\mathcal{NP}$  se reduz polinomialmente a **A**.

## Provas de $\mathcal{NP}$ -completude

$\mathcal{NP}$ -difícil

**A** é um problema  $\text{NP}$ -difícil se todo problema de  $\text{NP}$  se reduz polinomialmente a **A**.

$\mathcal{NP}$ -completo

$A$  é um problema  $\mathcal{NP}$ -completo (ou  $A$  está em  $\mathcal{NP}$ -completo) se:

- 1  $A \in \mathcal{NP}$ , e
  - 2  $A \in \mathcal{NP}$ -difícil.

## Provas de $\mathcal{NP}$ -completude

$\mathcal{NP}$ -difícil

$A$  é um problema  $\mathcal{NP}$ -difícil se todo problema de  $\mathcal{NP}$  se reduz polinomialmente a  $A$ .

$\mathcal{NP}$ -completo

$A$  é um problema  $\mathcal{NP}$ -completo (ou  $A$  está em  $\mathcal{NP}$ -completo) se:

- ①  $A \in \mathcal{NP}$ , e
  - ②  $A \in \mathcal{NP}$ -difícil.

- Stephen Cook provou que **SAT** é *NP*-completo.

## Provas de $\mathcal{NP}$ -completude

$\mathcal{NP}$ -difícil

$A$  é um problema  $\mathcal{NP}$ -difícil se todo problema de  $\mathcal{NP}$  se reduz polinomialmente a  $A$ .

## $\mathcal{NP}$ -completo

$A$  é um problema  $\mathcal{NP}$ -completo (ou  $A$  está em  $\mathcal{NP}$ -completo) se:

- ①  $A \in \mathcal{NP}$ , e
  - ②  $A \in \mathcal{NP}$ -difícil.

- Stephen Cook provou que **SAT** é  $\text{NP}$ -completo.
  - Dessa forma, a porta foi aberta para provar que outros problemas são  $\text{NP}$ -completos, mostrando que eles são problemas complexos.

# Provas de $\mathcal{NP}$ -completude: CLI

## Problema do clique (CLI) Versão de decisão

Dado um grafo não-direcionado  $G = (V, E)$ , e um valor inteiro  $k$ , decidir se  $G$  contém uma clique com  $k$  vértices.

### Teorema

**CLI** é  $\mathcal{NP}$ -completo.

# Provas de $\mathcal{NP}$ -completude: CLI

## Problema do clique (CLI) Versão de decisão

Dado um grafo não-direcionado  $G = (V, E)$ , e um valor inteiro  $k$ , decidir se  $G$  contém uma clique com  $k$  vértices.

### Teorema

**CLI** é  $\mathcal{NP}$ -completo.

Precisamos provar que:

- **CLI**  $\in \mathcal{NP}$ .
- **CLI**  $\in \mathcal{NP}$ -difícil. Vamos provar isso mostrando que **SAT**  $\propto_{\text{poli}}$  **CLI**.

## Provas de $\mathcal{NP}$ -completude: CLI

Em primeiro lugar, veja que **CLI**  $\in \text{NP}$ , pois nas aulas anteriores vimos que existe um algoritmo não-determinístico polinomial que resolve **CLI**.

## Provas de $\mathcal{NP}$ -completude: CLI

Em primeiro lugar, veja que **CLI**  $\in \mathcal{NP}$ , pois nas aulas anteriores vimos que existe um algoritmo não-determinístico polinomial que resolve **CLI**.

- Outra forma de mostrar que **CLI**  $\in \mathcal{NP}$  é simplesmente dizer:  
*“CLI  $\in \mathcal{NP}$ , pois existe um algoritmo determinístico polinomial que, dado um conjunto de vértices  $C$ , verifica se  $C$  é uma clique em tempo quadrático no tamanho de  $C$ , checando se existem todas as arestas possíveis entre cada par de vértices  $v, w \in C$ .”*
- Aqui estamos usando a definição equivalente de  $\mathcal{NP}$  vista na aula anterior:  
 $A \in \mathcal{NP}$  se existe um algoritmo verificador determinístico e polinomial para  $A$ .

## Provas de $\mathcal{NP}$ -completude: CLI

Em primeiro lugar, veja que **CLI**  $\in \mathcal{NP}$ , pois nas aulas anteriores vimos que existe um algoritmo não-determinístico polinomial que resolve **CLI**.

- Outra forma de mostrar que **CLI**  $\in \mathcal{NP}$  é simplesmente dizer:  
*“CLI  $\in \mathcal{NP}$ , pois existe um algoritmo determinístico polinomial que, dado um conjunto de vértices  $C$ , verifica se  $C$  é uma clique em tempo quadrático no tamanho de  $C$ , checando se existem todas as arestas possíveis entre cada par de vértices  $v, w \in C$ .”*
- Aqui estamos usando a definição equivalente de  $\mathcal{NP}$  vista na aula anterior:  
 $A \in \mathcal{NP}$  se existe um algoritmo verificador determinístico e polinomial para  $A$ .
- **Exemplo de justificativa errada:**  
*“CLI  $\in \mathcal{NP}$ , pois existe um algoritmo determinístico polinomial que, dado um conjunto de vértices, verifica se eles são uma solução.”*

## Provas de $\mathcal{NP}$ -completude: CLI

Em primeiro lugar, veja que **CLI**  $\in \mathcal{NP}$ , pois nas aulas anteriores vimos que existe um algoritmo não-determinístico polinomial que resolve **CLI**.

- Outra forma de mostrar que **CLI**  $\in \mathcal{NP}$  é simplesmente dizer:  
*“CLI  $\in \mathcal{NP}$ , pois existe um algoritmo determinístico polinomial que, dado um conjunto de vértices  $C$ , verifica se  $C$  é uma clique em tempo quadrático no tamanho de  $C$ , checando se existem todas as arestas possíveis entre cada par de vértices  $v, w \in C$ .”*
- Aqui estamos usando a definição equivalente de  $\mathcal{NP}$  vista na aula anterior:  
 $A \in \mathcal{NP}$  se existe um algoritmo verificador determinístico e polinomial para  $A$ .
- **Exemplo de justificativa errada:**  
*“CLI  $\in \mathcal{NP}$ , pois existe um algoritmo determinístico polinomial que, dado um conjunto de vértices, verifica se eles são uma solução.”*

Agora precisamos provar que **CLI**  $\in \mathcal{NP}$ -difícil. Vamos mostrar que  $SAT \propto_{\text{poli}} CLI$ .

## Provas de $\mathcal{NP}$ -completude: CLI

**Definição:** Um grafo  $G = (V, E)$  é  $t$ -partido se o conjunto de vértices pode ser particionado em  $t$  subconjuntos  $V_1, V_2, \dots, V_t$  tal que não há arestas em  $E$  ligando dois vértices em um mesmo subconjunto  $V_i$ , para todo  $i \in \{1, \dots, t\}$ .

# Provas de $\mathcal{NP}$ -completude: CLI

**Definição:** Um grafo  $G = (V, E)$  é  $t$ -partido se o conjunto de vértices pode ser particionado em  $t$  subconjuntos  $V_1, V_2, \dots, V_t$  tal que não há arestas em  $E$  ligando dois vértices em um mesmo subconjunto  $V_i$ , para todo  $i \in \{1, \dots, t\}$ .

**Transformação da instância do SAT em uma instância CLI.**

Seja  $\mathcal{F} = C_1 \cdot C_2 \cdot \dots \cdot C_m$  uma fórmula booleana com as variáveis  $x_1, \dots, x_n$ .

Construa o grafo  $m$ -partido  $G = ((V_1, V_2, \dots, V_m), E)$  de forma que:

- Em cada subconjunto  $V_i$  haverá um vértice  $v_j$  associado a cada uma das variáveis  $x_j$  que aparecem na cláusula  $C_i$  de  $\mathcal{F}$ ;

## Provas de $\mathcal{NP}$ -completude: CLI

**Definição:** Um grafo  $G = (V, E)$  é  $t$ -partido se o conjunto de vértices pode ser particionado em  $t$  subconjuntos  $V_1, V_2, \dots, V_t$  tal que não há arestas em  $E$  ligando dois vértices em um mesmo subconjunto  $V_i$ , para todo  $i \in \{1, \dots, t\}$ .

## Transformação da instância do SAT em uma instância CLI.

Seja  $\mathcal{F} = C_1 \cdot C_2 \cdot \dots \cdot C_m$  uma fórmula booleana com as variáveis  $x_1, \dots, x_n$ .

Construa o grafo  $m$ -partido  $G = ((V_1, V_2, \dots, V_m), E)$  de forma que:

- Em cada subconjunto  $V_i$  haverá um vértice  $v_j$  associado a cada uma das variáveis  $x_j$  que aparecem na cláusula  $C_i$  de  $\mathcal{F}$ ;
  - A aresta  $(v, w)$  está em  **$E$  se e somente se** duas condições são cumpridas:
    - ①  $v$  e  $w$  estão em subconjuntos distintos, e
    - ②  $v$  e  $w$  não representam simultaneamente uma variável e a sua negação.

# Provas de $\mathcal{NP}$ -completude: CLI

Analizando a **complexidade** da redução:

- A instância do **SAT** possui  $m$  cláusulas, cada uma com no máximo  $n$  variáveis.
- O número de vértices de  $G$  é  $O(m \cdot n)$ .
- O número de arestas é  $O((m \cdot n)^2) = O(m^2 n^2)$ .

# Provas de $\mathcal{NP}$ -completude: CLI

Analizando a **complexidade** da redução:

- A instância do **SAT** possui  $m$  cláusulas, cada uma com no máximo  $n$  variáveis.
- O número de vértices de  $G$  é  $O(m \cdot n)$ .
- O número de arestas é  $O((m \cdot n)^2) = O(m^2 n^2)$ .

A instância de **CLI** é: um grafo não-direcionado  $G = (V, E)$ , e um inteiro  $k$ .

Vamos fazer  $k = m$ .

# Provas de $\mathcal{NP}$ -completude: CLI

Analizando a **complexidade** da redução:

- A instância do **SAT** possui  $m$  cláusulas, cada uma com no máximo  $n$  variáveis.
- O número de vértices de  $G$  é  $O(m \cdot n)$ .
- O número de arestas é  $O((m \cdot n)^2) = O(m^2 n^2)$ .

A instância de **CLI** é: um grafo não-direcionado  $G = (V, E)$ , e um inteiro  $k$ .

Vamos fazer  $k = m$ .

Temos uma instância de **CLI**:

- De tamanho polinomial no tamanho da entrada de **SAT**.
- Criada em tempo polinomial no tamanho da entrada de **SAT**.

# Provas de $\mathcal{NP}$ -completude: CLI

Analizando a **complexidade** da redução:

- A instância do **SAT** possui  $m$  cláusulas, cada uma com no máximo  $n$  variáveis.
- O número de vértices de  $G$  é  $O(m \cdot n)$ .
- O número de arestas é  $O((m \cdot n)^2) = O(m^2 n^2)$ .

A instância de **CLI** é: um grafo não-direcionado  $G = (V, E)$ , e um inteiro  $k$ .

Vamos fazer  $k = m$ .

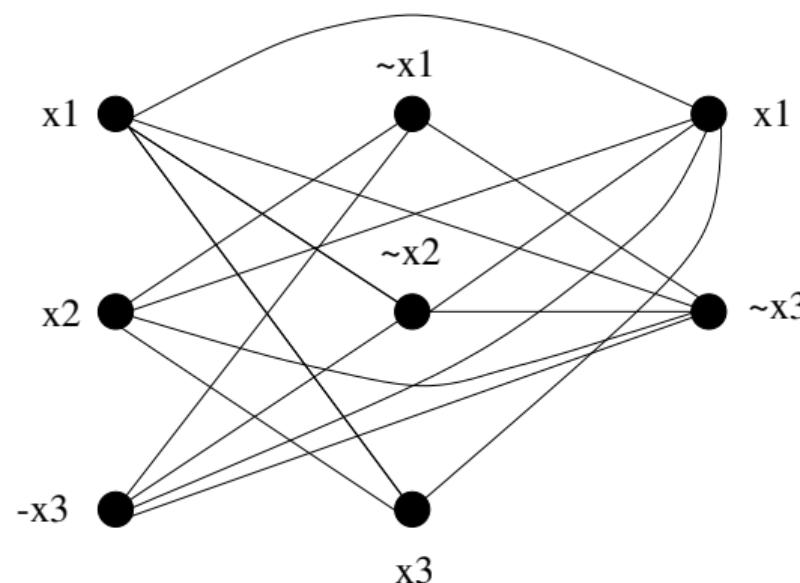
Temos uma instância de **CLI**:

- De tamanho polinomial no tamanho da entrada de **SAT**.
- Criada em tempo polinomial no tamanho da entrada de **SAT**.

Mostraremos que a fórmula  $\mathcal{F}$  é satisfatível por alguma atribuição de variáveis se e somente se o grafo  $m$ -partido  $G$  tem uma clique de tamanho  $m$ .

# Provas de $\mathcal{NP}$ -completude: CLI

**Exemplo** da redução: Seja  $\mathcal{F} = (x_1 + x_2 + \bar{x}_3) \cdot (\bar{x}_1 + \bar{x}_2 + x_3) \cdot (x_1 + \bar{x}_3)$ . O grafo correspondente à instância de **CLI** é dado por:



# Provas de $\mathcal{NP}$ -completude: CLI

Precisamos **sempre** justificar porque a nossa redução é válida:

A fórmula  $\mathcal{F}$  do **SAT** é satisfatível

**se e somente se**

o grafo  $m$ -partido gerado possui uma clique de tamanho  $m$ .

# Provas de $\mathcal{NP}$ -completude: CLI

Precisamos **sempre** justificar porque a nossa redução é válida:

A fórmula  $\mathcal{F}$  do **SAT** é satisfatível

**se e somente se**

o grafo  $m$ -partido gerado possui uma clique de tamanho  $m$ .

- A fórmula  $\mathcal{F}$  do **SAT** é satisfatível  
⇒ para cada cláusula  $C_i$  de  $\mathcal{F}$  existe uma variável que faz  $C_i$  ser verdadeira;

# Provas de $\mathcal{NP}$ -completude: CLI

Precisamos **sempre** justificar porque a nossa redução é válida:

A fórmula  $\mathcal{F}$  do **SAT** é satisfatível

**se e somente se**

o grafo  $m$ -partido gerado possui uma clique de tamanho  $m$ .

- A fórmula  $\mathcal{F}$  do **SAT** é satisfatível
  - ⇒ para cada cláusula  $C_i$  de  $\mathcal{F}$  existe uma variável que faz  $C_i$  ser verdadeira;
  - ⇒ esse conjunto de  $m$  variáveis correspondem a um vértice diferente em cada  $V_i$ ;

# Provas de $\mathcal{NP}$ -completude: CLI

Precisamos **sempre** justificar porque a nossa redução é válida:

A fórmula  $\mathcal{F}$  do **SAT** é satisfatível

**se e somente se**

o grafo  $m$ -partido gerado possui uma clique de tamanho  $m$ .

- A fórmula  $\mathcal{F}$  do **SAT** é satisfatível
  - ⇒ para cada cláusula  $C_i$  de  $\mathcal{F}$  existe uma variável que faz  $C_i$  ser verdadeira;
  - ⇒ esse conjunto de  $m$  variáveis correspondem a um vértice diferente em cada  $V_i$ ;
  - ⇒ nesse conjunto, não há duas variáveis que sejam uma negação da outra;

# Provas de $\mathcal{NP}$ -completude: CLI

Precisamos **sempre** justificar porque a nossa redução é válida:

A fórmula  $\mathcal{F}$  do **SAT** é satisfatível

**se e somente se**

o grafo  $m$ -partido gerado possui uma clique de tamanho  $m$ .

- A fórmula  $\mathcal{F}$  do **SAT** é satisfatível
  - ⇒ para cada cláusula  $C_i$  de  $\mathcal{F}$  existe uma variável que faz  $C_i$  ser verdadeira;
  - ⇒ esse conjunto de  $m$  variáveis correspondem a um vértice diferente em cada  $V_i$ ;
  - ⇒ nesse conjunto, não há duas variáveis que sejam uma negação da outra;
  - ⇒ todos os  $m$  vértices correspondentes às  $m$  variáveis estão ligados por arestas;

# Provas de $\mathcal{NP}$ -completude: CLI

Precisamos **sempre** justificar porque a nossa redução é válida:

A fórmula  $\mathcal{F}$  do **SAT** é satisfatível

**se e somente se**

o grafo  $m$ -partido gerado possui uma clique de tamanho  $m$ .

- A fórmula  $\mathcal{F}$  do **SAT** é satisfatível
  - ⇒ para cada cláusula  $C_i$  de  $\mathcal{F}$  existe uma variável que faz  $C_i$  ser verdadeira;
  - ⇒ esse conjunto de  $m$  variáveis correspondem a um vértice diferente em cada  $V_i$ ;
  - ⇒ nesse conjunto, não há duas variáveis que sejam uma negação da outra;
  - ⇒ todos os  $m$  vértices correspondentes às  $m$  variáveis estão ligados por arestas;
  - ⇒ este conjunto de  $m$  vértices é uma clique de tamanho  $m$ .

# Provas de $\mathcal{NP}$ -completude: CLI

Precisamos **sempre** justificar porque a nossa redução é **válida**:

A fórmula  $\mathcal{F}$  do **SAT** é satisfatível

**se e somente se**

o grafo  $m$ -partido gerado possui uma clique de tamanho  $m$ .

- O grafo  $m$ -partido gerado possui uma clique de tamanho  $m$   
⇒ todos os  $m$  vértices da clique estão ligados por arestas;

# Provas de $\mathcal{NP}$ -completude: CLI

Precisamos **sempre** justificar porque a nossa redução é **válida**:

A fórmula  $\mathcal{F}$  do **SAT** é satisfatível

**se e somente se**

o grafo  $m$ -partido gerado possui uma clique de tamanho  $m$ .

- O grafo  $m$ -partido gerado possui uma clique de tamanho  $m$ 
  - ⇒ todos os  $m$  vértices da clique estão ligados por arestas;
  - ⇒ cada vértice da clique está em um conjunto  $V_i$  diferente pois  $G$  é  $m$ -partido;

# Provas de $\mathcal{NP}$ -completude: CLI

Precisamos **sempre** justificar porque a nossa redução é **válida**:

A fórmula  $\mathcal{F}$  do **SAT** é satisfatível

**se e somente se**

o grafo  $m$ -partido gerado possui uma clique de tamanho  $m$ .

- O grafo  $m$ -partido gerado possui uma clique de tamanho  $m$ 
  - ⇒ todos os  $m$  vértices da clique estão ligados por arestas;
  - ⇒ cada vértice da clique está em um conjunto  $V_i$  diferente pois  $G$  é  $m$ -partido;
  - ⇒ nenhum par de vértices da clique foi gerado a partir de variáveis de  $\mathcal{F}$  que sejam uma negação da outra, pois nesse caso não haveria aresta;

## Provas de $\mathcal{NP}$ -completude: CLI

Precisamos **sempre** justificar porque a nossa redução é **válida**:

A fórmula  $\mathcal{F}$  do **SAT** é satisfatível

**se e somente se**

o grafo  $m$ -partido gerado possui uma clique de tamanho  $m$ .

- O grafo  $m$ -partido gerado possui uma clique de tamanho  $m$ 
  - ⇒ todos os  $m$  vértices da clique estão ligados por arestas;
  - ⇒ cada vértice da clique está em um conjunto  $V_i$  diferente pois  $G$  é  $m$ -partido;
  - ⇒ nenhum par de vértices da clique foi gerado a partir de variáveis de  $\mathcal{F}$  que sejam uma negação da outra, pois nesse caso não haveria aresta;
  - ⇒ em cada  $C_i$ , se  $v_j \in V_i$  é escolhido, podemos atribuir **1** à variável  $x_j$ , e **0** a  $\bar{x}_j$ ;

# Provas de $\mathcal{NP}$ -completude: CLI

Precisamos **sempre** justificar porque a nossa redução é **válida**:

A fórmula  $\mathcal{F}$  do **SAT** é satisfatível

**se e somente se**

o grafo  $m$ -partido gerado possui uma clique de tamanho  $m$ .

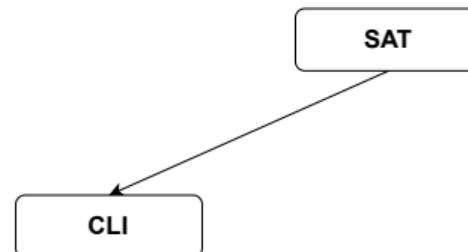
- O grafo  $m$ -partido gerado possui uma clique de tamanho  $m$ 
  - ⇒ todos os  $m$  vértices da clique estão ligados por arestas;
  - ⇒ cada vértice da clique está em um conjunto  $V_i$  diferente pois  $G$  é  $m$ -partido;
  - ⇒ nenhum par de vértices da clique foi gerado a partir de variáveis de  $\mathcal{F}$  que sejam uma negação da outra, pois nesse caso não haveria aresta;
  - ⇒ em cada  $C_i$ , se  $v_j \in V_i$  é escolhido, podemos atribuir **1** à variável  $x_j$ , e **0** a  $\bar{x}_j$ ;
  - ⇒ a fórmula  $\mathcal{F}$  do **SAT** é satisfatível.

# Provas de $\mathcal{NP}$ -completude

**Resumo:** “algoritmo” para provar, **na prática**, a  $\mathcal{NP}$ -completude de um problema  $A$ :

- $A \in \mathcal{NP}$ : mostrar que  $\exists$  algoritmo verificador determinístico polinomial para  $A$ .
- $A \in \mathcal{NP}$ -difícil: apresentar uma redução de um problema  $\mathcal{NP}$ -completo para  $A$ :
  - **Escolher** um problema adequado  $B$  que seja  $\mathcal{NP}$ -completo.
  - Apresentar um **algoritmo da redução** (de Karp) de  $B$  para  $A$ .
  - Justificar porque a redução apresentada é **válida**. Ou seja, mostrar que:
    - Se temos uma instância SIM para  $B$ , geramos uma instância SIM para  $A$ .
    - Se a instância gerada é SIM para  $A$ , ela é gerada a partir de uma instância SIM para  $B$ .

# Provas de $\mathcal{NP}$ -completude: nosso estado da arte



# Provas de $\mathcal{NP}$ -completude: PLI

## Problema de Programação Linear Inteira (PLI) Versão de decisão

**Entrada:** matriz  $A_{m \times n}$ , vetor  $b \in \mathbb{R}^m$ , vetor  $c \in \mathbb{R}^n$ , valor real  $K$ .

**Objetivo:** encontrar valores para as variáveis inteiras não negativas  $x_1, x_2, \dots, x_n = x$  tais que  $c^T x \leq K$ , respeitando às restrições  $Ax \leq b$ .

$$\begin{aligned}c^T x &\leq K, \\Ax &\leq b, x \geq 0\end{aligned}$$

### Teorema

**PLI** é  $\mathcal{NP}$ -completo.

# Provas de $\mathcal{NP}$ -completude: PLI

Precisamos provar que:

- **PLI**  $\in \mathcal{NP}$ . Existe um algoritmo verificador determinístico para **PLI**: dado um **certificado conciso** (o valor de cada variável), é suficiente checar em tempo linear se cada restrição é cumprida.

# Provas de $\mathcal{NP}$ -completude: PLI

Precisamos provar que:

- **PLI**  $\in \mathcal{NP}$ . Existe um algoritmo verificador determinístico para **PLI**: dado um **certificado conciso** (o valor de cada variável), é suficiente checar em tempo linear se cada restrição é cumprida.
- **PLI**  $\in \mathcal{NP}$ -difícil.
  - A partir de qual problema faremos a redução?  
Temos duas opções: **SAT** e **CLI**.

# Provas de $\mathcal{NP}$ -completude: PLI

Precisamos provar que:

- **PLI**  $\in \mathcal{NP}$ . Existe um algoritmo verificador determinístico para **PLI**: dado um **certificado conciso** (o valor de cada variável), é suficiente checar em tempo linear se cada restrição é cumprida.
- **PLI**  $\in \mathcal{NP}$ -difícil.
  - A partir de qual problema faremos a redução?  
Temos duas opções: **SAT** e **CLI**.

Vamos provar que **PLI**  $\in \mathcal{NP}$ -difícil mostrando que **SAT**  $\propto_{\text{poli}}$  **PLI**.

# Provas de $\mathcal{NP}$ -completude: PLI

**Transformação da instância do SAT em uma instância PLI.**

Seja  $\mathcal{F} = C_1 \cdot C_2 \cdot \dots \cdot C_m$  uma fórmula booleana com as variáveis  $x_1, \dots, x_n$ .

A instância de **PLI** terá as variáveis  $y_i$ , a mesma quantidade de variáveis da instância do **SAT**. Vamos construir um conjunto de restrições do problema de **PLI**:

# Provas de $\mathcal{NP}$ -completude: PLI

**Transformação da instância do SAT em uma instância PLI.**

Seja  $\mathcal{F} = C_1 \cdot C_2 \cdot \dots \cdot C_m$  uma fórmula booleana com as variáveis  $x_1, \dots, x_n$ .

A instância de **PLI** terá as variáveis  $y_i$ , a mesma quantidade de variáveis da instância do **SAT**. Vamos construir um conjunto de restrições do problema de **PLI**:

- Para cada cláusula  $C_i = x_a + x_b + \dots$ , criaremos a restrição  $\sum_{i=1}^{|C_i|} p_i \geq 1$ :
  - $p_i = y_a$  se a variável  $x_a$  aparece não negada em  $C_i$ , e
  - $p_i = 1 - y_a$  se a variável  $x_a$  aparece negada em  $C_i$ .

# Provas de $\mathcal{NP}$ -completude: PLI

**Transformação da instância do SAT em uma instância PLI.**

Seja  $\mathcal{F} = C_1 \cdot C_2 \cdot \dots \cdot C_m$  uma fórmula booleana com as variáveis  $x_1, \dots, x_n$ .

A instância de **PLI** terá as variáveis  $y_i$ , a mesma quantidade de variáveis da instância do **SAT**. Vamos construir um conjunto de restrições do problema de **PLI**:

- Para cada cláusula  $C_i = x_a + x_b + \dots$ , criaremos a restrição  $\sum_{i=1}^{|C_i|} p_i \geq 1$ :
  - $p_i = y_a$  se a variável  $x_a$  aparece não negada em  $C_i$ , e
  - $p_i = 1 - y_a$  se a variável  $x_a$  aparece negada em  $C_i$ .
  - Exemplo:**  $x_1 \vee \bar{x}_3 \vee x_7 \Rightarrow y_1 + (1 - y_3) + y_7 \geq 1$ .

# Provas de $\mathcal{NP}$ -completude: PLI

**Transformação da instância do SAT em uma instância PLI.**

Seja  $\mathcal{F} = C_1 \cdot C_2 \cdot \dots \cdot C_m$  uma fórmula booleana com as variáveis  $x_1, \dots, x_n$ .

A instância de **PLI** terá as variáveis  $y_i$ , a mesma quantidade de variáveis da instância do **SAT**. Vamos construir um conjunto de restrições do problema de **PLI**:

- Para cada cláusula  $C_i = x_a + x_b + \dots$ , criaremos a restrição  $\sum_{i=1}^{|C_i|} p_i \geq 1$ :
  - $p_i = y_a$  se a variável  $x_a$  aparece não negada em  $C_i$ , e
  - $p_i = 1 - y_a$  se a variável  $x_a$  aparece negada em  $C_i$ .
  - **Exemplo:**  $x_1 \vee \bar{x}_3 \vee x_7 \Rightarrow y_1 + (1 - y_3) + y_7 \geq 1$ .
- Além disso, para cada variável  $x_i$  do **SAT**, criaremos uma restrição  $y_i \leq 1$ .

# Provas de $\mathcal{NP}$ -completude: PLI

**Transformação da instância do SAT em uma instância PLI.**

Seja  $\mathcal{F} = C_1 \cdot C_2 \cdot \dots \cdot C_m$  uma fórmula booleana com as variáveis  $x_1, \dots, x_n$ .

A instância de **PLI** terá as variáveis  $y_i$ , a mesma quantidade de variáveis da instância do **SAT**. Vamos construir um conjunto de restrições do problema de **PLI**:

- Para cada cláusula  $C_i = x_a + x_b + \dots$ , criaremos a restrição  $\sum_{i=1}^{|C_i|} p_i \geq 1$ :
  - $p_i = y_a$  se a variável  $x_a$  aparece não negada em  $C_i$ , e
  - $p_i = 1 - y_a$  se a variável  $x_a$  aparece negada em  $C_i$ .
  - **Exemplo:**  $x_1 \vee \bar{x}_3 \vee x_7 \Rightarrow y_1 + (1 - y_3) + y_7 \geq 1$ .
- Além disso, para cada variável  $x_i$  do **SAT**, criaremos uma restrição  $y_i \leq 1$ .
- Criaremos as restrições de não-negatividade  $y_i \geq 0$ .

# Provas de $\mathcal{NP}$ -completude: PLI

**Transformação da instância do SAT em uma instância PLI.**

Seja  $\mathcal{F} = C_1 \cdot C_2 \cdot \dots \cdot C_m$  uma fórmula booleana com as variáveis  $x_1, \dots, x_n$ .

A instância de **PLI** terá as variáveis  $y_i$ , a mesma quantidade de variáveis da instância do **SAT**. Vamos construir um conjunto de restrições do problema de **PLI**:

- Para cada cláusula  $C_i = x_a + x_b + \dots$ , criaremos a restrição  $\sum_{i=1}^{|C_i|} p_i \geq 1$ :
  - $p_i = y_a$  se a variável  $x_a$  aparece não negada em  $C_i$ , e
  - $p_i = 1 - y_a$  se a variável  $x_a$  aparece negada em  $C_i$ .
  - **Exemplo:**  $x_1 \vee \bar{x}_3 \vee x_7 \Rightarrow y_1 + (1 - y_3) + y_7 \geq 1$ .
- Além disso, para cada variável  $x_i$  do **SAT**, criaremos uma restrição  $y_i \leq 1$ .
- Criaremos as restrições de não-negatividade  $y_i \geq 0$ .

A redução é claramente polinomial no tamanho da entrada do **SAT**.

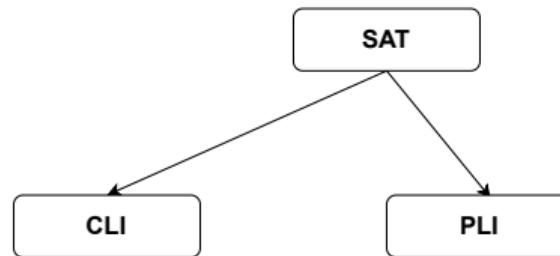
# Provas de $\mathcal{NP}$ -completude: PLI

A fórmula  $\mathcal{F}$  do **SAT** é satisfatível  
**se e somente se**  
a formulação de **PLI** gerada possui uma solução.

**Exercício:** Justificar porque a redução apresentada é **válida**. Ou seja, mostrar que:

- Se temos uma instância SIM para **SAT**, geramos uma instância SIM para **PLI**.
- Se a instância gerada é SIM para **PLI**, ela é gerada de instância SIM para **SAT**.

# Provas de $\mathcal{NP}$ -completude: nosso estado da arte



# Provas de $\mathcal{NP}$ -completude: IND

## Problema do conjunto independente (IND) Versão de decisão

Dado um grafo não-direcionado  $G = (V, E)$ , e um valor inteiro  $k$ , decidir se  $G$  contém um conjunto independente com  $k$  vértices.

**Lembrando:** Um conjunto independente é conjunto de vértices  $S \subseteq V$  tal que não existem dois vértices adjacentes contidos em  $S$ .

### Teorema

IND é  $\mathcal{NP}$ -completo.

# Provas de $\mathcal{NP}$ -completude: IND

- **IND**  $\in \mathcal{NP}$ , pois existe um algoritmo determinístico que, dado um conjunto de vértices  $C$  de um grafo, verifica, em tempo polinomial, se eles são um conjunto independente checando se não há nenhuma aresta entre eles.

# Provas de $\mathcal{NP}$ -completude: IND

- **IND**  $\in \mathcal{NP}$ , pois existe um algoritmo determinístico que, dado um conjunto de vértices **C** de um grafo, verifica, em tempo polinomial, se eles são um conjunto independente checando se não há nenhuma aresta entre eles.
- **IND**  $\in \mathcal{NP}$ -difícil.
  - A partir de qual problema faremos a redução?  
Temos três opções: **SAT**, **CLI** e **PLI**.

# Provas de $\mathcal{NP}$ -completude: IND

- **IND**  $\in \mathcal{NP}$ , pois existe um algoritmo determinístico que, dado um conjunto de vértices **C** de um grafo, verifica, em tempo polinomial, se eles são um conjunto independente checando se não há nenhuma aresta entre eles.
- **IND**  $\in \mathcal{NP}$ -difícil.
  - A partir de qual problema faremos a redução?  
Temos três opções: **SAT**, **CLI** e **PLI**.

Vamos provar que **PLI**  $\in \mathcal{NP}$ -difícil mostrando que **CLI**  $\propto_{\text{poli}}$  **IND**.

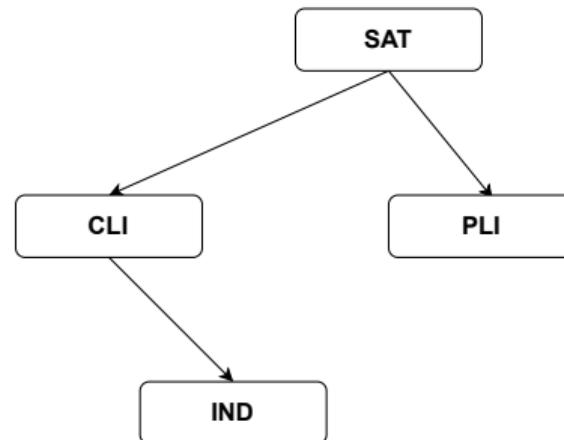
# Provas de $\mathcal{NP}$ -completude: IND

**Transformação da instância do CLI em uma instância do IND.**

Seja  $G = (V, E)$ ,  $k$  uma instância do **CLI**. Construa uma instância do **IND** formada pelo grafo complementar  $\bar{G} = (V, \bar{E})$ , e o mesmo inteiro  $k$ .

- **Lembrando:** as arestas em  $\bar{G}$  são as “inversas”:  $e \notin \bar{E}$  se e somente se  $e \in E$ .
- A redução é **polinomial**.
- A redução é **válida**:  $G, k$  é SIM para **CLI**  $\Leftrightarrow \bar{G}, k$  é SIM para **IND**.
  - $\Rightarrow$  Se há uma clique  $S$  de tamanho  $k$  em  $G$ ,  $S$  será um conjunto independente de tamanho  $k$  em  $\bar{G}$ , pois  $e \in E \Rightarrow e \notin \bar{E}$ .
  - $\Leftarrow$  Se há um conjunto independente  $S$  de tamanho  $k$  em  $\bar{G}$ ,  $S$  é uma clique de tamanho  $k$  em  $G$ , pois  $e \notin \bar{E} \Rightarrow e \in E$ .

# Provas de $\mathcal{NP}$ -completude: nosso estado da arte



# Provas de $\mathcal{NP}$ -completude: **CV**

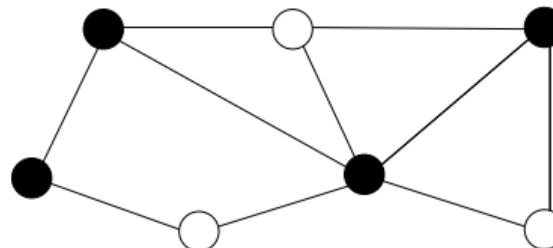
## Problema da cobertura de vértices (CV) Versão de decisão

Dado um grafo não-direcionado  $G = (V, E)$ , e um valor inteiro  $k$ , decidir se  $G$  contém uma cobertura de vértices com  $k$  vértices.

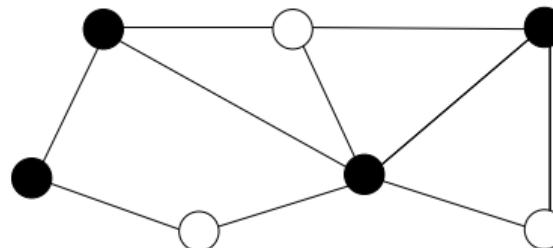
**Definição:** Uma cobertura de vértices em um grafo  $G = (V, E)$  é um conjunto de vértices  $S \subseteq V$  tal que toda aresta de  $E$  tem pelo menos um dos seus extremos em  $S$ .

### Teorema

**CV** é  $\mathcal{NP}$ -completo.

Provas de  $\mathcal{NP}$ -completude: **CV****Observações e interpretação:**

- Encontrar uma cobertura de vértices bem grande é fácil ( $V$  é uma cobertura). Difícil é encontrar uma cobertura com poucos vértices.
- Encontrar uma clique bem pequena é fácil: qualquer vértice é uma clique de tamanho 1, e os extremos de qualquer aresta formam uma clique de tamanho 2.

Provas de  $\mathcal{NP}$ -completude: **CV****Observações e interpretação:**

- Encontrar uma cobertura de vértices bem grande é fácil ( $V$  é uma cobertura). Difícil é encontrar uma cobertura com poucos vértices.
- Encontrar uma clique bem pequena é fácil: qualquer vértice é uma clique de tamanho 1, e os extremos de qualquer aresta formam uma clique de tamanho 2.
- **CV**: geralmente temos interesse na menor cobertura (**CV** mínima).
- **CLI**: geralmente temos interesse na maior clique (**CLI** máxima).

# Provas de $\mathcal{NP}$ -completude: **CV**

- **CV**  $\in \mathcal{NP}$ , pois existe um algoritmo determinístico que, dado uma conjunto de vértices  $S$  de um grafo, verifica, em tempo polinomial, se eles são uma cobertura de vértices checando se cada aresta tem pelo menos um extremo em  $S$ .

Provas de  $\mathcal{NP}$ -completude: **CV**

- **CV**  $\in \mathcal{NP}$ , pois existe um algoritmo determinístico que, dado uma conjunto de vértices  $S$  de um grafo, verifica, em tempo polinomial, se eles são uma cobertura de vértices checando se cada aresta tem pelo menos um extremo em  $S$ .
- **CV**  $\in \mathcal{NP}$ -difícil.
  - A partir de qual problema faremos a redução? Temos diversas opções!

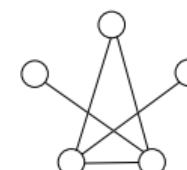
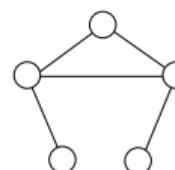
# Provas de $\mathcal{NP}$ -completude: **CV**

- **CV**  $\in \mathcal{NP}$ , pois existe um algoritmo determinístico que, dado uma conjunto de vértices  $S$  de um grafo, verifica, em tempo polinomial, se eles são uma cobertura de vértices checando se cada aresta tem pelo menos um extremo em  $S$ .
- **CV**  $\in \mathcal{NP}$ -difícil.
  - A partir de qual problema faremos a redução? Temos diversas opções!

Vamos provar que **CV**  $\in \mathcal{NP}$ -difícil mostrando que **CLI**  $\propto_{\text{poli}}$  **CV**.

Provas de  $\mathcal{NP}$ -completude: **CV**

**Atenção! Para algumas pessoas, esta parte da prova pode parecer confusa!**



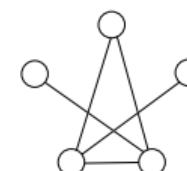
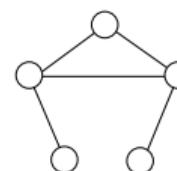
**Transformação da instância do CLI em uma instância CV.**

Seja  $G = (V, E)$ ,  $k$  uma instância do **CLI**. Construa uma instância do **CV** formada pelo grafo complementar  $\bar{G} = (\bar{V}, \bar{E})$ , e um inteiro (**diferente!**)  $n - k$ , onde  $n = |V|$ .

- A redução é **polinomial**.

Provas de  $\mathcal{NP}$ -completude: **CV**

**Atenção! Para algumas pessoas, esta parte da prova pode parecer confusa!**



**Transformação da instância do CLI em uma instância CV.**

Seja  $G = (V, E)$ ,  $k$  uma instância do **CLI**. Construa uma instância do **CV** formada pelo grafo complementar  $\bar{G} = (V, \bar{E})$ , e um inteiro (**diferente!**)  $n - k$ , onde  $n = |V|$ .

- A redução é **polinomial**.
- A redução é **válida**? Ou seja:

$G$ ,  $k$  é SIM para **CLI**  $\Leftrightarrow \bar{G}$ ,  $n - k$  é SIM para **CV**?

# Provas de $\mathcal{NP}$ -completude: **CV**

A redução **CLI**  $\propto_{\text{poli}}$  **CV** é válida. É possível mostrar que  $G, k$  é uma instância SIM de **CLI** se e somente se  $\overline{G}, n - k$  é uma instância SIM de **CV**.

## Proposição

O grafo  $G$  tem uma clique de tamanho  $k$  se e somente se o grafo complementar  $\overline{G}$  tem uma cobertura de tamanho  $n - k$ .

# Provas de $\mathcal{NP}$ -completude: **CV**

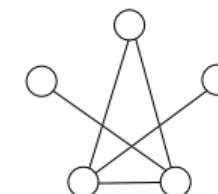
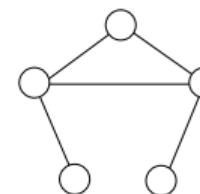
A redução **CLI**  $\propto_{\text{poly}}$  **CV** é válida. É possível mostrar que  $G, k$  é uma instância SIM de **CLI** se e somente se  $\overline{G}, n - k$  é uma instância SIM de **CV**.

## Proposição

O grafo  $G$  tem uma clique de tamanho  $k$  se e somente se o grafo complementar  $\overline{G}$  tem uma cobertura de tamanho  $n - k$ .

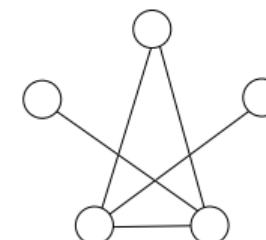
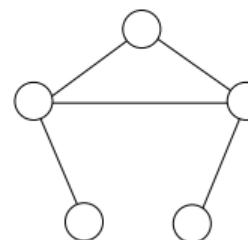
Vamos mostrar que:

$S$  é uma clique de tamanho  $k$  em  $G \iff V - S$  é uma CV de tamanho  $n - k$  em  $\overline{G}$ .



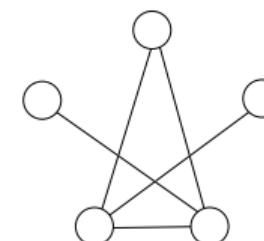
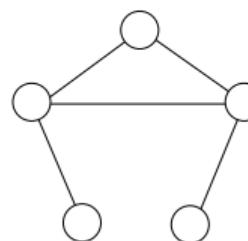
Provas de  $\mathcal{NP}$ -completude: CV

- $G$  tem uma clique de tamanho  $k$ 
  - ⇒ Existe um conjunto  $S$  de  $k$  vértices que é uma clique em  $G$
  - ⇒ Todas as arestas entre vértices de  $S$  estão em  $G$
  - ⇒ Nenhuma das arestas com ambos os extremos em  $S$ , estão em  $\overline{G}$
  - ⇒ As arestas em  $\overline{G}$  possuem pelo menos um extremo em  $V - S$
  - ⇒  $V - S$  é uma cobertura de vértices de tamanho  $n - k$  em  $\overline{G}$ .

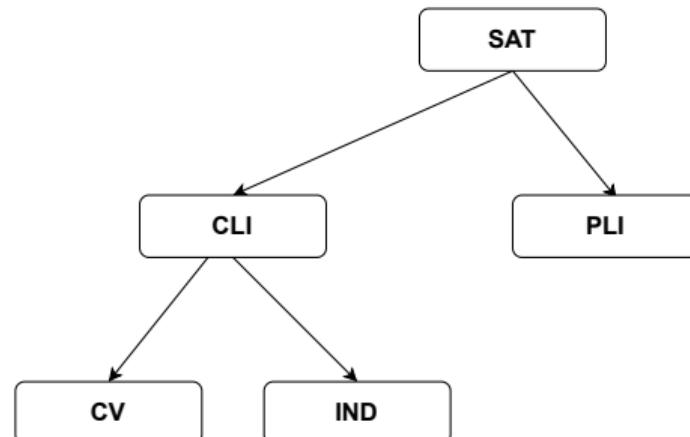


Provas de  $\mathcal{NP}$ -completude: CV

- Existe uma cobertura de vértices  $S'$  de tamanho  $n - k$  em  $\overline{G}$ 
  - ⇒ Todas as arestas em  $\overline{G}$  possuem pelo menos um extremo em  $S'$
  - ⇒ Não há nenhuma aresta em  $\overline{G}$  com ambos os extremos em  $V - S'$
  - ⇒ Todas as arestas entre vértices de  $V - S'$  estão em  $G$
  - ⇒ Existe um conjunto  $V - S'$  de  $n - (n - k)$  vértices que é uma clique em  $G$
  - ⇒  $G$  tem uma clique de tamanho  $k$ .



# Provas de $\mathcal{NP}$ -completude: nosso estado da arte



# Resumo

1 Revisão do conteúdo e objetivo

2 Provas de  $\mathcal{NP}$ -completude

3 Classe co- $\mathcal{NP}$

4 Síntese

# A classe co- $\mathcal{NP}$

Veja alguns **exemplos de problemas “diferentes”**:

## Problema UnSAT

Dado uma fórmula lógica  $\mathcal{F}$  com  $n$  variáveis na forma normal conjuntiva, decidir se **não** existe uma atribuição das variáveis  $x_1, \dots, x_n$  para a qual  $\mathcal{F} = 1$ .

# A classe co- $\mathcal{NP}$

Veja alguns **exemplos de problemas “diferentes”**:

## Problema UnSAT

Dado uma fórmula lógica  $\mathcal{F}$  com  $n$  variáveis na forma normal conjuntiva, decidir se **não** existe uma atribuição das variáveis  $x_1, \dots, x_n$  para a qual  $\mathcal{F} = 1$ .

## Problema No-CLI

Dado um grafo não-direcionado  $G = (V, E)$ , e um valor inteiro  $k$ , decidir se  $G$  **não** contém uma clique com  $k$  vértices.

Será que **UnSAT** e **No-CLI** estão em  $\mathcal{NP}$ ?...

# A classe co- $\mathcal{NP}$

Veja alguns **exemplos de problemas “diferentes”**:

## Problema UnSAT

Dado uma fórmula lógica  $\mathcal{F}$  com  $n$  variáveis na forma normal conjuntiva, decidir se **não** existe uma atribuição das variáveis  $x_1, \dots, x_n$  para a qual  $\mathcal{F} = 1$ .

## Problema No-CLI

Dado um grafo não-direcionado  $G = (V, E)$ , e um valor inteiro  $k$ , decidir se  $G$  **não** contém uma clique com  $k$  vértices.

Será que **UnSAT** e **No-CLI** estão em  $\mathcal{NP}$ ?...

Será que existe um **certificado conciso** (polinomial) para esses problemas?...

# A classe co- $\mathcal{NP}$

Os problemas **UnSAT** e **No-CLI** apresentados possuem a característica a seguir:

- Uma instância SIM do **UnSAT (No-CLI)** é instância NÃO do **SAT (CLI)**;
- Uma instância NÃO do **UnSAT (No-CLI)** é instância SIM do **SAT (CLI)**.

## Complemento de um problema

O complemento de um problema de decisão  $A$  é o problema  $\bar{A}$  cujas instâncias SIM são as instâncias NÃO de  $A$  e vice-versa.

# A classe co- $\mathcal{NP}$

## co- $\mathcal{NP}$

Um problema  $A$  é da classe co- $\mathcal{NP}$  se seu complemento  $\bar{A}$  é um problema  $\mathcal{NP}$ .

**Definição equivalente:** co- $\mathcal{NP}$  é a classe de problemas de decisão que possuem um verificador polinomial para instâncias NÃO.

# A classe co- $\mathcal{NP}$

## Outro exemplo:

**AGM:** Dado um grafo  $G = (V, E)$ , com pesos  $w_e$  para cada  $e \in E$ , existe uma árvore geradora em  $G$  com peso  $\leq W$ ?

**AGM:** Dado um grafo  $G = (V, E)$ , com pesos  $w_e$  para cada  $e \in E$ , toda árvore geradora em  $G$  tem peso  $> W$ ?

# A classe co- $\mathcal{NP}$

## Outro exemplo:

**AGM:** Dado um grafo  $G = (V, E)$ , com pesos  $w_e$  para cada  $e \in E$ , existe uma árvore geradora em  $G$  com peso  $\leq W$ ?

**AGM:** Dado um grafo  $G = (V, E)$ , com pesos  $w_e$  para cada  $e \in E$ , toda árvore geradora em  $G$  tem peso  $> W$ ?

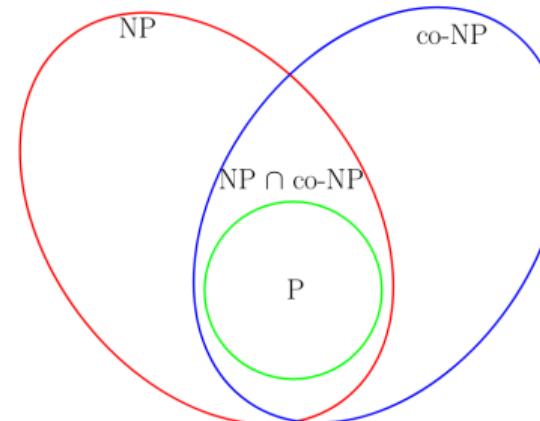
## Teorema

Se  $A \in \mathcal{P}$ , então  $\bar{A} \in \mathcal{P}$ .

# A classe co- $\mathcal{NP}$

Outra questão fundamental:  $\text{co-}\mathcal{NP} = \mathcal{NP}?$

- A maioria dos cientistas acredita que  $\text{co-}\mathcal{NP} \neq \mathcal{NP}$ .
- Se  $X \in \mathcal{NP}$ -completo e  $\bar{X} \in \mathcal{NP}$  então  $\text{co-}\mathcal{NP} = \mathcal{NP}$ .  
De novo, os problemas  $\mathcal{NP}$ -completos parecem ser a chave da questão.
- Não há consenso se  $P = (\mathcal{NP} \cap \text{co-}\mathcal{NP})$ .



## Resumo

- 1 Revisão do conteúdo e objetivo
  - 2 Provas de  $\mathcal{NP}$ -completude
  - 3 Classe co- $\mathcal{NP}$
  - 4 Síntese

Síntese

- É possível mostrar a **NP-completude** de diversos problemas que achávamos complexos.

# Material bibliográfico e exercícios

U. Manber. Introduction to Algorithms. – **Cap. 11.**

**Exercícios:** ver exercícios no final do Capítulo 11.

Bibliografia complementar:

*Combinatorial Optimization: Algorithms and Complexity*

C.H. Papadimitriou e K. Steiglitz, Dover, 1982.

*The Design and Analysis of Computer Algorithms*

A.V. Aho, J.E. Hopcroft e J.D. Ullman, Addison-Wesley, 1974.

## Dúvidas

## Dúvidas?