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

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

Mar 20-23, 2018

---

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

Answer:

Note that in all of the rules below except for the rule for pairs (e_1, e_2), the types in the premise and conclusion are connected only through constraints. The reason for this is the same as in the typing rule for function application, and for addition: we may not be able to derive that the premise has the appropriate type, e.g., for a projection $\#1\ e$, we may not be able to derive that $\Gamma \vdash e\!:\!\tau_1 \times \tau_2 \triangleright C$. We instead use constraints to ensure that the derived type is appropriate.

$$\frac{\Gamma \vdash e_1\!:\!\tau_1 \triangleright C_1 \qquad \Gamma \vdash e_2\!:\!\tau_2 \triangleright C_2}{\Gamma \vdash (e_1, e_2)\!:\!\tau_1 \times \tau_2 \triangleright C_1 \cup C_2}$$

$$\frac{\Gamma \vdash e\!:\!\tau \triangleright C}{\Gamma \vdash \#1\ e\!:\!X \triangleright C \cup \{\tau \equiv X \times Y\}}\ X, Y \textit{ are fresh} \qquad \frac{\Gamma \vdash e\!:\!\tau \triangleright C}{\Gamma \vdash \#2\ e\!:\!Y \triangleright C \cup \{\tau \equiv X \times Y\}}\ X, Y \textit{ are fresh}$$

$$\frac{\Gamma \vdash e\!:\!\tau \triangleright C}{\Gamma \vdash \mathit{inl}_{\tau_1+\tau_2}\ e\!:\!\tau_1 + \tau_2 \triangleright C \cup \{\tau \equiv \tau_1\}} \qquad \frac{\Gamma \vdash e\!:\!\tau \triangleright C}{\Gamma \vdash \mathit{inr}_{\tau_1+\tau_2}\ e\!:\!\tau_1 + \tau_2 \triangleright C \cup \{\tau \equiv \tau_2\}}$$

$$\frac{\Gamma \vdash e_1\!:\!\tau_1 \triangleright C_1 \qquad \Gamma \vdash e_2\!:\!\tau_2 \triangleright C_2 \qquad \Gamma \vdash e_3\!:\!\tau_3 \triangleright C_3}{\Gamma \vdash \mathit{case}\ e_1\ \mathit{of}\ e_2 \mid e_3\!:\!Z \triangleright C_1 \cup C_2 \cup C_3 \cup \{\tau_1 \equiv X + Y, \tau_2 \equiv X \to Z, \tau_3 \equiv Y \to Z\}}\ X, Y, Z \textit{ are fresh}$$

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

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

and give the derivation for it.

Answer:

$$C = \{B \equiv X \times Y \; , \; X \equiv \textit{\textbf{int}} \; , \; B \equiv Z \times W \; , \; A \equiv W \to U \; , \; U \equiv \textit{\textbf{int}} \; , \; A \equiv \textit{\textbf{int}} \to V \; , \; V \equiv \textit{\textbf{int}}\}$$
$$\tau \equiv A \to B \to \textit{\textbf{int}}$$

To see how we got these constraints, we will consider the subexpressions in turn (rather than trying to typeset a really really big derivation).

The expression $\#1\ y$ requires us to add a constraint that the type of y (i.e., B) is equal to a product type for some fresh variables X and Y, thus constraint $B \equiv X \times Y$. (And expression $\#1\ y$ has type X.)

The expression $(\#2\ y)$ similarly requires us to add a constraint that the type of y (i.e., B) is equal to a product type for some fresh variables Z and W, thus constraint $B \equiv Z \times W$. (And expression $\#2\ y$ has type W.)

The expression $x\ (\#2\ y)$ requires us to add a constraint that unifies the type of x (i.e., A) with a function type $W \to U$ (where W is the type of $\#2\ y$ and U is a fresh type variable).

*The expression $x\ 2$ requires us to add a constraint that unifies the type of x (i.e., A) with a function type **int** $\to V$ (where **int** is the type of expression 2 and V is a fresh type).*

*The addition operations leads us to add constraints $X \equiv$ **int**, $U \equiv$ **int**, and $V \equiv$ **int** (i.e., the types of expressions $(\#1\ y)$, $(x\ (\#2\ y))$ and $(x\ 2)$ must all unify with **int**.*

(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?

Answer: *The result is a substitution equivalent to*

$$[A \mapsto \textit{\textbf{int}} \to \textit{\textbf{int}} \; , \; B \mapsto \textit{\textbf{int}} \times \textit{\textbf{int}} \; , \; X \mapsto \textit{\textbf{int}} \; , \; Y \mapsto \textit{\textbf{int}} \; , \; Z \mapsto \textit{\textbf{int}} \; , \; W \mapsto \textit{\textbf{int}} \; , \; U \mapsto \textit{\textbf{int}} \; , \; V \mapsto \textit{\textbf{int}}]$$

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.)

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

Answer:

- $\Lambda A.\, \lambda x\!:\!A \to$ ***int***$.\, 42$ *has type*
$$\forall A.\, (A \to \textit{\textbf{int}}) \to \textit{\textbf{int}}$$
- $\lambda y\!:\!\forall X.\, X \to X.\, (y\ [$***int***$])\ 17$ *has type*
$$(\forall X.\, X \to X) \to \textit{\textbf{int}}$$

- $\Lambda Y.\, \Lambda Z.\, \lambda f\!:\!Y \to Z.\, \lambda a\!:\!Y.\, f\ a$ *has type*

$$\forall Y.\, \forall Z.\, (Y \to Z) \to Y \to Z$$

- $\Lambda A.\, \Lambda B.\, \Lambda C.\, \lambda f\!:\!A \to B \to C.\, \lambda b\!:\!B.\, \lambda a\!:\!A.\, f\ a\ b$ *has type*

$$\forall A.\, \forall B.\, \forall C.\, (A \to B \to C) \to B \to A \to C$$

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

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

Answer:

- $\forall X.\, X \to (X \to X)$ *is the type of*

$$\Lambda X.\, \lambda x\!:\!X.\, \lambda y\!:\!X.\, y$$

- $(\forall C.\, \forall D.\, C \to D) \to (\forall E.\, \textbf{int} \to E)$ *is the type of*

$$\lambda f\!:\!\forall C.\, \forall D.\, C \to D.\, \Lambda E.\, \lambda x\!:\!\textbf{int}.\, (f\ [\textbf{int}]\ [E])\ x$$

- $\forall X.\, X \to (\forall Y.\, Y \to X)$ *is the type of*

$$\Lambda X.\, \lambda x\!:\!X.\, \Lambda Y.\, \lambda y\!:\!Y.\, x$$

3 Records and Subtyping

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

(i) Write an expression with type

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

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

Answer: *The following expression has the appropriate type.*

$$\begin{array}{l} \textit{let}\ x = \textit{ref}\ 14\ \textit{in} \\ \{\ \mathit{cell} = x,\ \mathit{inc} = \lambda u\!:\!\textbf{unit}.\, x := (!x + 1)\ \} \end{array}$$

(ii) Assuming that the variable y is bound to the expression you wrote for part (i) above, write an expression that increments the contents of the cell twice.

Answer:

$$\textit{let}\ z = y.\mathit{inc}\ ()\ \textit{in}\ y.\mathit{inc}\ ()$$

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

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

Answer:

For brevity, let $e_1 \equiv \lambda x\!:\!\{dogs : \textbf{int}, cats : \textbf{int}\}.\, x.dogs{+}x.cats)$ and let $e_2 \equiv \{dogs = 2, cats = 7, mice = 19\}$. The derivation has the following form.

$$\text{T-APP}\ \frac{\dfrac{\vdots_1}{\vdash e_1 : \{dogs : \textbf{int}, cats : \textbf{int}\} \to \textbf{int}} \qquad \dfrac{\vdots_2}{\vdash e_2 : \{dogs : \textbf{int}, cats : \textbf{int}\}}}{\vdash e_1\ e_2 : \textbf{int}}$$

The derivation of e_1 is straight forward:

$$\text{T-ABS}\ \frac{\text{T-ADD}\ \dfrac{\text{T-FIELD}\ \dfrac{\text{T-VAR}\ \dfrac{}{x\!:\!\{dogs : \textbf{int}, cats : \textbf{int}\} \vdash x\!:\!\{dogs : \textbf{int}, cats : \textbf{int}\}}}{x\!:\!\{dogs : \textbf{int}, cats : \textbf{int}\} \vdash x.dogs : \textbf{int}} \qquad \text{T-FIELD}\ \dfrac{\text{T-VAR}\ \dfrac{}{x\!:\!\{dogs : \textbf{int}, cats : \textbf{int}\} \vdash x\!:\!\{dogs : \textbf{int}, cats : \textbf{int}\}}}{x\!:\!\{dogs : \textbf{int}, cats : \textbf{int}\} \vdash x.cats : \textbf{int}}}{x\!:\!\{dogs : \textbf{int}, cats : \textbf{int}\} \vdash x.dogs + x.cats : \textbf{int}}}{\vdash e_1 : \{dogs : \textbf{int}, cats : \textbf{int}\} \to \textbf{int}}$$

The derivation of e_2 requires the use of subsumption, since we need to show that $e_2 \equiv \{dogs = 2, cats = 7, mice = 19\}$ has type $\{dogs : \textbf{int}, cats : \textbf{int}\}$.

$$\dfrac{\dfrac{\vdash 2 : \textbf{int} \quad \vdash 7 : \textbf{int} \quad \vdash 19 : \textbf{int}}{\vdash \{dogs = 2, cats = 7, mice = 19\} : \{dogs : \textbf{int}, cats : \textbf{int}, mice : \textbf{int}\}} \qquad \dfrac{}{\{dogs : \textbf{int}, cats : \textbf{int}, mice : \textbf{int}\} \leq \{dogs : \textbf{int}, cats : \textbf{int}\}}}{\vdash \{dogs = 2, cats = 7, mice = 19\} : \{dogs : \textbf{int}, cats : \textbf{int}\}}$$