Ing. Josef Tvrdík, CSc
Ústav ekologie průmyslové krajiny ČSAV Ostrava

TYPY ÚSUDKU V PROGRAMOVÁNÍ

Bezprostředním podnětem pro napsání tohoto příspěvku bylo několik vět v úvodu zasvěceně a přitom srozumitelně napsaných učebních textů /1/. Tam se píše, že k řešení současných i budoucích problémů programování existuje jeden univerzální prostředek - správné myšlení.

S tímto tvrzením patrně většina programátorů intuitivně souhlasí. Správné myšlení (nejen v programování) je možno považovat za jeden z nejvyšších a nejúčinnějších prostředků člověkem využitelných. Úsilí o správné myšlení v programování jsou věnovány i tyto již tradiční semináře.

Ale už trochu podrobnější úvaha o výše uvedeném tvrzení přináší pochybnosti způsobené především obecností tohoto tvrzení. Vyvstává otázka, jaké cesty k tomuto jistě chvályhodnému cíli - správnému myšlení v programování - zvolit a jak vůbec objektivně rozeznat v různých situacích správné myšlení od nesprávného? Samozřejmě, že na velmi obecnou otázku lze dát velmi obecnou odpověď, že totiž kritériem správnosti myšlení v programování jsou praktické výsledky. Zdá se však, že tato odpověď v řešení aktuálních otázek programování příliš užitku nepřinese. Je potřeba odpovědí podrobnějších, které by poskytovaly nejen filosofické zabezpečení obecného problému, ale i zabezpečení metodické a návody na řešení různých dílčích úloh.

Zde je však dobrá rada drahá. Univerzální prostředek pro posouzení správnosti myšlení v programování neexistuje, alespoň autorů tohoto příspěvku není znám. Zaručeně si ale dovedeme představit, jak by byl užitečný. Proto stojí za to se úvahami o správném myšlení v programování hlouběji zabývat, i když předem nevíme, zda naše úsilí bude mít úspěch. Je totiž otázkou, zda vůbec lze nalézt univerzální prostředek pro posouzení správnosti myšlení v programování. Problém patrně nelze zformulovat tak, aby buď byl vyčerpávajícím způsobem vyřešen nebo byl podán důkaz jeho neřešitelnosti.

Typy úsudku

Je známo, že v našem úvažování užíváme úsudků a metod zkoumání, které lze rozdělit do dvou základních typů - dedukce a indukce. Dedukcí z obecných tvrzení - premis - odvozujeme tvrzení zvláštní - závěr. Schematicky můžeme tento typ úsudku vyjádřit

$$\frac{A_1, A_2, \ldots, A_n}{B} \quad , \quad n \geq 1$$

Dedukce je základním myšlenkovým postupem axiomaticky budovaných teorií v různých oblastech. Těmito postupy se získávají teorie s užitečnými a elegantními vlastnostmi (bezespornost, úplnost, rozhodnutelnost atd.). Pro další naše úvahy je však důležité připomenout, že korespondenci teorie a reality, tzn. obsahovou správnost axiomů a vztah modelu teorie (tj. nějaké formální relační struktury) k reálným objektům a relacím je nutno zkoumat nededuktivním postupem.

Indukce je forma úsudku, kdy z jednotlivých faktů docházíme k obecným závěrům. Induktivní úsudky můžeme dělit na úplnou (matematickou) a neúplnou indukci. Úplná indukce je obecným závěrem o celé třídě na základě zkoumání všech jejich prvků. Je to jediný typ induktivního úsudku, který vede ke zcela spolehlivým závěrům. Ostatní induktivní úsudky (neúplná indukce) vedou k závěrům, které jsou spolehlivé jen částečně. Pokud chceme spolehlivost těchto závěrů nějakým způsobem měřit, vyžaduje to dobře popsat podmínky použitelnosti těchto měr, tzn. definovat příslušné formální struktury a pravidla pro zacházení s nimi - viz např. matematická statistika. Zhruba platí, že čím přesněji chceme vyjádřit míru spolehlivosti (nebo nejistoty) induktivních závěrů, tím důkladněji musí být formalizována oblast použití. Schema induktivní inference (usuzování) lze vyjádřit ve tvaru /6/.

$$\frac{\text{předpoklady, observační tvrzení}}{\text{teoretická tvrzení}}$$

Patrně nikdo nevěří, že existuje obecné univerzální pravidlo racionálního induktivního usuzování. Volba racionálního pravidla induktivní inference závisí na mnoha okolnostech - na úrovni možné a užitečné formalizace zkoumané soustavy, na korespondenci konceptů a reálných objektů, na různých časových a účelových hlediscích. Důležitou roli hraje i subjekt řešitele.

Po tomto dosti zjednodušeném výkladu o dedukci a indukci napadne asi čtenáře otázka: nelze neúplnou indukci, která vede k ne zcela spolehlivým závěrům, z procesu tvorby programů vyloučit? Proč nepoužívat jen deduktivní úsudky a úplnou indukci, abychom dospěli k nepochybně správným závěrům? Konec konců, v matematice jsme byli tímto směrem vedeni od základní školy.

Odpověď na tuto otázku je stručná: nelze. Na programy je sice možno nahlížet jako na formální modely reality, je možno i ledacos o těchto modelech za pomocí deduktivního aparátu odvodit a dokázat. Pro programování je však důležité, jak formální model odpovídá realitě, např. jestli funkce programu odpovídá představě a požadavkům uživatele ap. Toto odpovídání - isomorfismus - reálného systému a jeho formálního modelu nelze dokázat žádným deduktivním postupem /5/.

Kromě toho intuitivní indukce (tj. bez přesně vymezených inferenčních pravidel) má řadu výhodných vlastností. Umožňuje řešení problémů, které jsou neúplně nebo neurčitě zadány (buď dospějeme k nějakému částečnému vyřešení nebo k lepší formulaci zadání). Takové problémy jsou v procesu tvorby programů velmi časté - viz např. fázi prvotního vnějšího návrhu, hledání střední úrovně řešení /8/, návrhu struktury programu /3/ nebo způsobu zápisu ve zdrojovém jazyku /2/. V nezbytnosti intuitivní indukce nejen v těchto fázích řešení asi spočívá původ názorů, že programování je umění. Rozdílnost v používání intuitivní indukce je hlavní příčinou často tak podstatně odlišné výkonnosti různých programátorů i úspěšnosti (především správnosti a spolehlivosti) jejich programů.

Předcházející odstavce můžeme shrnout - zápas o správné myšlení v programování se vede přes rozpory a z nich vyplývající kompromisy ve využívání dvou základních typů úsudků v jednotlivých fázích procesu tvorby software: na jedné straně mezi úsudky poskytujícími správné závěry o více či méně (většinou více) formalizovaných modelech a na druhé straně mezi indukcí (velmi často intuitivní), bez níž se při zkoumání reality neobejdeme. Podívejme se nyní z tohoto pohledu na některé programovací techniky a na způsoby, kterými je uvedený rozpor řešen.

Programovací metody, indukce a dedukce

Rozhodovací tabulky mají základ svého teoretického zabezpečení v systému výrokové logiky. Kvadrant podmínek je reprezentací jednoduchých výrokových formulí - elementárních konjunkcí, buď přímo nebo lze snadno na ně přejít. Rozhodovací tabulky jsou podobně jako výroková logika jednoduché, bezesporné, intuitivně přijatelné a přijímané. Potíže výrokové logiky, např. interpretace některých výroků s implikací ap. /4/ se rozhodovacích tabulek nedotýkají. Zdá se, že i toto jednoduché deduktivní, tj. jistotu poskytující zabezpečení přispělo k úspěšnému a širokému rozšíření rozhodovacích tabulek v procesu tvorby programů. Vlastnosti výrokových formulí umožňují zformulovat pravidla pro různé transformace tabulek (úplné, neúplné RT) i poměrně snadný automatický překlad RT do programovacího jazyka. Tento relativně jednoduchý deduktivní základ RT však kromě výhod přináší i omezení v použití této metody: je vhodná tam, kde provedení nějaké funkce programu je možno adekvátně podmínit pravdivostní hodnotou výrokové formule. Takových situací je v programování mnoho, pokud bychom nepožadovali adekvátnost, tak asi všechny. Avšak právě k posouzení této adekvátnosti potřebujeme jiné prostředky než RT, většinou intuitivní indukci.

Normované programování "poskytuje jistoty" jiným způsobem. Řeší logiku programu pro jistou třídu úloh (zpracování setříděných sekvenčních souborů). V tomto řešení je užito úsudků vedoucích ke spolehlivým závěrům, takže výsledkem je zaručeně bezchybně navržená logická kostra programu. Zda tato správně navržena kostra byla v programu správně realizována a zda jsou na ní správně navěšeny ostatní části daného specielního programu, je nutno ověřit laděním programu, tedy postupem induktivním.

Modulární programování, zejména návrh modulárních programů /1, 3/, je založeno především na induktivních úsudcích. Problém je rozkládán postupně a do takových částí, aby pravidla induktivní inference užívaná v každém z jednotlivých kroků byla triviální, tj. samozřejmě přijatelná. O tom, že takový postup rozkladu úlohy lze nalézt, svědčí např. skutečnost, že modulární návrhy téhož programu vypracované různými autory jsou většinou téměř shodné. Je ale obtížné tento postup zformulovat tak jednoznačně, aby jeho aplikací různí programátoři došli ke stejnému návrhu programu zaručeně vždy. Pro navrhování modulárních programů je takovým pokusem o návod postupu rozkladu s aplikací jednoduchých inferenčních pravidel Myersův souhrnný / strukturovaný návrh /1, 3/. V jiných fázích procesu tvorby software návody podobných vlastností většinou chybí.

Strukturované programování se v užívaných úsudcích velmi podobá modulárnímu. Opět návrh programu postupným zjemňováním vzniká převážně na základě induktivní inference. Kromě toho však v pozadí strukturovaného programování stojí dnes už dosti rozvinutý formální kalkul dynamické logiky /7/. Dynamická logika poskytuje prostředky k dokazování správnosti programů. "Ruční" dokazování správnosti programů je sice tak pracné, že nemá vyhlídky na širší uplatnění, avšak automatizované důkazy správnosti programů už nejsou vzdálenou utopií (viz zprávy o některých kompilátorech jazyka Pascal). Automatizované dokazování správnosti programů s výhledem, že velmi nejisté induktivně získané závěry z empirických údajů o ladění programů budou nahrazeny spolehlivými závěry získanými pomocí deduktivního aparátu dynamické logiky, je asi největším lákadlem a nejsilnějším argumentem pro strukturované programování.

V ostatních rozšířených programovacích metodách můžeme vystopovat podobně užité základní typy úsudku. Většinou deduktivní aparát je někde v hlubokém pozadí mnohdy bez zjevné souvislosti s danou metodou a vlastní metoda je založena především na induktivních úsudcích s triviálními inferenčními pravidly (např. parametrické programy, generátory programů, neprocedurální uživatelské jazyky ap.).

Závěr

Příspěvek je věnován typům úsudku v procesu tvorby programů. Zvolené téma předem téměř vylučuje v takto krátkém příspěvku dojít k rozumnému a jednoznačnému závěru. Kromě toho rozsah příspěvku není nejpodstatnějším omezením, se kterým se autor musí potýkat. Každá práce však nějaký závěr má mít. Proto nabízím tento:

Řešení problémů tvorby programů není možné bez neúplných induktivních úsudků.
Neexistuje universální racionální pravidlo induktivní inference pro velmi rozmanité problémy programování.
Není možno nalézt universální programovací metodu.

O odpovědi na otázku, jakým typem úsudku byl získán tento závěr, nechť uvažuje laskavý čtenář sám.

Jelikož s velikou pravděpodobností se autor nebude moci zúčastnit semináře "Programování 81", považuje za svou povinnost odpovědět předem na některé otázky a připomínky, které tento příspěvek může vyvolat.

Otázka: V příspěvku jsou indukce a dedukce stavěny do protikladu. Opravdu takový protiklad existuje?
Odpověď: Asi neexistuje, oba základní typy úsudku tvoří spíše jednotný navzájem se doplňující celek.

Otázka: Dost se napřemýšlíme při vlastním programování. Proč si ještě přidělávat starosti a úvahami o tom, jak máme a můžeme myslet, zvláště, když universální programovací metodu stejně nelze nalézt?

Odpověď: Můžeme však nalézt užitečnou metodu speciální. Navíc v používání každé metody (nejen v programování) je důležité si uvědomovat její možnosti a meze a ty patrně jinak než podobnými úvahami nelze nalézt.

Otázka: Není v tomto příspěvku otázka správného myšlení v programování chápána příliš úzce tím, že je omezena na typy úsudku a ty navíc jsou probírány velmi zjednodušeně

Odpověď: Je, avšak pokud příspěvek vyprovokuje k přemýšlení o souvisejících otázkách nebo k publikaci hlubší a zevrubnější, nebyl bezúčelný.

Literatura:

1. ČIMBURA V., JIŘÍČEK P., METZL K., PEŠKA J., Vybrané kapitoly z moderních programovacích metod, učební texty rezort. vzděláv. střediska FMPE, Havířov 1980

2. TVRDÍK J., Poznámky ke struktuře fortranských programů, in: sborník semináře "Metody programování počítačů III. generace", DT ČVTS Ostrava, 1977

3. ČIMBURA V., TVRDÍK J., Problémy návrhu modulárních programů, in: sborník semináře "Programování 80", DT ČVTS Ostrava, 1980

4. MLEZIVA M., Neklasické logiky, Svoboda, Praha, 1970

5. HRABEC E., Systémové hnutí ve vědách o složitém, Sdělení MMBÚ ČSAV A - 22 (1979)

6. HÁJEK P., HAVRÁNEK T., Mechanizing Hypothesis Formation, Springer, Berlín, Heidelberg, New York, 1978

7. HÁJEK P., KALÁŠEK P., KŮRKA P., Systémy dynamické logiky a programování,
in: Sborník SOFSEM "79", VVS Bratislava 1979

8. RAJLICH V., Úvod do teorie počítačů, SNTL, Praha 1979