Postov korešpondenčný problém

Peter Kostolányi

21. marca 2017

Minulý semester bola na prednáške a na cvičeniach dokázaná *nerozhodnuteľnosť* viacerých rozhodovacích problémov, ako napríklad *univerzálneho problému*, *diagonálneho problému* a *problému zastavenia*. Neexistuje teda žiaden algoritmus – deterministický Turingov stroj zastavujúci na každom vstupe – ktorý by riešil niektorý z týchto problémov. V nasledujúcom k týmto výsledkom pridáme nerozhodnuteľnosť takzvaného *Postovho korešpondenčného problému*[1] (PKP).

Všetky nerozhodnuteľné problémy z minulého semestra predpokladajú ako svoj vstup kód nejakého Turingovho stroja; hoci má nerozhodnuteľnosť takýchto problémov niekoľko dôležitých implikácií, stále možno (snáď aj oprávnene) namietať, že ide o problémy viac-menej umelé. Postov korešpondenčný problém je naopak formulovaný veľmi jednoducho a čisto kombinatoricky. To nám neskôr umožní jeho využitie na dôkaz nerozhodnuteľnosti pomerne širokej škály ďalších problémov, často aj pomerne významných pre prax.

PKP a modifikovaný PKP

Nech Σ je abeceda. *Postov korešpondenčný problém (PKP)* nad abecedou Σ je daný nasledovne:

Vstup: Prirodzené číslo $n \geq 1$ a neprázdne slová $x_1, \ldots, x_n, y_1, \ldots, y_n \in \Sigma^+$.

Výstup: „Áno“ práve vtedy, keď existuje prirodzené číslo $k \geq 1$ a indexy $i_1, \ldots, i_k \in \{1, \ldots, n\}$ tak, že $x_{i_1} x_{i_2} \ldots x_{i_k} = y_{i_1} y_{i_2} \ldots y_{i_k}$.

Prípadom Postovho korešpondenčného problému nad abecedou Σ nazývame dvojicu (X, Y), kde $X = (x_1, \ldots, x_n)$ a $Y = (y_1, \ldots, y_n)$ pre nejaké prirodzené číslo $n \geq 1$ a slová $x_1, \ldots, x_n, y_1, \ldots, y_n$ zo Σ^+. Prípad PKP je teda reprezentáciou jednej jeho sady vstupov. Hovoríme, že prípad PKP *má riešenie*, ak je výstupom PKP na zodpovedajúcom vstupe „áno“ – teda ak existuje prirodzené číslo $k \geq 1$ a indexy $i_1, \ldots, i_k \in \{1, \ldots, n\}$ tak, že $x_{i_1} x_{i_2} \ldots x_{i_k} = y_{i_1} y_{i_2} \ldots y_{i_k}$. Postupnosť takýchto indexov $i_1, \ldots, i_k$ nazývame *riešením* daného prípadu PKP. Je ale dôležité nemýliť si existenciu *riešenia prípadu* PKP s *algoritmickou riešiteľnosťou* Postovho korešpondenčného problému – algoritmicky riešiť PKP znamená vedieť *pre každý prípad* PKP zistiť, či má riešenie.

Prípad Postovho korešpondenčného problému si teda možno predstaviť ako sadu n druhov „dominových dlaždíc“

$$\left[\begin{array}{c} x_1 \\ y_1 \end{array}\right] \left[\begin{array}{c} x_2 \\ y_2 \end{array}\right] \ldots \left[\begin{array}{c} x_n \\ y_n \end{array}\right],$$

pričom z každého druhu je k dispozícii neobmedzene veľa kusov. Prípad PKP má riešenie, ak existuje neprázdna postupnosť „dlaždíc“ taká, že po ich priložení vedľa seba vznikne na oboch „poschodiach“ po zreťazení rovnaké slovo. „Dlaždice“ k sebe možno prikladať iba svojou „dlhšou stranou“ a nemožno ich otáčať.

Príklad 1. Uvažujme prípad (X, Y) PKP nad $\Sigma = \{a, b\}$, kde $X = (aa, a, ba)$ a $Y = (a, aa, aa)$. Ten si možno predstaviť ako sadu „dlaždíc“

$$\left[\begin{array}{c} aa \\ a \end{array}\right] \left[\begin{array}{c} a \\ aa \end{array}\right] \left[\begin{array}{c} ba \\ aa \end{array}\right].$$

Ľahko možno uhádnuť, že tento prípad PKP má riešenie

$$\left[\begin{array}{c} aa \\ a \end{array}\right] \left[\begin{array}{c} a \\ aa \end{array}\right]:$$

[1] Alebo *Postovho problému priradenia*; prívlastok „korešpondenčný“ je odvodený od slova „korešpondencia“ vo význame „zhoda“.

na hornom aj na spodnom „poschodí“ je po zreťazení rovnaké slovo *aaa*. Toto riešenie neobsahuje ani jeden výskyt tretej „dlaždice“ zo vstupu. Ľahko ale vidieť, že žiadne riešenie obsahujúce tretiu „dlaždicu“ neexistuje – tá totiž obsahuje na hornom „poschodí“ symbol b, ktorý sa nevyskytuje na spodnom „poschodí“ žiadnej „dlaždice“.

Postov korešpondenčný problém *je* zjavne *rekurzívne vyčísliteľný* pre všetky abecedy Σ – stačí postupne skúšať všetky postupnosti „dlaždíc“ zo vstupu a akceptovať, ak niektorá z týchto postupností zodpovedá riešeniu. V nasledujúcom ale ukážeme, že Postov korešpondenčný problém *nie je rozhodnuteľný* pre žiadnu aspoň dvojprvkovú abecedu Σ. To znamená, že neexistuje žiaden algoritmus, ktorý by pre daný prípad PKP vedel rozhodnúť, či má riešenie.

Najprv ale dokážeme nerozhodnuteľnosť variantu Postovho korešpondenčného problému, ktorý budeme nazývať *modifikovaný Postov korešpondenčný problém* (MPKP). Tento výsledok potom využijeme na dôkaz nerozhodnuteľnosti samotného PKP.

Nech Σ je abeceda. *Modifikovaný Postov korešpondenčný problém (MPKP)* nad abecedou Σ je daný nasledovne:

Vstup: Prirodzené číslo $n \geq 1$ a neprázdne slová $x_1, \ldots, x_n, y_1, \ldots, y_n \in \Sigma^+$.

Výstup: „Áno“ práve vtedy, keď existuje prirodzené číslo k a indexy $i_1, \ldots, i_k \in \{1, \ldots, n\}$ tak, že $x_1 x_{i_1} x_{i_2} \ldots x_{i_k} = y_1 y_{i_1} y_{i_2} \ldots y_{i_k}$.

Prípad MPKP a jeho riešenie definujeme obdobne ako pre PKP. Pri interpretácii pomocou „dominových dlaždíc“ možno riešenia prípadu MPKP popísať ako riešenia zodpovedajúceho prípadu PKP, ktoré navyše začínajú prvou „dlaždicou“. Každá „sada dlaždíc“ totiž určuje prípad PKP aj prípad MPKP – pri prípade MPKP sú ale podmienky kladené na jeho riešenia silnejšie, než pri prípade PKP.

Nerozhodnuteľnosť modifikovaného PKP

Dokážeme najprv, že modifikovaný PKP je nerozhodnuteľný nad jednou konkrétnou päťprvkovou abecedou. Priamym dôsledkom tohto tvrdenia je – ako ukážeme neskôr – nerozhodnuteľnosť MPKP nad ľubovoľnou aspoň dvojprvkovou abecedou.

Veta 1. *Modifikovaný Postov korešpondenčný problém nad* $\Sigma = \{0, 1, \#, \$, Q\}$ *je nerozhodnuteľný.*

Dôkaz. Označme jazyk zodpovedajúci modifikovanému Postovmu korešpondenčnému problému nad abecedou Σ symbolom L_{MPKP}. *Redukciou problému zastavenia* na MPKP nad Σ dokážeme, že MPKP nad Σ nie je rozhodnuteľný, a teda $L_{\text{MPKP}} \notin \mathscr{L}_{rec}$.

Budeme predpokladať, že slovo na vstupe problému zastavenia je neprázdne. Problém očividne zostane nerozhodnuteľný aj po takejto úprave – na dôkaz stačí ku každému Turingovmu stroju A skonštruovať Turingov stroj A', ktorý „odignoruje“ prvé písmeno vstupu a so zvyškom vstupu pracuje rovnako ako stroj A. Výstup problému zastavenia na vstupoch $\langle A \rangle$ a w je potom rovnaký ako na vstupoch $\langle A' \rangle$ a $0w$; slovo $0w$ je pritom neprázdne.

Za účelom jednoduchšej manipulácie s okrajovými prípadmi tiež budeme uvažovať *normálny tvar* Turingových strojov, v ktorom sa čítacia hlava nikdy nehýbe doľava z najľavejšieho zapísaného políčka pásky. Ak teda hlava číta „blank“, vždy ide o „blank“ napravo od zapísanej časti pásky (za predpokladu neprázdnosti vstupu je na páske vždy aspoň jedno zapísané políčko). Dôkaz, že skutočne ide o normálny tvar, prenechávame čitateľovi ako jednoduché cvičenie. Z neho by mala vyplynúť aj skutočnosť, že transformáciu do opísaného normálneho tvaru možno realizovať algoritmicky. Preto možno bez ujmy na všeobecnosti predpokladať, že všetky Turingove stroje na vstupe problému zastavenia sú v uvedenom normálnom tvare – problém zostane aj naďalej nerozhodnuteľný.

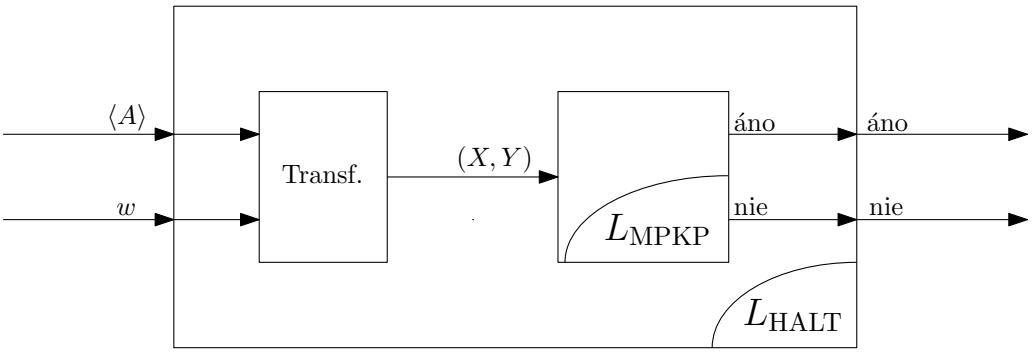
Vo zvyšku dôkazu sa budeme držať týchto dohovorov bez toho, aby sme na to zakaždým upozorňovali. Čitateľ by sa preto pred pokračovaním v dôkaze mal s predchádzajúcimi úvahami dôkladne zžiť.

Predpokladajme teda za účelom sporu, že modifikovaný Postov korešpondenčný problém nad abecedou $\Sigma = \{0, 1, \#, \$, Q\}$ je rozhodnuteľný – čiže $L_{\text{MPKP}} \in \mathscr{L}_{rec}$. Ukážeme, že v takom prípade musí byť rozhodnuteľný aj problém zastavenia (spor).

Skonštruujeme teda deterministický Turingov stroj, ktorý sa na každom vstupe zastaví a ktorý akceptuje jazyk L_{HALT} (zodpovedajúci nášmu variantu problému zastavenia). Vstupom takého stroja je kód $\langle A \rangle$ nejakého deterministického Turingovho stroja A nad vstupnou abecedou $\{0, 1\}$ a slovo $w \in \{0, 1\}^+$. Tieto vstupy treba akceptovať práve vtedy, keď sa výpočet stroja A na slove w zastaví.

Stroj rozhodujúci problém zastavenia môže pri rozhodovaní použiť stroj pre MPKP. Preto je potrebné prerobiť vstupy problému zastavenia na vhodný prípad MPKP tak, aby stroj pre problém zastavenia mohol nejakým spôsobom využiť výstup stroja pre MPKP.

Za týmto účelom zjavne postačí algoritmicky prerobiť vstupy $\langle A \rangle$ a w problému zastavenia na prípad $(X, Y) := (X(\langle A \rangle, w), Y(\langle A \rangle, w))$ MPKP nad Σ taký, že (X, Y) má riešenie práve vtedy, keď sa výpočet stroja A na w zastaví. Schéma takejto redukcie je znázornená na obrázku 1.



Obr. 1: Základná schéma redukcie problému zastavenia na MPKP nad abecedou Σ.

Zostáva teda nájsť spôsob, ako pre dané $\langle A \rangle$ a w zostrojiť zodpovedajúci prípad (X, Y). Idea konštrukcie bude spočívať v simulácii výpočtu stroja $A = (K, \Sigma, \Gamma, \delta, q_0, F)$ na slove w pomocou „dlaždíc“ tak, že na oboch „poschodiach“ sa budú postupne vytvárať platné postupnosti jeho konfigurácií. Pritom bude platiť, že počas simulácie výpočtu je horné „poschodie“ o jednu konfiguráciu pred spodným „poschodím“. Táto situácia je schematicky znázornená na obrázku 2.

#	C_0	#	C_1	#	C_2	#	$\cdots$	#	C_i	#	C_{i+1}	#
#	C_0	#	C_1	#	C_2	#	$\cdots$	#	C_i	#		

Obr. 2: Schéma simulácie výpočtu stroja A na jednotlivých „poschodiach“ vytváraných postupne prikladanými „dlaždicami“. Konfigurácia C_0 je počiatočná konfigurácia stroja A na slove w a pre všetky $j = 0, \ldots, i$ platí $C_j \vdash_A C_{j+1}$. „Prečnievajúcu“ konfiguráciu C_{i+1} možno využiť na zabezpečenie nadväznosti jednotlivých konfigurácií, ako uvidíme neskôr pri konštrukcií „dlaždíc“.

Špeciálny tvar jednotlivých „dlaždíc“ bude zabezpečovať, aby jednotlivé po sebe idúce konfigurácie boli navzájom v relácii $\vdash_A$. V prípade, že sa na hornom „poschodí“ vyskytne konfigurácia, z ktorej nemožno vo výpočte pokračovať, bude mať spodné „poschodie“ možnosť „dorovnať horné poschodie“, z čoho vyplýva, že daný prípad MPKP bude mať riešenie. To ale bude možné skutočne *iba* v prípade ukončenia výpočtu stroja A, čo znamená, že ak sa stroj A na slove w nezastaví, zodpovedajúci prípad MPKP nebude mať riešenie.

Budeme pracovať s definíciou konfigurácie ako slova, v ktorom stav určuje pozíciu hlavy. Teda napríklad $\mathbf{B}q_0w\mathbf{B}$ je počiatočná konfigurácia stroja A na slove w. „Pravý“ symbol $\mathbf{B}$ je v tejto definícii konfigurácie nadbytočný a píše sa iba z estetických dôvodov. Normálny tvar z úvodu tohto

dôkazu však umožňuje vypustiť aj „ľavý“ symbol **B**. Preto budeme pracovať s konfiguráciami bez ohraničujúcich symbolov **B**. Počiatočná konfigurácia stroja A na slove w teda bude q_0w.

Treba upozorniť na jeden dôležitý detail: konštruovať postupnosti takto definovaných konfigurácií stroja A je zjavne nemožné, keďže máme k dispozícii iba kód stroja $\langle A\rangle$, a teda iba binárne kódy jednotlivých stavov. Ak by sme v definícii konfigurácie nahradili symbol q priamo jeho binárnym kódom $\langle q\rangle$, došlo by k „pomiešaniu“ symbolov a stavov. Preto budeme v konfiguráciách namiesto symbolu pre stav q uvažovať reťazec $\$\langle q\rangle\$$, kde $\langle q\rangle$ je binárny kód stavu q. Podobná situácia nastane aj so symbolmi pracovnej abecedy, ktoré ale môžeme pri vhodnom kódovaní nahradiť priamo ich kódom. Namiesto počiatočnej konfigurácie stroja A na slove w teda budeme pracovať s reťazcom $\$\langle q_0\rangle\$\langle w\rangle$. Pri takejto reprezentácii už zjavne nemôže dôjsť k nejednoznačnostiam.[2] Na oddeľovanie jednotlivých konfigurácií bude slúžiť ďalší špeciálny symbol $\#$.

Môžeme pristúpiť ku konštrukcii jednotlivých „dlaždíc“ zodpovedajúceho prípadu MPKP:

Prvá „dlaždica“: Riešenie prípadu MPKP sa musí začínať prvou „dlaždicou“. To využijeme na to, aby sme zabezpečili, že sa postupnosť konfigurácií vytváraná postupným prikladaním „dlaždíc“ bude začínať počiatočnou konfiguráciou: ako prvú „dlaždicu“ vezmeme

$$\boxed{\begin{array}{c}\#\$\langle q_0\rangle\$\langle w\rangle\# \\ \hline \# \end{array}}.$$

Počiatočná konfigurácia stroja A je pre pevne dané slovo w iba jedna, preto možno takúto „dlaždicu“ na základe kódu $\langle A\rangle$ stroja A a slova w bez problémov zostrojiť. *Všetkých možných* konfigurácií stroja A je však nekonečne veľa, čo znamená, že pre ďalšie konfigurácie už nebudeme môcť mať samostatné „dlaždice“, ale budeme musieť zabezpečiť ich vytváranie „po častiach“.

Skupina 2: Každá ďalšia konfigurácia vznikne z predchádzajúcej konfigurácie iba lokálnou zmenou poblíž výskytu čítacej hlavy, pričom zvyšok konfigurácie sa iba skopíruje. Druhá skupina „dlaždíc“ bude obsahovať práve takéto „kopírovacie dlaždice“:

$$\boxed{\begin{array}{c}\langle c_1\rangle \\ \hline \langle c_1\rangle \end{array}} \cdots \boxed{\begin{array}{c}\langle c_m\rangle \\ \hline \langle c_m\rangle \end{array}} \boxed{\begin{array}{c}\# \\ \hline \# \end{array}},$$

kde $\Gamma = \{c_1, \ldots, c_m\}$. „Prečnievajúca konfigurácia na hornom poschodí“ pritom bude zabezpečovať, že skutočne pôjde o kopírovanie predchádzajúcej konfigurácie a nie o pridávanie ľubovoľných symbolov.

Skupina 3: „Dlaždice“ z tejto skupiny budú simulovať vlastný krok výpočtu – čiže lokálnu zmenu v konfigurácii, ktorá sa udeje v okolí políčka čítaného čítacou hlavou. Pre všetky $p, q \in K$ a všetky $a, b \in \Gamma$ také, že $\delta(p, a) = (q, b, 0)$ bude táto skupina „dlaždíc“ obsahovať „dlaždicu“

$$\boxed{\begin{array}{c}\$\langle q\rangle\$\langle b\rangle \\ \hline \$\langle p\rangle\$\langle a\rangle \end{array}}.$$

Podobne pre všetky $p, q \in K$ a všetky $a, b \in \Gamma$ také, že $\delta(p, a) = (q, b, 1)$ bude obsahovať „dlaždicu“

$$\boxed{\begin{array}{c}\langle b\rangle\$\langle q\rangle\$ \\ \hline \$\langle p\rangle\$\langle a\rangle \end{array}}$$

a pre všetky $p, q \in K$ a všetky $a, b \in \Gamma$ také, že $\delta(p, a) = (q, b, -1)$ bude pre všetky $c \in \Gamma$ obsahovať „dlaždicu“

$$\boxed{\begin{array}{c}\$\langle q\rangle\$\langle cb\rangle \\ \hline \langle c\rangle\$\langle p\rangle\$\langle a\rangle \end{array}}.$$

[2] Alternatívne by bolo možné pracovať s kódovaním Turingových strojov, pri ktorom nemôžu mať stavy a pracovné symboly rovnaké kódy. V takom prípade sa dá zaobísť aj bez symbolov \$.

V poslednom prípade si vystačíme s „dlaždicami“ tohto typu vďaka dohovoru o normálnom tvare z úvodu dôkazu. Uvedené „dlaždice“ tak riešia všetky situácie, keď hlava číta pracovný symbol. Zostáva doriešiť prípady, keď hlava číta „blank“ – ten musí byť vďaka spomínanému normálnemu tvaru napravo od zapísanej časti pásky.

Pre všetky $p, q \in K$ a $b \in \Gamma$ také, že $\delta(p, \mathbf{B}) = (q, b, 0)$ teda pridáme „dlaždicu“

$$\begin{array}{|c|}\hline \$\langle q\rangle\$\langle b\rangle\# \\ \hline \$\langle p\rangle\$\# \\ \hline \end{array}.$$

Podobne, pre všetky $p, q \in K$ a $b \in \Gamma$ také, že $\delta(p, \mathbf{B}) = (q, b, 1)$ pridáme „dlaždicu“

$$\begin{array}{|c|}\hline \langle b\rangle\$\langle q\rangle\$\# \\ \hline \$\langle p\rangle\$\# \\ \hline \end{array}$$

a pre všetky $p, q \in K$ a $b \in \Gamma$ také, že $\delta(p, \mathbf{B}) = (q, b, -1)$ pridáme pre všetky $c \in \Gamma$ „dlaždicu“

$$\begin{array}{|c|}\hline \$\langle q\rangle\$\langle cb\rangle\# \\ \hline \langle c\rangle\$\langle p\rangle\$\# \\ \hline \end{array}.$$

Skupina 4: V prípade, že na hornom „poschodí“ vznikne konfigurácia, z ktorej vo výpočte nemožno pokračovať ďalej, bude mať spodné „poschodie“ možnosť „dorovnať sa“. To však očividne nie je možné s použitím jedinej „dlaždice“, keďže posledná konfigurácia môže byť ľubovoľne dlhá. Preto je nutné v takejto situácii započať samostatnú fázu prikladania „dlaždíc“, ktorá sa bude vyznačovať tým, že kód stavu v konfigurácii bude nahradený špeciálnym symbolom Q. Ďalej sa už teda nebudú vytvárať konfigurácie v pravom slova zmysle, ale iba akési „kvázi konfigurácie“. „Dlaždice“ štvrtej skupiny zabezpečujú nahradenie kódu stavu symbolom Q v prípade, že v simulácii výpočtu nemožno ďalej pokračovať. Pre všetky $q \in K$ a $c \in \Gamma$ také, že $\delta(q, c) = \emptyset$ teda bude táto skupina obsahovať „dlaždicu“

$$\begin{array}{|c|}\hline Qc \\ \hline \$\langle q\rangle\$c \\ \hline \end{array}$$

a podobne, pre všetky $q \in K$ také, že $\delta(q, \mathbf{B}) = \emptyset$ bude obsahovať dlaždicu

$$\begin{array}{|c|}\hline Q\# \\ \hline \$\langle q\rangle\$\# \\ \hline \end{array}.$$

Skupina 5: „Dlaždice“ tejto skupiny budú umožňovať samotné „dorovnávanie“ oboch postupností konfigurácií tým, že symbol Q bude mať možnosť „mazať“ symboly vo svojom okolí. To znamená, že po konfigurácii, v ktorej sa stroj A zastaví, bude na oboch „poschodiach“ nasledovať postupnosť „kvázi konfigurácií“, kde každá vznikne zmazaním jedného pracovného symbolu z predchádzajúcej. Na konci tohto procesu vznikne „kvázi konfigurácia“ obsahujúca iba symbol Q.

Pre každý pracovný symbol $c \in \Gamma$ teda bude piata skupina obsahovať „dlaždice“

$$\begin{array}{|c|}\hline Q \\ \hline Q\langle c\rangle \\ \hline \end{array} \quad \begin{array}{|c|}\hline Q \\ \hline \langle c\rangle Q \\ \hline \end{array}.$$

Posledná „dlaždica“: „Dlaždica“

$$\begin{array}{|c|}\hline \# \\ \hline Q\#\# \\ \hline \end{array}.$$

umožní „dorovnanie oboch poschodí“ v prípade, že na hornom „poschodí“ vznikla „kvázi konfigurácia“ obsahujúca iba symbol Q.

Z uvedených zdôvodnení by malo byť zrejmé, že takto skonštruovaný prípad MPKP má riešenie práve vtedy, keď sa výpočet stroja A na vstupe w zastaví. Navyše možno ľahko nahliadnuť, že transformáciu kódu $\langle A \rangle$ a slova w na horeuvedenú sadu „dlaždíc“ možno realizovať algoritmicky. Tvrdenie je teda dokázané. □

Dôsledok 1. *Nech Σ je aspoň dvojprvková abeceda. Modifikovaný Postov korešpondenčný problém nad abecedou Σ je nerozhodnuteľný.*

Dôkaz. Ľahko možno nahliadnuť, že symboly $0, 1, \#, \$, Q$ možno algoritmicky zakódovať do ľubovoľnej abecedy obsahujúcej aspoň dva symboly tak, aby bolo dekódovanie *slov* jednoznačné. To znamená, že keby bol MPKP nad nejakou aspoň dvojprvkovou abecedou Σ rozhodnuteľný, bol by rozhodnuteľný aj MPKP nad abecedou $\{0, 1, \#, \$, Q\}$, čo je spor s predchádzajúcou vetou. □

Poznámka 1. Keďže je všetkých možných symbolov viac ako spočítateľne veľa (v skutočnosti ani neexistuje množina všetkých potenciálnych symbolov, keďže napríklad každú množinu možno vyhlásiť za symbol), je zrejmé, že nie je úplne korektné hovoriť o PKP resp. MPKP nad ľubovoľnou abecedou (takýto problém by sa nedal zakódovať do jazyka). Napriek tomu sa to v literatúre často robí a spravidla sa na túto skutočnosť ani neupozorňuje. Takto prezentované výsledky potom treba chápať jedným z nasledujúcich dvoch spôsobov. Buď sa vyberie spočítateľné univerzum všetkých objektov, ktoré možno chápať ako symboly a následne sa tieto symboly očíslujú. Potom možno narábať s kódmi jednotlivých symbolov podobne ako v prípade konečnej abecedy. Alebo možno poukázať na skutočnosť, že každý prípad PKP resp. MPKP možno „takmer jednoznačne“ zakódovať do binárnej abecedy, pričom takéto kódovanie je iné od prípadu k prípadu. Jazyk zodpovedajúci PKP nad ľubovoľnou abecedou by potom obsahoval takto zakódované prípady. Je zrejmé, že pri použití tohto prístupu sa časť informácie (konkrétne symboly) stráca, ale celá „podstatná informácia“ zostáva zachovaná.

Nerozhodnuteľnosť PKP (štandardná redukcia MPKP na PKP)

Vyššie sme dokázali nerozhodnuteľnosť MPKP (nad ľubovoľnou aspoň dvojprvkovou abecedou). Teraz využijeme túto skutočnosť na dôkaz, že ani PKP nie je rozhodnuteľný.

Veta 2. *Postov korešpondenčný problém nad abecedou $\Sigma = \{0, 1, \#\}$ je nerozhodnuteľný.*

Dôkaz. Redukciou MPKP nad binárnou abecedou na PKP nad Σ. Za účelom sporu predpokladajme, že je PKP nad abecedou Σ rozhodnuteľný. Ukážeme, že v takom prípade by bol rozhodnuteľný aj MPKP nad binárnou abecedou (spor).

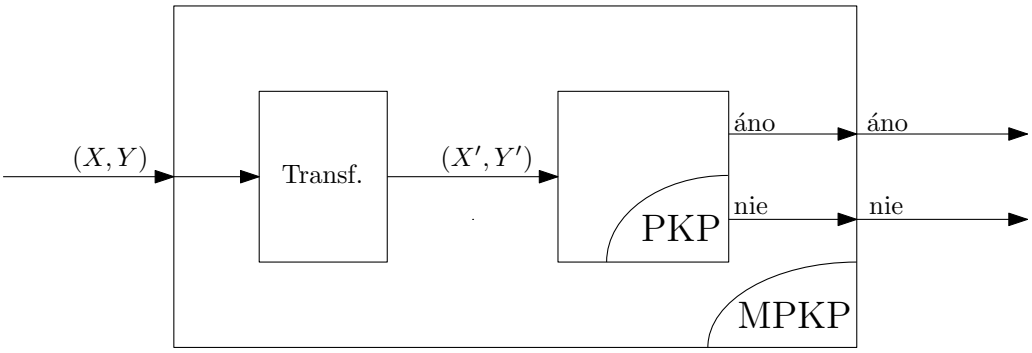
Schéma takejto redukcie je znázornená na obrázku 3. Turingov stroj rozhodujúci MPKP nad binárnou abecedou dostane na vstupe prípad MPKP (X, Y). Ten sa pomocou nejakej algoritmickej transformácie upraví na prípad (X', Y') PKP nad abecedou Σ taký, že (X', Y') má riešenie ako prípad PKP práve vtedy, keď (X, Y) má riešenie ako prípad MPKP. Na takto upravený vstup potom stroj pre MPKP zavolá stroj pre PKP (o ktorom predpokladáme, že existuje).

Zostáva ukázať, ako skonštruovať prípad (X', Y'). Nech $w \in \{0,1\}^+$ je slovo $w = a_1 a_2 \dots a_m$, $m \geq 1$. Symbolom $\overline{w}$ označíme slovo $\overline{w} = a_1 \# a_2 \# \dots \# a_m$. Následne využijeme skutočnosť, že ak sa má niektorá „dlaždica“ vyskytovať v riešení ako prvá, musia sa obidve jej „poschodia“ začínať rovnakým symbolom. „Dlaždice“ z pôvodnej sady pre MPKP upravíme vhodne pridanými symbolmi # tak, že túto vlastnosť bude spĺňať iba modifikácia prvej „dlaždice“ z pôvodnej sady, pričom ostatné „dlaždice“ sa budú môcť vyskytovať v riešení až za ňou. Rovnakú vlastnosť síce bude mať aj novopridaná „ukončovacia dlaždica“, ale pre tú bude ľahko vidieť, že sa na začiatku riešenia vyskytovať nemôže. Presnejšie: ak prvá „dlaždica“ pôvodnej sady pre MPKP je

$$\boxed{\begin{array}{|c|}\hline x_1 \\ \hline y_1 \\ \hline \end{array}},$$

nová sada pre PKP bude obsahovať dve zodpovedajúce „dlaždice“

$$\boxed{\begin{array}{|c|}\hline \#\overline{x_1}\# \\ \hline \#\overline{y_1} \\ \hline \end{array}} \quad \boxed{\begin{array}{|c|}\hline \overline{x_1}\# \\ \hline \#\overline{y_1} \\ \hline \end{array}},$$



Obr. 3: Základná schéma redukcie MPKP nad binárnou abecedou na PKP nad abecedou Σ.

pričom prvá z nich sa bude vyskytovať na začiatku prípadného riešenia. Pre všetky ostatné „dlaždice“

$$\left[\begin{array}{|c|}\hline x \\ \hline y \\ \hline\end{array}\right]$$

pôvodnej sady pre MPKP bude nová sada pre PKP obsahovať jednu zodpovedajúcu dlaždicu

$$\left[\begin{array}{|c|}\hline \overline{x}\# \\ \hline \#\overline{y} \\ \hline\end{array}\right],$$

ktorá sa zrejme na začiatku riešenia vyskytovať nemôže. Nakoniec pridáme ešte „ukončovaciu dlaždicu“

$$\left[\begin{array}{|c|}\hline \# \\ \hline \#\# \\ \hline\end{array}\right],$$

ktorá sa zjavne nemôže vyskytovať inde ako na konci riešenia (za predpokladu, že je toto riešenie minimálne). Dôkaz tvrdenia, že prípad PKP (X', Y') má riešenie práve vtedy, keď má riešenie prípad MPKP (X, Y), prenechávame čitateľovi. □

Dôsledok 2. *Nech Σ je aspoň dvojprvková abeceda. Postov korešpondenčný problém nad abecedou Σ je nerozhodnuteľný.*

Dôkaz. Rovnako ako pri dôsledku 1. □

Rovnakú redukciu ako pri dôkaze vety 2 možno aplikovať aj na redukciu MPKP nad ľubovoľnou abecedou Σ na PKP nad abecedou $\Sigma \cup \{\#\}$, kde $\#$ je nový symbol. Túto redukciu budeme nazývať *štandardnou redukciou* MPKP na PKP.

Úloha 1. Uvažujme prípad MPKP pozostávajúci z nasledujúcich dlaždíc:

$$\left[\begin{array}{|c|}\hline aab \\ \hline aa \\ \hline\end{array}\right] \left[\begin{array}{|c|}\hline ab \\ \hline bab \\ \hline\end{array}\right] \left[\begin{array}{|c|}\hline bba \\ \hline bb \\ \hline\end{array}\right] \left[\begin{array}{|c|}\hline ab \\ \hline a \\ \hline\end{array}\right] \left[\begin{array}{|c|}\hline a \\ \hline aab \\ \hline\end{array}\right].$$

Štandardnou konštrukciou zostrojte prípad PKP, ktorý má riešenie práve vtedy, keď ho má uvedený prípad MPKP.

Riešenie. Zodpovedajúci prípad PKP bude obsahovať nasledujúce „dlaždice“:

$$\left[\begin{array}{|c|}\hline \#a\#a\#b\# \\ \hline \#a\#a \\ \hline\end{array}\right] \left[\begin{array}{|c|}\hline a\#a\#b\# \\ \hline \#a\#a \\ \hline\end{array}\right] \left[\begin{array}{|c|}\hline a\#b\# \\ \hline \#b\#a\#b \\ \hline\end{array}\right] \left[\begin{array}{|c|}\hline b\#b\#a\# \\ \hline \#b\#b \\ \hline\end{array}\right] \left[\begin{array}{|c|}\hline a\#b\# \\ \hline \#a \\ \hline\end{array}\right] \left[\begin{array}{|c|}\hline a\# \\ \hline \#a\#a\#b \\ \hline\end{array}\right] \left[\begin{array}{|c|}\hline \# \\ \hline \#\# \\ \hline\end{array}\right]$$ □

Poznámka 2. Rovnako ako v poznámke 1 ešte podotknime, že uvedené výsledky sa v literatúre často zvyknú prezentovať v kontexte „PKP a MPKP nad ľubovoľnou abecedou“. Korektnú interpretáciu takto prezentovaných výsledkov sme už vysvetlili v poznámke 1.