Simply-typed λ-calculus exercises

Adrián Enríquez Ballester

January 9, 2022

Exercise 1

Give a proof in TA_λ for the following type assignment for function composition:

$$\vdash \lambda f.\lambda g.\lambda x.f\ (g\ x) : (b \to c) \to (a \to b) \to (a \to c)$$

Its proof tree is the following:

$$\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{}{f : b \to c \vdash f : b \to c}\,\mathrm{VAR} \quad \dfrac{\dfrac{}{g : a \to b \vdash g : a \to b}\,\mathrm{VAR} \quad \dfrac{}{x : a \vdash x : a}\,\mathrm{VAR}}{g : a \to b, x : a \vdash g\ x : b}\,\mathrm{APP}}{f : b \to c, g : a \to b, x : a \vdash f\ (g\ x) : c}\,\mathrm{APP}}{f : b \to c, g : a \to b \vdash \lambda x.f\ (g\ x) : a \to c}\,\mathrm{ABS}}{f : b \to c \vdash \lambda g.\lambda x.f\ (g\ x) : (a \to b) \to (a \to c)}\,\mathrm{ABS}}{\vdash \lambda f.\lambda g.\lambda x.f\ (g\ x) : (b \to c) \to (a \to b) \to (a \to c)}\,\mathrm{ABS}$$

Exercise 2

Provide a type for the combinator $\mathsf{S} = \lambda x.\lambda y.\lambda z.(x\ z)\ (y\ z)$.

Its type can be

$$(a \to b \to c) \to (a \to b) \to (a \to c)$$

and the corresponding proof tree is the following:

$$\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{}{x : a \to b \to c \vdash x : a \to b \to c}\,\mathrm{VAR} \quad \dfrac{}{z : a \vdash z : a}\,\mathrm{VAR}}{x : a \to b \to c, z : a \vdash x\ z : b \to c}\,\mathrm{APP} \quad \dfrac{\dfrac{}{y : a \to b \vdash y : a \to b}\,\mathrm{VAR} \quad \dfrac{}{z : a \vdash z : a}\,\mathrm{VAR}}{y : a \to b, z : a \vdash y\ z : b}\,\mathrm{APP}}{x : a \to b \to c, y : a \to b, z : a \vdash (x\ z)\ (y\ z) : c}\,\mathrm{APP}}{x : a \to b \to c, y : a \to b \vdash \lambda z.(x\ z)\ (y\ z) : a \to c}\,\mathrm{ABS}}{x : a \to b \to c \vdash \lambda y.\lambda z.(x\ z)\ (y\ z) : (a \to b) \to (a \to c)}\,\mathrm{ABS}}{\vdash \lambda x.\lambda y.\lambda z.(x\ z)\ (y\ z) : (a \to b \to c) \to (a \to b) \to (a \to c)}\,\mathrm{ABS}$$

Exercise 3

Provide a type for the term $\mathsf{II} = (\lambda x.x)\ (\lambda x.x)$.

Its type can be

$$a \to a$$

and the corresponding proof tree is the following:

$$\dfrac{\dfrac{\dfrac{}{x : a \to a \vdash x : a \to a}\,\text{VAR}}{\vdash \lambda x.x : (a \to a) \to (a \to a)}\,\text{ABS} \qquad \dfrac{\dfrac{}{x : a \vdash x : a}\,\text{VAR}}{\vdash \lambda x.x : a \to a}\,\text{ABS}}{\vdash (\lambda x.x)\ (\lambda x.x) : a \to a}\,\text{APP}$$

Exercise 4

Provide types for the terms in exercise 5 of the previous sheet (Church booleans) along with their corresponding derivation.

TRUE can be assigned the type $a \to b \to a$, and FALSE can be assigned the type $a \to b \to b$. Their respective proof trees are the following, from left to right:

$$\dfrac{\dfrac{\dfrac{}{x : a, y : b \vdash x : a}\,\text{VAR}}{x : a \vdash \lambda y.x : b \to a}\,\text{ABS}}{\vdash \lambda x.\lambda y.x : a \to b \to a}\,\text{ABS} \qquad \dfrac{\dfrac{\dfrac{}{x : a, y : b \vdash y : b}\,\text{VAR}}{x : a \vdash \lambda y.y : b \to b}\,\text{ABS}}{\vdash \lambda x.\lambda y.y : a \to b \to b}\,\text{ABS}$$

The term NOT can be assigned the type $(a \to b \to c) \to b \to a \to c$, as proved in the following derivation tree:

$$\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{}{b : a \to b \to c \vdash b : a \to b \to c}\,\text{VAR} \quad \dfrac{}{y : a \vdash y : a}\,\text{VAR}}{b : a \to b \to c, y : a \vdash b\ y : b \to c}\,\text{APP} \quad \dfrac{}{x : b \vdash x : b}\,\text{VAR}}{b : a \to b \to c, x : b, y : a \vdash b\ y\ x : c}\,\text{APP}}{b : a \to b \to c, x : b \vdash \lambda y.b\ y\ x : a \to c}\,\text{ABS}}{b : a \to b \to c \vdash \lambda x.\lambda y.b\ y\ x : b \to a \to c}\,\text{ABS}}{\vdash \lambda b.\lambda x.\lambda y.b\ y\ x : (a \to b \to c) \to b \to a \to c}\,\text{ABS}$$

The term CONJ can be assigned the type

$$(a \to b \to c) \to (d \to b \to a) \to d \to b \to c$$

as proved in the following derivation tree:

$$\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{}{b : a \to b \to c \vdash b : a \to b \to c}\,\text{VAR} \quad \Delta_1}{b : a \to b \to c, c : d \to b \to a, x : d, y : b \vdash b\ (c\ x\ y) : b \to c}\,\text{APP} \quad \dfrac{}{y : b \vdash y : b}\,\text{VAR}}{b : a \to b \to c, c : d \to b \to a, x : d, y : b \vdash b\ (c\ x\ y)\ y : c}\,\text{APP}}{b : a \to b \to c, c : d \to b \to a, x : d \vdash \lambda y.b\ (c\ x\ y)\ y : b \to c}\,\text{ABS}}{b : a \to b \to c, c : d \to b \to a \vdash \lambda x.\lambda y.b\ (c\ x\ y)\ y : d \to b \to c}\,\text{ABS}}{b : a \to b \to c \vdash \lambda c.\lambda x.\lambda y.b\ (c\ x\ y)\ y : (d \to b \to a) \to d \to b \to c}\,\text{ABS}}{\vdash \lambda b.\lambda c.\lambda x.\lambda y.b\ (c\ x\ y)\ y : (a \to b \to c) \to (d \to b \to a) \to d \to b \to c}\,\text{ABS}$$

where the subtree Δ_1 is the following:

$$\dfrac{\dfrac{\dfrac{}{c : d \to b \to a \vdash c : d \to b \to a}\,\text{VAR} \quad \dfrac{}{x : d \vdash x : d}\,\text{VAR}}{c : d \to b \to a, x : d \vdash c\ x : b \to a}\,\text{APP} \quad \dfrac{}{y : b \vdash y : b}\,\text{VAR}}{c : d \to b \to a, x : d, y : b \vdash c\ x\ y : a}\,\text{APP}$$

The term DISJ can be assigned the type

$$(a \to b \to c) \to (a \to d \to b) \to a \to d \to c$$

as proved in the following derivation tree:

$$\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{}{c : a \to d \to b \vdash c : a \to d \to b}\,\text{VAR} \quad \dfrac{}{x : a \vdash x : a}\,\text{VAR}}{c : a \to d \to b, x : a \vdash c\ x : d \to b}\,\text{APP} \quad \dfrac{}{y : d \vdash y : d}\,\text{VAR}}{c : a \to d \to b, x : a, y : d \vdash c\ x\ y : b}\,\text{APP}}{b : a \to b \to c, c : a \to d \to b, x : a, y : d \vdash b\ x\ (c\ x\ y) : c}\,\text{APP}}{b : a \to b \to c, c : a \to d \to b, x : a \vdash \lambda y.b\ x\ (c\ x\ y) : d \to c}\,\text{ABS}}{b : a \to b \to c, c : a \to d \to b \vdash \lambda x.\lambda y.b\ x\ (c\ x\ y) : a \to d \to c}\,\text{ABS}}{b : a \to b \to c \vdash \lambda c.\lambda x.\lambda y.b\ x\ (c\ x\ y) : (a \to d \to b) \to a \to d \to c}\,\text{ABS}}{\vdash \lambda b.\lambda c.\lambda x.\lambda y.b\ x\ (c\ x\ y) : (a \to b \to c) \to (a \to d \to b) \to a \to d \to c}\,\text{ABS}$$

where the subtree Δ_2 is the following:

$$\dfrac{\dfrac{}{b : a \to b \to c \vdash b : a \to b \to c}\,\text{VAR} \quad \dfrac{}{x : a \vdash x : a}\,\text{VAR}}{b : a \to b \to c, x : a \vdash b\ x : b \to c}\,\text{APP}$$

Exercise 5

Prove that if $\Gamma \vdash M : \sigma$ then $\Gamma[a \mapsto \tau] \vdash M : \sigma[a \mapsto \tau]$.

Recall the definition of a type-variable substitution, simplified for one substitution at a time, as

$$\begin{aligned} a[a \mapsto \tau] &= \tau \\ b[a \mapsto \tau] &= b \\ (\tau_1 \to \tau_2)[a \mapsto \tau] &= \tau_1[a \mapsto \tau] \to \tau_2[a \mapsto \tau] \end{aligned}$$

where a and b are distinct type variables, and τ, τ_1 and τ_2 are arbitrary types.

This definition is also extended to type contexts as

$$\Gamma[a \mapsto \tau] = x_1 : \tau_1[a \mapsto \tau], x_2 : \tau_2[a \mapsto \tau], ..., x_n : \tau_n[a \mapsto \tau]$$

where a is a type variable, τ is an arbitrary type, and $\Gamma = x_1 : \tau_1, x_2 : \tau_2, ..., x_n : \tau_n$ is a context.

Note also that if a context Γ is consistent, then also is $\Gamma[a \mapsto \tau]$ for any type variable a and type τ. This is because if a variable x were assigned two different types τ_1 and τ_2 in $\Gamma[a \mapsto \tau]$ and a single type τ in Γ, then $\tau[a \mapsto \tau] \neq \tau[a \mapsto \tau]$, which is not possible.

Every type judgement must follow from a proof involving only the rules of the deductive system (i.e. VAR, ABS and APP). We are going to prove the statement by induction over the proof tree of the type judgement. As the case distinction is made over type judgements of different shape, the context, types and variables used in each case do not refer exactly to the ones used in the general statement of the exercise.

If the bottommost rule of the proof tree is VAR, we have a judgement with shape $\Gamma, x : \sigma \vdash x : \sigma$. Trivially, $(\Gamma, x : \sigma)[a \mapsto \tau] \vdash x : \sigma[a \mapsto \tau]$ also follows due to the VAR axiom because $(\Gamma, x : \sigma)[a \mapsto \tau] = \Gamma[a \mapsto \tau], x : \sigma[a \mapsto \tau]$ and type-variable substitution preserves the context consistence.

Proceeding with the first of the inductive cases, if the bottommost rule of the proof tree is ABS, we have a judgement with shape $\Gamma \vdash \lambda x.M : \sigma \to \sigma'$. The ABS rule requires that $\Gamma, x : \sigma \vdash M : \sigma'$. By rule induction hypothesis on the proof tree of this premise, we also have

$$\begin{aligned} (\Gamma, x : \sigma)[a \mapsto \tau] &\vdash M : \sigma'[a \mapsto \tau] \\ \Gamma[a \mapsto \tau], x : \sigma[a \mapsto \tau] &\vdash M : \sigma'[a \mapsto \tau] \end{aligned}$$

where the consistence of $(\Gamma, x : \sigma)[a \mapsto \tau]$ has been preserved with the type-variable substitution. This satisfies also the ABS rule yielding to the conclusion that we were looking for:

$$\Gamma[a \mapsto \tau] \vdash \lambda x.M : \sigma[a \mapsto \tau] \to \sigma'[a \mapsto \tau]$$
$$\Gamma[a \mapsto \tau] \vdash \lambda x.M : (\sigma \to \sigma')[a \mapsto \tau]$$

Finally, for the last of the inductive cases, if the bottommost rule of the proof tree is APP, we have a judgement with shape $\Gamma \cup \Gamma' \vdash M\ M' : \sigma'$. The APP rule requires that $\Gamma \vdash M : \sigma \to \sigma'$ and $\Gamma' \vdash M' : \sigma$.

By rule induction hypothesis on the proof tree of both premises, we also have

$$\Gamma[a \mapsto \tau] \vdash M : (\sigma \to \sigma')[a \mapsto \tau]$$
$$\Gamma[a \mapsto \tau] \vdash M : \sigma[a \mapsto \tau] \to \sigma'[a \mapsto \tau]$$

and

$$\Gamma'[a \mapsto \tau] \vdash M' : \sigma[a \mapsto \tau]$$

These last statements satisfy the APP rule yielding to the conclusion that we were looking for:

$$\Gamma[a \mapsto \tau] \cup \Gamma'[a \mapsto \tau] \vdash M\ M' : \sigma'[a \mapsto \tau]$$
$$(\Gamma \cup \Gamma')[a \mapsto \tau] \vdash M\ M' : \sigma'[a \mapsto \tau]$$

Note again that the consistence of $\Gamma \cup \Gamma'$ has been preserved with the type-variable substitution.

Exercise 6

Prove that if $\Gamma, x : \sigma \vdash M : \tau$ and $\Gamma \vdash N : \sigma$ then $\Gamma \vdash M[x \mapsto N] : \tau$.

This time, we are going to prove the statement by structural induction on the lambda term where the substitution is taking place (i.e. M from the statement). Again, as the case distinction is made over lambda terms of different shape, the context, types and variables used in each case do not refer exactly to the ones used in the general statement of the exercise.

If the lambda term is a variable, we have two possible cases, according to the definition of capture-avoiding substitution, depending on whether the variable being replaced is the same as the term or not:

- If the term is a variable x, its stated type judgement has shape $\Gamma, x : \sigma \vdash x : \sigma$ due to the VAR rule and, as $x[x \mapsto N] = N$, the property holds for any N such that $\Gamma \vdash N : \sigma$.
- If the term is a variable y with $x \neq y$, its stated type judgement has shape $\Gamma, y : \tau, x : \sigma \vdash y : \tau$ due to the VAR rule and, as $y[x \mapsto N] = y$, and $\Gamma, y : \tau \vdash y : \tau$ due to the VAR rule, the property also holds.

Proceeding with one of the inductive cases, if the lambda term is an application, its stated type judgement has shape $\Gamma \cup \Gamma', x : \sigma \vdash M\ M' : \tau'$, that is the same as $(\Gamma, x : \sigma) \cup (\Gamma', x : \sigma) \vdash M\ M' : \tau'$ where its APP rule requires that $\Gamma, x : \sigma \vdash M : \tau \to \tau'$ and $\Gamma', x : \sigma \vdash M' : \tau$.

By structural induction on the respective subterms M and M' for these last type judgements, if $\Gamma \vdash N : \sigma$ then we also have

$$\Gamma \vdash M[x \mapsto N] : \tau \to \tau'$$
$$\Gamma' \vdash M'[x \mapsto N] : \tau$$

By applying the APP rule to these last premises, we reach the conclusion that we were looking for:

$$\Gamma \cup \Gamma' \vdash M[x \mapsto N]\ M'[x \mapsto N] : \tau'$$
$$\Gamma \cup \Gamma' \vdash (M\ M')[x \mapsto N] : \tau'$$

Finally, if the lambda term is an abstraction, we have again several cases to consider according to the definition of capture-avoiding substitution:

- Consider that the statement has shape $\Gamma, x : \sigma \vdash \lambda x.M : \tau \to \tau'$ (i.e. we are trying to replace the variable that is bound by an abstraction).

  On the one hand, its ABS rule premise requires $\Gamma, x : \tau \vdash M : \tau'$, and it means that $\Gamma \vdash \lambda x.M : \tau \to \tau'$.

  On the other hand, $(\lambda x.M)[x \mapsto N] = \lambda x.M$, thus

  $$\Gamma \vdash (\lambda x.M)[x \mapsto N] : \tau \to \tau'$$

  and the property is satisfied.

- Consider now the case where it has shape $\Gamma, x : \sigma \vdash \lambda y.M : \tau \to \tau'$ with $x \notin FV(M)$.

  On the one hand, as x does not appear free in M, it also does not appear free in $\lambda y.M$ and the type judgment will still true even without the type assignment for x in the context [1]:

  $$\Gamma \vdash \lambda y.M : \tau \to \tau'$$

  On the other hand, $(\lambda y.M)[x \mapsto N] = \lambda y.M$, thus

  $$\Gamma \vdash (\lambda y.M)[x \mapsto N] : \tau \to \tau'$$

  and the property is satisfied.

[1] This claim, although simple, may require its own proof.

- Consider now the case where it has shape $\Gamma, x : \sigma \vdash \lambda y.M : \tau \to \tau'$ with $x \in FV(M)$ and $y \notin FV(N)$ for a term N such that $\Gamma \vdash N : \sigma$.

  On the one hand, its ABS rule premise requires $\Gamma, x : \sigma, y : \tau \vdash M : \tau'$ and, by rule induction hypothesis on the subterm M for this last type judgement, we also have

  $$\Gamma, y : \tau \vdash M[x \mapsto N] : \tau'$$

  This satisfies the ABS rule leading to

  $$\Gamma \vdash \lambda y.M[x \mapsto N] : \tau \to \tau'$$

  On the other hand, in this case $(\lambda y.M)[x \mapsto N] = \lambda y.M[x \mapsto N]$, thus

  $$\Gamma \vdash (\lambda y.M)[x \mapsto N] : \tau \to \tau'$$

  and the property is satisfied.

- Finally, consider that it has shape $\Gamma, x : \sigma \vdash \lambda y.M : \tau \to \tau'$ with $x \in FV(M)$ and $y \in FV(N)$ for a term N such that $\Gamma \vdash N : \sigma$.

  On the one hand, its ABS rule premise requires $\Gamma, x : \sigma, y : \tau \vdash M : \tau'$.

  On the one hand, in this case $(\lambda y.M)[x \mapsto N] = \lambda z.M[y \mapsto z][x \mapsto N]$ where $z \notin FV(M) \cup FV(N)$.

  As z does not occur free neither in M nor N, we can assign it type τ by keeping the consistency of the respective contexts obtaining the following type judgements [2]:

  $$\begin{gathered} \Gamma, z : \tau \vdash z : \tau \\ \Gamma, x : \sigma, y : \tau, z : \tau \vdash M : \tau' \\ \Gamma, z : \tau \vdash N : \sigma \end{gathered}$$

  By applying the induction hypothesis on the subterm M for the first two judgements above, we have

  $$\Gamma, x : \sigma, z : \tau \vdash M[y \mapsto z] : \tau'$$

  and, by applying it again over $M[y \mapsto z]$ for this last judgement and the third one, we also have

  $$\Gamma, z : \tau \vdash M[y \mapsto z][x \mapsto N] : \tau'$$

[2] Again, this claim, although simple, may require its own proof.

These reached type judgement satisfy the ABS rule leading to

$$\Gamma \vdash \lambda z.M[y \mapsto z][x \mapsto N] : \tau \to \tau'$$
$$\Gamma \vdash (\lambda y.M)[x \mapsto N] : \tau \to \tau'$$

and therefore the property is satisfied.

Exercise 7

Prove that if $M \rightsquigarrow_\beta^* M'$ and $\Gamma \vdash M : \sigma$ then $\Gamma \vdash M' : \sigma$. Informally, this states that β-reduction preserves types.

We are going to prove the statement by induction on the length of the beta reduction chain.

On the one hand, if its length is 0 (i.e. $M \rightsquigarrow_\beta^0 M'$), as no β-reduction has been performed, $M = M'$ and it is trivial that if $\Gamma \vdash M : \sigma$ then $\Gamma \vdash M : \sigma$.

On the other hand, if the chain has length k then

$$M \rightsquigarrow_\beta^{k-1} M'' \rightsquigarrow_\beta M'$$

for some lambda term M''. By induction hypothesis, if $\Gamma \vdash M : \sigma$ then $\Gamma \vdash M'' : \sigma$.

For this last β-reduction to be possible, M'' has to be of the shape $(\lambda x.X)\ N$ with $M' = X[x \mapsto N]$ and, due to its type judgement stated above, it has a proof tree as follows:

$$\text{APP}\ \frac{\text{ABS}\ \dfrac{\dfrac{\Delta_1}{\Gamma_1, x : \sigma' \vdash X : \sigma}}{\Gamma_1 \vdash \lambda x.X : \sigma' \to \sigma} \qquad \dfrac{\Delta_2}{\Gamma_2 \vdash N : \sigma'}}{\Gamma_1 \cup \Gamma_2 \vdash (\lambda x.X)\ N : \sigma}$$

The bottomost few rules allow us to obtain the judgements $\Gamma_1, x : \sigma' \vdash X : \sigma$ and $\Gamma_2 \vdash N : \sigma'$ independently of the proof trees Δ_1 and Δ_2 and, as $\Gamma_1 \cup \Gamma_2$ must be consistent, it also implies that the type judgements

$$\Gamma_1 \cup \Gamma_2, x : \sigma' \vdash X : \sigma$$
$$\Gamma_1 \cup \Gamma_2 \vdash N : \sigma'$$

also hold[3]. Therefore, by the Exercise 6, $\Gamma_1 \cup \Gamma_2 \vdash X[x \mapsto N] : \sigma$, which is precisely M, the resulting term of β-reducing M''.

[3]If $x : \sigma'$ were not consistent with Γ_2, we could have performed an α-conversion to the abstraction $(\lambda x.X)$ in order to choose a bound variable which does not lead to this problem.