Dependent types

CS 152 (Spring 2024)

Harvard University

Thursday, April 4, 2024

Today, we will learn about

- Dependent types
  - Motivation: reasoning precisely about vectors
  - LF (Logical Framework) type system

Dependent types: motivation

$$e ::= x \mid \lambda x.\, e \mid e_1\ e_2 \mid n \mid (e_1, e_2) \mid () \mid \mathsf{true} \mid \mathsf{false}$$
$$\mid \mathsf{init} \mid \mathsf{index}$$
$$v ::= \lambda x.\, e \mid n \mid < v_1, \ldots, v_n > \mid (v_1, v_2) \mid () \mid \mathsf{true} \mid \mathsf{false}$$

$$\frac{}{\mathsf{init}\ k\ v \longrightarrow < v_1, \ldots, v_k >}\ \forall i \in 1..k.\ v_i = v$$

$$\frac{}{\mathsf{index}\ \ < v_1, \ldots, v_k >\ \ i \longrightarrow v_{i+1}}$$

First attempt at type system

$$\frac{\forall i \in 1..n. \quad \Gamma \vdash v_i : \mathbf{bool}}{\Gamma \vdash < v_1, \ldots, v_n >: \mathbf{boolvec}\ n}$$

$$\frac{\Gamma \vdash e_1 : \mathbf{nat} \qquad \Gamma \vdash e_2 : \mathbf{bool}}{\Gamma \vdash \text{init}\ e_1\ e_2 : \mathbf{boolvec}\ e_1}$$

$$\frac{\Gamma \vdash e_1 : \mathbf{boolvec}\ e_3 \qquad \Gamma \vdash e_2 : \mathbf{nat}}{\Gamma \vdash \text{index}\ e_1\ e_2 : \mathbf{bool}}\ e_2 \leq e_3$$

Issues (1/3)

In the type for init, $(n : \textbf{nat}) \to \textbf{bool} \to \textbf{boolvec}\ n$, the first argument is somehow bound to the variable n which occurs in the return type of the function. What does this mean?

Issues (2/3)

The type **boolvec** e contains an arbitrary expression expression e. What do the types **boolvec** $(9+1)$ or **boolvec** x mean? And what does it mean in the proposed typing rule for index to have a side condition $e_1 \leq e_3$?

Issues (3/3)

The expression e in the type **boolvec** e should be of type **nat**. How do we ensure that e is limited to expressions of type **nat**?

LF (Logical Framework)

Expressions	$e ::= x \mid \lambda x{:}\tau.\, e \mid e_1\ e_2 \mid n \mid e_1 + e_1$
	$\mid < v_1, \ldots, v_n > \mid \ldots$
Types	$\tau ::= \textbf{nat} \mid \textbf{boolvec} \mid \textbf{bool} \mid \textbf{unit}$
	$\mid \tau\ e \mid (x{:}\tau_1) \rightarrow \tau_2$
Kinds	$K ::= \textbf{Type} \mid (x{:}\tau) \Rightarrow K$

Judgment for Expressions: $\Gamma \vdash e : \tau$

$$\frac{\Gamma \vdash \tau :: K}{\Gamma \vdash x : \tau}\; x : \tau \in \Gamma \qquad \frac{}{\Gamma \vdash n : \textbf{nat}}\; n \in \mathbb{N}$$

$$\frac{\Gamma \vdash e_1 : \textbf{nat} \quad \Gamma \vdash e_2 : \textbf{nat}}{\Gamma \vdash e_1 + e_2 : \textbf{nat}}$$

$$\frac{\text{For all } i \in 1..n. \qquad \Gamma \vdash v_i : \textbf{bool}}{\Gamma \vdash < v_1, \ldots, v_n > : \textbf{boolvec } n}$$

$$\frac{\Gamma \vdash \tau :: \textbf{Type} \qquad \Gamma, x : \tau \vdash e : \tau'}{\Gamma \vdash \lambda x : \tau .\, e : (x : \tau) \to \tau'}$$

$$\frac{\Gamma \vdash e_1 : (x : \tau') \to \tau \qquad \Gamma \vdash e_2 : \tau'}{\Gamma \vdash e_1 \; e_2 : \tau\{e_2 / x\}}$$

Judgment for Expressions: $\Gamma \vdash e : \tau$

$$\textsc{Conversion}\ \frac{\Gamma \vdash e : \tau' \qquad \Gamma \vdash \tau \equiv \tau' :: \mathbf{Type}}{\Gamma \vdash e : \tau}$$

Judgment for Types: $\Gamma \vdash \tau :: K$

$$\frac{\Gamma \vdash K \text{ ok}}{\Gamma \vdash X :: K} \; X : K \in \Gamma$$

$$\frac{\Gamma \vdash \tau :: \textbf{Type} \qquad \Gamma, x : \tau \vdash \tau' :: \textbf{Type}}{\Gamma \vdash (x : \tau) \to \tau' :: \textbf{Type}}$$

$$\frac{\Gamma \vdash \tau :: (x : \tau') \Rightarrow K \qquad \Gamma \vdash e : \tau'}{\Gamma \vdash \tau \; e :: K\{e/x\}}$$

$$\textsc{Conversion} \; \frac{\Gamma \vdash \tau :: K' \qquad \Gamma \vdash K \equiv K'}{\Gamma \vdash \tau :: K}$$

Judgment for Kinds: $\Gamma \vdash K$ ok

$$\frac{}{\Gamma \vdash \textbf{Type} \text{ ok}} \qquad \frac{\Gamma \vdash \tau :: \textbf{Type} \qquad \Gamma, x{:}\tau \vdash K \text{ ok}}{\Gamma \vdash (x{:}\tau) \Rightarrow K \text{ ok}}$$

Judgments for equivalence

- We would like to consider the types **boolvec** 19 and **boolvec** $(12 + 7)$ to be equivalent.
- Relation means that (under context Γ) expressions/types/kinds are equivalent and have the given type/kind.
  - $\Gamma \vdash e_1 \equiv e_2 : \tau$
  - $\Gamma \vdash \tau_1 \equiv \tau_2 :: K$
  - $\Gamma \vdash K_1 \equiv K_2$

Judgments for term equivalence

$$\frac{\Gamma \vdash \tau_1 \equiv \tau_2 :: \textbf{Type} \qquad \Gamma, x : \tau_1 \vdash e_1 \equiv e_2 : \tau}{\Gamma \vdash \lambda x : \tau_1.\, e_1 \equiv \lambda x : \tau_2.\, e_2 : (x : \tau_1) \to \tau}$$

$$\frac{\Gamma \vdash e_1 \equiv e_2 : (x : \tau) \to \tau' \qquad \Gamma \vdash e_1' \equiv e_2' : \tau}{\Gamma \vdash e_1\ e_1' \equiv e_2\ e_2' : \tau'\{e_1'/x\}}$$

$$\frac{\Gamma, x : \tau \vdash e : \tau' \qquad \Gamma \vdash e' : \tau}{\Gamma \vdash (\lambda x : \tau.\, e)\ e' \equiv e\{e'/x\} : \tau'\{e'/x\}}$$

$$\frac{\Gamma \vdash e : (x : \tau) \to \tau' \qquad x \notin FV(e)}{\Gamma \vdash (\lambda x : \tau.\, e\ x) \equiv e : (x : \tau) \to \tau'}$$

Judgments for term equivalence

$$\frac{\Gamma \vdash e_1 \equiv e_2 : \mathbf{nat} \quad \Gamma \vdash e_1' \equiv e_2' : \mathbf{nat}}{\Gamma \vdash e_1 + e_1' \equiv e_2 + e_2' : \mathbf{nat}}$$

$$\frac{}{\Gamma \vdash k + m \equiv n : \mathbf{nat}} \; n \text{ is the sum of } k \text{ and } m$$

$$\frac{\Gamma \vdash e : \tau}{\Gamma \vdash e \equiv e : \tau} \qquad \frac{\Gamma \vdash e_1 \equiv e_2 : \tau}{\Gamma \vdash e_2 \equiv e_1 : \tau}$$

$$\frac{\Gamma \vdash e_1 \equiv e_2 : \tau \quad \Gamma \vdash e_2 \equiv e_3 : \tau}{\Gamma \vdash e_1 \equiv e_3 : \tau}$$

Judgments for type equivalence

$$\frac{\Gamma \vdash \tau_1 \equiv \tau_2 :: \textbf{Type} \qquad \Gamma, x : \tau_1 \vdash \tau_1' \equiv \tau_2' :: \textbf{Type}}{\Gamma \vdash (x : \tau_1) \to \tau_1' \equiv (x : \tau_2) \to \tau_2' :: \textbf{Type}}$$

$$\frac{\Gamma \vdash \tau_1 \equiv \tau_2 :: (x : \tau) \Rightarrow K \qquad \Gamma \vdash e_1 \equiv e_2 : \tau}{\Gamma \vdash \tau_1\ e_1 \equiv \tau_2\ e_2 :: K\{e_1/x\}}$$

$$\frac{\Gamma \vdash \tau :: K}{\Gamma \vdash \tau \equiv \tau :: K} \qquad \frac{\Gamma \vdash \tau_1 \equiv \tau_2 :: K}{\Gamma \vdash \tau_2 \equiv \tau_1 :: K}$$

$$\frac{\Gamma \vdash \tau_1 \equiv \tau_2 :: K \qquad \Gamma \vdash \tau_2 \equiv \tau_3 :: K}{\Gamma \vdash \tau_1 \equiv \tau_3 :: K}$$

Judgments for kind equivalence

$$\frac{\Gamma \vdash \tau_1 \equiv \tau_2 :: \textbf{Type} \qquad \Gamma, x : \tau_1 \vdash K_1 \equiv K_2}{\Gamma \vdash (x : \tau_1) \Rightarrow K_1 \equiv (x : \tau_2) \Rightarrow K_2}$$

$$\frac{\Gamma \vdash K \text{ ok}}{\Gamma \vdash K \equiv K} \qquad \frac{\Gamma \vdash K_1 \equiv K_2}{\Gamma \vdash K_2 \equiv K_1}$$

$$\frac{\Gamma \vdash K_1 \equiv K_2 \qquad \Gamma \vdash K_2 \equiv K_3}{\Gamma \vdash K_1 \equiv K_3}$$

Equivalence Examples?

- The types **boolvec** 42 and **boolvec** $(35 + 7)$ are equivalent.
- But what about if we are in a context where we have variables x and f of type **nat** and **nat** $\rightarrow$ **nat**, respectively, where we know that $f\ x = 7$? Should we consider the types **boolvec** $(f\ x)$ and **boolvec** 7 to be equivalent?

Back to vectors...

- **boolvec** *e*: enforce *e* of type **nat**.
- init: $(n : \mathbf{nat}) \rightarrow \mathbf{bool} \rightarrow \mathbf{boolvec}\ n$.
- also join: $(n : \mathbf{nat}) \rightarrow (k : \mathbf{nat}) \rightarrow \mathbf{boolvec}\ n \rightarrow \mathbf{boolvec}\ k \rightarrow \mathbf{boolvec}\ (n + k)$

Back to vectors...

What about the type of index?

Back to vectors...

What about the type of asPairs?
asPairs $< v_1, \ldots, v_n >$ evaluates to
$(v_1, (v_2, \ldots (v_n, ())\ldots))$