Složitost a NP-úplnost

RNDr. Ondřej Čepek, Ph.D.

Do formátu TEX převedl Ladislav Strojil
Připomínky, dotazy, opravy na emailu: Ladislav@Strojil.cz
Verze 1.1.1
Nejnovější verze k nalezení vždy na http://ladislav.strojil.cz/np.php

Obsah

1 Základní definice

1.1 Výpočetní model

Definice 1 (Turingův stroj).

Deterministický Turingův stroj (DTS) M s k-páskami, kde k je konstanta, je pětice

$$M = (Q, \Sigma, \delta, q_0, F) \tag{1}$$

$Q =$ *konečná množina stavů řídící jednotky*
$\Sigma =$ *konečná pásková abeceda*
$\delta : Q \times \Sigma^k \mapsto Q \times \Sigma^{k-1} \times \{L, N, R\}^k$ *je přechodová funkce (částečná)*
$q_0 \in Q =$ *počáteční stav*
$F \subseteq Q =$ *množina přijímajících stavů*

Následující definice popisuje Turingovy stroje trochu podrobněji a uvádí vysvětlení základních pojmů používaných ve vztahu k tomuto modelu.

Definice 2.

- *Všech k pásek je jednosměrně (potenciálně) nekonečných.*
- *Jedna z pásek je vstupní a pouze ke čtení.*
- *Na všech páskách se lze pohybovat oběma směry (off-line model), v on-line modelu se lze na vstupní pásce pohybovat pouze doprava.*
- *Konfigurace Turingova stroje je určena*
  1) *stavem řídící jednotky*
  2) *pozicí hlav na jednotlivých páskách*
  3) *slovy na jednotlivých páskách (obsahem pásek)*
- *Počáteční konfigurace Turingova stroje*
  1) q_0
  2) *všechny hlavy zcela vlevo*
  3) *na vstupní pásce vstupní slovo, ostatní pásky prázdné*
- *Displej Turingova stroje je stav řídící jednotky a obsah políček pod hlavami.*
- *Krok Turingova stroje je použití přechodové funkce na displej, přepsání políček poh hlavami a případný posun hlav a změna stavu.*
- *Výpočet Turingova stroje je posloupnost kroků začínající v počáteční konfiguraci, která končí zastavením stroje nebo je nekonečná.*
- *Zastavení Turingova stroje je situace, kdy pro daný displej není přechodová funkce definována (předpokládá se, že to platí pro všechna $q \in F$).*
- *Přijímající výpočet Turingova stroje je výpočet končící zastavením v přijímajícím stavu (vstupní slovo je přijato).*

- *Odmítající výpočet Turingova stroje je výpočet končící zastavením v jiném než přijímajícím stavu nebo nekonečný.*
- *Přijímaný jazyk Turingova stroje M značíme $L(M)$ a definujeme jako $\{w \in \Sigma^* | w$ w je přijímáno $M\}$.*

Definice 3 (Transducer). *Turingův stroj, který má navíc výstupní pásku, na kterou lze pouze psát (znak pod hlavou neovlivňuje přechodovou funkci) a na které se hlava může pohybovat pouze vpravo, nazýváme* transducer*. (Stroj bez této pásky se nazývá* acceptor*).*

Existují různé varianty Turingových strojů, o kterých lze ukázat, že mají stejnou výpočetní sílu, tedy že přijímají stejnou třídu jazyků (rekurzivně spočetné množiny). Jsou to například:

- on-line Turingovy stroje;
- jednopáskové Turingovy stroje;
- Turingovy stroje s oboustranně nekonečnou páskou;
- Turingovy stroje s více hlavami na jedné pásce.

Důvody, proč je výhodné používat Turingovy stroje jako výpočetní model jsou zřejmé. Předně je to jasně definovaný koncept *časové složitosti* výpočtu (běhu algoritmu) jako počet kroků daného Turingova stroje nad daným vstupem. Analogicky jasně definovaný koncept *prostorové složitosti* výpočtu (běhu algoritmu) jako počet použitých buněk na pracovních páskách stroje.

Hlavní nevýhodou je fakt, že tento model je příliš elementární a nehodí se ke "skutečnému programování".

1.2 Kódování Turingových strojů

1) Očíslujeme stavy $q \in Q$ a symboly $s \in \Sigma$ binárními čísly.
2) Popíšeme přechodovou funkci v abecedě $A = \{0, 1, \delta, (,), =, L, N, R, ,\}$ zřetězením zápisů typu $\delta(q, x_1, x_2, \ldots, x_k) = (q', y_1, \ldots, y_k, Z)$.
3) Přepíšeme zápis z abecedy A do abecedy $\{0, 1\}$

Potom při pevně zvoleném kódování každý Turingův stroj jednoznačně odpovídá přirozenému číslu.

Definice 4 (Gödelovo číslo). *Kód Turingova stroje M budeme nazývat jeho Gödelovým číslem.*

1.3 Univerzální Turingův stroj

Definice 5 (Univerzální Turingův stroj). *Univerzální Turingův stroj je Turingův stroj, který má jako vstup Gödelovo číslo nějakého Turingova stroje M a pro vstupní slovo w pro M pracuje tak, že simuluje práci M nad w.*

1.4 Zpracování výjimek

Pomocí přidání nových stavů řídící jednotky lze jednoduše ošetřit libovolnou pevně danou konečnou množinu vstupů.

Provedeme to tak, že zkontruujeme konečný automat (strom) rozpoznávající danou konečnou množinu slov (konečný automat odpovídá Turingově stroji bez pásek).

Turingův stroj potom napřed simuluje chod daného konečného automatu. V případě, že je na vstupu jedno z rozpoznávaných slov, Turingův stroj se zastaví.

Pokud ne, odjede hlavou na vstupní pásce na začátek a spustí původní Turingův stroj.

1.5 Nedeterminismus

Od deterministického Turingova stroje se jeho nedeterministická varianta liší pouze definicí přechodové funkce a stanovením odlišných podmínek, za kterých je slovo přijato.

Definice 6 (Nedeterministický turingův stroj).
Definujme stroj jako v definici deterministického Turingova stroje s následujícími rozdíly:

- $\delta : Q \times \Sigma^k \mapsto \mathcal{P}(Q \times \Sigma^{k-1} \times \{L, N, R\}^k)$ *je přechodová funkce (částečná)*
- *Slovo w je přijímáno, jestliže existuje alespoň jeden přijímající výpočet.*

Pro zjednodušení můžeme předpokládat binární větvení výpočetu. Snadno se dá ukázat, že se jedná o výpočetně ekvivalentní model.

1.6 Prostorová složitost

Definice 7.
Nechť M je deterministický Turingův stroj, pro který platí, že na každém vstupu délky n použije při výpočtu nejvýše $S(n)$ buněk na pracovních páskách. *Potom říkáme, že M má prostorovou složitost $S(n)$ a že jazyk $L(M)$ má prostorovou složitost $S(n)$.*

1.7 Časová složitost

Definice 8.
Nechť M je deterministický Turingův stroj, pro který platí, že na každém vstupu délky n provede při výpočtu nejvýše $T(n)$ kroků, než se zastaví. Potom říkáme, že M má časovou složitost $T(n)$ a že jazyk $L(M)$ má časovou složitost $T(n)$.

1.8 Nedeterministická prostorová a časová složitost

Definice jsou stejné jako u deterministických Turingových strojů s tím, že nyní požadujeme, aby $\forall w \in \Sigma^*, |w| = n$ všechny možné výpočty použily nejvýše $S(n)$ políček a skončily nejdéle za $T(n)$ kroků.

2 Základní třídy složitosti

$$\begin{aligned}
DSPACE(S(n)) &= \text{třída jazyků deterministické prostorové složitosti } S(n)\\
NSPACE(S(n)) &= \text{třída jazyků nedeterministické prostorové složitosti } S(n)\\
DTIME(T(n)) &= \text{třída jazyků deterministické časové složitosti } T(n)\\
NTIME(S(n)) &= \text{třída jazyků nedeterministické časové složitosti } T(n)
\end{aligned}$$

Tvrzení 1 (Triviální vztahy).

$$\forall\ S(n) : DSPACE(S(n)) \subseteq NSPACE(S(n)) \tag{2}$$
$$\forall\ T(n) : DTIME(T(n)) \subseteq NTIME(T(n)) \tag{3}$$

$$\begin{aligned}
\forall\ S_1(n) \leq S_2(n) \ \Rightarrow\ & DSPACE(S_1(n)) \subseteq DSPACE(S_2(n)) && (4)\\
& NSPACE(S_1(n)) \subseteq NSPACE(S_2(n))\\
\forall\ T_1(n) \leq T_2(n) \ \Rightarrow\ & DTIME(T_1(n)) \subseteq DTIME(T_2(n)) && (5)\\
& NTIME(T_1(n)) \subseteq NTIME(T_2(n))
\end{aligned}$$

Důkaz. Zřejmé. □

3 Prostorová a časová komprese

3.1 Prostorová komprese

Věta 1 (O lineární prostorové kompresi).
Nechť L je jazyk přijímaný k-páskovým deterministickým Turingovým strojem M s prostorovou složitostí S(n). Potom pro $\forall r \in \mathbf{N}^+$ existuje k-páskový deterministický Turingův stroj M' s prostorovou složitostí $\lceil \frac{1}{r} S(n) \rceil$, který přijímá jazyk L.

Důkaz. Každý znak abecedy stroje M' bude kódovat jednu r-tici znaků abecedy původního stroje. $|\Sigma_{M'}|$ (velikost abecedy M') bude rovno $|\Sigma_M|^r$ a vzhledem ke konečnosti Σ_M bude také konečná.

Jeden krok stroje M bude simulován jedním krokem stroje M'. Pozici původního stroje "uvnitř" políčka si stroj M' bude pamatovat ve stavech.

Nejprve nechť $k = 1$.

Je-li $Q_M = \{q_0, q_1, \ldots, q_s\}$ množina stavů stroje M, potom definujeme $Q_{M'} = \{q_0^1, q_0^2, \ldots, q_0^r, \ldots, q_s^1, q_s^2, \ldots, q_s^r\}$.

Přechodová funkce $\delta_{M'}$ bude simulovat pohyb hlavy změnou horního indexu stavu, v případě, že by index klesl na nulu nebo byl větší než r, dojde k vlastnímu posunu hlavy (stroje M').

Všechna pravidla tvaru $\delta_M(q_i, a) = (q_j, b, N)$ nahradíme množinou pravidel $\delta_{M'}(q_i^l, x^l) = (q_j^l, y^l, N)$, $\forall\ 1 \leq l \leq r$, $\forall$ dvojice $x, y \in \Sigma_{M'}$, kde x kóduje $(\ldots, a, \ldots)$ a y kóduje $(\ldots, b, \ldots)$ (y se liší od x pouze na l-té pozici).

Všechna pravidla tvaru $\delta_M(q_i, a) = (q_j, b, R)$ nahradíme množinou pravidel $\delta_{M'}(q_i^l, x^l) = (q_j^{l+1}, y^l, N)$, $\forall\ 1 \leq l < r$ a $\delta_{M'}(q_i^r, x^l) = (q_j^1, y^r, R)$.

Analogicky pro pohyb doleva. $\delta_M(q_i, a) = (q_j, b, L)$ nahradíme množinou pravidel $\delta_{M'}(q_i^l, x^l) = (q_j^{l-1}, y^l, N)$, $\forall\ 1 < l \leq r$ a $\delta_{M'}(q_i^1, x^l) = (q_j^r, y^1, L)$.

Nyní pro obecné k.

$Q_{M'} = \{q_i^p | \ 0 \leq i \leq s, \forall p \in \Sigma_M^r\}$
Proměnná p zde probíhá přes všechny r-tice znaků abecedy Σ_M.
Analogicky jako pro případ $k = 1$ se sestrojí přechodová funkce $\delta_{M'}$. □

Důsledek 1.
$\forall\, r \in \mathbf{N}^+ : DSPACE(S(n)) = DSPACE(\lceil \frac{S(n)}{r} \rceil)$

Důkaz. Plyne přímo z věty 1. □

Věta 2.
$\forall\, r \in \mathbf{N}^+ : NSPACE(S(n)) = NSPACE(\lceil \frac{S(n)}{r} \rceil)$

Důkaz. Zobecněním důkazu věty 1 pro nedeterministický případ. □

3.1.1 Redukce počtu pásek

Věta 3 (O redukci počtu pásek pro prostorovou složitost).
Nechť L je jazyk přijímaný k-páskovým deterministickým Turingovým strojem s prostorovou složitostí $S(n)$. Potom existuje deterministický Turingův stroj M' s jednou pracovní páskou a prostorovou složitostí $S(n)$ přijímající jazyk L.

Důkaz. Myšlenka důkazu: i-té políčko na pásce stroje M' bude kódovat obsah k políček (z každé pásky stroje M jedno). Dále bude v políčku zakódována kromě obsahu původních pásek i informace, které z hlav se nachází nad tímto políčkem.

Nová abeceda bude definována následovně: $\Sigma_{M'} = \{x_0, \ldots, x_{2^k-1} | \forall x \in \Sigma_M^k\}$

Indexy si můžeme představit v jejich binárním zápisu jako informaci o tom, které hlavy simulovaného stroje M se nachází nad daným políčkem.

Jeden krok stroje M je simulován v $O(S(n))$ krocích stroje M' tak, že hlava projde celou pásku, najde pozice hlav a odsimuluje krok M.

Vzhledem k tomu, že při simulaci nám nezáleží na časové složitosti, lze pro jednoduchost uvažovat výpočet, který má k fází a v každé fázi se odsimuluje práce M na jediné pásce. Na začátku fáze a po jejím ukončení bude hlava M' na začátku pásky.

Jak bude probíhat samotná simulace? Každou instrukci původního stroje nahradíme posloupností instrukcí M', kde stavy řídící jednotky M' obsahují informaci o displeji stroje M. Tato posloupnost začíná s počátečním displejem a končí s koncovým displejem simulované instrukce.

1) Instrukce M očíslujeme a tato čísla zakódujeme do stavů M'. Tím zajistíme, že po začátku provádění posloupnosti instrukcí bude tato posloupnost dokončena a nebudou se míchat různé posloupnosti mezi sebou. Dále si budeme ve stavu pamatovat číslo aktuální fáze (počet fází je omezen počtem pásek, to znamená konečný a pevně daný). Výsledný počet stavů je tedy také konečný.

2) instrukci $\delta_M(q_i, a_1, \ldots, a_k) = (q_j, b_1, \ldots, b_k, z_1, \ldots, z_k)$, která má číslo l, nahradíme posloupností stavů: $(l, q_i, a_1, \ldots, a_k, 1^R)$. Stroj M' začíná v tomto stavu na první pozici své pracovní pásky. Nyní M' projde pásku směrem doprava a hledá políčko, ve kterém je označena pozice první hlavy.

   - Když x nekóduje pozici první hlavy, $\delta_{M'}((l, q_i, a_1, \ldots, a_k, 1^R), x) = ((l, q_i, a_1, \ldots, a_k, 1^R), x, R)$.

- Když x kóduje pozici první hlavy, rozlišíme následující případy podle směru, kterým by se první hlava posunula při práci stroje M.
  a) $z_1 = N$ znak x kódující $(a_1, a_2, \dots, a_k)$ se přepíše na znak y kódující $(b_1, a_2, \dots, a_k)$ se stejnou množinou hlav. Zároveň se provede aktualizace displeje a 1^R se změní na 1^L.
  $\delta_{M'}((l, q_i, a_1, \dots, a_k, 1^R), x) = ((l, q_j, b_1, a_2, \dots, a_k, 1^L), y, N)$
  b) $z_1 = R$ znak x kódující $(a_1, a_2, \dots, a_k)$ se přepíše na znak y kódující $(b_1, a_2, \dots, a_k)$ se stejnou množinou hlav, s vyjímkou první hlavy.
  $\delta_{M'}((l, q_i, a_1, \dots, a_k, 1^R), x) = ((l, q_j, b_1, a_2, \dots, a_k, 1^N), y, R)$
  $\delta_{M'}((l, q_j, a_1, \dots, a_k, 1^N), z) = ((l, q_j, b_1, a_2, \dots, a_k, 1^L), z', N)$
  Zde z' kóduje stejnou k-tici jako z, ale navíc kóduje přítomnost první hlavy.
  c) $z_1 = L$. Zcela analogicky s případem R.
- Po a,b,c následuje odjezd hlavy M' na začátek pásky a přepnutí do stavu $(l, q_j, b_1, a_2, \dots, a_k, 2^R)$.
- Simulace dále pokračuje další fází.

□

Věta 4 (Nedeterministická verze).
Věta 3 platí ve stejném znění i pro nedeterministické Turingovy stroje.

Důkaz. Simulace uvedená v důkazu věty 3 je použitelná beze změn i pro případ nedeterministického Turingova stroje. □

3.2 Časová komprese

Lemma 1 (O lineárním zrychlení).
Nechť L je jazyk přijímaný k-páskovým deterministickým Turingovým strojem M s časovou složitostí $T(n)$. Dále nechť platí $k \geq 2$ a nechť r je celé číslo, $r > 0$.

Potom existuje deterministický Turingův stroj M' takový, že každý vstup w délky n přijímá právě, když jej přijímá M. Pracuje-li M nad w t kroků, pracuje M' nad w v čase $n + \lceil n/r \rceil + 6\lceil t/r \rceil$.

Poznámka: Předpokládáme, že na vstupní pásku je možno zapisovat.

Důkaz. V první fázi zkopíruje M' obsah vstupní pásky na pracovní pásku (ta existuje, neboť $k \geq 2$), současně s kopírováním M' provede překódování vstupu tak, že v jednom políčku cílové pásky je kódováno r políček vstupní pásky, kde r je libovolné celé číslo > 0. Po provedení se vrátí na začátek pásky. Nadále bude tuto pásku používat jako vstupní.

Simulace M: Konkrétní "jemnou" pozici vrámci jedné buňky nového stroje si budeme pamatovat ve stavech stroje M'.

- Stroj provede sekvenci pohybů vlevo, vpravo, vpravo, vlevo (4 kroky), pričemž si ve stavech zapamatuje obsah sousedních políček. Tím stroj získal informaci o políčku pod hlavou, políčku vpravo od hlavy a políčku vlevo od hlavy (všimněte si, že se jedná o $3 \cdot r$ políček stroje M).

- V další fázi M' zjistí stav těchto $3r$ políček stroje M po r krocích. Uvědomte si, že během r kroků stroj M nemůže tato políčka opustit, neboť se jeho hlava nacházela v prostřední třetině tohoto úseku. Navíc změna stavu tohoto úseku obsahuje jenom konečnou informaci a tedy může být zakódována do přechodové funkce. Tedy celá tato fáze odpovídá 1 kroku stroje M'. Není potřeba jej ale započítávat do délky výpočtu, neboť získání potřebné informace lze provést již v posledním kroku předchozí fáze (návrat hlavy nad prostřední políčko bude spojen s přechodem do odpovídajícího stavu).

  Navíc je zřejmé, že stroj M mohl během těchto r kroků modifikovat nejvýše 2 z těchto úseků (každý má totiž délku r, navíc nemohl modifikovat oba krajní úseky současně). V případě, že modifikoval pouze prostřední políčko, lze jeho nový obsah zapsat během jednoho kroku (v takovém případě udělá stroj jeden prázdný krok), jinak budou potřeba dva kroky (doprava a doleva, v odpovídajícím pořadí, podle toho, které z krajních políček bylo změněno).

- Stroj M' přijme či odmítne, jestliže stroj M během simulovaného úseku přijmul či odmítnul.

Předzpracování vstupní pásky vyžadovalo $n + \lceil n/r \rceil$ kroků (druhý sčítanec odpovídá času potřebnému pro návrat hlavy na začátek pásky).

Pro simulaci t kroků je zapotřebí $\lceil t/r \rceil$ bloků výpočtu M', z nichž každý trvá 6 kroků.

V součtu dostáváme tvrzení lemmatu. □

Věta 5 (O lineárním zrychlení).

Nechť L je jazyk přijímaný k-páskovým deterministickým Turingovým strojem M s časovou složitostí $t(n)$. Dále nechť platí $k \geq 2$ a $t(n) \in \omega(n)$. Potom pro $\forall c > 0$ existuje k-páskový deterministický Turingův stroj M' s časovou složitostí $c \cdot t(n)$ přijímající L.

Důkaz. Nechť r je celé číslo takové, že $r > 12/c$. Sestrojíme M' jako v předcházejícím lemmatu. M' pracuje nad vstupem délky n v čase $n + \lceil n/r \rceil + 6\lceil t(n)/r \rceil$. Pro $r \geq 2$ lze tento výraz zhora omezit jako $2n + 6 + (6/r) \cdot t(n)$.

Z předpokladu $t(n) \in \omega(n)$ dostáváme, že pro skoro všechna n je tento výraz menší než $(c/2) \cdot t(n) + (6/r) \cdot t(n)$. Z volby $r > 12/c$ dostáváme, že čas výpočtu je omezen $c \cdot t(n)$.

Konečně mnoho případů, pro které není nerovnost splněna, ošetříme konečným automatem a stroj M' nad nimi bude pracovat v konstantním čase. □

Důsledek 2.

Pokud $t(n) \in \omega(n)$, potom $\forall\, c > 0 : DTIME(t(n)) = DTIME(c \cdot t(n))$

Důkaz. Plyne přímo z věty 5. □

Věta 6 (O lineárním zrychlení, část 2).

Nechť L je jazyk přijímaný k-páskovým deterministickým Turingovým strojem M s časovou složitostí $t(n) = c \cdot n$. Dále nechť platí $k \geq 2$ a $c > 1$. Potom pro $\forall \varepsilon > 0$ existuje k-páskový deterministický Turingův stroj M' s časovou složitostí $(1 + \varepsilon) \cdot n$ přijímající L.

Důkaz. Budeme požadovat, aby

$$n + \lceil n/r \rceil + 6\lceil cn/r \rceil \leq (1+\varepsilon) \cdot n \quad (6)$$

$$n + n/r + 6cn/r + 7 \leq (1+\varepsilon) \cdot n \quad (7)$$

$$(1 + 1/r + 6c/r + 7/n) \cdot n \leq (1+\varepsilon) \cdot n \quad (8)$$

Vztah (6) je požadovaná nerovnost, v (7) je na levé straně horní odhad levé strany (6), tedy stačí ukázat (7). Nerovnost (8) je prostá úprava (7).

Ukážeme, že lze volit r a n_0 tak, že pro $n > n_0$ platí $(1/r + 6c/r + 7/n) < \varepsilon$. Požadujeme:

$$1/r \leq \varepsilon/3 \qquad \text{dostáváme } r \geq 3/\varepsilon \quad (9)$$

$$6c/r \leq \varepsilon/3 \qquad \text{dostáváme } r \geq 18c/\varepsilon \quad (10)$$

$$7/n_0 \leq \varepsilon/3 \qquad \text{dostáváme } n_0 \geq 21/\varepsilon \quad (11)$$

Z (9) a (10) dostáváme $r = max\{3/\varepsilon, 18c/\varepsilon\} = 18c/\varepsilon$.
Z (11) dostáváme $n_0 = 21/\varepsilon$.
Tvrzení věty je dokázáno. □

Důsledek 3.
Pokud $t(n) = cn$, kde $c > 1$, potom
$\forall\, \varepsilon > 0 : DTIME(t(n)) = DTIME((1+\varepsilon)n)$

Důkaz. Plyne přímo z věty 6. □

Věta 7 (Nedeterministická verze).
Pokud $t(n) \in \omega(n)$, potom $\forall c > 0 : NTIME(t(n)) = NTIME(c \cdot t(n))$
Pokud $t(n) = cn$, $c > 1$, potom $\forall \varepsilon > 0 : NTIME(t(n)) = NTIME((1+\varepsilon)n)$

Důkaz. Předchozí důkazy lze zobecnit (pomocí odhadu počtu dosažitelných konfigurací) i pro nedeterministický případ. □

3.2.1 Redukce počtu pásek

Věta 8 (O redukci počtu pásek pro časovou složitost).
Nechť $L \in DTIME(T(n))$[1]. Potom existuje jednopáskový deterministický Turingův stroj M s časovou složitostí $(T(n))^2$, který přijímá L.

Důkaz. Použijeme kombinaci věty o zrychlení a věty o redukci počtu pásek pro prostorovou složitost.

Pokud je L přijímán jednopáskovým strojem, není co dokazovat.

Nechť je tedy přijímán k-páskovým strojem, kde $k > 1$.

1) Nechť $T(n) \in \omega(n)$. Dle věty o lineárním zrychlení existuje deterministický Turingův stroj M' přijímající L v čase $\frac{1}{\sqrt{2k}} \cdot T(n) = T'(n)$ v prostoru $S'(n)$.

   Nyní použijeme větu o redukci počtu pásek pro prostorovou složitost a z ní odvodíme stroj, který má jednu pásku a který přijímá L v čase $T''(n) \leq 2k \cdot S'(n) \cdot T'(n) \leq 2k \cdot (T'(n))^2 = (T(n))^2$

[1] $T(n)$ splňuje jisté "mírné požadavky": předpokládejme, že buď $T(n) \in \omega(n)$ nebo $T(n) = cn$, kde $c > \sqrt{2k}$

2) Nechť $T(n) = cn$, kde $c > \sqrt{2k}$. Dle věty o lineárním zrychlení zvolíme ε tak, aby platilo

$$1 + \varepsilon = \frac{c}{\sqrt{2k}} \quad \text{tedy} \quad (1+\varepsilon)n = \frac{1}{\sqrt{2k}} T(n) \tag{12}$$

Zbytek stejně jako v bodě 1.

□

Důsledek 4.
Pokud $L \in NTIME(T(n))$, potom existuje jednopáskový nedeterministický Turingův stroj s časovou složitostí $(T(n))^2$ přijímající L.

Důkaz. Plyne přímo z věty 8. □

Věta 9 (O redukci počtu pásek pro časovou složitost, 2).
Nechť je L přijímán k-páskovým deterministickým Turingovým strojem M s časovou složitostí $T(n)$. Potom existuje 2-páskový deterministický Turingův stroj M' s časovou složitostí $T(n)\log_2 T(n)$ přijímající L.

Důkaz. První páska stroje bude mít $2k$ stop, teda každý znak abecedy $\Sigma_{M'}$ bude kódovat $2k$ znaků původní abecedy (plus speciální prázdný symbol).

Druhá (pomocná) páska bude sloužit k přesunům dat na pásce první.

Obě pásky předpokládáme oboustranně nekonečné.

Hlavní myšlenka: Abychom se vyhnuli hledání displeje původního stroje, budeme udržovat displej na jediném políčku. Tedy místo pohybu hlav budeme pohybovat obsahem celé pásky. Přesuny dat budeme dělat tak, aby malé přesuny byly častější a velké (vzdálené) přesuny byly vzácné.

První páska stroje bude rozdělena do bloků $\ldots, B_{-2}, B_{-1}, B_0, B_1, B_2, \ldots$. Blok B_0 se sestává z jediného políčka a bude obsahovat displej stroje M. Pro ostatní bloky platí: $\forall i, |i| \geq 1 : |B_i| = 2^{|i|-1}$. Bloky jsou odděleny značkami, které jsou umisťovány při prvním použití bloku.

Ukážeme simulaci práce M na první pásce (stroje M).

Na začátku výpočtu je obsah první pásky M na spodní stopě pásky M'.

Po celou dobu simulace budou platit následující invarianty:

1) Pro každé $i > 0$ nastává právě jedna ze tří možností.

    a) B_i má obě stopy plné a B_{-i} má obě stopy prázdné;

    b) B_{-i} má obě stopy plné a B_i má obě stopy prázdné;

    c) B_i i B_{-i} mají spodní stopu plnou a horní stopu prázdnou.

2) $\forall i$ B_i obsahuje souvislý interval pásky stroje M. Pro $i < 0$ obsahuje horní stopa symboly vpravo od dolní stopy, pro $i > 0$ horní stopa obsahuje symboly vlevo od dolní stopy.

3) B_0 obsahuje symbol pouze na dolní stopě, horní stopa obsahuje speciální značku, která slouží pro vyhledání tohoto bloku.

4) $\forall i$ B_i obsahuje interval pásky stroje M bezprostředně nalevo od obsahu bloku B_{i+1}.

Simulace jednoho kroku M. Displej je v B_0. Krok se skládá ze změny dat na pásce (zřejmé) a simulace posunu hlavy. Ukážeme si případ posunu hlavy doleva, tedy data budeme posunovat doprava.

i) Hlava jede doprava, dokud nenajde blok B_i takový, že má alespoň jednu stopu volnou. (To znamená, že všechny bloky, které přejel, byly zcela plné!)

ii) Hlava jede zpátky a kopíruje obsah bloků $B_{i-1}, B_{i-2}, \ldots, B_0$ na pomocnou pásku. Bloky projede každý třikrát, neboť musí překopírovat obsah obou stop a to ve správném pořadí.

iii) Hlava jede doprava a kopíruje obsah pomocné pásky do spodní stopy bloků $B_1, B_2, \ldots, B_{i-1}$. Horní stopy vyprázdní. V okamžiku, kdy hlava dojede na konec bloku B_{i-1}, zbyde na pracovní pásce přesně $|B_i|$ nepřekopírovaných políček.

   a) V případě, že byl B_i zcela prázdný, napíše je M' do spodní stopy.
   b) V případě, že byl B_i poloprázdný, napíše je M' do horní stopy.

iv) Hlava jede vlevo a na pomocnou pásku si počítá vzdálenost bloku B_i od B_0.

v) Hlava jede vlevo a pomocí počítadla najde levý okraj B_{-i}.

vi) Hlava jede vpravo a na pomocnou pásku kopíruje obsah

   a) horní stopy B_{-i}, v případě, že byl B_{-i} zcela plný (tj. B_i byl zcela prádzný)
   b) dolní stopy B_{-i}, v případě, že byl B_{-i} poloplný (tj. B_i byl poloplný)

vii) Hlava jede vpravo a do spodní stopy bloků $B_{-i+1}, B_{-i+2}, \ldots, B_0$ kopíruje obsah pomocné pásky. Bloky $B_{-i+1}, B_{-i+2}, \ldots, B_0$ musely být prázdné, neboť B_i byl první blok napravo od B_0, který nebyl plný.

Kroky i - vii nazveme B_i operací. Pro B_i operaci platí, že zachovává platnost uvedených invarantů a trvá čas $O(|B_i|)$.

Jaká je časová složitost celé simulace? Operace B_i může být provedena pouze jednou za 2^{i-1} kroků stroje M. Důvodem je fakt, že po jejím provedení jsou horní stopy bloků $B_1, \ldots, B_{i-1}$ prázdné a tedy je potřeba posunout pásku alespoň 2^{i-1}-krát, což vyžaduje nejméně takový počet kroků.

Pokud stroj M udělá $T(n)$ kroků během výpočtu, může se B_i operace provést nejvýše $\lfloor T(n)/2^{i-1} \rfloor$-krát.

Stačí uvažovat B_i operace pro $i \leq \lceil \log_2 T(n) \rceil \leq \log_2 T(n) + 1$.

Nechť m je konstanta taková, že každá B_i operace trvá nejvýše $m \cdot 2^{i-1}$ kroků. Stroj M' tedy udělá maximálně

$$\begin{aligned} T'(n) &\leq \sum_{i=1}^{\log_2 T(n)+1} m \cdot 2^{i-1} \lfloor T(n)/2^{i-1} \rfloor \leq \\ &\leq \sum_{i=1}^{\log_2 T(n)+1} m \cdot T(n) \leq \log_2 T(n) \cdot m \cdot T(n) \end{aligned} \tag{13}$$

Pro simulaci práce na všech k páskách je potřeba $k \cdot m \cdot T(n) \log_2 T(n)$.

Po použití věty o zrychlení dostáváme deterministický Turingův stroj M'', který přijímá L v čase $T(n) \log_2 T(n)$. $\square$

4 Konstruovatelnost funkcí

Definice 9. *Funkce* $f : \mathbf{N} \to \mathbf{N}$ *je rekurzivní, jestliže existuje deterministický Turingův stroj* M*, který pro vstup* 1^n *vydá výstup* $1^{f(n)}$*.*

Definice 10. *Funkce* $f : \mathbf{N} \to \mathbf{N}$ *je vyčíslitelná v čase* $O(f)$*, jestliže je rekurzivní a* $\exists c \geq 1$ *takové, že příslušný deterministický Turingův stroj* M *udělá nejvýše* $c \cdot f(n)$ *kroků, než vydá výstup* $1^{f(n)}$*.*

Definice 11. *Funkce* $f : \mathbf{N} \to \mathbf{N}$ *je vyčíslitelná v prostoru* $O(f)$*, jestliže je rekurzivní a* $\exists c \geq 1$ *takové, že příslušný deterministický Turingův stroj* M *použije nejvýše* $c \cdot f(n)$ *prostoru, než vydá výstup* $1^{f(n)}$*.*

Definice 12. *Funkce* $f : \mathbf{N} \to \mathbf{N}$ *je časově konstruovatelná, pokud existuje deterministický Turingův stroj* M *takový, že pro každý vstup délky* n *zastaví právě po* $f(n)$ *krocích.*

Definice 13. *Funkce* $f : \mathbf{N} \to \mathbf{N}$ *je prostorově konstruovatelná, pokud existuje deterministický Turingův stroj* M *takový, že pro každý vstup délky* n *zastaví s právě* $f(n)$ *páskovými políčky neprázdnými a během výpočtu žádný jiný prostor na pracovních páskách nebyl použit.*

Lemma 2.
Nechť $f_1 + f_2$ *a* f_2 *jsou časově konstruovatelné funkce a nechť* $\exists \varepsilon > 0\ \exists n_0\ \forall n > n_0 : f_1(n) \geq \varepsilon f_2(n) + (1 + \varepsilon)n$*. Potom je* f_1 *časově konstruovatelná.*

Důkaz. Metodou podobnou té z důkazu věty o lineárním zrychlení zrychlíme výpočet trvající $f_1(n) + f_2(n)$ kroků tak, aby trval přesně $f_1(n)$ kroků.

Mějme M_1 deterministický Turingův stroj pracující v čase $f_1(n) + f_2(n)$ a M_2 deterministický Turingův stroj pracující v čase $f_2(n)$ (z předpokladů).

Zkonstruujeme M_r, který pracuje v čase $f_1(n)$ pro skoro všechny vstupy.

Práce M_r je rozdělena do 4 fází.

1) V první fázi přepíše vstup na první pracovní pásku, na kterou jej zapíše r-krát zahuštěný. Zároveň přepíše $(r-6)$-krát zahuštěný vstup také na 2. a 3. pracovní pásku. Na to potřebuje n kroků. Zároveň řídící jednotka počítá modulo $(r-6)$ a po skončení přepisu provede maximálně $r-5$ kroků tak, aby celkový počet kroků první fáze byl dělitelný $(r-6)$. Tedy dostáváme

$$T_1 = (r-6) \cdot \left\lceil \frac{n}{r-6} \right\rceil . \tag{14}$$

2) V druhé fázi M_r souběžně simuluje stroj M_1 na první pásce a M_2 na druhé pásce. Šest kroků (jedna sekvence) stroje M_r odsimuluje r kroků M_1 a $(r-6)$ kroků M_2.

   Zvolíme r dostatečně velké, aby simulace stroje M_2 skončila dříve, a to po $\lceil f_2(n)/(r-6) \rceil$ sekvencích stroje M_r.

   Potřebujeme zajistit:

$$\left\lceil \frac{f_1(n) + f_2(n)}{r} \right\rceil > \left\lceil \frac{f_2(n)}{r-6} \right\rceil \tag{15}$$

Přibližně:

$$
\begin{aligned}
(f_1(n) + f_2(n)) \cdot (r-6) &> r \cdot f_2(n) \\
f_1(n) \cdot (r-6) &\geq 6 f_2(n) \\
r - 6 &\geq \frac{6 f_2(n)}{f_1(n)} \\
f_1(n) &\geq \frac{6}{r-6} f_2(n) \\
r &\geq \frac{6(f_1(n) + f_2(n))}{f_1(n)}
\end{aligned}
$$

V okamžiku, kdy M_2 skončí, zastaví stroj M' simulaci. Pomocí řídící jednotky si počítal modulo $(r-6)$. V případě, že $f_2(n)$ nebylo násobkem $(r-6)$, provede stroj M' přesně $(r-6)\lceil f_2(n)/(r-6)\rceil - f_2(n)$ prázdných kroků, čímž se trvání druhé fáze doplní na nejbližší větší násobek $(r-6)$. Z uvedeného dostáváme, že počet kroků stroje M' potřebný pro druhou fázi je

$$
\begin{aligned}
T_2 = 6 \cdot \left\lceil \frac{f_2(n)}{r-6} \right\rceil + (r-6) \left\lceil \frac{f_2(n)}{r-6} \right\rceil - f_2(n) = \\
= r \cdot \left\lceil \frac{f_2(n)}{r-6} \right\rceil - f_2(n)
\end{aligned}
\tag{16}
$$

Z předpokladů plyne, že T_2 je pro skoro všechna n kladné.

Je zřejmé, že během této fáze bylo odsimulováno $r \cdot \lceil f_2(n)/(r-6)\rceil$ kroků stroje M_1, neboť během každé ze sekvencí bylo odsimulováno r kroků stroje M_1 (kroky na doplnění na násobek $(r-6)$ jsou prázdné, M_1 se v nich nesimuluje).

3) V třetí fázi pokračuje M' simulací M_1 stejným tempem jako předtím (tedy 6 kroků za sekvenci odsimuluje r kroků půvopdního stroje). Tato fáze bude trvat $\lceil n/(r-6)\rceil + 1$ sekvencí, což zajistíme tak, že po každé sekvenci přečteme jeden symbol z přepsaného komprimovaného vstupu (viz fáze 1).

   Čas pro tuto fázi je

$$
T_3 = 6 \cdot \left(\left\lceil \frac{n}{r-6} \right\rceil + 1 \right) \tag{17}
$$

   a je odsimulováno $r \cdot (\lceil n/(r-6)\rceil + 1)$ kroků stroje M_1.

   Při srovnání času výpočtu stroje M_1 s dosud odsimulovaným počtem je zřejmé, že pro dostatečně velké hodnoty r tato simulace pro skoro všechna n nedosáhne celého výpočtu M_1.

4) Ve čtvrté fázi pokračuje M' *real-time* simulací stroje M_1 (jeden krok M' odpovídá jednomu kromu M_1), dokud se M_1 nezastaví. Na závěr provede M' $r-6$ prázdných kroků a zastaví se. Čas na tuto fázi je zbývající čas stroje M_1 plus $r-6$, z toho dostáváme:

$$
T_4 = f_1(n) + f_2(n) - r \cdot \left\lceil \frac{f_2(n)}{r-6} \right\rceil - r \cdot \left(\left\lceil \frac{n}{r-6} \right\rceil + 1 \right) + r - 6 \tag{18}
$$

Hodnota r musí být dost velká, aby tento výraz byl pro skoro všechna n kladný. Existence takového r plyne z předpokladů tvrzení.

Součtem T_1, T_2, T_3, T_4 dostáváme, že čas pro běh stroje M' je přesně $f_1(n)$ pro skoro všechna n. $\square$

Věta 10 (O ekvivalenci definic 10 a 12).
Nechť $f : \mathbf{N} \to \mathbf{N}$ je funkce taková, že $\exists\varepsilon>0\ \exists n_0\ \forall n\geq n_0 : f(n) \geq (1+\varepsilon)n$. Potom f je časově konstruovatelná právě, když f je vyčíslitelná v čase $O(f)$.

Důkaz. Implikace zleva doprava je zřemá. Stroj dokazující časovou konstruovatelnost modifikujeme tak, že na novou pásku v každém kroku napíše jeden symbol. Nový stroj vyčísluje f v čase $O(f)$.

Nyní opačná implikace. Nechť M je deterministický Turingův stroj, který vyčísluje f v čase $O(f)$. Označme $g(n)$ počet kroků stroje M nad vstupem 1^n (zřejmě $\exists c>0 : g(n) \leq cf(n)$).

1) g je časově konstruovatelná (díky existenci stroje M).

2) $g+f$ je časově konstruovatelná. Modifikuji M tak, aby počítal výstup na zvláštní pásku a po skončení výpočtu přejel na začátek této pásky. Doba výpočtu je $g(n)$, délka výstupu je $f(n)$.

3) $\exists\varepsilon>0\ \exists n_0\ \forall n>n_0 : f(n) \geq \varepsilon g(n) + (1+\varepsilon)n$

   Zvolíme:

   a) ε_1, jehož existenci nám zajišťují předpoklady věty: $\exists n_0\ \forall n\geq n_0 : f(n) \geq (1+\varepsilon_1)n$
   b) ε_2 zvolíme tak, aby $(1+\varepsilon_1)(1-\varepsilon_2) > 1$
   c) $\varepsilon_3 = (1+\varepsilon_1)(1-\varepsilon_2) - 1$
   d) $\varepsilon_4 = \min\{\varepsilon_2/c, \varepsilon_3\}$

Potom

$$\begin{aligned} f(n) = \varepsilon_2\cdot f(n) + (1-\varepsilon_2)f(n) &\geq \varepsilon_2/c\cdot g(n) + (1+\varepsilon_1)(1-\varepsilon_2)n = \\ &= (\varepsilon_2/c)\cdot g(n) + (1+\varepsilon_3)n \geq \varepsilon_4\cdot g(n) + (1+\varepsilon_4)n. \end{aligned} \tag{19}$$

Funkce f a g splňují předpoklady lemmatu 2 a tedy $f(n)$ je časově konstruovatelná. $\square$

Věta 11 (O ekvivalenci definic 11 a 13).
Funkce $f : \mathbf{N} \to \mathbf{N}$ je prostorově kontruovatelná právě, když f je vyčíslitelná v prostoru $O(f)$.

Důkaz. Implikace zleva doprava je zřemá. Stroj dokazující prostorovou konstruovatelnost modifikujeme tak, že na novou výstupní pásku v každém kroku, ve kterém bylo zabráno nové políčko na pracovní pásce, napíše jeden symbol. Stroj musí rozlišovat původní a nový symbol pro prázdné políčko. Nový stroj vyčísluje f v protoru $O(f)$.

Nechť M je k-páskový deterministický Turingův stroj vyčíslující $f(n)$ v prostoru $c\cdot f(n)$. Podle věty o lineární kompresi zkonstruujeme k-páskový stroj M', který vyčísluje $f(n)$ v prostoru přesně $f(n)$. Uvědomte si, že M' vyčísluje $f(n)$, tedy musí pracovat v prostoru alespoň $f(n)$. Stroj M' dále převedeme podle věty o redukci počtu pásek na jednopáskový stroj M'', který již dokazuje prostorovou konstruovatelnost funkce f. $\square$

Důsledek 5.
Každá časově konstruovatelná funkce je prostorově konstruovatelná.

Důkaz. Funkce f je časově konstruovatelná, tedy f je vyčíslitelná v čase $O(f)$, tím spíše je f vyčíslitelná v protoru $O(f)$ a tedy dostáváme dle předchozí věty, že f je prostorově konstruovatelná. □

Věta 12 (O rychlosti růstu časově konstruovatelných funkcí).
Nechť $f\colon \mathbf{N}\to\mathbf{N}$ je rekurzivní funkce. Potom existuje časově konstruovatelná funkce $g\colon \mathbf{N}\to\mathbf{N}$ taková, že $\forall n\ g(n) > f(n)$.

Důkaz. Víme, že f je rekurzivní, tedy existuje deterministický Turingův stroj M, který pro vstup 1^n vydá výstup $1^{f(n)}$.

Definujeme $g(n)$ jako počet kroků stroje M nad vstupem 1^n zvětšený o jedna. Je zřejmé, že $f(n) < g(n)$ a $g(n)$ je časově konstruovatelná (samotný M tuto vlastnost přímo dokazuje). □

5 Hierarchie tříd složitosti

5.1 Časová hierarchie

Definice 14. *Jazyk L je rekurzivní, jestliže existuje deterministický Turingův stroj M, který se pro každý vstup x zastaví a který přijme právě, když $x \in L$.*

Věta 13 (O otevřenosti časové hierarchie shora).
Nechť $T : \mathbf{N} \to \mathbf{N}$ je rekurzivní funkce. Potom existuje rekurzivní jazyk L takový, že $L \notin DTIME(T(n))$.

Důkaz. Ukážeme konstrukci takového jazyka. Nejprve očíslujeme všechny deterministické Turingovy stroje Gödelovými čísly a také očíslujeme všechny řetězce nad $\{0,1\}$. Očíslování provedeme systematicky, aby bylo možno tyto řetězce generovat.

Definujeme $L = \{x_i : M_i \text{ nepřijímá } x_i \text{ v čase } T(|x_i|)\}$.

Ukažme nejprve, že L je rekurzivní.

1) Pro vstup $w \in \{0,1\}^*$ délky n generujeme na pomocnou pásku počítadlo $T(n)$ (lze, neboť T se vždy zastaví).

2) Postupným generováním najdeme i takové, že $w = x_i$. To lze v konečném počtu kroků.

3) Uvažuji i jako Gödelovo číslo nějakého stroje.

4) Provedu simulaci stroje M_i nad x_i po $T(n)$ kroků. Slovo x_i přijmu v následujících případech:

   a) M_i zastaví po méně než $T(|x_i|) + 1$ krocích a odmítne;
   b) M_i běží více jak $T(|x_i|) + 1$ kroků.

Nyní ukažme, že $L \notin DTIME(T(n))$. Pro spor předpokládejme, že uvedené platí, tedy $L \in DTIME(T(n))$. Potom existuje deterministický Turinův stroj M, který rozpozná L v čase $T(n)$. Nechť i je jeho Gödelovo číslo ($M_i = M$).

1) $x_i \in L$, potom M_i přijímá x_i v čase $T(|x_i|)$, z čehož plyne, že x_i není přijmuto strojem M, ale $M = M_i$. Spor.

2) $x_i \notin L$, potom M_i nepřijímá x_i v čase $T(|x_i|)$, z čehož plyne, že x_i je přijmuto strojem M, ale $M = M_i$. Spor.

Tím je věta dokázána. □

Důsledek 6 (Nekonečná časová hierarchie).
Existuje nekonečná posloupnost funkcí $T_1, T_2, \ldots$ taková, že $DTIME(T_1(n)) \subsetneq DTIME(T_2(n)) \subsetneq \ldots$.

Důkaz. Nechť je hiearchie vybudována až po $DTIME(T_i(n))$.

Z předchozí věty víme, že existuje rekurzivní jazyk L takový, že pro něj platí $L \notin DTIME(T_i(n))$. Definujeme funkci T_{i+1} takovou, aby majorizovala funkci T_i a zároveň i dobu výpočtu stroje, který přijímá jazyk L (nechť je tento výpočet omezen funkcí T').

Tím budeme mít zaručeno, že $DTIME(T_i(n)) \subseteq DTIME(T_{i+1}(n))$ a také $L \in DTIME(T_{i+1}(n))$. Z faktu, že $L \notin DTIME(T_i(n))$ dostáváme ostrou inkluzi.

K tomu stačí ale položit $T_{i+1} = \max\{T_i(n), T'(n)\}$. □

5.2 Prostorová hierarchie

Věta 14 (O otevřenosti prostorové hierarchie shora).
Nechť $S : \mathbf{N} \to \mathbf{N}$ je rekurzivní funkce. Potom existuje rekurzivní jazyk L takový, že $L \notin DSPACE(S(n))$.

Důkaz. Důkaz je analogický důkazu časové verze této věty.

Definujeme $L = \{x_i : M_i \text{ nepřijímá } x_i \text{ v prostoru } S(|x_i|)\}$.

Rozdíl oproti časové verzi je v tom, že stroj M_i nyní může odmítnout konečným výpočtem ve vymezeném prostoru, překročením vymezeného prostoru nebo vstupem do nekonečné smyčky uvnitř vymezeného prostoru. To lze ale ošetřit odhadem počtu konfigurací (na omezeném prostoru existuje konečně mnoho konfigurací) a přidáním budíku, který stroj zastaví v okamžiku, kdy počet kroků bude vyšší než počet konfigurací. □

Důsledek 7 (Nekonečná prostorová hierarchie).
Existuje nekonečná posloupnost funkcí $S_1, S_2, \ldots$ taková, že $DSPACE(S_1(n)) \subsetneq DSPACE(S_2(n)) \subsetneq \ldots$.

Důkaz. Analogicky jako v případě časové hierarchie. □

5.3 Věty o hierarchii

Věta 15 (O prostorové hierarchii).
Nechť $S_1 : \mathbf{N} \to \mathbf{N}$ a $S_2 : \mathbf{N} \to \mathbf{N}$ jsou funkce takové, že $S_2 \in \omega(S_1)$ a S_2 je prostorově konstruovatelná.

Potom existuje jazyk L takový, že $L \in DSPACE(S_2(n)) \setminus DSPACE(S_1(n))$.

Důkaz. Naším cílem bude zkonstruovat deterministický Turingův stroj pracující v prostoru $S_2(n)$, který se od každého stroje pracujícího v prostoru $S_1(n)$ liší alespoň na jednom vstupu.

Prefixově si očíslujeme všechny jednopáskové stroje nad abecedou $\{0, 1\}$.

Popíšeme práci M na vstupu w, $|w| = n$. V první fázi si M označí $S_2(n)$ buněk na pracovní pásce. Dojde-li v dalších fázích k opuštění vymezeného prostoru, stroj se zastaví a odmítne.

V druhé fázi M simuluje M_w a přijme, pokud M_w odmítne w a neopustí přitom vymezený prostor.

Platí $L \in DSPACE(S_2(n))$ a $L \notin DSPACE(S_1(n))$. První vztah je zřejmý, ukážeme druhý (sporem).

Nechť pro spor $L \in DSPACE(S_1(n))$, potom existuje deterministický Turingův stroj M' takový, že $L(M') = L(M)$. M' má velikost pracovní abecedy t a pracuje v prostoru $S_1(n)$. Bez újmy na obecnosti můžeme předpokládat, že M' se vždy zastaví a je jednopáskový.

Kdyby M' neměl požadované vlastnosti, lze jej modifiovat tak, že bude schopen rozpoznat nekonečný cyklus (pomocí horního odhadu na počet konfigurací). Počet možných konfigurací je $s \cdot (n+1) \cdot S_1(n) \cdot t^{S_1(n)}$, kde s je počet stavů a t je počet páskových symbolů. Tento počet lze odhadnout jako $c^{S_1(n)}$ pro vhodnou konstantu c. Uložíme-li si toto číslo v základu c, vejde se do prostoru $S_1(n)$. Tedy stroj si bude na nové pásce počítat kroky a přesáhne-li počet kroků počet možných konfigurací, odmítne. Podle věty o redukci počtu pásek dotáváme jednopáskový stroj s požadovanými vlastnostmi.

K simulaci M' potřebujeme $\lceil \log_2 t \rceil \cdot S_1(n)$ prostoru. Vzhledem k volbě prefixového číslování strojů můžeme zvolit w tak, že kóduje stejný stroj, ale současně platí $\lceil \log_2 t \rceil \cdot S_1(|w|) \leq S_2(|w|)$. To jistě lze, neboť S_2 roste rychleji než S_1.

Pakliže stroj M' přijímá w, potom $L \in L(M') = L(M)$, ale dle definice stroje M platí $w \notin L(M)$. Jestliže naopak M' odmítne w, potom $w \notin L(M') = L(M)$, ale M' odmítne w ve vymezeném prostoru, tedy podle definice M platí $w \in L(M)$. V obou případech dostáváme spor. Tedy $L \notin DSPACE(S_1(n))$. $\square$

Věta 16 (O časové hierarchii).
Nechť $T_1 : \mathbf{N} \to \mathbf{N}$ *a* $T_2 : \mathbf{N} \to \mathbf{N}$ *jsou funkce takové, že* $T_2 \in \omega(T_1 \cdot \log T_1)$ *a* T_2 *je časově konstruovatelná.*

Potom existuje jazyk L *takový, že* $L \in DTIME(T_2(n)) \setminus DTIME(T_1(n))$.

Důkaz. Naším cílem bude zkonstruovat deterministický Turingův stroj pracující v čase $T_2(n)$, který se od každého stroje pracujícího v čase $T_1(n)$ liší alespoň na jednom vstupu.

Prefixově si očíslujeme všechny dvoupáskové stroje.

Práce M na vstupu w, kde $|w| = n$.

1) Na dvou páskách simuluje práci stroje M_w.

2) Na dalších páskách stopuje $T_2(n)$ kroků. To jistě lze, neboť T_2 je časově konstruovatelná.

3) M přijme, pokud simulace skončí nejvýše v $T_2(n)$ krocích odmítnutím w.

Označme $L = L(M)$. Ukážeme, že $L \in DTIME(T_2(n)) \setminus DTIME(T_1(n))$.

Použitím čítače kroků máme zajištěno, že $L \in DTIME(T_2(n))$. Ukažme tedy, že $L \notin DTIME(T_1(n))$.

Důkaz budeme dělat sporem. Předpokládejme, že existuje stroj M' takový, že M' přijímá L v čase $T_1(n)$. Potom existuje podle věty o redukci počtu pásek stroj M'', který má dvě pásky a přijímá L v čase $T_1(n) \cdot \log_2 T_1(n)$

Nechť pásková abeceda stroje M'' má t symbolů. M bude na simulaci M'' potřebovat $c(t) \cdot T_1(n) \cdot \log_2 T_1(n)$, kde $c(t)$ je konstanta závislá na t.

Nechť v je Gödelovo číslo stroje M''. Zvolíme $w = \alpha v$, kde α je prefix takové délky, aby platilo $c(t)T_1(|w|)\log T_1(|w|)) \leq T_2(|w|)$. Taková délka prefixu existuje z předpokladů věty.

Nyní má M dostatek času k simulaci M'' (trik byl v tom, že jsme právě zvětšili vstup stroje, přestože stroj bude počítat totéž, pouze na to bude mít více času).

Rozlišme dva případy:

1) $M_w(= M'')$ přijímá w, potom
   a) $w \in L$, protože $L = L(M) = L(M'')$,
   b) $w \notin L$, protože M odmítne w, když jej M_w přijme.

2) M_w odmítá w, potom
   a) $w \notin L$, protože $L = L(M) = L(M'')$,
   b) $w \in L$, protože M přijme w, když jej M_w odmítne.

Spor s předpokladem, že $L \in DTIME(T_1(n))$. Věta je dokázána. □

5.4 Věta o vtazích mezi třídami složitosti

Věta 17 (O vztazích mezi třídami složitosti).

a) $DTIME(f(n)) \subseteq NTIME(f(n))$
$DSPACE(f(n)) \subseteq NSPACE(f(n))$

b) $DTIME(f(n)) \subseteq DSPACE(f(n))$

c) $L \in DSPACE(f(n))$ *a* $f(n) \geq \log_2 n$, *potom* $\exists c_L : L \in DTIME({c_L}^{f(n)})$, *kde* c_L *je konstanta závislá na* L

d) $L \in NTIME(f(n))$, *potom* $\exists c_L : L \in DTIME({c_L}^{f(n)})$, *kde* c_L *je konstanta závislá na* L

Důkaz.

a) Triviálně platí.

b) Z vět o lineární kompresi a o redukci počtu pásek.

c) Nechť M je jednopáskový stroj dokazující $L \in DSPACE(f(n))$. Počet jeho konfigurací je omezen $s \cdot (n+1) \cdot (f(n)+1) \cdot t^{f(n)} \leq d^{f(n)}$, pro vhodné d (s je počet stavů, t je počet páskových symbolů).

   Zkonstruujeme M', který simuluje stroj M a nejvýše po $d^{f(n)}$ krocích se zastaví. (Potřebujeme předpoklad časové konstruovatelnosti funkce f).

d) Nechť M je nedeterministický k-páskový stroj, který dokazuje, že $L \in NTIME(f(n))$. Počet jeho konfigurací je omezen $s \cdot (f(n)+1)^k \cdot t^{k \cdot f(n)} \leq d^{f(n)}$, pro vhodné d (s je počet stavů, t je počet páskových symbolů).

   Zkonstruujeme deterministický M' takový, že vygeneruje seznam všech konfigurací dosažitelných z počáteční konfigurace (například průchodem stromu výpočtu M).

   Generování seznamu lze provést v čase kvadratickém vzhledem k délce výsledného seznamu. Tato délka je omezena součinem počtu konfigurací a délky zápisu jedné konfigurace. Tedy $l \leq d^{f(n)} \cdot (k \cdot f(n) + 1 + k \cdot \log f(n)) \leq c^{f(n)}$.

   Tedy počet kroků je omezen $c^{2 \cdot f(n)}$.

□

Věta 18 (Rozšíření věty o vztazích).

b') $NTIME(f(n)) \subseteq NSPACE(f(n))$

c') $L \in NSPACE(f(n))$ a $f(n) \geq \log_2 n$, potom $\exists c_L : L \in DTIME({c_L}^{f(n)})$, kde c_L je konstanta závislá na L

Důkaz.

b') Na pomocné pásce kódujeme aktuální průchod stromem výpočtu. Jednotlivé kódy můžeme systematicky generovat, $(1,1,\ldots,1)$ až $(r,r,\ldots,r)$, kde r je maximální stupeň větvení.

   Vlastní simulaci jedné větve výpočtu lze provést v prostoru $f(n)$, na uložení informací o průchodu potřebujeme $\log r \cdot f(n)$. Simulaci tedy lze provést v prostoru $\log r \cdot f(n)$.

c') Počet možných konfigurací na omezeném prostoru je $d^{f(n)}$, kde d je konstanta.

   Uvažujme graf G všech konfigurací. Hrana (a,b) znamená, že konfigurace b je z konfigurace a dosažitelná v jednom kroku.

$$\begin{aligned} |V(G)| &\leq d^{f(n)} \\ |E(G)| &\leq c \cdot d^{f(n)} \end{aligned} \tag{20}$$

   Omezení počtu hran plyne z faktu, že změny v konfiguraci po jednom kroku jsou pouze lokální, tedy každá konfigurace má pouze konstatní počet sousedů.

   Na pracovní pásku vygenerujeme seznam všech konfigurací, to zabere čas $d^{f(n)} \cdot q \cdot f(n)$, kde q je konstanta.

   Deterministický stroj potom pracuje tak, že prochází graf G do hloubky a na pomocné pásce si zaznamenává, které vrcholy již navštívil. Na zjištění, který z vrcholů byl již navštíven, potřebuje čas úměrný délce dosavadního výpočtu. Stroj tedy pracuje v čase *počet vrcholů · zkopírování příslušné konfigurace na pracovní pásku.*

$$\begin{aligned} T(n) = |V(G)|qf(n) + |E(G)d^{f(n)}qf(n)| = \\ = qf(n)(d^{f(n)} + kd^{f(n)}d^{f(n)}) \leq h(d^2)^{f(n)} \leq c_L^{f(n)} \end{aligned} \tag{21}$$

Najde-li stroj při průchodu grafem přijímající konfiguraci, je tato dosažitelná z iniciální konfigurace a stroj přijme.

□

5.5 Savičova věta

Věta 19 (Savičova).
Nechť $S : \mathbf{N} \to \mathbf{N}$ *je prostorově konstruovatelná funkce, pro kterou platí* $S(n) \geq \log_2 n$. *Potom* $NSPACE(S(n)) \subseteq DSPACE(S^2(n))$.

Důkaz. Nechť M_1 je nedeterministický Turingův stroj přijímající jazyk L v prostoru $S(n)$, potom existuje konstanta c_L taková, že M_1 má nejvýše ${c_L}^{S(n)}$ konfigurací.

Pokud M_1 přijímá w, pak existuje posloupnost nejvýše ${c_L}^{S(n)} = 2^{S(n) \cdot \log c_L}$ kroků končící přijetím w.

Označme $I_1 \to_i I_2$, pokud lze z konfigurace I_1 přejít do I_2 za méně než nebo přesně 2^i kroků. Test stačí provádět pro $i \leq S(n) \cdot \log c_L = m \cdot S(n)$.

Popišme způsob rozhodování, že $I_1 \to_i I_2$.

procedure TEST(I_1, I_2, i)
  if $i = 0$ **then if** $(I_1 = I_2)$ **or** $(I_1 \to I_2)$ **then** return true **else** return false
    else
      for $\forall$ konfigurace $I' \in K$ **do**
        if TEST($I_1, I', i-1$) **and** TEST($I', I_2, i-1$) **then** return true
      enddo
  return false
end TEST

Kolik místa bude potřeba na jednu kopii procedury TEST? Procedura si musí pamatovat 3 konfigurace a parametr i. Jedna konfigurace obsahuje

- stav - konstantní prostor ($\log s$, kde s je počet stavů)
- pozice vstupní hlavy - $\log_2 n \leq S(n)$
- pozice pracovní hlavy - $\log_2 n \leq S(n)$
- obsah pracovní pásky - $c' \cdot S(n)$, kde $c' = \log_2 t$, t je počet páskových symbolů.

Celkem je tedy potřeba na zapsání jedné konfigurace $O(S(n))$ prostoru. Parametr i lze zapsat v prostoru $m \cdot \log S(n)$. Tedy prostor potřebný pro jednu kopii procedury TEST je $O(S(n))$.

Celá simulace potom probíhá tak, že pro všechny přijímající stavy I_j voláme proceduru $TEST(I_0, I_j, m \cdot S(n))$ a jakmile alespoň jednou odpoví kladně, vstup přijmeme.

Při simulaci funguje pracovní páska jako zásobník na uložení parametrů procedur TEST. V každý okamžik je na zásobníku $O(S(n))$ záznamů. Deterministický stroj simulující M_1 pracuje v prostoru $O(S^2(n))$ a přijímá jazyk L. □

6 Pokročilé věty o třídách složitosti

Lemma 3 (Translační).

Nechť $S_1 : \mathbf{N} \to \mathbf{N}$, $S_2 : \mathbf{N} \to \mathbf{N}$, $f : \mathbf{N} \to \mathbf{N}$ *jsou prostorově konstruovatelné a* $\forall n\ S_2(n) \geq n, f(n) \geq n$. *Potom*

$$NSPACE(S_1(n)) \subseteq NSPACE(S_2(n)) \Rightarrow$$
$$NSPACE(S_1(f(n))) \subseteq NSPACE(S_2(f(n))).$$

Důkaz. Nechť $L_1 \in NSPACE(S_1(f(n)))$ a nechť M_1 je stroj, který to ukazuje.

Sestrojíme jazyk L_2, který bude patřit do $NSPACE(S_1(n))$.

Nechť $L_2 = \{x\$^i | M_1$ přijímá x v prostoru $S_1(|x| + i)\}$. Zřejmě pro všechna i taková, že $|x|+i \geq f(|x|)$, platí $x \in L_1 \Leftrightarrow x\$^i \in L_2$. Tento argument se nazývá *padding argument*. Jde o to zvětšit délku slova bez přidání nové informace, čímž se stroji, který nad slovem pracuje, poskytne více prostoru (případně času).

Zkonstruujeme M_2 přijímající L_2 v prostoru $S_1(n)$.

Vstup: $x\$^i$

Algoritmus: Stroj M_2 nejprve označí $S_1(|x| + i)$ políček na pracovní pásce. Dále pokračuje M_2 simulací stroje M_1 a přijme právě tehdy, pokud přijme M_1 a neopustí přitom vyznačený prostor. M_2 zřejmě pracuje v prostoru $S_1(n)$, tedy $L \in NSPACE(S_1(n))$. Navíc platí $x \in L_1 \Leftrightarrow \exists i < f(|x|) : x\$^i \in L_2$. Jinými slovy, existuje i takové, že stroj M_1 bude mít dost prostoru pro přijetí v prostoru $S_1(|x| + i)$.

Protože podle předpokladu platí $NSPACE(S_1(n)) \subseteq NSPACE(S_2(n))$, dostáváme, že existuje nedeterministický Turingův stroj M_3, který přijímá L_2 v prostoru $S_2(n)$.

Nyní sestrojíme M_4, který přijímá L_1 v prostoru $S_2(f(n))$.

Popis práce M_4 nad vstupem x.

- Na půlené pracovní pásce si označí na horní stopě $f(|x|)$ políček a potom si na dolní stopě označí $S_2(f(|x|))$ políček, přičemž používá horní stopu jako vstup. To lze, neboť f i S_2 jsou prostorově konstruovatelné. Protože $S_2(|x|) \geq |x|$, M_4 zabere přesně $S_2(f(|x|))$.

- M_4 simuluje M_3 na vstupu $x\$^i$ (pro dostatečně velké i) s tím, že pokud je vstupní hlava M_3 nad x, je hlava M_4 na odpovídající pozici, je-li vstupní hlava M_3 nad $\$^i$, tak vstupní hlava M_4 zůstává na konci slova x a pozice hlavy je počítána na zvláštní pracovní pásce M_4.

- Počítadlo nepustí vstupní hlavu M_3 za pozici, kdy už by čítač přetekl z prostoru $S_2(f(|x|))$, tedy $i \leq 2^{S_2(f(|x|))}$, neboť v prostoru $S_2(f(|x|))$ může počítadlo binárně počítat až do $2^{S_2(f(|x|))}$.

- M_4 přijme x právě, když M_3 přijme.

Platí: $x \in L_1 \Leftrightarrow \exists i < f(|x|) : x\$^i \in L_2 = L(M_3)$. Víme ale, že $S_2(n) \geq n$, tedy $S_2(f(|x|)) \geq f(|x|)$, z čehož dostáváme $2^{S_2(f(|x|))} \geq f(|x|)$. Již dříve jsme ukázali, že stačí $i \leq f(|x|)$, tedy naše simulace umožňuje zkoušet dostatečně velká i.

Tedy i je dostatečně velké.

Na závěr dostáváme $x \in L(M_4) \Leftrightarrow x\$^i \in L(M_3) \Leftrightarrow x \in L_1$ a tedy $L_1 \in NSPACE(S_2(f(n)))$. $\square$

Věta 20 (O nedeterministické prostorové hierarchii).
Nechť $\varepsilon > 0$ a $r \geq 1$, potom $NSPACE(n^r) \subsetneq NSPACE(n^{r+\varepsilon})$.

Důkaz. Díky hustotě racionálních čísel víme, že

$$\forall r\ \forall \varepsilon\ \exists s,t \in \mathbf{N}:\ r \leq \frac{s}{t} < \frac{s+1}{t} \leq r + \varepsilon. \tag{22}$$

Ukážeme, že $NSPACE(n^{s/t}) \subsetneq NSPACE(n^{(s+1)/t})$. Důkaz provedeme sporem. Nechť tedy $NSPACE(n^{(s+1)/t}) \subseteq NSPACE(n^{s/t})$. Použijeme $(s+1)$-krát translační lemma, postupně pro funkce $f(n) = n^{(s+i)t}$ pro $i = 0, \dots, s$.

$$\begin{array}{ll} i = 0 & NSPACE(n^{(s+1)s}) \subseteq NSPACE(n^{s^2}) \qquad (23) \\ i = 1 & NSPACE(n^{(s+1)(s+1)}) \subseteq NSPACE(n^{s(s+1)}) \\ i = 2 & NSPACE(n^{(s+1)(s+2)}) \subseteq NSPACE(n^{s(s+2)}) \\ \vdots & \\ i = s-1 & NSPACE(n^{(s+1)(2s-1)}) \subseteq NSPACE(n^{s(2s-1)}) \\ i = s & NSPACE(n^{(s+1)(2s)}) \subseteq NSPACE(n^{s(2s)}) \end{array}$$

Pravá strana každé inkluze je vnořená do levé strany inkluze pro i o jedničku menší. Dostaneme prostým porovnáním exponentů u n.

Zřetězením dostaneme

$$NSPACE(n^{(s+1)(2s)}) \subseteq NSPACE(n^{s^2}) \subseteq DSPACE(n^{2s^2}) \subsetneq \tag{24}$$

$$\subsetneq DSPACE(n^{2s^2+2s}) \subseteq NSPACE(n^{(s+1)(2s)}). \tag{25}$$

Uvedené schéma vede ke sporu, neboť vpravo i vlevo je stejná třída a uprostřed je ostrá inkluze. □

Definice 15.
Tvrzení parametrizované číslem n je pravdivé skoro všude, platí-li na všech kromě konečně mnoha n.

Tvrzení parametrizované číslem n je pravdivé nekonečně často, platí-li na nekonečně mnoha n.

Lemma 4.
Nechť L je jazyk přijímaný deterministickým Turingovým strojem s prostorovou složitostí $S(n)$ skoro všude. Potom L je přijímaný jiným strojem M' s prostorovou složitostí $S(n)$.

Důkaz. Konečně mnoho výjimek, kde výpočet přesáhne $S(n)$ ošetříme pomocí tabulky. Takový stroj ale nelze efektivně sestrojit, protože nedokážeme vyjímky identifikovat. □

Lemma 5.
Nechť je dán deterministický Turingův stroj M, délka vstupu n a číslo m. Existuje algoritmus zjišťující, zda m je maximální počet použitých buněk na pracovních páskách při práci M na vstupech délky n. Algoritmus vždy skončí.

Důkaz. Stačí simulovat výpočet M a počítat kroky pro detekci případných nekonečných smyček (překročení počtu možných konfigurací). □

6.1 Věta o mezerách (Borodin)

Věta 21 (Borodinova, o mezerách).

Nechť $g(n) \geq n$ je rekurzivní funkce. Potom existuje rekurzivní funkce $S(n)$ taková, že $DSPACE(S(n)) = DSPACE(g(S(n)))$

Důkaz. Nechť $M_1, M_2, \ldots$ je očíslování všech deterministických strojů. Označme $S_i(n) =$ maximum použitých buněk stroje M_i na vstupu délky n.

Pokud se M_i vždy zastaví, je $S_i(n)$ jeho prostorová složitost. Pokud se M_i na nějakém vstupu délky n nezastaví, je $S_i(n)$ nedefinováno (respektive rovno $+\infty$).

Poznámka: zjistit, zda $S_i(n) = +\infty$ nelze, ale pro libovolné pevně dané m lze zjistit, zda $S_i(n) \leq m$ (viz lemma 5).

Zkonstruujeme funkci $S(n)$ takovou, že pro všechna k buď

$$S_k(n) \leq S(n) \text{ skoro všude} \tag{26}$$

nebo

$$S_k(n) > g(S(n)) \text{ nekonečně často.} \tag{27}$$

Uvedené nerovnosti požadujeme, abychom zajistili:

$$\forall k : L(M_k) \in DSPACE(S(n)) \Leftrightarrow L(M_k) \in DSPACE(g(S(n))) \tag{28}$$

Stroj M_k splňující 26 lze pomocí konečně mnoha výjimek změnit tak, aby nerovnost platila všude podle lemma 4, a stroj splňující 27 naopak nelze pomocí konečně mnoha výjimek změnit tak, aby platila opačná nerovnost.

Z 26 a 27 plyne, že neexistuje k takové, že $S(n) < S_k(n) \leq g(S(n))$ skoro všude. Stroj M_k by potom přijímal jazyk, který není v $DSPACE(S(n))$, ale je v $DSPACE(g(S(n)))$. Tomu se chceme v konstrukci vyhnout.

Popíšeme nyní konstrukci $S(n)$ pro danou hodnotu n. Vezmeme $M_1, \ldots, M_n$ a zvolíme $S(n)$ tak, že $\forall i, 1 \leq i \leq n : S_i(n) \leq S(n)$ nebo $g(S(n)) < S_i(n)$.

V případě, že jsou všechna $S_i(n)$ konečná, stačí vzít za $S(n)$ maximum, jak bylo ale již výše uvedeno, nelze o konečnosti $S_i(n)$ rozhodnout. Proto budeme problém řešit "z druhého konce".

Nechť $j = 1$ a spočítejme $g(j)$ (to lze, neboť g je rekurzivní). Zjistíme, zda existuje stroj M_i takový, že $j < S_i(n) \leq g(j)$ (to nám umožňuje lemma 5, neboť nyní máme horní odhad na $S_i(n)$).

Jestliže takové M_i neexistuje, můžeme položit $S(n) = j$, neboť žádný stroj M_i nepočítá s protorovou složitostí mezi $S(n)$ a $g(S(n))$.

Jestliže takové M_i naopak existuje, volíme $j = S_i(n)$. Pokračujeme spočítáním $g(j)$. Cyklus se zastaví nejvýše po n interacích, neboť v každé se jedno S_i stane dolní hranicí intervalu.

Takto konstruovaná funkce $S(n)$ splňuje podmínky 26 nebo 27, neboť pro dané k byl stroj M_k uvažován pro všechna $n \geq k$ v množině $M_1, \ldots, M_n$ při konstrukci $S(n)$. Tedy $S_k(n)$ nebylo v intervalu $(S(n), g(S(n))]$. Proto muselo být buď nekonečně mnohokrát nad tímto intervalem nebo skoro vždy pod.

Nechť $L \in DSPACE(g(S(n)))$, potom existuje deterministický Turingův stroj M_k rozpoznávající L v prostoru $g(S(n))$, tedy $\forall n\ S_k(n) \leq g(S(n))$. Z konstrukce $S(n)$ plyne, že pro $n > k$ platí $S_k(n) \leq S(n)$ skoro všude. Tedy M_k

pracuje v prostoru $S(n)$ skoro všude a podle lemma 4 existuje jiný stroj, který přijímá L v prostoru $S(n)$ všude, tedy $L \in DSPACE(S(n))$.

Máme $DSPACE(g(S(n))) \subseteq DSPACE(S(n))$. Inkluze na druhou stranu je triviální, tedy dostáváme rovnost a tvrzení věty je dokázáno. $\square$

6.2 Věta o zrychlení

Věta 22 (Bloomova, o zrychlení - prostorová verze).

Nechť $r(n)$ je rekurzivní funkce. Potom existuje rekurzivní jazyk L takový, že pro každý deterministický Turingův stroj M_i rozpoznávající L v prostoru $S_i(n)$ existuje deterministický Turingův stroj M_j rozpoznávající L v prostoru $S_j(n)$, kde $r(S_j(n)) \leq S_i(n)$ skoro všude.

Než přistoupíme k samotnému důkazu věty, uvedeme si dvě pomocná tvrzení.

Lemma 6.

Bez újmy na obecnosti můžeme předpokládat, že funkce r je neklesající a prostorově konstruovatelná. Navíc platí $r(n) \geq n^2$.

Důkaz. Pokud $r(n)$ nesplňuje uvedené požadavky, definujeme

$$r'(n) = \max\{\text{počet buněk použitých k výpočtu } r(1), \ldots, r(n), n^2\} \qquad (29)$$

Je zřejmé, že $r'(n)$ je neklesající a prostorově konstruovatelná. Také zřejmě platí $r'(n) \geq r(n)$ a tedy stačí dokázat větu pro $r'(n)$. $\square$

Lemma 7.

Definujeme-li $h(n) = r(h(n-1))$, $h(1) = 2$, platí, že $h(n)$ je prostorově konstruovatelná.

Důkaz. Plyne z prostorové konstruovatelnosti $r(n)$. $\square$

Důkaz. Nechť M_1, M_2, $\ldots$ jsou očíslované deterministické Turingovy stroje s jednoprvkovou vstupní abecedou. Dále budeme předpokládat, že délka zakódování stroje M_i je nejvýše $\log_2 i$ (bez důkazu).

Zkonstruujeme jazyk L takový, že

1) pokud stroj M_i rozpoznává L, pak $S_i(n) \geq h(n-i)$ skoro všude

2) $\forall k\ \exists j\ M_j : L(M_j) = L$ a $S_j(n) \leq h(n-k)$ skoro všude

Pokud takový L najdeme, jsme hotoví, neboť je-li dáno i takové, že $L(M_i) = L$, potom zvolíme $k = i+1$ a díky podmínce 2 víme, že $\exists j : L(M_j) = L$ a $S_j(n) \leq h(n-i-1)$, díky podmínce 1 potom $S_i(n) \geq h(n-i) = r(h(n-i-1)) \geq r(S_j(n))$.

Popišme nyní konstrukci takového jazyka L (nad abecedou $\{0\}$).

Nechť máme pevné n. Definujeme

$\sigma(n) = \min\{j : j \leq n, S_j(n) < h(n-j), \forall i < n : \sigma(i) \neq j\}$

Pokud je $\sigma(n)$ definováno, řekneme, že stroj $M_{\sigma(n)}$ je zrušen číslem n.

V jednom kroku konstrukce uvažuji pevně dané n a množinu těch strojů, které nebyly zrušeny žádným k, $k < n$. Tato množina je v každém kroku konstrukce konečná. Z této množiny vyberu stroj s nejmenším indexem $j = \sigma(n)$ takový, že $S_j(n) < h(n-j)$, pokud takový index existuje.

Nyní zaručím, aby žádný zrušený stroj nepřijímal jazyk L: $0^n \in L \Leftrightarrow$ $\sigma(n)$ je definované a $M_{\sigma(n)}$ nepřijímá O^n

Ukažme, že L splňuje obě uvedené podmínky.

Nechť i je takové, že $L(M_i) = L$ a uvažujme stroje $M_1, \ldots, M_{i-1}$. Některé z těchto strojů jsou zrušeny, každý je zrušený nějakým konkrétním n. Nechť n_0 je maximum z těchto čísel. Potom platí: $\forall n > \max\{n_0, i\} : S_i(n) \geq h(n-i)$. Dokážeme sporem. Nechť $\exists n > \max\{n_0, i\} : S_i(n) < h(n-i)$, potom je ale splněna podmínka pro zrušení M_i a tedy $L(M_i) \neq L$. Tedy podmínka 1 je splněna.

Ukažme dále, že L splňuje i podmínku 2.

Nechť k je libovolné pevné, zkonstruujme M_j takový, že $L(M_j) = L$ a $S_j(n) \leq h(n-k)$ skoro všude. Jinými slovy, M_j bude na rozhodnutí, zda 0^n patří do L, potřebovat prostor nejvýše $h(n-k)$ skoro všude.

Pro vstup O^n musí M_j:

1) zjistit, zda je $\sigma(n)$ definováno a pokud ano, tak $\sigma(n)$ spočítat,

2) po zjištění $\sigma(n)$ odsimulovat $M_{\sigma(n)}$, aby mohl rozhodnout, zda $0^n \in L$.

Chceme-li zjistit $\sigma(n)$, musíme nejprve zjistit, které stroje M_i, $i \leq n$ byly zrušeny nějakým číslem $l < n$. Pro pevné i a l stačí pracovat v prostoru $h(l-i)$, protože pokud číslo l zruší stroj M_i, tak $S_i(l) \leq h(l-i)$. Pro vymezení uvedeného množství prostoru potřebujeme, aby byla h prostorově konstruovatelná.

Zde se ovšem objeví problém v případě, že $l - i > n - k$, protože potom se simulace do vymezeného prostoru nevejde. Problém obejdeme drobný trikem. Uvažujme $M_1, \ldots, M_k$ a nechť n_1 je největší číslo, kterým je některý z těchto strojů zrušen. Modifikujeme M tak, že $\forall l \leq n_1$ bude řídící jednotka obsahovat informace o tom, zda $O^l \in L$ a číslo stroje zrušeného číslem l, pokud takový existuje. Potom $\forall n \leq n_1$ pracuje M s nulovou prostorovou složitostí, pro $n > n_1$ stačí odsimulovat stroje M_i pro $i = k+1, \ldots, n$.

Stroj M_i pracuje na vstupu 0^n v prostoru $h(l-i) \leq h(n-k)$, protože $l \leq n$ a $i \geq k$. Tedy stroj M_i se do uvedeného prostoru vejde, ještě je třeba ukázat, že se do něj vejde i stroj M při simulaci M_i.

Pro $n > n_1$, je-li definováno $\sigma(n)$, tak $\sigma(n)$ je alespoň k, tedy $M_{\sigma(n)}$ na vstupu O^n pracuje v prostoru $h(n - \sigma(n))$, což je nejvýše $h(n-k)$, protože $\sigma(n) > k$.

M postupně simuluje stroje M_i pro $k+1 \leq i \leq n$ na vstupech 0^l pro $l \leq n$. Pro každou dvojici (i, l) stačí reprezentovat na stroji $h(l-i)$ buněk stroje M_i.

$i \leq n$, M_i je zakódován v nejvýše $\log_2 i \leq \log_2 n$ buňkách stroje M, tedy tím spíše libovolný symbol stroje M_i lze zakódovat v $\log_2 n$ symbolech stroje M.

Protože víme, že $r(x) \geq x^2$, platí $h(x) \geq 2^{2^{x-2}}$ (důkaz lze indukcí).

Tedy $h(n-k) \geq 2^{2^{n-k-2}} \geq \log_2 n$ skoro všude, tedy prostor $h(n-k)$ stačí pro simulaci M_i pro skoro všechna n.

Kromě místa na simulaci potřebuje M také prostor na ukládání seznamu zrušených strojů. Zrušených strojů je nejvýše n, tedy je potřeba nejvýše $n \cdot \log_2 n$ místa. Platí $n \cdot \log_2 n \leq n^2 \leq h(n-k)$ skoro všude.

M tedy pracuje v prostoru $2h(n-k)$ skoro všude, dle lemma 4 existuje M' pracující v prostoru $2h(n-k)$ všude a podle věty o kompresi existuje stroj M'', který pracuje v prostoru $h(n-k)$ všude. $\square$