Harvard School of Engineering and Applied Sciences — CS 152: Programming Languages

Type Inference; Parametric Polymorphism; Records and Subtyping Section and Practice Problems

Mar 24-25, 2016

---

1 Type Inference

(a) Recall the constraint-based typing judgment $\Gamma \vdash e\!:\!\tau \triangleright C$. Give inference rules for products and sums. That is, for the following expressions.

- (e_1, e_2)
- $\#1\ e$
- $\#2\ e$
- $\mathsf{inl}_{\tau_1+\tau_2}\ e$
- $\mathsf{inr}_{\tau_1+\tau_2}\ e$
- $\mathsf{case}\ e_1\ \mathsf{of}\ e_2 \mid e_3$

(b) Determine a set of constraints C and type τ such that

$$\vdash\ \lambda x\!:\!A.\, \lambda y\!:\!B.\, (\#1\ y) + (x\ (\#2\ y)) + (x\ 2)\ \ :\tau \triangleright C$$

and give the derivation for it.

(c) Recall the unification algorithm from Lecture 14. What is the result of $\mathit{unify}(C)$ for the set of constraints C from Question 1(b) above?

2 Parametric polymorphism

(a) For each of the following System F expressions, is the expression well-typed, and if so, what type does it have? (If you are unsure, try to construct a typing derivation. Make sure you understand the typing rules.)

- $\Lambda A.\, \lambda x\!:\!A \rightarrow \textbf{int}.\, 42$
- $\lambda y\!:\!\forall X.\, X \rightarrow X.\, (y\ [\textbf{int}])\ 17$
- $\Lambda Y.\, \Lambda Z.\, \lambda f\!:\!Y \rightarrow Z.\, \lambda a\!:\!Y.\, f\ a$
- $\Lambda A.\, \Lambda B.\, \Lambda C.\, \lambda f\!:\!A \rightarrow B \rightarrow C.\, \lambda b\!:\!B.\, \lambda a\!:\!A.\, f\ a\ b$

(b) For each of the following types, write an expression with that type.

- $\forall X.\, X \rightarrow (X \rightarrow X)$
- $(\forall C.\, \forall D.\, C \rightarrow D) \rightarrow (\forall E.\, \textbf{int} \rightarrow E)$
- $\forall X.\, X \rightarrow (\forall Y.\, Y \rightarrow X)$

3 Records and Subtyping

(a) Assume that we have a language with references and records.

(i) Write an expression with type

$$\{\, cell : \textbf{int ref}, inc : \textbf{unit} \rightarrow \textbf{int} \,\}$$

such that invoking the function in the field *inc* will increment the contents of the reference in the field *cell*.

(ii) Assuming that the variable y is bound to your expression, write an expression that increments the contents of the cell twice.

(b) The following expression is well-typed (with type **int**). Show its typing derivation. (Note: you will need to use the subsumption rule.)

$$(\lambda x\!:\!\{dogs : \textbf{int}, cats : \textbf{int}\}.\ x.dogs + x.cats)\ \{dogs = 2, cats = 7, mice = 19\}$$