NP-úplnost

Důkazy NP-úplnosti problémů můžeme rozdělit na:

1. Přímé - podobně jako pro SAT se ukáže přímo převoditelnost všech možných problémů z NP na zadaný problém. Nejčastěji se využívá posloupnosti konfigurací Turingova stroje - odpověď na problém pro který dokazujeme NP-úplnost bude ANO právě tehdy, když existuje přijímající výpočet daného Turingova stroje na daném slově.
2. Nepřímé - převodem z nějakého již známého NP-úplného problému. Vzniká vlastně sekvence převodů začínající nějakým s přímo dokázanou NP-úplností a končící problémem, pro který NP-úplnost právě dokazujeme.

Všechny důkazy, které budeme dále dělat, budou vedeny způsobem 2. Ten je obecně častěji používaný.

Polynomiální převoditelnost

Definice

Problém P_1 je **polynomiálně převeditelný** na problém P_2, jestliže existuje algoritmus s polynomiální časovou složitostí, který pro libovolný vstup w problému P_1 sestrojí vstup w' problému P_2, přičemž platí, že odpověď na otázku problému P_1 pro vstup w je stejná jako odpověď na otázku problému P_2 pro vstup w'.

- Algoritmus převodu musí být obecný, tedy ke každému možnému vstupu problému P_1 umět sestrojit vstup problému P_2
- Musíme ověřit, že z každého vstupu s odpovědí ANO vznikne vstup s odpovědí ANO (nebo ekvivalentně, že pokud vznikne vstup s odpovědí NE, tak i původní měl odpověď NE).
- Musíme ověřit, že z každého vstupu s odpovědí NE vznikne vstup s odpovědí NE (nebo ekvivalentně, že pokud vznikne vstup s odpovědí ANO, tak i původní měl odpověď ANO).

SAT - Splnitelnost logických formulí

Vstup: Logická formule Φ v konjunktivní normální formě

Výstup: Ano, pokud existuje ohodnocení μ proměnných takové, aby pravdivostní hodnota formule $\mu(\Phi) = 1$

Příklad: Mějme formuli
$\Phi = (a_1 \vee \neg a_2 \vee a_4) \wedge (\neg a_1 \vee a_3) \wedge (\neg a_1 \vee \neg a_3 \vee \neg a_4)$
Řešení: Pokud ohodnotíme např. $\mu(a_1) = \mu(a_3) = 1$, $\mu(a_2) = \mu(a_4) = 0$ dostaneme $\mu(\Phi) = 1$, tedy odpověď je, že formule je splnitelná.

Příklad: Mějme formuli $\Phi = (a_1 \vee a_2) \wedge \neg a_1 \wedge \neg a_2$
Řešení: Formule není splnitelná.

3-SAT - Splnitelnost logických formulí

Vstup: Logická formule Φ v konjunktivní normální formě taková, že každá klauzule obsahuje právě 3 literály

Výstup: ANO, pokud existuje ohodnocení μ proměnných takové, aby pravdivostní hodnota formule $\mu(\Phi) = 1$

Příklad: Mějme formuli
$\Phi = (a_1 \vee \neg a_2 \vee a_4) \wedge (\neg a_1 \vee a_3 \vee a_3) \wedge (\neg a_1 \vee \neg a_3 \vee \neg a_4)$
Řešení: Pokud ohodnotíme např. $\mu(a_1) = \mu(a_3) = 1$, $\mu(a_2) = \mu(a_4) = 0$ dostaneme $\mu(\Phi) = 1$, tedy odpověď je, že formule je splnitelná.

3-SAT - Splnitelnost logických formulí

Problém 3-SAT je NP-úplný.

- Příslušnost do třídy NP ukážeme stejně jako v případě SAT. Stačí jako svědka vzít ohodnocení všech proměnných a ověřit, zda pro dané ohodnocení je formule pravdivá.
- NP-obtížnost ukážeme převodem z problému SAT

Převod SAT na 3-SAT

Algoritmus převodu musí k obecné formuli v konjunktivní normální formě sestrojit formuli, která bude mít v každé klauzuli právě 3 literály.
Problém působí pouze klauzule, kde jsou méně než 3 nebo více než 3 literály.
Algoritmus převodu:

- Ke klauzuli s 1 literálem x (resp $\neg x$) sestrojíme klauzuli $(x \vee x \vee x)$ (resp. $(\neg x \vee \neg x \vee \neg x)$)
- Ke klauzuli se 2 literály $(x \vee y)$ (kde x, y jsou buď proměnné nebo jejich negace) sestrojíme klauzuli $(x \vee x \vee y)$
- Ke klauzuli s $n > 3$ literály $(x_1 \vee x_2 \vee \ldots \vee x_n)$ sestrojíme $n - 2$ klauzulí:
  $(x_1 \vee x_2 \vee y_1) \wedge (\neg y_1 \vee x_3 \vee y_2) \wedge (\neg y_2 \vee x_4 \vee y_3) \wedge \ldots \wedge (\neg y_{n-3} \vee x_{n-1} \vee x_n)$,
  kde $x_1, x_2, \ldots, x_n$ jsou literály (proměnné nebo negace proměnných) a $y_1, y_2, \ldots, y_{n-3}$ jsou nově přidané proměnné.

Převod SAT na 3-SAT

Příklad: Mějme formuli
$\Phi = (a_1 \vee \neg a_2 \vee a_4) \wedge (\neg a_1 \vee a_3) \wedge (\neg a_1 \vee \neg a_3 \vee \neg a_4 \vee a_5 \vee \neg a_5) \wedge a_5$.
Řešení:

- Klauzule $(a_1 \vee \neg a_2 \vee a_4)$ může zůstat
- Klauzuli $(\neg a_1 \vee a_3)$ nahradíme $(\neg a_1 \vee \neg a_1 \vee a_3)$
- Klauzuli a_5 nahradíme $(a_5 \vee a_5 \vee a_5)$
- Klauzuli $(\neg a_1 \vee \neg a_3 \vee \neg a_4 \vee a_5 \vee \neg a_5)$ nahradíme $(\neg a_1 \vee \neg a_3 \vee y_1) \wedge (\neg y_1 \vee \neg a_4 \vee y_2) \wedge (\neg y_2 \vee a_5 \vee \neg a_5)$

Výsledkem je $(a_1 \vee \neg a_2 \vee a_4) \wedge (\neg a_1 \vee \neg a_1 \vee a_3) \wedge (\neg a_1 \vee \neg a_3 \vee y_1) \wedge (\neg y_1 \vee \neg a_4 \vee y_2) \wedge (\neg y_2 \vee a_5 \vee \neg a_5) \wedge (a_5 \vee a_5 \vee a_5)$

Převod SAT na 3-SAT

- x, $(x \vee x)$ i $(x \vee x \vee x)$ mají stejnou hodnotu při stejném ohodnocení literálu x. Rozšíření klauzule o literál, který se v ní vyskytuje nemá tedy vliv na splnitelnost.
- Pokud byla formule splnitelná vznikne splnitelná formule:
  - Ohodnocení proměnných muselo alespoň jednomu literálu z klauzule $(x_1 \vee x_2 \vee \ldots \vee x_n)$ přiřadit 1.
  - Stejné ohodnocení znamená, že v $(x_1 \vee x_2 \vee y_1) \wedge (\neg y_1 \vee x_3 \vee y_2) \wedge$ $\wedge (\neg y_2 \vee x_4 \vee y_3) \wedge \ldots \wedge (\neg y_{n-3} \vee x_{n-1} \vee x_n)$ bude alespoň jedna klauzule mít hodnotu 1 bez ohledu na hodnoty $y_1, \ldots, y_{n-3}$
  - Pokud to není první klauzule z nově vzniklých, je v ní je literál $\neg y_i$, který můžeme ohodnotit 0 a v předchozí klauzuli je y_i ohodnocena 1 a hodnota této klauzule je tedy také 1. Obdobně nejde-li o poslední klauzuli, je v ní je literál $\neg y_{i+1}$, který můžeme ohodnotit 0 a v následující klauzuli je y_{i+1} ohodnocena 1 a hodnota klauzule je 1.
  - Podobnou úvahu můžeme induktivně opakovat až k první a poslední klauzuli. Pokud v nějakém ohodnocení byla hodnota původní klauzule 1, bude existovat ohodnocení takové, že hodnota všech nových klauzulí bude 1.

Převod SAT na 3-SAT

- Pokud vznikne splnitelná formule byla i původní splnitelná:
  - Ohodnocení, které přiřadí formuli $(x_1 \vee x_2 \vee y_1) \wedge (\neg y_1 \vee x_3 \vee y_2) \wedge$ $\wedge (\neg y_2 \vee x_4 \vee y_3) \wedge \ldots \wedge (\neg y_{n-3} \vee x_{n-1} \vee x_n)$ hodnotu 1 musí $n-2$ výskytům literálů přiřadit 1. Proměnných y je jen $n-3$, jen jedna z y nebo $\neg y$ může být 1 a každá má ve formuli jen jeden výskyt v negované a jeden v nenegované verzi. Tedy alespoň jeden z literálů x musí být ohodnocen 1.
  - Uvažujeme-li na původní formuli stejné ohodnocení, tak musí mít stejný literál x také hodnotu 1.
- Ukázali jsme tedy, že po převodu z odpovědi Ano pro SAT plyne odpověd Ano pro 3-SAT a opačně
- Délka vytvořené formule nebude více než trojnásobně delší než původní
- Problém 3-SAT je tedy NP-úplný

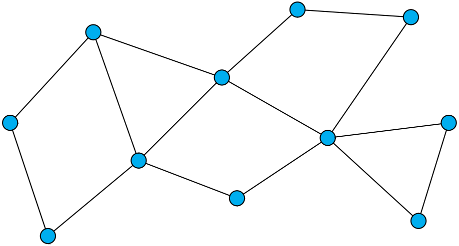
Barvení grafu 3 barvami

3-COL - Problém „Barvení grafu 3 barvami“

Vstup: Neorientovaný graf G

Výstup: Lze vrcholy grafu obarvit 3 barvami tak, aby žádné dva vrcholy spojené hranou neměly stejnou barvu?

Příklad:

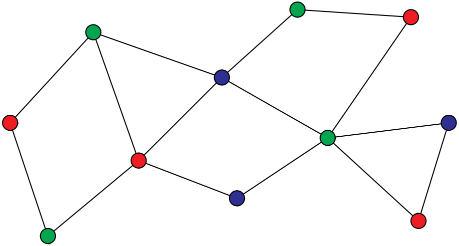


3-COL - Problém „Barvení grafu 3 barvami“

Vstup: Neorientovaný graf G

Výstup: Lze vrcholy grafu obarvit 3 barvami tak, aby žádné dva vrcholy spojené hranou neměly stejnou barvu?

Příklad:



Odpověď: Ano

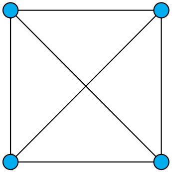
Barvení grafu 3 barvami

3-COL - Problém „Barvení grafu 3 barvami“

Vstup: Neorientovaný graf G

Výstup: Lze vrcholy grafu obarvit 3 barvami tak, aby žádné dva vrcholy spojené hranou neměly stejnou barvu?

Příklad:



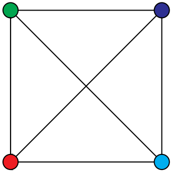
Barvení grafu 3 barvami

3-COL - Problém „Barvení grafu 3 barvami“

Vstup: Neorientovaný graf G

Výstup: Lze vrcholy grafu obarvit 3 barvami tak, aby žádné dva vrcholy spojené hranou neměly stejnou barvu?

Příklad:



Odpověď: NE

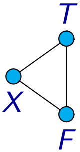
Převod 3-SAT na 3-COL

Hledáme algoritmus, který sestrojí k zadané formuli v konjunktivní normální formě s právě 3 literály v každé klauzuli graf tak, že formule bude splnitelná právě tehdy, když půjde graf obarvit korektně 3 barvami.

- Graf bude obsahovat vrcholy X, F, T propojené do trojúhelníku
- Pro každou proměnnou bude graf mít dvojici vrcholů $x, \neg x$ popojené hranou. Každý z nich bude propojen hranou s X.
- Pro každou klauzuli přidáme 6 vrcholů $c_1, c'_1, c_2, c'_2, c_3, c'_3$ a hrany $(c_1, c_2), (c_1, c_3), (c_2, c_3), (c_1, c'_1), (c_2, c'_2), (c_3, c'_3)$
- Přidáme hrany mezi c'_i a x_j nebo $\neg x_j$ podle toho, jaký literál se nachází na i-té pozici v dané klauzuli

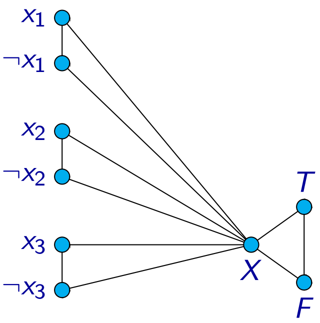
Převod 3-SAT na 3-COL

Příklad: Uvažujme formuli $(x_1 \vee \neg x_2 \vee x_3) \wedge (\neg x_1 \vee x_2 \vee x_3)$

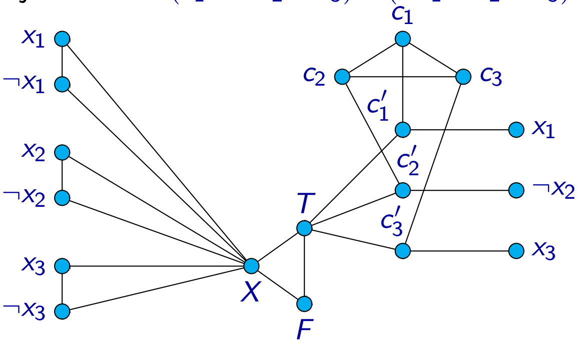


Převod 3-SAT na 3-COL

Příklad: Uvažujme formuli $(x_1 \vee \neg x_2 \vee x_3) \wedge (\neg x_1 \vee x_2 \vee x_3)$

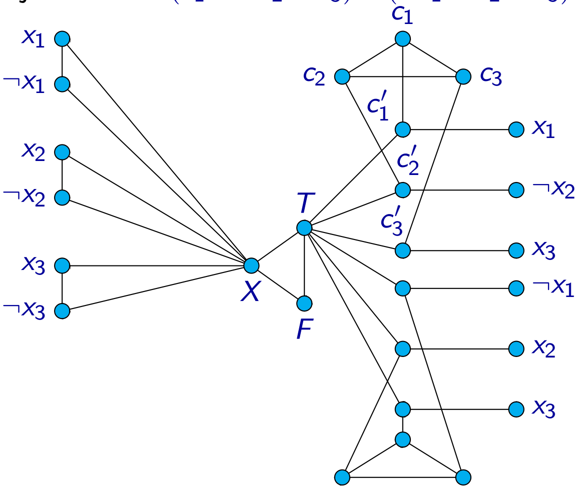


Příklad: Uvažujme formuli $(x_1 \vee \neg x_2 \vee x_3) \wedge (\neg x_1 \vee x_2 \vee x_3)$



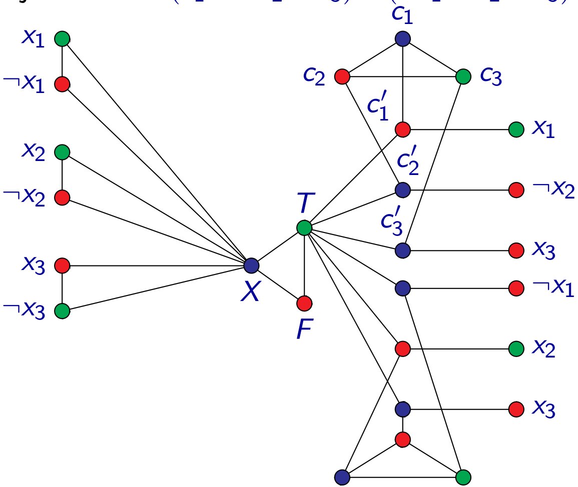
Převod 3-SAT na 3-COL

Příklad: Uvažujme formuli $(x_1 \vee \neg x_2 \vee x_3) \wedge (\neg x_1 \vee x_2 \vee x_3)$



Převod 3-SAT na 3-COL

Příklad: Uvažujme formuli $(x_1 \vee \neg x_2 \vee x_3) \wedge (\neg x_1 \vee x_2 \vee x_3)$



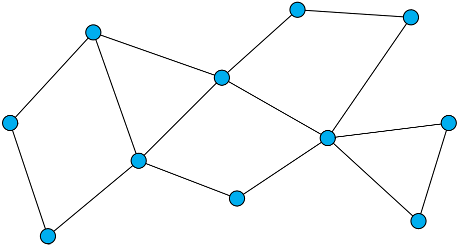
Nezávislá množina

IS - Problém „Nezávislá množina“

Vstup: Neorientovaný graf G a přirozené číslo k

Výstup: Existuje v grafu nezávislá množina velikosti k (množina k vrcholů, které vzájemně nejsou propojeny hranou)?

Příklad: $k = 5$



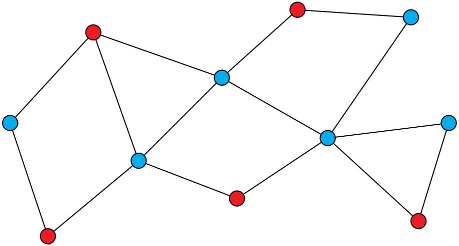
Nezávislá množina

IS - Problém „Nezávislá množina“

Vstup: Neorientovaný graf G a přirozené číslo k

Výstup: Existuje v grafu nezávislá množina velikosti k (množina k vrcholů, které vzájemně nejsou propojeny hranou)?

Příklad: $k = 5$



Odpověď: ANO

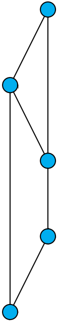
Převod 3-COL na IS

Hledáme algoritmus, který sestrojí ke grafu G graf G' a číslo k tak, že G bude obarvitelný 3 barvami právě tehdy, když v G' bude existovat nezávislá množina velikosti k.

- Graf G' bude obsahovat 3 kopie grafu G
- Všechny tři kopie každého vrcholu budou propojeny hranami do trojúhelníku
- Číslo k položíme rovno $|G|$

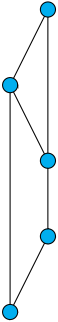
Převod 3-COL na IS

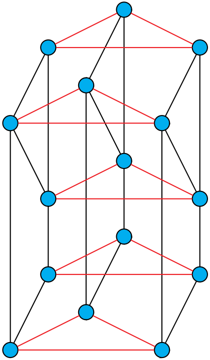
Příklad:



Převod 3-COL na IS

Příklad:

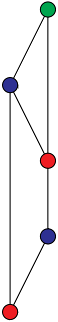


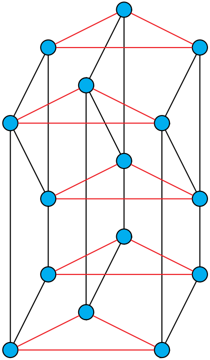


$k = 5$

Převod 3-COL na IS

Příklad:

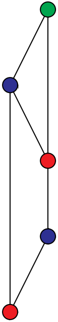


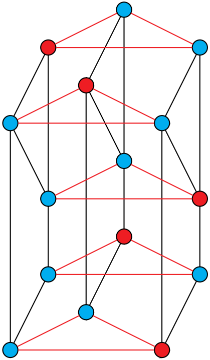


$k = 5$

Převod 3-COL na IS

Příklad:





$k = 5$

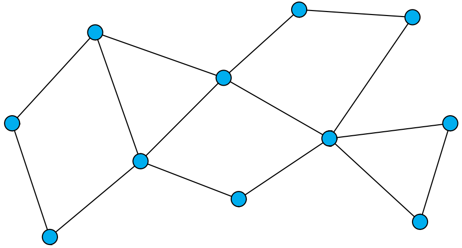
Vrcholové pokrytí

VC - Problém „Vrcholové pokrytí“

Vstup: Neorientovaný graf G a přirozené číslo k

Výstup: Existuje v grafu množina vrcholů velikosti k taková, že každá hrana má alespoň jeden svůj vrchol v této množině?

Příklad: $k = 6$



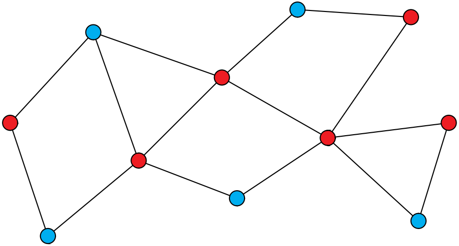
Vrcholové pokrytí

VC - Problém „Vrcholové pokrytí“

Vstup: Neorientovaný graf G a přirozené číslo k

Výstup: Existuje v grafu množina vrcholů velikosti k taková, že každá hrana má alespoň jeden svůj vrchol v této množině?

Příklad: $k = 6$



Odpověď: Ano

Vrcholové pokrytí

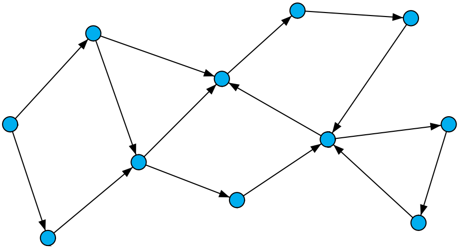
- Vrcholové pokrytí patří do NP. Stačí jako svědka vzít množinu k vrcholů a ověřit, že tato množina tvoří pokrytí.
- NP-obtížnost lehce ukážeme převodem z problému IS
- Graf nemusíme měnit, jen limit k pro pokrytí bude $|G| - n$, kde n je limit pro nezávislou množinu
- Pokud existovala nezávislá množina, tak žádná hrana v ní nemohla mít oba krajní vrcholy. Tedy alespoň jeden krajní vrchol každé hrany je v doplňku nezávislé množiny a ta tedy tvoří pokrytí požadované velikosti.
- Pokud bude existovat pokrytí, každá hrana v něm má alespoň jeden vrchol. Tedy v doplňku nejsou oba krajní vrcholy žádné hrany (žádné dva vrcholy nejsou spojeny hranou) a tedy tento doplněk tvoří nezávislou množinu velikosti $n = |G| - k$

HC - Problém „Hamiltonovský cyklus“

Vstup: Orientovaný graf G

Výstup: Existuje v grafu Hamiltonovský cyklus (orientovaná kružnice procházející každý vrchol právě jednou)?

Příklad:

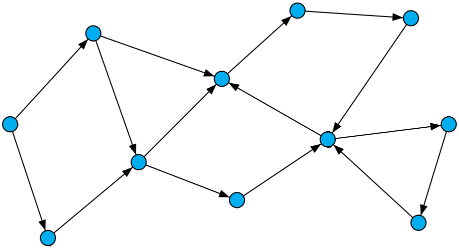


HC - Problém „Hamiltonovský cyklus“

Vstup: Orientovaný graf G

Výstup: Existuje v grafu Hamiltonovský cyklus (orientovaná kružnice procházející každý vrchol právě jednou)?

Příklad:



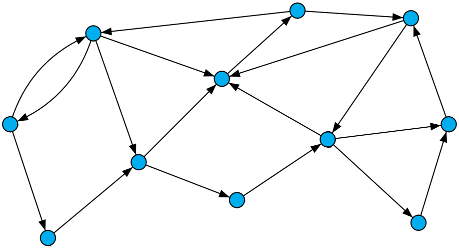
Odpověď: NE

HC - Problém „Hamiltonovský cyklus"

Vstup: Orientovaný graf G

Výstup: Existuje v grafu Hamiltonovský cyklus (orientovaná kružnice procházející každý vrchol právě jednou)?

Příklad:



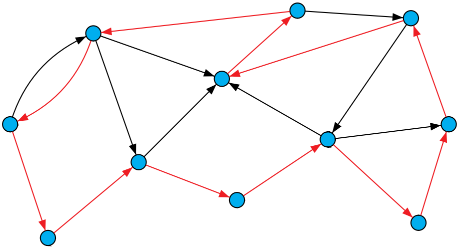
Hamiltonovský cyklus

HC - Problém „Hamiltonovský cyklus“

Vstup: Orientovaný graf G

Výstup: Existuje v grafu Hamiltonovský cyklus (orientovaná kružnice procházející každý vrchol právě jednou)?

Příklad:



Odpověď: ANO

Hamiltonovský cyklus

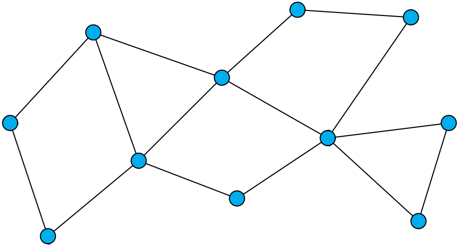
- Problém patří do třídy NP. Stačí vzít jako svědka množinu hran a ověřit, že tyto hrany tvoří cyklus a každý vrchol má právě jednu vstupní a výstupní hranu.
- NP-obtížnost můžeme ukázat například převodem z problému VC nebo SAT
- Oba důkazy jsou složitější na popsání, proto je vynecháme

HK - Problém „Hamiltonovská kružnice“

Vstup: Neorientovaný graf G

Výstup: Existuje v grafu Hamiltonovská kružnice (neorientovaná kružnice procházející každý vrchol právě jednou)?

Příklad:



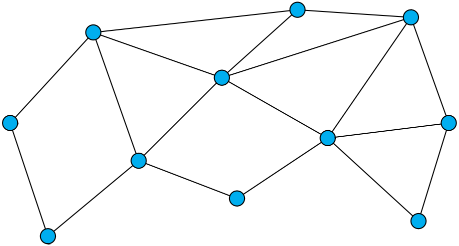
Odpověď: NE

HK - Problém „Hamiltonovská kružnice“

Vstup: Neorientovaný graf G

Výstup: Existuje v grafu Hamiltonovská kružnice (neorientovaná kružnice procházející každý vrchol právě jednou)?

Příklad:

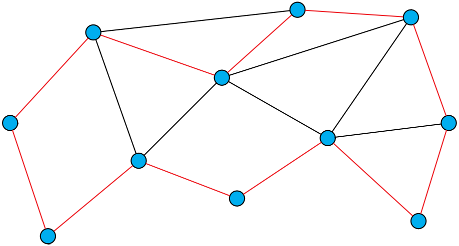


HK - Problém „Hamiltonovská kružnice"

Vstup: Neorientovaný graf G

Výstup: Existuje v grafu Hamiltonovská kružnice (neorientovaná kružnice procházející každý vrchol právě jednou)?

Příklad:

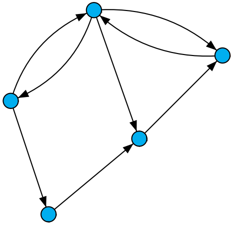


Odpověď: Ano

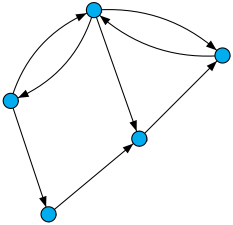
Hamiltonovská kružnice

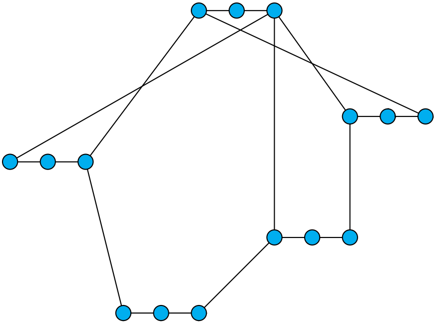
- Problém patří do třídy NP. Stačí vzít jako svědka množinu hran a ověřit, že tyto hrany tvoří kružnici a každý vrchol má právě dvě hrany.
- NP-obtížnost můžeme ukázat převodem z problému HC
- Ke každému vrcholu x orientovaného grafu dáme do neorientovaného tři vrcholy x_1, x_2, x_3, které spojíme hranami $(x_1, x_2), (x_2, x_3)$.
- Každá orientovaná hrana (x, y) bude reprezentována hranou (x_3, y_1)

Převod HC na HK

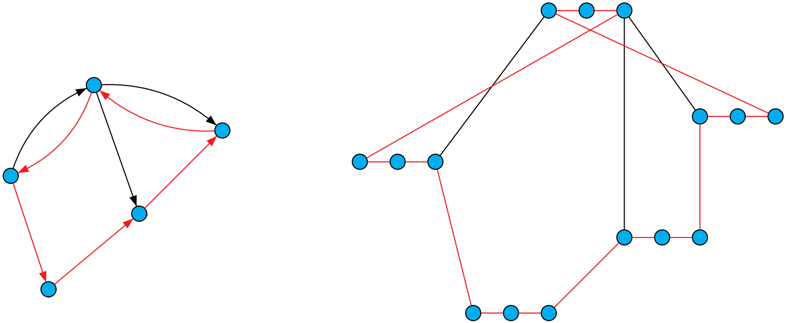


Převod HC na HK





Převod HC na HK



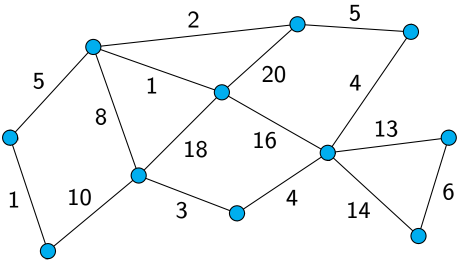
Problém obchodního cestujícího

TSP - Problém „obchodního cestujícího“

Vstup: Neorientovaný graf G s hranami ohodnocenými přirozenými čísly a číslo k

Výstup: Existuje v grafu uzavřený sled procházející všemi vrcholy a mající součet délek hran (včetně opakovaných) maximálně k?

Příklad: $k = 70$

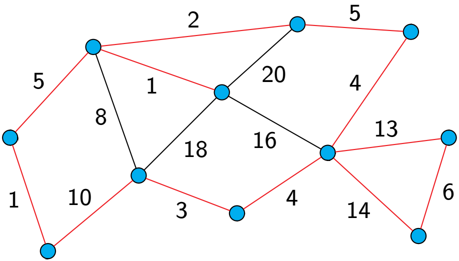


TSP - Problém „obchodního cestujícího“

Vstup: Neorientovaný graf G s hranami ohodnocenými přirozenými čísly a číslo k

Výstup: Existuje v grafu uzavřený sled procházející všemi vrcholy a mající součet délek hran (včetně opakovaných) maximálně k?

Příklad: $k = 70$



Odpověď: Ano, protože byl nalezen sled se součtem 69

Problém obchodního cestujícího

- Problém patří do třídy NP. Stačí vzít jako svědka množinu hran a ověřit, že tyto hrany tvoří uzavřený sled procházející každým vrcholem a součet je menší nebo roven k.
- NP-obtížnost můžeme ukázat převodem z problému HK
- Vstupní graf do problému HK G převedeme na G' (vstup do TSP) tak, že jen ohodnotíme všechny hrany 1.
- Jako limit k pro délku cesty obchodního cestujícího vezmeme $|G|$

PART - Problém „loupežníka“

Vstup: Množina přirozených čísel $A = \{a_1, a_2, \ldots, a_n\}$

Výstup: ANO, jestliže je možné najít množinu X tak, že $X \subset A$, $\sum_{a_i \in X} a_i = \sum_{a_j \in A-X} a_j$.

Příklad: $A = \{10, 40, 60, 70, 100, 120, 140, 160, 300\}$

Řešení:

$X = \{10, 70, 120, 140, 160\}$, $\sum_{a_i \in X} a_i = 500$

$A - X = \{40, 60, 100, 300\}$, $\sum_{a_j \in A-X} a_j = 500$

Převod 3-SAT na problém loupežníka

- Budeme reprezentovat proměnné a klauzule jako úseky bitů v dlouhých binárních číslech
- Pro každou proměnnou je v každém čísle vyhrazen 1 bit
- Pro každou klauzuli jsou vyhrazeny 3 bity
- Pro každý možný literál přidáme jedno číslo s 1 na místě reprezentujícím proměnnou tohoto literálu a dále počtem výskytů literálu v klauzuli v místě vyhrazeném každé klauzuli
- Přidáme jedno číslo F (reprezentující hodnotu false) s 1 na místě všech klauzulí
- Pro každou klauzuli přidáme 2 čísla s jedničkou jen na místě klauzule, pro kterou jsou určena

Převod 3-SAT na problém loupežníka

Příklad: Uvažujme formuli $(x_1 \vee \neg x_2 \vee x_3) \wedge (\neg x_1 \vee x_2 \vee x_3)$

		c_1	c_2
x_1	100	001	000
$\neg x_1$	100	000	001
x_2	010	000	001
$\neg x_2$	010	001	000
x_3	001	001	001
$\neg x_3$	001	000	000
F	000	001	001
c_1	000	001	000
c_1	000	001	000
c_2	000	000	001
c_2	000	000	001

Převod 3-SAT na problém loupežníka

Příklad: Uvažujme formuli $(x_1 \vee \neg x_2 \vee x_3) \wedge (\neg x_1 \vee x_2 \vee x_3)$

		c_1	c_2
x_1	100	001	000
$\neg x_1$	100	000	001
x_2	010	000	001
$\neg x_2$	010	001	000
x_3	001	001	001
$\neg x_3$	001	000	000
F	000	001	001
c_1	000	001	000
c_1	000	001	000
c_2	000	000	001
c_2	000	000	001

		c_1	c_2
x_1	100	001	000
x_2	010	000	001
$\neg x_3$	001	000	000
c_2	000	000	001
c_2	000	000	001
c_1	000	001	000
c_1	000	001	000
	111	011	011

$\neg x_1$	100	000	001
$\neg x_2$	010	001	000
x_3	001	001	001
F	000	001	001
	111	011	011

ILP - Problém „Celočíselné lineární programování“

Vstup: Matice $\mathbf{A}$ a vektor $\vec{b}$ určující soustavu lineárních nerovnic s proměnnými z_i.

$$\mathbf{A} \cdot \vec{z} \leq \vec{b}$$

Výstup: Existuje vektor $\vec{z} \in \{0,1\}^*$ takový, že je soustava nerovnic $\mathbf{A} \cdot \vec{z} \leq \vec{b}$ splněná?

Příklad:

$3z_1 - 2z_2 + 4z_3 \leq 1$
$-4z_1 - 3z_2 \leq -6$
$2z_1 - z_3 \leq 2$

Řešení:

Pro $z_1 = 1$, $z_2 = 1$, $z_3 = 0$ platí všechny nerovnice, takže opověď je ANO.

Převod PART na ILP

- Uvažujme $A = \{a_1, a_2, \ldots, a_n\}$ množinu přirozených čísel, která je vstupem problému loupežníka
- Nechť $a = \sum_{a_i \in A} a_i$.
- Sestavíme sestavu dvou nerovnic:

$$a_1 z_1 + a_2 z_2 + \ldots + a_n z_n \leq a/2$$

$$a_1(1 - z_1) + a_2(1 - z_2) + \ldots + a_n(1 - z_n) \leq a/2$$

- Hodnota $z_i = 1$ nám říká, že číslo a_i dáme do první množiny rozkladu
- Hodnota $(1 - z_i) = 1$ nám říká, že číslo a_i dáme do druhé množiny rozkladu