Redukcie

Peter Kostolányi

21. marca 2017

Doposiaľ sme s pojmom redukcie pracovali v čisto intuitívnej rovine a pod redukciou rozhodovacieho problému P na rozhodovací problém R sme v zásade chápali dôkaz, že rozhodnuteľnosť (alebo rekurzívna vyčísliteľnosť) problému R implikuje rozhodnuteľnosť (alebo rekurzívnu vyčísliteľnosť) problému P. Táto pomerne nepresná predstava je postačujúca, ak je jediným cieľom dokázať nerozhodnuteľnosť problému P. Akonáhle sa však centrom pozornosti stanú redukcie samotné, je nutné podstatné spresnenie ich definície.

Problémom uvedenej naivnej „definície“ redukcie je predovšetkým jej nedostatočná prísnosť. Uvažujme napríklad ľubovoľný nerozhodnuteľný problém R. Podľa horeuvedenej „definície“ by potom nutne existovala redukcia ľubovoľného rozhodovacieho problému P na problém R, keďže v implikácii „ak R je rozhodnuteľný, tak aj P je rozhodnuteľný“ je ľavá strana nepravdivá, a teda implikácia ako celok je pravdivá. Tento poznatok však neposkytuje vôbec žiaden návod, ako s využitím existencie Turingovho stroja rozhodujúceho problém R skonštruovať Turingov stroj rozhodujúci problém P. Ukazuje sa dokonca, že existujú aj také jazyky P, pre ktoré je takáto konštrukcia principiálne nemožná.[1]

Tento nedostatok je pomerne vážny. Pri všetkých redukciách rozhodovacích problémov, s ktorými sme sa doposiaľ stretli, sme totiž kládli dôraz na skutočnosť, že v prípade rozhodnuteľnosti problému R vieme *skonštruovať* stroj rozhodujúci problém P, ktorý môže využívať (hyptotetický) stroj pre problém R ako „podprogram“. V nasledujúcom preto zavedieme pojem tzv. *turingovskej redukcie*, ktorá zachycuje práve túto vlastnosť danej dvojice problémov. Ak budeme neskôr hovoriť o „redukcii“ (bez ďalšieho prívlastku), vždy budeme mať na mysli turingovskú redukciu.

Takto definovaný pojem turingovskej redukcie navyše umožňuje *klasifikáciu* nerozhodnuteľných problémov podľa ich obtiažnosti. Pre turingovskú redukciu už totiž neplatí, že na každý nerozhodnuteľný problém R možno redukovať všetky rozhodovacie problémy P. Naopak, ak možno rozhodovací problém P redukovať na rozhodovací problém R, je to známkou toho, že problém P je *najviac taký ťažký* ako problém R.

Neskôr na niekoľkých riešených úlohách preskúmame vzájomnú (turingovskú) redukovateľnosť niektorých variantov Postovho korešpondenčného problému. V samotnom závere týchto poznámok potom zavedieme dva ďalšie typy redukcií, ktoré sú ešte prísnejšie, než turingovská redukcia; na tomto predmete s nimi síce ďalej pracovať nebudeme, v teórii vypočítateľnosti sú však užitočným nástrojom na klasifikáciu neriešiteľných problémov (jemnejšiu, než v prípade turingovskej redukcie).

Turingovská redukcia a turingovská ekvivalencia

Turingovská redukcia problému P na problém R je teda formalizáciou intuitívnej predstavy o tom, že keby sme mali dovolené využívať „podprogram“ rozhodujúci problém R (kde tento predpoklad je samozrejme čisto hypotetický), vedeli by sme rozhodovať aj problém P. Na formálnu definíciu turingovskej redukcie síce budeme potrebovať pojem deterministického Turingovho stroja s orákulom, v ňom však čitateľ istotne bez väčších problémov nájde paralelu s intuitívnym ponímaním vysvetleným vyššie. Pri aplikáciách pojmu turingovskej redukcie už budeme pracovať výlučne v neformálnej rovine.

Nech $L \subseteq \Sigma^*$ je ľubovoľný jazyk. *Deterministický Turingov stroj s orákulom* L je deterministický Turingov stroj so „zázračnou“ znalosťou príslušnosti slov do jazyka L. Presnejšie: ide o deterministický Turingov stroj, ktorý má navyše k dispozícii jednu špeciálnu dopytovaciu pásku (s ktorou pracuje ako s bežnou pracovnou páskou) a tri špeciálne stavy $q_?$, q_{yes} a q_{no}. Ak sa stroj dostane do stavu $q_?$, pričom obsahom dopytovacej pásky je v takejto konfigurácii slovo $x \in \Sigma^*$, v ďalšom kroku stroj „automaticky“ prejde do stavu q_{yes} (ak $x \in L$) resp. q_{no} (ak $x \notin L$). Obsah

[1] Formalizácia tohto tvrdenia využíva pojem turingovskej redukcie definovaný nižšie. Jeho dôkaz ale presahuje rámec tohto predmetu.

dopytovacej pásky sa v takomto kroku môže zmazať, prípadne môže ostať nezmenený (ide o technický detail, ktorý nijako nemení silu výsledného modelu). Formálnu definíciu deterministického Turingovho stroja s orákulom ako usporiadanej k-tice, ako aj definíciu konfigurácie, kroku výpočtu a akceptovaného jazyka, prenechávame čitateľovi ako jednoduché (avšak silno odporúčané) cvičenie.

Takto definovaný Turingov stroj s orákulom umožňuje zaviesť pojem turingovskej redukcie tak, ako bol neformálne opísaný vyššie.

Definícia 1. Nech P, R sú rozhodovacie problémy a $L_P, L_R \subseteq \Sigma^*$ sú im zodpovedajúce jazyky. Hovoríme, že problém P je *turingovsky redukovateľný* na problém R, ak existuje na každom vstupe zastavujúci deterministický Turingov stroj A s orákulom L_R taký, že $L(A) = L_P$. V takom prípade píšeme $P \leq_T R$ resp. $L_P \leq_T L_R$.

Zápis $P \leq_T R$ možno čítať aj ako „problém P je ľahší alebo rovnako ťažký ako problém R“ (vzhľadom na turingovskú redukciu). Odôvodnenie posktuje nasledujúca veta, ktorá je kľúčového významu.

Veta 1. *Nech P, R sú rozhodovacie problémy také, že $P \leq_T R$. Ak je problém R rozhodnuteľný, tak je rozhodnuteľný aj problém P.*

Dôkaz. Ľahko možno nahliadnuť, že v prípade rozhodnuteľnosti problému R možno každý deterministický Turingov stroj s orákulom L_R prerobiť na ekvivalentný deterministický Turingov stroj (bez orákula). Stačí každé „volanie“ orákula nahradiť „volaním“ deterministického Turingovho stroja rozhodujúceho R. □

Z naznačeného dôkazu vety 1 vyplýva aj skutočnosť, že v prípade existencie stroja rozhodujúceho problém R je možné skonštruovať stroj rozhodujúci problém P, ktorý využíva stroj pre R ako „podprogram“. Intuitívny význam turingovskej redukcie je teda skutočne taký, ako bolo avizované vyššie.

Pojem turingovskej redukcie možno okrem iného využiť na klasifikáciu nerozhodnuteľných problémov podľa obtiažnosti. Základom na vybudovanie takejto teórie[2] je zavedenie pojmu turingovskej ekvivalencie.

Definícia 2. Nech P, R sú rozhodovacie problémy a $L_P, L_R \subseteq \Sigma^*$ sú im zodpovedajúce jazyky. Hovoríme, že problém P je *rovnako ťažký* ako problém R (vzhľadom na turingovskú redukciu), ak $P \leq_T R$ a zároveň $R \leq_T P$. V takom prípade píšeme $P \equiv_T R$ resp. $L_P \equiv_T L_R$ a hovoríme tiež, že problém P je *turingovsky ekvivalentný* problému R.

V nasledujúcom zhrnieme niekoľko elementárnych vlastností turingovskej redukcie:

- Ak $L_1, L_2 \in \mathscr{L}_{rec}$, tak $L_1 \equiv_T L_2$. Ak totiž $L_1 \in \mathscr{L}_{rec}$, tak existuje deterministický Turingov stroj A_1, ktorý sa na každom vstupe zastaví a pre ktorý platí $L(A_1) = L_1$. Ľahko vidieť, že stroj A_1 možno prerobiť na ekvivalentný deterministický Turingov stroj s orákulom L_2 – ten bude jednoducho simulovať stroj A_1 a orákulum nikdy nezavolá. Preto platí $L_1 \leq_T L_2$. Na dôkaz $L_2 \leq_T L_1$ môžno použiť symetrickú argumentáciu.

- Ak $L_1 \in \mathscr{L}_{rec}$ a $L_2 \notin \mathscr{L}_{rec}$, tak nutne platí $L_1 \leq_T L_2$ a neplatí $L_2 \leq_T L_1$. Dôkaz prvého tvrdenia je rovnaký ako v predchádzajúcom prípade. Na dôkaz druhého tvrdenia si stačí uvedomiť, že keby platilo $L_2 \leq_T L_1$, bol by problém L_2 podľa vety 1 tiež rozhodnuteľný, čo je spor.

- Ak $L_1, L_2 \in \mathscr{L}_{RE} - \mathscr{L}_{rec}$, tak vo všeobecnosti *nemusí platiť* $L_1 \equiv_T L_2$. Dá sa dokonca dokázať, že existuje dvojica jazykov $L_1, L_2 \in \mathscr{L}_{RE} - \mathscr{L}_{rec}$ takých, že neplatí ani $L_1 \leq_T L_2$, ani $L_2 \leq_T L_1$. Tento výsledok je známy aj ako Friedbergova-Mučnikova veta a jeho dôkaz výrazne presahuje rámec tohto predmetu. Napriek tomu je dobré mať na zreteli skutočnosť, že rekurzívna vyčísliteľnosť dvoch nerozhodnuteľných problémov ešte neimplikuje, že obidva problémy sú rovnako ťažké (vzhľadom na turingovskú redukciu).

[2] Ktorá však z väčšej časti presahuje rámec tohto predmetu.

- Pre všetky jazyky L platí $L \equiv_T L^C$. Na rozhodovanie problému zodpovedajúceho jazyku L totiž stačí pre každý vstup w „zavolať“ orákulum L^C a zistiť tak, či $w \in L^C$ – potom už iba stačí výstup orákula znegovať. N. B.: ak $L \in \mathscr{L}_{RE} - \mathscr{L}_{rec}$, tak $L^C \notin \mathscr{L}_{RE}$; napriek tomu sú zodpovedajúce problémy (vzhľadom na turingovskú redukciu) rovnako ťažké.
- Relácia $\leq_T$ je očividne reflexívna a tranzitívna, no nie je antisymetrická – preto *nie je čiastočným usporiadaním.*
- Relácia $\equiv_T$ *je reláciou ekvivalencie* na triede všetkých rozhodovacích problémov (jazykov). Triedy ekvivalencie relácie $\equiv_T$ sa nazývajú *Turingove triedy.*
- Reláciu $\leq_T$ možno prirodzene rozšíriť na Turingove triedy. V takom prípade už *je čiastočným usporiadaním.* To vyplýva zo skutočnosti, že porušenie antisymetrie pre dvojicu problémov znamená, že tieto problémy sú rovnako ťažké (vzhľadom na turingovskú redukciu).

Poznámka 1. Aj keď je koncept Turingovho stroja s orákulom nutný na formálnu definíciu turingovskej redukcie, v nasledujúcom ho nahradíme intuitívnejším konceptom algoritmov, ktoré majú možnosť „zavolať podprogram“ zodpovedajúci orákulu. V prípade, že prijmeme Turingovu tézu, sú oba tieto koncepty navzájom ekvivalentné. To znamená, že pre dva rozhodovacie problémy P, R platí $P \leq_T R$ práve vtedy, keď existuje algoritmus rozhodujúci problém P za predpokladu, že má možnosť zavolať ako „podprogram“ (hypotetický) algoritmus rozhodujúci problém R. To súhlasí s intuitívnou predstavou o redukciách, ako sme ich na tomto predmete používali doposiaľ.

Príklad 1. Dokážeme, že modifikovaný Postov korešpondenčný problém je rovnako ťažký (vzhľadom na turingovskú redukciu) ako Postov korešpondenčný problém. Skutočnosť MPKP $\leq_T$ PKP sme už v podstate dokázali v poznámkach k Postovmu korešpondenčnému problému – stačí nahliadnuť, že redukcia, ktorú sme tam použili, je turingovská. To je však pomerne očividné, keďže stačí transformovať prípad MPKP na zodpovedajúci prípad PKP a naň jedenkrát „zavolať podprogram“ (resp. orákulum) pre PKP.

Dokážme teraz, že PKP $\leq_T$ MPKP. Za týmto účelom treba popísať algoritmus, ktorý rozhoduje PKP za predpokladu, že má možnosť pristupovať k „podprogramu“ rozhodujúcemu MPKP. Ten ale môže pracovať tak, že postupne zavolá MPKP pre všetky možné voľby prvej „dlaždice“ v danom prípade PKP. Ak je pre aspoň jeden z týchto prípadov MPKP výstupom „áno“, daný prípad PKP má riešenie; v opačnom prípade zjavne riešenie nemá.

Riešené úlohy

V nasledujúcom vyriešime niekoľko ukážkových úloh, v ktorých je cieľom zistiť, či sú dané varianty Postovho korešpondenčného problému rovnako ťažké (vzhľadom na turingovskú redukciu) ako „klasický“ PKP. Buď teda treba dokázať, že pre daný variant PKP (označme ho PKP$'$) platí PKP$'$ $\leq_T$ PKP a zároveň PKP $\leq_T$ PKP$'$, alebo treba ukázať, že niektorým smerom redukciu urobiť nemožno. Jedinou metódou, ktorú pritom máme k dispozícii na *vyvrátenie* turingovskej redukovateľnosti je dôkaz, že jeden z problémov je rozhodnuteľný, kým druhý nie je. Treba však mať na pamäti, že existujú aj iné dvojice problémov, ktoré nie sú rovnako ťažké (hoci dôkaz tejto skutočnosti by vyžadoval použitie pomerne pokročilých techník).

V riešeniach nasledujúcich úloh nebudeme venovať zvláštnu pozornosť skutočnosti, že použité redukcie sú turingovské. Namiesto toho sa vrátime k intuitívnemu ponímaniu redukcií, ako sme ich používali doposiaľ – „turingovskosť“ redukcií bude zväčša očividná. Je však dobré mať na pamäti, že v pozadí je formálne definovaný pojem turingovskej redukcie (obzvlášť preto, aby nenastali žiadne nedorozumenia ohľadom významu pojmu „rovnako ťažký“).

Úloha 1. Uvažujme rozhodovací problém PKP$'$, kde na vstupe je prípad PKP a treba rozhodnúť, či má daný prípad PKP riešenie dĺžky aspoň dva. Zistite, či je tento problém rovnako ťažký ako PKP (vzhľadom na turingovskú redukciu) a svoje tvrdenie dokážte.

Riešenie. Problémy *sú rovnako ťažke* – platí teda PKP$'$ $\equiv_T$ PKP. Obidve redukcie sú dokonca triviálne, keďže prípad PKP má riešenie práve vtedy, keď má riešenie dĺžky dva (v prípade riešenia

pozostávajúceho z jednej „dlaždice“ je riešením aj „priloženie“ dvoch takýchto „dlaždíc“ za sebou). To znamená, že na rozhodnutie prípadu PKP$'$ stačí na nezmenenom vstupe „zavolať“ (hypotetický) algoritmus rozhodujúci PKP a na rozhodnutie prípadu PKP stačí na nezmenenom vstupe „zavolať“ (hypotetický) algoritmus rozhodujúci PKP$'$. □

Úloha 2. Uvažujme rozhodovací problém PKP$'$, kde na vstupe je prípad PKP s jednou označenou „dlaždicou“ a treba rozhodnúť, či má daný prípad PKP riešenie dĺžky aspoň dva také, že predposledná „dlaždica“ v tomto riešení je označená. Zistite, či je tento problém rovnako ťažký ako PKP (vzhľadom na turingovskú redukciu) a svoje tvrdenie dokážte.

Riešenie. Problémy *sú rovnako ťažke* – platí teda PKP$' \equiv_T$ PKP. Je vcelku očividné, že keby sme vedeli rozhodovať PKP$'$, vedeli by sme rozhodovať aj PKP. Stačilo by totiž pre daný prípad PKP vyskúšať všetky možné voľby označenej „dlaždice“ a pre každú „zavolať“ algoritmus rozhodujúci PKP$'$. Ak je pre nejakú voľbu označenej „dlaždice“ výstupom „áno“, riešenie daného prípadu PKP$'$ je súčasne aj riešením pôvodného prípadu PKP. Naopak, ak má prípad PKP riešenie dĺžky aspoň dva, stačí označiť „dlaždicu“, ktorá je v ňom predposledná a zodpovedajúci prípad PKP$'$ má riešenie. Stačí sa teda odvolať na skutočnosť z riešenia predchádzajúcej úlohy: ak má prípad PKP riešenie, tak má aj riešenie dĺžky aspoň dva. Dokázali sme teda, že platí PKP $\leq_T$ PKP$'$.

Ostáva ukázať, že keby sme vedeli rozhodovať PKP, vedeli by sme rozhodovať aj PKP$'$. Majme daný prípad PKP$'$. Ten upravíme na prípad PKP pomocou omriežkovania, podobne ako pri štandardnej redukcii MPKP na PKP. Výsledná sada bude obsahovať jednu počiatočnú „dlaždicu“

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

pričom neskôr bude zrejmé, že sa bude môcť v riešení vyskytovať iba raz, a to na začiatku riešenia. Ďalej, pre každú „dlaždicu“

$$\boxed{\begin{array}{|c|}\hline u \\ \hline v \\ \hline \end{array}}$$

z prípadu PKP$'$ bude zodpovedajúci prípad PKP obsahovať „dlaždicu“

$$\boxed{\begin{array}{|c|}\hline \overline{u}\# \\ \hline \#\overline{v} \\ \hline \end{array}},$$

kde $\overline{a_1 a_2 \dots a_n} = a_1 \# a_2 \# \dots \# a_n$. A nakoniec, ak

$$\boxed{\begin{array}{|c|}\hline u_{ozn} \\ \hline v_{ozn} \\ \hline \end{array}}$$

je označená „dlaždica“ v prípade PKP$'$, tak pre každú ďalšiu „dlaždicu“

$$\boxed{\begin{array}{|c|}\hline u \\ \hline v \\ \hline \end{array}}$$

tohto prípadu PKP$'$ bude zodpovedajúci prípad PKP obsahovať „zdvojenú dlaždicu“

$$\boxed{\begin{array}{|c|}\hline \overline{u_{ozn}}\overline{u}\# \\ \hline \#\overline{v_{ozn}}\overline{v}\# \\ \hline \end{array}}.$$

Každé riešenie takto skonštruovaného prípadu PKP musí zjavne začínať počiatočnou „dlaždicou“ a končiť „zdvojenou dlaždicou“, pričom inak sa v ňom môžu vyskytovať už iba „obyčajné dlaždice“ (za predpokladu, že je dané riešenie minimálne). „Zdvojená dlaždica“ na konci riešenia zjavne zabezpečí, že v zodpovedajúcom riešení PKP$'$ je na predposlednej pozícii skutočne označená „dlaždica“. Platí teda aj PKP$' \leq_T$ PKP. □

Úloha 3. Uvažujme rozhodovací problém PKP$'$, kde na vstupe je prípad PKP *pozostávajúci z jedinej „dlaždice“* a treba rozhodnúť, či má daný prípad PKP riešenie. Zistite, či je tento problém rovnako ťažký ako PKP (vzhľadom na turingovskú redukciu) a svoje tvrdenie dokážte.

Riešenie. Problémy *nie sú rovnako ťažké*. Problém PKP$'$ je totiž zjavne rozhodnuteľný (jeho prípad má riešenie práve vtedy, keď daná „dlaždica“ obsahuje na oboch „poschodiach“ rovnaké slovo), a teda pre nerozhodnuteľný PKP nemôže platiť PKP $\leq_T$ PKP$'$. □

* Ďalšie druhy redukcií

V teórii vypočítateľnosti sa okrem turingovskej redukcie intenzívne študujú aj ďalšie, prísnejšie druhy redukcií. Dôležitým príkladom takýchto redukcií sú tzv. *„many-one“ redukcie*. Podobne ako pri turingovskej redukcii, problém P je „many-one“-redukovateľný na problém R, ak za predpokladu možnosti pristupovať k „podprogramu“ rozhodujúcemu problém R existuje algoritmus rozhodujúci problém P. Tento „podprogram“ však možno „zavolať“ iba raz a jeho výstup musí byť aj výstupom algoritmu rozhodujúceho P (výstupy teda napríklad nemožno „prevrátiť“).

Definícia 3. Nech P, R sú rozhodovacie problémy a $L_P, L_R \subseteq \Sigma^*$ sú im zodpovedajúce jazyky. Hovoríme, že problém P je *„many-one“-redukovateľný* na problém R, ak existuje vypočítateľná[3] funkcia $f\colon \Sigma^* \to \Sigma^*$ taká, že pre všetky $w \in \Sigma^*$ platí $w \in L_P$ práve vtedy, keď $f(w) \in L_R$. V takom prípade píšeme $P \leq_m R$ resp. $L_P \leq_m L_R$.

Napríklad štandardná redukcia MPKP na PKP je „many-one“ redukciou. Funkcia f v takom prípade realizuje štandardnú transformáciu prípadu MPKP na zodpovedajúci prípad PKP. Naopak redukcia PKP na MPKP z príkladu 1 nie je „many-one“ redukciou, keďže k „podprogramu“ pre MPKP sa v nej pristupuje niekoľkokrát.

Pomenovanie „many-one“ redukcia vychádza zo skutočnosti, že funkcia f v jej definícii môže zobraziť aj viacero vstupov problému P na jeden rovnaký vstup problému R. V prípade, že je táto funkcia f injektívna, hovoríme o *„one-one“ redukcii*.

Definícia 4. Nech P, R sú rozhodovacie problémy a $L_P, L_R \subseteq \Sigma^*$ sú im zodpovedajúce jazyky. Hovoríme, že problém P je *„one-one“-redukovateľný* na problém R, ak existuje vypočítateľná injektívna funkcia $f\colon \Sigma^* \to \Sigma^*$ taká, že pre všetky $w \in \Sigma^*$ platí $w \in L_P$ práve vtedy, keď $f(w) \in L_R$. V takom prípade píšeme $P \leq_1 R$ resp. $L_P \leq_1 L_R$.

Je zrejmé, že ak pre dvojicu rozhodovacích problémov P, R platí $P \leq_1 R$, tak platí aj $P \leq_m R$ a ak platí $P \leq_m R$, tak platí aj $P \leq_T R$. Tieto implikácie však nemožno obrátiť. Z toho dôvodu sú „many-one“ a „one-one“ redukcie užitočnými nástrojmi na klasifikáciu nerozhodnuteľných problémov podľa ich obtiažnosti, keďže táto klasifikácia je nutne jemnejšia, než je tomu pri klasifikácii na základe turingovskej redukovateľnosti.

[3]Funkcia $f\colon \Sigma^* \to \Sigma^*$ je vypočítateľná, ak existuje deterministický Turingov stroj so vstupnou a výstupnou páskou taký, že pre všetky obsahy vstupnej pásky $w \in \Sigma^*$ sa stroj v konečnom čase zastaví v konfigurácii, kde obsahom výstupnej pásky je slovo $f(w)$.