Programowanie 2009 Programming 2009

Egzamin rozszerzony poprawkowy Extended repetition exam

8 września 2009 September 8, 2009

Czas trwania egzaminu: 180 minut. Oceny: Duration of the exam: 180 minutes. Grades:

punkty/points	ocena/grade
0–14	2.0
15–17	3.0
18–20	3.5
21–23	4.0
24–26	4.5
27–30	5.0

Zadanie 1 (10 pkt). Denotację instrukcji `repeat` p `until` b możemy zdefiniować jako najmniejszą funkcję $[\![\texttt{repeat}\ p\ \texttt{until}\ b]\!]$ spełniającą następującą zależność rekurencyjną:

Problem 1 (10 p). We can define the denotation of the instruction `repeat` p `until` b as the smallest function $[\![\texttt{repeat}\ p\ \texttt{until}\ b]\!]$ satisfying the following recursive equation:

$$[\![\texttt{repeat}\ p\ \texttt{until}\ b]\!]\pi \quad = \quad \begin{cases} [\![p]\!]\pi, & \text{if } \pi \in \mathrm{Dom}([\![p]\!]) \text{ and } [\![b]\!]([\![p]\!]\pi) = \mathsf{T}; \\ [\![\texttt{repeat}\ p\ \texttt{until}\ b]\!]([\![p]\!]\pi), & \text{if } \pi \in \mathrm{Dom}([\![p]\!]), \\ & [\![p]\!]\pi \in \mathrm{Dom}([\![\texttt{repeat}\ p\ \texttt{until}\ b]\!]), \\ & \text{and } [\![b]\!]([\![p]\!]\pi) = \mathsf{F}; \\ \text{undefined}, & \text{otherwise.} \end{cases}$$

Niech Π będzie przestrzenią stanów maszyny. Zdefiniuj funkcjonał $\Phi : \Pi^{\subseteq\Pi} \to \Pi^{\subseteq\Pi}$ o tej własności, że $h \in \Pi^{\subseteq\Pi}$ spełnia podaną wyżej zależność rekurencyjną wtedy i tylko wtedy, gdy jest punktem stałym Φ. Pokaż, że Φ jest ciągły. Wywnioskuj stąd (przytocz odpowiednie twierdzenie omawiane w poprzednim semestrze), że definicja denotacji instrukcji `repeat` jest poprawna.

Let Π be the set of states of the machine. Define a functional $\Phi : \Pi^{\subseteq\Pi} \to \Pi^{\subseteq\Pi}$ such that $h \in \Pi^{\subseteq\Pi}$ satisfies the recursive equation given above if and only if it is a fixpoint of Φ. Prove that Φ is continuous. Conclude (cite an appropriate theorem considered in the previous semester) that the definition of the denotation of the `repeat` instruction is correct.

Zadanie 2 (10 pkt). Rozważmy sygnaturę złożoną z symboli $0/0$, $'/1$ i $+/2$. Niech

Problem 2 (10 p). Consider a signature consisting of symbols $0/0$, $'/1$ and $+/2$. Let

$$E \quad = \quad \{x + y' = (x + y)',\ x + 0 = x\}.$$

Udowodnij, że algebra początkowa dla zbioru równości E jest izomorficzna ze zbiorem liczb naturalnych z zerem, nastepnikiem i dodawaniem.

Prove that the initial algebra for the set of equations E is isomorphic to the set of natural numbers with zero, successor and addition.

Zadanie 3 (10 pkt). Rozważmy następujący wariant rachunku lambda z typami prostymi:

Problem 3 (10 p). Consider the following variant of the simply typed lambda calculus:

$$\begin{aligned} t &::= x \mid t_1 t_2 \mid \lambda x : \sigma.t \\ \sigma &::= o \mid \sigma_1 \to \sigma_2 \end{aligned}$$

gdzie o jest stałą (o oznacza typ *bazowy*, taki jak `int`). Nie ma zmiennych typowych. Jest to zatem język, w którym typy są monomorficzne i występują w termach *explicite* (tak jak w Pascalu). Reguły typowania są następujące:

where o is a constant (o stands for a *base* type, like `int`). There are no type variables. Hence this is a language in which types are monomorphic and occur *explicite* in terms (like in Pascal). The typing rules are as follows:

$$\frac{}{\Gamma, x : \sigma \vdash x : \sigma} \qquad \frac{\Gamma \vdash t_1 : \sigma \to \tau \quad \Gamma \vdash t_2 : \sigma}{\Gamma \vdash t_1 t_2 : \tau} \qquad \frac{\Gamma, x : \sigma \vdash t : \tau}{\Gamma \vdash \lambda x : \sigma . t : \sigma \to \tau}$$

Wyrażenie jest *zamknięte*, jeśli nie zawiera zmiennych wolnych. Wprowadzamy pojęcie *rzędu* typu:

An term is *closed* if does not contain free variables. We introduce the notion of the *order* of a type:

$$\begin{aligned} \mathrm{order}(o) &= 1, \\ \mathrm{order}(\sigma_1 \to \ldots \to \sigma_n \to o) &= 1 + \max(\mathrm{order}(\sigma_1), \ldots, \mathrm{order}(\sigma_n)). \end{aligned}$$

Moc typu, oznaczana $\mathrm{card}(\sigma)$, to liczba różnych zamkniętych wyrażeń tego typu w postaci normalnej (utożsamiamy wyrażenia różniące się jedynie nazwami zmiennych związanych). Napis $\sigma^k \to \tau$ oznacza $\sigma \to \ldots \to \sigma \to \tau$, gdzie σ występuje k razy. Formalnie: $\sigma^0 \to \tau = \tau$ oraz $\sigma^{k+1} \to \tau = \sigma \to \sigma^k \to \tau$. Udowodnij twierdzenie:

1. Jeśli $\mathrm{order}(\sigma) = 1$, to $\sigma = o$ i $\mathrm{card}(\sigma) = 0$.
2. Jeśli $\mathrm{order}(\sigma) = 2$, to $\sigma = o^k \to o$ i $\mathrm{card}(o^k \to o) = k$.
3. Jeśli $\mathrm{order}(\sigma) > 2$, to $\mathrm{card}(\sigma) = 0$ lub $\mathrm{card}(\sigma) = \infty$. Ponadto $\mathrm{card}(\sigma \to \tau) = \infty$ wtedy i tylko wtedy, gdy $\mathrm{card}(\sigma) = 0$ lub $\mathrm{card}(\tau) = \infty$.

The cardinality of a type, denoted $\mathrm{card}(\sigma)$, is the number of different closed terms of that type in normal form (we identify terms that differ only in names of bound variables). The formula $\sigma^k \to \tau$ stands for $\sigma \to \ldots \to \sigma \to \tau$, where σ occurs k times. Formally, $\sigma^0 \to \tau = \tau$, and $\sigma^{k+1} \to \tau = \sigma \to \sigma^k \to \tau$. Prove the theorem:

1. If $\mathrm{order}(\sigma) = 1$, then $\sigma = o$ and $\mathrm{card}(\sigma) = 0$.
2. If $\mathrm{order}(\sigma) = 2$, then $\sigma = o^k \to o$ and $\mathrm{card}(o^k \to o) = k$.
3. If $\mathrm{order}(\sigma) > 2$, then $\mathrm{card}(\sigma) = 0$ or $\mathrm{card}(\sigma) = \infty$. Moreover $\mathrm{card}(\sigma \to \tau) = \infty$ if and only if $\mathrm{card}(\sigma) = 0$ or $\mathrm{card}(\tau) = \infty$.