



FAKULTA MATEMATIKY, FYZIKY
A INFORMATIKY
UNIVERZITY KOMENSKÉHO
KATEDRA INFORMATIKY



Základy teórie programovania

Zbierka riešených príkladov

ONDREJ JOMBÍK

release 0.5 build 2003-03-28

Vďaka zmene systému zo známkovacieho na kreditové, bolo možné zapísať si ZÁKLADY TEÓRIE PROGRAMOVANIA v troch po sebe idúcich školských rokoch. Treba mať na pamäti, že autori podieľajúci sa na tvorbe tohto dokumentu, tento predmet aj trikrát zapísaný mali. Z toho vyplýva, že určite nepatria k absolútnym odborníkom na danú problematiku. Tento dokument si vytvorili len pre svoju osobnú potrebu ako užitočnú pomôcku pri štúdiu.

Dokument je len doplnkovým materiálom. V žiadnom prípade nemá slúžiť ako náhrada za dobré skriptá alebo prednášku zo ZÁKLADOV TEÓRIE PROGRAMOVANIA. Autori nenesú žiadnu zodpovednosť za prípadné chyby, preklepy alebo nepresnosti, ale privítajú ich opravy či vylepšenia. Rozšírenie dokumentu o ďalšie príklady a poznámky je taktiež vítané.

Osobitné poďakovanie patrí LUBOMÍROVI HOSTOVI za vytvorenie skvelého pracovného a zostavovacieho rámca pre prácu so systémami L^AT_EX a pdfT_EX.

Obsah

1	Programové schémy	2
1.1	Štandardné programové schémy	2
1.2	Nerozhodnuteľnosť vlastností schém	5
1.3	Porovnávanie tried programových schém	7
2	Správnosť programov	12
3	Sémantika programov	13
	Literatúra	14

Kapitola 1

Programové schémy

1.1 Štandardné programové schémy

Príklad 1 Máme program P_1 . Zistite, čo program počíta a napíšte jeho schému.

```
 $P_1$  : begin    $[y_1, y_2] := [1, 1]$   
          1 : if  $y_2 \geq x$  then goto end  
          2 :  $[y_1, y_2] := [y_1 + 1, (y_1 + 1)^2]$   
          3 : goto 1  
          end    $[z] := [y_1]$ 
```

Riešenie 1 Po krátkej analýze je zrejmé, že program počíta $\lceil \sqrt{x} \rceil$ (hornú celú časť odmocniny x).

Schéma S , abstrakcia programu vzhľadom na riadiace štruktúry, vyzerá takto:

```
 $S$  : begin    $[y_1, y_2] := [a_1, a_2]$   
          1 : if  $p(y_2, x)$  then goto end  
          2 :  $[y_1, y_2] := [f_1(y_1), f_2(y_1)]$   
          3 : goto 1  
          end    $[z] := [y_1]$ 
```

Príklad 2 Napíšte interpretáciu I_1 tak, aby sme pomocou nej a predchádzajúcej schémy S dostali pôvodný program P_1 , tj. aby platilo $P_1 = (S, I_1)$.

Riešenie 2 Interpretácia $I_1 = (D_1, i_1)$:

$$\begin{aligned} I_1 : \quad i_1(p(y, x)) &= y \geq x \\ i_1(f_1(y)) &= y + 1 \\ i_1(f_2(y)) &= (y + 1)^2 \\ i_1(a_1) &= 1 \\ i_1(a_2) &= 1 \\ D_1 &= \mathbb{N} \end{aligned}$$

Príklad 3 Nájdite interpretáciu I_2 , ktorá spolu s predchádzajúcou schémou S vytvorí program, ktorý bude počítat $\lceil \log_2 x \rceil$.

Riešenie 3 Interpretácia $I_2 = (D_2, i_2)$:

$$\begin{aligned} I_2 : \quad i_2(p(y, x)) &= y \geq x \\ i_2(f_1(y)) &= y + 1 \\ i_2(f_2(y)) &= 2^{y+1} \\ i_2(a_1) &= 1 \\ i_2(a_2) &= 0 \\ D_2 &= \mathbb{N} \end{aligned}$$

Pomocou príkladu sme si ukázali, že nad jednou schémou S môžu byť postavené viaceré programy $(S, I_1), (S, I_2) \dots (S, I_n)$ riešiace odlišné úlohy.

Príklad 4 Napíšte históriu výpočtu pre program $P_2 = (S, I_2)$ s hodnotou vstupnej premennej $x = 7$. Formálne sa ohodnotenie vstupných premenných zapisuje ako $v[x \leftarrow 7]$.

Riešenie 4 Je nutné si uvedomiť, že históriu výpočtu môžeme zapísať vzhľadom na stavy výpočtu, ale taktiež vzhľadom na konfigurácie. V našom riešení použijeme druhú možnosť, tj. históriu výpočtu vzhľadom na konfigurácie.

$$\begin{aligned} &[1, 1]_{begin} \\ &[1, 1]_1 \\ &[2, 4]_2 \\ &[2, 4]_3 \\ &[2, 4]_1 \\ &[3, 9]_2 \\ &[3, 9]_3 \\ &[3, 9]_1 \\ &[3]_{end} \end{aligned}$$

Príklad 5 Zostrojte Herbrandovské univerzum pre predchádzajúcu schému S .

Riešenie 5 Herbrandovo univerzum je množina reťazcov symbolov zostrojených zo vstupných premenných a funkčných symbolov.

Riešenie začneme tým, že zoberieme všetky vstupné premenné schémy (v našom prípade vstupnú premennú x) a všetky použité konštanty (v našom prípade a_1 a a_2).

$$"x", "a_1", "a_2"$$

Ďalej zoberieme funkcie f_1 a f_2 a aplikujeme na všetky dostupné termy, ktoré doteraz reprezentujú konštanty a_1 , a_2 a vstupná premenná x .

$$\begin{aligned} &"f_1(x)", "f_1(a_1)", "f_1(a_2)" \\ &"f_2(x)", "f_2(a_1)", "f_2(a_2)" \end{aligned}$$

Týmto krokom sa nám teraz rozšírila množina termov, takže uvedeným spôsobom aplikovania funkcií f_1 a f_2 na termy pokračujeme ďalej.

$$\begin{aligned} &"f_1(f_1(x))", "f_1(f_1(a_1))", "f_1(f_1(a_2))" \\ &"f_1(f_2(x))", "f_1(f_2(a_1))", \dots \\ &"f_2(f_1(x))", "f_2(f_1(a_1))", \dots \\ &"f_2(f_2(x))", \dots \end{aligned}$$

Takto je možné pokračovať do nekonečna. Uvedeným postupom sa teda dá zostaviť množina reťazcov Herbrandovského univerza vzhľadom na príslušnú schému. V našom prípade je táto množina spojená so schémou S .

Keď bližšie preskúmame schému S , zistíme, že napríklad term $"f_1(f_2(a_2))"$ nemôže nikdy počas behu výpočtu vzniknúť. Napriek tomu sa však v množine Herbrandovského univerza nachádza.

Príklad 6 Schéma S sa zastaví práve vtedy, keď sa zastaví výpočet pre každú Herbrandovú interpretáciu schémy S^1 .

¹Ide o zadanie domácej úlohy číslo 1 v roku 2003. Riešenie nie je overené, takže môže byť a pravdepodobne aj je nepresné alebo nesprávne.

Riešenie 6

\Rightarrow : Z definície vieme, že schéma S sa zastaví, ak pre každú interpretáciu I sa zastaví program (S, I) . Program $P = (S, I)$ sa zastaví, ak pre každé ohodnotenie v vstupných premenných \bar{x} je hodnota $val(S, I, v)$ definovaná. Ku každej interpretácii I s ohodnotením v existuje zladená Herbrandová interpretácia. Keďže sme predpokladali, že schéma S sa zastaví, tj. všetky výpočty na nej sa zastavia, zastavia sa aj príslušné zladené Herbrandové interpretácie.

\Leftarrow : Ku každej Herbrandovej interpretácii schémy S existuje interpretácia I a ohodnotenie vstupných premenných v , ktoré sú s ňou zladené. Keďže sme predpokladali, že sa zastavania všetky Herbrandové interpretácie, zastavia sa aj všetky ich zladené výpočty. Ak sa zastavia všetky výpočty na schéme S , zastavia sa aj všetky programy. Ak sa zastavia všetky programy (S, I) , potom sa aj schéma S zastaví.

1.2 Nerozhodnuteľnosť vlastností schém

Príklad 7 Máme danú schému S .

```

S : begin  [y1, y2] := [a, a]
          1 : if p(y1) then goto end
          2 : [y1] := [f(y1)]
          3 : if p(y1) then goto end
          4 : [y1, y2] := [f(y1), f(y1)]
          5 : if p(y1) then goto 7
          6 : goto end
          7 : if p(y2) then goto 4
          8 : goto 2
          end  [z] := [a]

```

Nájdite interpretáciu I_1 takú, že program $P_1 = (S, I_1)$ diverguje.

Riešenie 7 Aby program divergoval, potrebujeme vytvoriť večný cyklus, čiže zabezpečiť beh programu bez dosiahnutia príkazu na návěstí **end**. Smer behu programu ovplyvňujú predikáty. Pre náš zámer bude vhodné, ak predikát $p(x)$ bude dávať napríklad takéto výsledky.

$$\begin{aligned}
 p(a) &= false \\
 p(f(a)) &= false \\
 p(f(f(a))) &= true
 \end{aligned}$$

Vždy po vykonaní príkazu na riadku 4 obsahujú premenné y_1 a y_2 rovnaké hodnoty. Preto ak výsledok testu na riadku 5 bude *true*, potom bude *true* aj výsledok testu na riadku 7. Pre večný cyklus teda stačí, aby ešte platila nasledujúca podmienka.

$$\forall n \geq 2 : p(f^n(a)) = \text{true}$$

Hľadaná interpretácia $I_1 = (D_1, i_1)$ môže potom vyzeráť takto:

$$\begin{aligned} I_1 : \quad i_1(p(x)) &= x \geq 2 \\ i_1(f(x)) &= x + 1 \\ i_1(a) &= 0 \\ D_1 &= \mathbb{N} \end{aligned}$$

Príklad 8 Dokážte alebo vyvráťte tvrdenie, že pre schému S z predchádzajúceho príkladu, pre každú konečnú doménu D_2 a pre každý interpretačný morfizmus i platí, že program $P_2 = (S, (D_2, i_2))$ sa zastaví.

Riešenie 8 Tvrdenie neplatí. Ak na konečnej doméne $D_2 = \{0, 1, 2\}$ upravíme funkciu f tak, že bude vstupný parameter inkrementovať najnajvyšš po hodnotu 2, dostávame dokonca divergentný program P_2 .

$$\begin{aligned} i_2(p(x)) &= x \geq 2 \\ i_2(f(x)) &= \max(x + 1, 2) \\ i_2(a) &= 0 \end{aligned}$$

Príklad 9 Rozhodnite, či je problém dosiahnuteľnosti príkazu v štandardnej schéme rozhodnuteľný. Svoje tvrdenie dokážte².

Riešenie 9 Problém nie je ani čiastočne rozhodnuteľný.

Dôkaz vykonáme sporom. Ak by bol problém rozhodnuteľný alebo čiastočne rozhodnuteľný, vedeli by sme rozhodnúť alebo čiastočne rozhodnúť dosiahnuteľnosť príkazu **end**. Problém dosiahnuteľnosti príkazu **end** je ale totožný s problémom divergencie. Ak je príkaz **end** dosiahnuteľný, schéma nie je divergentná. Čiže pomocou dosiahnuteľnosti príkazu **end** by sme vedeli rozhodovať problém divergencie. Problém divergencie však nie je ani

²Ide o zadanie domácej úlohy číslo 2 v roku 2003. Riešenie nie je overené, takže môže byť a pravdepodobne aj je nepresné alebo nesprávne.

čiasťočne rozhodnuteľný a tak aj problém dosiahnuteľnosť koncového príkazu **end** nie je ani čiasťočne rozhodnuteľný.

Keďže dosiahnuteľnosť koncového príkazu nie je ani čiasťočne rozhodnuteľný problém, potom tiež dosiahnuteľnosť ľubovoľného príkazu v štandardnej schéme nie je ani čiasťočne rozhodnuteľný problém.

1.3 Porovnávanie tried programových schém

Príklad 10 Rozhodnite, či je schéma S voľná.

```

 $S$  : begin    $[y_1, y_2] := [a, a]$ 
           1 : if  $p(y_1)$  then goto 4
           2 :  $[y_1] := [f(y_1)]$ 
           3 : goto 1
           4 : if  $p(y_2)$  then goto end
           5 :  $[y_1, y_2] := [g(y_1), f(y_2)]$ 
           6 : goto 4
       end    $[z] := [y_1]$ 

```

Riešenie 10 Z definície vieme, že schéma S je voľná, keď pre každú cestu vedúcu zo začiatočného príkazu existuje interpretácia I a ohodnotenie vstupných premenných v také, že výpočet (S, I, v) sleduje túto cestu.

Schéma S reprezentuje dva cykly. Na začiatku sa premenné y_1 a y_2 inicializujú na rovnakú hodnotu. V prvom cykle sa iteruje podľa y_1 , v druhom cykle sa iteruje podľa y_2 . V oboch prípadoch testuje ukončenie cyklu predikát p aplikovaný na iteračnú premennú. Takže oba cykly budú mať vždy rovnaký počet opakovaní.

To je ale v rozpore s voľnosťou schémy, pretože nie sú možné výpočty, kde počet opakovaní prvého cyklu nie je rovnaký ako pri druhom cykle. Takže sa nedá spraviť napríklad 3-násobné opakovanie prvého cyklu nasledované 2-násobným opakovaním cyklu druhého. Schéma S teda nie je voľná.

Príklad 11 Máme danú Janovovu schému S .

```

S : begin   [y] := [x]
           1 : [y] := [f(y)]
           2 : if p(y) then goto 6
           3 : [y] := [f(y)]
           4 : if p(y) then goto 6
           5 : goto 2
           6 : if q(y) then goto 8
           7 : goto 4
           8 : [y] := [f(y)]
        end   [z] := [y]

```

Nájdite ku schéme S ekvivalentnú voľnú Janovovu schému S_v . Schému napíšte a stručne zdôvodnite, prečo je schéma S_v voľná a ekvivalentná so schémou S .

Riešenie 11 Riešením je schéma S_v :

```

S_v : begin  [y] := [x]
            1 : [y] := [f(y)]
            2 : if p(y) then goto 4
            3 : goto 1
            4 : if q(y) then goto 6
            5 : goto 5
            6 : [y] := [f(y)]
        end   [z] := [y]

```

Voľnosť: Ak predikát $p(y)$ platí, tak sa už ďalej netestuje. Ak neplatí, tak pred ďalším testom toho istého predikátu sa zmení premenná y . Predikát $q(y)$ sa testuje len raz. Pre každú cestu existuje interpretácia I_v a valuácia v_v taká, že výpočet (S_v, I_v, v_v) sleduje túto cestu, takže schéma je voľná.

Ekvivalencia: Oproti pôvodnej schéme sme zmenili príkaz 7 : **goto** 4 na nový príkaz 5 : **goto** 5. V pôvodnej schéme bol tento príkaz dosiahnuteľný iba ak na riadku 4 platil predikát $p(y)$ a na riadku 6 neplatil predikát $q(y)$, čoho dôsledkom bol opäť skok na riadok 4 a rovnaké testy s rovnakými hodnotami, čiže večný cyklus. Cyklusom 5 : **goto** 5 sme teda dosiahli ekvivalentnú schému.

Taktiež sme vynechali podmienku na riadku 4, pretože môže byť nahradená ekvivalentnou podmienkou na riadku 2 pôvodnej schémy. Nakoniec sme zredukovali príkazy na riadkoch 1 a 3 pôvodnej schémy, pretože sa po malých úpravách novej schémy dajú nahradiť jedným príkazom.

Príklad 12 Daná je štandardná schéma S .

```

 $S$  :  begin    $[y] := [x]$ 
          1 : if  $p(y)$  then goto end
          2 : if  $q(y)$  then goto 6
          3 :  $[y] := [f_1(y)]$ 
          4 : if  $p(y)$  then goto 2
          5 : goto end
          6 : if  $p(y)$  then goto 10
          7 :  $[y] := [f_2(y)]$ 
          8 : if  $q(y)$  then goto end
          9 : goto 1
         10 :  $[y] := [f_3(y)]$ 
         11 : goto 8
        end    $[z] := [y]$ 

```

Nájdite ku schéme S ekvivalentnú voľnú schému S_v ³.

Riešenie 12 Riešením je schéma S_v .

```

 $S_v$  :  begin    $[y] := [x]$ 
          1 : if  $p(y)$  then goto end
          2 : if  $q(y)$  then goto 6
          3 :  $[y] := [f_1(y)]$ 
          4 : if  $p(y)$  then goto 10
          5 : goto end
          6 :  $[y] := [f_2(y)]$ 
          7 : if  $q(y)$  then goto end
          8 : if  $p(y)$  then goto end
          9 : goto 3
         10 : if  $q(y)$  then goto 12
         11 : goto 3
         12 :  $[y] := [f_3(y)]$ 
         13 : goto 7
        end    $[z] := [y]$ 

```

Príklad 13 Máme danú štandardnú schému S .

³Ide o zadanie domácej úlohy číslo 3 v roku 2003. Riešenie nie je overené, takže môže byť a pravdepodobne aj je nepresné alebo nesprávne.

```

S : begin    $[y_1, y_2] := [x, a]$ 
          1 : if  $p(y_1)$  then goto end
          2 :  $[y_1, y_2] := [f(y_1), g(y_1, y_2)]$ 
          3 : goto 1
          end    $[z] := [y_2]$ 

```

Nájdite ku schéme S ekvivalentnú rekurzívnu schému R .

Riešenie 13 Ekvivalentnú rekurzívnu schému R zostrojíme pomocou štandardného postupu. Návestia štandardnej schémy prepisujeme pomocou funkčných premenných ϕ_i (simulácia toku riadenia) tak, aby v ich rekurzívnych definíciach vektory vstupných argumentov \bar{y} zodpovedali vektorom pracovných premenných (simulácia zmenu stavu výpočtu).

$$\begin{aligned}
\phi_b(y_1, y_2) &= z = \phi_1(x, a) \\
\phi_1(y_1, y_2) &= \text{if } p(y_1) \text{ then } \phi_e(y_1, y_2) \text{ else } \phi_2(y_1, y_2) \\
\phi_2(y_1, y_2) &= \phi_3(f(y_1), g(y_1, y_2)) \\
\phi_3(y_1, y_2) &= \phi_1(y_1, y_2) \\
\phi_e(y_1, y_2) &= y_2
\end{aligned}$$

Jednoduchým dosadením do funkčných premenných ϕ_i dosiahneme zjednodušenie systému rekurzívnych definícií a výslednú schému R .

```

R : begin    $[y_1, y_2] := [x, a]$ 
           $\phi_1(y_1, y_2) \Leftarrow \text{if } p(y_1) \text{ then } y_2 \text{ else } \phi_1(f(y_1), g(y_1, y_2))$ 
          end    $[z] := [\phi_1(x, a)]$ 

```

Príklad 14 Máme danú rekurzívnu schému R .

```

R : begin    $[\dots] := [\dots]$ 
           $\phi(y) \Leftarrow \text{if } p(y) \text{ then } f(y) \text{ else } h(\phi(g(y)))$ 
          end    $[z] := [\phi(a)]$ 

```

Zistite, či existuje k tejto schéme štandardná schéma S . V prípade, že existuje, nájdite ju.

Riešenie 14 Výstupná premenná z schémy R je tvaru $h^n f g^n(a)$, kde hodnota n vyjadruje hĺbku rekurzívneho vnorenia. V triede štandardných schém

\mathcal{S} existuje schéma S ekvivalentná s R generujúca výstupné premenné uvedeného tvaru.

```

 $S$  : begin    $[y_1, y_2] := [a, a]$ 
           1 : if  $p_1(y_1)$  then goto 4
           2 :  $[y_1] := [g(y_1)]$ 
           3 : goto 1
           4 :  $[y_1] := [f(y_1)]$ 
           5 : if  $p(y_2)$  then goto end
           6 :  $[y_1, y_2] := [h(y_1), g(y_2)]$ 
           7 : goto 5
       end    $[z] := [y_1]$ 

```

Obsah pracovných premenných $[y_1, y_2]$ v príkaze **begin** bol $[a, a]$, pred riadkom 4 bol $[g^n(a), a]$, po riadku 4 bol $[fg^n(a), a]$ a nakoniec v príkaze **end** bol obsah pracovných premenných $[h^nfg^n(a), g^n(a)]$. Výstupnej premennej z sa priradzuje hodnota y_1 , ktorej obsah je v žiadanom tvare.

Kapitola 2

Správnošť programov

Kapitola 3

Sémantika programov

Literatúra

- [1] IGOR PRÍVARA, *Základy teórie programovania*, Fakulta matematiky, fyziky a informatiky UK, Bratislava 19??