17 Rekurzivní a r.e. jazyky

Definice. Jazyk nazveme:

- *rekurzivně spočetný*, jestliže je akceptován nějakým Turingovým strojem.
- *rekurzivní*, jestliže je akceptován nějakým úplným Turingovým strojem (tj. strojem, který na libovolném vstupu v konečném čase skončí výpočet).

Věta 17.1. Třídy rekurzivních i rekurzivně spočetných jazyků jsou uzavřeny na sjednocení.

Důkaz. Mějme rekurzivně spočetné jazyky L_1 a L_2 a Turingovy stroje $\mathcal{M}_1$ a $\mathcal{M}_2$ pro jejich výpočet. Turingův stroj $\mathcal{M}$ pro jazyk $L_1 \cup L_2$ nedeterministicky zvolí, který ze strojů $\mathcal{M}_1$ nebo $\mathcal{M}_2$ bude simulovat. Jakmile tento stroj akceptuje nebo zamítá, také on akceptuje nebo zamítá. Pokud jsou jazyky L_1 a L_2 rekurzivní a stroje $\mathcal{M}_1$ a $\mathcal{M}_2$ úplné, pak i stroj $\mathcal{M}$ je úplný a jazyk $L_1 \cup L_2$ rekurzivní. □

Věta 17.2. Třídy rekurzivních a rekurzivně spočetných jazyků jsou uzavřeny na průnik.

Důkaz. Mějme rekurzivně spočetné jazyky L_1 a L_2 a stroje $\mathcal{M}_1$ a $\mathcal{M}_2$ pro jejich rozpoznání. Stroj $\mathcal{M}$ rozpoznávající jazyk $L_1 \cap L_2$ bude pracovat následovně: Nejprve na vstupu simuluje $\mathcal{M}_1$. Jestliže tento stroj zamítá, pak rovněž zamítá, jestliže akceptuje, pak na původním slově simuluje stroj $\mathcal{M}_2$. Jestliže tento stroj akceptuje, pak rovněž akceptuje, jinak zamítá. Stroj $\mathcal{M}$ bude zřejmě rozpoznávat jazyk $L_1 \cap L_2$, čímž je tvrzení dokázáno pro rekurzivně spočetné jazyky. Pokud jsou stroje $\mathcal{M}_1$ a $\mathcal{M}_2$ úplné, je zřejmě i stroj $\mathcal{M}$ úplný. Tím tvrzení platí i pro třídu rekurzivních jazyků. □

Věta 17.3. Třídy rekurzivních i rekurzivně spočetných jazyků jsou uzavřeny na zřetězení.

Důkaz. Mějme stroje $\mathcal{M}_1$ a $\mathcal{M}_2$ rozpoznávající postupně jazyky L_1 a L_2. Stroj $\mathcal{M}$ rozpoznávající $L_1 \cdot L_2$ nedeterministicky opíše předponu slova w ze vstupu na první pomocnou pásku a zbytek na druhou pomocnou pásku. Posléze simuluje činnost stroje $\mathcal{M}_1$ na první pomocné pásce a stroje $\mathcal{M}_2$ na druhé. Pokud oba akceptují, pak rovněž akceptuje. Pokud byl vstup zřetězením slova z prvního jazyka a slova z druhého jazyka, pak existuje jeho rozdělení, při kterých oba simulované stroje akceptují a existuje akceptující výpočet. Tím je tvrzení dokázáno pro rekurzivně spočetné jazyky. Pokud jsou stroje $\mathcal{M}_1$ a $\mathcal{M}_2$ úplné, je i stroj $\mathcal{M}$ úplný, čímž je tvrzení dokázáno pro rekurzivní jazyky. □

Věta 17.4. Třídy rekurzivních i rekurzivně spočetných jazyků jsou uzavřeny na iteraci.

Důkaz. Mějme stroj $\mathcal{M}$ rozpoznávající jazyk L. Stroj $\mathcal{N}$ rozpoznávající jazyk L^* bude pracovat následovně: Nejprve ověří, zda je na vstupu prázdné slovo. Pokud ano, akceptuje. V opačném případě nedeterministicky zvolí předponu slova na vstupu a opíše ji na druhou pásku. Na té simuluje činnost stroje $\mathcal{M}$. Jestliže tento stroj akceptuje, nedeterministicky opíše další část vstupu na druhou pásku a celý proces opakuje, dokud neprojde celý vstup. Důležité je, že se hlava na vstupní pásce pohybuje pouze do prava. Jestliže vstupní slovo patří do jazyka L^+, musí na něm existovat alespoň jeden akceptující výpočet stroje $\mathcal{N}$. Pokud je navíc stroj $\mathcal{M}$ úplný, je i stroj $\mathcal{N}$ úplný. □

Věta 17.5. Třída rekurzivních jazyků je uzavřena na operaci doplněk.

Důkaz. V úplném Turingově stroji stačí zaměnit akceptující a zamítající stav. □

• Zde je na místě malé srovnání. Situace s úplným Turingovým strojem je podobná jako s konečnými automaty. Máme zde jistotu, že proces výpočtu vždy skončí a to buď v zamítajícím nebo akceptujícím stavu. S nedeterministickým strojem bychom předchozí konstrukci provést nemohli. Využíváme tu silně vlastnosti, že nedeterministický stroj lze vždy převést na deterministický. V případě zásobníkových automatů jsme třeba tento luxus neměli, což vyústilo dokonce v neuzavřenost třídy bezkontextových jazyků na doplněk.

• Dále je třeba poznamenat, že uzavřenost třídy rekurzivních jazyků na doplněk má v tomto případě určitou těsnou vlastnost. Rekurzivní jazyky jsou právě ty rerkuzivně spočetné jazyky, jejichž doplněk zůstává ve třídě rekurzivně spočetných jazyků. Přesný výraz je obsahem následující věty:

Věta 17.6. Je-li L i L^C rekurzivně spočetný jazyk, pak je L rekurzivní.

Důkaz. Mějme stroj $\mathcal{M}_1$ pro jazyk L a stroj $\mathcal{M}_2$ pro jazyk L^C. Chceme zkonstruovat úplný stroj pro jazyk L. To provedeme pomocí paralelní kompozice těchto automatů. Na začátku vstup w opíšeme na dvě pomocné pásky, na jedné budeme simulovat činnost stroje $\mathcal{M}_1$ a na druhé činnost stroje $\mathcal{M}_2$. Tuto simulaci budeme provádět strategií vždy krok jednoho automatu následovaný krokem druhého automatu. Nemůžeme totiž dovolit, aby se nám některý z nich zacyklil. Nám stačí to, že se nemohou zacyklit oba naráz, protože alespoň jeden z nich musí slovo w akceptovat. Podle toho také akceptujeme nebo zamítáme. □

• Celkem jsme dokázali následující *Postovu větu*: Jazyk je rekurzivní právě tehdy, když jsou on i jeho doplněk rekurzivně spočené.

• Důsledkem Postovy věty je fakt, že třída rekurzivně spočetných jazyků je uzavřena na doplněk právě tehdy, když každý rekurzivně spočetný jazyk je rekurzivní. Pokud tedy nalezneme alespoň jeden rekurzivně spočetný jazyk, který není rekurzivní, máme zároveň příklad jazyka, totiž jeho doplňku, který není ani rekurzivně spočetný. Důležitým předělem v dějinách teoretické informatiky bylo právě nalezení takového jazyka Alanem Turingem.

• Pro konstrukci jazyků, které nejsou rekurzivně spočetné, je potřeba kódovat Turingovy stroje do nějaké textové podoby. Dá se sestrojit jednoznačné zakódování již nad dvoupísmennou abecedou. Budeme se tedy k Turingovým strojům chovat, jako by to byly textové řetězce.

Věta 17.7. Následující jazyk je rekurzivně spočetný, ale není rekurzivní:

$$\text{ACCEPT} = \{\mathcal{M}\#w \mid \text{stroj } \mathcal{M} \text{ akceptuje } w\}$$

Důkaz. Snadno se ukáže, že L je rekurzivně spočetný. Turingův stroj pro tento jazyk dostane na vstupu $\mathcal{M}\#w$, opíše w na pomocnou pásku a simuluje na ní činnost stroje $\mathcal{M}$. Pokud sroj $\mathcal{M}$ akceptuje, pak akceptuje, pokud $\mathcal{M}$ zamítá, pak rovněž zamítá, ve zbylých případech simulovaný i simulující stroj cyklí. Takový stroj zřejmě rozpoznává L. Daleko těžší je dokázat, že jazyk L není rekurzivní. K tomu se používá Cantorův diagonalizační argument. Protože všechny

Turingovy stroje umíme jednoznačně reprezentovat řetězcem písmen, umíme je také postupně očíslovat přirozenými čísly. Totéž umíme udělat se vstupy w. Vytvořme nekonečnou tabulku, jejíž řádky budou odpovídat všem Turingovým strojům a sloupce všem vstupům. Pokud bychom uměli sestrojit úplný Turingův stroj pro jazyk L, uměli bychom také sestrojit stroj, který pro každé $n \in \mathbb{N}$ vstup w_n akceptuje, jestliže stroj $\mathcal{M}_n$ zamítá nebo cyklí, a zamítá, jestliže stroj $\mathcal{M}_n$ vstup w_n akceptuje, tj. chová se na diagonále tabulky jinak než všechny ostatní Turingovy stroje a tedy sám není v tabulce. To je ale spor s tím, že jsou v tabulce všechny Turingovy stroje. Jazyk L tedy nemůže být rekurzivní. □

Definice. Problém příslušnosti do jazyka L z předchozí věty nazýváme *problém akceptování* a Turingův stroj pro tento jazyk *univerzální Turingův stroj*.

- Ujasněme si základní metaforu poslední teorie: Turingovy stroje jsou vlastně algoritmy (je to naše metoda, jak nějaký algoritmus definovat naprosto formálně). Potom textová reprezentace Turingova stroje je totéž, co přepis algoritmu do nějakého konkrétního programovacího jazyka. V tom případě není univerzální Turingův stroj nic jiného než interpret tohoto programovacího jazyka.
- Podle Postovy věty není doplněk jazyka L ani rekurzivně spočetný. Argumentaci z předchozí věty již nebudeme pro další jazyky používat a ani v praxi se nepoužívá. Pro posouzení, kam jazyky v hierarchii: rekuzivní, rekurzivně spočetný, ani r.e. patří, budeme používat srovnání právě s jazykem L nebo L^C. Tomu se věnujeme v kapitole o redukci.