10.2 10.3 10.4 10.9

encountering a variable:

$$\Gamma \vdash x : T \quad \text{if} \quad \Gamma(x) = T$$

$(\lambda x{:}num.\ (+\ x\ \ulcorner 37 \urcorner))$

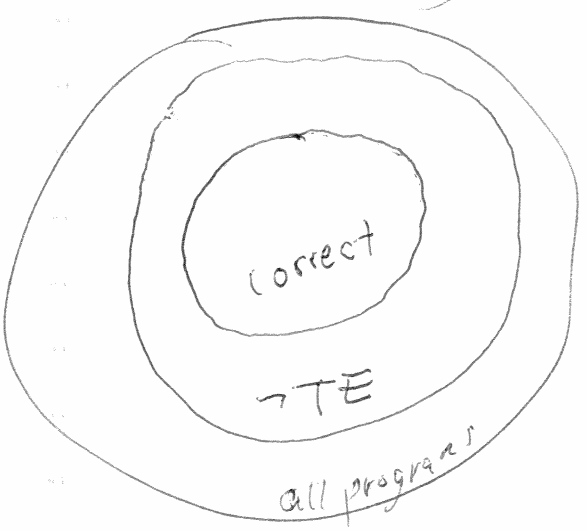
$$\frac{\Gamma[x \leftarrow T] \vdash M : S}{\Gamma \vdash (\lambda x{:}T.\ M) : (T \to S)}$$

$(M\ N)$

$$\frac{\Gamma \vdash M : (T \to S) \quad \Gamma \vdash N : T}{\Gamma \vdash (M\ N) : S}$$

complement of soundness is completeness.

$((if0\ \ulcorner 0 \urcorner)\ (\lambda x{:}num.\ \ulcorner 0 \urcorner)\ (\lambda x{:}(num \to num).\ x))$
$\ulcorner 42 \urcorner)\ldots$



"reduces to"

$e \to e$

[number + number']

[entering math zone]

number + number'

[exit math zone]

$((5+3)+(8+9)) \to (8+(8+9)) \to (8+17)$

$\to 25$

[The job of the ":" is to predict what "$\to$" does] <Soundness>

2 properties for Soundness

① Progress: $\forall e,t.\ e:t \Rightarrow (\exists e'.\ e \to e')$ or e is a value

② Preservation: $\forall e,t,e'.\ e:t \wedge e \to e' \Rightarrow e':t$

values = number or true

stuck!

$((8+3)+true) \to (11 + true) \Rightarrow$ Type Error

not a value!

Simply Typed ISWIM

Meta variables

$$M ::= X \mid (\lambda X : T.\ M) \mid (M\ M) \mid b \mid (O^n\ M \ldots M)$$

$$T ::= B \mid (T \to T)$$

$(\lambda x : Num.\ (+\ x\ \ulcorner 2 \urcorner))$ $\quad (Num \to Num)$ (Domain → Num, Range → Num)

$(Num \to (Num \to Num))$

$B : b \to T$

$\Delta^n : O^n \underbrace{T \ldots T}_{n} \to T$

$$\frac{\vdash M_1 : B_1 \quad \ldots \quad \vdash M_N : B_N}{\vdash (O^n\ M_1 \ldots M_N) : T} \text{ if } \Delta^n(O^n\ B_1 \ldots B_n) = T$$

How do we Type-check $(\lambda x : T.\ M)$ and $(M\ M)$

We need a Type Environment Γ gamma

... mapping of variables to types

Type Judgement $\Gamma \vdash M : T$ "The expression M is classified as Type T in the type environment Γ"

Types only correspond to Values in the language.

Type Relation → "has type"

$e : t$ → some type t

program expression

- A language with NO Type Errors

$e = number \mid e + e$ $\quad T = Num$

antecedent → $\frac{}{number : Num}$ ← consequent

$\frac{e : Num \quad e' : Num}{e + e' : Num}$

$\exists e$, s.t. $\forall t \in T \ \neg(e : T)$? — No

$e = number \mid (e + e) \mid true$ $\quad T = Num \mid Boolean$

$\frac{}{number : Num}$ $\quad \frac{e : Num \quad e' : Num}{(e + e') : Num}$ $\quad \frac{}{true : Bool}$

$(3 + true) \Rightarrow$ Type Error

$\exists e, \forall T \ \neg(e : T) \rightarrow$ Type Error

Type Systems

What is a Type System?

- static, syntatic discipline for avoiding errors.
  - disallow the evaluation of expressions that may get stuck or signal run-time errors.

Examples in ISWIM

(1) (+ ⌜5⌝ ⌜7⌝)
(2) (+ ⌜8⌝ −) → Not well-formed
(3) (+ ⌜3⌝ (λx.⌜9⌝)) → not well typed

- Type systems only apply to syntatically sound or well-formed programs

- Type systems consist of
  (1) a language of claims
  (2) claim checker

(1) type language is notation for formulating simple claims about programs.
  ↪ these are called Types
(2) checks if types follow the set of principles or type rules.

→ when a type cannot be validated for an expression → Type Error