Teorija programskih jezikov: 1. izpit

31. januar 2020

Čas pisanja je 180 minut. Doseči je možno 80 točk. Veliko uspeha!

1. naloga (20 točk)

V λ-računu lahko vsako naravno število n predstavimo s *Churchevim naravnim številom*

$$e_n = \lambda s.\lambda z.\underbrace{s(s(\cdots(s}_{n}\ z)\cdots))$$

a) Zapišite vse korake v evalvaciji izraza $(e_2\,(\lambda n.2 * n))\,1$.

b) Napišite funkcijo f, ki vsako Churchevo naravno število pretvori v običajno naravno število. Veljati mora torej $f\ e_n \leadsto^* n$.

c) Izračunajte najbolj splošen tip izraza e_2.

2. naloga (20 točk)

Operacijsko semantiko programskega jezika IMP z ukazi

$$c ::= \texttt{if}\ b\ \texttt{then}\ c_1\ \texttt{else}\ c_2 \mid \texttt{while}\ b\ \texttt{do}\ c \mid c_1; c_2 \mid \ell := e \mid \texttt{skip}$$

smo običajno podali z velikimi koraki, torej z relacijami

$$s, e \Downarrow n, \qquad s, b \Downarrow p, \qquad s, c \Downarrow s',$$

vendar bi lahko podobno storili tudi z malimi koraki

$$s, e \leadsto e', \qquad s, b \leadsto b', \qquad s, c \leadsto s', c'$$

Pri tem aritmetični izrazi s koraki končajo, ko dosežejo število n, Booleovi izrazi takrat, ko dosežejo logično vrednost p, ukazi pa takrat, ko dosežejo $s, \texttt{skip}$.

a) Podajte pravila, ki določajo relacijo $s, c \leadsto s', c'$. Relacij za aritmetične in Booleove izraze ni treba pisati.

b) Jezik IMP razširimo s prepletenim izvajanjem $c_1 \leftrightharpoons c_2$, ki najprej izvede en korak ukaza c_1, nato en korak ukaza c_2, nato naslednji korak ukaza c_1 in tako naprej. Če je ukaz, ki je na vrsti, z izvajanjem končal, do konca izvedemo preostali ukaz. Zapišite dodatna pravila, s katerimi morate razširiti operacijsko semantiko.

3. naloga (20 točk)

Dana naj bo domena D in naj bo $\mathit{fix}\colon [D \to D] \to D$ preslikava, ki vsaki zvezni preslikavi $f : D \to D$ priredi njeno najmanjšo fiksno točko $\mathit{fix}(f) \in D$. Dokažite, da je preslikava fix zvezna.

4. naloga (20 točk)

Imejmo λ-račun samo z Booleovimi vrednostmi in funkcijami

$$e ::= x \mid \texttt{true} \mid \texttt{false} \mid \texttt{if}\, e\, \texttt{then}\, e_1\, \texttt{else}\, e_2 \mid \lambda x.e \mid e_1\, e_2$$

ter z običajnimi pravili za neučakano operacijsko semantiko malih korakov in določanje tipov. Ker v jeziku nimamo rekurzije, vsak program, ki ima tip, tudi konvergira, vendar preprosta indukcija žal ne zadošča za dokaz.[1]

Namesto tega za vsak tip A definirajmo množico *dobrih izrazov* $\mathscr{D}_A$ kot

$$\mathscr{D}_{\texttt{bool}} = \{e \mid (e \leadsto^* \texttt{true}) \vee (e \leadsto^* \texttt{false})\} \qquad \mathscr{D}_{A_1 \to A_2} = \{e \mid \forall e' \in \mathscr{D}_{A_1}. e\, e' \in \mathscr{D}_{A_2}\}$$

in dokaz razbijemo na tri dele...

a) Dokažite, da vsi dobri izrazi konvergirajo, torej da za vsak tip A in vsak $e \in \mathscr{D}_A$ obstaja vrednost v, da velja $e \leadsto^* v$.

b) Dokažite, da za vsak tip A iz $e \leadsto e'$ in $e' \in \mathscr{D}_A$ sledi $e \in \mathscr{D}_A$.

c) Dokažite, da za vsak izraz $x_1 : A_1, \dots, x_n : A_n \vdash e : A$ in vse vrednosti $v_1 \in \mathscr{D}_{A_1}, \dots, v_n \in \mathscr{D}_{A_n}$ velja tudi $e[v_1/x_1, \dots, v_n/x_n] \in \mathscr{D}_A$. Torej, vsak izraz, ki ima tip, je dober, če vse proste spremenljivke zamenjamo z dobrimi vrednostmi.

[1] Za čast in slavo lahko poiščete izraz e, ki divergira, torej da obstaja neskončno zaporedje $e \leadsto e_1 \leadsto e_2 \leadsto \cdots$