UNIVERZA V LJUBLJANI
FAKULTETA ZA MATEMATIKO IN FIZIKO

Matematika – 1. stopnja

Tomaž Stepišnik Perdih

Generiranje izrekov iz tipov funkcij v polimorfnem lambda računu

Delo diplomskega seminarja

Mentor: izred. prof. dr. Andrej Bauer
Somentor: asist. dr. Matija Pretnar

Ljubljana, 2013

Kazalo

Generiranje izrekov iz tipov funkcij v polimorfnem lambda računu

Povzetek

Če tipe v polimorfnem lambda računu, razširjenem s seznami, interpretiramo z relacijami, lahko iz tipa polimorfne funkcije izpeljemo izrek, ki mu zadošča vsaka funkcija tega tipa. Na ta način lahko preprosto dobimo veliko izrekov, s pomočjo katerih lahko izvajanje programa bistveno pospešimo.

Generating theorems from types of functions in polymorphic lambda calculus

Abstract

If we interpret types in polymorphic lambda calculus extended with lists using relations, we can derive theorems from types of polymorphic functions. Each function of the same type satisfies the theorem derived from that type. This enables us to easily derive theorems that can speed up running times of programs significantly.

Math. Subj. Class. (2010): 68Q55

Ključne besede: polimorfni lambda račun, parametrični polimorfizem, denotacijska semantika

Keywords: polymorphic lambda calculus, parametric polymorphism, denotational semantics

1. Uvod

Glasbo imamo lahko v različnih oblikah: na USB ključu, gramofonski plošči, avdio kaseti, ..., vendar pa vsi vemo, da glasbe na gramofonski plošči ne moremo poslušati s predvajalnikom kaset. Da bi podobno sklepanje omogočili tudi računalnikom, v programskih jezikih obstajajo različni podatkovni tipi, na primer cela števila, seznami, nizi, itn. Tipi računalniku omogočajo lažje odkrivanje napak, saj če nekdo funkcijo, ki v seznamu nizov obrne vrstni red elementov, želi uporabiti na naravnem številu, je očitno prišlo do napake. V tem primeru bi taki funkciji določili, da slika iz seznamov nizov v sezname nizov.

Vendar to programiranje tudi zaplete, saj če bi sedaj želeli obrniti elemente v seznamu naravnih števil, bi morali definirati novo funkcijo, ki bi sicer delovala enako, le na drugem tipu podatkov. Zato se splača uporabljati polimorfne funkcije, to so funkcije, ki nimajo točno določenega tipa, s katerim operirajo, ampak jim tip podamo podobno kot ostale argumente.

Iz tipa funkcije pa lahko tudi izpeljemo izrek, ki mu mora vsaka funkcije takega tipa zadoščati. To je tudi glavna tema diplome. Osnova za nalogo je članek *Theorems for free!*[4], avtorja Philipa Wadlerja. Na začetku je definiran preprost programski jezik in predstavljena uporabnost tipov. Glavni del naloge je interpretacija tipov in izrazov v definiranem programskem jeziku, s čimer pridemo do izreka, ki nam omogoča izpeljavo izrekov iz tipov funkcij. Na koncu je dodanih še nekaj primerov, kako se ti izreki izpeljejo.

Rad bi se zahvalil Matiji Pretnarju, s katerim sem pri pisanju naloge največ sodeloval. Ko sem začel, niti nisem dobro vedel, v kaj se spuščam, samo da se mi tema zdi zanimiva. Področje pa mi je bilo popolnoma neznano. Tekom celega leta si je Matija pogosto jemal čas za sestanke z mano in mi potrpežljivo razlagal njemu najverjetneje preproste stvari, nekatere tudi večkrat. Zahvalil bi se tudi Andreju Bauerju za vso pomoč in svetovanje pri kočljivih problemih. Obema iskrena hvala!

2. Polimorfni lambda račun

Diplomska naloga bo temeljila na polimorfnem lambda računu, razširjenem s seznami. Za začetek si bomo pogledali samo *razširjeni lambda račun*. To je programski jezik, sestavljen iz naslednjih *izrazov*:

$$\begin{aligned}\textbf{Izraz } t ::= {} & \text{true} \mid \text{false} \mid \text{if } t_1 \text{ then } t_2 \text{ else } t_3 \mid 0 \mid \text{succ } t_1 \mid \text{pred } t_1 \mid \text{iszero } t_1 \mid [\,] \mid \\ & t_1 :: t_2 \mid \text{match } t_1 \text{ with } [\,] \mapsto t_2 \mid x_1 :: x_2 \mapsto t_3 \mid x \mid \lambda x.t_1 \mid t_1\ t_2\end{aligned}$$

Primer 2.1. V tem jeziku vidimo, da je 0 veljaven izraz. Ker je tudi succ t veljaven izraz za poljuben veljaven izraz t, sta tudi izraza succ 0 in succ succ 0 veljavna.

Na kratko si poglejmo intuitivni pomen izrazov. Izraza true in false sta logični konstanti. Z izrazom if t_1 then t_2 else t_3 se glede na pogoj t_1 odločimo med izrazoma t_2 in t_3. Z izrazi 0, succ t_1 (naslednik) in pred t_1 (predhodnik) predstavimo naravna števila, iszero t_1 pa preveri, če je argument t_1 enak 0. Izraz $[\,]$ predstavlja prazen seznam, $t_1 :: t_2$ pa seznam, ki se začne z elementom t_1, naprej pa je enak seznamu t_2. Izraz match t_1 with $[\,] \mapsto t_2 \mid x_1 :: x_2 \mapsto t_3$ vrne t_2, če je seznam t_1 prazen, če pa je t_1 oblike $x_1 :: x_2$, pa vrne izraz t_3. Izraz x predstavlja spremenljivko, $\lambda x.t_1$ funkcijo,

ki sprejme argument x in vrne izraz t_1. Z izrazom $t_1\ t_2$ pa funkcijo t_1 uporabimo na argumentu t_2.

Za izraz rečemo, da je *zaprt*, če ne vsebuje nobene proste spremenljivke. Tako je izraz $\lambda x.t$ zaprt, če se v izrazu t prosto pojavi samo spremenljivka x, izraz match ... pa je zaprt, če sta t_1 in t_2 zaprta, v t_3 pa lahko prosto nastopata samo spremenljivki x_1 in x_2. Zaprtim izrazom pravimo *programi*, označevali pa jih bomo s črko p.

Kako bi računalnik obravnaval oziroma vrednotil programe, bomo opisali z *operacijsko semantiko malih korakov*. Pravila za vrednotenje bomo podali z relacijo $p \rightsquigarrow p'$, ki pomeni, da se program p v enem koraku (zato semantika malih korakov) pretvori v program p'. Zapis

$$\frac{p_1 \rightsquigarrow p_1'}{p_2 \rightsquigarrow p_2'}$$

pomeni, da če se program p_1 pretvori v program p_1', se program p_2 pretvori v program p_2'. Takšen zapis bo uporabljen večkrat tekom diplomske naloge, vedno pa velja, da so nad črto naštete predpostavke (lahko jih je več), pod črto pa sklep, ki iz njih sledi. Pri pravilih, ki nimajo predpostavk, bomo črto kar izpustili.

Pri semantiki malih korakov moramo določiti tudi programe, pri katerih se vrednotenje konča. Tem programom pravimo *vrednosti*, označili pa jih bomo s črko v. V našem jeziku za vrednosti izberemo:

$$\textbf{Vrednost}\ v ::= \ 0 \mid \text{succ}\ v_1 \mid \text{true} \mid \text{false} \mid [\,] \mid v_1 :: v_2 \mid \lambda x.t$$

Sedaj za programe, ki še niso vrednosti, podamo pravila za vrednotenje:

$$\frac{p_1 \rightsquigarrow p_1'}{\text{if } p_1 \text{ then } p_2 \text{ else } p_3 \rightsquigarrow \text{if } p_1' \text{ then } p_2 \text{ else } p_3} \qquad \text{if true then } p_2 \text{ else } p_3 \rightsquigarrow p_2$$

$$\text{if false then } p_2 \text{ else } p_3 \rightsquigarrow p_3 \qquad \frac{p \rightsquigarrow p'}{\text{succ } p \rightsquigarrow \text{succ } p'} \qquad \frac{p \rightsquigarrow p'}{\text{pred } p \rightsquigarrow \text{pred } p'}$$

$$\text{pred (succ } v) \rightsquigarrow v \qquad \text{pred } 0 \rightsquigarrow 0 \qquad \frac{p \rightsquigarrow p'}{\text{iszero } p \rightsquigarrow \text{iszero } p'} \qquad \text{iszero } 0 \rightsquigarrow \text{true}$$

$$\text{iszero (succ } v) \rightsquigarrow \text{false} \qquad \frac{p_1 \rightsquigarrow p_1'}{p_1 :: p_2 \rightsquigarrow p_1' :: p_2} \qquad \frac{p \rightsquigarrow p'}{v :: p \rightsquigarrow v :: p'}$$

$$\frac{p_1 \rightsquigarrow p_1'}{(\text{match } p_1 \text{ with } [\,] \mapsto p_2 \mid x_1 :: x_2 \mapsto t_3) \rightsquigarrow (\text{match } p_1' \text{ with } [\,] \mapsto p_2 \mid x_1 :: x_2 \mapsto t_3)}$$

$$(\text{match } [\,] \text{ with } [\,] \mapsto p_2 \mid x_1 :: x_2 \mapsto t_3) \rightsquigarrow p_2$$

$$(\text{match } v_1 :: v_2 \text{ with } [\,] \mapsto p_2 \mid x_1 :: x_2 \mapsto t_3) \rightsquigarrow t_3[v_1/x_1, v_2/x_2] \qquad \frac{p_1 \rightsquigarrow p_1'}{p_1\ p_2 \rightsquigarrow p_1'\ p_2}$$

$$\frac{p_2 \rightsquigarrow p_2'}{v_1\ p_2 \rightsquigarrow v_1\ p_2'} \qquad (\lambda x.t)v \rightsquigarrow t[v/x]$$

Opomba 2.2. Zapis $t[v/x]$ pomeni izraz t, v katerem vse pojavitve spremenljivke x zamenjamo z vrednostjo v. Podobno zapis $t_3[v_1/x_1, v_2/x_2]$ pomeni izraz t_3, v katerem vse pojavitve x_1 zamenjamo z v_1, vse pojavitve x_2 pa z v_2. Pred menjavo po potrebi preimenujemo preostale vezane spremenljivke, če so slučajno označene enako kot podane vrednosti, na primer

$$(\lambda x.y\ x)[x/y] = (\lambda x'.y\ x')[x/y] = \lambda x'.x\ x'.$$

Primer 2.3.

Poglejmo si nekaj primerov vrednotenja izrazov.

- if iszero(pred(succ 0)) then pred 0 else [ ]
  $\rightsquigarrow$ if iszero 0 then pred 0 else [ ]
  $\rightsquigarrow$ if true then pred 0 else [ ]
  $\rightsquigarrow$ pred 0
  $\rightsquigarrow$ 0

- $(\lambda x.$match x with [ ] $\mapsto$ [ ] $\mid x_1 :: x_2 \mapsto x_2)$ (false :: [ ])
  $\rightsquigarrow$ match false :: [ ] with [ ] $\mapsto$ [ ] $\mid x_1 :: x_2 \mapsto x_2$
  $\rightsquigarrow$ x_2[false/x_1, [ ]/x_2]
  $=$ [ ]

- $(\lambda x.(\lambda y.$if x then y else 0)) true
  $\rightsquigarrow$ $\lambda y.$if true then y else 0

- succ (iszero (succ 0)) $\rightsquigarrow$ succ false

Pri zadnjem primeru se vrednotenje konča, saj ne moremo uporabiti nobenega pravila za pretvarjanje programov. Kljub temu pa nismo prišli do vrednosti. V takem primeru rečemo, da program *blokira*.

Programom, ki blokirajo, bi se radi izognili, saj povzročajo napake ob izvajanju. Dobro bi bilo, če bi lahko v naš programski jezik dodali način, kako ločiti programe, ki se zagotovo izvedejo, od tistih, ki lahko blokirajo. To dosežemo z vpeljavo *tipov*.

$$\textbf{Tip}\ T ::= \text{nat} \mid \text{bool} \mid T^* \mid T_1 \to T_2$$

Izrazom v našem jeziku bi radi po nekih pravilih določali tipe. Intuitivno bomo tip nat določali izrazom, ki nam predstavljajo naravna števila, tip bool za logični konstanti, T^* za sezname z elementi tipa T in tip $T_1 \to T_2$ za funkcije, ki jih uporabimo na argumentu tipa T_1 in vrnejo rezultat tipa T_2. Da bomo vsakemu izrazu tip določili enolično, jih najprej nekoliko dopolnimo (in s tem tudi vrednosti).

$$\textbf{Izraz}\ t ::= \ldots \mid [\,]_T \mid \text{match } t_1 \text{ with } [\,]_T \mapsto t_2 \mid x_1 :: x_2 \mapsto t_3 \mid \lambda x : T.t$$
$$\textbf{Vrednost}\ v ::= \ldots \mid [\,]_T \mid \lambda x : T.t$$

Večina jih ostane enakih, do sprememb je prišlo le pri izrazu $[\,]_T$ in stavku match, kjer praznemu seznamu sedaj določimo, katerega tipa elemente lahko vsebuje, ter izrazu $\lambda x : T.t$, kjer funkciji določimo, katerega tipa argumente sprejema.

Pri določanju tipov izrazom naletimo še na eno težavo. Dovoljeni izrazi v našem jeziku lahko vsebujejo proste spremenljivke (na primer izraz succ x). V tem primeru bo tip izraza odvisen od tipa spremenljivke x. Zato tip vedno določamo v nekem *kontekstu*.

Definicija 2.4. *Kontekst* je množica, katere elementi so pari $x : T$, vsaka spremenljivka pa se lahko pojavi v kvečjemu enem paru. Če je par $x : T$ element konteksta, pravimo, da ima v tem kontekstu spremenljivka x tip T. Kontekst ponavadi označimo s črko Γ.

Zapis $\Gamma \vdash t : T$ pomeni, da ima v kontekstu Γ izraz t tip T. Pri pravilih bomo uporabili podoben zapis kot pri vrednotenju izrazov: predpostavke bodo zapisane nad črto, sklepi, ki iz njih sledijo, pa pod njo.

Sedaj lahko podamo pravila, kako izrazom določamo tipe.

(1) $\dfrac{x : T \in \Gamma}{\Gamma \vdash x : T}$

(2) $\Gamma \vdash \text{true} : \text{bool}$

(3) $\Gamma \vdash \text{false} : \text{bool}$

(4) $\dfrac{\Gamma \vdash t_1 : \text{bool} \quad \Gamma \vdash t_2 : T \quad \Gamma \vdash t_3 : T}{\Gamma \vdash \text{if } t_1 \text{ then } t_2 \text{ else } t_3 : T}$

(5) $\Gamma \vdash 0 : \text{nat}$

(6) $\dfrac{\Gamma \vdash t : \text{nat}}{\Gamma \vdash \text{succ } t : \text{nat}}$

(7) $\dfrac{\Gamma \vdash t : \text{nat}}{\Gamma \vdash \text{pred } t : \text{nat}}$

(8) $\dfrac{\Gamma \vdash t : \text{nat}}{\Gamma \vdash \text{iszero } t : \text{bool}}$

(9) $\Gamma \vdash [\,]_T : T^*$

(10) $\dfrac{\Gamma \vdash t_1 : T \quad \Gamma \vdash t_2 : T^*}{\Gamma \vdash t_1 :: t_2 : T^*}$

(11) $\dfrac{\Gamma \vdash t_1 : T^* \quad \Gamma \vdash t_2 : T_1 \quad \Gamma, x_1 : T, x_2 : T^* \vdash t_3 : T_1}{\Gamma \vdash (\text{match } t_1 \text{ with } [\,]_T \mapsto t_2 \mid x_1 :: x_2 \mapsto t_3) : T_1}$

(12) $\dfrac{\Gamma, x : T_1 \vdash t : T_2}{\Gamma \vdash (\lambda x : T_1.t) : T_1 \to T_2}$

(13) $\dfrac{\Gamma \vdash t_2 : T_1 \to T_2 \;\; \Gamma \vdash t_1 : T_1}{\Gamma \vdash t_2\; t_1 : T_2}$

Opomba 2.5. Zapis $\Gamma, x : T$ pomeni kontekst Γ, razširjen s parom $x : T$, torej $\Gamma \cup \{x : T\}$. Iz pravila (1) vidimo, da mora kontekst vsebovati vse proste spremenljivke, ki nastopajo v izrazu, kateremu določamo tip.

Primer 2.6.

Poglejmo si nekaj primerov, kako s temi pravili izrazu določimo tip.

- Začnimo z izrazom $\lambda x : \text{bool}.\ x :: [\,]_{\text{bool}}$. Izraz $[\,]_{\text{bool}}$ ima po pravilu (9) tip bool^*. Zato ima po pravilu (10) tudi $x :: [\,]_{\text{bool}}$ tip bool^*, če je x tipa bool. Torej

ima celoten izraz po pravilu (12) tip bool $\to$ bool*.

• Določimo tip izraza if iszero 0 then [ ]$_{\text{nat}}$ else 0 :: [ ]$_{\text{nat}}$. Izraz iszero 0 ima po pravilu (8) tip bool. Ker imata izraza [ ]$_{\text{nat}}$ in 0 :: [ ]$_{\text{nat}}$ oba tip nat*, ima celoten izraz po pravilu (4) tudi tip nat*.

• Poskusimo sedaj določiti tip izraza succ (iszero (succ 0)), ki je blokiral. Izraz succ 0 ima tip nat, ker ima 0 tip nat (pravili (5) in (6)). Torej ima po pravilu (8) iszero(succ 0) tip bool. Celotnemu izrazu zato ne moremo določiti tipa po nobenem od prej naštetih pravil.

V zadnjem primeru smo videli, da programu, ki je blokiral, tipa ne moremo določiti. Da s tipi res lahko ločimo dobre programe od tistih, ki lahko blokirajo, pa pove naslednji izrek.

Izrek 2.7. *Program, ki ima tip, ne blokira.*

Opomba 2.8. Temu izreku pravimo izrek o varnosti. Ker služi zgolj kot motivacija za tipe in nima neposredne povezave s temo diplomske naloge, dokaz ni vključen. Če bralca zanima, lahko več o tem najde v [1] in [2].

Implementacija tipov prinese tudi nekaj neželenih posledic. Izraz $\lambda x : \text{nat}.x$ je funkcija, ki sprejme argument x tipa nat in ga vrne. To je torej identiteta za naravna števila. Njen tip je nat $\to$ nat. Če bi v jeziku s tipi želeli identiteto za tip bool, bi morali v tem izrazu samo zamenjati nat z bool. Za vsak tip moramo torej definirati svojo identiteto, čeprav vse delujejo popolnoma enako.

Tej slabi lastnosti tipov se bomo izognili tako, da bomo v programskem jeziku omogočili *polimorfne funkcije*, ki tipa argumentov in rezultata nimajo točno določenga vnaprej, ampak jim ga podamo pred uporabo. Takšno univerzalno identiteto bomo potem zapisali z izrazom $\Lambda X.\lambda x : X.x$, njen tip pa označili $\forall X.X \to X$.

Programskemu jeziku bomo dodali potrebne nove tipe, izraze in vrednosti, da bo podpiral polimorfne funkcije. S tem bomo dobili *razširjeni polimorfni lambda račun*.

$$\begin{aligned}
&\textbf{Tip } T \ ::= \ \ldots \mid X \mid \forall X.T \\
&\textbf{Izraz } t ::= \ \ldots \mid \Lambda X.t \mid t_T \\
&\textbf{Vrednost } v \ ::= \ \ldots \mid \Lambda X.t
\end{aligned}$$

Opomba 2.9. Podobno kot x za izraze je X spremenljivka za tipe. Z izrazom t_T pa polimorfno funkcijo t uporabimo na tipu T. Izraz $\Lambda X.t$ je zaprt, če je X edina spremenljivka, ki v izrazu t nastopa prosto.

Dodati moramo tudi potrebna nova pravila za določanje tipov izrazom in vrednotenje programov. Program $\Lambda X.t$ je že vrednost, za p_T pa velja:

$$\frac{p \rightsquigarrow p'}{p_T \rightsquigarrow p'_T} \qquad\qquad (\Lambda X.t)_T \rightsquigarrow t[T/X]$$

Naš jezik sedaj vsebuje tudi spremenljivke za tipe, zato v kontekstu dovolimo pare oblike $X : *$. Tukaj je $*$ samo oznaka, ki pove, katere spremenljivke v kontekstu so za tipe. Za razliko od navadnih spremenljivk jim ničesar ne določimo, prav nam bodo prišle v naslednjem poglavju. Novi pravili za določanje tipov sta:

$$(14)\frac{\Gamma, X : * \vdash t : T}{\Gamma \vdash \Lambda X.t : \forall X.T} \qquad (15)\frac{\Gamma \vdash t : \forall X.T_1}{\Gamma \vdash t_T : T_1[T/X]}$$

Oznaka $\Gamma, X : *$ pomeni kontekst Γ, razširjen s parom $X : *$, torej $\Gamma \cup \{X : *\}$. Zapis $t[T/X]$ označuje izraz t, v katerem so vse pojavitve spremenljivke za tipe X zamenjane s tipom T, zapis $T_1[T/X]$ pa tip T_1, v katerem vse pojavitve spremenljivke za tipe X zamenjamo s tipom T. Ponovno po potrebi preimenujemo vezane spremenljivke, da oznake ne sovpadajo.

Opomba 2.10. Izrek o varnosti velja tudi za jezik s polimorfizmom.

Primer 2.11.

- Preverimo, ali ima univerzalna identiteta $\Lambda X.\lambda x : X.x$ res želeni tip

  $$\forall X.X \to X.$$

  Izraz x ima po pravilu (1) v kontekstu, kjer ima x tip X, tudi tip X. Zato ima po (12) izraz $\lambda x : X.x$ tip $X \to X$. Celoten izraz ima tako po pravilu (14) res tip $\forall X.X \to X$.

- Označimo z id univerzalno identiteto. Določimo tip izraza id_{nat}. Vemo, da ima id tip $\forall X.X \to X$. Po pravilu (15) ima potem id_{nat} tip nat $\to$ nat.

- Poglejmo si še kako se izraz id_{nat} ovrednoti.

  $$(\Lambda X.\lambda x : X.x)_{\text{nat}} \rightsquigarrow \lambda x : \text{nat}.x$$

  To je identiteta za tip nat.

3. Semantika

V tem poglavju bomo tipe in izraze v našem programskem jeziku predstavili z matematičnimi objekti. Tipe bomo interpretirali z elementi neke množice $\mathbf{U}$, vrednosti tipa, ki ga interpretiramo z $A \in \mathbf{U}$, pa z elementi neke množice D_A. Podali bomo pravila, kako določiti elemente teh množic, s katerimi interpretiramo posamezne tipe in izraze, na koncu poglavja pa z izrekom pokazali, da so bila ta pravila smiselna.

3.1. **Semantika tipov.**

Denimo, da bi radi tipe interpretirali z elementi množice $\mathbf{U}$. Da to lahko naredimo, mora $\mathbf{U}$ zadoščati nekaterim pogojem. Najprej mora vsebovati vsaj dva elementa, ki nam predstavljata tipa nat in bool, označimo ju z N in B. Ker za poljuben tip T v našem jeziku obstaja tudi tip seznamov T^*, mora za vsak element A v množici $\mathbf{U}$ obstajati še neki element $A^* \in \mathbf{U}$. Za poljubna tipa T_1 in T_2 obstaja tudi tip $T_1 \to T_2$, zato mora tudi za poljubna elementa A in B iz $\mathbf{U}$ obstajati še neki element $A \to B \in \mathbf{U}$. Tudi $\forall X.T$ je tip za poljuben tip T (lahko odvisen od X). Zato zahtevamo, da za poljubno funkcijo $F : \mathbf{U} \to \mathbf{U}$ obstaja neki element $\forall F$ v $\mathbf{U}$.

Imamo torej preslikave $nat \mapsto N$, $bool \mapsto B$, $\to$ iz $\mathbf{U} \times \mathbf{U}$ v $\mathbf{U}$, * iz $\mathbf{U}$ v $\mathbf{U}$ in $\forall$ iz $\mathbf{U}^{\mathbf{U}}$ v $\mathbf{U}$. Za njih zahtevamo, da so skupno injektivne, zato da različnim tipom pripadajo različni elementi $\mathbf{U}$. S tem pa naletimo na težavo, saj iz injektivnosti $\forall$ sledi $|\mathbf{U}| \geqslant |\mathbf{U}^{\mathbf{U}}|$. Ker ima $\mathbf{U}$ vsaj dva elementa, je $|\mathbf{U}^{\mathbf{U}}| \geqslant |2^{\mathbf{U}}|$, kar je po Cantorjevem izreku strogo večje od $|\mathbf{U}|$. Takšna množica zato v običajni teoriji množic ne obstaja. Ker jo kljub temu želimo imeti, bomo od tu dalje privzeli, da aksiom izbire in aksiom o izključeni tretji možnosti ne veljata. Pod tema predpostavkama je obstoj takšnega $\mathbf{U}$ možen, obstajajo namreč modeli intuicionistične teorije množic, v katerih želeni $\mathbf{U}$ obstaja. Več o tem lahko bralec najde v [3]. Odslej moramo zato paziti, da vse dokažemo brez aksioma izbire in aksioma o izključeni tretji možnosti.

Definicija 3.1. *Okolje za tipe* $\bar{A}$ za kontekst Γ je preslikava, ki slika spremenljivke za tipe v Γ (torej vse spremenljivke iz parov oblike $X : *$) v elemente množice $\mathbf{U}$. Element prirejen spremenljivki X označimo z $\bar{A}(X)$.

Z okolji za tipe bi radi poljuben tip interpretirali z nekim elementom v $\mathbf{U}$. Element, ki ga z okoljem za tipe $\bar{A}$ priredimo tipu T, označimo z $[\![T]\!]\bar{A}$, določimo pa ga po naslednjih pravilih.

$$
\begin{array}{lcl}
[\![\text{nat}]\!]\bar{A} & = & N \\
[\![\text{bool}]\!]\bar{A} & = & B \\
[\![X]\!]\bar{A} & = & \bar{A}(X) \\
[\![T_1 \to T_2]\!]\bar{A} & = & [\![T_1]\!]\bar{A} \to [\![T_2]\!]\bar{A} \\
[\![T^*]\!]\bar{A} & = & ([\![T]\!]\bar{A})^* \\
[\![\forall X.T]\!]\bar{A} & = & \forall(A \in \mathbf{U} \mapsto [\![T]\!]\bar{A}[A/X])
\end{array}
$$

Zaradi naštetih pogojev za množico $\mathbf{U}$ so te definicije dobre. Nekaj dodatne pozornosti namenimo samo zadnji. Oznaka $\bar{A}[A/X]$ pomeni okolje za tipe, ki spremenljivko X slika v A, sicer pa deluje enako kot okolje $\bar{A}$. Funkcija $A \in \mathbf{U} \mapsto [\![T]\!]\bar{A}[A/X]$ zato slika iz $\mathbf{U}$ v $\mathbf{U}$, kar pomeni, da je $\forall(A \in \mathbf{U} \mapsto [\![T]\!]\bar{A}[A/X])$ res element $\mathbf{U}$.

Lema 3.2.

$$[\![T_1]\!]\bar{A}[[\![T]\!]\bar{A}/X] = [\![T_1[T/X]]\!]\bar{A}$$

Dokaz. Lema se dokaže z indukcijo po strukturi tipa T_1. Vemo, da je

$$T_1 = \text{nat} \mid \text{bool} \mid X \mid A^* \mid A \to B \mid \forall X.A,$$

kjer sta A in B neka tipa. Indukcijska predpostavka je, da lema velja za tipa A in B, iz tega pa pokažemo, da velja tudi za T_1 v vseh zgornjih primerih. □

3.2. **Semantika izrazov.**

Sedaj želimo za vsak element $A \in \mathbf{U}$ interpretirati vrednosti tipa, ki mu ta element ustreza, z elementi neke množice D_A.

- Radi bi, da nam množica D_N predstavlja vrednosti tipa nat. Naj bo $[nat]$ množica teh vrednosti, torej $[\text{nat}] = \{0, \text{succ } 0, \text{succ succ } 0, \dots\}$. Potem morata obstajati taki funkciji

$$\phi_{\text{nat}} : D_N \to [\text{nat}]$$

$$\psi_{\text{nat}} : [\text{nat}] \to D_N,$$

  da je $\phi_{\text{nat}} \circ \psi_{\text{nat}}$ identiteta na $[\text{nat}]$.

- Vrednosti tipa bool bi radi interpretirali z množico D_B. Če je $[\text{bool}] = \{\text{true}, \text{false}\}$, morata obstajati taki funkciji

$$\phi_{\text{bool}} : D_B \to [\text{bool}]$$

$$\psi_{\text{bool}} : [\text{bool}] \to D_B,$$

  da je $\phi_{\text{bool}} \circ \psi_{\text{bool}}$ identiteta na $[\text{bool}]$.

- Radi bi, da nam množica $D_{A \to B}$ predstavlja funkcije iz D_A v D_B. Če z $[D_A \to D_B]$ označimo to množico, morata obstajati taki funkciji

$$\phi_{A \to B} : D_{A \to B} \to [D_A \to D_B]$$

$$\psi_{A \to B} : [D_A \to D_B] \to D_{A \to B},$$

  da je $\phi_{A \to B} \circ \psi_{A \to B}$ identiteta na $[D_A \to D_B]$.

- Z množico D_{A*} bi radi interpretirali sezname elementov iz D_A. Če označimo množico teh seznamov z $[D_A^*]$, morata obstajati taki funkciji

$$\phi_{A*} : D_{A*} \to [D_A^*]$$

$$\psi_{A*} : [D_A^*] \to D_{A*},$$

  da je $\phi_{A*} \circ \psi_{A*}$ identiteta na $[D_A^*]$.

- Naj bo $F : \mathbf{U} \to \mathbf{U}$ funkcija. Radi bi, da nam elementi $D_{\forall F}$ predstavljajo funkcije, ki vsak $A \in \mathbf{U}$ preslikajo v neki element množice $D_{F(A)}$. Označimo z $[\forall A \in \mathbf{U}.D_{F(A)}]$ množico takih funkcij. Potem morata obstajati taki funkciji

$$\phi_{\forall F} : D_{\forall F} \to [\forall A \in \mathbf{U}.D_{F(A)}]$$

$$\psi_{\forall F} : [\forall A \in \mathbf{U}.D_{F(A)}] \to D_{\forall F},$$

  da je $\phi_{\forall F} \circ \psi_{\forall F}$ identiteta na $[\forall A \in \mathbf{U}.D_{F(A)}]$.

Opomba 3.3. Ker je ob uporabi večinoma jasno, za katere od teh funkcij ϕ in ψ gre, bomo indekse navadno kar izpustili.

Definicija 3.4. Okolje za spremenljivke $\bar{a}$ za kontekst Γ je preslikava, ki vsaki spremenljivki x v Γ (torej spremenljivkam iz parov oblike $X : T$) priredi neko vrednost $\bar{a}(x)$.

Okolji $\bar{A}, \bar{a}$ sta *usklajeni v kontekstu* Γ, če za vsak par $x : T$ v Γ velja, da je $\bar{a}(x) \in D_{[\![T]\!]\bar{A}}$. Z usklajenimi okolji želimo interpretirati poljuben izraz, ki mu lahko določimo tip. Naj velja $\Gamma \vdash t : T$. Element, ki ga priredimo izrazu t, označimo z $[\![t]\!]\bar{A}\bar{a}$ in ga določimo po sledečih pravilih.

$$
\begin{aligned}
[\![x]\!]\bar{A}\bar{a} &= \bar{a}(x) \\
[\![0]\!]\bar{A}\bar{a} &= \psi(0) \\
[\![\text{succ } t]\!]\bar{A}\bar{a} &= \psi(\text{succ } \phi([\![t]\!]\bar{A}\bar{a}))
\end{aligned}
$$

$$
[\![\text{pred } t]\!]\bar{A}\bar{a} = \begin{cases} \psi(a); & \phi([\![t]\!]\bar{A}\bar{a}) = \text{succ } a \\ \psi(0); & \phi([\![t]\!]\bar{A}\bar{a}) = 0 \end{cases}
$$

$$
[\![\text{iszero } t]\!]\bar{A}\bar{a} = \begin{cases} \psi(\text{true}); & \phi([\![t]\!]\bar{A}\bar{a}) = 0 \\ \psi(\text{false}); & \phi([\![t]\!]\bar{A}\bar{a}) = \text{succ } a \end{cases}
$$

$$
\begin{aligned}
[\![\text{true}]\!]\bar{A}\bar{a} &= \psi(\text{true}) \\
[\![\text{false}]\!]\bar{A}\bar{a} &= \psi(\text{false})
\end{aligned}
$$

$$
[\![\text{if } t_1 \text{ then } t_2 \text{ else } t_3]\!]\bar{A}\bar{a} = \begin{cases} [\![t_2]\!]\bar{A}\bar{a}; & \phi([\![t_1]\!]\bar{A}\bar{a}) = \text{true} \\ [\![t_3]\!]\bar{A}\bar{a}; & \phi([\![t_1]\!]\bar{A}\bar{a}) = \text{false} \end{cases}
$$

$$
\begin{aligned}
[\![[\,]_T]\!]\bar{A}\bar{a} &= \psi_{([\![T]\!]\bar{A})^*}([\,]) \\
[\![t_1 :: t_2]\!]\bar{A}\bar{a} &= \psi([\![t_1]\!]\bar{A}\bar{a} :: \phi([\![t_2]\!]\bar{A}\bar{a}))
\end{aligned}
$$

$$
[\![\text{match } t_1 \text{ with } [\,]_T \mapsto t_2 \mid x_1 :: x_2 \mapsto t_3]\!]\bar{A}\bar{a} = \begin{cases} [\![t_2]\!]\bar{A}\bar{a}; & \phi([\![t_1]\!]\bar{A}\bar{a}) = [\,] \\ [\![t_3]\!]\bar{A}\bar{a}[v_1/x_1, \psi(v_2)/x_2]; & \phi([\![t_1]\!]\bar{A}\bar{a}) = v_1 :: v_2 \end{cases}
$$

$$
\begin{aligned}
[\![\lambda x : T.t]\!]\bar{A}\bar{a} &= \psi(a \in D_{[\![T]\!]\bar{A}} \mapsto [\![t]\!]\bar{A}\bar{a}[a/x]) \\
[\![t_1\ t_2]\!]\bar{A}\bar{a} &= \phi([\![t_1]\!]\bar{A}\bar{a})([\![t_2]\!]\bar{A}\bar{a}) \\
[\![\Lambda X.t]\!]\bar{A}\bar{a} &= \psi(A \in \mathbf{U} \mapsto [\![t]\!]\bar{A}[A/X]\bar{a}) \\
[\![t_T]\!]\bar{A}\bar{a} &= \phi([\![t]\!]\bar{A}\bar{a})([\![T]\!]\bar{A})
\end{aligned}
$$

Opomba 3.5. Oznaka $\bar{a}[a/x]$ pomeni okolje za spremenljivke, ki spremenljivki x priredi vrednost a, na vseh ostalih spremenljivkah pa deluje enako kot okolje $\bar{a}$.

Če za vsak par okolij $\bar{A}, \bar{a}$, ki je usklajen v Γ, velja, da je $[\![t]\!]\bar{A}\bar{a} \in D_{[\![T]\!]\bar{A}}$, potem pišemo $\Gamma \models t : T$

Izrek 3.6. *Če velja $\Gamma \vdash t : T$, potem velja tudi $\Gamma \models t : T$.*

Dokaz. Pri dokazovanju izreka se večinoma samo upošteva definicije interpretacij tipov in izrazov. Dokaz poteka z indukcijo na izpeljavo sklepa $\Gamma \vdash t : T$, do katerega smo prišli z uporabo pravil za določanje tipov. Pri vsakem pravilu za določanje tipov bomo privzeli, da izrek velja za predpostavke (naštete nad črto), in pokazali, da potem velja tudi za sklep pod črto. Baza indukcije so primeri, ki predpostavk nimajo, saj nas določanje tipa izrazu vedno privede do kakšnega od teh pravil. V dokazu sta $\bar{A}$ in $\bar{a}$ poljubni okolji, usklajeni v Γ.

(1) Naj velja $\Gamma \vdash x : T$. Potem mora biti par $x : T \in \Gamma$. Zaradi usklajenosti je $[\![x]\!]\bar{A}\bar{a} = \bar{a}(x) \in D_{[\![T]\!]\bar{A}}$. Torej res velja $\Gamma \vDash x : T$.

(2) Naj velja $\Gamma \vdash \text{true} : \text{bool}$. Vemo, da je $[\![\text{true}]\!]\bar{A}\bar{a} = \psi(\text{true}) \in D_B$. Ker je $[\![\text{bool}]\!]\bar{A} = B$, velja $\Gamma \vDash \text{true} : \text{bool}$.

(3) Naj velja $\Gamma \vdash \text{false} : \text{bool}$. Vemo, da je $[\![\text{false}]\!]\bar{A}\bar{a} = \psi(\text{false}) \in D_B$. Ker je $[\![\text{bool}]\!]\bar{A} = B$, velja $\Gamma \vDash \text{false} : \text{bool}$.

(4) Naj velja $\Gamma \vdash \text{if } t_1 \text{ then } t_2 \text{ else } t_3 : T$. Da smo prišli do tega sklepa, mora veljati $\Gamma \vdash t_1 : \text{bool}$, $\Gamma \vdash t_2 : T$ in $\Gamma \vdash t_3 : T$. Indukcijske predpostavke so potem $\Gamma \vDash t_1 : \text{bool}$, $\Gamma \vDash t_2 : T$ in $\Gamma \vDash t_3 : T$.

Iz indukcijskih predpostavk vemo, da je $[\![t_1]\!]\bar{A}\bar{a} \in D_B$, zato je $\phi([\![t_1]\!]\bar{A}\bar{a})$ enako true ali false. V obeh primerih velja:

$$[\![\text{if } t_1 \text{ then } t_2 \text{ else } t_3]\!]\bar{A}\bar{a} \in \{[\![t_2]\!]\bar{A}\bar{a}, [\![t_3]\!]\bar{A}\bar{a}\}$$

Iz indukcijskih predpostavk sledi tudi, da sta $[\![t_2]\!]\bar{A}\bar{a}$ in $[\![t_3]\!]\bar{A}\bar{a}$ v množici $D_{[\![T]\!]\bar{A}}$. Zato res velja $\Gamma \vDash \text{if } t_1 \text{ then } t_2 \text{ else } t_3 : T$.

(5) Naj velja $\Gamma \vdash 0 : \text{nat}$. Vemo, da je $[\![0]\!]\bar{A}\bar{a} = \psi(0) \in D_N$. Ker je $[\![\text{nat}]\!]\bar{A} = N$, je res $\Gamma \vDash 0 : \text{nat}$.

(6) Naj velja $\Gamma \vdash \text{succ } t : \text{nat}$. To pomeni, da mora veljati $\Gamma \vdash t : \text{nat}$. Indukcijska predpostavka je torej $\Gamma \vDash t : \text{nat}$. Velja:

$$[\![\text{succ } t]\!]\bar{A}\bar{a} = \psi(\text{succ } \phi([\![t]\!]\bar{A}\bar{a}))$$

Iz indukcijskih predpostavk sledi $[\![t]\!]\bar{A}\bar{a} \in D_N$, zato je tudi $\psi(\text{succ } \phi([\![t]\!]\bar{A}\bar{a})) \in D_N$. Ker je $N = [\![\text{nat}]\!]\bar{A}$, velja $\Gamma \vDash \text{succ } t : \text{nat}$.

(7) Naj velja $\Gamma \vdash \text{pred } t : \text{nat}$. To pomeni, da mora veljati $\Gamma \vdash t : \text{nat}$. Indukcijska predpostavka je v tem primeru $\Gamma \vDash t : \text{nat}$. Velja:

$$[\![\text{pred } t]\!]\bar{A}\bar{a} = \begin{cases} \psi(a); & \phi([\![t]\!]\bar{A}\bar{a}) = \text{succ } a \\ \psi(0); & \phi([\![t]\!]\bar{A}\bar{a}) = 0 \end{cases}$$

Po indukcijski predpostavki je $[\![t]\!]\bar{A}\bar{a} \in D_N$, zato je $\phi([\![t]\!]\bar{A}\bar{a})$ enako 0 ali pa oblike succ a za neki $a \in [\text{nat}]$. V obeh primerih sledi, da je $[\![\text{pred } t]\!]\bar{A}\bar{a} \in D_N$, zato velja $\Gamma \vDash \text{pred } t : \text{nat}$

(8) Naj velja $\Gamma \vdash \text{iszero } t : \text{bool}$. To pomeni, da mora veljati $\Gamma \vdash t : \text{nat}$. Indukcijska predpostavka je potem $\Gamma \vDash t : \text{nat}$. Velja:

$$[\![\text{iszero } t]\!]\bar{A}\bar{a} = \begin{cases} \psi(\text{true}); & \phi([\![t]\!]\bar{A}\bar{a}) = 0 \\ \psi(\text{false}); & \phi([\![t]\!]\bar{A}\bar{a}) = \text{succ } a \end{cases}$$

Po indukcijski predpostavki je $[\![t]\!]\bar{A}\bar{a} \in D_N$, zato je $\phi([\![t]\!]\bar{A}\bar{a})$ enako 0 ali pa oblike succ a za neki $a \in [\text{nat}]$. V obeh primerih je $[\![\text{iszero } t]\!]\bar{A}\bar{a} \in D_B$ in zato velja $\Gamma \vDash \text{iszero } t : \text{bool}$.

(9) Naj velja $\Gamma \vdash [\ ]_T : T^*$. Vemo, da je $[\![[\ ]_T]\!]\bar{A}\bar{a} = \psi_{([\![T]\!]\bar{A})^*}([\ ]) \in [\![T^*]\!]\bar{A}$ po definiciji preslikave $\psi_{([\![T]\!]\bar{A})^*}$ in zato velja $\Gamma \vDash [\ ]_T : T^*$.

(10) Naj velja $\Gamma \vdash t_1 :: t_2 : T^*$. To pomeni, da mora veljati $\Gamma \vdash t_1 : T$ in $\Gamma \vdash t_2 : T^*$. Indukcijski predpostavki sta potem $\Gamma \vDash t_1 : T$ in $\Gamma \vDash t_2 : T^*$.

$$[\![t_1 :: t_2]\!]\bar{A}\bar{a} = \psi([\![t_1]\!]\bar{A}\bar{a} :: \phi([\![t_2]\!]\bar{A}\bar{a}))$$

Iz indukcijskih predpostavk sledi, da je $[\![t_1]\!]\bar{A}\bar{a} \in D_{[\![T]\!]\bar{A}}$ in $[\![t_2]\!]\bar{A}\bar{a} \in D_{([\![T]\!]\bar{A})^*}$ (torej seznam z elementi iz $D_{[\![T]\!]\bar{A}}$). Zato je tudi $[\![t_1]\!]\bar{A}\bar{a} :: \phi([\![t_2]\!]\bar{A}\bar{a})$ seznam z elementi iz $D_{[\![T]\!]\bar{A}}$. Po definiciji interpretacije tipov je $[\![T^*]\!]\bar{A} = ([\![T]\!]\bar{A})^*$, torej je $[\![t_1 :: t_2]\!]\bar{A}\bar{a} \in D_{[\![T^*]\!]\bar{A}}$ in res velja $\Gamma \vDash t_1 :: t_2 : T^*$.

(11) Naj velja $\Gamma \vdash \text{match } t_1 \text{ with } [\ ]_T \mapsto t_2 \mid x_1 :: x_2 \mapsto t_3 : T_1$. To pomeni, da mora veljati $\Gamma \vdash t_1 : T^*$, $\Gamma \vdash t_2 : T_1$ in $\Gamma, x_1 : T, x_2 : T^* \vdash t_3 : T_1$. Indukcijske predpostavke so torej $\Gamma \vDash t_1 : T^*$, $\Gamma \vDash t_2 : T_1$ in $\Gamma, x_1 : T, x_2 : T^* \vDash t_3 : T_1$.

$$[\![\text{match } t_1 \text{ with } [\ ]_T \mapsto t_2 \mid x_1 :: x_2 \mapsto t_3]\!]\bar{A}\bar{a} = \\ = \begin{cases} [\![t_2]\!]\bar{A}\bar{a}; & \phi([\![t_1]\!]\bar{A}\bar{a}) = [\ ] \\ [\![t_3]\!]\bar{A}\bar{a}[v_1/x_1, \psi(v_2)/x_2]; & \phi([\![t_1]\!]\bar{A}\bar{a}) = v_1 :: v_2 \end{cases}$$

Po indukcijski predpostavki je $[\![t_1]\!]\bar{A}\bar{a} \in D_{([\![T]\!]\bar{A})^*}$, zato je interpretacija dobro definirana. Sedaj ločimo dva primera.

Če je $\phi([\![t_1]\!]\bar{A}\bar{a}) = [\ ]$, je

$$[\![\text{match } t_1 \text{ with } [\ ]_T \mapsto t_2 \mid x_1 :: x_2 \mapsto t_3]\!]\bar{A}\bar{a} = [\![t_2]\!]\bar{A}\bar{a},$$

kar je po indukcijski predpostavki v množici $D_{[\![T_1]\!]\bar{A}}$.

Druga možnost je, da je $\phi([\![t_1]\!]\bar{A}\bar{a}) = v_1 :: \phi(v_2)$ za neka $v_1 \in D_{[\![T]\!]\bar{A}}$ in $v_2 \in D_{[\![T^*]\!]\bar{A}}$. Okolji $\bar{A}$ in $\bar{a}[v_1/x_1, v_2/x_2]$ sta tako usklajeni v kontekstu $\Gamma, x_1 : T, x_2 : T^*$. V tem primeru je

$$[\![\text{match } t_1 \text{ with } [\ ]_T \mapsto t_2 \mid x_1 :: x_2 \mapsto t_3]\!]\bar{A}\bar{a} = [\![t_3]\!]\bar{A}\bar{a}[v_1/x_1, v_2/x_2],$$

kar je po indukcijski predpostavki v množici $D_{[\![T_1]\!]\bar{A}}$. V obeh primerih tako velja

$$\Gamma \vDash \text{match } t_1 \text{with } [\ ]_T \mapsto t_2 \mid x_1 :: x_2 \mapsto t_3 : T_1.$$

(12) Naj velja $\Gamma \vdash (\lambda x : T_1.t) : T_1 \to T_2$. To pomeni, da mora veljati $\Gamma, x : T_1 \vdash t : T_2$ in zato je indukcijska predpostavka $\Gamma, x : T_1 \vDash t : T_2$.

$$[\![\lambda x : T_1.t]\!]\bar{A}\bar{a} = \psi(a \in D_{[\![T_1]\!]\bar{A}} \mapsto [\![t]\!]\bar{A}\bar{a}[a/x])$$

Naj bo $a \in D_{[\![T_1]\!]\bar{A}}$. Potem sta okolji $\bar{A}$ in $\bar{a}[a/x]$ usklajeni v kontekstu $\Gamma, x : T_1$, zato po indukcijski predpostavki velja $[\![t]\!]\bar{A}\bar{a}[a/x] \in D_{[\![T_2]\!]\bar{A}}$. To pomeni, da je

$$a \in D_{[\![T_1]\!]\bar{A}} \mapsto [\![t]\!]\bar{A}\bar{a}[a/x]$$

funkcija iz $D_{[\![T_1]\!]\bar{A}}$ v $D_{[\![T_2]\!]\bar{A}}$. Zato je $[\![\lambda x : T_1.t]\!]\bar{A}\bar{a} \in D_{[\![T_1 \to T_2]\!]\bar{A}}$ in velja $\Gamma \vDash (\lambda x : T_1.t) : T_1 \to T_2$.

(13) Naj velja $\Gamma \vdash t_2\ t_1 : T_2$. To pomeni, da mora veljati $\Gamma \vdash t_2 : T_1 \to T_2$ in $\Gamma \vdash t_1 : T_1$. Indukcijski predpostavki sta potem $\Gamma \vDash t_2 : T_1 \to T_2$ in $\Gamma \vDash t_1 : T_1$.

$$[\![t_2\ t_1]\!]\bar{A}\bar{a} = \phi([\![t_2]\!]\bar{A}\bar{a})([\![t_1]\!]\bar{A}\bar{a})$$

Po indukcijski predpostavki je $[\![t_2]\!]\bar{A}\bar{a} \in D_{[\![T_1 \to T_2]\!]\bar{A}}$, torej je $\phi([\![t_2]\!]\bar{A}\bar{a})$ funkcija iz $D_{[\![T_1]\!]\bar{A}}$ v $D_{[\![T_2]\!]\bar{A}}$. Ker je $[\![t_1]\!]\bar{A}\bar{a} \in D_{[\![T_1]\!]\bar{A}}$, je $[\![t_2\ t_1]\!]\bar{A}\bar{a} \in D_{[\![T_2]\!]\bar{A}}$ in tako velja $\Gamma \vDash t_2\ t_1 : T_2$.

(14) Naj velja $\Gamma \vdash \Lambda X.t : \forall X.T$. To pomeni, da mora veljati $\Gamma, X : * \vdash t : T$, indukcijska predpostavka pa je $\Gamma, X : * \vDash t : T$.

$$[\![\Lambda X.t]\!]\bar{A}\bar{a} = \psi(A \in \mathbf{U} \mapsto [\![t]\!]\bar{A}[A/X]\bar{a})$$

Po definiciji interpretacije tipov je

$$[\![\forall X.T]\!]\bar{A} = \forall(A \in \mathbf{U} \mapsto [\![T]\!]\bar{A}[A/X]).$$

Naj bo $A \in \mathbf{U}$. Okolji $\bar{A}[A/X]$ in $\bar{a}$ sta potem usklajeni v kontekstu $\Gamma, X : *$. Po indukcijski predpostavki je tako $[\![t]\!]\bar{A}[A/X]\bar{a} \in D_{[\![T]\!]\bar{A}[A/X]}$, zato je

$$A \in \mathbf{U} \mapsto [\![t]\!]\bar{A}[A/X]\bar{a}$$

funkcija, ki A preslika v element iz $D_{[\![T]\!]\bar{A}[A/X]}$. Torej je $[\![\Lambda X.t]\!]\bar{A}\bar{a} \in D_{[\![\forall X.T]\!]\bar{A}}$ in res velja $\Gamma \vDash \Lambda X.t : \forall X.T$.

(15) Naj velja $\Gamma \vdash t_T : T_1[T/X]$. To pomeni, da mora veljati $\Gamma \vdash t : \forall X.T_1$, indukcijska predpostavka pa je zato $\Gamma \vDash t : \forall X.T_1$.

$$[\![t_T]\!]\bar{A}\bar{a} = \phi([\![t]\!]\bar{A}\bar{a})([\![T]\!]\bar{A})$$

Po indukcijski predpostavki je $\phi([\![t]\!]\bar{A}\bar{a})$ funkcija, ki sprejme $A \in \mathbf{U}$ in vrne element iz $D_{[\![T_1]\!]\bar{A}[A/X]}$. Zato je $[\![t_T]\!]\bar{A}\bar{a} \in D_{[\![T_1]\!]\bar{A}[[\![T]\!]\bar{A}/X]}$ in ker je po lemi 3.2

$$[\![T_1]\!]\bar{A}[[\![T]\!]\bar{A}/X] = [\![T_1[T/X]]\!]\bar{A},$$

velja $\Gamma \vDash t_T : T_1[T/X]$. □

4. Interpretacija tipov z relacijami

V prejšnjem poglavju smo tipe interpretirali z elementi množice $\mathbf{U}$, vrednosti posameznega tipa T v okolju $\bar{A}$ pa z elementi množice $D_{[\![T]\!]\bar{A}}$. V tem poglavju bomo tipe interpretirali z relacijami med množicami D_A, tako da bodo za dovolj lepo izbrana okolja interpretacije izrazov v relaciji.

Pravimo, da je $\bar{\mathcal{A}}$ *relacijsko okolje* za kontekst Γ, če vsaki spremenljivki za tipe v Γ priredi neko relacijo. Relacije bomo označevali z zapisom $R : A \leftrightarrow B$, ki pomeni, da je R relacija med množicama D_A in D_B (torej R je podmnožica $D_A \times D_B$).

Denimo, da je $\bar{\mathcal{A}}$ neko relacijsko okolje. Relacijo, ki jo to okolje priredi spremenljivki za tipe X, označimo z $\bar{\mathcal{A}}(X)$, relacijo, s katero interpretiramo tip T v okolju $\bar{\mathcal{A}}$, pa z $[\![T]\!]\bar{\mathcal{A}}$. Podajmo pravila, po katerih bomo določali relacije, ki bodo interpretirale posamezne tipe.

- Tip nat interpretiramo z identično relacijo na množici D_N. Velja torej $[\![\text{nat}]\!]\bar{\mathcal{A}} : N \leftrightarrow N$ in $(x, y) \in [\![\text{nat}]\!]\bar{\mathcal{A}}$ natanko takrat, ko $x = y$.

- Tip bool interpretiramo z identično relacijo na množici D_B. Analogno velja $[\![\text{bool}]\!]\bar{\mathcal{A}} : B \leftrightarrow B$ in $(x, y) \in [\![\text{bool}]\!]\bar{\mathcal{A}}$ natanko takrat, ko $x = y$.

- Spremenljivke za tipe interpretiramo kar z relacijami, ki jim jih relacijsko okolje priredi. To pomeni, da je $[\![X]\!]\bar{\mathcal{A}} = \bar{\mathcal{A}}(X)$.

- Naj bo $\mathcal{A} : A \leftrightarrow B$ neka relacija. Definirajmo relacijo $\mathcal{A}^* : A^* \leftrightarrow B^*$ takole:

$$(\psi(a_1 :: a_2 :: \ldots :: a_n :: [\,]), \psi(b_1 :: b_2 :: \ldots :: b_m :: [\,])) \in \mathcal{A}^*$$

$$\Longleftrightarrow$$

$$m = n \ \wedge \ (a_i, b_i) \in \mathcal{A} \text{ za vsak } i \in \{1, 2, \ldots, n\}$$

  Seznama sta torej v relaciji, če sta enako dolga, in so v relaciji istoležeči elementi. Interpretacijo seznamov definiramo takole:

$$[\![T^*]\!]\bar{\mathcal{A}} = ([\![T]\!]\bar{\mathcal{A}})^*$$

- Naj bosta $\mathcal{A} : A_1 \leftrightarrow A_2$ in $\mathcal{B} : B_1 \leftrightarrow B_2$ dve relaciji. Potem definiramo relacijo $\mathcal{A} \to \mathcal{B} : (A_1 \to B_1) \leftrightarrow (A_2 \to B_2)$ na sledeč način:

$$(f_1, f_2) \in \mathcal{A} \to \mathcal{B} \Leftrightarrow ((a_1, a_2) \in \mathcal{A} \Rightarrow (\phi(f_1)\ a_1, \phi(f_2)\ a_2) \in \mathcal{B})$$

  Funkciji sta si torej v relaciji, če slikata argumente, ki so si v relaciji, v rezultate, ki so si v relaciji. Sedaj lahko definiramo:

$$[\![T_1 \to T_2]\!]\bar{\mathcal{A}} = [\![T_1]\!]\bar{\mathcal{A}} \to [\![T_2]\!]\bar{\mathcal{A}}$$

- Naj bosta F_1, F_2 neki funkciji iz $\mathbf{U}$ v $\mathbf{U}$ in naj bo $\mathcal{F}$ funkcija, ki vsaki relaciji $\mathcal{A} : A_1 \leftrightarrow A_2$ priredi relacijo $\mathcal{F}(\mathcal{A}) : F_1(A_1) \leftrightarrow F_2(A_2)$. Potem definiramo relacijo $\forall\mathcal{F} : \forall F_1 \leftrightarrow \forall F_2$ tako:

$$(g_1, g_2) \in \forall\mathcal{F} \Longleftrightarrow$$

$$\text{za vsako relacijo } \mathcal{A} : A_1 \leftrightarrow A_2 \text{ velja } (\phi(g_1)\ A_1, \phi(g_2)\ A_2) \in \mathcal{F}(\mathcal{A})$$

  Interpretacijo polimorfnega tipa definiramo takole:

$$[\![\forall X.T]\!]\bar{\mathcal{A}} = \forall(\mathcal{A} \mapsto [\![T]\!]\bar{\mathcal{A}}[\mathcal{A}/X])$$

Lema 4.1.

$$[\![T_1]\!]\bar{\mathcal{A}}[[\![T]\!]\bar{\mathcal{A}}/X] = [\![T_1[T/X]]\!]\bar{\mathcal{A}}$$

Dokaz. Dokaže se z indukcijo, podobno kot izreka 3.6 in 4.2. □

Naj bo $\bar{\mathcal{A}}$ relacijsko okolje in $\bar{A}$,$\bar{A}'$ dve okolji za tipe za kontekst Γ. Če za vsako spremenljivko za tipe X v Γ velja $\bar{\mathcal{A}}(X) : \bar{A}(X) \leftrightarrow \bar{A}'(X)$, pišemo $\bar{\mathcal{A}} : \bar{A} \leftrightarrow \bar{A}'$.

Naj bosta sedaj še $\bar{a}, \bar{a}'$ dve okolji za spremenljivke za kontekst Γ. Pravimo, da so okolja $\bar{\mathcal{A}}, \bar{A}, \bar{A}', \bar{a}, \bar{a}'$ usklajena v kontekstu Γ, če velja $\bar{\mathcal{A}} : \bar{A} \leftrightarrow \bar{A}'$ in $(\bar{a}(x), \bar{a}'(x)) \in [\![T]\!]\bar{\mathcal{A}}$ za vsak par $x : T$ v Γ.

Če je $([\![t]\!]\bar{A}\bar{a}, [\![t]\!]\bar{A}'\bar{a}') \in [\![T]\!]\bar{\mathcal{A}}$ za poljubna okolja $\bar{\mathcal{A}}, \bar{A}, \bar{A}', \bar{a}, \bar{a}'$, usklajena v Γ, potem pišemo $\Gamma \Vdash t : T$.

Izrek 4.2. *Če $\Gamma \vdash t : T$, potem je tudi $\Gamma \Vdash t : T$.*

Dokaz. Dokaz poteka z indukcijo na izpeljavo sklepa $\Gamma \vdash t : T$, enako kot dokaz izreka 3.6. Naj bodo okolja $\bar{\mathcal{A}}, \bar{A}, \bar{A}', \bar{a}, \bar{a}'$ usklajena v kontekstu Γ.

(1) Naj velja $\Gamma \vdash x : T$. To pomeni, da je $x : T$ element Γ. Po definiciji usklajenosti okolij velja
$$([\![t]\!]\bar{A}\bar{a}, [\![t]\!]\bar{A}'\bar{a}') = (\bar{a}(x), \bar{a}'(x)) \in [\![T]\!]\bar{\mathcal{A}}.$$
Torej je res $\Gamma \Vdash x : T$.

(2) Naj velja $\Gamma \vdash \text{true} : \text{bool}$. Po definiciji je $[\![\text{bool}]\!]\bar{\mathcal{A}}$ identična relacija na množici D_B. Ker je
$$[\![\text{true}]\!]\bar{A}\bar{a} = [\![\text{true}]\!]\bar{A}'\bar{a}' = \psi(\text{true}),$$
velja $\Gamma \Vdash \text{true} : \text{bool}$.

(3) Naj velja $\Gamma \vdash \text{false} : \text{bool}$. Po definiciji je $[\![\text{bool}]\!]\bar{\mathcal{A}}$ identična relacija na množici D_B. Ker je
$$[\![\text{false}]\!]\bar{A}\bar{a} = [\![\text{false}]\!]\bar{A}'\bar{a}' = \psi(\text{false}),$$
velja $\Gamma \Vdash \text{false} : \text{bool}$.

(4) Naj velja $\Gamma \vdash \text{if } t_1 \text{ then } t_2 \text{ else } t_3 : T$. To pomeni, da velja $\Gamma \vdash t_1 : \text{bool}$, $\Gamma \vdash t_2 : T$ in $\Gamma \vdash t_3 : T$. Indukcijske predpostavke so potem $\Gamma \Vdash t_1 : \text{bool}$, $\Gamma \Vdash t_2 : T$ in $\Gamma \Vdash t_3 : T$.

Iz indukcijskih predpostavk vemo, da je $([\![t_1]\!]\bar{A}\bar{a}, [\![t_1]\!]\bar{A}'\bar{a}') \in [\![\text{bool}]\!]\bar{\mathcal{A}}$, kar pomeni, da je
$$[\![t_1]\!]\bar{A}\bar{a} = [\![t_1]\!]\bar{A}'\bar{a}' \Rightarrow \phi([\![t_1]\!]\bar{A}\bar{a}) = \phi([\![t_1]\!]\bar{A}'\bar{a}') \in \{\text{true}, \text{false}\}.$$
Ločimo dva primera. Če je $\phi([\![t_1]\!]\bar{A}\bar{a}) = \phi([\![t_1]\!]\bar{A}'\bar{a}') = \text{true}$, velja
$$[\![\text{if } \ldots]\!]\bar{A}\bar{a} = [\![t_2]\!]\bar{A}\bar{a},$$
$$[\![\text{if } \ldots]\!]\bar{A}'\bar{a}' = [\![t_2]\!]\bar{A}'\bar{a}'.$$
Sicer je $\phi([\![t_1]\!]\bar{A}\bar{a}) = \phi([\![t_1]\!]\bar{A}'\bar{a}') = \text{false}$ in velja
$$[\![\text{if } \ldots]\!]\bar{A}\bar{a} = [\![t_3]\!]\bar{A}\bar{a},$$
$$[\![\text{if } \ldots]\!]\bar{A}'\bar{a}' = [\![t_3]\!]\bar{A}'\bar{a}'.$$
Po indukcijskih predpostavkah je $([\![t_2]\!]\bar{A}\bar{a}, [\![t_2]\!]\bar{A}'\bar{a}') \in [\![T]\!]\bar{\mathcal{A}}$ in tudi $([\![t_3]\!]\bar{A}\bar{a}$, $[\![t_3]\!]\bar{A}'\bar{a}') \in [\![T]\!]\bar{\mathcal{A}}$. To pomeni, da je v obeh primerih
$$([\![\text{if } \ldots]\!]\bar{A}\bar{a}, [\![\text{if } \ldots]\!]\bar{A}'\bar{a}') \in [\![T]\!]\bar{\mathcal{A}}$$

in velja $\Gamma \Vdash$ if t_1 then t_2 else $t_3 : T$.

(5) Naj velja $\Gamma \vdash 0 : \text{nat}$. Po definiciji je $[\![\text{nat}]\!]\bar{\mathcal{A}}$ identična relacija na množici D_N. Ker je

$$[\![0]\!]\bar{A}\bar{a} = [\![0]\!]\bar{A}'\bar{a}' = \psi(0),$$

velja $\Gamma \Vdash 0 : \text{nat}$.

(6) Naj velja $\Gamma \vdash \text{succ } t : \text{nat}$. To pomeni, da mora veljati $\Gamma \vdash t : \text{nat}$, indukcijska predpostavka pa je zato $\Gamma \Vdash t : \text{nat}$.

Ker je $[\![\text{nat}]\!]\bar{\mathcal{A}}$ identična relacija, je po indukcijski predpostavki $[\![t]\!]\bar{A}\bar{a} = [\![t]\!]\bar{A}'\bar{a}'$. Zato je tudi $[\![\text{succ } t]\!]\bar{A}\bar{a} = [\![\text{succ } t]\!]\bar{A}'\bar{a}'$ in velja $\Gamma \Vdash \text{succ } t : \text{nat}$.

(7) Naj velja $\Gamma \vdash \text{pred } t : \text{nat}$. To pomeni, da mora veljati $\Gamma \vdash t : \text{nat}$, indukcijska predpostavka pa je zato $\Gamma \Vdash t : \text{nat}$.

Kot v prejšnjem primeru vidimo, da je po predpostavki $[\![t]\!]\bar{A}\bar{a} = [\![t]\!]\bar{A}'\bar{a}'$ in zato $[\![\text{pred } t]\!]\bar{A}\bar{a} = [\![\text{pred } t]\!]\bar{A}'\bar{a}'$. Torej res velja $\Gamma \Vdash \text{pred } t : \text{nat}$.

(8) Naj velja $\Gamma \vdash \text{iszero } t : \text{bool}$. To pomeni, da mora veljati $\Gamma \vdash t : \text{nat}$, indukcijska predpostavka pa je $\Gamma \Vdash t : \text{nat}$.

Ponovno je po predpostavki $[\![t]\!]\bar{A}\bar{a} = [\![t]\!]\bar{A}'\bar{a}'$ in zato $[\![\text{iszero } t]\!]\bar{A}\bar{a} = [\![\text{iszero } t]\!]\bar{A}'\bar{a}'$. Torej res velja $\Gamma \Vdash \text{iszero } t : \text{nat}$.

(9) Naj velja $\Gamma \vdash [\ ]_T : T^*$. Vemo, da je $[\![[\ ]_T]\!]\bar{A}\bar{a} \in D_{([\![T]\!]\bar{A})^*}$ in $[\![[\ ]_T]\!]\bar{A}'\bar{a}' \in D_{([\![T]\!]\bar{A}')^*}$, zaradi usklajenosti okolij pa je

$$[\![T^*]\!]\bar{\mathcal{A}} : ([\![T]\!]\bar{A})^* \leftrightarrow ([\![T]\!]\bar{A}')^*.$$

Seznama sta si po definiciji v relaciji, če sta enako dolga in so si istoležeči elementi v relaciji. Zato je $([\![[\ ]_T]\!]\bar{A}\bar{a}, [\![[\ ]_T]\!]\bar{A}'\bar{a}') \in [\![T^*]\!]\bar{\mathcal{A}}$ in velja $\Gamma \Vdash [\ ]_T : T^*$.

(10) Naj velja $\Gamma \vdash t_1 :: t_2 : T^*$. To pomeni, da mora veljati $\Gamma \vdash t_1 : T$ in $\Gamma \vdash t_2 : T^*$. Indukcijski predpostavki sta tako $\Gamma \Vdash t_1 : T$ in $\Gamma \Vdash t_2 : T^*$. Iz predpostavk vemo

$$([\![t_1]\!]\bar{A}\bar{a}, [\![t_1]\!]\bar{A}'\bar{a}') \in [\![T]\!]\bar{\mathcal{A}},$$

$$([\![t_2]\!]\bar{A}\bar{a}, [\![t_2]\!]\bar{A}'\bar{a}') \in [\![T^*]\!]\bar{\mathcal{A}}.$$

Po definiciji relacije $([\![T]\!]\bar{\mathcal{A}})^* = [\![T^*]\!]\bar{\mathcal{A}}$ je zato tudi

$$([\![t_1 :: t_2]\!]\bar{A}\bar{a}, [\![t_1 :: t_2]\!]\bar{A}'\bar{a}') \in [\![T^*]\!]\bar{\mathcal{A}}$$

in tako velja $\Gamma \Vdash t_1 :: t_2 : T^*$.

(11) Naj velja $\Gamma \vdash \text{match } t_1 \text{ with } [\ ]_T \mapsto t_2 \mid x_1 :: x_2 \mapsto t_3 : T_1$. To pomeni, da mora veljati $\Gamma \vdash t_1 : T^*$, $\Gamma \vdash t_2 : T_1$ in $\Gamma, x_1 : T, x_2 : T^* \vdash t_3 : T_1$. V tem primeru so indukcijske predpostavke $\Gamma \Vdash t_1 : T^*$, $\Gamma \Vdash t_2 : T_1$ in $\Gamma, x_1 : T, x_2 : T^* \Vdash t_3 : T_1$.

Po predpostavki je $([\![t_1]\!]\bar{A}\bar{a}, [\![t_1]\!]\bar{A}'\bar{a}') \in [\![T^*]\!]\bar{\mathcal{A}}$, kar pomeni, da sta seznama $\phi([\![t_1]\!]\bar{A}\bar{a})$ in $\phi([\![t_1]\!]\bar{A}'\bar{a}')$ enako dolga. Ločimo dve možnosti. Če sta ta dva seznama prazna, potem velja

$$[\![\text{match } \ldots]\!]\bar{A}\bar{a} = [\![t_2]\!]\bar{A}\bar{a},$$

$$[\![\text{match } \ldots]\!]\bar{A}'\bar{a}' = [\![t_2]\!]\bar{A}'\bar{a}'.$$

Po predpostavki sta ta elementa v relaciji $[\![T_1]\!]\bar{\mathcal{A}}$, zato velja $\Gamma \Vdash \text{match } \ldots : T_1$.

Druga možnost je, da sta seznama oblike

$$\phi([\![t_1]\!]\bar{A}\bar{a}) = v_1 :: v_2,$$

$$\phi([\![t_1]\!]\bar{A}'\bar{a}') = v_1' :: v_2',$$

pri čemer mora po predpostavki veljati $(v_1, v_1') \in [\![T]\!]\bar{\mathcal{A}}$ in $(\psi(v_2), \psi(v_2')) \in [\![T^*]\!]\bar{\mathcal{A}}$. Zato so okolja $\bar{\mathcal{A}}$, $\bar{A}$, $\bar{A}'$, $\bar{a}[v_1/x_1, \psi(v_2)/x_2]$, $\bar{a}'[v_1'/x_1, \psi(v_2')/x_2]$ usklajena v kontekstu $\Gamma, x_1 : T, x_2 : T^*$. Po definiciji je v tem primeru

$$[\![\text{match } \ldots]\!]\bar{A}\bar{a} = [\![t_3]\!]\bar{A}\bar{a}[v_1/x_1, \psi(v_2)/x_2],$$

$$[\![\text{match } \ldots]\!]\bar{A}'\bar{a}' = [\![t_3]\!]\bar{A}'\bar{a}'[v_1'/x_1, \psi(v_2')/x_2].$$

Iz indukcijske predpostavke sledi, da sta ta dva elementa v relaciji $[\![T_1]\!]\bar{\mathcal{A}}$, in zato velja $\Gamma \Vdash \text{match } \ldots : T_1$.

(12) Naj velja $\Gamma \vdash (\lambda x : T_1.t) : T_1 \to T_2$. To pomeni, da mora veljati $\Gamma, x : T_1 \vdash t : T_2$, indukcijska predpostavka pa je $\Gamma, x : T_1 \Vdash t : T_2$. Vemo:

$$[\![\lambda x : T_1.t]\!]\bar{A}\bar{a} = \psi(a \in D_{[\![T_1]\!]\bar{A}} \mapsto [\![t]\!]\bar{A}\bar{a}[a/x]),$$

$$[\![\lambda x : T_1.t]\!]\bar{A}'\bar{a}' = \psi(a' \in D_{[\![T_1]\!]\bar{A}'} \mapsto [\![t]\!]\bar{A}'\bar{a}'[a'/x]).$$

Naj bo $(a, a') \in [\![T_1]\!]\bar{\mathcal{A}}$. Potem so okolja $\bar{\mathcal{A}}$, $\bar{A}$, $\bar{A}'$, $\bar{a}[a/x]$, $\bar{a}'[a'/x]$ usklajena v kontekstu $\Gamma, x : T_1$. Iz predpostavk sledi $([\![t]\!]\bar{A}\bar{a}[a/x], [\![t]\!]\bar{A}'\bar{a}'[a'/x]) \in [\![T_2]\!]\bar{\mathcal{A}}$, od tod pa sledi

$$([\![\lambda x : T_1.t]\!]\bar{A}\bar{a}, [\![\lambda x : T_1.t]\!]\bar{A}'\bar{a}') \in [\![T_1 \to T_2]\!]\bar{\mathcal{A}},$$

saj sta to funkciji, ki elementa v relaciji, preslikata v elementa v relaciji. Zato velja $\Gamma \Vdash (\lambda x : T_1.t) : T_1 \to T_2$.

(13) Naj velja $\Gamma \vdash t_2\ t_1 : T_2$. To pomeni, da mora veljati $\Gamma \vdash t_1 : T_1$ in $\Gamma \vdash t_2 : T_1 \to T_2$. Indukcijski predpostavki sta potem $\Gamma \Vdash t_1 : T_1$ in $\Gamma \Vdash t_2 : T_1 \to T_2$. Vemo:

$$[\![t_2\ t_1]\!]\bar{A}\bar{a} = \phi([\![t_2]\!]\bar{A}\bar{a})([\![t_1]\!]\bar{A}\bar{a}),$$

$$[\![t_2\ t_1]\!]\bar{A}'\bar{a}' = \phi([\![t_2]\!]\bar{A}'\bar{a}')([\![t_1]\!]\bar{A}'\bar{a}').$$

Po predpostavkah velja

$$([\![t_1]\!]\bar{A}\bar{a}, [\![t_1]\!]\bar{A}'\bar{a}') \in [\![T_1]\!]\bar{\mathcal{A}},$$

$$([\![t_2]\!]\bar{A}\bar{a}, [\![t_2]\!]\bar{A}'\bar{a}') \in [\![T_1 \to T_2]\!]\bar{\mathcal{A}}.$$

Po definiciji relacije $[\![T_1 \to T_2]\!]\bar{\mathcal{A}}$ to pomeni, da je

$$([\![t_2\ t_1]\!]\bar{A}\bar{a}, [\![t_2\ t_1]\!]\bar{A}'\bar{a}') \in [\![T_2]\!]\bar{\mathcal{A}}.$$

Torej velja $\Gamma \Vdash t_2\ t_1 : T_2$.

(14) Naj velja $\Gamma \vdash \Lambda X.t : \forall X.T$. To pomeni, da mora veljati $\Gamma, X : * \vdash t : T$, indukcijska predpostavka pa je tako $\Gamma, X : * \Vdash t : T$. Vemo:

$$[\![\Lambda X.t]\!]\bar{A}\bar{a} = \psi(A \in \mathbf{U} \mapsto [\![t]\!]\bar{A}[A/X]\bar{a}),$$
$$[\![\Lambda X.t]\!]\bar{A}'\bar{a}' = \psi(A' \in \mathbf{U} \mapsto [\![t]\!]\bar{A}'[A'/X]\bar{a}').$$

Naj bo $\mathcal{A} : A \leftrightarrow A'$ neka relacija. Potem so okolja $\bar{\mathcal{A}}[\mathcal{A}/X]$, $\bar{A}[A/X]$, $\bar{A}'[A'/X]$, $\bar{a}$, $\bar{a}'$ usklajena v kontekstu $\Gamma, X : *$. Po predpostavki to pomeni, da je

$$([\![t]\!]\bar{A}[A/X]\bar{a}, [\![t]\!]\bar{A}'[A'/X]\bar{a}') \in [\![T]\!]\bar{\mathcal{A}}[\mathcal{A}/X].$$

Po definiciji relacije $[\![\forall X.T]\!]\bar{\mathcal{A}}$, je potem

$$([\![\Lambda X.t]\!]\bar{A}\bar{a}, [\![\Lambda X.t]\!]\bar{A}'\bar{a}') \in [\![\forall X.T]\!]\bar{\mathcal{A}}.$$

Torej velja $\Gamma \Vdash \Lambda X.t : \forall X.T$.

(15) Naj velja $\Gamma \vdash t_T : T_1[T/X]$. To pomeni, da mora veljati $\Gamma \vdash t : \forall X.T_1$, indukcijska predpostavka pa je tako $\Gamma \Vdash t : \forall X.T_1$. Vemo:

$$[\![t_T]\!]\bar{A}\bar{a} = \phi([\![t]\!]\bar{A}\bar{a})([\![T]\!]\bar{A}),$$
$$[\![t_T]\!]\bar{A}'\bar{a}' = \phi([\![t]\!]\bar{A}'\bar{a}')([\![T]\!]\bar{A}').$$

Po predpostavki je

$$([\![t]\!]\bar{A}\bar{a}, [\![t]\!]\bar{A}'\bar{a}') \in [\![\forall X.T_1]\!]\bar{\mathcal{A}},$$

kar po definiciji te relacije pomeni, da je

$$([\![t_T]\!]\bar{A}\bar{a}, [\![t_T]\!]\bar{A}'\bar{a}') \in [\![T_1]\!]\bar{\mathcal{A}}[[\![T]\!]\bar{\mathcal{A}}/X] = [\![T_1[T/X]]\!]\bar{\mathcal{A}},$$

po lemi 4.1. Torej je res $\Gamma \Vdash t_T : T_1[T/X]$. □

5. Primeri uporabe izreka

V tem poglavju bomo pokazali, kako lahko s pomočjo izreka 4.2 iz tipa izpeljemo izrek, ki mu morajo zadoščati programi tega tipa. Najprej pa si podrobneje oglejmo interpretacijo programov.

Ker se v programu po definiciji nobena spremenljivka ne pojavi prosto, je njegov tip neodvisen od konteksta, v katerem mu ga določamo. To pomeni, da je njegova interpretacija neodvisna od okolij. Zato lahko interpretacijo programa p označimo kar z $[\![p]\!]$. Analogno velja za zaprte tipe.

Primer 5.1. Začeli bomo s tipom identitete. Dobljeni izrek sicer ne bo preveč zanimiv, je pa zato izpeljava najlažja. Naj bo torej p program tipa $\forall X.X \to X$. Po izreku velja

$$([\![p]\!], [\![p]\!]) \in [\![\forall X.X \to X]\!].$$

Po definiciji je

$$[\![\forall X.X \to X]\!] = \forall(\mathcal{A} \mapsto [\![X \to X]\!][\mathcal{A}/X]) = \forall(\mathcal{A} \mapsto (\mathcal{A} \to \mathcal{A})).$$

Ker je $[\![p]\!]$ v tej relaciji sam s sabo, to pomeni, da za vsako relacijo $\mathcal{A} : A \leftrightarrow B$, kjer sta $A, B \in \mathbf{U}$, velja

$$(\phi([\![p]\!])(A), \phi([\![p]\!])(B)) \in \mathcal{A} \to \mathcal{A}.$$

To pa po definiciji relacije $\mathcal{A} \to \mathcal{A}$ pomeni, da za vsak par $(a, b) \in \mathcal{A}$ velja

$$(\phi(\phi([\![p]\!])(A))(a), \phi(\phi([\![p]\!])(B))(b)) \in \mathcal{A}.$$

Naj bo f funkcija iz D_A v D_B. Predstavimo jo lahko z relacijo $\mathcal{F} : A \leftrightarrow B$, definirano takole:

$$(a, b) \in \mathcal{F} \Leftrightarrow f(a) = b.$$

Relaciji $\mathcal{F}$ pravimo *graf funkcije* f. Če za relacijo $\mathcal{A}$ vzamemo kar $\mathcal{F}$, dobimo:

$$f(a) = b \Rightarrow f(\phi(\phi([\![p]\!])(A))(a)) = \phi(\phi([\![p]\!])(B))(b) = \phi(\phi([\![p]\!])(B))\ (f(a)).$$

Torej je $f \circ \phi(\phi([\![p]\!])(A)) = \phi(\phi([\![p]\!])(B)) \circ f$.

Ker je takšna izpeljava nepregledna, bomo pri naslednjih primerih manj formalni. Preslikave ϕ ne bomo več pisali, za uporabo funkcij pa bomo uporabili kar zapis iz programskega jezika, torej f_T je funkcija, uporabljena na tipu, $f\ a$ pa na argumentu. Poleg tega bomo namesto $[\![p]\!]$ pisali samo p. V tem primeru bi torej dobili rezultat:

$$f \circ p_A = p_B \circ f.$$

Za funkcije tega tipa je torej vseeno, v katerem vrstnem redu jih komponiramo z f. Za identiteto ta rezultat ni presenetljiv.

Primer 5.2. Naj bo p program tipa $\forall X.X^* \to X^*$. Takšen tip ima na primer funkcija, ki vrne seznam brez prvega elementa, prazen seznam pa slika v prazen seznam. Po izreku velja

$$(p, p) \in [\![\forall X.X^* \to X^*]\!].$$

Po definiciji je

$$\begin{aligned}[\![\forall X.X^* \to X^*]\!] &= \forall(\mathcal{A} \mapsto [\![X^* \to X^*]\!][\mathcal{A}/X]) \\ &= \forall(\mathcal{A} \mapsto (\mathcal{A}^* \to \mathcal{A}^*)).\end{aligned}$$

To pomeni, da za vsako relacijo $\mathcal{A} : A \leftrightarrow B$, kjer sta $A, B \in \mathbf{U}$, velja

$$(p_A, p_B) \in \mathcal{A}^* \to \mathcal{A}^*.$$

Iz definicije relacije $\mathcal{A}^* \to \mathcal{A}^*$ sedaj sledi, da za vsak par $(a, b) \in \mathcal{A}^*$ velja

$$(p_A\ a, p_B\ b) \in \mathcal{A}^*.$$

Naj bo f funkcija iz D_A v D_B. Označimo z f^* funkcijo iz D_{A*} v D_{B*} (torej iz seznamov z elementi iz D_A v sezname z elementi iz D_B), ki na vsakem elementu seznama uporabi funkcijo f. Če z $\mathcal{F}$ označimo graf od f, potem je graf funkcije f^* ravno relacija $\mathcal{F}^*$. Če sedaj za $\mathcal{A}$ vzamemo $\mathcal{F}$, za p velja:

$$f^*\ a = b \Rightarrow f^*\ (p_A\ a) = p_B\ b = p_B\ (f^*\ a).$$

Velja torej $f^* \circ t_A = t_B \circ f^*$.

Ponovno nam izrek pove, da je vrstni red uporabe funkcij nepomemben, hitrost izvajanja programa pa je lahko različna. V našem primeru za funkcijo, ki spusti prvi

element seznama, bi v primeru $f^* \circ t_A$ funkcijo f uporabili na enem elementu manj kot v primeru $t_B \circ f^*$ (razen seveda za prazen seznam). Razlika v številu elementov pa bi lahko bila še večja (elementov bi lahko bilo tudi več).

Primer 5.3. Naj bo p program tipa $\forall X.X^* \to X^* \to X^*$. Ker je $\to$ desno asociativna, je ta tip enak $\forall X.X^* \to (X^* \to X^*)$. To je torej funkcija, ki slika seznam tipa X^* v neko funkcijo tipa $X^* \to X^*$, ki ponovno sprejme neki seznam tipa X^* in vrne neki seznam tipa X^*. Na p lahko torej gledamo kot na funkcijo dveh argumentov: sprejme dva seznama tipa X^* in vrne neki seznam tipa X^*. Tak tip ima na primer funkcija, ki združi dva seznama. Po izreku velja

$$(p, p) \in [\![\forall X.X^* \to X^* \to X^*]\!].$$

Po definiciji je

$$\begin{aligned}[\![\forall X.X^* \to X^* \to X^*]\!] &= \forall(\mathcal{A} \mapsto [\![X^* \to X^* \to X^*]\!][\mathcal{A}/X]) \\ &= \forall(\mathcal{A} \mapsto (\mathcal{A}^* \to \mathcal{A}^* \to \mathcal{A}^*)).\end{aligned}$$

Za vsako relacijo $\mathcal{A} : A \leftrightarrow B$ torej velja

$$(p_A, p_B) \in \mathcal{A}^* \to \mathcal{A}^* \to \mathcal{A}^*.$$

Vzemimo za $\mathcal{A}$ graf funkcije $f : D_A \to D_B$. Potem dobimo

$$\begin{aligned}f^*\ a = b &\Rightarrow (p_A\ a, p_B\ b) \in \mathcal{A}^* \to \mathcal{A}^* \\ f^*\ a = b, f^*\ c = d &\Rightarrow f^*(p_A\ a\ c) = p_B\ b\ d\end{aligned}$$

oziroma $f^*(p_A\ a\ c) = p_B\ (f^*\ a)\ (f^*\ c)$. Za f je torej ponovno vseeno, ali jo uporabimo na argumentih ali na rezultatu funkcije p.

Primer 5.4. Naj bo p program tipa $\forall X.(X \to \text{bool}) \to X^* \to X^*$. To je funkcija dveh argumentov, ki sprejme funkcijo tipa $X \to \text{bool}$ in seznam tipa X^* ter vrne seznam tipa X^*. Primer bi bila funkcija filter, ki iz začetnega seznama vzame samo elemente, ki se s podano funkcijo slikajo v true. Po izreku velja

$$(p, p) \in [\![\forall X.(X \to \text{bool}) \to X^* \to X^*]\!].$$

Po definiciji je

$$\begin{aligned}[\![\forall X.(X \to \text{bool}) \to X^* \to X^*]\!] &= \forall(\mathcal{A} \mapsto [\![(X \to \text{bool}) \to X^* \to X^*]\!][\mathcal{A}/X]) \\ &= \forall(\mathcal{A} \mapsto ((\mathcal{A} \to \text{bool}) \to \mathcal{A}^* \to \mathcal{A}^*)).\end{aligned}$$

Za vsako relacijo $\mathcal{A} : A \leftrightarrow B$ mora veljati

$$(p_A, p_B) \in (\mathcal{A} \to \text{bool}) \to \mathcal{A}^* \to \mathcal{A}^*.$$

Naj za funkciji z, z' velja $(z, z') \in \mathcal{A} \to \text{bool}$. To pomeni, da za vsak par $(a, a') \in \mathcal{A}$ velja $z\ a = z'\ a'$. Za program p potem velja

$$(p_A\ z, p_B\ z') \in \mathcal{A}^* \to \mathcal{A}^*.$$

Vzemimo za relacijo $\mathcal{A}$ graf funkcije $f : D_A \to D_B$. To pomeni, da je $z = z' \circ f$ in velja

$$f^*\ a = b \Rightarrow f^*(p_A\ z\ a) = p_B\ z'\ b = p_B\ z'\ (f^*\ a).$$

Torej velja

$$f^* \circ p_A\ z = f^* \circ p_A(z' \circ f) = p_B\ z' \circ f^*.$$

Za funkcijo filter to pomeni, da je vseeno, ali elemente seznama izbira s funkcijo $(z' \circ f)$ in potem izbrane elemente preslika z f ali pa najprej vse elemente seznama preslika z f in jih potem izbira s funkcijo z'. Pri drugem načinu je očitno manj dela.

Primer 5.5. Naj bo p program tipa $\forall X.\forall Y.(X \to Y \to Y) \to Y \to X^* \to Y$. Po izreku mora veljati

$$(p, p) \in [\![\forall X.\forall Y.(X \to Y \to Y) \to Y \to X^* \to Y]\!].$$

Poglejmo si interpretacijo tega tipa.

$$\begin{aligned}
&[\![\forall X.\forall Y.(X \to Y \to Y) \to Y \to X^* \to Y]\!] \\
&\qquad = \forall(\mathcal{A} \mapsto [\![\forall Y.(X \to Y \to Y) \to Y \to X^* \to Y]\!][\mathcal{A}/X]) \\
&\qquad = \forall(\mathcal{A} \mapsto [\![\forall Y.(\mathcal{A} \to Y \to Y) \to Y \to \mathcal{A}^* \to Y]\!]) \\
&\qquad = \forall(\mathcal{A} \mapsto (\forall(\mathcal{B} \mapsto [\![(\mathcal{A} \to Y \to Y) \to Y \to \mathcal{A}^* \to Y]\!][\mathcal{B}/Y])) \\
&\qquad = \forall(\mathcal{A} \mapsto (\forall(\mathcal{B} \mapsto (\mathcal{A} \to \mathcal{B} \to \mathcal{B}) \to \mathcal{B} \to \mathcal{A}^* \to \mathcal{B}))
\end{aligned}$$

Za poljubni relaciji $\mathcal{A} : A \to A'$ in $\mathcal{B} : B \to B'$ potem velja

$$(p_{AB}, p_{A'B'}) \in (\mathcal{A} \to \mathcal{B} \to \mathcal{B}) \to \mathcal{B} \to \mathcal{A}^* \to \mathcal{B}.$$

Če sedaj upoštevamo definicije relacij, dobimo:

$$\begin{aligned}
&\text{za vse } (z, z') \in \mathcal{A} \to \mathcal{B} \to \mathcal{B} \\
&\qquad \text{za vse } (u, u') \in \mathcal{B} \\
&\qquad\qquad \text{velja } (p_{AB}\ z\ u, p_{A'B'}\ z'\ u') \in \mathcal{A}^* \to \mathcal{B}.
\end{aligned}$$

Vzemimo sedaj za relacijo $\mathcal{A}$ graf funkcije $f : D_A \to D_{A'}$, za relacijo $\mathcal{B}$ pa graf funkcije $g : D_B \to D_{B'}$. Predpostavka $(z, z') \in \mathcal{A} \to \mathcal{B} \to \mathcal{B}$ je potem ekvivalentna pogoju, da za poljubna $a \in D_A$ in $b \in D_B$ velja

$$f\ a = a' \wedge g\ b = b' \Rightarrow g(z\ a\ b) = z'\ a'\ b' = z'(f\ a)(g\ b).$$

Predpostavka $(u, u') \in \mathcal{B}$ pa je ekvivalentna zahtevi $g\ u = u'$. Za program p potem velja:

$$\begin{gathered}
\text{če za poljubna } a \in D_A, b \in D_B \text{ velja } g(z\ a\ b) = z'(f\ a)(g\ b) \text{ in } g\ u = u \\
\text{potem je } g \circ p_{AB}\ z\ u = p_{A'B'}\ z'\ u' \circ f^*.
\end{gathered}$$

Tip $\forall X.\forall Y.(X \to Y \to Y) \to Y \to X^* \to Y$ ima na primer funkcija fold, ki sprejme neko funkcijo tipa $X \to Y \to Y$, začetni element tipa Y in seznam

elementov tipa X. Podano funkcijo najprej uporabi na podanem elementu tipa Y in prvem elementu seznama, nato pa jo uporabi na dobljenem rezultatu tipa Y in drugem elementu seznama, itd.

Poglejmo, kaj nam za to funkcijo pove izrek. Kompozitum $g \circ \mathrm{fold}_{AB}\ z\ u$ je funkcija, ki elemente podanega seznama zaporedoma združuje s funkcijo z, začne pa z elementom u. Na koncu dobljeni rezultat še preslika s funkcijo g. Kompozitum $p_{A'B'}\ z'\ u' \circ f^*$ pa elemente podanega seznama najprej preslika s funkcijo f, potem pa jih združuje s pomočjo funkcije z', začne pa z elementom u'. Izrek pove, da je za primerno izbrani funkciji z in z' ter začetna elementa u, u' rezultat enak.

Literatura

[1] A. Bauer, *Teorija programskih jezikov*, zapiski s predavanj, verzija december 2007, dostopno na `http://andrej.com/tpj/tpj.pdf`.

[2] B. C. Pierce, *Types and programming languages*, MIT Press, Cambridge, 2002.

[3] A. M. Pitts, *Polymorphism is set theoretic, constructively*, v: Category Theory and Computer Science, Edinburgh, September 7-9, 1987 : proceedings (ur. D. H. Pitt, A. Poign D. E. Rydeheard), Lecture notes in computer science **283**, Springer, Berlin, 1987, str. 12–39.

[4] P. Wadler, *Theorems for free!*, v: 4th International Symposium on Functional Programming Languages and Computer Architecture, Imperial College, London, September 1989, ACM Press, New York, 1989, str. 347–359.