Lecture 05.27.26. Wed

- Announcements
  - I will post slides + notes
  - I will post a link to make OH by appt.
- Agenda
  - Review the typing rules: Γ
  - Unification (Gofai)
- Review of typing rules

$e ::= n \mid b \mid x \mid e1 + e2 \mid \backslash x \to e \mid e1\ e2 \mid$
$\quad\quad \text{let } x = e1 \text{ in } e2$

$T ::= \text{Int} \mid \text{Bool} \mid T1 \to T2$

$$\frac{}{\Gamma \vdash n :: \text{Int}}\ [\text{T-Int}] \qquad \frac{}{\Gamma \vdash b :: \text{Bool}}\ [\text{T-bool}]$$

$$\frac{\Gamma \vdash e1 :: \text{Int} \qquad \Gamma \vdash e2 :: \text{Int}}{\Gamma \vdash e1 + e2 :: \text{Int}}\ [\text{T-Add}]$$

$$\frac{(x, T) \text{ in } \Gamma}{\Gamma \vdash x :: T}\ [\text{T-Var}]$$

$$\frac{\Gamma \vdash e1 :: T1 \qquad (x, T1) : \Gamma \vdash e2 :: T2}{\Gamma \vdash \text{let } x = e1 \text{ in } e2 :: T2}\ [\text{T-Let}]$$

$\backslash x \to x + 3$

$$\frac{???}{\Gamma \vdash \backslash x \to e :: T_1 \to T2}$$

$$\frac{(x, T1) : \Gamma \vdash e :: T2}{\Gamma \vdash \backslash x \to e :: T_1 \to T2}\ [\text{T-Lam}]$$

$$\frac{\Gamma \vdash e1 :: T1 \to T2 \qquad \Gamma \vdash e2 :: T1}{\Gamma \vdash e1\ e2 :: T_2}\ [\text{T-App}]$$

* well-typed or "ill typed"

- e is well-typed in a type env. Γ if we can write down a derivation for $\Gamma \vdash e :: T$ for some T
  - If we can't, then the expression is ill-typed.
- Let's derive $\underline{\backslash x \to x + 1}$

$$\frac{(x, \text{Int}) \text{ in } [(x, \text{Int})]}{}$$

prove

$[(x, Int)] \vdash x :: Int \quad [(x, Int)] \vdash 1 :: Int$

$[(x, Int)] \vdash x + 1 :: Int$

- - - - - - - - - - - - - - - - - -

$[\,] \vdash \backslash x \to \underline{x+1} :: ???$

$Int \to Int$

(x, a) in $[(x, a)]$

$[(x, a)] \vdash x :: a \quad [(x, a)] \vdash 1 :: a$

$\to [(x, a)] \vdash x + 1 :: a$

- - - - - - - - - - - - - - - - - -

$[\,] \vdash \backslash x \to \underline{x+1} :: ???$

$a \to a$

$$\frac{(x, T1) : \Gamma \vdash e :: T2}{\Gamma \vdash \backslash x \to e :: T1 \to T2} \text{[T-Lam]}$$

$(x, T1) : \Gamma \to (x, a) : [\,] \to [(x, a)]$

* UNIFICATION (yay!)

- placeholders: type variables
  a, b, c
- [a/Int , b/c → c]
  substitutions
- Apply a substitution to a type. If we have the substitut. above: [a/Int, b/c → c]
  a → a you'd get Int → Int

* Unification

- Finding a substitution that makes two types the same when it is applied to both of them.
- We call this substitution a unifier for those types

example - types a → Int Bool → b
[a / Bool, b / Int] make both Bool → Int.