Třídy složitosti P a NP, NP-úplnost

Cíle přednášky:

1. Definovat, za jakých okolností můžeme problém považovat za efektivně algoritmicky řešitelný.
2. Charakterizovat určitou skupinu úloh, pro které
   - není známo, zda je můžeme efektivně řešit (otevřený problém)
   - pokud ale řešení úlohy známe, můžeme efektivně ověřit jeho správnost
   - jedná se obvykle o *grafové* a *kombinatorické* úlohy, které se často objevují v praxi.

Rozhodovací problém

Rozhodovací problém P :

instance problému, odpověď **ANO** či **NE**

Ekvivalentně - jazyk L :

vstupní slovo w, $w \in L$ nebo $w \notin L$

Polynomiální časová složitost

Co je to efektivní algoritmus?

- každý s polynomiální časovou složitostí

  $O(n^d)$ .. d je libovolná konstanta

Co v případě polynomiálního času s vysokým stupněm polynomu?

n^{100}

Trvá-li vykonání jedné instukce .. 1 μs

2^{100} instrukcí .. doba delší než je stáří vesmíru

Polynomiální časová složitost

1. Praxe: Časová složitost známých polynomiálních algoritmů je obvykle nízkého stupně n , n^2, n^3, n^4

2. Robustní pojem: Výpočetní modely lze vzájemně simulovat v polynomiálním čase (RAM a Turingův stroj).

Polynomiální vs. exponenciální čas

Složitost \ Velikost vstupu	10	20	30	40	50	60
n	.00001 s	.00002 s	.00003 s	.00004 s	.00005 s	.00006 s
n^2	.0001 s	.0004 s	.0009 s	.0016 s	.0025 s	.0036 s
n^3	.001 s	.008 s	.027 s	.064 s	.125 s	.216 s
n^5	.1 s	3.2 s	24.3 s	1.7 min	5.2 min	13 min
2^n	.001 s	1 s	17.9 min	12.7 dní	35.7 let	366 století
3^n	.059 s	58 min	6.5 let	3855 století	2×10^8 století	1.3×10^{13} století

Třída P

- Množina jazyků (rozhodovacích problémů)
- $L \in P \Leftrightarrow$ existuje deterministický Turingův stroj, který rozhoduje L v polynomiálním čase

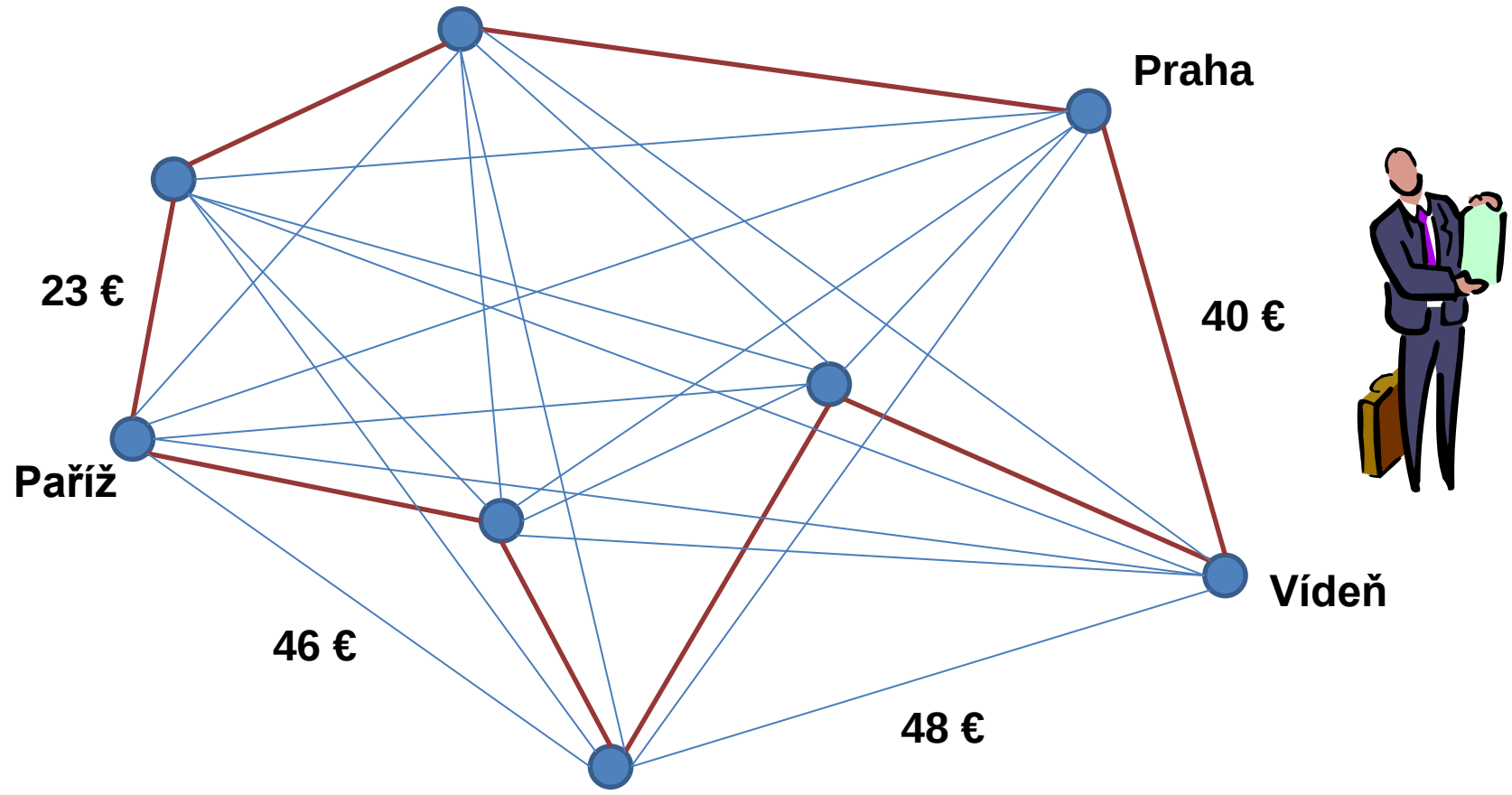
Příklad:

Vstup: Ohodnocený graf G a reálné číslo c

Otázka: Existuje v G kostra s cenou $\leq c$?

Je ve třídě P.

Problém obchodního cestujícího



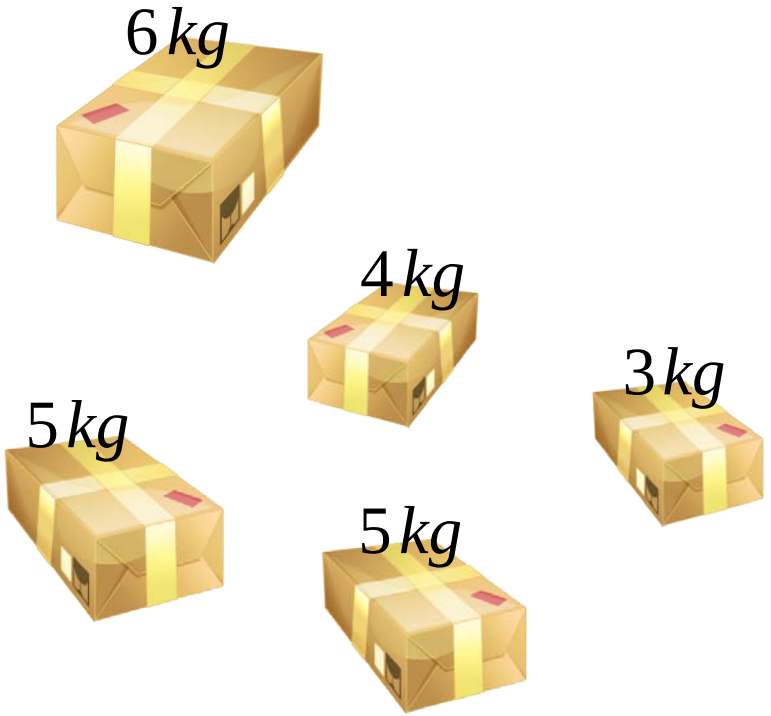
Otázka: Existuje trasa s cenou $\leq c$?

Počet všech tras: $(1/2)(|V|-1)!$ pro $|V| \geq 3$

Problém batohu

Lze vybrat předměty tak, aby jejich celková hmotnost odpovídala nosnosti batohu?

nosnost $12\,kg$



Počet možných výběrů předmětů: 2^n

Problém SAT

SAT = satisfability (splnitelnost)

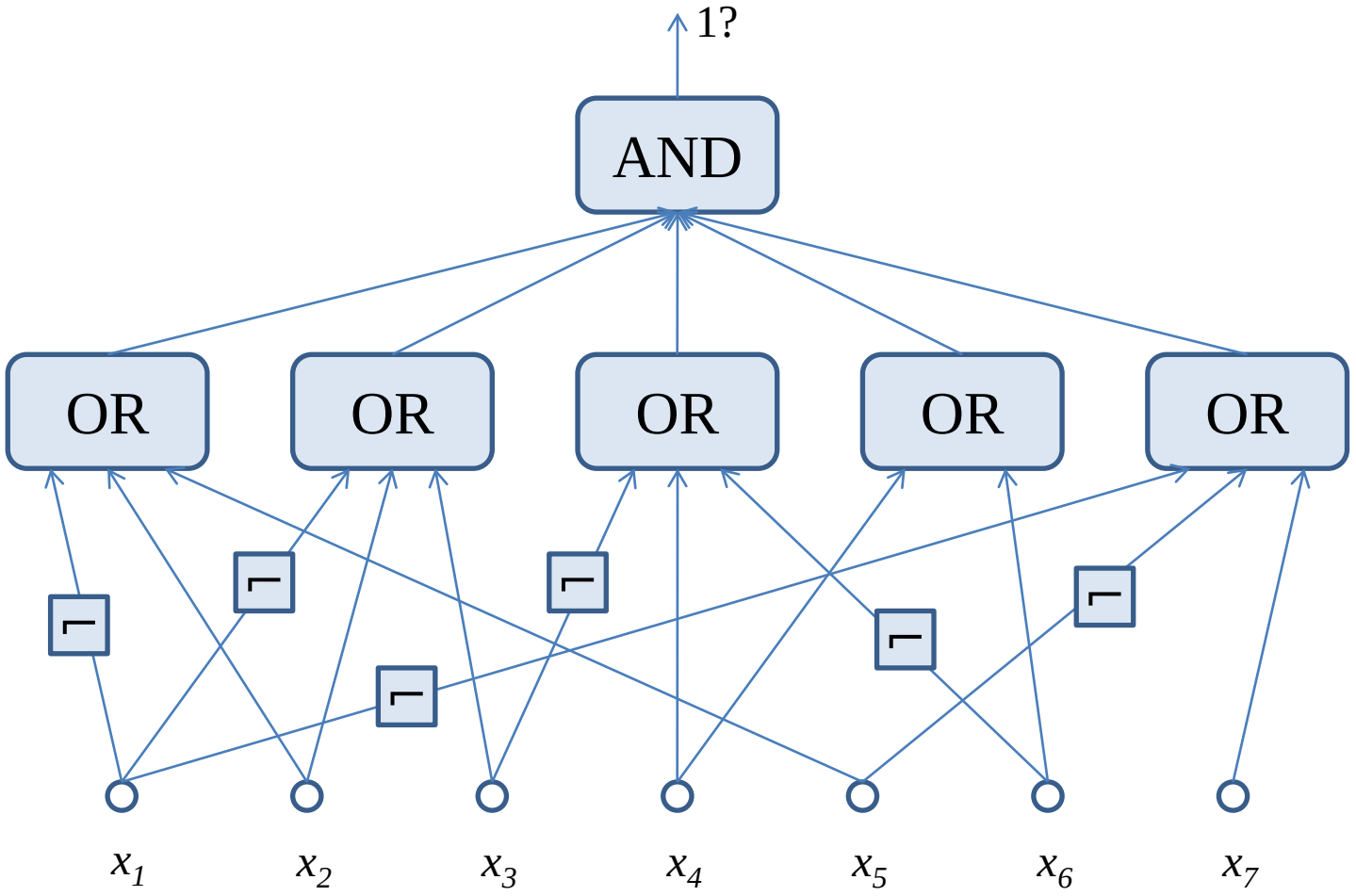
Vstup: Výroková formule v konjunktivně normální formě.

$$\Phi(x_1, x_2, x_3) = (x_1 \vee \neg x_3 \vee x_2) \wedge (x_2 \vee x_1) \wedge (\neg x_2 \vee x_3)$$

Otázka: Je formule splnitelná?

$$\Phi(1,1,1) = 1$$

Logický obvod



počet různých vstupních *n*-tic: 2^n

Třída NP

NP = **N**ondeterministic **p**olynomial time

Definice č.1: $L \in NP \Leftrightarrow$ existuje nedeterministický TS, který rozhoduje L v polynomiálním čase

Definice č.2: $L \in NP \Leftrightarrow$ existuje verifikační algoritmus $A(x, y)$ s polynomiální časovou složitostí p takový, že

$$x \in L \Leftrightarrow \exists y : | y | \leq p(| x |) \wedge A(x, y) = true$$

y se nazývá ověření

Nedeterministický Turingův stroj

- V dané konfiguraci je možné aplikovat více instrukcí.
  např.: $(q_0, a) \to (q_0, a, R)$ a $(q_0, a) \to (q_1, b, L)$

- Výpočet se větví – výpočetní strom
- Vstupní slovo u je přijato, pokud existuje výpočetní větev, která skončí v přijímacím stavu.
- Doba výpočtu pro u:
  $t(u) =$ počet kroků nejdelší výpočetní větve

- Nedeterministický Turingův stroj lze simulovat deterministicky průchodem výpočetního stromu do šířky.

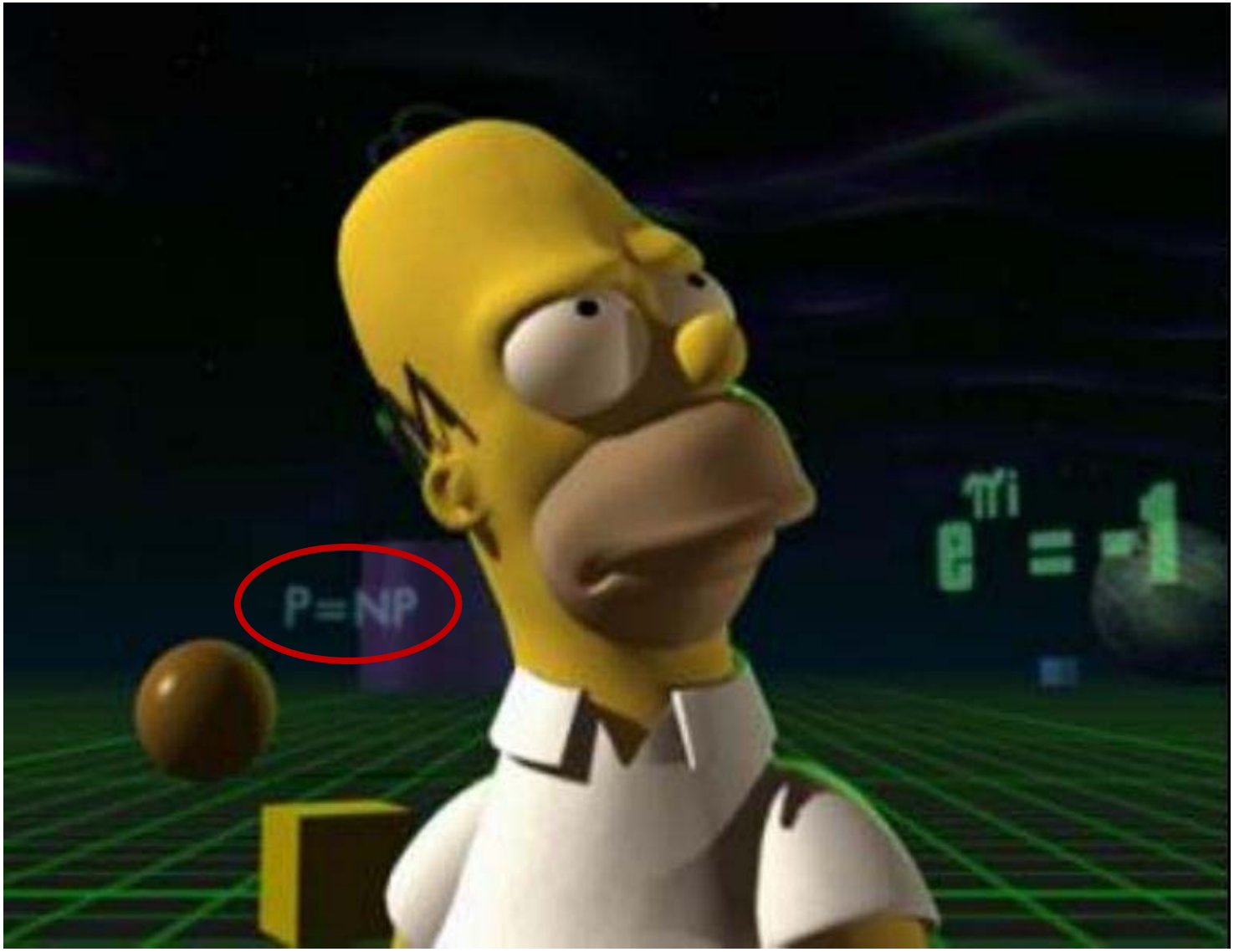
P vs. NP

$P \stackrel{?}{=} NP$ … hlavní otevřená otázka v teoretické informace

Očekávaný výsledek: $P \neq NP$

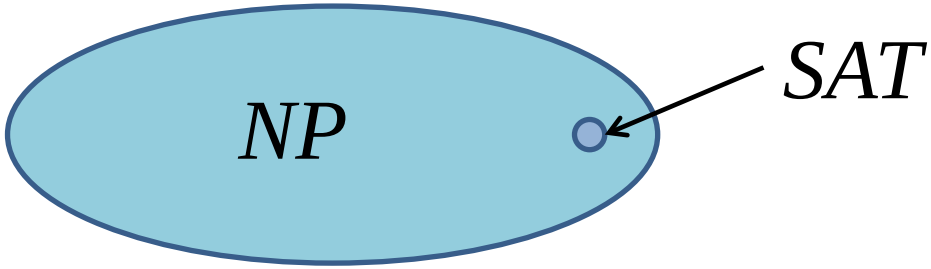
Odměna vypsaná za vyřešení: 1.000.000 USD
(Clay Mathematics Institute, Cambridge)

P vs. NP

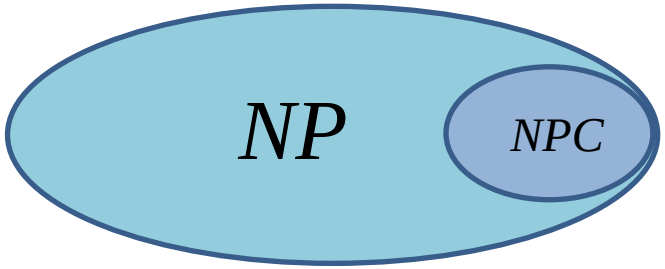


NP-úplné problémy

S. Cook, 1971 – v *NP* existuje problém, pomocí kterého je možné vyřešit všechny ostatní



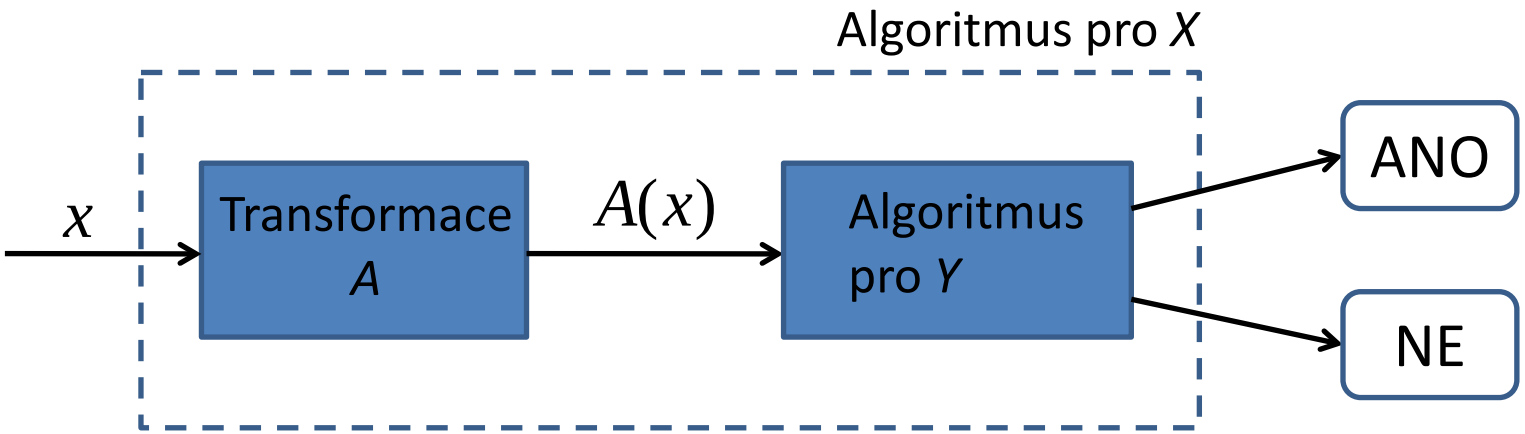
R. Karp, 1972 – cca 20 dalších problémů patří do „elitního klubu“



Polynomiální převeditelnost

Problém (jazyk) X je polynomiálně převeditelný na problém (jazyk) Y právě tehdy, když existuje polynomiální algoritmus (deterministický TS) A takový, že

$$\forall x : x \in X \Leftrightarrow A(x) \in Y$$



Označení: $X \leq_P Y$

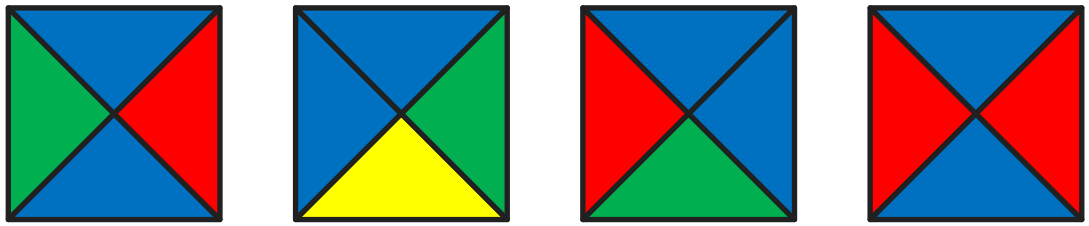
NP-úplnost

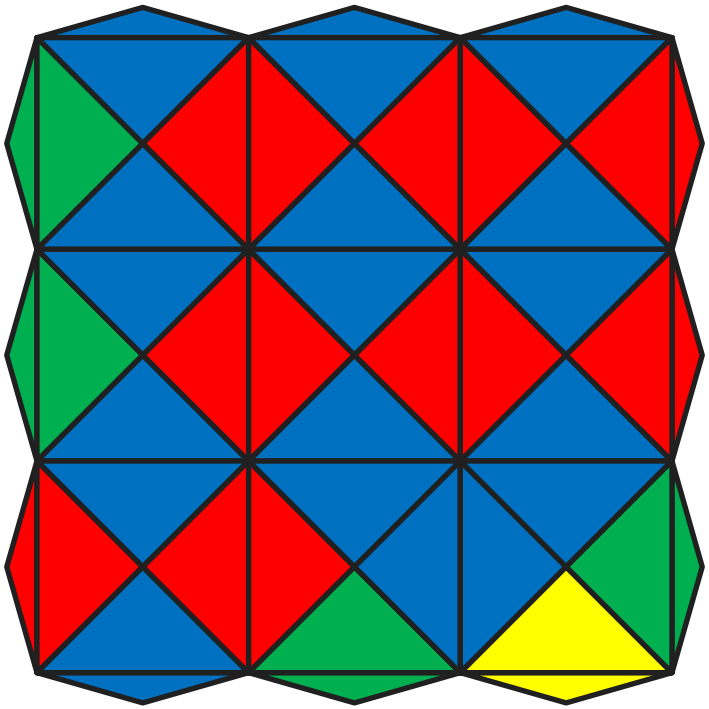
Problém Y je NP-úplný jestliže

1. $Y \in NP$
2. pro každé $X \in NP$ platí $X \leq_P Y$

Pokud problém Y splňuje druhou podmínku, říkáme, že je NP-těžký.

Problém ETERNITY





Omezení: čtvercové díly nemůžeme otáčet

Zakódování instance ETERNITY

- instanci problému chceme reprezentovat pomocí slova nad konečnou abecedou

$(K, m, n, O) \qquad \Sigma = \{0, 1, B\}$

- K … posloupnost čtveřic tvaru (b_H, b_L, b_D, b_R)
- $m \times n$ … rozměry čtvercové mřížky
- O … posloupnost $2m + 2n$ barev (okraje)

Čísla kódujeme binárně, znak B slouží jako oddělovač.

ETERNITY je NP-úplný

Podle definice potřebujeme dokázat:

1. *ETERNITY* je v NP
2. Každý problém v NP je na *ETERNITY* polynomiálně převeditelný (tj., že je *ETERNITY* NP-těžký)

Nechť M je libovolný nedeterministický TS s polynomiální časovou složitostí p, rozhodující jazyk L .

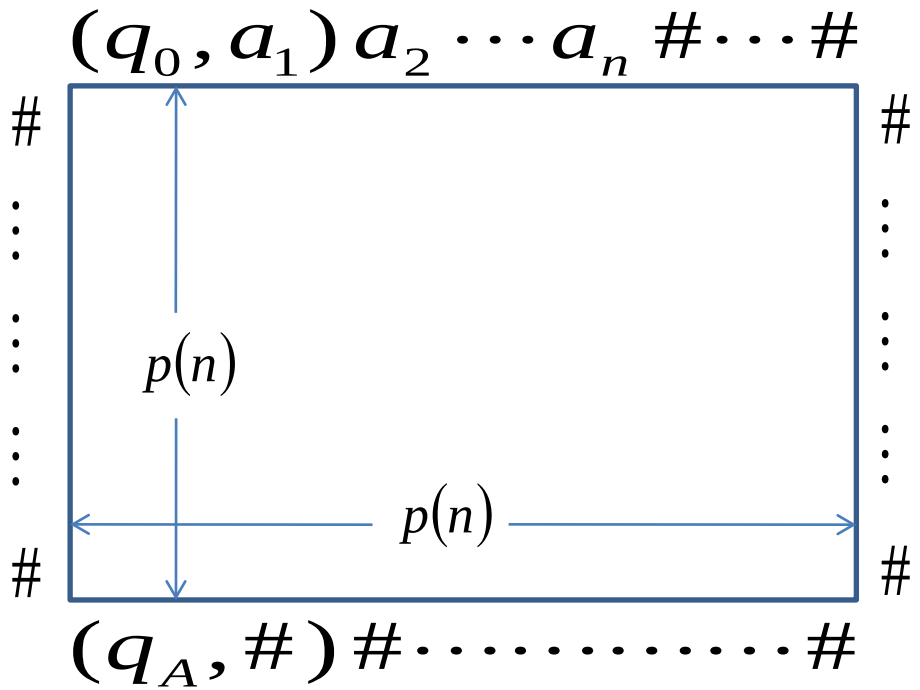
Cíl: ukázat, že $L \leq_P ETERNITY$

Pro vstup w stroje M navrhneme mřížku a čtverce tak, aby složení bylo možné pouze tehdy, pokud existuje přijímací výpočet.

ETERNITY je NP-úplný

Řádky kódují konfigurace stroje M

- horní okraj je počáteční konfigurace
- dolní okraj je koncová konfigurace

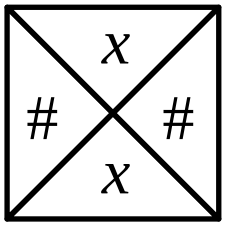


barvy:

$$\Gamma \cup Q \times \{\leftarrow, \rightarrow\} \cup Q \times \Gamma$$

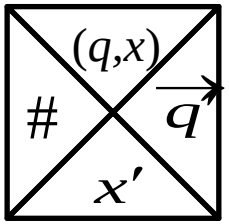
ETERNITY je NP-úplný

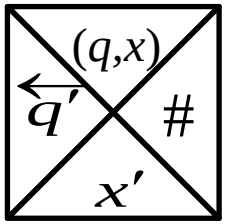
Typy čtverců:

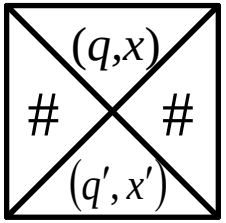


$x, x' \in \Gamma \qquad q, q' \in Q$

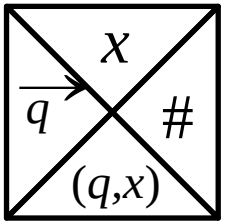
kopírování symbolu z horního řádku

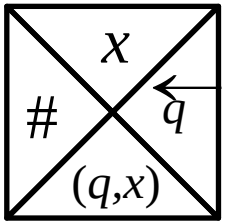






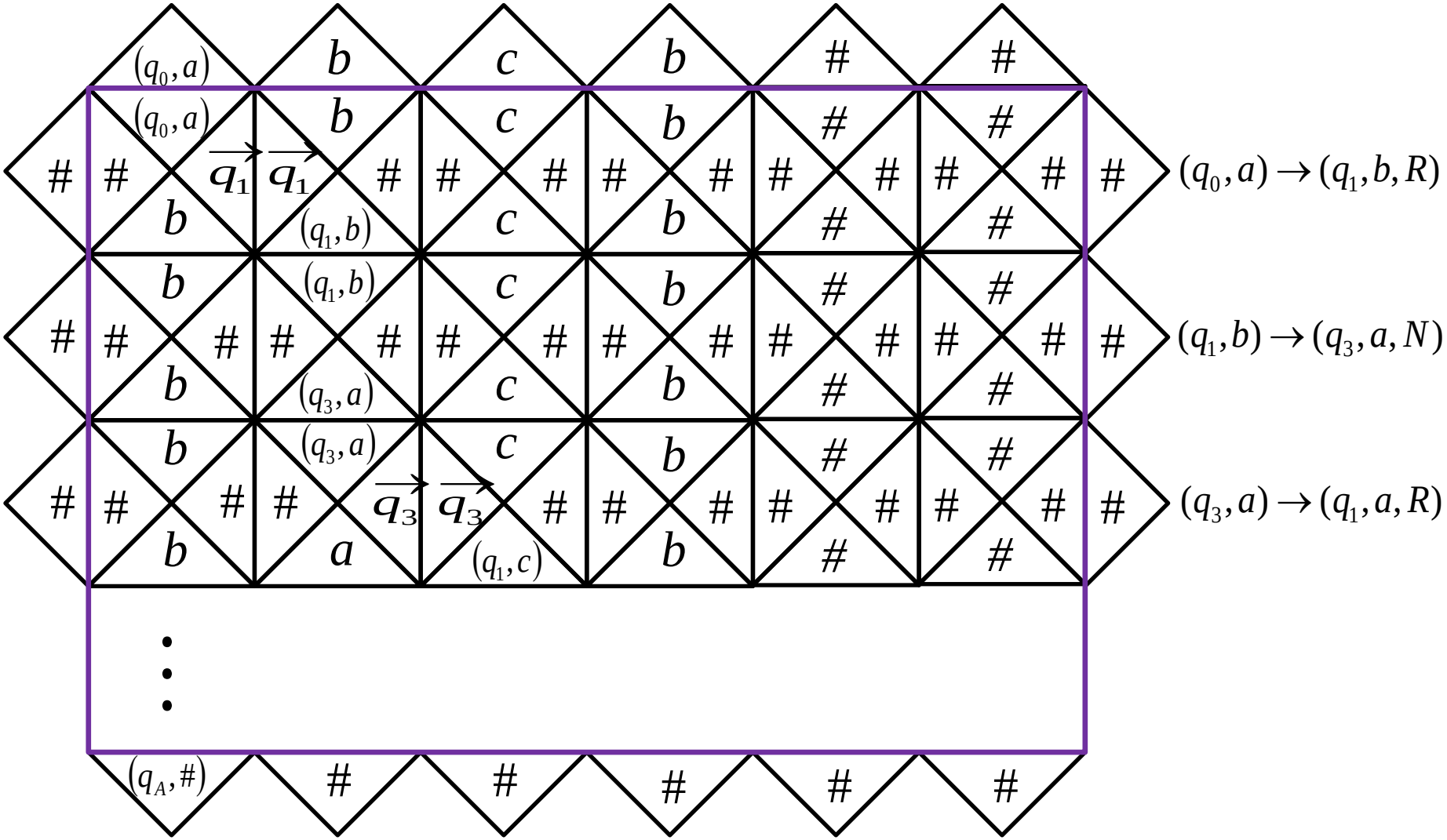
provedení instrukce s posunem hlavy R, L a N





zaznamenání posunu hlavy R a L do příslušného políčka

ETERNITY je NP-úplný



SAT je NP-úplný

1. *SAT* je v NP

$$\Phi(x_1, x_2, x_3) = (x_1 \vee \neg x_3 \vee x_2) \wedge (x_2 \vee x_1) \wedge (\neg x_2 \vee x_3)$$

… existuje ověřovací algoritmus

2. *SAT* je NP-těžký

$\leq_P$ je tranzitivní:

$$X \leq_P Y \wedge Y \leq_P Z \Rightarrow X \leq_P Z$$

tzn., stačí dokázat $ETERNITY \leq_P SAT$

SAT je NP-úplný

Instanci problému *ETERNITY* chceme transformovat na výrokovou formuli, která bude splnitelná právě tehdy, když je možné *ETERNITY* složit.

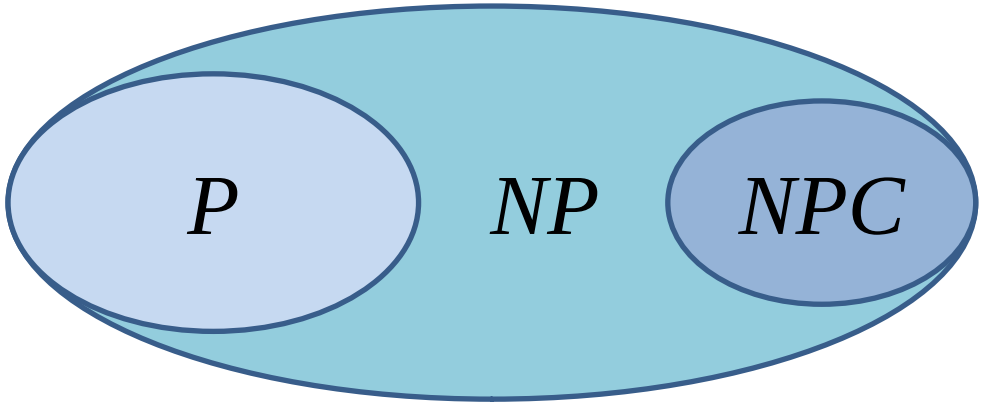
Boolovské proměnné $x_{i,j,k}$ … je k-tý čtverec na pozici (i, j) ?

Musí být splněny tyto podmínky:

1. Alespoň 1 čtverec na (i, j): $\bigwedge_{i,j} \left(\bigvee_k x_{i,j,k} \right)$
2. Ne víc než 1 čtverec na (i, j): $\bigwedge_{i,j} \bigwedge_{k \neq k'} \left(\neg x_{i,j,k} \vee \neg x_{i,j,k'} \right)$
3. Návaznost čtverců v řádcích: $\bigwedge_{i,j} \bigwedge_{k,k'} \left(\neg x_{i,j,k} \vee \neg x_{i,j+1,k'} \right)$
4. Návaznost čtverců ve sloupcích: $\bigwedge_{i,j} \bigwedge_{k,k'} \left(\neg x_{i,j,k} \vee \neg x_{i+1,j,k'} \right)$
5. Podmínky pro obvod: $\neg x_{1,j,k}$ $\neg x_{n,j,k}$ $\neg x_{i,1,k}$ $\neg x_{i,n,k}$

Shrnutí

NPC … třída všech NP-úplných problémů



je-li $P \neq NP$

Pokud by existovat polynomiální algoritmus pro jeden z NP-úplných problémů, bylo by možné řešit všechny NP-úplné problémy v polynomiálním čase.

Seznam významných NP-úplných problémů:
http://en.wikipedia.org/wiki/List_of_NP-complete_problems