Ćwiczenia

Wojciech Jedynak Paweł Wieczorek

22 września 2011

Meta notatki

Proponuję wybrać jakąś notację dla teorii typów Martina-Löfa. Może tę z książki „Programming in...”?

Czy chcemy mieć narysowane drzewa w proponowanych rozwiązań? W książce Martina-Löf-a on czasem rysuje drzewa by coś pokazać. Musielibyśmy się zdecydować by było konsekwentnie.

Zadania do których nie zrobimy rozwiazań wzorcowych będą do wywalenia.

Zaznaczam przy zadaniach teoretycznych które można zrobić w Agdzie, możemy zrobić tak że dajemy ludziom te zadania + definicje w Agdzie, i ich wola czy zrobią tylko w Agdzie czy tylko na papierze, czy tu i tu.

1 Agda i teoria

1.1 Proste

Zadanie 1 (Da się rozwiązać w Agdzie)**.** *Ustalmy typy A oraz rodzinę $B(a)$ dla każdego $a \in A$ oraz $C(a, a')$ dla każdych $a, a' \in A$. Zbuduj termy o następujących typach:*

$$((\Pi x \in A)\ (\Pi y \in A)\ C(x, y)) \to (\Pi y \in A)\ (\Pi x \in A)\ C(x, y)$$

$$((\Sigma x \in A)\ (\Pi y \in A)\ C(x, y)) \to (\Pi y \in A)\ (\Sigma x \in A)\ C(x, y)$$

$$((\Pi x \in A)\ (\Pi y \in A)\ B(x) \to B(y)) \to (\Pi x \in A)\ (\Pi y \in A)\ \neg B(y) \to \neg B(x)$$

$$(\neg(\Sigma x \in A)\ B(x)) \to (\Pi x \in A)\ \neg B(x)$$

$$((\Pi x \in A)\ B(x)) \to (\Pi x \in A)\ B(x)$$

Zadanie 2 (Da się rozwiązać w Agdzie)**.** *Ustalmy typy A, B, zbuduj termy o następujących typach:*

$$(\Pi s \in A)\ s \equiv_A s$$

$$(\Pi s \in A)(\Pi t \in A)\ s \equiv_A t \to t \equiv_A s$$

$$(\Pi s \in A)(\Pi t \in A)(\Pi r \in A)\ s \equiv_A t \to t \equiv_A r \to s \equiv_A r$$

$$(\Pi f \in A \to B)(\Pi s \in A)(\Pi t \in A)\ s \equiv_A t \to f\ t \equiv_B f\ s$$

Zadanie 3 (Da się rozwiązać w Agdzie). *Zdefiniuj dodawanie i mnożenie na liczbach naturalnych, a następnie skonstruuj termy o następujących typach:*

$$(\Pi m \in N)\ add\, 0\ m \equiv m$$

$$(\Pi n \in N)\ add\, n\ 0 \equiv n$$

$$(\Pi n \in N)\ (\Pi m \in N)\ add\, n\ m \equiv add\, m\ n$$

$$(\Pi m \in N)\ mult\, 0\ m \equiv 0$$

$$(\Pi n \in N)\ mult\, n\ 0 \equiv 0$$

$$(\Pi n \in N)\ (\Pi m \in N)\ mult\, n\ m \equiv mult\, m\ n$$

$$(\Pi n \in N)\ (\Pi m \in N)\ mult\, (add\, 1\ n)\ m \equiv add\, n\ (mult\, m\ n)$$

$$(\Pi n \in N)\ (\Pi m \in N)\ mult\, (add\, 2\ n)\ m \equiv add\, (mult\, 2\ n)\ (mult\, m\ n)$$

Zadanie 4 (Da się rozwiązać w Agdzie). *Zdefiniuj poprzednik na liczbach naturalnych, a następnie zbuduj termy o następujących typach:*

$$pred\, 0 \equiv 0$$

$$(\Pi n \in N)\ pred\, (add\, 1\ n) \equiv n$$

$$(\Pi n \in N)$$

Zadanie 5 (Da się rozwiązać w Agdzie). *Ustalmy typ A, zbubuj funkcję $toCh$ o następującym typie,*

$$N \to (A \to A) \to A \to A$$

a następnie termy o takich typach:

$$(\Pi f \in A \to A)\ (\Pi x \in A)\ toCh\, 0\ f\ x \equiv x$$

$$(\Pi f \in A \to A)\ (\Pi x \in A)\ toCh\, (add\, 1\ n)\ f\ x \equiv f\ (toCh\, n\ f\ x)$$

$$(\Pi n \in N)\ (\Pi m \in N)\ (\Pi f \in A \to A)\ (\Pi x \in A)\ toCh\, (add\, n\ m)\ f\ x \equiv toCh\, n\ f\ (toCh\, m\ f\ x)$$

Zadanie 6 (Da się rozwiązać w Agdzie). *Ustalmy dowolne typy A, B, C. Pokaż, że typy $A \to B \to C$ oraz $A \times B \to C$ są izomorficzne. To jest, oprócz zdefiniowana dobrze znanych funkcji zbuduj także dowód że są swoimi odwrotnościami, tzn termy o następujących typach:*

$$(\Pi f \in A \to B \to C)\ curry\, (uncurry\, f)\ \equiv f$$

$$(\Pi f \in A \times B \to C)\ uncurry\, (curry\, f)\ \equiv f$$

(sprawdzić czy się da, czy trzeba dodać jeszcze argumenty, tzn $(\cdots f \cdots)\ x\ y \equiv f\ x\ y$

1.2 Bardziej teoretyczne

Zadanie 7 (Da się rozwiązać w Agdzie). *Ustalmy typ A, zakoduj za pomocą W-typów typ $Maybe\ A$ znany z Haskella.*

Zadanie 8 (NIE da się rozwiązać w Agdzie). *W książce „Intuinistic type theory” pojawia się dodatkowa reguła wnioskowania dotyczaca równości:*

$$\frac{H \in x \equiv_A y}{x = y : A}$$

Mówi ona, że jeżeli posiadamy dowód, że dwa termy są równe to są one konwertowalne. Ta reguła sprawia, że type-checking jest nierozstrzygalny, a taką równość i teorię typów nazywany ekstensjonalną. Korzystając z tej reguły udowodnij ekstensjonalność funkcji, tzn pokaż że w ekstensjonalnej teorii typów dla ustalonych typów A, B możemy zbudować term o typie

$$(\Pi f \in A \to B)\ (\Pi g \in A \to B)\ ((\Pi x \in A)\ f\ x \equiv g\ x) \to f \equiv g$$

Zadania nie da się rozwiązać w Agdzie ponieważ nie można rozszerzać systemu o nowe reguły wnioskowania. (Sprawdzić czy η-ekspansja jest potrzebna, czy jest dowodliwa z tą równością)

Zadanie 9. *Skonstruuj negację bitową, tzn term $negb : N_2 \to N_2$ a następnie skontruuj term o typie:*

$$(\Pi b \in N_2)\neg(b \equiv negb\ b)$$

Wskazówka: Trudność to dowód że 0_2 jest różne od 1_2, ponieważ nie mamy typów indukcyjnych to to nie jest oczywiste, trzeba użyć uniwersum jak z czwartym aksjomatem Peano na seminarium.

Zadanie 10 (Da się rozwiązać w Agdzie). *Udowodnij, że nie istnieje funkcja z liczb naturalnych w ciągi zero-jedynkowych, która ma funkcję odwrotną. To jest skonstruuj term thm o następującym typie:*

$$\neg(\Sigma f \in N \to BinSeq)\,(\Sigma g \in BinSeq \to N)\,(\Pi s \in BinSeq)\,f(g\,s) \equiv s$$

gdzie $BinSeq$ oznacza $N \to N_2$.

Wskazówka: Dowód tego twierdzenia to standardowy przykład metody przekątniowej, można znaleźć rozwiązanie w Whitebooku. Trudność polega na przeniesieniu tego na naturalną dedukcję (Zaskakujące może być, że to twierdzenie jest konstruktywne!).

Propozycja rozwiązania. Zanim sformalizujemy metodę przekątniową przypomnijmy sobie ten dowód, by ustalić co konkretnie chcemy uzyskać.

Pokażmy, że dla dowolnych funkcji f i g potrafimy dojśc do sprzeczności o ile g jest odwrotnością f. Stwórzmy „przekątniowy” ciąg zero-jedynkowy $h : BinSeq$:

$$h = \lambda n.\mathrm{negb}\ (f\ n\ n)$$

Element tego ciągu o numerze n jest równy negacji n-tego elementu w n-tym ciągu w wyznaczonej numeracji przez funkcję f. Za pomocą funkcji g możemy uzyskać numer tego ciągu, niech $n_h = g\ h$. Teraz, zauważmy że

$$h\ n_h = \mathrm{negb}\ (f\ n_h\ n_h) = \mathrm{negb}\ (f\ (g\ h)\ n_h) = \mathrm{negb}\ (h\ n_h)$$

czyli sprzeczność.

Możemy teraz zacząć zastanawiać się jak przenieść powyższe rozumowanie na naturalną dedukcję, musimy spróbować skonstruować funkcję *diagonal* o następującym typie:

$$(\Pi f \in N \to BinSeq)\,(\Pi g \in BinSeq \to N)\ \to \big((\Pi s \in BinSeq)\, f(g\,s) \equiv s\big) \to N_0$$

Chcemy aby odpowiadała ona przedstawionemu rozumowaniu. Term zacząć pisać prosto:

$$\text{diagonal} = \lambda f.\lambda g.\lambda C.\ \boxed{?}$$

W miejscu $\boxed{?}$ chcemy skonstruować absurdalną wartość. Ale jak? Sprzeczność uzyskaliśmy pokazując, że $h\ n_h = \text{negb}\ (h\ n_h)$, bo wiemy że dla dowolnego b zachodzi $b \neq \text{negb}\ b$.

Wykorzystajmy te szczegóły w praktyce: z poprzedniego zadania mamy term Hnegb : $(\Pi b \in N_2)\neg(b \equiv \text{negb}\ b)$, czyli pamiętając co rozumiemy jako negację - jesteśmy w posiadaniu metody, która z dowodu $b \equiv \text{negb}\ b$ konstruuje absurdalną wartość. Wykorzystajmy tę metodę dla $h\ n_h$.

$$\text{diagonal} = \lambda f.\lambda g.\lambda C.\ \text{Hnegb}\ (h\ n_h)\ Heq$$

gdzie

$$Heq = \boxed{?} : h\ n_h \equiv \text{negb}\ (h\ n_h)$$

By skonstruować świadka tej równości musimy przeanalizować ciąg równości w oryginalnym rozumowaniu - pierwsze dwie

$$h\ n_h = \text{negb}\ (f\ n_h\ n_h) = \text{negb}\ (f\ (g\ h)\ n_h)$$

mamy z definicji h oraz n_h. Czyli interesuje nas jedynie term typu

$$\text{negb}\ (f\ (g\ h)\ n_h) \equiv \text{negb}\ (h\ n_h)$$

...

–

Dowód twierdzenia, to funkcja która ze świadka istnienia takich funkcji f i g ma konstruować wartość absurdalną, która jedyne co musi zrobić to rozpakować dane z argumentu i zaaplikować do nich funkcję *diagonal*.

$$thm = \lambda H.\ \Sigma\text{-elim}\ (\lambda f.\lambda H'.\ \Sigma\text{-elim}\ (\lambda g.\lambda H''.\ \text{diagonal}\ f\ g\ H'')\ H')\ H$$

□

2 Tylko Agda