

## 2001 Paper 7 Question 13

### Types

- (a) What does it mean to say that an ML type scheme  $\sigma$  *generalises* an ML type  $\tau$ ? Writing  $\sigma \succ \tau$  for this relation and defining

$$\begin{aligned}\sigma_1 &= \forall\{\alpha, \beta\}(\alpha \rightarrow \beta) & \sigma_2 &= \forall\{\alpha\}(\alpha \rightarrow \beta) \\ \tau_1 &= (\alpha \rightarrow \beta) \rightarrow \alpha & \tau_2 &= (\beta \rightarrow \alpha) \rightarrow \beta\end{aligned}$$

say whether or not  $\sigma_i \succ \tau_j$  holds for each of the four possibilities. [5 marks]

- (b) Give the axioms and rules for inductively generating ML typing judgements of the form

$$\Gamma \vdash M : \tau$$

where  $\Gamma = (\Gamma_{\text{tv}}, \Gamma_{\text{ta}})$  with  $\Gamma_{\text{tv}}$  a finite set of type variables,  $\Gamma_{\text{ta}}$  a finite function mapping some variables to type schemes whose free type variables are in  $\Gamma_{\text{tv}}$ , and  $\tau$  is a type whose type variables are in  $\Gamma_{\text{tv}}$ . You may restrict attention to expressions  $M$  involving variables, boolean values, conditionals, function abstraction and application, and let-expressions. [8 marks]

- (c) Explain why the expressions **let**  $x = M$  **in**  $M'$  and  $(\lambda x(M'))M$  can have different typing properties in the ML type system even though their evaluation behaviour is the same; illustrate your answer by taking  $M$  to be  $\lambda y(y)$  and  $M'$  to be  $xx$ . [7 marks]