OD PROBLÉMOV KU PROGRAMOM II

(dokončenie)

JOZEF HVORECKÝ, JOZEF KELEMEN, Bratislava

5. Odstraňovanie chýb

Tvorba postupu riešenia daného problému prebieha tak, že využijúc poznatky o riešenom probléme (pozri časť 3) a svoju skúsenosť s tvorbou postupov riešení, vyberáme na každom kroku tvorby niektoré spomedzi spôsobov plánovania, ktoré sú znázornené na *obr. 2.* Vybraný spôsob potom použijeme na vykonanie ďalšieho kroku tvorby. Takto dostaneme krok po kroku plán postupu riešenia problému. V opísanom procese sa môžeme dopustiť omylov, ktoré sa vo vytvorenom postupe riešenia prejavujú ako chyby. Ich existenciu zistíme zvyčajne na základe jednej z dvoch vlastností postupu riešenia:

1. získaný postup riešenia je nerealizovateľný (napr. v prípade, keď postupy riešení majú tvar programov, počítač sa nezastavuje — program sa *zacyklil*), alebo

2. výsledok realizácie postupu riešenia nie je riešením nášho problému.

Ak nastane jeden z dvoch uvedených prípadov, nasleduje po etape plánovania druhá etapa procesu tvorby postupov riešení — odstraňovanie chýb. Na *obr. 1* je znázornená prerušovanými čiarami.

Odstrániť chybu znamená predovšetkým *nájsť* ju. Len čo ju nájdeme, pristupujeme k jej *oprave.*

Nàjsť chybu znamená dať odpoveď na dve otázky: kde sa chyba nachádza (v ktorej časti postupu riešenia), a o akú chybu ide (zistiť druh chyby).

Zistiť, kde sa chyba nachádza — *lokalizovať* ju — možno zvyčajne analýzou vytvoreného postupu riešenia a zistením tej jej časti, o ktorej

predpokladáme, že má za následok neúspech typu (1) alebo (2)[1]. Spätným sledovaním procesu tvorby postupu riešenia problému sa usilujeme odhaliť, v ktorej etape sme sa dopustili omylu. Náš omyl môže spočívať alebo v nesprávnom, alebo nevhodnom použití pravidla tvorby postupu riešenia. Tretím zdrojom chýb a príčinou neúspechov pri realizácii vytvorených postupov riešenia je nedostatočná pozornosť venovaná poznatkom, úzko zviazaným s problémovým prostredím, v ktorom riešime náš konkrétny problém.

5.1 Druhy chýb

V tejto časti sa zameriame na vymedzenie tých druhov chýb, ktorých výskyt vyplýva z nevhodného alebo nesprávneho použitia niektorého z pravidiel tvorby, prípadne z nedostatočnej pozornosti, venovanej špecifikám problémového prostredia (pozri časť 3). Chyby ktoré majú spomínané príčiny, budeme, v zhode s terminológiou z [6] a [9], nazývať *racionálnymi* chybami. Odlíšime ich tak napr. od chýb, ktorých výskyt je zapríčinený púhou nepozornosťou (napr. preklepy).

Prvého druhu *racionálnej* chyby sa dopúšťame vtedy, keď vhodne vybraté (zvolené) pravidlo tvorby postupu riešenia nesprávne použijeme. Pritom nesprávne použiť pravidlo znamená alebo nevykonať všetko, čo pravidlo predpisuje, alebo nevykonať predpísané činnosti v stanovenom poradí, prípadne ignorovať podmienky, ktorých plnenie rozhoduje o použití alebo nepoužití niektorých častí postupu. Chyby tohto druhu budeme nazývať *syntaktickými.*

Syntaktickej chyby sa môžeme ľahko dopustiť napr. pri dôkazoch matematickou indukciou (tým, že vynecháme prvý krok indukcie).

Ignorovaním špecifických poznatkov, týkajúcich sa problému, ktorý práve riešime, dopúšťame sa druhého druhu *racionálnej* chyby, a to chyby *sémantickej.* Poznatky viazané na konkrétny problém treba striktne

[1] Vo vytvorenom postupe riešenia sa, pochopiteľne, môže vyskytovať aj viacero chýb. Pokladáme za nerealistické opraviť všetky chyby naraz. Preto sa radšej prikláňame k druhej krajnosti — vo vytvorenom postupe riešenia opraviť vždy iba jedinú chybu (s výnimkou, keď nájdená chyba je bezprostrednou príčinou nasledujúcich chýb). Odstránením chyby dostaneme nový postup riešenia, ktorý môžeme v prípade potreby opäť podrobiť procesu odstraňovania chýb. Takto sme schopní odstrániť všetky chyby pôvodného postupu riešenia.

zohľadniť pri výbere spôsobov tvorby, lebo práve ony stanovujú, čo možno považovať za triviálny problém alebo problém so známym riešením, ako zvoliť podmienky, napr. pri konštrukcii I., aj to, akú taktiku zvoliť pri plánovaní. Poznatky o probléme tu nevystupujú ako pomocné fakty (ako to bude pri treťom druhu *racionálnych* chýb), ale ako prvotné východiská, bez ktorých nie je možné nájsť správny postup riešenia. Povahu sémantických chýb sa pokúsime ilustrovať príkladom prevzatým z [5]:

Čitateľský krúžok, v ktorom bolo 20 ľudí — dospelých aj detí — urobil zbierku na doplnenie knižnice. Vklad dospelého člena bol 30 Kčs, vklad dieťaťa 10 Kčs. Koľko bolo v krúžku dospelých, keď sa vyzbieralo 350 Kčs?

Označme počet dospelých x. Potom počet detí bude $20-x$. Peniaze, zozbierané od detí, predstavujú sumu $10\cdot(20-x)$, od dospelých $30\cdot x$ Kčs. Z toho vyplýva:

$$10\cdot(20-x)+30\cdot x=350, \quad \text{teda} \quad x=7{,}5$$

Tento syntakticky správny postup riešenia nie je sémanticky správny, pretože počet dospelých (aj detí) musí byť nezáporné celé číslo. Základnou chybou tvorby tohto postupu je, že sme operácie, povolené nad reálnymi číslami, používali pre celé čísla.

Posledného druhu racionálnych chýb sa dopúšťame vtedy, keď nevenujeme pri tvorbe postupov riešení dostatočnú pozornosť ich praktickej realizovateľnosti (pri programovaní napr. ignorujeme časovú alebo pamäťovú náročnosť vytvorených programov) alebo rôznym ekonomizačným kritériám. Okrem toho, niektoré pravidlá plánovania nás nútia rozhodnúť sa, ktorým z alternatívnych pravidiel sa budeme riadiť pri ďalšom vytváraní postupu riešenia, t. j. ktorou z vetiev *stromu* na *obr. 2* sa vyberieme. Naše rozhodovanie sa v takýchto prípadoch riadi pravidlami, ktoré nie sme schopní sformulovať podobným spôsobom ako tie pravidlá, ktoré sme formulovali v predchádzajúcej časti. Korene rýchto pravidiel sú v našej subjektívnej skúsenosti, v zručnosti riešiť niektoré typy problémov, v ovládaní niektorých, exaktne neformulovaných *trikov* pri tvorbe postupov riešení a pod. Týmto chybám, zapríčineným v konečnom dôsledku nevhodným výberom spôsobu, ako pokračovať pri plánovaní postupu riešenia problémov, budeme hovoriť *pragmatické* chyby. Na nevyhnut-

nosti použiť neštandardný, a teda pre bežné úlohy pragmaticky chybný, výber pravidla pri tvorbe postupu riešenia je založených veľa hlavolamov.

5.2 Oprava chýb

V etape odstraňovania chýb, keď pristupujeme k ich *oprave*, máme už k dispozícii dve dôležité informácie: vieme, kde sme sa chyby dopustili a poznáme aj to, akého druhu je táto chyba. Úvahy o oprave chýb sa preto budú dotýkať pomerne všeobecne formulovaných pravidiel, ako lokalizovanú chybu zisteného druhu opraviť. Každý druh chyby sa opravuje inak.

Ak opravujeme syntaktickú chybu, je naše počínanie (v porovnaní s opravou sémantickej alebo pragmatickej chyby) pomerne jednoduché. Oprava spočíva v nevyhnutnosti zahrnúť do postupu aj vykonanie kroku, ktorý v pôvodnej verzii nefiguroval, alebo v zmene nesprávneho poradia vykonávaných činností (na správne), prípadne v doplnení podmienky, ktorú sme do pôvodného postupu začlenili.

Intelektuálne náročnejšia je oprava sémantickej chyby. Sémantické chyby vznikajú, ako sme už napísali, ignorovaním poznatkov, viazaných na problém, ktorého postup riešenia tvoríme. Zistiť, ktoré poznatky sme ignorovali, môže byť v niektorých prípadoch dosť namáhavé. Úsilie odstrániť sémantické chyby vedie veľmi často k nevyhnutnosti prehlbovať si vedomosti v problematike, ktorej sa riešený problém dotýka. Spomeňme ako príklad sémantickej chyby nesprávnu substitúciu pri integrovaní. Schopnosť správnej voľby substitúcie je podmienená dostatočne hlbokými znalosťami integrálneho počtu. Bez týchto znalostí sa vzniknutá chyba odstraňuje iba sériou pokusov a omylov, trvajúcou až dovtedy, kým *neuhádneme* vhodnú substitúciu.

Ak opravujeme pragmatickú chybu, musíme sa opäť rozhodnúť, ktorou z alternatívnych možností sa budeme riadiť pri ďalšej tvorbe postupu riešenia problému. Pri tomto rozhodovaní musíme kriticky prehodnotiť naše pôvodné rozhodnutie aj príčiny, ktoré nás k nemu viedli, a venovať pri novom rozhodovaní pozornosť aj tým otázkam realizovateľnosti a efektívnosti, ktoré prvé rozhodovanie (to, ktoré bolo mylné a bolo príčinou chyby) nepodmieňovali.

6. Záver

Cieľom tohto článku bolo poukázať na niektoré relatívne univerzálne metódy vytvárania postupov riešení problémov. Domnievame sa, že opísané metódy tvorby sa môžu použiť pri riešení problémov z najrozličnejších oblastí. Vzhľadom na to, že v niektorých prípadoch nás pravidlá tvorby privedú až k vyjadreniu postupov riešení v jazyku, podobnom programovacím jazykom, pokladáme systém pravidiel plánovania a odstraňovania chýb, sformulovaný v článku, za jednu z alternatívnych možností dorozumenia sa medzi programátormi a ľuďmi iných profesií, ktorí chcú počítač využívať ako pomocníka pri nachádzaní riešení vlastných problémov. Formulácia postupu riešenia daného problému, vytvorená v súlade s pravidlami z časti 4, by mohla byť užitočným východiskom pri vytvorení programu, riešiaceho daný problém aj v tom prípade, keby ju sformuloval odborník napr. z biológie, psychológie alebo národopisu. Treba si uvedomiť, že v súčasnosti jestvuje okolo programovania celý okruh vedných disciplín. Vyžadovať od používateľov výpočtovej techniky, aby mali z týchto disciplín viac než čiastočné poznatky, je nereálne a aj nezmyselné. To, čo sa od nich musí vyžadovať je, aby boli schopní postupy riešenia svojich problémov formulovať v takej artikulovanej forme, ktorá môže poslúžiť ako východisko pri vytvorení programu. Vytvoriť tento program, overiť jeho správnosť, optimalizovať jeho parametre, to je už úloha odborníkov-programátorov.

Posledné riadky netreba chápať tak, akoby znalosť niektorého z programovacích jazykov, základné poznatky o činnosti počítačov, alebo základy systematických metód programovania mali ostať používateľom neznáme. Práve naopak, tieto poznatky by sa mali stať čím skôr súčasťou všeobecného vzdelania podobne ako základy matematiky, biológie, dejepisu, chémie a pod. Naša poznámka sa týka realizácie veľkých programových celkov, ktorá, ak má byť efektívna, vyžaduje spoluprácu s profesionálnymi programátormi.

Hoci sme napísali, že sa zaoberáme iba niektorými univerzálnymi metódami vytvárania postupov riešení problémov, aby sme predišli možnej nesprávnej, prípadne priveľmi všeobecnej interpretácii článku, upozorňujeme čitateľov, že popri nami opísaných metódach existujú aj iné. My sme sa pokúsili spracovať najznámejšie metódy *šablónovitého* mysle-

nia, ktoré používa väčšina ľudí pri riešení väčšiny svojich problémov. Niektorí ľudia však majú vzácny dar myslieť *nešablónovito.* Parafrázujúc A. Szent-Györgyiho môžeme povedať, že hoci vidíme (a vieme) to, čo vidia (a vedia) oni, predsa si na to, na čo oni, nepomyslíme. Dnes je už aj takýto druh myslenia predmetom výskumov a niektoré výsledky môže záujemca nájsť v knihe [3]. Zo spomínanej knihy vyberáme vytrvalému čitateľovi, ktorý dočítal článok až k tomuto miestu, jednu hádanku, rozriešenie ktorej vyžaduje spomínané *nešablónovité* myslenie:

Chudobný dlžník dlhoval úžerníkovi veľkú sumu peňazí, ktorú nemal možnosť vrátiť. Úžerník od neho žiadal vrátenie dlhu pod hrozbou, že ho dá uvrhnúť do žalára, ak peniaze nevráti. Bol však ochotný celú dlžobu odpustiť, keď mu dlžník dá za ženu svoju dcéru. Tá však, ako to už býva, úžerníka nenávidela a o vydaji nechcela ani počuť. Mala však rada svojho otca a nechcela pripustiť, aby sa dostal do väzenia. Úžerník, predstierajúc trochu ľudskosti, rozhodol sa losovať. „Do vrecúška dám dva kamienky — čierny a biely. Ak dcéra vytiahne čierny, stane sa mojou ženou, alebo pôjdeš do temnice" povedal dlžníkovi. „Ak vytiahne biely kamienok, tvoj dlh je odpustený a ona ostane slobodná". Hneď sa aj pobrali na štrkom vysypaný dvor, aby losovanie uskutočnili. Dcéra, ktorá mala ťahať, však zbadala, že podlý úžerník dal do vrecúška dva čierne kamienky. Predsa dosiahla, že losovanie dopadlo v jej prospech. Ako?

Literatúra

1. Cole, M.—Scribner, S.: Culture and Thought. New York, John Wiley and Sons 1974 (rus. prekl.: Moskva, Progress 1977).
2. Dahl, O. J.—Dijkstra, E. V.—Hoare, C. A. R.: Structured Programming, London, Academic Press 1972 (rus. prekl.: Moskva, Mir 1975).
3. De Bono, E.: The Use of Lateral Thinking. London, Penguin Books 1972 (rus. prekl.: Moskva, Progress 1976).
4. Dijkstra, E. W.: A Discipline of Programming. Englewood Cliffs, Prentice Hall 1976 (rus. prekl. Moskva, Mir 1978).
5. Fridman, L. M.: Logicko-psichologičeskij analiz školnych učebnych zadač. Moskva, Pedagogika 1977.
6. Goldstein, I. P.—Miller, M. L.: Structured Planning and Debugging — A Linguistic Theory of Design, Massachusetts Institute of Technology, AI Memo 387 (Logo Memo 34), 1976.

7. Hvorecký, J.—Kelemen, J.: Programovací jazyk Staviteľ. Matematické obzory, zv. 12, 1978. Bratislava, Alfa 1978.
8. Kelemen, J.: Mýtus o myslení a o mysliacich strojoch. Matematické obzory, zv. 14 1979. Bratislava, Alfa 1979.
9. Miller, M. L.—Goldstein, I. P.: Overview of a Linguistic Theory of Design, Massachusetts Institute of Technology, AI Memo 383 (Logo Memo 30), 1976.
10. Polya, G.: How to Solve It. New York, Doubleday Anchor Books 1957 (rus. prekl.: Moskva, Učpedgiz 1959).
11. Polya, G.: Mathematical Discovery, vol. I—II. New York, John Wiley and Sons 1962—1963 (rus. prekl.: Moskva, Nauka 1976 — obidva diely v jednom zväzku).
12. Polya, G.: Mathematics and Plausible Reasoning, vol. I.—II., New Jersey, Princeton University Press 1967—1968 (rus. prekl.: Moskva, Nauka 1975 — obidva diely v jednom zväzku).
13. Sussman, G. J.: A Computational Model of Skill Acquisition. New York, American Elsevier 1975.