Vlastnosti bezkontextových jazyků

Věta 3.58. (a 3.61.) Třída bezkontextových jazyků ($\mathcal{L}_2$) **je** uzavřena vzhledem k operacím

1. sjednocení
2. zřetězení
3. iterace
4. pozitivní iterace
5. průnik s regulárním jazykem

Věta 3.60. Třída bezkontextových jazyků ($\mathcal{L}_2$) **není** uzavřena vzhledem k operacím

1. průnik
2. doplněk

Sjednocení

L_1 je generován CFG $\mathcal{G}_1 = (N_1, \Sigma_1, P_1, S_1)$ a
L_2 je generován CFG $\mathcal{G}_2 = (N_2, \Sigma_2, P_2, S_2)$

Bez újmy na obecnosti můžeme předpokládat $N_1 \cap N_2 = \emptyset$.

Definujeme $\mathcal{G} = (N_1 \cup N_2 \cup \{S\}, \Sigma_1 \cup \Sigma_2, P, S)$,
kde S je nový symbol a

$$P = P_1 \cup P_2 \cup \{S \to S_1, S \to S_2\}$$

Každá derivace v $\mathcal{G}$ začne použitím buď $S \to S_1$ nebo $S \to S_2$.
Podmínka $N_1 \cap N_2 = \emptyset$ zaručí, že při použití $S \to S_1$ (resp. $S \to S_2$)
lze v dalším derivování používat jen pravidla z P_1 (resp. P_2).

Jazyk $L = L_1 \cup L_2$ je generován gramatikou $\mathcal{G}$.

Zřetězení

L_1 je generován CFG $\mathcal{G}_1 = (N_1, \Sigma_1, P_1, S_1)$ a
L_2 je generován CFG $\mathcal{G}_2 = (N_2, \Sigma_2, P_2, S_2)$

Bez újmy na obecnosti můžeme předpokládat $N_1 \cap N_2 = \emptyset$.

Definujeme $\mathcal{G} = (N_1 \cup N_2 \cup \{S\}, \Sigma_1 \cup \Sigma_2, P, S)$,
kde S je je nový symbol a

$$P = P_1 \cup P_2 \cup \{S \to S_1 S_2\}$$

Jazyk $L = L_1.L_2$ je generován gramatikou $\mathcal{G}$.

Iterace a pozitivní iterace

L_1 je generován CFG $\mathcal{G}_1 = (N_1, \Sigma_1, P_1, S_1)$

Definujeme $\mathcal{G} = (N_1 \cup \{S\}, \Sigma_1, P, S)$, kde S je je nový symbol a

$$P = P_1 \cup \{S \to SS_1 \mid \varepsilon\}$$

Jazyk $L = L_1^*$ je generován gramatikou $\mathcal{G}$.

Definujeme $\mathcal{G} = (N_1 \cup \{S\}, \Sigma_1, P, S)$, kde S je je nový symbol a

$$P = P_1 \cup \{S \to SS_1 \mid S_1\}$$

Jazyk $L = L_1^+$ je generován gramatikou $\mathcal{G}$.

Korektnost konstrukce pro iteraci

Dokážeme $L(\mathcal{G}) = L_1^*$.

Průnik a doplněk

$L_1 = \{a^n b^n c^m \mid m, n \geq 1\}$ $\quad$ $L_2 = \{a^m b^n c^m \mid m, n \geq 1\}$

Oba tyto jazyky jsou CFL.

Kbyby $\mathcal{L}_2$ byla uzavřena vzhledem k operaci průniku, pak i $L_1 \cap L_2 = \{a^n b^n c^n \mid n \geq 1\}$ musel být bezkontextový, což však není.

Neuzavřenost $\mathcal{L}_2$ vůči doplňku plyne z její uzavřenosti na sjednocení, neuzavřenosti na průnik a z De Morganových pravidel:

$$L_1 \cap L_2 = \text{co–}(\text{co–}L_1 \cup \text{co–}L_2),$$

tj., kdyby $\mathcal{L}_2$ byla uzavřena na doplněk, musela by být uzavřena i na průnik, což však není.

Průnik s regulárním jazykem

$L = L(\mathcal{P})$, kde $\mathcal{P}$ je PDA $\mathcal{P} = (Q_1, \Sigma, \Gamma, \delta_1, q_1, Z_0, F_1)$
$R = L(\mathcal{A})$, kde $\mathcal{A}$ je deterministický FA $\mathcal{A} = (Q_2, \Sigma, \delta_2, q_2, F_2)$

Sestrojíme PDA $\mathcal{P}'$ takový, že $L(\mathcal{P}')=L \cap R$.

$\mathcal{P}' = (Q, \Sigma, \Gamma, \delta, q_0, Z_0, F)$, kde

- $Q = Q_1 \times Q_2$
- $q_0 = \langle q_1, q_2 \rangle$
- $F = F_1 \times F_2$
- δ : pro každé $p \in Q_1$, $q \in Q_2$, $a \in \Sigma \cup \{\varepsilon\}$, $Z \in \Gamma$ platí:

$$\delta(\langle p, q \rangle, a, Z) = \{(\langle p', q' \rangle, \gamma) \mid (p', \gamma) \in \delta_1(p, a, Z) \text{ a } \widehat{\delta_2}(q, a) = q'\}$$

Zřejmě platí $w \in L(\mathcal{P}') \iff w \in L(\mathcal{P}) \cap L(\mathcal{A})$.

Rozhodnutelné problémy pro bezkontextové jazyky

Problém příslušnosti
Existuje algoritmus, který pro libovolnou danou CFG $\mathcal{G}$ a slovo w rozhoduje, zda $w \in L(\mathcal{G})$ či nikoliv.

Problém prázdnosti
Existuje algoritmus, který pro libovolnou danou CFG $\mathcal{G}$ rozhoduje, zda $L(\mathcal{G}) = \emptyset$ či nikoliv.

Problém konečnosti
Existuje algoritmus, který pro libovolnou danou CFG $\mathcal{G}$ rozhoduje, zda $L(\mathcal{G})$ je konečný či nikoliv.

Konečnost

Věta 3.68. Ke každé CFG $\mathcal{G}$ lze sestrojit čísla m, n taková, že $L(\mathcal{G})$ je nekonečný právě když existuje slovo $z \in L(\mathcal{G})$ takové, že $m < |z| \leq n$.

Důkaz. Předpokládejme, že $\mathcal{G}$ je v CNF.

Nechť p, q jsou čísla s vlastnostmi popsanými v Lemmatu o vkládání. Položme $m = p$ a $n = p + q$.

($\Longleftarrow$) Jestliže $z \in L(\mathcal{G})$ je takové slovo, že $|z| > p$, pak existuje rozdělení $z = uvwxy$ splňující $vx \neq \varepsilon$ a $uv^i wx^i y \in L(\mathcal{G})$ pro všechna $i \geq 0$. Tedy jazyk $L(\mathcal{G})$ obsahuje nekonečně mnoho slov tvaru $uv^i wx^i y$, je tedy nekonečný.

($\Longrightarrow$) Nechť $L(\mathcal{G})$ je nekonečný. Pak obsahuje i nekonečně mnoho slov délky větší než p – tuto množinu slov označme M. Zvolme z M libovolné takové slovo z, které má minimální délku a ukažme, že musí platit $p < |z| \leq p + q$.

Kdyby $|z| > p + q$, pak (opět dle Pumping lemmatu pro CFL) lze z psát ve tvaru $z = uvwxy$, kde $vx \neq \varepsilon, |vwx| \leq q$ a $uv^i wx^i y \in L(\mathcal{G})$ pro všechna $i \geq 0$.

Pro $i = 0$ dostáváme, že $uwy \in L(\mathcal{G})$ a současně $|uwy| < |uvwxy|$.

Z nerovností $|uvwxy| > p + q$ a $|vwx| \leq q$ plyne, že $|uwy| > (p + q) - q = p$. Tedy $uwy \in M$, což je spor s volbou z jako slova z M s minimální délkou. Celkem tedy musí být $|z| \leq p + q$. $\square$

Vlastnost sebevložení

Definice 3.70. Nechť $\mathcal{G} = (N, \Sigma, P, S)$ je CFG. Řekneme, že $\mathcal{G}$ má **vlastnost sebevložení**, jestliže existují $A \in N$ a $u, v \in \Sigma^+$ taková, že $A \Rightarrow^+ uAv$.

CFL L má **vlastnost sebevložení**, jestliže každá bezkontextová gramatika, která jej generuje, má vlastnost sebevložení.

Věta 3.71. CFL L má vlastnost sebevložení, právě když L není regulární.

Důkaz ve skriptech obsahuje závažnou chybu. Kdo mi jako první pošle mail s popisem chyby, získá 1 tvrdý bod. **Deadline: 30. 11. 2013**

Nerozhodnutelné problémy pro bezkontextové jazyky

Problém regularity
Neexistuje algoritmus, který pro libovolnou danou CFG $\mathcal{G}$ rozhoduje, zda $L(\mathcal{G})$ je regulární či nikoliv.
(Tedy není rozhodnutelné, zda $L(\mathcal{G})$ má vlastnost sebevložení či nikoliv.)

Problém univerzality
Neexistuje algoritmus, který pro libovolnou danou CFG $\mathcal{G}$ rozhoduje, zda $L(\mathcal{G}) = \Sigma^*$ či nikoliv.

Problémy ekvivalence a inkluze také nejsou rozhodnutelné (plyne z nerozhodnutelnosti problému univerzality).

Deterministické zásobníkové automaty

Definice 3.72. Řekneme, že PDA $\mathcal{M} = (Q, \Sigma, \Gamma, \delta, q_0, Z_0, F)$ je **deterministický** (DPDA), jestliže jsou splněny tyto podmínky:

1. pro všechna $q \in Q$ a $Z \in \Gamma$ platí: kdykoliv $\delta(q, \varepsilon, Z) \neq \emptyset$, pak $\delta(q, a, Z) = \emptyset$ pro všechna $a \in \Sigma$;

2. pro žádné $q \in Q, Z \in \Gamma$ a $a \in \Sigma \cup \{\varepsilon\}$ neobsahuje $\delta(q, a, Z)$ více než jeden prvek.

Řekneme, že L je **deterministický bezkontextový jazyk** (DCFL), právě když existuje DPDA $\mathcal{M}$ takový, že $L = L(\mathcal{M})$.

Vlastnosti deterministických bezkontextových jazyků

Věta 3.82. Třída DCFL **je** uzavřena na doplňek.

Intuice: DPDA má nad každým slovem právě jeden výpočet. Pro doplňek stačí zaměnit koncové a nekoncové stavy.

Komplikace 1: DPDA nemusí dočíst vstupní slovo do konce, protože se vyprázdní zásobník nebo přechod není definován.

Řešení:

Komplikace 2: DPDA nemusí dočíst vstupní slovo do konce, protože přestane číst vstup a neustále provádí ε-kroky pod kterými zásobník neomezeně roste.

Řešení: $s = |Q| \quad t = |\Gamma|$

$$r = \max\{|\gamma| \mid (p', \gamma) \in \delta(p, \varepsilon, Z),\ p, p' \in Q,\ Z \in \Gamma\}$$

zásobník neomezeně roste při ε-krocích $\iff$ během posloupnosti ε-kroků jeho délka vzroste o více než $r \cdot s \cdot t$

Komplikace 3: DPDA nemusí dočíst vstupní slovo do konce, protože přestane číst vstup a neustále provádí ε-kroky pod kterými zásobník neroste neomezeně, tj. po jistém počtu kroků se jeho obsah opakuje.

Řešení: $s = |Q| \quad t = |\Gamma|$

$$r = \max\{|\gamma| \mid (p', \gamma) \in \delta(p, \varepsilon, Z),\ p, p' \in Q,\ Z \in \Gamma\}$$

Komplikace 4: DPDA dočte slovo, ale pak pod ε-kroky prochází koncové i nekoncové stavy (tj. některá slova jsou akceptována původním DPDA i DPDA se zaměněnými koncovými stavy).

Řešení:

Průnik a sjednocení

Věta. Třída DCFL **není** uzavřena na průnik.

Důkaz. $L_1 = \{a^n b^n c^m \mid m, n \geq 1\}$ a $L_2 = \{a^m b^n c^m \mid m, n \geq 1\}$ jsou DCFL, ale $L_1 \cap L_2 = \{a^n b^n c^n \mid n \geq 1\}$ není ani bezkontextový. □

Věta. Třída DCFL **je** uzavřena na průnik s regulárním jazykem.

Věta. Třída DCFL **není** uzavřena na sjednocení.

Důkaz. Plyne z uzavřenosti na doplňek, neuzavřenosti na průnik a z De Morganových pravidel:

$$L_1 \cap L_2 = \text{co–}(\text{co–}L_1 \cup \text{co–}L_2)$$

(Z uzavřenosti na sjednocení by plynula uzavřenost na průnik.) □

Vztah deterministických a nedeterministických CFL

Věta. Třída DCFL tvoří vlastní podtřídu třídy bezkontextových jazyků. Zejména existují bezkontextové jazyky, které nejsou DCFL.

Příklad. Jazyk co–$\{ww \mid w \in \{a, b\}^*\}$ je CFL, ale není DCFL.

Aplikace (deterministických) bezkontextových jazyků

- syntaxe programovacích jazyků je definována pomocí CFG (dobře uzávorkované výrazy, *if* - *then* - *else* konstrukty)
- DTD (Document Type definition) umožňuje definovat bezkontexové jazyky - využití ve značkovacích jazycích (HTML, XML,. . .)
- nástroje pro tvorbu parserů/překladačů využívají různé algoritmy pro lineární deterministickou syntaktickou analýzu:
  LALR(1) - Yacc, Bison, javacup
  LL(k) - JavaCC, ANTLR

Turingův stroj – syntaxe

Definice. (Deterministický) Turingův stroj (Turing Machine, TM) je devítice $\mathcal{M} = (Q, \Sigma, \Gamma, \triangleright, \sqcup, \delta, q_0, q_{\mathrm{acc}}, q_{\mathrm{rej}})$, kde

- Q je konečná množina, jejíž prvky nazýváme **stavy**,
- Σ je konečná množina, tzv. **vstupní abeceda**,
- Γ je konečná množina, tzv. **pracovní abeceda**, $\Sigma \subseteq \Gamma$,
- $\triangleright \in \Gamma \setminus \Sigma$ je **levá koncová značka**,
- $\sqcup \in \Gamma \setminus \Sigma$ je symbol označující **prázdné políčko**,
- $\delta : \ (Q \setminus \{q_{\mathrm{acc}}, q_{\mathrm{rej}}\}) \times \Gamma \ \to \ Q \times \Gamma \times \{L, R\}$
  je **totální přechodová funkce**,
- $q_0 \in Q$ je **počáteční stav**,
- $q_{\mathrm{acc}} \in Q$ je **akceptující stav**,
- $q_{\mathrm{rej}} \in Q$ je **zamítající stav**.

Dále požadujeme, aby pro každé $q \in Q$ existoval $p \in Q$ takový, že $\delta(q, \rhd) = (p, \rhd, R)$ (tj. $\rhd$ nelze přepsat ani posunout hlavu za okraj pásky).

Označení. $\sqcup^\omega = \sqcup\sqcup\sqcup\sqcup\sqcup\sqcup\ldots$

Definice. Konfigurace Turingova stroje je trojice $(q, z, n) \in Q \times \{y\sqcup^\omega \mid y \in \Gamma^*\} \times \mathbb{N}_0$, kde

- q je stav,
- $y\sqcup^\omega$ je obsah pásky,
- n značí pozici hlavy na pásce.

Počáteční konfigurace pro vstup $w \in \Sigma^*$ je trojice $(q_0, \rhd w\sqcup^\omega, 0)$.
Akceptující konfigurace je každá trojice tvaru (q_{acc}, z, n).
Zamítající konfigurace je každá trojice tvaru (q_{rej}, z, n).

Výpočet Turingova stroje

Označení. Pro libovolný nekonečný řetěz z nad Γ, z_n označuje n-tý symbol řetězu z (z_0 je nejlevější symbol řetězu z). Dále $s_b^n(z)$ označuje řetěz vzniklý ze z nahrazením z_n symbolem b.

Definice. Na množině všech konfigurací stroje $\mathcal{M}$ definujeme binární relaci $\vdash_{\mathcal{M}}$ (**krok výpočtu**) takto:

$$(p, z, n) \quad \vdash_{\mathcal{M}} \quad \begin{cases} (q, s_b^n(z), n+1) & \text{pro } \delta(p, z_n) = (q, b, R) \\ (q, s_b^n(z), n-1) & \text{pro } \delta(p, z_n) = (q, b, L) \end{cases}$$

Výpočet TM $\mathcal{M}$ na vstupu w je maximální (konečná nebo nekonečná) posloupnost konfigurací $K_0, K_1, K_2, \ldots$, kde K_0 je počáteční konfigurace pro w a $K_i \vdash_{\mathcal{M}} K_{i+1}$ pro všechna $i \geq 0$.

Stroj $\mathcal{M}$ **akceptuje** slovo w právě když výpočet $\mathcal{M}$ na w je konečný a jeho poslední konfigurace je akceptující.

Stroj $\mathcal{M}$ **zamítá** slovo w právě když výpočet $\mathcal{M}$ na w je konečný a jeho poslední konfigurace je zamítající.

Stroj $\mathcal{M}$ pro vstup w **cyklí** právě když výpočet $\mathcal{M}$ na w je nekonečný.

Jazyk akceptovaný TM $\mathcal{M}$ definujeme jako

$$L(\mathcal{M}) = \{w \in \Sigma^* \mid \mathcal{M} \text{ akceptuje } w\}.$$

Ukázky Turingových strojů

Simulátor TM: `http://www.fi.muni.cz/~xbarnat/tafj/turing/`

Různé úrovně popisu TM

- formální
- neformální implementační
- vysokoúrovňový

Vícepáskový Turingův stroj

Definice. k-**páskový Turingův stroj** je definován stejně jako TM s vyjímkou přechodové funkce δ, která je definována jako totální funkce

$$\delta : \ (Q \setminus \{q_{\mathrm{acc}}, q_{\mathrm{rej}}\}) \times \Gamma^k \ \rightarrow \ Q \times \Gamma^k \times \{L, R\}^k.$$

Konfigurace mají tvar $(q, z_1, \ldots, z_k, n_1, \ldots, n_k) \in Q \times (\Gamma^*.\{\sqcup^\omega\})^k \times \mathbb{N}_0^k$.
Počáteční konfigurace pro $w \in \Sigma^*$ je $(q_0, \rhd w \sqcup^\omega, \rhd \sqcup^\omega, \ldots, \rhd \sqcup^\omega, 0, \ldots, 0)$.
Definice **akceptující**/**zamítající konfigurace** a $\vdash_{\mathcal{M}}$ se změní podobně.

Ekvivalence vícepáskového a jednopáskového TM

Věta. Pro každý vícepáskový Turingův stroj existuje ekvivalentní (jednopáskový) TM.

Důkaz.

1. Neprázdný obsah k pásek a polohy hlav zapíšeme za sebe na 1 pásku.
2. Simulaca jednoho kroku = zjistit informace pod hlavami, zapsat nové a posunout hlavy.
3. Je-li třeba další políčko nějaké pásky, posuneme zbývající obsah. □

Nedeterministický Turingův stroj

Definice. Nedeterministický Turingův stroj $\mathcal{M}$ je definován stejně jako TM s vyjímkou přechodové funkce δ, která je definována jako totální funkce $\delta : (Q \setminus \{q_{\mathrm{acc}}, q_{\mathrm{rej}}\}) \times \Gamma \rightarrow 2^{Q \times \Gamma \times \{L,R\}}$.

Všechny pojmy se definují stejně jako u deterministického TM. Drobné změny jsou jen u definice kroku výpočtu $\vdash\!\!\!\!-\!\!\!\!-_{\mathcal{M}}$ a akceptace slova.

$$(p, z, n) \vdash\!\!\!\!-\!\!\!\!-_{\mathcal{M}} (q, s_b^n(z), n+1) \quad \text{jestliže} \quad (q, b, R) \in \delta(p, z_n)$$
$$(p, z, n) \vdash\!\!\!\!-\!\!\!\!-_{\mathcal{M}} (q, s_b^n(z), n-1) \quad \text{jestliže} \quad (q, b, L) \in \delta(p, z_n)$$

$\mathcal{M}$ **akceptuje** slovo w, právě když existuje výpočet z počáteční konfigurace pro w do nějaké akceptující konfigurace.

Ekvivalence nedeterministického a deterministického TM

Věta. Pro každý nedeterministický Turingův stroj $\mathcal{N}$ existuje ekvivalentní deterministický TM.

Intuice:

Důkaz. Sestrojíme 3-páskový deterministický TM prozkoumávající výpočtový strom stroje $\mathcal{N}$. Tento 3-páskový stroj lze převést na jednopáskový deterministický TM.

Nechť $k = \max\{|\delta(q, z)| \mid q \in Q \setminus \{q_{\text{acc}}, q_{\text{rej}}\}, z \in \Gamma\}$.

1. páska obsahuje vždy pouze vstup, nemění se.
2. páska slouží k simulaci nedeterministického stroje.
3. páska obsahuje řetězec $\{1, \ldots, k\}^*$ určující, který uzel stromu je právě prohledáván.

Hledáme akceptující konfiguraci ve výpočtovém stromě prohledáním do šířky. Kontrola jednoho uzlu výpočtového stromu:

1. Zkopíruj první pásku na druhou.

2. Na 2. pásce simuluj $\mathcal{N}$, nedeterministické volby řeš podle čísel na 3. pásce. Narazíš-li na akceptující stav, akceptuj. V ostatních případech (příslušná volba neexistuje nebo $\mathcal{N}$ dojde do zamítajícího stavu nebo došly čísla na 3. pásce) pokud pokračuj dalším krokem.

3. Nahraď obsah řetězce na 3. pásce jeho následníkem v lexikografickém uspořádání a začni znovu.

□

Další varianty Turingova stroje

- Turingův stroj s oddělenou vstupní páskou
- Turingův stroj s oboustranně nekonečnou páskou
- Stroj se dvěma zásobníky
- Stroj se vstupní páskou a dvěma počítadly
- . . .

Všechny tyto varianty mají tutéž vyjadřovací sílu.

Churchova teze

Churchova (Church-Turingova) teze: Každý proces, který lze intuitivně nazvat algoritmem, se dá realizovat na Turingově stroji.

Další ekvivalentní formalismy:

- Minského stroje
- λ-kalkul
- while-programy
- ...

Turingovy stroje a třídy jazyků

Věta. Jazyk L je rekursivně spočetný (tj. generovaný gramatikou typu 0) $\iff$ L je akceptovaný nějakým Turingovým strojem.

Definice. Turingův stroj se vstupní abecedou Σ se nazývá **úplný**, je-li každý jeho výpočet konečný (akceptující nebo zamítající).

Jazyk se nazývá **rekursivní**, pokud je akceptovaný nějakým úplným Turingovým strojem.

Přehled jazykových tříd

Jazyky	Gramatiky (typ)	Automaty
rekursivně spočetné	frázové (0)	Turingovy stroje
rekursivní	-	úplné Turingovy stroje
kontextové	kontextové (1)	lineárně ohraničené TM
bezkontextové	bezkontextové (2)	zásobníkové automaty
deterministické CFL	-	deterministické PDA
regulární	regulární (3)	konečné automaty

Třída na nižším rádku je vždy vlastní podtřídou třídy na vyšším řádku.

Problémy jako jazyky

Problém rozhodnout, zda dané w má vlastnost P lze ztotožnit s množinou $\{w \mid w \text{ má vlastnost } P\}$.

Objekty w lze kódovat jako slova $\langle w \rangle$. Problém pak ztotožníme s jazykem $\{\langle w \rangle \mid w \text{ má vlastnost } P\}$.

Příklad. Problém rozhodnout, zda daný konečný graf je souvislý, ztotožníme s jazykem $\{\langle G \rangle \mid G \text{ je konečný souvislý graf}\}$.

Rozhodnutelnost problémů

Definice. Problém P je

- **rozhodnutelný** $\iff$ $\{\langle w \rangle \mid w$ má vlastnost $P\}$ je rekursivní
- **nerozhodnutelný** $\iff$ $\{\langle w \rangle \mid w$ má vlastnost $P\}$ není rekursivní
- **částečně rozhodnutelný (semirozhodnutelný)** $\iff$
  $\iff$ $\{\langle w \rangle \mid w$ má vlastnost $P\}$ je rekursivně spočetný

Problém zastavení (Halting Problem)

Problém zastavení je problém rozhodnout, zda daný TM $\mathcal{M}$ akceptuje dané slovo w nad jeho vstupní abecedou. Problém ztotožníme s jazykem $\{\langle \mathcal{M}, w \rangle \mid M$ je TM, w je slovo nad jeho vstupní abecedou a $\mathcal{M}$ akceptuje $w\}$.

Věta. Problém zastavení je částečně rozhodnutelný.

Důkaz. Stačí dekódovat $\mathcal{M}$ a w a simulovat výpočet $\mathcal{M}$ na w. □

Věta. Problém zastavení je nerozhodnutelný.

Důkaz. Předpokládejme, že existuje TM $\mathcal{H}$ rozhodující problém zastavení. Tedy $\mathcal{H}$ vstupu $\langle \mathcal{M}, w \rangle$ akceptuje, právě když pokud $\mathcal{M}$ akceptuje w.

S využitím $\mathcal{H}$ zkonstruujeme TM $\mathcal{D}$: dostane-li $\mathcal{D}$ na vstupu zakódovaný stroj $\langle \mathcal{M} \rangle$, zeptá se stroje $\mathcal{H}$, zda $\mathcal{M}$ akceptuje svůj vlastní kód $\langle \mathcal{M} \rangle$ a následně odpověd otočí. Tedy

$\mathcal{D}$ akceptuje $\langle \mathcal{M} \rangle$, pokud $\mathcal{M}$ neakceptuje $\langle \mathcal{M} \rangle$ a
$\mathcal{D}$ neakceptuje $\langle \mathcal{M} \rangle$, pokud $\mathcal{M}$ akceptuje $\langle \mathcal{M} \rangle$.

Nyní spustíme $\mathcal{D}$ na vstupu $\langle \mathcal{D} \rangle$:

$\mathcal{D}$ akceptuje $\langle \mathcal{D} \rangle$, pokud $\mathcal{D}$ neakceptuje $\langle \mathcal{D} \rangle$ a
$\mathcal{D}$ neakceptuje $\langle \mathcal{D} \rangle$, pokud $\mathcal{D}$ akceptuje $\langle \mathcal{D} \rangle$.

To je spor. Stroj $\mathcal{H}$ tedy neexistuje a problém zastavení je nerozhodnutelný. □