Triedy zložitosti

Výpočtová zložitosť problému

- Časová výpočtová zložitosť $T_P(n)$ problému P rozsahu n je minimum z výpočtových zložitostí $T_A(n)$ všetkých algoritmov A, ktoré riešia problém P.
  - Napríklad: P je problém triedenia, jeho časová výpočtová zložitosť je minimum zo zložitostí všetkých triediacich algoritmov, napr. bubblesort $O(n^2)$, insertsort $O(n^2)$, selectsort $O(n^2)$, mergesort $O(n.\log n)$, quicksort $O(n.\log n)$, heapsort $O(n.\log n)$.
  - Výpočtová zložitosť problému triedenia je $O(n \log n)$ – existuje triediaci algoritmus so zložitosťou $O(n \log n)$
  - Výpočtová zložitosť problému triedenia je $\Omega(n \log n)$ – $n \log n$ je minimálny počet porovnaní, ktoré treba urobiť pri triedení porovnávacím algoritmom
- analogicky pamäťová výpočtová zložitosť problému

Funkcie zložitosti pre n =10, 100, 1000

	10	100	1000
O(1)	1	1	1
O(log n)	3,32	6,64	9,97
O(n)	10	100	1000
O(n log n)	33,22	664,39	9965,78
O(n^2)	100	10000	1000000
O(n^3)	1000	1000000	1000000000
O(2^n)	1024	1,26765E+30	1,0715E+301
O($n!$)	3628800	9,3326E+157	veľa
O(n^n)	10000000000	1E+200	veľa

Triedy zložitosti

- Trieda PTIME (P) je množina problémov, ktoré sa dajú riešiť deterministickým algoritmom v polynomiálnom čase $O(n^{\text{konšt}})$.
- Trieda NPTIME (NP) je množina problémov, ktoré sa dajú riešiť nedeterministickým algoritmom v polynomiálnom čase $O(n^{\text{konšt}})$.

Polynomiálna funkcia:
$p(n) = a_0 + a_1 n + a_2 n^2 + \ldots + a_c n^c = O(n^c)$

Čo je NP úloha?

- je taká úloha (problém), ktorá sa dá riešiť nedeterministicky v polynomiálnom čase
  - hĺbka výpočtového stromu závisí od vstupu polynomiálne
  - každá vetva stromu – nedeterministický výpočet – má polynomiálnu dĺžku
  - ak existuje riešenie problému, vieme overiť jeho správnosť deterministicky v polynomiálnom čase
  - na overenie neexistencie riešenia treba prehľadať celý výpočtový strom – v exponenciálnom čase

Príklady

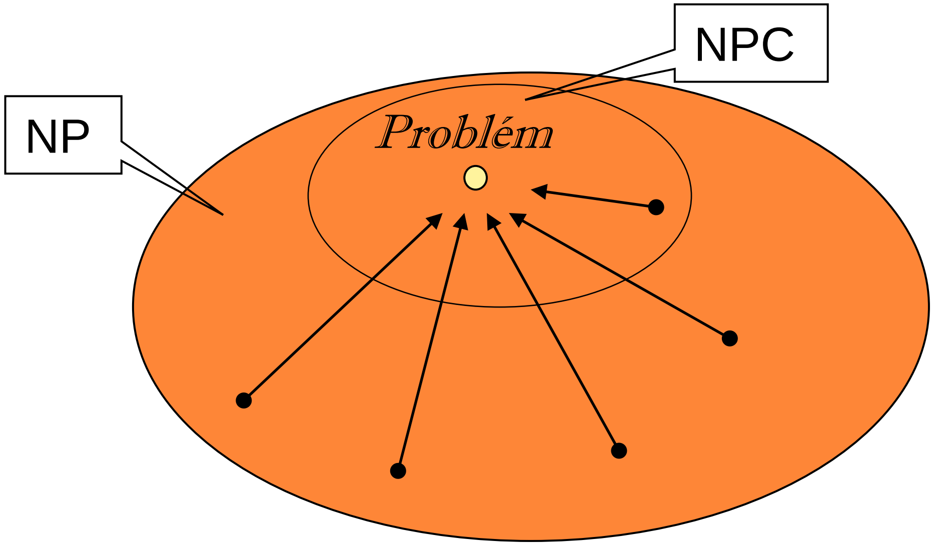
- Vynásobte dve n-ciferné čísla.
- Vyfarbite vrcholy grafu tromi farbami tak, aby vrcholy spojené hranou mali rôznu farbu.
- Prejdite šachovnicu koňom tak, aby na každé pole stúpil práve raz.
- Problém Hanojských veží. Preložte n kotúčov usporiadaných od najväčšieho dolu po najmenší hore z jednej tyče na druhú pomocou tretej tak, aby v žiadnom kroku výpočtu nebol väčší kotúč na menšom.

Príklady

- Vynásobte dve n-ciferné čísla. (P)
- Vyfarbite vrcholy grafu tromi farbami tak, aby vrcholy spojené hranou mali rôznu farbu. (NP)
- Prejdite šachovnicu koňom tak, aby na každé pole stúpil práve raz. (NP)
- Problém Hanojských veží. Preložte n kotúčov usporiadaných od najväčšieho dolu po najmenší hore z jednej tyče na druhú pomocou tretej tak, aby v žiadnom kroku výpočtu nebol väčší kotúč na menšom. (EXP)

Triedy zložitosti

- Trieda NPC (NP-complete) je množina *Problémov* (tzv. NP-úplných), pre ktoré platí:
  - *Problém* ∈NP
  - Pre všetky problémy *Problém'* ∈NP platí, že *Problém'* je redukovateľný v polynomiálnom čase na *Problém*



PRÍKLADY NPC PROBLÉMOV

- Zistiť, či je daná boolovská formula v konjunktívnej normálovej forme splniteľná.

$$(a \vee b \vee c) \wedge (a \vee b' \vee d) \wedge (c \vee d') \wedge b'$$

- Zistiť, či sa daný graf dá zafarbiť tromi farbami tak, aby žiadne dva vrcholy spojené hranou neboli zafarbené rovnakou farbou.
- Zistiť, či v danom grafe existuje k-prvková množina nezávislých vrcholov (navzájom nespojených žiadnou hranou).

Príklady NPC problémov

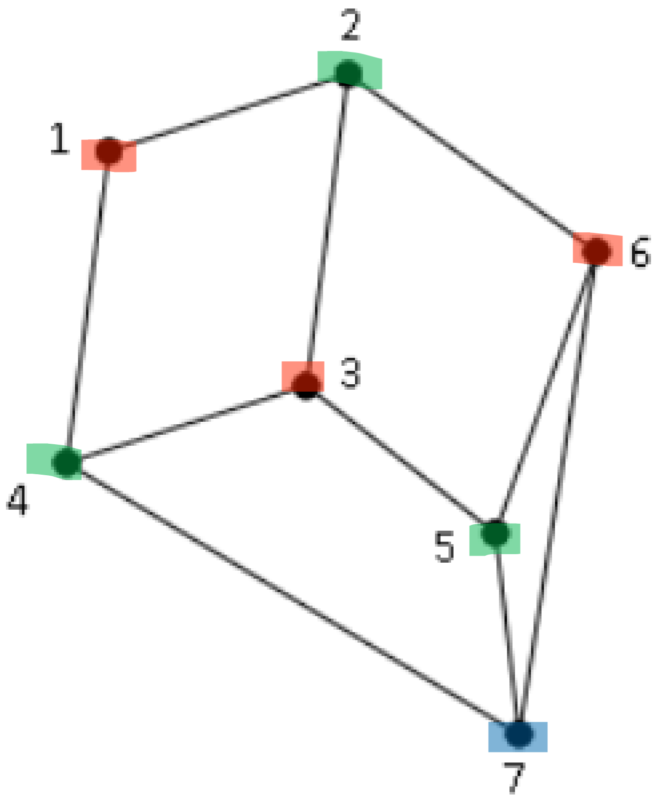
- Zistiť, či je daná boolovská formula v konjunktívnej normálovej forme splniteľná.

$$(a \vee b \vee c) \wedge (a \vee b' \vee d) \wedge (c \vee d') \wedge b'$$

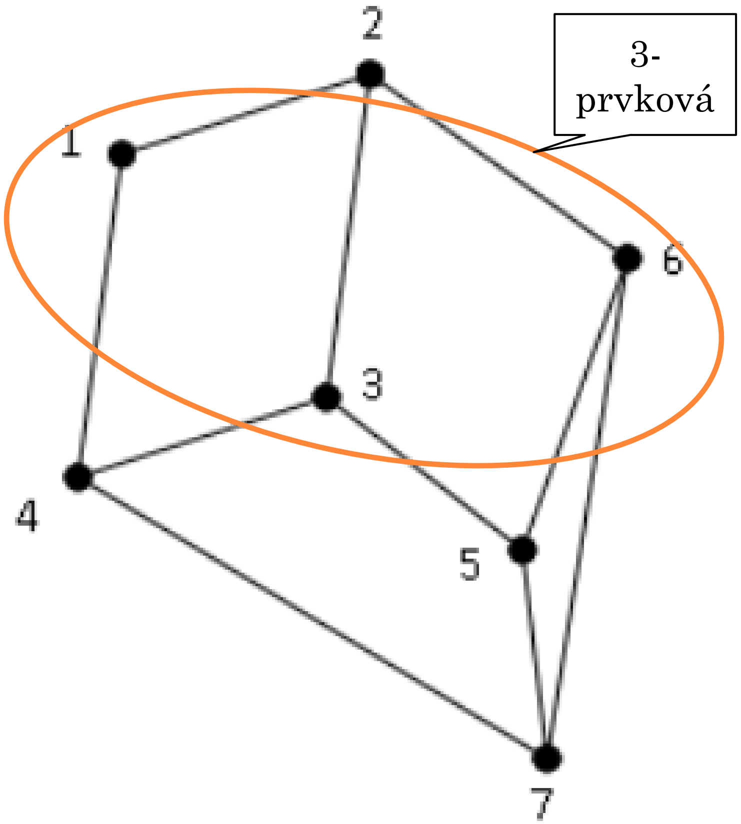
a	b	c	d	$a \vee b \vee c$	$a \vee b' \vee d$	$c \vee d'$	b'	
0	0	0	0	0	1	1	1	0
0	0	0	1	0	1	0	1	0
0	0	1	0	1	1	1	1	1
0	0	1	1	1	1	1	1	1
0	1	0	0	1	0	1	0	0
0	1	0	1	1	1	0	0	0
0	1	1	0	1	0	1	0	0
0	1	1	1	1	1	1	0	0
1	0	0	0	1	1	1	1	1
1	0	0	1	1	1	0	1	0
1	0	1	0	1	1	1	1	1
1	0	1	1	1	1	1	1	1
1	1	0	0	1	1	1	0	0
1	1	0	1	1	1	0	0	0
1	1	1	0	1	1	1	0	0
1	1	1	1	1	1	1	0	0

Príklady NPC problémov

- Zistiť, či sa daný graf dá zafarbiť tromi farbami tak, aby žiadne dva vrcholy spojené hranou neboli zafarbené rovnakou farbou.



PRÍKLADY NPC PROBLÉMOV



- Zistiť, či v danom grafe existuje *k*-prvková množina nezávislých vrcholov (navzájom nespojených žiadnou hranou).

ľudia = vrcholy grafu

vzťah nepoznajú sa = hrana

skupina k ľudí, ktorí sa všetci navzájom poznajú = k-prvková množina nezávislých vrcholov

PRÍKLADY NPC PROBLÉMOV

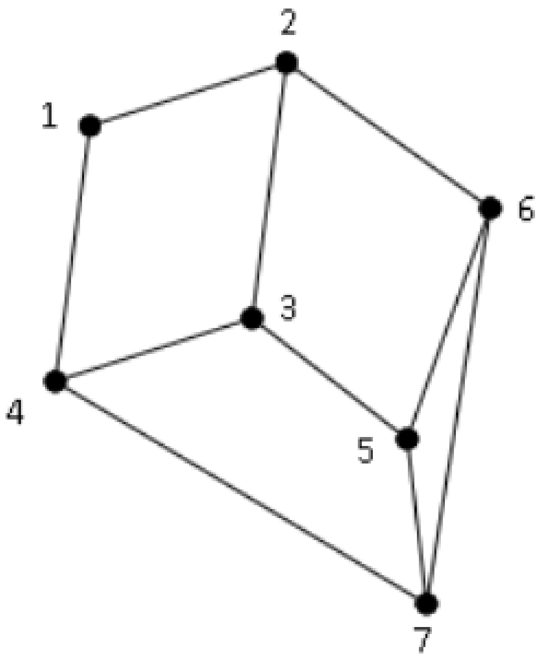
- Existuje *4*-prvková množina nezávislých vrcholov (navzájom nespojených žiadnou hranou)?

  35 možností:

1	2	3	4
1	2	3	5
1	2	3	6
1	2	3	7
1	2	4	5
1	2	4	6
1	2	4	7
1	2	5	6
1	2	5	7
1	2	6	7

1	3	4	5
1	3	4	6
1	3	4	7
1	3	5	6
1	3	5	7
1	3	6	7
1	4	5	6
1	4	5	7
1	4	6	7
1	5	6	7

2	3	4	5
2	3	4	6
2	3	4	7
2	3	5	6
2	3	5	7
2	3	6	7
2	4	5	6
2	4	5	7
2	4	6	7
2	5	6	7
3	4	5	6
3	4	5	7
3	4	6	7
3	5	6	7
4	5	6	7



Príklad polynomiálnej redukcie:

Problém 3-zafarbiteľnosti grafu *polynomiálne redukujeme* na problém splniteľnosti boolovskej formuly v CNF:

- Ku grafu $G=(V,E)$ vytvoríme boolovskú formulu b v CNF takú, že b je splniteľná práve vtedy, keď G je 3-zafarbiteľný.
- Premenné:
  - B_i = true, keď i-ty vrchol v G má bielu farbu (pre každé $i \in V$)
  - M_i = true, keď i-ty vrchol v G má modrú farbu (pre každé $i \in V$)
  - C_i = true, keď i-ty vrchol v G má červenú farbu (pre každé $i \in V$)

- Časť formuly vyjadrujúca, že každý vrchol má aspoň jednu farbu:

$$\prod_{i \in V} (B_i \vee M_i \vee C_i) \qquad \text{dĺžka} = O(n)$$

- Časť formuly vyjadrujúca, že každý vrchol má najviac jednu farbu:

$$\prod_{i \in V} (\neg B_i \vee \neg M_i) \wedge (\neg B_i \vee \neg C_i) \wedge (\neg C_i \vee \neg M_i) \qquad O(n)$$

- Časť formuly vyjadrujúca, že dva susedné vrcholy nemajú rovnakú farbu:

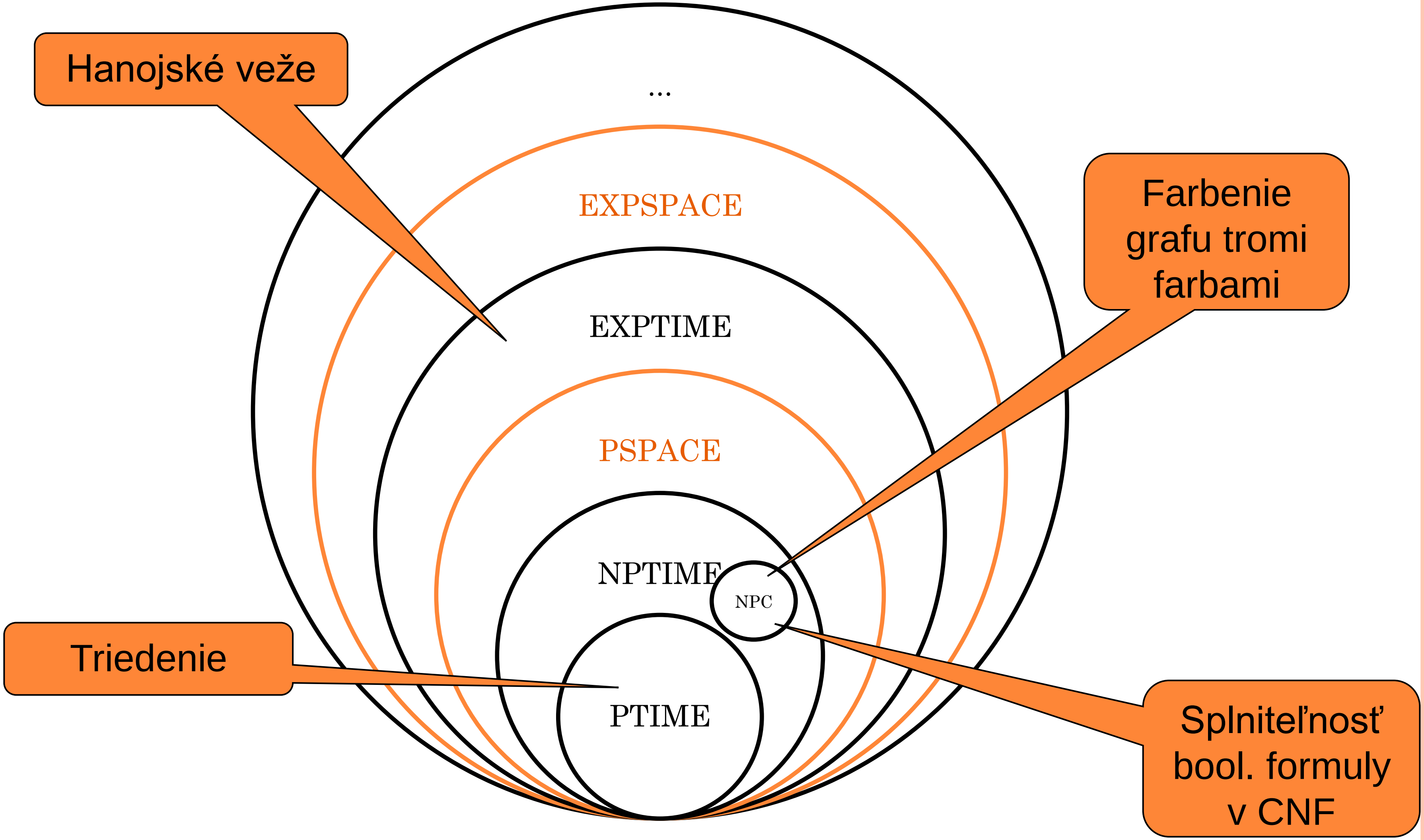
$$\prod_{(i,j) \in E} (\neg B_i \vee \neg B_j) \wedge (\neg C_i \vee \neg C_j) \wedge (\neg M_i \vee \neg M_j) \qquad O(n^2)$$

- b dostaneme ako logický súčin všetkých troch častí formuly
- b je splniteľná práve vtedy, keď G je zafarbiteľný tromi farbami
- Zložitosť redukcie je $O(n^2)$
- Problém 3-zafarbiteľnosti je polynomiálne redukovateľný na problém splniteľnosti boolovskej formuly v CNF.
- Podobne sa dajú redukovať aj *všetky* ostatné NP problémy na problém splniteľnosti boolovskej formuly v CNF (dá sa formálne dokázať).

Vzťah P a NP

- $P \subset NP$
- Otvorený problém:
  - Je P=NP ?
  - Dá sa nejaký problém z NPC vyriešiť v polynomiálnom čase aj determinsticky?
  - Predpokladá sa, že nie.

Triedy zložitosti



ALGORITMICKY NERIEŠITEĽNÉ PROBLÉMY

- algoritmus je návod, ktorý pre každý vstup generuje *konečný* výpočet
- ak návod generuje pre nejaký vstup nekonečný výpočet, tak to nie je algoritmus
- algoritmicky neriešiteľné sú také problémy, na riešenie ktorých neexistuje algoritmus, t.j. nedajú sa vyriešiť v konečnom čase pre každý vstup
- príklad: **Problém zastavenia Turingovho stroja**
  - *Je daný Turingov stroj a vstupné slovo. Úloha je zistiť, či sa výpočet Turingovho stroja na danom vstupe zastaví, alebo nie.*

ALGORITMICKY NERIEŠITEĽNÉ PROBLÉMY

- príklad: **Problém zastavenia Turingovho stroja:**

  Je daný Turingov stroj a vstupné slovo. Úloha je zistiť (algoritmicky), či sa výpočet Turingovho stroja na danom vstupe zastaví, alebo nie.
- riešenie: simulovanie výpočtu daného Turingovho stroja na danom slove
  - konečný výpočet v prípade kladnej odpovede
  - nekonečný výpočet v prípade zápornej odpovede

Rekurzívne vyčísliteľný a nevyčísliteľný jazyk

- **Rekurzívny jazyk** – existuje Turingov stroj, ktorý ho rozpoznáva a zastaví sa pre každý vstup (teda príslušnosť slova do jazyka sa dá rozhodovať algoritmicky – v konečnom čase).
- **Rekurzívne vyčísliteľný jazyk** – existuje Turingov stroj, ktorý ho rozpoznáva (nemusí sa zastaviť pre každý vstup).
- Ak L je rekurzívny, tak aj $\sum^*$-L (doplnok L) je rekurzívny jazyk.

  L je rekurzívny $\Rightarrow$ existuje TS M, ktorý ho rozpoznáva a zastaví sa pre všetky vstupy.

  Zostrojíme TS M', ktorého koncové stavy sú nie koncové stavy pôvodného TS M, a teda rozpoznáva doplnok k jazyku L a tiež sa zastaví pre všetky vstupy.

Rekurzívne nevyčísliteľný jazyk

- Existuje jazyk, ktorý je rekurzívne vyčísliteľný, ale nie je rekurzívny.

  $L_1 = \{x_i, M_i$ akceptuje $x_i\}$

  L_1 je rekurzívne vyčísliteľný, ale nie je rekurzívny.

  Dôkaz sporom:

  Nech TS M, ktorý rozpoznáva L_1, simuluje činnosť Turingovho stroja M_i na slove x_i.

  Predpokladajme, že L_1 je rekurzívny.

  Potom jeho doplnok $L_2 = \Sigma^* - L_1 = \{x_i, M_i$ neakceptuje $x_i\}$ musí byť tiež rekurzívny.

Rekurzívne nevyčísliteľný jazyk

	x_1	x_2	x_3	x_4	…	x_k	…
M_1	A	A	N	A		N	
M_2	N	A	A	A		A	
M_3	A	N	N	N		N	
M_4	A	A	N	A		A	
…							
M_k	N	N	A	N		?	
…							

- neexistuje Turingov stroj M_k, ktorý by rozpoznával jazyk $L_2 = \{x_i, M_i \text{ neakceptuje } x_i\}$
- teda L_2 nie je rekurzívne vyčísliteľný, a tobôž nie rekurzívny jazyk $\Rightarrow$ **jeho doplnok L_1 nemôže byť rekurzívny**

Problém zastavenia Turingovho stroja

- Problém zastavenia Turingovho stroja je nerozhodnuteľný, t.j. neexistuje algoritmus, ktorý pre daný TS a dané vstupné slovo rozhodne v konečnom čase, či sa TS na danom slove zastaví alebo nie.
- Dokaz sporom:

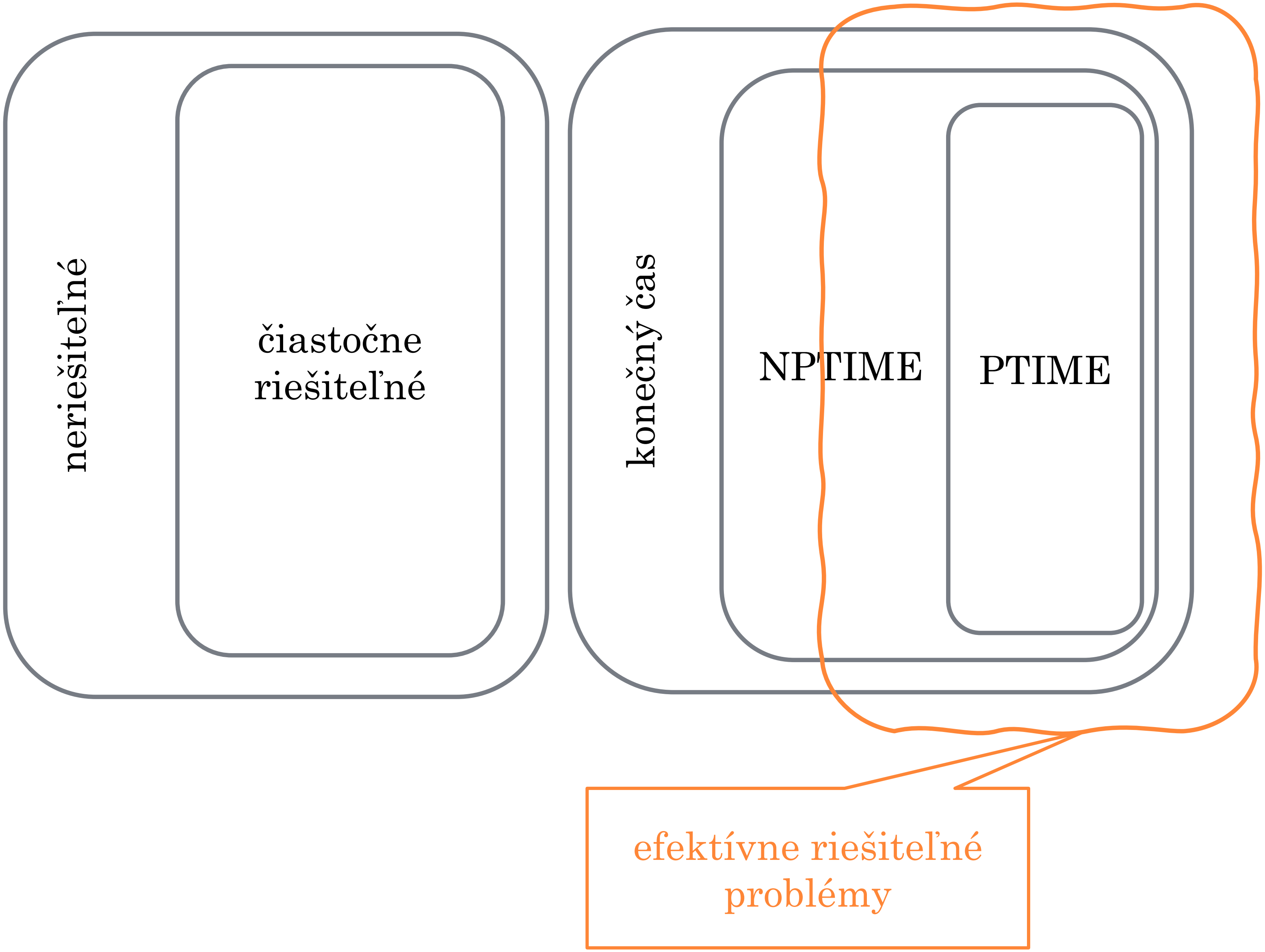
  Predpokladajme, že existuje TS, ktorý rozhoduje problém zastavenia $\Rightarrow$ existuje algoritmus na rozhodovanie príslušnosti slova x jazyka $L_2 = \{x_i,\ T_i \text{ neakceptuje } x_i\}$:

  1. pre vstupné slovo x určí jeho poradie v zozname všetkých slov a kód príslušného Turingovho stroja M
  2. zistí, či sa M zastaví na x
  3. ak sa zastaví, tak simuluje činnosť M na x, výsledok neguje
  4. ak sa nezastaví, tak slovo x akceptuje

  Z existencie konečného algoritmu vyplýva, že L_2 je rekurzívne vyčísliteľný jazyk $\Rightarrow$ spor

Hierarchia algoritmicky neriešiteľných problémov

- Problém zastavenia Turingovho stroja: Zistiť, či sa výpočet daného Turingovho stroja na danom vstupnom slove zastaví alebo nie.
  - Problém zastavenia Turingovho stroja je nerozhodnuteľný (*čiastočne rozhodnuteľný*, výpočet je nekonečný v prípade zápornej odpovede).
- Zistiť, či sa výpočet daného Turingovho stroja zastaví pre každý vstup.
  - Problém je úplne nerozhodnuteľný (výpočet je nekonečný v prípade kladnej aj zápornej odpovede).


neriešiteľné
čiastočne riešiteľné
konečný čas
NPTIME
PTIME
efektívne riešiteľné problémy