18 Redukce

Definice. Problém nazveme:

- *rozhodnutelný*, jestliže jeho jazyková reprezentace je rekurzivní jazyk.
- *částečně rozhodnutelný*, jestliže jeho jazyková reprezentace je rekurzivně spočetný jazyk.
- *nerozhodnutelný*, jestliže není rozhodnutelný.

Cíl. Naším cílem je zavést formalizmus, pomocí něhož bychom mohli porovnávat obtížnost problémů. To znamená, že hledáme relaci na třídě všech problémů, která je alespoň reflexivní a tranzitivní.

Motivace. Mějme problém P_1 určit, zda délka slova nad abecedou $\Sigma = \{a, b\}$ je dělitelná 13, a předpokládejme, že máme algoritmický postup (v tomto případě stačí dokonce konečný automat) na problém P_2 určit, zda délka slova nad Σ je dělitelná 26. K vyřešení P_1 chceme využít P_2, čímž se vyhneme vymyšlení nového algoritmu. Cíl je tedy převést instanci w_1 problému P_1 na instanci w_2 problému P_2. To lze udělat např. tak, že položíme $w_2 = w_1 w_1$. Zřejmě w_2 je dělitelné 26 právě tehdy, když w_1 bylo dělitelné 13. Po aplikaci algoritmu pro P_2 na w_2 získáme rozhodnutí o instanci w_1 pro P_1.

Definice. Mějme funkci $f\colon \Sigma^* \to \Gamma^*$, kde Σ a Γ jsou nějaké abecedy. Řekneme, že f je *totálně vyčíslitelná*, jestliže existuje úplný Turingův stroj, který po ukončení výpočtu na vstupu w má na pásce řetězec $f(w)$. Mějme nyní nějaké jazyky $A \subseteq \Sigma^*$ a $B \subseteq \Gamma^*$. Řekneme, že A se *redukuje* na B, píšeme $A \leq B$ (zkracuji zde běžné značení $\leq_m$), jestliže existuje totálně vyčíslitelná funkce $f\colon \Sigma^* \to \Gamma^*$ splňující:

$$w \in A \Leftrightarrow f(w) \in B$$

Píšeme $A \equiv B$, pokud $A \leq B$ a zároveň $B \leq A$.

Úmluva. Připomeňme, že symbolem $\langle \mathcal{M} \rangle$ označujeme nějaké zakódování Turingova stroje $\mathcal{M}$ jako řetězce nad zvolenou abecedou (toto umíme jednoznačně provést už pro libovolnou dvouprvkovou abecedu). Tento proces potřebujeme pro to, abychom mohli jazykově reprezentovat problémy a vlastnosti Turingových strojů.

Příklad 1. Mějme *problém zastavení* definovaný jako problém příslušnosti do jazyka:

$$\mathrm{HALT} = \{\langle \mathcal{M} \rangle \#w \mid \text{stroj } \mathcal{M} \text{ zastaví na vstupu } w\}$$

Ukážeme, že problém zastavení a problém akceptování jsou ekvivalentní:

$$\mathrm{ACCEPT} \equiv \mathrm{HALT}$$

K tomu musíme provést dvě redukce:

ACCEPT $\leq$ HALT: Nejprve si přeložme, co vlastně znamená provést tuto redukci: Naším úkolem je nalézt postup, jak z řetězce $\mathcal{M}\#w$ udělat řetězec $\mathcal{M}'\#w'$ takovým způsobem, že $\mathcal{M}$ akceptuje w právě tehdy, když $\mathcal{M}'$ zastaví na vstupu w'. To můžeme provést třeba následujícím způsobem: Stroj $\mathcal{M}'$ vznikne ze stroje

$\mathcal{M}$ tak, že z něj odstraníme všechny přechody do zamítajícího stavu. Přitom ponecháme $w = w'$. Potom zřejmě $\mathcal{M}'$ zastaví na vstupu w tehdy a jen tehdy, pokud jej akceptuje, tj. pokud jej akceptoval již automat $\mathcal{M}$. Tento postup přitom zřejmě můžeme provést algoritmicky, čili tato redukce je totálně vyčíslitelná, jak se v definici požaduje.

HALT $\leq$ ACCEPT: V opačném směru chceme z řetězce $\mathcal{M}\#w$ udělat řetězec $\mathcal{M}'\#w'$ takový, že stroj $\mathcal{M}$ zastaví na vstupu w právě tehdy, když stroj $\mathcal{M}'$ akceptuje vstup w'. Opět nebude potřeba měnit vstup, čili položíme $w' = w$. Stroj $\mathcal{M}'$ získáme ze stroje $\mathcal{M}$ tak, že v něm všechny přechody vedoucí do zamítajícího stavu zavedeme do akceptujícího. Tímto způsobem bude stroj $\mathcal{M}'$ akceptovat právě všechny vstupy, které stroj $\mathcal{M}$ akceptovat nebo zamítal, tedy na nichž svůj výpočet zastavil v konečném čase.

Věta 18.1. Důležité vlastnosti redukce:

1. Relace $\leq$ je reflexivní a transitivní.
2. Je-li B rekurzivní a $A \leq B$, pak A je rekurzivní.
3. Je-li B rekurzivně spočetný a $A \leq B$, pak A je rekurzivně spočetný.
4. Pokud A není rekurzivní a $A \leq B$, pak B není rekurzivní.
5. Pokud A není rekurzivně spočetný a $A \leq B$, pak B není rekurzivně spočetný.

Příklad 2. Problém zastavení je pouze částečně rozhodnutelný, ale není rozhodnutelný. Snadno se ukáže, že tento problém je částečně rozhodnutelný tak, že pro něj sestrojíme Turingův stroj. Tento stroj bude fungovat podobně jako univerzální Turingův stroj tím, že bude simulovat činnost stroje na vstupu pouze s tím rozdílem, že akceptujeme, jestliže simulovaný stroj akceptuje nebo zamítá. V ostatních případech simulovaný i simulující stroj cyklí. Částečná rozhodnutelnost problému zastavení také plyne z výše dokázané redukce HALT $\leq$ ACCEPT. Z opačné redukce ACCEPT $\leq$ HALT zase plyne, že problém zastavení je nerozhodnutelný.

Poznámka. Předchozí příklad ilustruje dva postupy, které budeme používat k důkazu, že je nějaký jazyk A pouze rekurzivně spočetný:

1. Buď nalezneme ekvivalenci $A \equiv R$, kde R je jiný pouze rekurzivně spočetný jazyk,
2. nebo sestrojíme Turingův stroj rozpoznávající A a nalezneme redukci $N \leq A$, kde N je nějaký nerekurzivní jazyk.

Příklad 3. *Problém neprázdnosti* definovaný jako problém příslušnosti do jazyka:

$$\text{NONEMPTY} = \{\langle\mathcal{M}\rangle \mid \mathcal{L}(\mathcal{M}) \neq \emptyset\}$$

je částečně rozhodnutelný, ale není rozhodnutelný. Turingův stroj pro tento jazyk bude pracovat následovně: Dostaneme na vstup stroj $\mathcal{M}$. Nedeterministicky na pomocnou pásku zapíšeme slovo a simulujeme činnost $\mathcal{M}$ na pomocné pásce,

akceptujeme právě tehdy, když akceptuje stroj $\mathcal{M}$. Takto zřejmě budeme akceptovat právě ty stroje, které akceptují alespoň jedno slovo. Nyní sestrojíme redukci:

$$\text{ACCEPT} \leq \text{NONEMPTY}$$

Mějme řetězec $\mathcal{M}\#w$. Sestrojme $\mathcal{N}$, který na začátku provede kontrolu, zda má na vstupu w. Pokud ne, pak automaticky zamítá. V opačném případě pokračuje ve výpočtu jako stroj $\mathcal{M}$. Je zřejmé, že stroj $\mathcal{M}$ akceptuje vstup w právě tehdy, když $\mathcal{N}$ rozpoznává neprázdný jazyk. Tím je naše redukce hotová. Protože problém akceptování není rozhodnutelný, není rozhodnutelný ani problém neprázdnosti.

Příklad 4. Problém určit, zda Turingův stroj akceptuje alespoň 2000 slov (označme příslušný jazyk TS2000), je částečně rozhodnutelný, ale není rozhodnutelný. Turingův stroj pro tento jazyk by pracoval následovně: Dostane na vstup stroj $\mathcal{M}$. Paralelně by procházel všechny možné vstupy a simuloval na nich činnost stroje $\mathcal{M}$. To může dělat třeba tak, že provede jeden krok výpočtu na prázdném slově, potom jeden krok výpočtu na všech slovech délky jedna, posléze se vrátí a provede další krok výpočtu na prázdném slově, poté na slovech délky jedna a poté na slovech délky dva a znovu od prázdného slova atd. Takto zaručíme, že simulaci stroje $\mathcal{M}$ na každém možném vstupu, aniž bychom mu dovolili se na některém z nich zacyklit. Zvlášť si pak budeme na druhé pásce počítat, kolik slov akceptuje. Jakmile toto číslo dosáhne 2000, akceptujeme. Na druhou stranu nikdy nezamítáme, protože nemůžeme v konečném čase zjistit, zda stroj bude v budoucnu akceptovat nebo cyklit. Nyní provedeme redukci:

$$\text{NONEMPTY} \leq \text{TS2000}$$

Mějme stroj $\mathcal{M}$ s páskovou abecedou Σ. Sestrojíme stroj $\mathcal{N}$ s páskovou abecedou $\Sigma \cup \{x\}$, kde x je nový symbol nevyskytující se v Σ. Stroj $\mathcal{N}$ na začátku prochází pásku, a pokud se na ní vyskytuje kterýkoli z řetězců x^n pro $n \in \{1, \ldots, 1999\}$, pak akceptuje. V opačném případě pokračuje v činnosti jako stroj $\mathcal{M}$. Zřejmě stroj $\mathcal{N}$ akceptuje alespoň 2000 slov právě tehdy, když původní stroj akceptoval alespoň jedno slovo.

Příklad 5. *Problém ekvivalence* definovaný jako problém příslušnosti do jazyka:

$$\text{EKVIV} = \{\langle\mathcal{M}\rangle \# \langle\mathcal{N}\rangle \mid \mathcal{L}(\mathcal{M}) = \mathcal{L}(\mathcal{N})\}$$

není ani částečně rozhodnutelný. Z dříve dokázaných uzávěrových vlastností víme, že doplněk jazyka NONEMPTY není ani rekurzivně spočetný (Postova věta), potom ani následující jazyk není r.e.:

$$\text{EMPTY} = \{\langle\mathcal{M}\rangle \mid \mathcal{L}(\mathcal{M}) = \emptyset\}$$

neboť se od doplňku jazyka NONEMPTY liší pouze ve všech řetězcích, které nejsou kódem žádného Turingova stroje a které dokážeme snadno algoritmicky „odfiltrovat“. Fakt, že EKVIV není ani rekurzivně spočetný dokážeme právě redukcí $\text{EMPTY} \leq \text{EKVIV}$. Snadno můžeme sestrojit stroj $\mathcal{P}$, o kterém víme, že rozpoznává prázdný jazyk, např. stroj s prázdnou přechodovou funkcí. Redukce vezme Turingův stroj $\mathcal{M}$ a vrátí $\mathcal{M}\#\mathcal{P}$. Potom stroje $\mathcal{M}$ a $\mathcal{P}$ rozpoznávají tentýž jazyk právě tehdy, když $\mathcal{M}$ rozpoznává prázdný jazyk. Tím je redukce hotova.

Cvičení 1. Dokažte, že následující jazyk není ani rekurzivně spočetný:

$$\text{FINITE} = \{\langle M \rangle \mid \mathcal{L}(M) \text{ je konečný}\}$$

Cvičení 2. Dokažte, že následující jazyk není ani rekurzivně spočetný:

$$\text{INFINITE} = \{\langle M \rangle \mid \mathcal{L}(M) \text{ je nekonečný}\}$$

Cvičení 3. Dokažte, že následující jazyk není ani rekurzivně spočetný:

$$\text{REG} = \{\langle M \rangle \mid \mathcal{L}(M) = R\}$$

kde R je nějaký pevně zvolený regulární jazyk.

Definice. Mějme $\mathcal{C}$ třídu jazyků. Řekneme, že jazyk A je:

- $\mathcal{C}$-těžký, jestliže pro libovolné $X \in \mathcal{C}$ platí $X \leq A$.
- $\mathcal{C}$-úplný, jestliže je $\mathcal{C}$-těžký a zároveň $A \in \mathcal{C}$.

Poznámka. Bez důkazu uveďme, že jazyk ACCEPT je úplný pro třídu rekurzivně spočetných jazyků. To souvisí s univerzalitou příslušného Turingova stroje - interpreta. Rovněž libovolný jazyk X, který je ekvivalentní ACCEPT, je úplný pro třídu rekurzivně spočetných jazyků. Ukázali jsme, že to jsou například jazyky HALT nebo NONEMPTY.

Cvičení 4. Rozhodněte, zda platí následující implikace. Své rozhodnutí zdůvodněte:

a) $A \leq B \Rightarrow A^C \leq B^C$

b) $A \leq B$ a B je regulární $\Rightarrow A$ je regulární

c) A je rekurzivně spočetný a $A^C \leq A \Rightarrow A$ je rekurzivní

d) A je rekurzivně spočetný a $A \leq A^C \Rightarrow A$ je rekurzivní

e) $A \leq B$ a A je rekurzivní $\Rightarrow B$ je rekurzivní

f) A je rekurzivně spočetný $\Rightarrow A \leq \text{HALT}$

Definice. *Postův systém nad* Σ je konečná množina dvojic:

$$P = \{(\alpha_i, \beta_i) \mid \alpha_i, \beta_i \in \Sigma^*, 1 \leq i \leq n\}$$

Řešení Postova systému je neprázdná konečná posloupnost přirozených čísel $i_1, \ldots, i_k \in \{1, \ldots, n\}$ taková, že:

$$\alpha_{i_1} \cdots \alpha_{i_k} = \beta_{i_1} \cdots \beta_{i_k}$$

Definujeme:

- *Postův korespondenční problém* jako problém pro zadaný Postův systém P určit, zda má nějaké řešení.

$$\text{PCP} = \{\langle P \rangle \mid P \text{ je Postův systém, který má řešení}\}$$

• *iniciální Postův korespondenční problém* jako problém pro daný Postův systém určit, zda má nějaké řešení začínající číslem 1.

$$\text{inPCP} = \{\langle P \rangle \mid P \text{ je Postův systém, který má řešení začínající číslem 1}\}$$

Příklad 6. Mějme Postův systém:

$$\{(c, abc), (ca, b), (a, ca), (ab, a)\}$$

Tento systém má řešení 42341:

$$ab \cdot ca \cdot a \cdot ab \cdot c = a \cdot b \cdot ca \cdot a \cdot abc$$

Tento systém na druhou stranu nemá řešení začínající číslem 1, neboť se nemůžou rovnat dvě slova, kde první začíná znakem c a druhé řetězcem abc.

Cvičení 5. Nalezněte řešení následujícího Postova systému:

$$\{(aa, a), (ab, abab), (b, a), (aba, b)\}$$

Věta 18.2. Oba problémy PCP i inPCP jsou nerozhodnutelné. Lze provést následující redukce:

$$\text{ACCEPT} \leq \text{inPCP} \leq \text{PCP}$$

Cvičení 6. Ukažte, že Postův korespondenční problém je nerozhodnutelný, i když se omezíme na abecedu $\{0, 1\}$.