Disciplina CCM-104 – Teoria da Computação

Teoria da complexidade

Profa. Carla Negri Lintzmayer

carla.negri@ufabc.edu.br

www.professor.ufabc.edu.br/∼carla.negri

Centro de Matemática, Computação e Cognição – Universidade Federal do ABC

Aviso legal

Estes slides não contêm um conteúdo completo sobre teoria da complexidade: são apenas um apoio gráfico a uma aula específica dada em sala.

Redução

Redução informalmente

- Reduzir de uma linguagem A para uma linguagem B significa usar uma máquina que decide/reconhece B para decidir/reconhecer A.

$$\omega_B \longrightarrow \boxed{MT_B} \longrightarrow \text{aceita/rejeita}$$

$$\omega_A \longrightarrow \boxed{\xrightarrow{f} \omega_B \longrightarrow \boxed{MT_B} \longrightarrow \text{aceita/rejeita}} \longrightarrow \text{aceita/rejeita}$$

Redução formalmente

Redução polinomial

A linguagem A é *redutível em tempo polinomial* à linguagem B se existe uma função computável em tempo polinomial $f\colon \Sigma^* \to \Sigma^*$ tal que para toda $\omega \in \Sigma^*$

$$\omega \in A \text{ se e somente se } f(\omega) \in B\,.$$

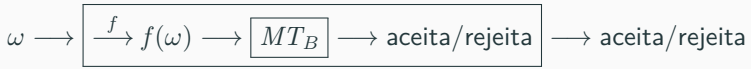
Dizemos também que A é redutível eficientemente para B.

Mas qual a importância da redução?

Suponha que a linguagem A reduz em tempo polinomial para a linguagem B:

Mas qual a importância da redução?

Suponha que a linguagem A reduz em tempo polinomial para a linguagem B:

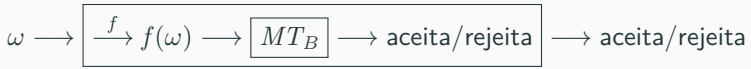


Que conclusão podemos tirar se:

1. existir MT eficiente para A (isto é, $A \in$ P)?
2. existir MT eficiente para B (isto é, $B \in$ P)?
3. não existir algoritmo eficiente para A?
4. não existir algoritmo eficiente para B?

Mas qual a importância da redução?

Suponha que a linguagem A reduz em tempo polinomial para a linguagem B:



Que conclusão podemos tirar se:

1. existir MT eficiente para A (isto é, $A \in \mathsf{P}$)?
2. existir MT eficiente para B (isto é, $B \in \mathsf{P}$)?
3. não existir algoritmo eficiente para A?
4. não existir algoritmo eficiente para B?

Ou seja, B é tão difícil quanto A.
Ou então, A não é mais difícil do que B.

Classes de complexidade

Classes de complexidade

- Nem todos os problemas/linguagens do mundo estão em P.

Classes de complexidade

- Nem todos os problemas/linguagens do mundo estão em P.
- Ao menos é o que se sabe até agora...

Classes de complexidade

- Nem todos os problemas/linguagens do mundo estão em P.
- Ao menos é o que se sabe até agora...
- Até agora, não encontramos algoritmos de tempo polinomial que resolvem CAMINHOHAM, CLIQUE, SUBSETSUM...

Classes de complexidade

- Nem todos os problemas/linguagens do mundo estão em P.
- Ao menos é o que se sabe até agora...
- Até agora, não encontramos algoritmos de tempo polinomial que resolvem CAMINHOHAM, CLIQUE, SUBSETSUM...
- E estes estão em NP.

Classes de complexidade

- Nem todos os problemas/linguagens do mundo estão em P.
- Ao menos é o que se sabe até agora...
- Até agora, não encontramos algoritmos de tempo polinomial que resolvem CAMINHOHAM, CLIQUE, SUBSETSUM...
- E estes estão em NP.
- Estar em NP é sinal de dificuldade de resolver um problema?

Classe NP-completos

Linguagens NP-completas

NP-completo

Uma linguagem Q é NP-completa se

- $Q \in$ NP e
- toda linguagem de NP se reduz polinomialmente a Q.

O que isso significa?

Linguagens NP-completas

NP-completo

Uma linguagem Q é NP-completa se

- $Q \in$ NP e
- toda linguagem de NP se reduz polinomialmente a Q.

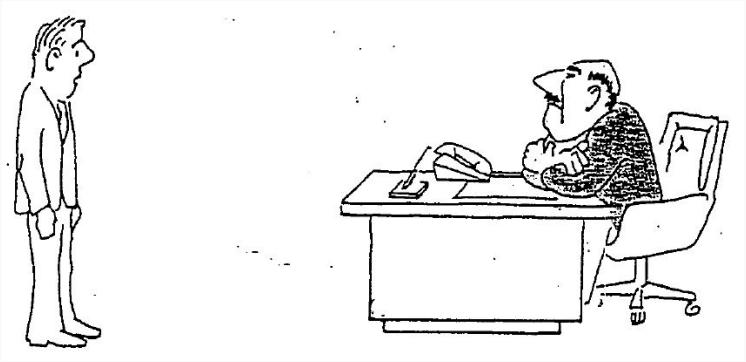
O que isso significa?

Teorema

Se A é NP-completa e $A \in P$, então $P = NP$.

Resolva esse problema

Ele está em NP!



"I can't find an efficient Turing machine, I guess I'm just too dumb."

Ele é NP-completo!

"I can't find an efficient Turing machine, but neither can all these famous people."

Linguagens NP-completas

Como mostrar que uma linguagem é NP-completa?

Linguagens NP-completas

Como mostrar que uma linguagem é NP-completa? Sigamos a definição:

- Primeiro mostramos que a linguagem está em NP.
- Depois enumeramos todas as linguagens de NP e fazemos uma redução polinomial de cada uma delas para a nossa linguagem.

Linguagens NP-completas

Como mostrar que uma linguagem é NP-completa? Sigamos a definição:

- Primeiro mostramos que a linguagem está em NP.
- Depois enumeramos todas as linguagens de NP e fazemos uma redução polinomial de cada uma delas para a nossa linguagem.

Impossível!!

Linguagens NP-completas

Como mostrar que uma linguagem é NP-completa? Sigamos a definição:

- Primeiro mostramos que a linguagem está em NP.
- Depois enumeramos todas as linguagens de NP e fazemos uma redução polinomial de cada uma delas para a nossa linguagem.

Impossível!!

Lembre-se que a operação de redução pode ser composta:

Linguagens NP-completas

Como mostrar que uma linguagem é NP-completa? Sigamos a definição:

- Primeiro mostramos que a linguagem está em NP.
- Depois enumeramos todas as linguagens de NP e fazemos uma redução polinomial de cada uma delas para a nossa linguagem.

Impossível!!

Lembre-se que a operação de redução pode ser composta:

- Primeiro mostramos que a linguagem está em NP.
- Depois encontramos uma linguagem que já é NP-completa e a reduzimos polinomialmente para a nossa.

Linguagens NP-completas

Mas e a primeira linguagem NP-completa?

Linguagens NP-completas

Mas e a primeira linguagem NP-completa?

A linguagem SAT

SAT $= \{\langle \phi \rangle : \phi$ é uma fórmula Booleana satisfatível$\}$

Linguagens NP-completas

Mas e a primeira linguagem NP-completa?

A linguagem SAT

SAT $= \{\langle\phi\rangle : \phi$ é uma fórmula Booleana satisfatível$\}$

Exemplo: $\phi = (\overline{x_1} \wedge x_2) \vee (x_1 \wedge \overline{x_3})$

Linguagens NP-completas

Mas e a primeira linguagem NP-completa?

A linguagem SAT

SAT $= \{\langle\phi\rangle : \phi$ é uma fórmula Booleana satisfatível$\}$

Exemplo: $\phi = (\overline{x_1} \wedge x_2) \vee (x_1 \wedge \overline{x_3})$

Teorema (Cook-Levin)

SAT é NP-completa.

Mas e a primeira linguagem NP-completa?

A linguagem SAT

SAT $= \{\langle\phi\rangle : \phi$ é uma fórmula Booleana satisfatível$\}$

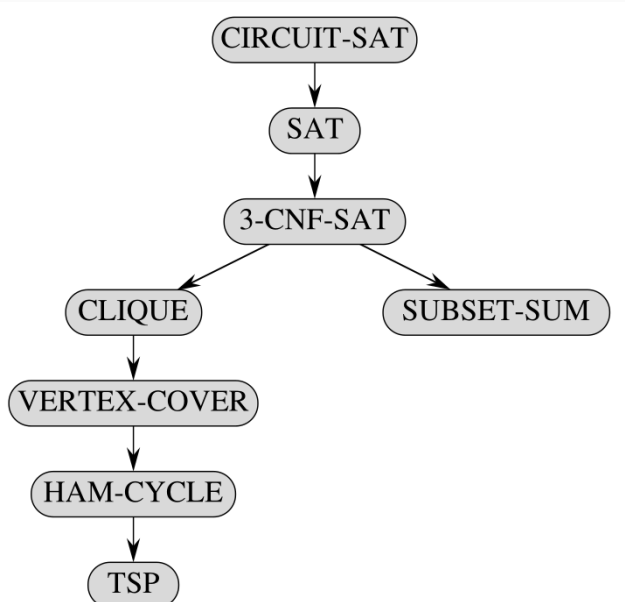
Exemplo: $\phi = (\overline{x_1} \wedge x_2) \vee (x_1 \wedge \overline{x_3})$

Teorema (Cook-Levin)

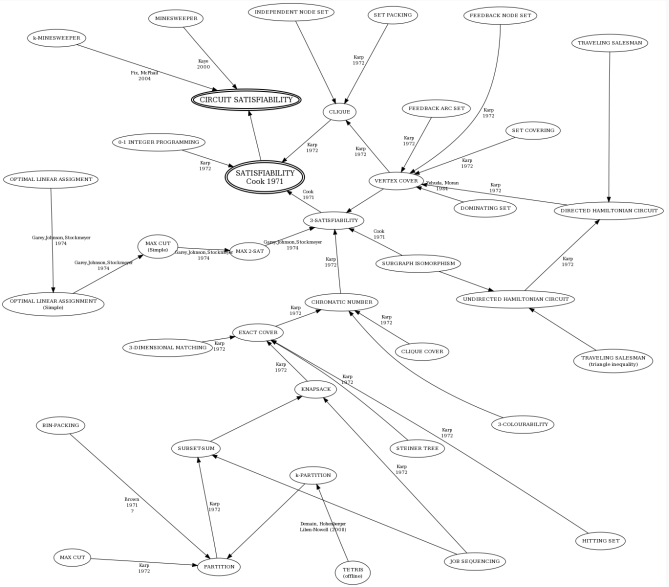
SAT é NP-completa.

Claramente, SAT $\in$ NP: um certificado é uma atribuição de valores `true` ou `false` para as variáveis $x_1, \ldots, x_n$, e pode-se verificar em tempo polinomial se ϕ é verdadeira ou não com essa atribuição.

Linguagens NP-completas



Linguagens NP-completas



O problema 3-SAT

Outra linguagem NP-completa: 3SAT

3SAT $= \{\langle\phi\rangle : \phi$ é uma 3cnf-fórmula Booleana satisfatível$\}$

Exemplo: $\phi = (x_1 \vee \overline{x_2} \vee x_3) \wedge (\overline{x_1} \vee x_2 \vee x_4) \wedge (\overline{x_1} \vee x_2 \vee \overline{x_3}) \wedge (\overline{x_1} \vee \overline{x_2} \vee \overline{x_3})$

Teorema

3SAT é NP-completa.

Demonstração: Primeiramente, 3SAT $\in$ NP: o certificado é uma atribuição de valores `true` ou `false` para as variáveis e pode-se verificar em tempo polinomial (na quantidade de cláusulas) se ϕ é verdadeira ou não com essa atribuição.

Outra linguagem NP-completa: 3SAT

3SAT $= \{\langle\phi\rangle : \phi$ é uma 3cnf-fórmula Booleana satisfatível$\}$

Exemplo: $\phi = (x_1 \vee \overline{x_2} \vee x_3) \wedge (\overline{x_1} \vee x_2 \vee x_4) \wedge (\overline{x_1} \vee x_2 \vee \overline{x_3}) \wedge (\overline{x_1} \vee \overline{x_2} \vee \overline{x_3})$

Teorema

3SAT é NP-completa.

Demonstração: Primeiramente, 3SAT $\in$ NP: o certificado é uma atribuição de valores `true` ou `false` para as variáveis e pode-se verificar em tempo polinomial (na quantidade de cláusulas) se ϕ é verdadeira ou não com essa atribuição.

Agora, vamos mostrar que SAT reduz polinomialmente para 3SAT *ou então* adaptamos a prova do Teorema de Cook-Levin. (Essa demonstração está, portanto, incompleta). C.Q.D.

O problema da Clique

Outra linguagem NP-completa: CLIQUE

CLIQUE $= \{\langle G, k\rangle : G$ é um grafo que tem uma clique de tamanho $\geq k\}$.

Outra linguagem NP-completa: CLIQUE

CLIQUE $= \{\langle G, k\rangle : G$ é um grafo que tem uma clique de tamanho $\geq k\}$.

Teorema

CLIQUE é NP-completa.

Demonstração: Primeiro mostramos que CLIQUE $\in$ NP e depois faremos uma redução polinomial de 3SAT para CLIQUE.

Outra linguagem NP-completa: CLIQUE

CLIQUE $= \{\langle G, k\rangle : G$ é um grafo que tem uma clique de tamanho $\geq k\}$.

Teorema

CLIQUE é NP-completa.

Demonstração: Primeiro mostramos que CLIQUE $\in$ NP e depois faremos uma redução polinomial de 3SAT para CLIQUE.

CLIQUE está em NP pois, dado um conjunto S qualquer de vértices, podemos verificar se $|S| \geq k$ e se cada vértice de S, que são $O(|V(G)|)$, é vizinho de todos os outros vértices em S, o que pode ser feito olhando sua vizinhança e portanto leva tempo $O(|V(G)||E(G)|)$.

Outra linguagem NP-completa: CLIQUE

Teorema

CLIQUE é NP-completa.

Demonstração (continuação): Agora considere uma fórmula 3CNF ϕ com m cláusulas sobre os literais das variáveis $v_1, \ldots, v_n$.

Outra linguagem NP-completa: CLIQUE

Teorema

CLIQUE é NP-completa.

Demonstração (continuação): Agora considere uma fórmula 3CNF ϕ com m cláusulas sobre os literais das variáveis $v_1, \ldots, v_n$.
Vamos construir um grafo G com $3m$ vértices: para cada cláusula, temos 3 vértices representando seus literais.

Outra linguagem NP-completa: CLIQUE

Teorema

CLIQUE é NP-completa.

Demonstração (continuação): Agora considere uma fórmula 3CNF ϕ com m cláusulas sobre os literais das variáveis $v_1, \ldots, v_n$.
Vamos construir um grafo G com $3m$ vértices: para cada cláusula, temos 3 vértices representando seus literais.
Haverá aresta entre dois vértices x e y se e somente se os literais representados por eles estiverem em cláusulas diferentes, x corresponde a um literal v e y não corresponde ao literal $\bar{v}$.

Outra linguagem NP-completa: CLIQUE

Teorema

CLIQUE é NP-completa.

Demonstração (continuação): Agora considere uma fórmula 3CNF ϕ com m cláusulas sobre os literais das variáveis $v_1, \ldots, v_n$.
Vamos construir um grafo G com $3m$ vértices: para cada cláusula, temos 3 vértices representando seus literais.
Haverá aresta entre dois vértices x e y se e somente se os literais representados por eles estiverem em cláusulas diferentes, x corresponde a um literal v e y não corresponde ao literal $\bar{v}$.
Por fim, tome $k = m$.

Outra linguagem NP-completa: CLIQUE

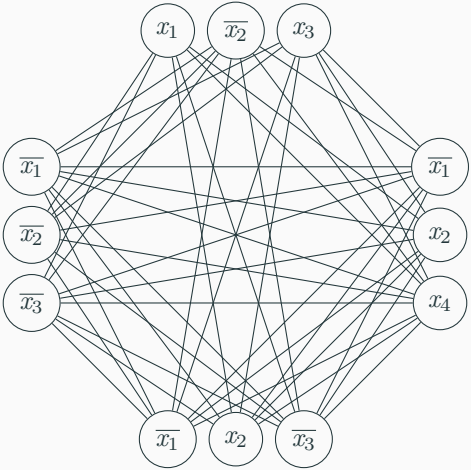
Teorema

CLIQUE é NP-completa.

Demonstração (continuação): Agora considere uma fórmula 3CNF ϕ com m cláusulas sobre os literais das variáveis $v_1, \ldots, v_n$.
Vamos construir um grafo G com $3m$ vértices: para cada cláusula, temos 3 vértices representando seus literais.
Haverá aresta entre dois vértices x e y se e somente se os literais representados por eles estiverem em cláusulas diferentes, x corresponde a um literal v e y não corresponde ao literal $\bar{v}$.
Por fim, tome $k = m$.
Temos então uma instância $\langle G, k \rangle$ para CLIQUE gerada em tempo polinomial, pois o grafo tem $3m$ vértices e no máximo $3m(3m-1)/2$ arestas. Resta verificar se $\langle \phi \rangle \in$ 3SAT se e somente se $\langle G, k \rangle \in$ CLIQUE.

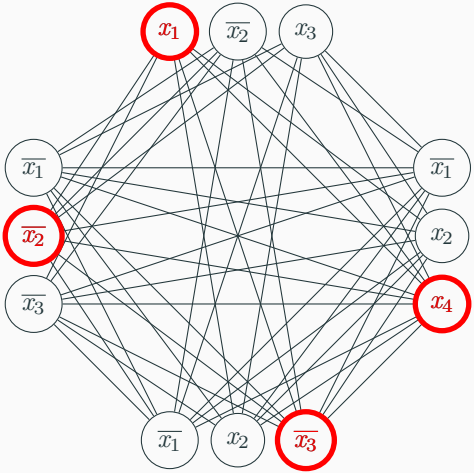
Exemplo

Dado $\phi = (x_1 \vee \overline{x_2} \vee x_3) \wedge (\overline{x_1} \vee x_2 \vee x_4) \wedge (\overline{x_1} \vee x_2 \vee \overline{x_3}) \wedge (\overline{x_1} \vee \overline{x_2} \vee \overline{x_3})$, construa:



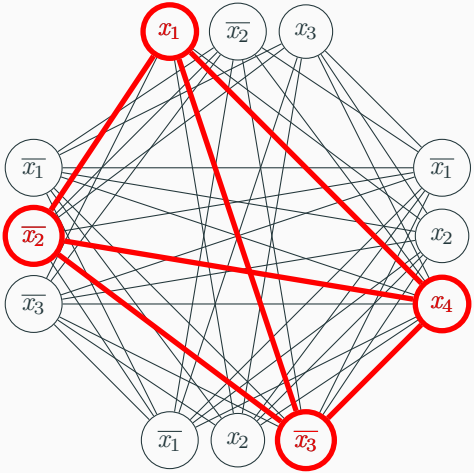
Exemplo

Dado $\phi = (x_1 \vee \overline{x_2} \vee x_3) \wedge (\overline{x_1} \vee x_2 \vee x_4) \wedge (\overline{x_1} \vee x_2 \vee \overline{x_3}) \wedge (\overline{x_1} \vee \overline{x_2} \vee \overline{x_3})$, construa:



Exemplo

Dado $\phi = (x_1 \vee \overline{x_2} \vee x_3) \wedge (\overline{x_1} \vee x_2 \vee x_4) \wedge (\overline{x_1} \vee x_2 \vee \overline{x_3}) \wedge (\overline{x_1} \vee \overline{x_2} \vee \overline{x_3})$, construa:



Outra linguagem NP-completa: CLIQUE

Teorema

CLIQUE é NP-completa.

Demonstração (continuação): Suponha que $\langle \phi \rangle \in$ 3SAT. Então ϕ é satisfatível.

Outra linguagem NP-completa: CLIQUE

Teorema

CLIQUE é NP-completa.

Demonstração (continuação): Suponha que $\langle \phi \rangle \in$ 3SAT. Então ϕ é satisfatível. Então em cada cláusula, pelo menos um literal tem valor `true`. Tome os vértices correspondentes a esses literais em um conjunto S.

Outra linguagem NP-completa: CLIQUE

Teorema

CLIQUE é NP-completa.

Demonstração (continuação): Suponha que $\langle \phi \rangle \in$ 3SAT. Então ϕ é satisfatível.
Então em cada cláusula, pelo menos um literal tem valor `true`. Tome os vértices correspondentes a esses literais em um conjunto S.
Como um literal e sua negação não podem ter valor `true`, existe uma aresta entre quaisquer dois vértices que representam esses literais em G.

Outra linguagem NP-completa: CLIQUE

Teorema

CLIQUE é NP-completa.

Demonstração (continuação): Suponha que $\langle \phi \rangle \in$ 3SAT. Então ϕ é satisfatível.
Então em cada cláusula, pelo menos um literal tem valor `true`. Tome os vértices correspondentes a esses literais em um conjunto S.
Como um literal e sua negação não podem ter valor `true`, existe uma aresta entre quaisquer dois vértices que representam esses literais em G.
Então S é uma clique em G com pelo menos m vértices, já que pegamos pelo menos um por cláusula, e note que $m = k$.

Outra linguagem NP-completa: CLIQUE

Teorema

CLIQUE é NP-completa.

Demonstração (continuação): Suponha que $\langle \phi \rangle \in$ 3SAT. Então ϕ é satisfatível.
Então em cada cláusula, pelo menos um literal tem valor `true`. Tome os vértices correspondentes a esses literais em um conjunto S.
Como um literal e sua negação não podem ter valor `true`, existe uma aresta entre quaisquer dois vértices que representam esses literais em G.
Então S é uma clique em G com pelo menos m vértices, já que pegamos pelo menos um por cláusula, e note que $m = k$. Logo, $\langle G, k \rangle \in$ CLIQUE.

Outra linguagem NP-completa: CLIQUE

Teorema

CLIQUE é NP-completa.

Demonstração (continuação): Agora suponha que $\langle G, k \rangle \in$ CLIQUE. Então existe conjunto $S \subseteq V(G)$ que é clique e $|S| \geq k$.

Outra linguagem NP-completa: CLIQUE

Teorema

CLIQUE é NP-completa.

Demonstração (continuação): Agora suponha que $\langle G, k \rangle \in$ CLIQUE. Então existe conjunto $S \subseteq V(G)$ que é clique e $|S| \geq k$.

Então existe uma aresta entre quaisquer dois vértices de S.

Outra linguagem NP-completa: CLIQUE

Teorema

CLIQUE é NP-completa.

Demonstração (continuação): Agora suponha que $\langle G, k \rangle \in$ CLIQUE. Então existe conjunto $S \subseteq V(G)$ que é clique e $|S| \geq k$.

Então existe uma aresta entre quaisquer dois vértices de S.

Então esses dois não representam literais que são a negação um do outro e eles estão em cláusulas diferentes.

Outra linguagem NP-completa: CLIQUE

Teorema

CLIQUE é NP-completa.

Demonstração (continuação): Agora suponha que $\langle G, k \rangle \in$ CLIQUE. Então existe conjunto $S \subseteq V(G)$ que é clique e $|S| \geq k$.

Então existe uma aresta entre quaisquer dois vértices de S.

Então esses dois não representam literais que são a negação um do outro e eles estão em cláusulas diferentes.

Então dar valor `true` aos literais representados pelos vértices de S satisfaz ϕ.

Outra linguagem NP-completa: CLIQUE

Teorema

CLIQUE é NP-completa.

Demonstração (continuação): Agora suponha que $\langle G, k \rangle \in$ CLIQUE. Então existe conjunto $S \subseteq V(G)$ que é clique e $|S| \geq k$.

Então existe uma aresta entre quaisquer dois vértices de S.

Então esses dois não representam literais que são a negação um do outro e eles estão em cláusulas diferentes.

Então dar valor `true` aos literais representados pelos vértices de S satisfaz ϕ. Logo, $\langle \phi \rangle \in$ 3SAT. C.Q.D.

O problema da cobertura por vértices

Outra linguagem NP-completa: VERTEXCOVER

Uma *cobertura por vértices* é um conjunto $S \subseteq V(G)$ tal que toda aresta $uv \in E(G)$ tem $u \in S$ ou $v \in S$.

VERTEXCOVER
$= \{\langle G, k\rangle : G$ é um grafo que tem uma cobertura por vértices de tamanho $\leq k\}$

Outra linguagem NP-completa: VERTEXCOVER

Uma *cobertura por vértices* é um conjunto $S \subseteq V(G)$ tal que toda aresta $uv \in E(G)$ tem $u \in S$ ou $v \in S$.

VERTEXCOVER
$= \{\langle G, k\rangle : G$ é um grafo que tem uma cobertura por vértices de tamanho $\leq k\}$

Teorema

VERTEXCOVER é NP-completa.

Demonstração: Primeiro mostramos que VERTEXCOVER está em NP e depois faremos uma redução polinomial de 3SAT para VERTEXCOVER.

Outra linguagem NP-completa: VERTEXCOVER

Uma *cobertura por vértices* é um conjunto $S \subseteq V(G)$ tal que toda aresta $uv \in E(G)$ tem $u \in S$ ou $v \in S$.

VERTEXCOVER
$= \{\langle G, k\rangle : G$ é um grafo que tem uma cobertura por vértices de tamanho $\leq k\}$

Teorema

VERTEXCOVER é NP-completa.

Demonstração: Primeiro mostramos que VERTEXCOVER está em NP e depois faremos uma redução polinomial de 3SAT para VERTEXCOVER.

VERTEXCOVER está em NP pois, dado um conjunto S de vértices, pode-se verificar se $|S| \leq k$ e se todas as arestas de G têm ao menos um extremo em S em tempo polinomial, bastando percorrer S, cujo tamanho é $O(|V(G)|)$, e cada aresta de G.

Outra linguagem NP-completa: VERTEXCOVER

Teorema

VERTEXCOVER é NP-completa.

Demonstração (continuação): Agora considere uma fórmula 3CNF ϕ com m cláusulas sobre os literais das variáveis $v_1, \ldots, v_n$.

Outra linguagem NP-completa: VERTEXCOVER

Teorema

VERTEXCOVER é NP-completa.

Demonstração (continuação): Agora considere uma fórmula 3CNF ϕ com m cláusulas sobre os literais das variáveis $v_1, \ldots, v_n$.

Vamos construir um grafo G da seguinte forma: para cada variável v_i, crie dois nós com rótulos v_i e $\bar{v}_i$ e adicione uma aresta entre eles – chame-os de *gadgets das variáveis*; para cada cláusula $a \wedge b \wedge c$, crie um triângulo cujos rótulos são a, b e c – chame-os de *gadgets das cláusulas*; conecte cada vértice de um gadget de cláusula ao seu vértice de mesmo rótulo no gadget de variável correspondente.

Outra linguagem NP-completa: VERTEXCOVER

Teorema

VERTEXCOVER é NP-completa.

Demonstração (continuação): Agora considere uma fórmula 3CNF ϕ com m cláusulas sobre os literais das variáveis $v_1, \ldots, v_n$.
Vamos construir um grafo G da seguinte forma: para cada variável v_i, crie dois nós com rótulos v_i e $\bar{v}_i$ e adicione uma aresta entre eles – chame-os de *gadgets das variáveis*; para cada cláusula $a \wedge b \wedge c$, crie um triângulo cujos rótulos são a, b e c – chame-os de *gadgets das cláusulas*; conecte cada vértice de um gadget de cláusula ao seu vértice de mesmo rótulo no gadget de variável correspondente.
Por fim, tome $k = n + 2m$.

Outra linguagem NP-completa: VERTEXCOVER

Teorema

VERTEXCOVER é NP-completa.

Demonstração (continuação): Agora considere uma fórmula 3CNF ϕ com m cláusulas sobre os literais das variáveis $v_1, \ldots, v_n$.

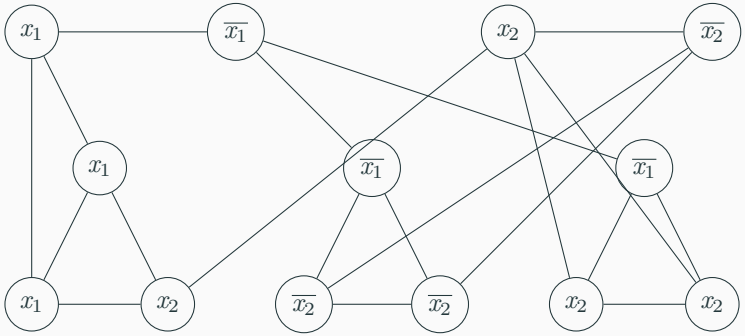
Vamos construir um grafo G da seguinte forma: para cada variável v_i, crie dois nós com rótulos v_i e $\bar{v}_i$ e adicione uma aresta entre eles – chame-os de *gadgets das variáveis*; para cada cláusula $a \wedge b \wedge c$, crie um triângulo cujos rótulos são a, b e c – chame-os de *gadgets das cláusulas*; conecte cada vértice de um gadget de cláusula ao seu vértice de mesmo rótulo no gadget de variável correspondente.

Por fim, tome $k = n + 2m$.

Temos então uma instância $\langle G, k \rangle$ para VERTEXCOVER gerada em tempo polinomial, pois o grafo tem $2n + 3m$ vértices. Resta verificar se $\langle \phi \rangle \in$ 3SAT se e somente se $\langle G, k \rangle \in$ VERTEXCOVER.

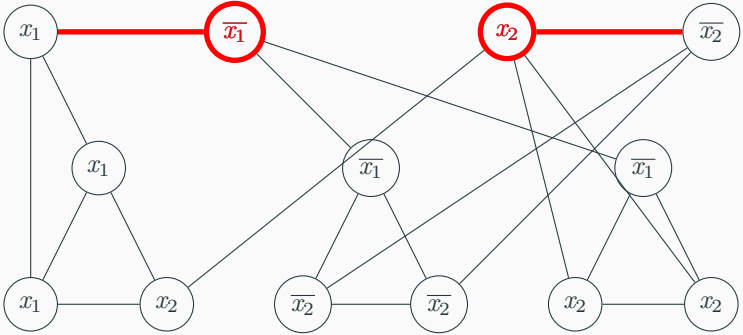
Exemplo

Dado $\phi = (x_1 \vee x_1 \vee x_2) \wedge (\overline{x_1} \vee \overline{x_2} \vee \overline{x_2}) \wedge (\overline{x_1} \vee x_2 \vee x_2)$, construa:



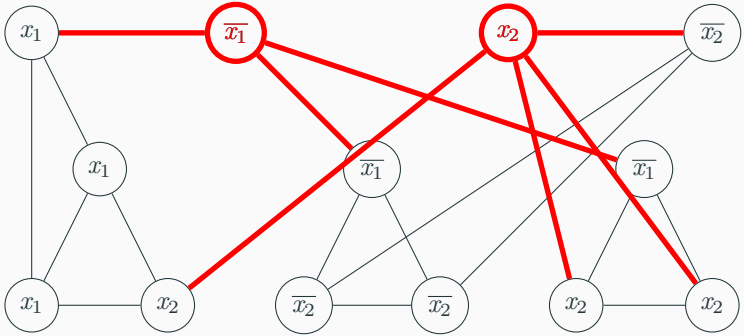
Exemplo

Dado $\phi = (x_1 \vee x_1 \vee x_2) \wedge (\overline{x_1} \vee \overline{x_2} \vee \overline{x_2}) \wedge (\overline{x_1} \vee x_2 \vee x_2)$, construa:



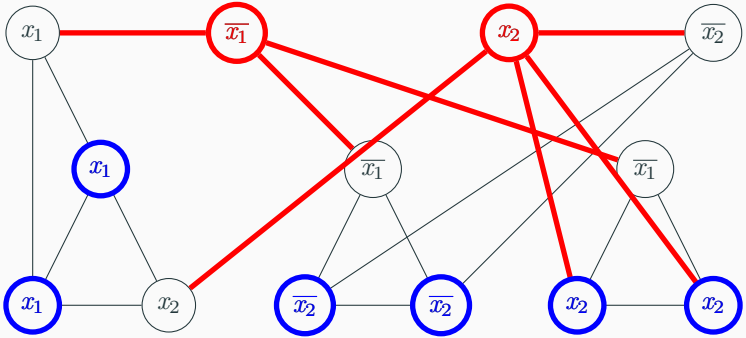
Exemplo

Dado $\phi = (x_1 \vee x_1 \vee x_2) \wedge (\overline{x_1} \vee \overline{x_2} \vee \overline{x_2}) \wedge (\overline{x_1} \vee x_2 \vee x_2)$, construa:



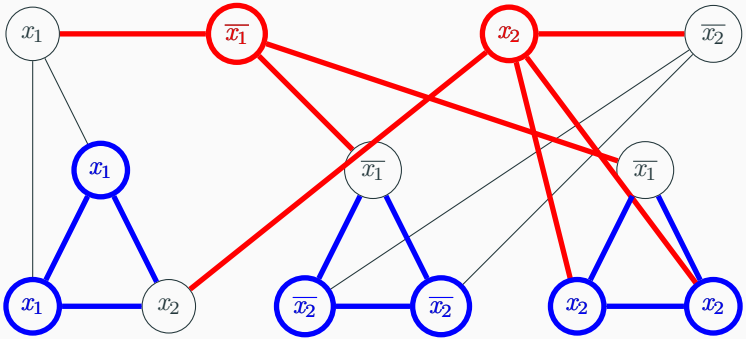
Exemplo

Dado $\phi = (x_1 \vee x_1 \vee x_2) \wedge (\overline{x_1} \vee \overline{x_2} \vee \overline{x_2}) \wedge (\overline{x_1} \vee x_2 \vee x_2)$, construa:



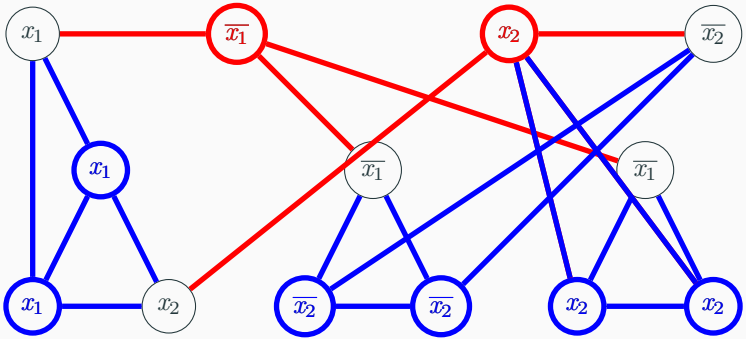
Exemplo

Dado $\phi = (x_1 \vee x_1 \vee x_2) \wedge (\overline{x_1} \vee \overline{x_2} \vee \overline{x_2}) \wedge (\overline{x_1} \vee x_2 \vee x_2)$, construa:



Exemplo

Dado $\phi = (x_1 \vee x_1 \vee x_2) \wedge (\overline{x_1} \vee \overline{x_2} \vee \overline{x_2}) \wedge (\overline{x_1} \vee x_2 \vee x_2)$, construa:



Outra linguagem NP-completa: VERTEXCOVER

Teorema

VERTEXCOVER é NP-completa.

Demonstração (continuação): Suponha que $\langle \phi \rangle \in$ 3SAT. Então ϕ é satisfatível.

Outra linguagem NP-completa: VERTEXCOVER

Teorema

VERTEXCOVER é NP-completa.

Demonstração (continuação): Suponha que $\langle \phi \rangle \in$ 3SAT. Então ϕ é satisfatível.
Em cada gadget de variável, escolha o vértice correspondente ao literal que tem valor `true` e coloque em um conjunto S (note que não será possível escolher ambos os vértices do gadget).

Outra linguagem NP-completa: VERTEXCOVER

Teorema

VERTEXCOVER é NP-completa.

Demonstração (continuação): Suponha que $\langle \phi \rangle \in$ 3SAT. Então ϕ é satisfatível.
Em cada gadget de variável, escolha o vértice correspondente ao literal que tem valor `true` e coloque em um conjunto S (note que não será possível escolher ambos os vértices do gadget). Isso cobre a aresta do gadget de variável e (ao menos) uma aresta que vai para um gadget de cláusula.

Outra linguagem NP-completa: VERTEXCOVER

Teorema

VERTEXCOVER é NP-completa.

Demonstração (continuação): Suponha que $\langle \phi \rangle \in$ 3SAT. Então ϕ é satisfatível.
Em cada gadget de variável, escolha o vértice correspondente ao literal que tem valor `true` e coloque em um conjunto S (note que não será possível escolher ambos os vértices do gadget). Isso cobre a aresta do gadget de variável e (ao menos) uma aresta que vai para um gadget de cláusula.
Se um literal escolhido em um gadget de variável aparece em uma cláusula, coloque os outros dois vértices do gadget dessa cláusula em S.

Outra linguagem NP-completa: VERTEXCOVER

Teorema

VERTEXCOVER é NP-completa.

Demonstração (continuação): Suponha que $\langle \phi \rangle \in$ 3SAT. Então ϕ é satisfatível.
Em cada gadget de variável, escolha o vértice correspondente ao literal que tem valor `true` e coloque em um conjunto S (note que não será possível escolher ambos os vértices do gadget). Isso cobre a aresta do gadget de variável e (ao menos) uma aresta que vai para um gadget de cláusula.
Se um literal escolhido em um gadget de variável aparece em uma cláusula, coloque os outros dois vértices do gadget dessa cláusula em S.
Isso cobre todas as arestas do gadget de cláusula e as outras arestas que vão para os gadgets de variável que ainda não estavam cobertas.

Outra linguagem NP-completa: VERTEXCOVER

Teorema

VERTEXCOVER é NP-completa.

Demonstração (continuação): Suponha que $\langle \phi \rangle \in$ 3SAT. Então ϕ é satisfatível.
Em cada gadget de variável, escolha o vértice correspondente ao literal que tem valor `true` e coloque em um conjunto S (note que não será possível escolher ambos os vértices do gadget). Isso cobre a aresta do gadget de variável e (ao menos) uma aresta que vai para um gadget de cláusula.
Se um literal escolhido em um gadget de variável aparece em uma cláusula, coloque os outros dois vértices do gadget dessa cláusula em S.
Isso cobre todas as arestas do gadget de cláusula e as outras arestas que vão para os gadgets de variável que ainda não estavam cobertas.
Então S é uma cobertura por vértices com $n + 2m$ vértices, e $n + 2m = k$.

Outra linguagem NP-completa: VERTEXCOVER

Teorema

VERTEXCOVER é NP-completa.

Demonstração (continuação): Suponha que $\langle \phi \rangle \in$ 3SAT. Então ϕ é satisfatível.
Em cada gadget de variável, escolha o vértice correspondente ao literal que tem valor `true` e coloque em um conjunto S (note que não será possível escolher ambos os vértices do gadget). Isso cobre a aresta do gadget de variável e (ao menos) uma aresta que vai para um gadget de cláusula.
Se um literal escolhido em um gadget de variável aparece em uma cláusula, coloque os outros dois vértices do gadget dessa cláusula em S.
Isso cobre todas as arestas do gadget de cláusula e as outras arestas que vão para os gadgets de variável que ainda não estavam cobertas.
Então S é uma cobertura por vértices com $n + 2m$ vértices, e $n + 2m = k$. Logo, $\langle G, k \rangle \in$ VERTEXCOVER.

Outra linguagem NP-completa: VERTEXCOVER

Teorema

VERTEXCOVER é NP-completa.

Demonstração (continuação): Suponha agora que $\langle G, k \rangle \in$ VERTEXCOVER. Então existe $S \subseteq V(G)$ que é uma cobertura por vértices e tal que $|S| \leq k$.

Outra linguagem NP-completa: VERTEXCOVER

Teorema

VERTEXCOVER é NP-completa.

Demonstração (continuação): Suponha agora que $\langle G, k \rangle \in$ VERTEXCOVER. Então existe $S \subseteq V(G)$ que é uma cobertura por vértices e tal que $|S| \leq k$.
Certamente ao menos um vértice de cada gadget de variável está em S, para cobrir a aresta desse gadget. Também, ao menos dois vértices de cada gadget de cláusula estão em S, para cobrir as três arestas desse gadget.

Outra linguagem NP-completa: VERTEXCOVER

Teorema

VERTEXCOVER é NP-completa.

Demonstração (continuação): Suponha agora que $\langle G, k \rangle \in$ VERTEXCOVER. Então existe $S \subseteq V(G)$ que é uma cobertura por vértices e tal que $|S| \leq k$.
Certamente ao menos um vértice de cada gadget de variável está em S, para cobrir a aresta desse gadget. Também, ao menos dois vértices de cada gadget de cláusula estão em S, para cobrir as três arestas desse gadget.
Mas isso já são $n + 2m$ vértices, de forma que concluímos que $|S| = k$. Tome então os n vértices que estão nas gadgets de variáveis e atribua `true` para os literais correspondentes.

Outra linguagem NP-completa: VERTEXCOVER

Teorema

VERTEXCOVER é NP-completa.

Demonstração (continuação): Suponha agora que $\langle G, k \rangle \in$ VERTEXCOVER. Então existe $S \subseteq V(G)$ que é uma cobertura por vértices e tal que $|S| \leq k$.
Certamente ao menos um vértice de cada gadget de variável está em S, para cobrir a aresta desse gadget. Também, ao menos dois vértices de cada gadget de cláusula estão em S, para cobrir as três arestas desse gadget.
Mas isso já são $n + 2m$ vértices, de forma que concluímos que $|S| = k$. Tome então os n vértices que estão nas gadgets de variáveis e atribua `true` para os literais correspondentes. Isto é uma atribuição que satisfaz ϕ, já que toda cláusula tem uma aresta entre ela e um gadget de variável que é coberta justamente por um vértice do gadget de variável, já que apenas dois vértices do gadget de cláusula estão em S.

Outra linguagem NP-completa: VERTEXCOVER

Teorema

VERTEXCOVER é NP-completa.

Demonstração (continuação): Suponha agora que $\langle G, k \rangle \in$ VERTEXCOVER. Então existe $S \subseteq V(G)$ que é uma cobertura por vértices e tal que $|S| \leq k$.
Certamente ao menos um vértice de cada gadget de variável está em S, para cobrir a aresta desse gadget. Também, ao menos dois vértices de cada gadget de cláusula estão em S, para cobrir as três arestas desse gadget.
Mas isso já são $n + 2m$ vértices, de forma que concluímos que $|S| = k$. Tome então os n vértices que estão nas gadgets de variáveis e atribua `true` para os literais correspondentes. Isto é uma atribuição que satisfaz ϕ, já que toda cláusula tem uma aresta entre ela e um gadget de variável que é coberta justamente por um vértice do gadget de variável, já que apenas dois vértices do gadget de cláusula estão em S. Logo, $\langle \phi \rangle \in$ 3SAT. C.Q.D.

O problema da cobertura por vértices – parte 2

Outra demonstração de NP-completude para VERTEXCOVER

Teorema

VERTEXCOVER é NP-completa.

Demonstração alternativa: Já sabemos que VERTEXCOVER está em NP e depois faremos uma redução polinomial de CLIQUE para VERTEXCOVER.
Considere um grafo G e um valor k inteiro, instância para CLIQUE.

Outra demonstração de NP-completude para VERTEXCOVER

Teorema

VERTEXCOVER é NP-completa.

Demonstração alternativa: Já sabemos que VERTEXCOVER está em NP e depois faremos uma redução polinomial de CLIQUE para VERTEXCOVER.
Considere um grafo G e um valor k inteiro, instância para CLIQUE.
O grafo complemento de G é o grafo $\overline{G}$, em que $V(\overline{G}) = V(G)$ e $E(\overline{G}) = \{uv \colon uv \notin E(G)\}$.

Outra demonstração de NP-completude para VERTEXCOVER

Teorema

VERTEXCOVER é NP-completa.

Demonstração alternativa: Já sabemos que VERTEXCOVER está em NP e depois faremos uma redução polinomial de CLIQUE para VERTEXCOVER.
Considere um grafo G e um valor k inteiro, instância para CLIQUE.
O grafo complemento de G é o grafo $\overline{G}$, em que $V(\overline{G}) = V(G)$ e $E(\overline{G}) = \{uv\colon uv \notin E(G)\}$.
Tomando $t = |V(G)| - k$, temos então uma instância $\langle \overline{G}, t \rangle$ para VERTEXCOVER, claramente construída em tempo polinomial no tamanho de G.

Outra demonstração de NP-completude para VERTEXCOVER

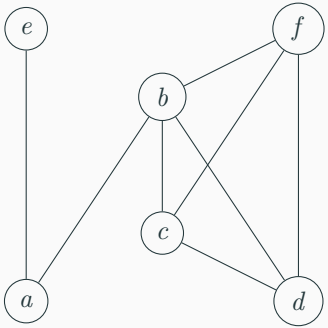
Teorema

VERTEXCOVER é NP-completa.

Demonstração alternativa: Já sabemos que VERTEXCOVER está em NP e depois faremos uma redução polinomial de CLIQUE para VERTEXCOVER.
Considere um grafo G e um valor k inteiro, instância para CLIQUE.
O grafo complemento de G é o grafo $\overline{G}$, em que $V(\overline{G}) = V(G)$ e $E(\overline{G}) = \{uv\colon uv \notin E(G)\}$.
Tomando $t = |V(G)| - k$, temos então uma instância $\langle \overline{G}, t\rangle$ para VERTEXCOVER, claramente construída em tempo polinomial no tamanho de G. Resta verificar que $\langle G, k\rangle \in$ CLIQUE se e somente se $\langle \overline{G}, t\rangle \in$ VERTEXCOVER.

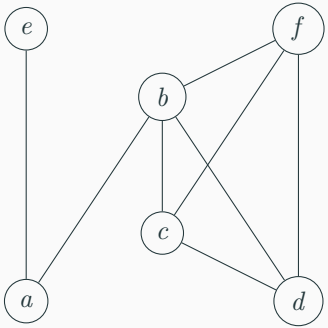
Exemplo

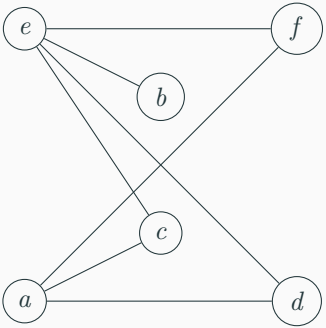
Dado G, considere $\overline{G}$:



Exemplo

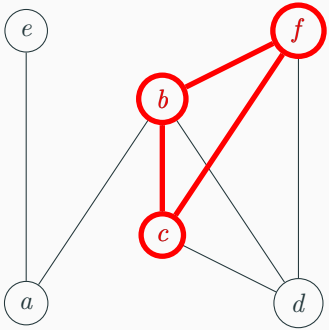
Dado G, considere $\overline{G}$:

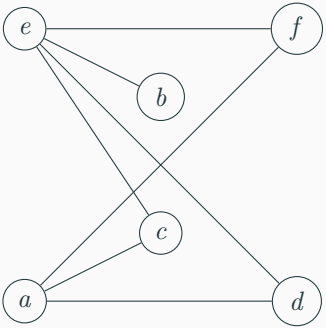




Exemplo

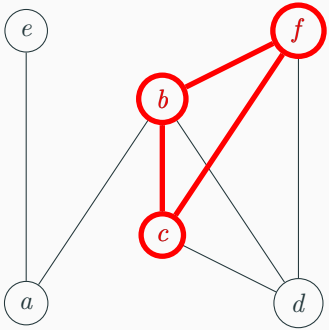
Dado G, considere $\overline{G}$:

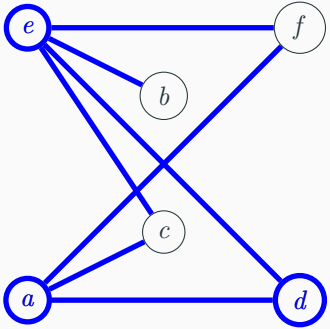




Exemplo

Dado G, considere $\overline{G}$:





Outra demonstração de NP-completude para VERTEXCOVER

Teorema

VERTEXCOVER é NP-completa.

Demonstração alternativa: Suponha que $\langle G, k \rangle \in$ CLIQUE. Então G contém uma clique S de tamanho pelo menos k.

Outra demonstração de NP-completude para VERTEXCOVER

Teorema

VERTEXCOVER é NP-completa.

Demonstração alternativa: Suponha que $\langle G, k \rangle \in$ CLIQUE. Então G contém uma clique S de tamanho pelo menos k.
Isso significa que para todo par $u, v \in S$ temos $uv \in E(G)$, o que implica em $uv \notin E(\overline{G})$.

Outra demonstração de NP-completude para VERTEXCOVER

Teorema

VERTEXCOVER é NP-completa.

Demonstração alternativa: Suponha que $\langle G, k \rangle \in$ CLIQUE. Então G contém uma clique S de tamanho pelo menos k.
Isso significa que para todo par $u, v \in S$ temos $uv \in E(G)$, o que implica em $uv \notin E(\overline{G})$.
Então para toda aresta $xy \in E(\overline{G})$, devemos ter que $x \notin S$ ou $y \notin S$, isto é, $x \in V(G) \setminus S$ ou $y \in V(G) \setminus S$.

Outra demonstração de NP-completude para VERTEXCOVER

Teorema

VERTEXCOVER é NP-completa.

Demonstração alternativa: Suponha que $\langle G, k \rangle \in$ CLIQUE. Então G contém uma clique S de tamanho pelo menos k.
Isso significa que para todo par $u, v \in S$ temos $uv \in E(G)$, o que implica em $uv \notin E(\overline{G})$.
Então para toda aresta $xy \in E(\overline{G})$, devemos ter que $x \notin S$ ou $y \notin S$, isto é, $x \in V(G) \setminus S$ ou $y \in V(G) \setminus S$.
Logo, $V(G) \setminus S$ é uma cobertura por vértices de $\overline{G}$.

Outra demonstração de NP-completude para VERTEXCOVER

Teorema

VERTEXCOVER é NP-completa.

Demonstração alternativa: Suponha que $\langle G, k\rangle \in$ CLIQUE. Então G contém uma clique S de tamanho pelo menos k.
Isso significa que para todo par $u, v \in S$ temos $uv \in E(G)$, o que implica em $uv \notin E(\overline{G})$.
Então para toda aresta $xy \in E(\overline{G})$, devemos ter que $x \notin S$ ou $y \notin S$, isto é, $x \in V(G) \setminus S$ ou $y \in V(G) \setminus S$.
Logo, $V(G) \setminus S$ é uma cobertura por vértices de $\overline{G}$. Como $|S| \geq k$, temos $|V(G) \setminus S| = |V(G)| - |S| \leq |V(G)| - k = t$. Então $\langle \overline{G}, t\rangle \in$ VERTEXCOVER.

Outra demonstração de NP-completude para VERTEXCOVER

Teorema

VERTEXCOVER é NP-completa.

Demonstração alternativa: Agora suponha que $\langle \overline{G}, t \rangle \in$ VERTEXCOVER. Então $\overline{G}$ contém uma cobertura por vértices S de tamanho no máximo t.

Outra demonstração de NP-completude para VERTEXCOVER

Teorema

VERTEXCOVER é NP-completa.

Demonstração alternativa: Agora suponha que $\langle \overline{G}, t \rangle \in$ VERTEXCOVER. Então $\overline{G}$ contém uma cobertura por vértices S de tamanho no máximo t.
Isso significa que se $uv \in E(\overline{G})$, então $u \in S$ ou $v \in S$.

Outra demonstração de NP-completude para VERTEXCOVER

Teorema

VERTEXCOVER é NP-completa.

Demonstração alternativa: Agora suponha que $\langle \overline{G}, t \rangle \in$ VERTEXCOVER. Então $\overline{G}$ contém uma cobertura por vértices S de tamanho no máximo t.
Isso significa que se $uv \in E(\overline{G})$, então $u \in S$ ou $v \in S$.
A contrapositiva disso nos diz que se $u \notin S$ e $v \notin S$, então $uv \notin E(\overline{G})$, o que por sua vez implica em $uv \in E(G)$.

Outra demonstração de NP-completude para VERTEXCOVER

Teorema

VERTEXCOVER é NP-completa.

Demonstração alternativa: Agora suponha que $\langle \overline{G}, t \rangle \in$ VERTEXCOVER. Então $\overline{G}$ contém uma cobertura por vértices S de tamanho no máximo t.
Isso significa que se $uv \in E(\overline{G})$, então $u \in S$ ou $v \in S$.
A contrapositiva disso nos diz que se $u \notin S$ e $v \notin S$, então $uv \notin E(\overline{G})$, o que por sua vez implica em $uv \in E(G)$.
Logo, $V(G) \setminus S$ é uma clique em G.

Outra demonstração de NP-completude para VERTEXCOVER

Teorema

VERTEXCOVER é NP-completa.

Demonstração alternativa: Agora suponha que $\langle \overline{G}, t \rangle \in$ VERTEXCOVER. Então $\overline{G}$ contém uma cobertura por vértices S de tamanho no máximo t.
Isso significa que se $uv \in E(\overline{G})$, então $u \in S$ ou $v \in S$.
A contrapositiva disso nos diz que se $u \notin S$ e $v \notin S$, então $uv \notin E(\overline{G})$, o que por sua vez implica em $uv \in E(G)$.
Logo, $V(G) \setminus S$ é uma clique em G. Como $S \leq t = |V(G)| - k$, temos
$|V(G) \setminus S| = |V(G)| - |S| \geq |V(G)| - (|V(G)| - k) = k$.

Outra demonstração de NP-completude para VERTEXCOVER

Teorema

VERTEXCOVER é NP-completa.

Demonstração alternativa: Agora suponha que $\langle \overline{G}, t \rangle \in$ VERTEXCOVER. Então $\overline{G}$ contém uma cobertura por vértices S de tamanho no máximo t.
Isso significa que se $uv \in E(\overline{G})$, então $u \in S$ ou $v \in S$.
A contrapositiva disso nos diz que se $u \notin S$ e $v \notin S$, então $uv \notin E(\overline{G})$, o que por sua vez implica em $uv \in E(G)$.
Logo, $V(G) \setminus S$ é uma clique em G. Como $S \leq t = |V(G)| - k$, temos $|V(G) \setminus S| = |V(G)| - |S| \geq |V(G)| - (|V(G)| - k) = k$. Então $\langle G, k \rangle \in$ CLIQUE.

O problema do ciclo Hamiltoniano

Outra linguagem NP-completa: CICLOHAMILT

Um *ciclo Hamiltoniano* é um ciclo que passa por todos os vértices; isto é, é uma sequência $(v_1, \ldots, v_n, v_1 = v_{n+1})$ dos vértices tais que $v_i v_{i+1} \in E(G)$ para todo $1 \leq i \leq n$.

CICLOHAMILT $= \{\langle G \rangle : G$ é um grafo que tem um ciclo Hamiltoniano$\}$

Outra linguagem NP-completa: CICLOHAMILT

Um *ciclo Hamiltoniano* é um ciclo que passa por todos os vértices; isto é, é uma sequência $(v_1, \ldots, v_n, v_1 = v_{n+1})$ dos vértices tais que $v_i v_{i+1} \in E(G)$ para todo $1 \leq i \leq n$.

CICLOHAMILT $= \{\langle G \rangle : G$ é um grafo que tem um ciclo Hamiltoniano$\}$

Teorema

CICLOHAMILT é NP-completa.

Demonstração: Primeiro mostramos que CICLOHAMILT está em NP e depois faremos uma redução polinomial de VERTEXCOVER para CICLOHAMILT.

Outra linguagem NP-completa: CICLOHAMILT

Um *ciclo Hamiltoniano* é um ciclo que passa por todos os vértices; isto é, é uma sequência $(v_1, \ldots, v_n, v_1 = v_{n+1})$ dos vértices tais que $v_i v_{i+1} \in E(G)$ para todo $1 \leq i \leq n$.

CICLOHAMILT $= \{\langle G \rangle : G$ é um grafo que tem um ciclo Hamiltoniano$\}$

Teorema

CICLOHAMILT é NP-completa.

Demonstração: Primeiro mostramos que CICLOHAMILT está em NP e depois faremos uma redução polinomial de VERTEXCOVER para CICLOHAMILT.

CLICLOHAMILT está em NP pois, dada uma sequência de vértices, podemos verificar se todos os vértice estão nela e se todo par de vértices consecutivos tem uma aresta entre eles no grafo. Isso pode ser feito apenas olhando a vizinhança de cada vértice da sequência e, portanto, leva tempo $O(|V(G)||E(G)|)$.

Outra linguagem NP-completa: CICLOHAMILT

Teorema

CICLOHAMILT é NP-completa.

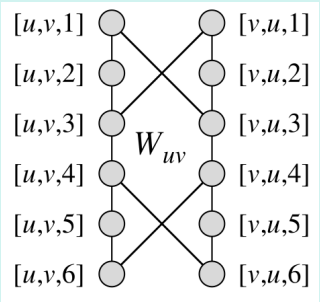
Demonstração (continuação): Considere um grafo G e um valor k inteiro, instância para VERTEXCOVER. Vamos construir um grafo G' da seguinte forma.

Outra linguagem NP-completa: CICLOHAMILT

Teorema

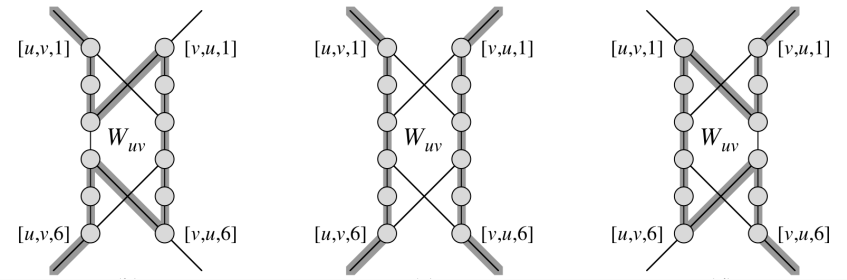
CICLOHAMILT é NP-completa.

Demonstração (continuação): Considere um grafo G e um valor k inteiro, instância para VERTEXCOVER. Vamos construir um grafo G' da seguinte forma.
Para cada aresta $uv \in E(G)$, construa o seguinte grafo W_{uv} (*widget*) que tem 12 vértices e 14 arestas e inclua-o em G':



Outra linguagem NP-completa: CICLOHAMILT

Intuição: apenas os vértices $[u, v, 1]$, $[u, v, 6]$, $[v, u, 1]$ e $[v, u, 6]$ terão arestas incidentes de fora do widget.



Outra linguagem NP-completa: CICLOHAMILT

Teorema

CICLOHAMILT é NP-completa.

Demonstração (continuação): Dado um vértice $u \in V(G)$, sejam $v_1, \ldots, v_{d(u)}$ seus vizinhos. Adicione em G' as arestas $\{\{[u, v_i, 6], [u, v_{i+1}, 1]\} : 1 \leq i < d(u)\}$.

Intuição: note que isso cria um caminho em G' passando por todos os widgets que correspondem às arestas incidentes ao vértice u.

Outra linguagem NP-completa: CICLOHAMILT

Teorema

CICLOHAMILT é NP-completa.

Demonstração (continuação): Dado um vértice $u \in V(G)$, sejam $v_1, \ldots, v_{d(u)}$ seus vizinhos. Adicione em G' as arestas $\{\{[u, v_i, 6], [u, v_{i+1}, 1]\} : 1 \leq i < d(u)\}$.

Intuição: note que isso cria um caminho em G' passando por todos os widgets que correspondem às arestas incidentes ao vértice u.

Por fim, insira em G' outros k vértices $s_1, \ldots, s_k$, denominados *seletores*. Inclua as arestas $\{\{s_j, [u, v_1, 1]\} : u \in V(G) \text{ e } 1 \leq j \leq k\} \cup \{\{s_j, [u, v_{d(u)}, 6]\} : u \in V(G) \text{ e } 1 \leq j \leq k\}$.

Intuição: cada seletor vai indicar um vértice para a cobertura; veja que apenas duas arestas do ciclo Hamiltoniano podem passar por cada s_j.

Outra linguagem NP-completa: CICLOHAMILT

Teorema

CICLOHAMILT é NP-completa.

Demonstração (continuação): Dado um vértice $u \in V(G)$, sejam $v_1, \ldots, v_{d(u)}$ seus vizinhos. Adicione em G' as arestas $\{\{[u, v_i, 6], [u, v_{i+1}, 1]\} : 1 \leq i < d(u)\}$.

Intuição: note que isso cria um caminho em G' passando por todos os widgets que correspondem às arestas incidentes ao vértice u.

Por fim, insira em G' outros k vértices $s_1, \ldots, s_k$, denominados *seletores*. Inclua as arestas $\{\{s_j, [u, v_1, 1]\} : u \in V(G) \text{ e } 1 \leq j \leq k\} \cup \{\{s_j, [u, v_{d(u)}, 6]\} : u \in V(G) \text{ e } 1 \leq j \leq k\}$.

Intuição: cada seletor vai indicar um vértice para a cobertura; veja que apenas duas arestas do ciclo Hamiltoniano podem passar por cada s_j.

A construção levou tempo polinomial no tamanho de G, pois, sendo $n = |V(G)|$ e $m = |E(G)|$, note que G' tem $12m + k \leq 12m + n$ vértices e $16m + (2k - 1)n \leq 16m + (2n - 1)n$ arestas.

Outra linguagem NP-completa: CICLOHAMILT

Teorema

CICLOHAMILT é NP-completa.

Demonstração (continuação): Dado um vértice $u \in V(G)$, sejam $v_1, \ldots, v_{d(u)}$ seus vizinhos. Adicione em G' as arestas $\{\{[u, v_i, 6], [u, v_{i+1}, 1]\} : 1 \leq i < d(u)\}$.

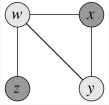
Intuição: note que isso cria um caminho em G' passando por todos os widgets que correspondem às arestas incidentes ao vértice u.

Por fim, insira em G' outros k vértices $s_1, \ldots, s_k$, denominados *seletores*. Inclua as arestas $\{\{s_j, [u, v_1, 1]\} : u \in V(G) \text{ e } 1 \leq j \leq k\} \cup \{\{s_j, [u, v_{d(u)}, 6]\} : u \in V(G) \text{ e } 1 \leq j \leq k\}$.

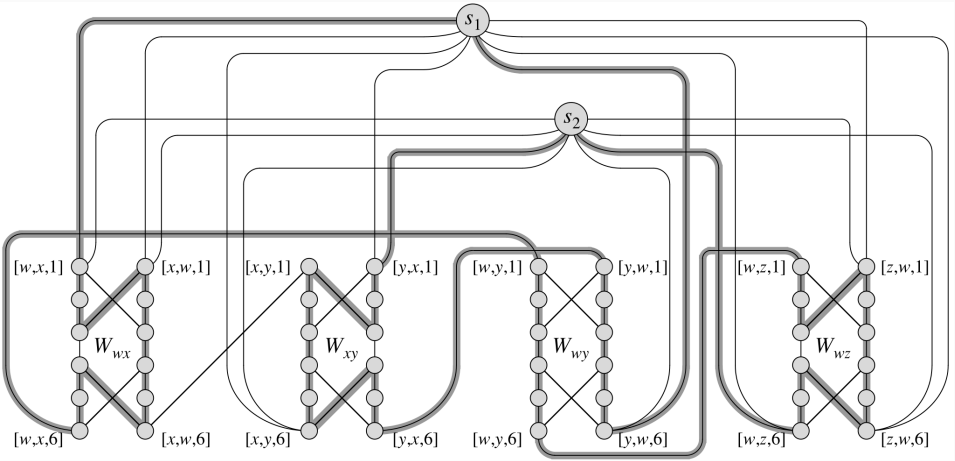
Intuição: cada seletor vai indicar um vértice para a cobertura; veja que apenas duas arestas do ciclo Hamiltoniano podem passar por cada s_j.

A construção levou tempo polinomial no tamanho de G, pois, sendo $n = |V(G)|$ e $m = |E(G)|$, note que G' tem $12m + k \leq 12m + n$ vértices e $16m + (2k - 1)n \leq 16m + (2n - 1)n$ arestas.
Resta mostrar que $\langle G, k \rangle \in$ VERTEXCOVER se e somente se $\langle G' \rangle \in$ CICLOHAMILT. $\star$ C.Q.D.

Exemplo

Para o grafo G com 4 vértices 

, o grafo G' correspondente é



O problema do Caixeiro Viajante

Outra linguagem NP-completa: TSP

TSP $= \{\langle G, w, k\rangle : G$ é um grafo completo, $w \in E(G) \to \mathbb{Z}$ e $k \in \mathbb{Z}$ são tais que G tem um ciclo Hamiltoniano de custo $\leq k\}$

O custo de um ciclo é a soma dos custos de suas arestas.

Outra linguagem NP-completa: TSP

TSP $= \{\langle G, w, k\rangle : G$ é um grafo completo, $w \in E(G) \to \mathbb{Z}$ e $k \in \mathbb{Z}$ são tais que G tem um ciclo Hamiltoniano de custo $\leq k\}$

O custo de um ciclo é a soma dos custos de suas arestas.

Teorema

TSP é NP-completa.

Demonstração: Primeiro mostramos que TSP está em NP e depois faremos uma redução polinomial de CICLOHAMILT para TSP.

Outra linguagem NP-completa: TSP

TSP $= \{\langle G, w, k\rangle : G$ é um grafo completo, $w \in E(G) \to \mathbb{Z}$ e $k \in \mathbb{Z}$ são tais que G tem um ciclo Hamiltoniano de custo $\leq k\}$

O custo de um ciclo é a soma dos custos de suas arestas.

Teorema

TSP é NP-completa.

Demonstração: Primeiro mostramos que TSP está em NP e depois faremos uma redução polinomial de CICLOHAMILT para TSP.

TSP está em NP pois, dada uma sequência de vértices, podemos verificar se todos os vértice estão nela, se todo par de vértices consecutivos tem uma aresta entre eles no grafo e se o custo desse ciclo é no máximo k. Isso pode ser feito apenas olhando a vizinhança de cada vértice da sequência e, portanto, leva tempo $O(|V(G)||E(G)|)$.

Outra linguagem NP-completa: CICLOHAMILT

Teorema

CICLOHAMILT é NP-completa.

Demonstração (continuação): Considere um grafo G, instância para CICLOHAMILT.

Outra linguagem NP-completa: CICLOHAMILT

Teorema

CICLOHAMILT é NP-completa.

Demonstração (continuação): Considere um grafo G, instância para CICLOHAMILT.

Crie um novo grafo G' completo, com os mesmos vértices de G. Atribua os custos da seguinte forma: $w(e) = 0$ se $e \in E(G)$ e $w(e) = 1$ se $e \notin E(G)$. Por fim, tome $k = 0$.

Outra linguagem NP-completa: CICLOHAMILT

Teorema

CICLOHAMILT é NP-completa.

Demonstração (continuação): Considere um grafo G, instância para CICLOHAMILT.

Crie um novo grafo G' completo, com os mesmos vértices de G. Atribua os custos da seguinte forma: $w(e) = 0$ se $e \in E(G)$ e $w(e) = 1$ se $e \notin E(G)$. Por fim, tome $k = 0$.

Resta provar que $\langle G \rangle \in$ CICLOHAMILT se e somente se $\langle G', w, k \rangle \in$ TSP. (Exercício)

Resumo

Dizer que um problema está em NP não é argumento para sua intratabilidade.

Dizer que ele é NP-completo, sim.