Teorija programskih jezikov: 1. izpit

31. januar 2020

Čas pisanja je 180 minut. Možno je doseči 20 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 $(e_2\ 1)\,(\lambda n.2 * n)$

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

c) Izračunajte najbolj splošen tip izraza e_2.

2. naloga (20 točk)

Operacijsko semantiko programskega jezika IMP 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

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

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 $s, c \rightsquigarrow s', c'$.

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. Ko prvi izmed ukazov konča z izvajanjem, preostanek drugega ukaza izvedemo do konca. 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 $f \colon D \to D$ 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

$$A ::= \texttt{bool} \mid A_1 \to A_2 \qquad e ::= x \mid \texttt{true} \mid \texttt{false} \mid \texttt{if}\ e\ \texttt{then}\ e_1\ \texttt{else}\ e_2 \mid \lambda x.e \mid e_1\,e_2$$

ter z običajnimi pravili za neučakano operacijsko semantiko malih korakov in določanje tipov. Definirajmo $e\downarrow\ = (\exists v. e \rightsquigarrow^* v)$. Ker v jeziku nimamo rekurzije, za vsak program $\vdash e : A$ velja tudi $e\downarrow$, vendar preprosta indukcija žal ne zadošča za dokaz.

Namesto tega za vsak tip A definirajmo množico *normalnih izrazov* $\mathcal{N}_A$ kot

$$\mathcal{N}_{\texttt{bool}} = \{e \mid (\vdash e : \texttt{bool}) \wedge e\downarrow\} \qquad \mathcal{N}_{A_1 \to A_2} = \{e \mid (\vdash e : A_1 \to A_2) \wedge \forall e' \in \mathcal{N}_{A_1}. e\ e' \in \mathcal{N}_{A_2}\}$$

in dokaz razbijemo na dva dela...

a) Dokažite, da za vsak tip A in vsak $e \in \mathcal{N}_A$ velja $e\downarrow$.

b) Dokažite, da za vsak izraz $x_1 : A_1, \ldots, x_n : A_n \vdash e : A$, vse vrednosti $v_1 \in \mathcal{N}_{A_1}, \ldots, v_n \in \mathcal{N}_{A_n}$ velja tudi $e[v_1/x_1, \ldots, v_n/x_n] \in \mathcal{N}_A$. Torej, vsak izraz, ki ima tip, je normalen, če vse proste spremenljivke zamenjamo z normalnimi izrazi.