Types and λ-calculus

Problem Sheet 6

* 1. Which of the following pairs of type constraints have solutions? Give the most general unifier for those that do.

(a) $a \stackrel{?}{=} b \to c$ and $b \to b \stackrel{?}{=} c$

(b) $a \to a \stackrel{?}{=} b \to c$ and $a \stackrel{?}{=} c \to c$

(c) $a \to b \stackrel{?}{=} b \to c \to d$ and $c \stackrel{?}{=} d$

Solution

Recall that most general unifiers are only unique up-to variable renaming.

(a) $[b \to b \to b/a,\, b \to b/c]$

(b) No solutions since the equations imply $a \stackrel{?}{=} a \to a$

(c) $[d \to d/a,\, d \to d/b,\, d/c]$

** 2.

(a) Compute the set of type constraints that characterises valid types of closed term $\lambda x y.\, x$ using the constraint generation algorithm.

(b) Solve the constraints obtained in the previous part using the unification algorithm and exhibit the most general unifier.

(c) Give the principal type of this term.

Solution

(a) For questions of this form, I leave it up to you exactly how you want to execute the computation. I prefer to write out a type derivation and label it with type variables and then write down the constraints (you will still need to consult the definition of CGen to know which constraints are associated with which typing rules). You may prefer to carry out the recursive computation of CGen to the letter. However you proceed you should obtain a set of constraints that is the same as the following, modulo your choice of variable names:

$$\begin{array}{rcl} a_1 & \stackrel{?}{=} & a_2 \to a_3 \\ a_2 & \stackrel{?}{=} & a_5 \\ a_3 & \stackrel{?}{=} & a_4 \to a_5 \end{array}$$

(b) Solved forms are unique up to the choice of variable names because most general unifiers are unique up to renaming. Again, I don't mind how you reach the following conclusion (and you need not show me, though it will be harder to give you partial credit in some cases):

$$\left[\begin{array}{rcl} a_5 \to a_4 \to a_5 & / & a_1 \\ a_5 & / & a_2 \\ a_4 \to a_5 & / & a_3 \end{array}\right]$$

(c) The type variable associated with the root of the derivation in this case is a_1, so the principal type is:

$$a_5 \to a_4 \to a_5$$

** 3. Give an alternative proof that $\lambda x.\, x x$ is untypable by computing the set of type constraints using Hindley Milner type inference and showing, using unification, that these constraints are unsolvable.

Read the proof of the same result in Lemma 8.1 and reflect on the similarities.

Solution

Constraints:

$$\begin{array}{rcl} a_1 & \stackrel{?}{=} & a_3 \\ a_1 & \stackrel{?}{=} & a_4 \\ a_3 & \stackrel{?}{=} & a_4 \to a_2 \\ a_0 & \stackrel{?}{=} & a_1 \to a_2 \end{array}$$

By applying rule (4) to the first constraint and then rule 4 to the second constraint, we obtain:

$$\begin{aligned} a_1 &\stackrel{?}{=} a_4 \\ a_3 &\stackrel{?}{=} a_4 \\ a_4 &\stackrel{?}{=} a_4 \to a_2 \\ a_0 &\stackrel{?}{=} a_4 \to a_2 \end{aligned}$$

No more rules are applicable to this constraint set (in particular, rule 4 is not applicable since none of the variables on the left occur on the right except for a_4 which occurs on both sides of the third constraint). The constraint set is not in solved form, so the original constraints are unsolvable. Hence, there is no type that can be given to $\lambda x.\, xx$.

** 4. Suppose closed term M has a normal form $\lambda x.\, xx$. Prove that all reducts of M are untypable, i.e. $M \twoheadrightarrow_\beta N$ implies N untypable.

Solution

Assume M has normal form $\lambda x.\, xx$ and suppose $M \twoheadrightarrow_\beta N$. Then we have $M \twoheadrightarrow_\beta \lambda x.\, xx$ and $M \twoheadrightarrow_\beta N$ so, by Confluence, N and $\lambda x.\, xx$ have a common reduct. But $\lambda x.\, xx$ is a normal form, so we must have that $N \twoheadrightarrow_\beta \lambda x.\, xx$. We claim that N is therefore untypable. To see why, suppose that N were typable, then by Subject Reduction, so is $\lambda x.\, xx$. However, we know that this is impossible.

*** 5. The following property is a variation of subject reduction:

if $M =_\beta N$ and $\Gamma \vdash M : A$ then $\Gamma \vdash N : A$

Is this property true for our type system? Either prove it or give a counterexample.

Solution

It is not true in our system. To see why, take e.g. $\lambda y.\, y =_\beta (\lambda x.\, (\lambda y.\, y))(\lambda x.\, xx)$, which holds because the RHS reduces to the LHS in one step. We have $\vdash \lambda y.\, y : a \to a$ so the hypotheses of the implication are satisfied, but $(\lambda x.\, (\lambda y.\, y))(\lambda x.\, xx)$ is untypable because it includes $\lambda x.\, xx$ as a subterm.