Rozhodnuteľnosť

Peter Kostolányi

6. decembra 2017

V nasledujúcom budeme skúmať *algoritmickú riešiteľnosť* výpočtových problémov – budeme sa teda zaoberať existenciou algoritmov. Obmedzíme sa pritom na špeciálnu triedu výpočtových problémov, takzvané *rozhodovacie problémy*. Ide o problémy, ktorých výstupom je pre každý vstup jedna z pravdivostných hodnôt „áno“ alebo „nie“; pre každý vstup je teda nutné *rozhodnúť*, či preň platí nejaká vlastnosť. Algoritmicky riešiteľné rozhodovacie problémy nazveme *rozhodnuteľnými* a v nasledujúcom tak vybudujeme základy teórie *algoritmickej rozhodnuteľnosti*.

Na dôkaz rozhodnuteľnosti daného výpočtového problému stačí prísť s algoritmom, ktorý ho rieši (rozhoduje). Našim cieľom (okrem iného) bude nájsť problém, ktorý rozhodnuteľný nie je, a teda preň *neexistuje žiaden algoritmus*. Dôkaz takéhoto tvrdenia sa však nezaobíde bez formálnej definície rozhodovacieho problému a formálnej definície algoritmu. V nasledujúcom preto:

1. Sformalizujeme pojem rozhodovacieho problému – vhodnou formalizáciou je *jazyk*.
2. Sformalizujeme pojem algoritmu – tu sa vhodnou formalizáciou javí byť *Turingov stroj zastavujúci na každom vstupe*, čo je založené na takzvanej *Turingovej téze*.
3. Obidve formalizácie využijeme na vybudovanie základov teórie rozhodnuteľnosti.

Rozhodovacie problémy

Pod *rozhodovacím problémom* rozumieme výpočtový problém, ktorého výstupom je pre každý vstup booleovská hodnota „áno“ alebo „nie“. Každá sada vstupov rozhodovacieho problému by mala byť reprezentovateľná ako slovo nad nejakou abecedou Σ. Rozhodovací problém je potom jednoznačne určený množinou – *jazykom* – tých vstupných slov, pre ktoré je výstup „áno“. Pomocou kódovania je v prípade potreby možné zaručiť, že pôjde o jazyk nad abecedou $\Sigma = \{0, 1\}$.

Formalizácia 1. Pojem *rozhodovacieho problému* formalizujeme ako *jazyk* nad nejakou abecedou Σ obsahujúci práve všetky vstupy problému, pre ktoré je výstupom „áno“.

Príklad 1. Uvažujme rozhodovací problém daný nasledovne:

Vstup: Prirodzené číslo $n \in \mathbb{N}$ dané svojou binárnou reprezentáciou.

Výstup: „Áno“ práve vtedy, keď je n prvočíslo.

Tomuto problému zodpovedá jazyk

$$L = \{w \in \{0,1\}^* \mid w \text{ je binárny zápis prvočísla}\} = \{10, 11, 101, 111, 1011, \ldots\}.$$

Príklad 2. Uvažujme rozhodovací problém daný nasledovne:

Vstup: Prirodzené číslo $n \in \mathbb{N}$ dané svojou dekadickou reprezentáciou; cifra $d \in \{0, 1, \ldots, 9\}$.

Výstup: „Áno“ práve vtedy, keď je n-tá cifra desatinného rozvoja čísla π rovná d.

Obidva vstupy musia byť reprezentované ako jediné slovo – to môžeme docieliť napríklad ich oddelením špeciálnym symbolom #. Uvedenému rozhodovaciemu problému tak zodpovedá jazyk

$$L = \{w\#d \mid w \in \Sigma^*;\ d \in \Sigma;\ \text{dec}(w)\text{-ta cifra desatinného rozvoja } \pi \text{ je rovná } d\},$$

kde $\Sigma = \{0, 1, \ldots, 9\}$ a kde pre $w \in \Sigma^*$ definujeme $\text{dec}(w)$ ako prirodzené číslo, ktorého dekadickou reprezentáciou je slovo w. Keďže $\pi = 3.14159\ldots$, platí

$$L = \{1\#1, 2\#4, 3\#1, 4\#5, 5\#9, \ldots\}.$$

Poznamenajme ešte, že v obidvoch príkladoch ide iba o jedno z viacerých možných kódovaní vstupov. Na konkrétnom kódovaní väčšinou nezáleží.

Turingova téza

Dospeli sme teda k formálnej definícii rozhodovacieho problému – ide o jazyk nad nejakou abecedou Σ. V podobnom duchu ešte potrebujeme formalizovať aj pojem algoritmu riešiaceho daný rozhodovací problém. Vhodnou formalizáciou sa javia byť *Turingove stroje, ktoré sa zastavia na každom vstupe* a ktoré akceptujú jazyk zodpovedajúci danému rozhodovaciemu problému.

Práca Alana Turinga, ktorej výsledkom bola definícia matematickej abstrakcie počítacieho zariadenia dnes známej ako Turingov stroj, bola do veľkej miery motivovaná práve potrebou formalizácie pojmu algoritmus. Turingova téza je (principiálne neoveriteľné) tvrdenie, že každý (v neformálnom zmysle) algoritmický výpočet možno realizovať na Turingovom stroji. Turingovu tézu nemožno ani dokázať ani vyvrátiť, pretože ide o tvrdenie, ktoré hovorí, že Turingove stroje sú vhodnou formalizáciou inak čisto intuitívne chápaného pojmu algoritmus. Inými slovami: Turingov stroj možno chápať ako pokus o formálnu definíciu algoritmu, pričom Turingova téza je neformálne tvrdenie takúto definíciu obhajujúce.

Napriek principiálnej neoveriteľnosti Turingovej tézy existujú viaceré argumenty na jej podporu. Tým najdôležitejším je skutočnosť, že v zásade všetky pokusy Turingových súčasníkov a následníkov o alternatívnu definíciu algoritmu vyústili v model, ktorý je z hľadiska výpočtovej sily s Turingovými strojmi ekvivalentný. Jednou z takýchto alternatívnych formalizácií je napríklad λ-kalkul Alonza Churcha, vďaka čomu je Turingova téza v literatúre známa aj ako Churchova-Turingova téza. Ďalšími formalizáciami sú napríklad rekurzívne funkcie, Minského registrové stroje, Markovove algoritmy, atď. V nasledujúcom prijmeme Turingovu tézu a Turingove stroje využijeme ako prostriedok na vybudovanie základov teórie algoritmickej vypočítateľnosti.

Formalizácia 2. Pojem *algoritmu* rozhodujúceho daný rozhodovací problém formalizujeme ako *Turingov stroj, ktorý sa na každom vstupe zastaví* a ktorý akceptuje jazyk zodpovedajúci danému rozhodovaciemu problému.

Stroje zastavujúce na každom vstupe a rekurzívne jazyky

Požiadavka zastavenia Turingovho stroja na každom vstupe odzrkadľuje neformálnu požiadavku, aby sa každý výpočet algoritmu raz skončil. Jej význam možno ilustrovať aj nasledujúcim myšlienkovým experimentom.

Nech A je deterministický Turingov stroj pracujúci na vstupe w. Bez ujmy na všeobecnosti môžeme predpokladať, že stroj A neobsahuje žiadne prechody vedúce z akceptačných stavov. Stroj A sa teda v akceptačnej konfigurácii vždy „zasekne“. Výpočet stroja A na slove w tak môže prebiehať troma principiálne odlišnými spôsobmi:

1. Po nejakom počte krokov príde do akceptačnej konfigurácie a zastaví sa. Potom $w \in L(A)$.
2. Po nejakom počte krokov sa zastaví v neakceptačnej konfigurácii. Potom $w \notin L(A)$.
3. Nikdy sa nezastaví a všetky konfigurácie, cez ktoré prejde, sú neakceptačné. Potom $w \notin L(A)$.

Uvažujme teraz pozorovateľa výpočtu stroja A na slove w, ktorý sa iba na základe neho snaží zistiť, či $w \in L(A)$. Jedinou informáciou, ktorú má pozorovateľ k dispozícii je, či výpočet skončil a ak áno, tak s akým výsledkom. Ľahko vidieť, že pozorovateľ nedokáže v žiadnom bode výpočtu odlíšiť prípad, keď je výpočet nekonečný (a teda $w \notin L(A)$) od prípadu, keď je nutné odsimulovať ešte niekoľko krokov a výpočet sa zastaví (a teda môže platiť $w \in L(A)$ ako aj $w \notin L(A)$).

Požiadavka zastavenia Turingovho stroja na každom vstupe je teda opodstatnená – stroj zastavujúci na každom vstupe totiž môže pracovať iba dvoma odlišnými spôsobmi:

1. Po nejakom počte krokov príde do akceptačnej konfigurácie a zastaví sa. Potom $w \in L(A)$.
2. Po nejakom počte krokov sa zastaví v neakceptačnej konfigurácii. Potom $w \notin L(A)$.

Jazyky akceptované takýmito strojmi nazveme *rekurzívnymi*.

Definícia 1. Nech L je jazyk. Jazyk L je *rekurzívny*, ak existuje deterministický Turingov stroj $A = (K, \Sigma, \Gamma, \delta, q_0, F)$, ktorý sa na každom vstupe zastaví a pre ktorý platí $L(A) = L$. Triedu všetkých rekurzívnych jazykov označujeme symbolom $\mathscr{L}_{rec}$.

Neskôr ukážeme, že táto definícia je netriviálna – existuje teda Turingov stroj, ku ktorému neexistuje žiaden ekvivalentný Turingov stroj zastavujúci na každom vstupe. Inak povedané, $\mathscr{L}_{rec}$ je *vlastnou* podtriedou $\mathscr{L}_{RE}$.

Rozhodnuteľné, nerozhodnuteľné a rekurzívne vyčísliteľné problémy

Na základe Turingovej tézy sme pojem algoritmu rozhodujúceho daný rozhodovací problém sformalizovali ako Turingov stroj, ktorý sa na každom vstupe zastaví a ktorý akceptuje jazyk zodpovedajúci danému rozhodovaciemu problému.

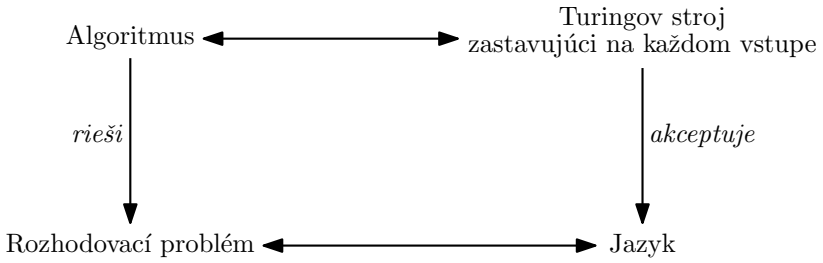
Rozhodovací problém nazveme *rozhodnuteľným*, ak existuje algoritmus, ktorý ho rozhoduje. To je práve vtedy, keď existuje Turingov stroj, ktorý sa na každom vstupe zastaví a ktorý akceptuje jazyk zodpovedajúci danému rozhodovaciemu problému. A to je (podľa predchádzajúceho oddielu) práve vtedy, keď je jazyk zodpovedajúci danému rozhodovaciemu problému rekurzívny.

Rozhodovací problém nazveme *nerozhodnuteľným*, ak nie je rozhodnuteľný. Existenciu nerozhodnuteľných problémov dokážeme neskôr.

Rozhodovací problém nazveme *rekurzívne vyčísliteľným* (niekde tiež *čiastočne rozhodnuteľným*), ak existuje Turingov stroj, ktorý akceptuje jemu zodpovedajúci jazyk (pričom sa nemusí zastaviť na každom vstupe). To je práve vtedy, keď je jazyk zodpovedajúci danému rozhodovaciemu problému rekurzívne vyčísliteľný. Rekurzívne vyčísliteľný problém môže byť rozhodnuteľný (ak mu zodpovedá rekurzívny jazyk), alebo nerozhodnuteľný (v opačnom prípade).

Rozhodovacie problémy, algoritmy a ich formalizácie

Vzťahy neformálnych pojmov rozhodovacieho problému a algoritmu k ich formalizáciám sú znázornené na obrázku 1.



Obr. 1: Rozhodovacie problémy, algoritmy a ich formalizácie.

Kľúčové pojmy a ich formalizácie sú zhrnuté v tabuľke 1 – definíciu rekurzívnych jazykov a triedy $\mathscr{L}_{rec}$ možno nájsť vyššie.

Pojem	**Formalizácia**
Rozhodovací problém	Jazyk (nad nejakou abecedou Σ)
Algoritmus	Turingov stroj zastavujúci na každom vstupe
Rozhodnuteľný problém	Jazyk z triedy $\mathscr{L}_{rec}$
Nerozhodnuteľný problém	Jazyk mimo triedy $\mathscr{L}_{rec}$
Rekurzívne vyčísliteľný problém	Jazyk z triedy $\mathscr{L}_{RE}$

Tabuľka 1: Zhrnutie kľúčových pojmov a ich formalizácií.

Existencia jazykov mimo $\mathscr{L}_{RE}$

Rekurzívne vyčísliteľných jazykov nad abecedou $\Sigma = \{0, 1\}$ je najviac toľko, čo Turingových strojov s binárnou vstupnou abecedou. Turingových strojov s binárnou vstupnou abecedou je najviac toľko, čo ich kódov. Kód Turingovho stroja je binárny reťazec a množina všetkých (konečných) binárnych reťazcov je spočítateľne nekonečná. Všetkých binárnych rekurzívne vyčísliteľných jazykov je teda tiež len spočítateľne veľa. Na druhej strane, všetkých jazykov nad abecedou $\Sigma = \{0, 1\}$ je nespočítateľne veľa, lebo ide o podmnožiny spočítateľne nekonečnej množiny Σ^*. Musí preto existovať jazyk nad abecedou $\Sigma = \{0, 1\}$, ktorý nie je rekurzívne vyčísliteľný.

Rekurzívne vyčísliteľné jazyky zodpovedajú rekurzívne vyčísliteľným problémom. Z uvedeného teda vyplýva, že existuje rozhodovací problém, ktorý nie je rekurzívne vyčísliteľný. Každý rozhodnuteľný problém je nutne aj rekurzívne vyčísliteľný. Z dokázaného tvrdenia teda tiež vyplýva, že existuje rozhodovací problém, ktorý nie je rozhodnuteľný.

Uvedený dôkaz je čisto existenčný. V nasledujúcom ale nájdeme aj *konkrétny* jazyk (rozhodovací problém), o ktorom dokážeme, že nie je rekurzívne vyčísliteľný.

Diagonálny problém a jeho nerozhodnuteľnosť

Diagonálny problém je rozhodovací problém daný nasledovne:

Vstup: Kód $\langle A \rangle$[1] deterministického Turingovho stroja[2] A nad vstupnou abecedou $\{0, 1\}$.

Výstup: „Áno“ práve vtedy, keď $\langle A \rangle \in L(A)$.

Diagonálnemu problému zodpovedá *diagonálny jazyk*:

$$L_D = \{\langle A \rangle \mid A \text{ je det. TS nad vstupnou abecedou } \{0, 1\};\ \langle A \rangle \in L(A)\}.$$

Podobne možno uvažovať aj *komplementárny diagonálny problém*, ktorý je daný takto:

Vstup: Kód $\langle A \rangle$ deterministického Turingovho stroja A nad vstupnou abecedou $\{0, 1\}$.

Výstup: „Áno“ práve vtedy, keď $\langle A \rangle \notin L(A)$.

Komplementárnemu diagonálnemu problému zodpovedá *komplement diagonálneho jazyka*:

$$L_D^C = \{\langle A \rangle \mid A \text{ je det. TS nad vstupnou abecedou } \{0, 1\};\ \langle A \rangle \notin L(A)\}.$$

Poznámka 1. Skutočnosť, že komplementárnemu diagonálnemu problému zodpovedá jazyk L_D^C je dôsledkom toho, že každý binárny reťazec je kódom nejakého Turingovho stroja – „nezmyselné“ reťazce zodpovedajú stroju akceptujúcemu prázdny jazyk.

Veta 1. *Komplementárny diagonálny problém nie je rekurzívne vyčísliteľný:* $L_D^C \notin \mathscr{L}_{RE}$.

Dôkaz. Sporom. Predpokladajme, že $L_D^C \in \mathscr{L}_{RE}$. Potom existuje deterministický Turingov stroj M taký, že $L(M) = L_D^C$.

Ak $\langle M \rangle \in L(M)$, tak z definície diagonálneho jazyka vyplýva $\langle M \rangle \in L_D$. To je ale spor, pretože $L(M) = L_D^C$, a teda $L(M)$ nemôže obsahovať slovo $\langle M \rangle \in L_D$.

Preto $\langle M \rangle \notin L(M)$. V takom prípade ale z definície diagonálneho jazyka vyplýva $\langle M \rangle \in L_D^C$ a z definície stroja M plynie $\langle M \rangle \in L(M)$, čo je opäť spor, lebo v takom prípade $\langle M \rangle \in L_D$. □

Veta 2. *Diagonálny problém je rekurzívne vyčísliteľný, ale nie je rozhodnuteľný:* $L_D \in \mathscr{L}_{RE} - \mathscr{L}_{rec}$.

Dôkaz. Zjavne $L_D \in \mathscr{L}_{RE}$ – stroj akceptujúci jazyk L_D môže pracovať tak, že pre každý vstup $\langle A \rangle$ odsimuluje výpočet univerzálneho Turingovho stroja na vstupe $\langle A \rangle \# \langle A \rangle$.

Ďalej sporom, nech $L_D \in \mathscr{L}_{rec}$. Potom existuje deterministický Turingov stroj M zastavujúci na každom vstupe taký, že $L(M) = L_D$. Nech M' pracuje ako M s rozdielom, že M' akceptuje práve vtedy, keď M zamieta. Zjavne $L(M') = L_D^C$, z čoho $L_D^C \in \mathscr{L}_{rec} \subseteq \mathscr{L}_{RE}$ – spor. □

[1] Pod týmto zápisom chápeme *ľubovoľný* z viacerých možných kódov stroja A – bližšie informácie možno nájsť v poznámkach ku kódovaniu Turingových strojov.

[2] Deterministickým Turingovým strojom rozumieme *vždy* stroj v normálnom tvare bez prechodov vedúcich z akceptačných stavov – ten je popísaný v poznámkach o kódovaní Turingových strojov. Viac už na tento predpoklad nebudeme upozorňovať.

Vzťah medzi triedami $\mathscr{L}_{rec}$ a $\mathscr{L}_{RE}$

Keďže sme našli príklad rekurzívne vyčísliteľného jazyka, ktorý nie je rekurzívny a keďže je inklúzia $\mathscr{L}_{rec} \subseteq \mathscr{L}_{RE}$ zrejmá, dokázali sme $\mathscr{L}_{rec} \subsetneq \mathscr{L}_{RE}$. Našli sme tiež príklad jazyka, ktorý nie je ani rekurzívne vyčísliteľný. Trieda $\mathscr{L}_{RE}$ je teda vlastnou podtriedou triedy všetkých jazykov.

V nasledujúcom zavŕšime náš obraz o uvedených triedach jazykov dôkazom tvrdenia charakterizujúceho triedu rekurzívnych jazykov pomocou rekurzívne vyčísliteľných jazykov a ich komplementov. Najprv ešte ale ukážeme, že trieda $\mathscr{L}_{rec}$ je uzavretá na komplement (čo je vlastnosť, ktorá kvôli jazyku L_D pre triedu $\mathscr{L}_{RE}$ neplatí).

Veta 3. *Trieda jazykov $\mathscr{L}_{rec}$ je uzavretá na komplement.*

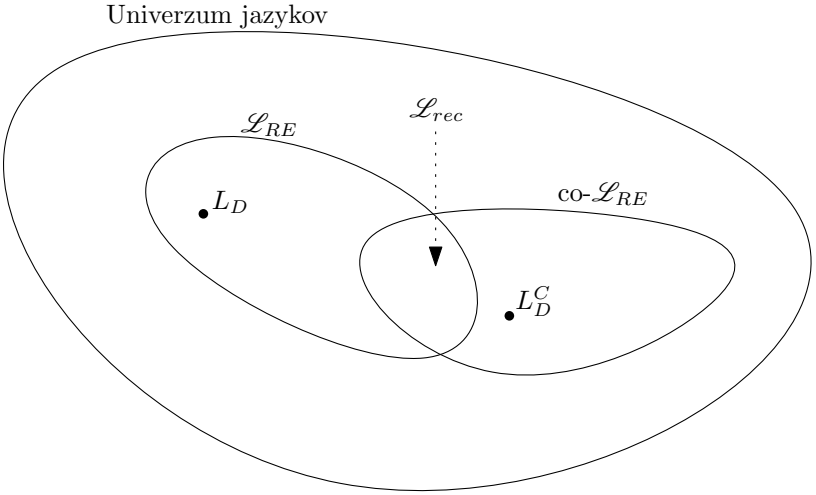
Dôkaz. Nech $L \in \mathscr{L}_{rec}$. Potom existuje deterministický Turingov stroj A, ktorý sa na každom svojom vstupe zastaví a pre ktorý platí $L(A) = L$. Jazyk L^C je potom akceptovaný deterministickým Turingovým strojom A', ktorý simuluje výpočet stroja A až kým sa nezastaví a následne akceptuje práve vtedy, keď stroj A vstup zamietne. □

Veta 4. *Nech L je jazyk. Potom $L \in \mathscr{L}_{rec}$ práve vtedy, keď $L \in \mathscr{L}_{RE}$ a $L^C \in \mathscr{L}_{RE}$.*

Dôkaz. Dokážeme postupne jednotlivé implikácie:

- $\Rightarrow$: Ak $L \in \mathscr{L}_{rec}$, vďaka vete 3 je aj $L^C \in \mathscr{L}_{rec}$. Tvrdenie potom vyplýva z inklúzie $\mathscr{L}_{rec} \subsetneq \mathscr{L}_{RE}$.
- $\Leftarrow$: Nech $L \in \mathscr{L}_{RE}$ a $L^C \in \mathscr{L}_{RE}$. Potom existuje deterministický Turingov stroj A_1 taký, že $L(A_1) = L$ a deterministický Turingov stroj A_2 taký, že $L(A_2) = L^C$. Na každom vstupe zastavujúci deterministický Turingov stroj A akceptujúci jazyk L môže pracovať tak, že striedavo simuluje vždy jeden krok výpočtu stroja A_1 a jeden krok výpočtu stroja A_2. Jeden z týchto strojov svoj vstup istotne po nejakom konečnom počte krokov akceptuje. Ak akceptuje stroj A_1, akceptuje aj stroj A. Ak akceptuje stroj A_2, stroj A svoj vstup zamietne a zastaví sa.

Tvrdenie je dokázané. □



Obr. 2: Vzájomné vzťahy medzi triedami $\mathscr{L}_{rec}$, $\mathscr{L}_{RE}$ a co-$\mathscr{L}_{RE}$ a príklady jazykov v jednotlivých triedach. Trieda $\mathscr{L}_{rec}$ zodpovedá rozhodnuteľným problémom, všetko ostatné zodpovedá nerozhodnuteľným problémom. Trieda $\mathscr{L}_{RE}$ zodpovedá rekurzívne vyčísliteľným problémom, ktoré môžu byť rozhodnuteľné (prienik s co-$\mathscr{L}_{RE}$, t.j. trieda $\mathscr{L}_{rec}$) alebo nerozhodnuteľné (zvyšok triedy $\mathscr{L}_{RE}$).

Ak teda definujeme triedu co-$\mathscr{L}_{RE}$ ako triedu všetkých jazykov, ktorých komplement je v $\mathscr{L}_{RE}$ (**pozor**: nejde o triedu $\mathscr{L}_{RE}^C$), možno vetu 4 preformulovať aj ako $\mathscr{L}_{rec} = \mathscr{L}_{RE} \cap$ co-$\mathscr{L}_{RE}$. Vzájomné vzťahy tried $\mathscr{L}_{rec}$, $\mathscr{L}_{RE}$ a co-$\mathscr{L}_{RE}$ tak môžu byť znázornené ako na obrázku 2.

Redukcie: nerozhodnuteľnosť univerzálneho problému a problému zastavenia

Častou metódou dokazovania výsledkov o nerozhodnuteľnosti sú *redukcie*. Zredukovať rozhodovací problém A na rozhodovací problém B zhruba znamená dokázať, že keby bol problém B rozhodnuteľný (rekurzívne vyčísliteľný), bol by rozhodnuteľný (rekurzívne vyčísliteľný) aj problém A. Keby sme teda mali k dispozícii „čiernu krabičku“ – „hypotetický“ Turingov stroj – pre problém B, vedeli by sme skonštruovať Turingov stroj[3] aj pre problém A.

V prípade, že o probléme A už máme dokázané, že nie je rozhodnuteľný (rekurzívne vyčísliteľný), dostaneme takýmto spôsobom spor – ani problém B teda nemôže byť rozhodnuteľný (rekurzívne vyčísliteľný). Túto všeobecne platnú dôkazovú schému možno samozrejme sformulovať aj v terminológii rekurzívnych a rekurzívne vyčísliteľných jazykov.

V nasledujúcom pomocou redukcií dokážeme nerozhodnuteľnosť dvoch významných rozhodovacích problémov – univerzálneho problému a problému zastavenia.

Univerzálny problém je daný nasledovne:

Vstup: Kód $\langle A\rangle$ deterministického Turingovho stroja A nad vstupnou abecedou $\{0,1\}$; slovo $w \in \{0,1\}^*$.

Výstup: „Áno“ práve vtedy, keď $w \in L(A)$.

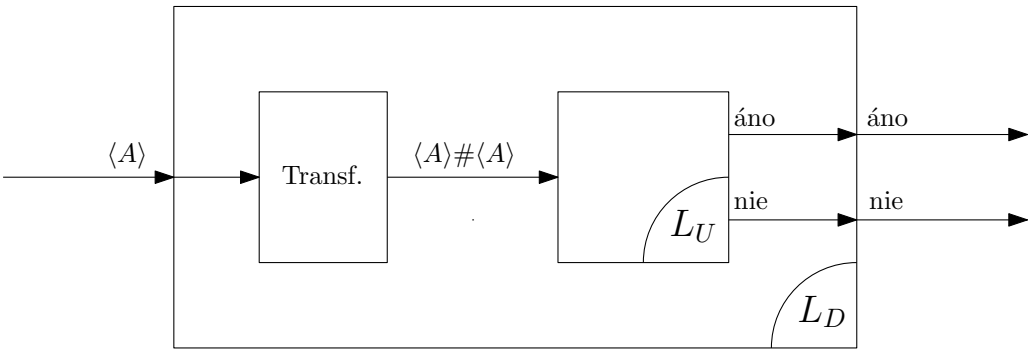
Univerzálnemu problému zodpovedá *univerzálny jazyk*, známy už z poznámok o kódovaní Turingových strojov a univerzálnom Turingovom stroji:

$$L_U = \{\langle A\rangle \# w \mid A \text{ je det. TS nad vstupnou abecedou } \{0,1\};\ w \in \{0,1\}^*;\ w \in L(A)\}.$$

Je intuitívne zrejmé, že univerzálny problém nebude rozhodnuteľný: potrebujeme simulovať Turingov stroj zo vstupu, ktorý sa nemusí zastaviť. To však pravdepodobne nepôjde urobiť pomocou Turingovho stroja, ktorý sa zastaviť má. Na poriadny dôkaz tohto tvrdenia použijeme redukciu diagonálneho problému na univerzálny problém.

Veta 5. *Univerzálny problém je rekurzívne vyčísliteľný, no nie je rozhodnuteľný:* $L_U \in \mathscr{L}_{RE} - \mathscr{L}_{rec}$.

Dôkaz. Univerzálny jazyk je akceptovaný univerzálnym Turingovým strojom – univerzálny problém je preto rekurzívne vyčísliteľný. V nasledujúcom dokážeme, že univerzálny problém nie je rozhodnuteľný.



Obr. 3: Schéma redukcie diagonálneho problému na univerzálny problém. „Krabička“ pre L_U zodpovedá stroju M a výsledná „krabička“ pre L_D zodpovedá stroju M'.

Sporom, nech je rozhodnuteľný. Potom existuje deterministický Turingov stroj M, ktorý sa na každom vstupe zastaví a pre ktorý platí $L(M) = L_U$. Ukážeme, že s použitím stroja M –

[3]Od týchto dvoch strojov požadujeme zastavenie na každom vstupe podľa toho, či sa zaujímame o rozhodnuteľnosť (vtedy sa na každom vstupe zastaviť musia), alebo iba o rekurzívnu vyčísliteľnosť (vtedy sa na každom vstupe zastaviť nemusia).

čiže „čiernej krabičky“ rozhodujúcej univerzálny problém – by sme vedeli skonštruovať na každom vstupe zastavujúci deterministický Turingov stroj M' taký, že $L(M') = L_D$. To bude spor s vyššie dokázanou skutočnosťou, že diagonálny problém nie je rozhodnuteľný.

Stroj M' môže pracovať tak, že svoj vstup $\langle A \rangle$ (kód nejakého Turingovho stroja A) upraví na $\langle A \rangle \# \langle A \rangle$ a následne na tomto slove odsimuluje výpočet stroja M. Ten sa v konečnom čase zastaví, pričom buď akceptuje alebo zamietne. Ak akceptuje, platí $\langle A \rangle \in L(A)$ a stroj M' svoj vstup taktiež akceptuje. Ak stroj M svoj vstup zamietne, platí $\langle A \rangle \notin L(A)$, a teda stroj M' tiež zamietne. Zjavne $L(M') = L_D$. Schéma redukcie je na obrázku 3. □

Problém zastavenia je daný nasledovne:

Vstup: Kód $\langle A \rangle$ deterministického Turingovho stroja A nad vstupnou abecedou $\{0, 1\}$; slovo $w \in \{0, 1\}^*$.

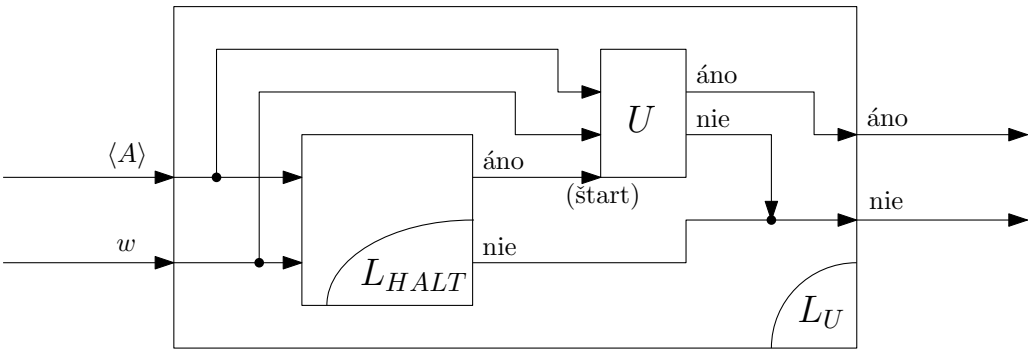
Výstup: „Áno“ práve vtedy, keď sa stroj A na vstupe w zastaví.

Problému zastavenia zodpovedá jazyk

$L_{HALT} = \{\langle A \rangle \# w \mid A \text{ je det. TS nad vstupnou abecedou } \{0, 1\};\ w \in \{0, 1\}^*;\ A \text{ sa na } w \text{ zastaví}\}$.

Veta 6. *Problém zastavenia je rekurzívne vyčísliteľný, ale nie je rozhodnuteľný. Inak povedané, platí* $L_{HALT} \in \mathscr{L}_{RE} - \mathscr{L}_{rec}$.

Dôkaz. Problém je rekurzívne vyčísliteľný, pretože stačí simulovať výpočet stroja A na slove w a ak sa zastaví, akceptovať.



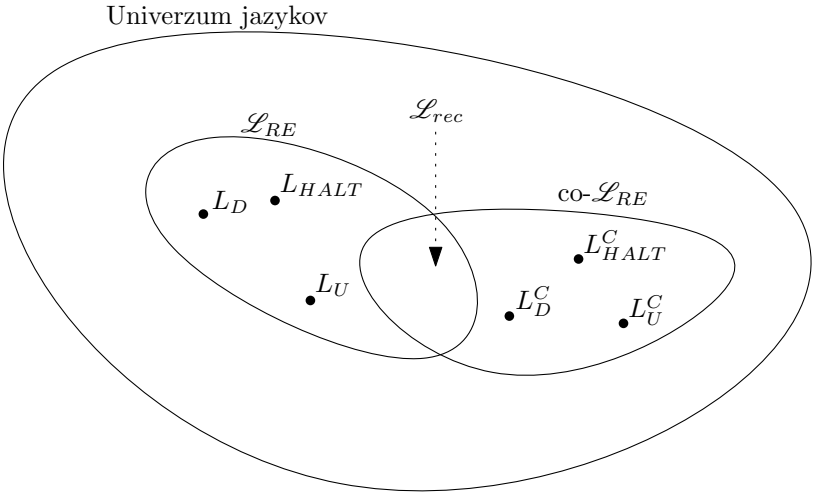
Obr. 4: Schéma redukcie univerzálneho problému na problém zastavenia.

Nerozhodnuteľnosť problému zastavenia dokážeme redukciou univerzálneho problému na problém zastavenia. Skutočne, ak by sme mali k dispozícii „čiernu krabičku“ rozhodujúcu problém zastavenia, vedeli by sme skonštruovať Turingov stroj rozhodujúci univerzálny problém, ktorý by pracoval nasledovne:

1. Na vstupe dostane kód $\langle A \rangle$ Turingovho stroja A a slovo w.
2. Zavolá na týchto vstupoch „čiernu krabičku“ pre problém zastavenia.
3. Ak „čierna krabička“ vráti „áno“, stroj A sa na slove w v konečnom čase zastaví. Univerzálny problém teda možno rozhodnúť jednoduchou simuláciou výpočtu stroja A na univerzálnom Turingovom stroji U.
4. Ak „čierna krabička“ vráti „nie“, stroj A sa na slove w nezastaví, a teda ho nemôže akceptovať. Stroj teda môže vstup zamietnuť.

Konštrukciu stroja pre univerzálny problém na základe stroja pre problém zastavenia možno znázorniť diagramom na obrázku 4. □

Obrázok 2 teraz môžeme doplniť o pozíciu jazykov L_U a L_{HALT} ako aj ich komplementov. Výsledná situácia je znázornená na obrázku 5. Treba si ale uvedomiť, že jazyky L_U^C resp. L_{HALT}^C nezodpovedajú „komplementárnemu problému“ tak ako pri diagonálnom probléme – obsahujú totiž aj všetky slová nad abecedou $\{0, 1, \#\}$, ktoré neobsahujú práve jeden výskyt symbolu #.



Obr. 5: Situácia z obrázku 2 doplnená o pozíciu jazykov L_U, L_{HALT} a ich komplementov.

Jednoduchšie riešené úlohy

Úloha 1. Uvažujme rozhodovací problém daný nasledovne:

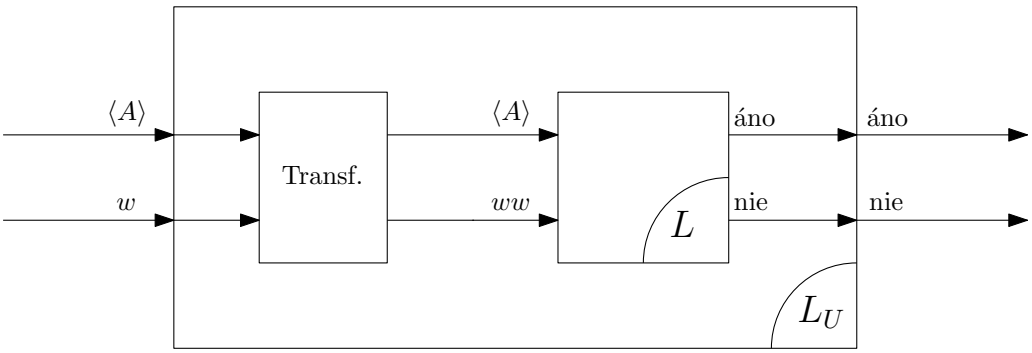
Vstup: Kód $\langle A \rangle$ deterministického Turingovho stroja A nad vstupnou abecedou $\{0, 1\}$; slovo ww, kde $w \in \{0, 1\}^*$.

Výstup: „Áno“ práve vtedy, keď $w \in L(A)$.

Zistite, či je uvedený problém rozhodnuteľný. Ak nie, je aspoň rekurzívne vyčísliteľný? Svoje tvrdenia dokážte.

Riešenie. Problému zo zadania zodpovedá jazyk

$$L = \{\langle A \rangle \# ww \mid A \text{ je det. TS nad vstupnou abecedou } \{0,1\};\ w \in \{0,1\}^*;\ w \in L(A)\}.$$



Obr. 6: Schéma redukcie univerzálneho problému na problém zo zadania.

Dokážeme, že problém zo zadania *nie je rozhodnuteľný*, ale *je rekurzívne vyčísliteľný* – jazyk L teda *patrí do* $\mathscr{L}_{RE}$, ale *nepatrí do* $\mathscr{L}_{rec}$.

Problém je zjavne rekurzívne vyčísliteľný – Turingov stroj akceptujúci jazyk L môže pracovať nasledovne: najprv overí korektnosť vstupu, t.j. či je vstup naozaj tvaru $\langle A\rangle\#ww$ pre nejaké $w \in \{0,1\}^*$. Ak áno, zistí slovo w a spustí univerzálny stroj na vstupe $\langle A\rangle\#w$.

Nerozhodnuteľnosť dokážeme redukciou univerzálneho problému na problém zo zadania. Keby sme mali k dispozícii „čiernu krabičku“ rozhodujúcu problém zo zadania, vedeli by sme skonštruovať aj Turingov stroj rozhodujúci univerzálny problém – ten upraví vstup $\langle A\rangle\#w$ na $\langle A\rangle\#ww$ a na ňom spustí „čiernu krabičku“ pre problém zo zadania. Tá vráti odpoveď „áno“ práve vtedy, keď $\langle A\rangle\#ww \in L$, čo je práve vtedy, keď $w \in L(A)$ – čiže práve vtedy, keď $\langle A\rangle\#w \in L_U$. Výstup „čiernej krabičky“ teda môžeme priamo použiť ako výstup stroja rozhodujúceho univerzálny problém. Schéma redukcie je na obrázku 6. □

Redukciu z predchádzajúcej úlohy možno považovať za relatívne jednoduchú, pretože nevyžaduje žiaden zásah do kódu $\langle A\rangle$ a transformujú sa iba zvyšné vstupy. V nasledujúcich úlohách dokážeme nerozhodnuteľnosť ďalších rozhodovacích problémov. Redukcie už však budú vyžadovať aj transformáciu kódu $\langle A\rangle$ zo vstupu redukovaného rozhodovacieho problému.

Úloha 2. Uvažujme rozhodovací problém daný nasledovne:

Vstup: Kód $\langle A\rangle$ deterministického Turingovho stroja A nad vstupnou abecedou $\{0,1\}$; kód $\langle c\rangle$ nejakého pracovného symbolu c stroja A; slovo $w \in \{0,1\}^*$.

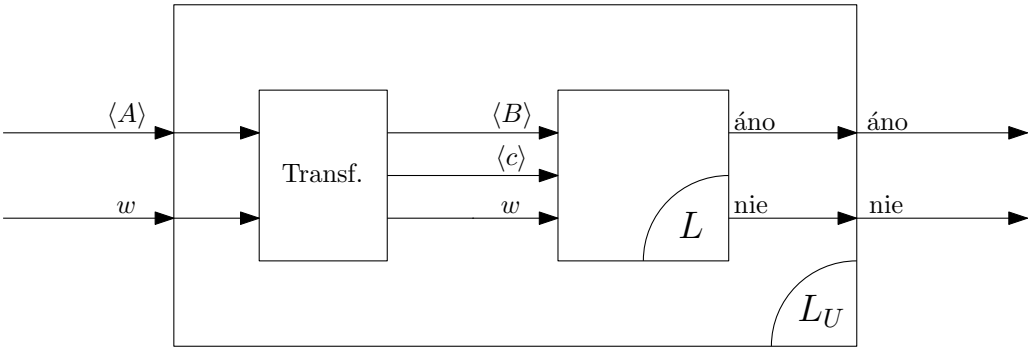
Výstup: „Áno“ práve vtedy, keď stroj A počas výpočtu na vstupe w aspoň raz zapíše na pásku symbol c.

Zistite, či je uvedený problém rozhodnuteľný. Ak nie, je aspoň rekurzívne vyčísliteľný? Svoje tvrdenia dokážte.

Riešenie. Označme symbolom L jazyk zodpovedajúci danému rozhodovaciemu problému. Redukciou z univerzálneho problému najprv dokážeme, že problém *nie je rozhodnuteľný*, čiže $L \notin \mathscr{L}_{rec}$. Za účelom sporu predpokladajme, že problém je rozhodnuteľný. Ukážeme, že v takom prípade by bol rozhodnuteľný aj univerzálny problém (spor).

Pokúsme sa teda za uvedeného predpokladu skonštruovať deterministický Turingov stroj, ktorý sa na každom vstupe zastaví a ktorý akceptuje jazyk L_U (zodpovedajúci univerzálnemu problému). Vstupom takéhoto stroja je kód $\langle A\rangle$ nejakého Turingovho stroja A nad vstupnou abecedou $\{0,1\}$ a slovo $w \in \{0,1\}^*$.

Stroj pre L_U musí rozhodnúť, či $w \in L(A)$, pričom pri rozhodovaní môže použiť stroj pre L. Preto je potrebné upraviť vstupy $\langle A\rangle$, w stroja pre L_U na vhodné vstupy stroja pre L tak, aby stroj pre L_U mohol zmysluplne využiť výstup stroja pre L.



Obr. 7: Schéma redukcie univerzálneho problému na problém zo zadania.

Za týmto účelom zjavne postačí prerobiť kód $\langle A\rangle$ na kódy $\langle B\rangle$, $\langle c\rangle$ stroja B a nejakého jeho pracovného symbolu c s nasledujúcou vlastnosťou: $w \in L(A)$ práve vtedy, keď B na vstupe w

zapíše na pásku aspoň raz symbol c. Potom totiž $\langle B\rangle\#\langle c\rangle\#w \in L$ práve vtedy, keď $\langle A\rangle\#w \in L_U$ – ako výstup stroja rozhodujúceho univerzálny problém tak možno priamo použiť výstup „čiernej krabičky“ rozhodujúcej problém zo zadania.

Stroj B s požadovanou vlastnosťou ale môže pracovať napríklad tak, že bude simulovať výpočet stroja A a ak ten akceptuje, zapíše na pásku nejaký *nový* symbol c.

Ukážeme, že na základe kódu stroja A vieme kódy stroja B a symbolu c algoritmicky skonštruovať. Nech $A = (K, \Sigma, \Gamma, \delta, q_0, F)$. Pre stroj B bude platiť $B = (K', \Sigma, \Gamma', \delta', q_0, F')$, pričom $K' = K \cup \{q_{novy}\}$, kde q_{novy} je nový stav, $\Gamma' = \Gamma \cup \{c\}$, kde c je nový symbol a prechodová funkcia δ' je definovaná nasledovne:

$$\forall q \in K - F\ \forall a \in \Gamma : \delta'(q, a) = \delta(q, a),$$
$$\forall q \in F\ \forall a \in \Gamma : \delta'(q, a) = (q_{novy}, c, 0).$$

Množina akceptačných stavov F' je zjavne nepodstatná, môžeme preto položiť $F' = \emptyset$.

Na prerobenie kódu $\langle A\rangle$ na kód $\langle B\rangle$ teda v zásade stačí zakomponovať do $\langle A\rangle$ jeden nový stav, jeden nový pracovný symbol (ktorého kódom bude $\langle c\rangle$) a niekoľko nových prechodov. To však zrejme ide urobiť algoritmicky (tzn. na Turingovom stroji).

Stroj rozhodujúci univerzálny problém teda bude pracovať tak, že svoje vstupy $\langle A\rangle$, w prerobí na $\langle B\rangle$, $\langle c\rangle$, w, na ktorých spustí stroj pre problém zo zadania. Výstup tohto stroja potom bude aj výstupom pre univerzálny problém. Schéma redukcie je na obrázku 7.

Ostáva teda dokázať alebo vyvrátiť rekurzívnu vyčísliteľnosť. Dokážeme, že problém *je rekurzívne vyčísliteľný*. Turingov stroj akceptujúci jazyk L zodpovedajúci problému zo zadania totiž môže pracovať tak, že pomocou univerzálneho Turingovho stroja simuluje výpočet stroja A zo vstupu na slove w zo vstupu a ak stroj A zapíše symbol c, stroj pre jazyk L akceptuje. □

Úloha 3. Uvažujme rozhodovací problém daný nasledovne:

Vstup: Kód $\langle A\rangle$ deterministického Turingovho stroja A nad vstupnou abecedou $\{0, 1\}$; kód $\langle q\rangle$ nejakého stavu q stroja A; slovo $w \in \{0, 1\}^*$.

Výstup: „Áno“ práve vtedy, keď sa stroj A na vstupe w dostane aspoň raz do konfigurácie so stavom q.

Zistite, či je uvedený problém rozhodnuteľný. Ak nie, je aspoň rekurzívne vyčísliteľný? Svoje tvrdenia dokážte.

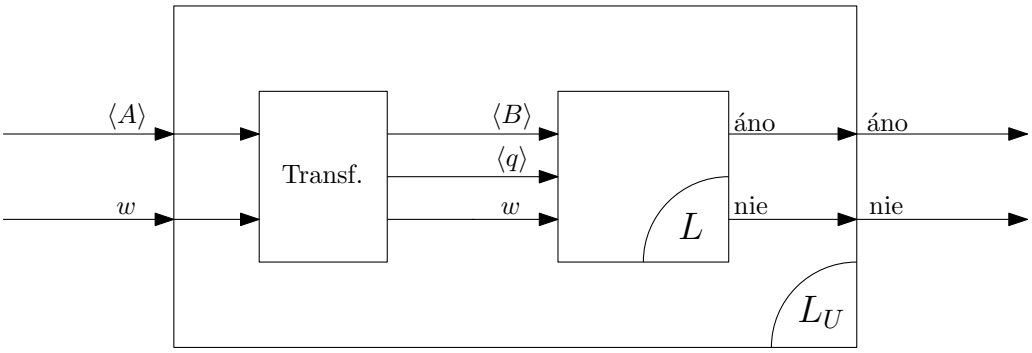
Riešenie. Problém *je rekurzívne vyčísliteľný*, pretože stačí pomocou univerzálneho Turingovho stroja simulovať výpočet stroja A na vstupe w a v prípade, že sa tento výpočet dostane do stavu q, akceptovať.

Nech L označuje jazyk zodpovedajúci problému zo zadania. Redukciou z univerzálneho problému na problém zo zadania dokážeme, že problém zo zadania *nie je rozhodnuteľný*. Za účelom sporu predpokladajme, že existuje deterministický Turingov stroj, ktorý sa na každom vstupe zastaví a ktorý akceptuje jazyk L. Ukážeme, že v tom prípade vieme takýto stroj skonštruovať aj pre univerzálny jazyk L_U, čo bude spor.

Vstupom stroja pre univerzálny problém je kód $\langle A\rangle$ nejakého Turingovho stroja A a slovo w, pričom treba rozhodnúť, či $w \in L(A)$. Pri konštrukcii stroja rozhodujúceho univerzálny problém potrebujeme nejakým spôsobom využiť stroj rozhodujúci problém zo zadania (jazyk L). To znamená, že vstupy $\langle A\rangle$, w stroja pre L_U treba upraviť na vhodné vstupy stroja pre L tak, aby stroj pre L_U mohol zmysluplne využiť výstup stroja pre L.

Za týmto účelom stačí prerobiť kód $\langle A\rangle$ stroja A na kód $\langle B\rangle$ nejakého iného stroja B a kód $\langle q\rangle$ nejakého jeho stavu q tak, aby platilo, že $w \in L(A)$ práve vtedy, keď sa stroj B počas výpočtu na slove w dostane do konfigurácie so stavom q.

Stroj pre L_U totiž môže v takom prípade pracovať nasledovne: transformuje svoje vstupy $\langle A\rangle$, w na $\langle B\rangle$, $\langle q\rangle$, w, na ktorých spustí stroj pre L. Ten sa v konečnom čase zastaví, pričom jeho odpoveď je „áno“ práve vtedy, keď sa stroj B na vstupe w dostane do konfigurácie so stavom



Obr. 8: Schéma redukcie univerzálneho problému na problém zo zadania.

q. To je ale podľa vyššie uvedenej vlastnosti práve vtedy, keď $w \in L(A)$, čo znamená, že výstup stroja pre L je aj výstupom stroja pre L_U. Schéma redukcie je na obrázku 8.

Zostáva doriešiť spôsob, ako skonštruovať stroj B a nejaký jeho stav q tak, aby platila vyššie uvedená vlastnosť a aby sa kódy $\langle B\rangle$, $\langle q\rangle$ dali z kódu stroja A algoritmicky (čiže na Turingovom stroji) zostrojiť. Jeden spôsob je napríklad nasledovný: stroj B bude na každom vstupe w pracovať ako stroj A. Ak ten akceptuje, stroj B prejde do nejakého *nového* stavu q. Takto definovaný stroj B má zjavne vlastnosť, že jeho výpočet na slove w sa dostane do stavu q práve vtedy, keď $w \in L(A)$. Od stroja A sa navyše líši iba pridaním jedného stavu a niekoľkých prechodov. Preto je táto transformácia algoritmicky realizovateľná. □

Zložitejšie riešené úlohy

Redukcie, ktoré sme použili pri riešení predchádzajúcich dvoch úloh mali vlastnosť, že transformácia kódu $\langle A\rangle$ Turingovho stroja A zo vstupu bola rovnaká pre všetky vstupné slová w. V nasledujúcich dvoch úlohách použijeme redukcie, ktoré túto vlastnosť nemajú a transformácia kódu Turingovho stroja závisí od jeho vstupu.

Úloha 4. Uvažujme rozhodovací problém daný nasledovne:

Vstup: Kód $\langle A\rangle$ deterministického Turingovho stroja A nad vstupnou abecedou $\{0,1\}$.

Výstup: „Áno“ práve vtedy, keď $\varepsilon \in L(A)$.

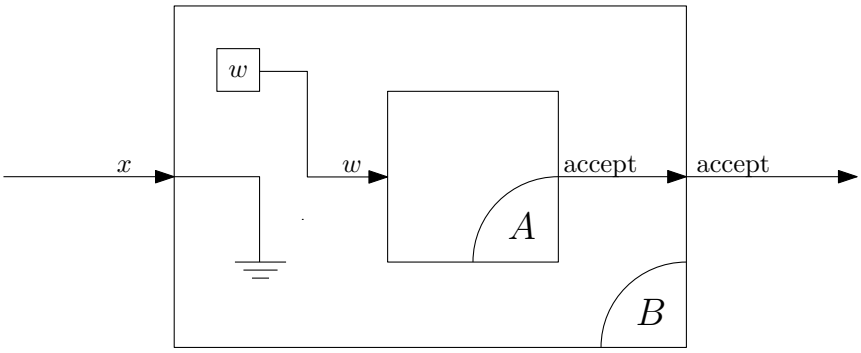
Zistite, či je uvedený problém rozhodnuteľný. Ak nie, je aspoň rekurzívne vyčísliteľný? Svoje tvrdenia dokážte.

Riešenie. Dokážeme najprv, že problém *nie je rozhodnuteľný*. Budeme postupovať redukciou univerzálneho problému na problém zo zadania.

Za účelom sporu predpokladajme, že problém zo zadania (reprezentovaný nejakým jazykom L_ε) je rozhodnuteľný, a teda existuje deterministický Turingov stroj akceptujúci jazyk L_ε, ktorý sa na každom vstupe zastaví. K sporu dospejeme tým, že na základe tohto stroja skonštruujeme stroj rozhodujúci univerzálny problém.

Ten dostane na vstupe kód $\langle A\rangle$ nejakého Turingovho stroja A nad vstupnou abecedou $\{0,1\}$ a slovo $w \in \{0,1\}^*$. Aby sme pomocou stroja rozhodujúceho L_ε vedeli rozhodnúť univerzálny problém, stačí na základe kódu $\langle A\rangle$ a slova w vyrobiť kód $\langle B\rangle$ Turingovho stroja B, ktorý má nasledujúcu vlastnosť: $w \in L(A)$ práve vtedy, keď $\varepsilon \in L(B)$. Je teda zrejmé, že stroj B musí závisieť nielen od A, ale aj od w. Presnejšie by sme teda namiesto B mohli písať aj $B(A,w)$.

Stroj B, ktorý skonštruujeme, bude v prípade $w \in L(A)$ akceptovať jazyk $\{0,1\}^*$ (a teda aj ε) a v opačnom prípade bude akceptovať prázdny jazyk. Pracovať bude nasledovne: dostane na vstupe slovo x, ktoré ihneď zahodí a nahradí ho slovom w, ktoré má „pevne zadrôtované v prechodovej

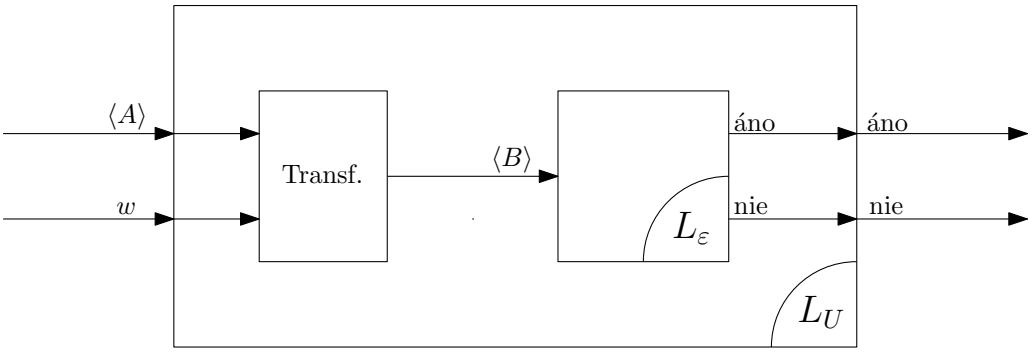


Obr. 9: Schematické znázornenie konštrukcie stroja B.

funkcii“. Následne odsimuluje výpočet stroja A na vstupe w a akceptuje práve vtedy, keď akceptuje stroj A. Schéma stroja B je na obrázku 9.

Čitateľovi dostatočne zbehlému v písaní prechodových funkcií Turingových strojov by malo byť zrejmé, že transformácia kódu $\langle A \rangle$ stroja A a slova w na kód $\langle B \rangle$ stroja B je iba otázkou vytrvalosti a „programátorskej zručnosti“. Malo by teda byť očividné, že sa táto transformácia dá zrealizovať na Turingovom stroji.

Stroj rozhodujúci univerzálny problém teda môže pracovať tak, že prerobí svoje vstupy $\langle A \rangle$, w na $\langle B \rangle$ a následne „zavolá“ stroj pre problém zo zadania so vstupom $\langle B \rangle$. Výstup tohto stroja je aj výstupom pre univerzálny problém. Schéma redukcie je na obrázku 10.



Obr. 10: Schéma redukcie univerzálneho problému na problém zo zadania.

Je očividné, že problém *je rekurzívne vyčísliteľný*, keďže tu stačí „zavolať“ univerzálny Turingov stroj so vstupmi $\langle A \rangle$ a ε. □

Úloha 5. Uvažujme rozhodovací problém daný nasledovne:

Vstup: Kód $\langle A \rangle$ deterministického Turingovho stroja A nad vstupnou abecedou $\{0, 1\}$.

Výstup: „Áno“ práve vtedy, keď $10011 \notin L(A)$.

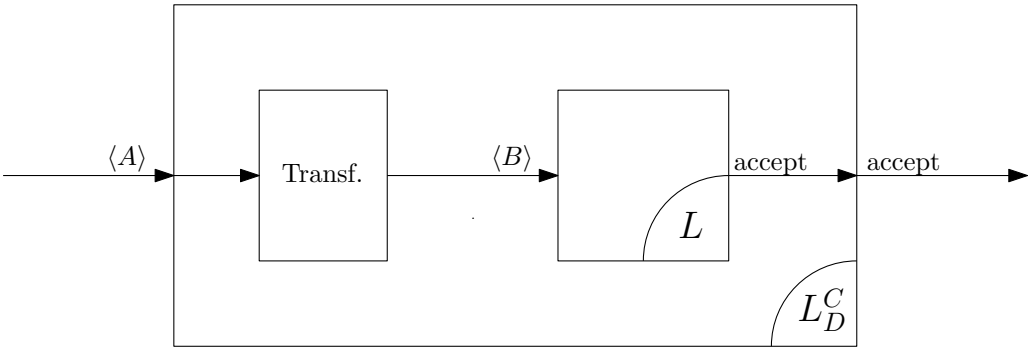
Zistite, či je uvedený problém rozhodnuteľný. Ak nie, je aspoň rekurzívne vyčísliteľný? Svoje tvrdenia dokážte.

Riešenie. Dokážeme, že tento problém *nie je rekurzívne vyčísliteľný* (a teda nemôže byť ani rozhodnuteľný). To je intuitívne zrejmé z nasledujúcej úvahy: pri simulácii prípadného nekonečného výpočtu stroja A na slove 10011 nikdy nemáme istotu, že je výpočet naozaj nekonečný a netreba na jeho skončenie čakať ešte o niečo dlhšie. V prípade nekonečného výpočtu na slove 10011 ale platí $10011 \notin L(A)$, pričom po prípadnom skončení výpočtu môže nastať aj situácia, keď $10011 \in L(A)$.

To znamená, že v rozhodovacom probléme zo zadania pravdepodobne nevieme s istotou povedať odpoveď „áno“, t.j. problém pravdepodobne nebude ani rekurzívne vyčísliteľný.

Uvedené odôvodnenie však nie je poriadnym dôkazom. Ten spravíme redukciou z komplementárneho diagonálneho problému. Nech problému zo zadania zodpovedá jazyk L. Dokážeme, že ak by existoval deterministický Turingov stroj akceptujúci jazyk L (t.j. ak by problém zo zadania bol rekurzívne vyčísliteľný), existoval by aj Turingov stroj akceptujúci jazyk L_D^C, čo je spor (pretože máme dokázané, že $L_D^C \notin \mathscr{L}_{RE}$).

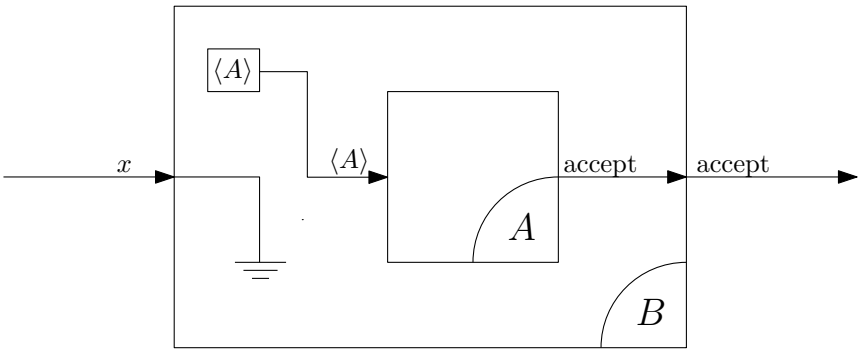
Vstupom komplementárneho diagonálneho problému je kód $\langle A \rangle$ nejakého Turingovho stroja A, pričom treba akceptovať ak $\langle A \rangle \notin L(A)$. Stroj akceptujúci jazyk L_D^C môže pracovať tak, že tento vstup transformuje na kód $\langle B \rangle$ Turingovho stroja B s vlastnosťou, že $10011 \notin L(B)$ práve vtedy, keď $\langle A \rangle \notin L(A)$. Na vstupe $\langle B \rangle$ potom už len spustí Turingov stroj pre jazyk L, ktorý akceptuje práve vtedy, keď $10011 \notin L(B)$, čo je podľa uvedenej vlastnosti stroja B práve vtedy, keď $\langle A \rangle \notin L(A)$. Schéma redukcie je na obrázku 11.



Obr. 11: Schéma redukcie komplementárneho diagonálneho problému na problém zo zadania.

Zostáva už len popísať konštrukciu stroja B a zdôvodniť, že je algoritmicky realizovateľná. Stroj B skonštruujeme tak, že v prípade $\langle A \rangle \in L(A)$ bude platiť $L(B) = \{0,1\}^*$ a v prípade $\langle A \rangle \notin L(A)$ bude platiť $L(B) = \emptyset$. Zrejme potom bude splnená aj vlastnosť, že $10011 \notin L(B)$ práve vtedy, keď $\langle A \rangle \notin L(A)$.

Stroj B bude pracovať tak, že svoj vstup x hneď na začiatku zahodí a nahradí ho slovom $\langle A \rangle$, ktoré má „pevne zadrôtované v prechodovej funkcii“. Na tomto slove potom spustí simuláciu stroja A, ktorého prechodovú funkciu má v sebe tiež „pevne zadrôtovanú“. Ak stroj A akceptuje, akceptuje aj stroj B.



Obr. 12: Schematické znázornenie konštrukcie stroja B.

Schéma konštrukcie stroja B je na obrázku 12. Vzťah jazyka akceptovaného strojom B k jazyku akceptovanému strojom A je zjavne taký, ako je uvedené vyššie. Navyše je zrejmé, že transformácia

kódu $\langle A \rangle$ na kód $\langle B \rangle$ je síce trochu komplikovaná, ale každopádne algoritmicky realizovateľná. Tvrdenie je dokázané. □

Na záver tohto oddielu ešte uveďme jednu úlohu, v ktorej je problém zo zadania rozhodnuteľný – z hľadiska náročnosti dôkazu ide o najľahší z možných prípadov.

Úloha 6. Uvažujme rozhodovací problém daný nasledovne:

Vstup: Kód $\langle A \rangle$ deterministického Turingovho stroja A nad vstupnou abecedou $\{0, 1\}$.

Výstup: „Áno“ práve vtedy, keď sa vo výpočte stroja A na vstupe ε vyskytne niektorý stav aspoň dvakrát.

Zistite, či je uvedený problém rozhodnuteľný. Ak nie, je aspoň rekurzívne vyčísliteľný? Svoje tvrdenia dokážte.

Riešenie. Problém *je rozhodnuteľný.* Každý Turingov stroj A má totiž konečný počet stavov k_A. Každý výpočet stroja A dĺžky aspoň k_A pozostáva z aspoň $k_A + 1$ konfigurácií, čo znamená, že nejaký stav sa nutne musel vyskytnúť aspoň dvakrát.

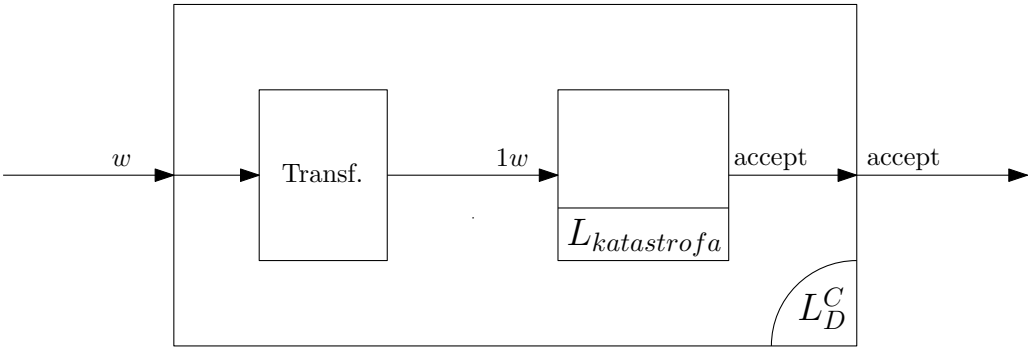
Stroj rozhodujúci problém zo zadania preto môže pracovať napríklad tak, že bude simulovať prvých k_A krokov výpočtu stroja A na vstupe ε. Ak sa tento výpočet zastavil pred vykonaním k_A-teho kroku, stačí overiť, či sa nejaký stav vyskytol dvakrát a podľa toho vrátiť odpoveď na výstup. V opačnom prípade sa nejaký stav dvakrát určite vyskytol, takže stačí na výstup vrátiť odpoveď „áno“. □

*** Jazyk, ktorý nie je v $\mathscr{L}_{RE}$ ani v co-$\mathscr{L}_{RE}$**

Uvažujme teraz jazyk $L_{katastrofa}$ definovaný takto:

$$L_{katastrofa} = \{0u \mid u \in \{0,1\}^*;\ u \in L_D\} \cup \{1v \mid v \in \{0,1\}^*;\ v \in L_D^C\}.$$

Dokážeme, že on ani jeho komplement nie sú rekurzívne vyčísliteľné. Inými slovami, jazyk $L_{katastrofa}$ nebude ani v $\mathscr{L}_{RE}$ ani v co-$\mathscr{L}_{RE}$.

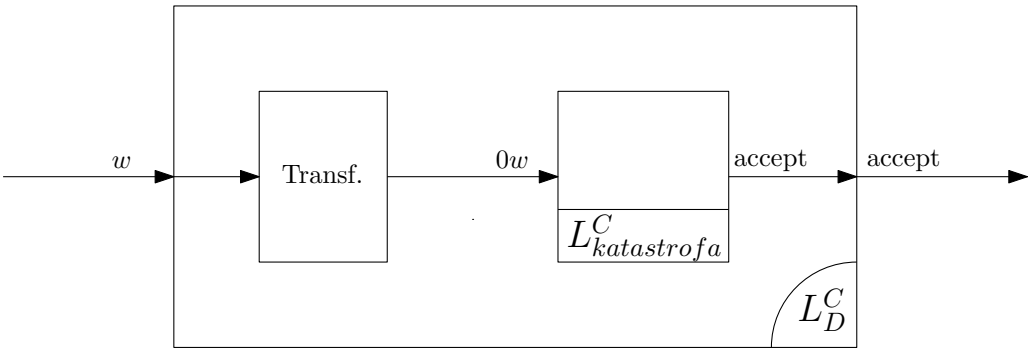


Obr. 13: Schéma redukcie komplementárneho diagonálneho problému na problém reprezentovaný jazykom $L_{katastrofa}$.

Dokážeme najprv, že $L_{katastrofa}$ nie je v $\mathscr{L}_{RE}$. Redukciou z komplementárneho diagonálneho problému. Ak by platilo $L_{katastrofa} \in \mathscr{L}_{RE}$, existoval by deterministický Turingov stroj M akceptujúci tento jazyk. Na základe stroja M by sme ale mohli skonštruovať Turingov stroj akceptujúci L_D^C, ktorý najprv vstup w upraví na $1w$ a na tomto vstupe odsimuluje stroj M. Ten akceptuje práve vtedy, keď $w \in L_D^C$, čo sme chceli dosiahnuť. Schéma tejto redukcie je znázornená na obrázku 13.

Dokážeme ešte, že jazyk

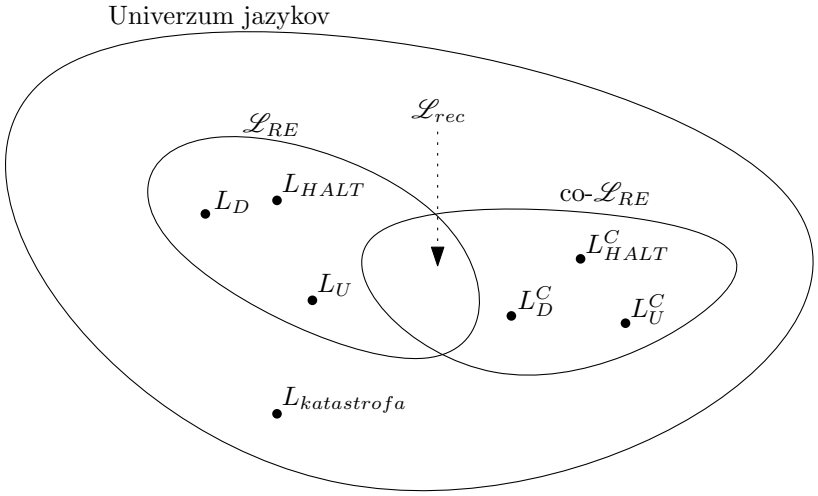
$$L_{katastrofa}^C = \{0u \mid u \in \{0,1\}^*;\ u \in L_D^C\} \cup \{1v \mid v \in \{0,1\}^*;\ v \in L_D\} \cup \{\varepsilon\}$$



Obr. 14: Schéma redukcie komplementárneho diagonálneho problému na problém reprezentovaný jazykom $L^C_{katastrofa}$.

nie je v $\mathscr{L}_{RE}$. Opäť budeme postupovať pomocou redukcie z komplementárneho diagonálneho problému. Ak by platilo $L^C_{katastrofa} \in \mathscr{L}_{RE}$, existoval by deterministický Turingov stroj M taký, že $L(M) = L^C_{katastrofa}$. Potom by sme ale vedeli skonštruovať Turingov stroj akceptujúci jazyk L^C_D, ktorý najprv vstup w upraví na $0w$ a na tomto vstupe odsimuluje stroj M. Stroj M zrejme akceptuje práve vtedy, keď $w \in L^C_D$. Schéma redukcie je na obrázku 14.

Jazyk $L_{katastrofa}$ teda skutočne nie je v $\mathscr{L}_{RE}$ ani v co-$\mathscr{L}_{RE}$. Situácia z obrázku 5 doplnená o pozíciu jazyka $L_{katastrofa}$ je na obrázku 15.



Obr. 15: Situácia z obrázku 5 doplnená o pozíciu jazyka $L_{katastrofa}$.