Základy teórie programovania

skúška, termín 1, 27. mája 2003

Kurzívou sú vyznačené moje poznámky k príkladom, ktoré však rozhodne nemusia viesť k zaručene správnym riešeniam.

Ondrej Jombík, 30. mája 2003

1. Sformulujte a dokážte tvrdenie o (ne)rozhodnuteľnosti vlastností divergencie a zastavenia voľných schém a o príslušnosti štandardnej schémy k triede voľných schém. (20b)

   divergencia - rozhodnuteľný (existuje cesta do **end***?); zastavenie - rozhodnuteľný (existuje v schéme cyklus?); príslušnosť - nerozhodnuteľný*

2. Uvažujme triedu schém $\mathcal{L}$ s rozhodnuteľným problémom dosiahnuteľnosti príkazu (napr. liberálne schémy).

   Dokážte alebo vyvráťte nasledujúce tvrdenie: Problém či je ľubovoľná schéma z triedy $\mathcal{L}$ priechodná je rozhodnuteľný. (10b)

   Len na prvý pohľad je tento príklad triviálny. Úvahy:

   Majme len štandardné schémy. Tam vieme priechodnosť čiastočne rozhodovať. Hrán je konečný počet a tak si ich všetky vložíme do množiny. Začneme prehľadávať do šírky (tak ako zastavenie na $\mathcal{S}$) a vždy keď prejdeme hranou vyberieme ju z množiny. Až nám ostane prázdna množina vyhlásime, že schéma je priechodná. Inak pokračujeme do nekonečna.

   Teraz skúmajme, či nám dosiahnuteľnosť príkazu nejako pomôže pri rozhodovaní priechodnosti. Ja hovorím, že nie. Ak by do každého príkazu viedla iba jedna hrana, vedeli by sme, že ak je príkaz dosiahnuteľný, určite je táto hrana prichodná. Do príkazu však môže viesť dve alebo viac hrán a my nevieme určiť, ktorou hranou sme do príkazu prišli (teda ktorá hrana z tých dvoch je priechodná).

   Po písomke sa však ozval jeden expert, ktorý povedal, že prípady, kde do príkazu vedie dve alebo viac hrán vie ošetriť takto: ak vedie do príkazu n hrán, ponechá len jednu a zvyšných $n-1$ odstráni. Potom sa spýta na dosiahnuteľnosť príkazu a teda aj priechodnosť ponechanej hrany. Takto to opakuje pre všetky hrany vedúce do príkazu. Ak je príkaz vždy dosiahnuteľný, tak aj všetky doňho vedúce hrany sú priechodné.

   Ja ale hovorím, že môže existovať schéma, do ktorej príkazu vedú dve hrany, pričom obe sú priechodné. Tou druhou sa však prejde len vtedy, ak sa predtým prešlo prvou:

$$\begin{array}{rl} \textbf{begin} & [y] := [x] \\ 1: & [y] := [y] \\ 2: & \textbf{goto } 1 \end{array}$$

V tomto prípade sa z riadku 2 na 1 prejde až po prejdení z **begin** *na 1. Lenže ak my hranu z* **begin** *do 1 odstránime, algoritmus prehlási, že hrana z 2 do 1 je nepriechodná!*

Ako som vravel, zaujímavý príklad. Môj názor je, že problém je čiastočne rozhodnuteľný, pričom dosiahnuteľnosť príkazu na to nemá žiadny vplyv. Veľmi rád sa ale nechám poučiť ak je to inak.

3. Uvažujme triedu štruktúrovaných schém W^b, obohatenú o booleovské premenné a dobre otypované priradenie do booleovských premenných. Ktorú z uvedených konštrukcií je (nie je) možné preložiť do ekvivalentnej schémy z triedy W^b? Zdôvodnite prečo. (15B)

$$\begin{array}{rl} W_1: & \textbf{while } b_1 \wedge b_2 \textbf{ do } S \textbf{ od} \\ W_2: & \textbf{while } b_1 \vee b_2 \textbf{ do } S \textbf{ od} \\ W_3: & \textbf{while } \neg b \textbf{ do } S \textbf{ od} \end{array}$$

Základom je nepomýliť si tento príklad s prekladaním programových konštrukcií do $\mathcal{W}$. :-)

Preložiť idú všetky. K dispozícii sú totiž aj konštanty $TRUE$ a $FALSE$, ktoré je možné priradiť do booleovskej premennej. Ak by však aj tieto konštanty neboli, W_1 a W_2 by stále preložiť išli len za pomoci booleovských priradení. Navyše W_2 ide preložiť aj do $\mathcal{W}$, tzn. nie sú booleovské premenné a priradenia potrebné.

4. Definujte význam induktívnej formuly $\{p\}P\{q\}$ a uveďte axiómy a inferenčné pravidlá Hoareovského systému pre imperatívny programovací jazyk obsahujúci

   – priraďovací príkaz,
   – kompozíciu príkazov,
   – podmienkový príkaz (**if** ... **then** ... **else** ... **fi**),
   – príkaz cyklu (**while** ... **do** ... **od**).

   Definujte význam induktívnej formuly $[p]P[q]$ tak, aby vyjadrovala totálnu správnosť programu. (15B)

   Čo dodať? Učebné texty má predsa každý...

5. Určite a zdôvodnite, či sú nasledujúce dve inferenčné pravidlá <u>zdravé</u>. (10B)

$$\frac{\{p \wedge b\}\ S\ \{(p \wedge \neg b) \Rightarrow q\}}{\{p\}\ \textbf{while}\ b\ \textbf{do}\ S\ \textbf{od}\ \{q\}}$$

$$\frac{\{p \wedge b\}\ S\ \{p \wedge (\neg b \Rightarrow q\}}{\{p\}\ \textbf{while}\ b\ \textbf{do}\ S\ \textbf{od}\ \{q\}}$$

Väčšine včetne mňa vyšlo, že sú obe nezdravé. Boli však aj takí, čo vraveli, že druhé je zdravé. Nepočul som však nič o tom, že by niekto tvrdil, že prvé je zdravé. Počkáme si na výsledky...

6. Na základe Floydovej metódy sformulujte postačujúce podmienky <u>čiastočnej správnosti</u> schémy S vzhľadom na vstupnú podmineku $p(x)$ a výstupnú podmienku $q(x,z)$ a invarianty $I_i(x,\overline{y})$ v príslušných deliacich bodoch.

$$\begin{array}{rrl}
S: & \textbf{begin} & [y_1, y_2] := [a||x??, a] \\
 & 1: & [y_1] := [g(y_1, y_2)] \\
 & 2: & \textbf{if } p_1(y_1) \textbf{ then goto } 5 \\
 & 3: & [y_1, y_2] := [f_1(y_1, f_2(y_2)] \\
 & 4: & \textbf{goto } 1 \\
 & 5: & [y_2] := [g(y_2, y_1)] \\
 & 6: & \textbf{if } p_2(y_2) \textbf{ then goto end} \\
 & 7: & [y_1, y_2] := [g_1(y_1), g_2(y_2)] \\
 & 8: & \textbf{goto } 1 \\
 & \textbf{end} & [z] := [g_1(y_1)]
\end{array}$$

(a) Nájsť deliace body (doporučujem na riadkoch 2 a 6).

(b) Sformulovať invarianty I_2 a I_6 (ja som mal $I_2 : y_1 = g(y_1, y_2)$ a $I_6 : y_1 = g(y_1, y_2) \wedge y_2 = g(y_2, y_1)$).

(c) Odvodiť verifikačné podmienky (pre všetky cesty treba odvodiť R_{AB} a r_{AB} a potom spolu s invariantami dosadiť do všeobecnej rovnice).

(d) Dokázať verifikačné podmienky (v tomto prípade bude stačiť, keď sa prehlási, že dokázaním týchto podmienok dokážeme aj čiastočnú správnosť schémy).

7. Vyjadrite operačnú vstupno-výstupnú sémantiku $\mathcal{O}$ a denotačnú sémantiku $\mathcal{M}$ jednoduchého imperatívneho jazyka so "štandardnými" príkazmi priradenia, kompozície, vetvenia a s príkazom cyklu

$$\textbf{loop}\ S_1;\ \textbf{when}\ b\ \textbf{exit}; S_2\ \textbf{pool}$$

a zdôvodnite, že platí $\mathcal{M}[\![S]\!]\sigma = \mathcal{O}[\![S]\!]\sigma$. (20B)

V skriptách je tvrdenie $\mathcal{M}[\![S]\!]\sigma = \mathcal{O}[\![S]\!]\sigma$ veľmi podrobne dokázané. Až nato, že ako cyklus sa používa klasický **while** *a nie uvedený jeden a pol cyklus. Inak ale ide v podstate o to isté. Lenže na ten jeden a pol cyklus treba brať pri dokazovaní veľký zreteľ! Dá sa rozložiť pomocou kompozície na dve časti: S_1 a konštrukciu* **while** $\neg b$ **do** $S_2; S_1;$ **od**. *A to už je viac-menej klasický* **while**, *čo sa môže hodiť.*

8. Uvažujme hypotetický iteratívny príkaz **loop** (b, S_1, S_2) definovaný sémantickou rovnicou

$$\mathcal{M}[\![\mathbf{loop}\ (b, S_1, S_2)]\!] \;=\; \bigsqcup_0^\infty \{\phi_i\}$$

kde

$$\begin{aligned}\phi_0 &= \lambda\sigma \cdot \bot \\ \phi_{i+1} &= \lambda\sigma \cdot \mathbf{if}\ \mathcal{W}[\![b]\!]\sigma\ \mathbf{then}\ \phi_i(\mathcal{M}[\![S_1]\!]\sigma) \\ &\qquad\qquad\qquad\quad \mathbf{else}\ \mathcal{M}[\![S_2]\!]\sigma\end{aligned}$$

Na základe základných príkazov (priradenie, kompozícia, vetvenie a cyklus) definujte programový segment S taký, že platí

$$\mathcal{M}[\![\mathbf{loop}\ (b, S_1, S_2)]\!] \;=\; \mathcal{M}[\![S]\!]$$

Tvrdenie dokážte. (10B)

Vyriešené v zbierke, hoci z dôkazu nemám veľmi dobrý pocit.

9. Sformulujte a dokážte vetu o najmenšom pevnom bode funkcionálu (s potrebnými predpokladmi na funkcionál a príslušné obory definície a hodnôt).

Overte a zdôovodnite, či je funkcia f najmenším pevným bodom funkcionálu, zodpovedajúceho rekurzívnej definícii ϕ nad oborom prirodzených čísiel. (20B)

$$\begin{aligned}&f(x,y):\ \mathbf{if}\ x > 0\ \mathbf{then}\ y\ \mathbf{else}\ \bot \\ &\phi(x,y)\ \Leftarrow\ \mathbf{if}\ x = 0\ \mathbf{then}\ y\ \mathbf{else}\ \mathbf{if}\ y = 0\ \mathbf{then}\ \phi(x-1,1)\ \mathbf{else}\ \phi(x-1,\phi(x,y-1))\end{aligned}$$

Ehm... Vraj je nato nejaký štandardný postup.

10. Uvažujme nasledujúce výpočtové pravidlo SPOS (sekvenčné POS). Pravidlo SPOS vyberie paralelne najvonkajšejšie výskyty (ako POS), ale "prideľuje" ich na substitúciu postupne zľava-doprava.

Určite, či je pravidlo SPOS bezpečné (v jazyku L_1), resp. či je korektné. Svoje tvrdenie zdôvodnite. (15B)

Čo hovorí? Uráža nás? :-)