Kapitola 4

Turingovy stroje

Hlavním cílem celéno kurzu bylo seznámit se s různými konečnými popisy konečných a hlavně nekonečných množin slov. Už jsem probrali konečné automaty, které přijímají právě regulární jazyky; pak jsme se věnovali bezkontextovým jazykům, pro které existují zásobníkové automaty, které je příjímají. Víme ale, že gramatiky generují nejen bezkontextové jazyky, ale též kontextové a také jazyky typu 0 v chomského hiearchii, viz **??**. A Turingovy stroje jsou právě „stroje“, které přijímají právě jazyky typu 0.

4.1 Turingův stroj

Nejprve uvedeme Turingův stroj neformálně a teprve potom uvedeme jeho formální definici.

Turingův stroj si můžeme představit takto: skládá se

- z řídící jednotky, která se může nacházet v jednom z konečně mnoha stavů,
- potenciálně nekonečné pásky rozdělené na jednotlivá pole a
- hlavy, která umožňuje číst obsah polí a přepisovat obsah polí pásky.

Na základě informace X, která je přečtena na pásce, a na základě stavu q, ve kterém se nachází řídící jednotka Turingova stroje, se řídící jednotka přesune do stavu p, pole pásky přepíše na Y a hlava se přesune buď doprava nebo doleva, nebo nemůže pokračovat a zastaví svou práci (tato akce je popsána tzv. přechodovou funkcí).

4.1.1 Formální definice. Turingův stroj je sedmice $(Q, \Sigma, \Gamma, \delta, q_0, B, F)$, kde

- Q je konečná množina stavů,
- Σ je konečná neprázdná množina vstupních symbolů,
- Γ je konečná množina páskových symbolů, přitom $\Sigma \subset \Gamma$,
- B je prázdný symbol (též nazývaný *blank*), jedná se o páskový symbol, který není vstupním symbolem, (tj. $B \in \Gamma \setminus \Sigma$),
- δ je přechodová funkce, tj. parciální zobrazení z množiny $(Q \setminus F) \times \Gamma$ do množiny $Q \times \Gamma \times \{L, R\}$, (zde L znamená pohyb hlavy o jedno pole doleva, R znamená pohyb hlavy o jedno pole doprava),
- $q_0 \in Q$ je počáteční stav a
- $F \subseteq Q$ je množina koncových stavů.

4.1.2 Konfigurace. Konfiguraci Turingova stroje plně popisuje obsah pásky, pozice hlavy na pásce a stav, ve kterém se nachází řídící jednotka. Jestliže na pásce jsou v k polích symboly $X_1\, X_2\, \ldots\, X_k$, všechna pole s větším i menším číslem již obsahují pouze B, řídící jednotka je ve stavu q a hlava čte symbol X_i, tak danou konfiguraci zapisujeme

$$X_1\, X_2\, \ldots\, X_{i-1}\, q\, X_i\, X_{i+1}\, \ldots\, X_k.$$

4.1.3 Počátek práce Turingova stroje. Na začátku práce se Turingův stroj nachází v počátečním stavu q_0, na pásce má na n polích vstupní slovo $a_1\,a_2\,\ldots\,a_n$ ($a_i \in \Sigma$), ostatní pole obsahují blank B a hlava čte symbol a_1. Tedy formálně je počáteční konfigurace $q_0\,a_1\,\ldots\,a_n$.

4.1.4 Krok Turingova stroje. Předpokládejme, že se Turingův stroj nachází v konfiguraci $X_1\,X_2\,\ldots\,X_{i-1}\,q\,X_i\,\ldots\,X_k$. Pak v jednom kroku udělá následující:

Jestliže $\delta(q, X_i) = (p, Y, R)$, stroj se přesune do stavu p, na pásku napíše symbol Y (místo X_i) a hlavu posune o jedno pole doprava. Formálně to zapisujeme:

$$X_1\,X_2\,\ldots\,X_{i-1}\,q\,X_i\,\ldots\,X_k \vdash X_1\,X_2\,\ldots\,X_{i-1}\,Y\,p\,X_{i+1}\,\ldots\,X_k.$$

Jestliže je $i = k$ a Turingův stroj se má posunout hlavu doprava, pak

$$X_1\,X_2\,\ldots\,q\,X_k \vdash X_1\,X_2\,\ldots\,X_{k-1}\,Y\,p\,B.$$

Jestliže $\delta(q, X_i) = (p, Y, L)$, stroj napíše na pásku Y (místo X_i) a posune hlavu o jedno pole doleva. Formálně to zapisujeme:

$$X_1\,X_2\,\ldots\,X_{i-1}\,q\,X_i\,\ldots\,X_k \vdash X_1\,X_2\,\ldots\,X_{i-2}\,p\,X_{i-1}\,Y\,X_{i+1}\,\ldots\,X_k.$$

Jestliže je $i = 1$ a Turingův stroj se má posunout hlavu doleva, pak

$$q\,X_1\,X_2\,\ldots\,X_k \vdash p\,B\,Y\,X_2\,\ldots\,X_k.$$

4.1.5 Výpočet Turingova stroje je posloupnost jeho kroků. Tedy jedná se o reflexivní a tranzitivníuzávěr $\vdash^\star$ relace $\vdash$ (na množině všech konfigurací daného Turingova stroje). Práce nad slovem je výpočet, který začíná v počáteční konfiguraci.

Říkáme, že se *Turingův stroj zastavil* v případě, kdy nemůže udělat další krok.

Poznámka. Je zřejmé, že se Turingův stroj nemusí vždy nad vstupem zastavit. Může se zacyklit, ale také může pracovat do nekonečna, aniž by se zacyklil – např. může stále na konec použité pásky připisovat symbol. Na druhé straně se v koncovém stavu vždy zastaví. V takovém případě říkáme, že se Turingův stroj zastaví *úspěšně*, zastaví-li se v nekoncovém stavu, říkáme, že se zastavil *neúspěšně*.

4.1.6 Jazyk přijímaný Turingovým strojem. Vstupní slovo $w \in \Sigma^\star$ je *přijato* Turingovým strojem právě tehdy, když se Turingův stroj při práci na slově w dostane do koncového stavu. Tedy formálně: slovo $w \in \Sigma^\star$ je *přijato Turingovým strojem* právě tehdy, když

$$q_0\,w \vdash^\star \alpha q \beta \;\text{ pro nějaké }\; q \in F \text{ a } \alpha, \beta \in \Gamma^\star.$$

Množina slov $w \in \Sigma^\star$, které Turingův stroj přijímá, se nazývá *jazyk přijímaný* Turingovým strojem M a značíme ho $L(M)$. Jestliže se Turingův stroj M na každém vstupu zastaví, ať již úspěšně nebo neúspěšně, říkáme, že jazyk $L(M)$ *rozhoduje*.

4.1.7 Poznámka. Existuje několik základních modelů Turingova stroje. My jsme si vybrali ten, který má pásku nekonečnou „na obě strany“, posouvá hlavu vždy o jedno políčko a v koncovém stavu již nemůže provádět další krok.

Jiné varianty jsou:

- pásku s pevným levým okrajem. Pak je na začátku práce Turingova stroje vstupní slovo napsáno vždy na prvních k polích pásky a hlava čte první pole pásky. V tomto modelu je trochu obtížnější návrh strojů, protože je potřeba dát pozor, aby Turingův stroj nepřekročil levý okraj — pak by se vždy zastavil ať již úspěšně nebo neúspěšně.
- Turingův stroj může kromě posunu hlavy doprava nebo doleva nechat hlavu stát. To znamená, že po provedení kroku čte hlava stejné pole jako v předchozím kroku.
- Přechodová funkce může být definována i pro koncový stav a některý páskový symbol.

Všechny tyto modifikace ale neznamenají žádný rozdíl v tom, jaké jazyky jsou Turingovým strojem přijaty.

4.1.8 Věta. Jestliže jazyk L je přijímán Turingovým strojem, pak k němu existuje gramatika typu 0, která ho generuje.

Ke každé gramatice $\mathcal{G}$ typu 0 existuje Turingův stroj, který přijímá jazyk $L(\mathcal{G})$. □

Důkaz této věty neuvádíme.

4.1.9 Poznámka Předchozí věta nám říká, že Turingovy stroje jsou „výpočetním modelem" pro jazyky typu 0, tj. jazyky generované tou nejobecnější gramatikou. Již víme, že „výpočetním modelem" regulárních jazyků jsou konečné automaty, „výpočetním modelem" bezkontextových jazyků pak zásobníkové automaty.

Obdobný model pro kontextové jazyky (tj. jazyky typu 1) jsou lineárně omezené automaty. Jedná se v podstatě o Turingův stroj, kde ovšem je možné používat pouze pole pásky, ve kterých se nacházelo vstupní slovo. Jakékoli překročení ať doprava nebo doleva znamená neúspěšné zastavení.

4.2 Nerozhodnutelnost

V této sekci si ukážeme, že existují jazyky / úlohy, pro které neexistuje Turingův stroj, který by je rozhodoval (a tudíž ani algoritmus, který by je řešil). Tato část textu je značně neformální a neobsahuje potřebná zdůvodnění zde uvedených tvrzení – podrobnější zdůvodnění je nad rámec předmětu a studenti se s ním seznámí v předmětu Teorie algoritmů.

4.2.1 Rozhodovací úlohy a jazyky Jistě jste se již setkali s úlohami, kde „řešení" je odpověď „ano" nebo „ne". Jako příklad můžeme uvést třeba splnitelnost booleovských formulí – ano, formule je buď splnitelná nebo nesplnitelná. Jiný příklad může být úloha rozhodnout, zda daný graf obsahuje hamiltonovskou kružnici. Ke každé takové úloze můžeme přiřadit jazyk nad vhodnou abecedou. Uděláme to např. takto: každou formuli (graf) zakódujeme jako slovo nad vhodnou abecedou a do jazyk úlohy obsahuje ta slova, která odpovídají formulím (grafům), kde odpověď je „ano".

Churchova teze zhruba říká, že vše, co je možné spočítat algoritmem, je přesně to, co může udělat Turingův stroj.

4.2.2 Definice. Jazykům (úlohám), pro které neexistuje Turingův stroj, který by je rozhodl, (algoritmus, který by je vyřešil) říkáme, že jsou *nerozhodnutelné algoritmicky neřešitelné*).

4.2.3 Věta. Problém zastavení Turingova stroje (tzv. halting problem) je algoritmicky neřešitelný; tj. jeho jazyk je nerozhodnutelný. □

Důkaz je založen na několika konstrukcích/pozorování:

- Nejprve se zkonstruuje univerzální Turingův stroj, který umí simulovat práci libovolného Turingova stroje nad libovolným slovem.
- Kdyby tento univerzální Turingův stroj uměl rozhodnout, zda se Turingův stroj zastaví nad nějakým vstupem, tak bychom také měli Turingův stroj M, který by rozhodl, že se Turingův stroj nad slovem **nezastaví**.
- Kdybychom nyní pustili M na vstup, který je kódem M, dostaneme spor — M by se měl zastavit právě tehdy, když se nezastaví.

4.2.4 Může se zdát, že problém zastavení Turingova stroje není zrovna problém, který by byl v praxi užitečný. To je sice pravda, ale na základě algoritmické neřešitelnosti jednoho rozhodovacího problému (nerozhodnutelnosti jednoho jazyka) je možné ukázat velkou řadu dalších problémů, které jsou také algoritmicky neřešitelné. Naším cílem je ukázat, že některé „rozumné" otázky týkající se bezkontextových gramatik, jsou algoritmicky neřešitelné. K tomu využijeme fakt, že následující úloha PCP je algoritmicky neřešitelná.

4.2.5 Postův korespondenční problém (PCP). Jsou dány dva seznamy neprázdných slov A, B nad danou abecedou Σ.

$$A = (w_1, w_2, \ldots, w_k), \quad B = (x_1, x_2, \ldots, x_k),$$

kde $w_i, x_i \in \Sigma^+$, $i = 1, 2, \ldots, k$. Řekneme, že dvojice A, B *má řešení*, jestliže existuje posloupnost $i_1, i_2, \ldots, i_r$ indexů, tj $i_j \in \{1, 2, \ldots, k\}$, taková, že

$$w_{i_1}\, w_{i_2}\, \ldots\, w_{i_r} \;=\; x_{i_1}\, x_{i_2}\, \ldots\, x_{i_r}.$$

Otázka: Má daná instance řešení?

4.2.6 Příklady.

1. Jsou dány seznamy

	1	2	3	4	5
A	011	0	101	1010	010
B	1101	00	01	00	0

   Tato instance má řešení, např. $2, 1, 5$ je

$$w_2\, w_1\, w_5 = 0011010 = x_2\, x_1\, x_1\, x_4\, x_1\, x_5.$$

   Dalšími řešeními jsou např. $2, 1, 1, 3, 5$ nebo $2, 1, 1, 4, 1, 5$.

2. Jsou dány seznamy

	1	2	3	4	5
A	11	0	101	1010	010
B	101	00	01	00	0

   Tato instance nemá řešení.

4.2.7 Věta. Kdyby byl algoritmicky řešitelný Postův korespondenční problém, byl by algoritmicky řešitelný i problém zastavení Turingova stroje. □

Větu nedokazujeme.

PCP se dá dále využít k důkazům, že následující úlohy týkající se bezkontextových gramatik jsou také algoritmicky neřešitelné. Zdůvodnění faktu, že úlohy 1 a 3 z následujícího seznamu jsou algoritmicky neřešitelné si ukážeme na přednáše.

4.2.8 Další algoritmicky neřešitelné úlohy:

1. Pro dané bezkontextové gramatiky $\mathcal{G}_1$ a $\mathcal{G}_2$ rozhodnout, zda obě generují aspoň jedno stejné slovo, tj. zda $L(\mathcal{G}_1) \cap L(\mathcal{G}_2) \neq \emptyset$.
2. Pro dané bezkontextové gramatiky $\mathcal{G}_1$ a $\mathcal{G}_2$ rozhodnout, zda přijímají stejný jazyk, tj. zda $L(\mathcal{G}_1) = L(\mathcal{G}_2)$.
3. Pro danou bezkontextovou gramatiku $\mathcal{G}_1$ rozhodnout, zda je víceznačná.
4. Pro danou bezkontextovou gramatiku $\mathcal{G}_1$ rozhodnout, zda přijímá všechna slova, tj. zda $L(\mathcal{G}_1) = \Sigma^\star$.
5. Pro danou bezkontextovou gramatiku $\mathcal{G}_1$ a regulární jazyk R rozhodnout, zda $R \subseteq L(\mathcal{G}_1)$.

4.2.9 Připomeňme, že jsou algoritmicky řešitelné následující úlohy:

1. Zda daná bezkontextová gramatika $\mathcal{G}$ generuje alespoň jedno slovo; tj. zda $L(\mathcal{G}) \neq \emptyset$.
2. Pro danou bezkontextovou gramatiku $\mathcal{G}$ a dané slovo w rozhodnout, zda $\mathcal{G}$ generuje w, tj. zda $w \in L(\mathcal{G})$.
3. Pro danou bezkontextovou gramatiku $\mathcal{G}$ a regulární jazyk R rozhodnout, zda $L(\mathcal{G}) \subseteq R$.