24-1

$T_1 = T_1 \Rightarrow X$

$T = B \mid T \Rightarrow T \mid \forall x.\ T \mid X \mid T + T \mid T \times T$

base type num — functions — polymorphism — disjoint unions or variants — pairs

A list of numbers is either
1) a NULL (or empty)
2) a pair of a number and a list of numbers

$NList = (MT) + (Num \times NList)$

$MT \in B$ $Num \in B$

$T = \ldots \mid uX.\ T$

$NList = uX.\ MT + (Num \times X)$

$Bin = uX.\ LEAF + (X \times Num \times X)$

$$\frac{\Gamma \vdash M : T_1 \to T_2 \quad \Gamma \vdash N : T_1}{\Gamma \vdash M\ N : T_2}$$

switch on expre e
case e = (M N) :
  T_m = rec (m)
  check if $\exists T_1, T_2.\ T_m = T_1 \to T_2$
  T_N = rec (N)
  check if $T_n = T_1$
  return T_2

$$\frac{}{Nat = Nat}$$

$$\frac{T_1 = T_3 \quad T_2 = T_4}{T_1 \Rightarrow T_2 = T_3 \Rightarrow T_4}$$

$(\forall A.\ A \to A) = \forall 0 \to 0$

$(\forall a. \forall B.\ A \to B) = \forall\forall 1 \to 0$

$(\forall A.\ A \Rightarrow A) = (\forall B.\ B \Rightarrow B)$? ✓

$(uA.\ Num + (Num \times A)) = (uB.\ Num + (Num \times B))$? ✓

$(uA.\ \boxed{A \Rightarrow num}) = (uA.\ A \Rightarrow num) \Rightarrow num$?

rep => X $(A \to Num)[A \Leftarrow uA. A \to num]$

$$\frac{T[A \leftarrow uA.T] = T'}{uA.T = T'}$$
unfolding

$$\frac{T = T'[A \leftarrow uA.T']}{T = uA.T'}$$
folding

A type system with these rules is
equi-recursive
(a type is equal to its own unfold)

No efficient alg for when to apply rule
Type inference on equi-recursive systems is <u>undecidable</u>

---

iso-recursive puts fold & unfold into the language

$M = \ldots \mid \text{fold } M \mid \text{unfold } M$
$V = \ldots \mid \text{fold } V$
$E = \ldots \mid \text{fold } E \mid \text{unfold } E$

$$E[\text{unfold } (\text{fold } V)] \mapsto E[V]$$

(fold T M)

$$\frac{\Gamma \vdash M : T[A \leftarrow uA.T]}{\Gamma \vdash (\text{fold } M) : uA.T}$$

$$\frac{\Gamma \vdash M : uA.T}{\Gamma \vdash (\text{unfold } M) : T[A \leftarrow uA.T]}$$

A constructor always <u>folds</u> the object
An accessor always <u>unfolds</u> the object
(or pattern matcher)

24-3

$List = \forall \alpha . \mu \ell . Bool + (\alpha \times \ell)$

null := $\Lambda \alpha .$ (fold (inL false)) $(: List[\alpha]$

cons := $\Lambda \alpha .$ $(: \alpha \to List[a] \to List[a])$

$\lambda v : \alpha .$

$\lambda r : List[\alpha] .$

(fold (inR (pair v r)))

first := $\Lambda \alpha .$ $(: List[a] \to a)$

$\lambda l : List[\alpha]$

match (unfold l) with

case inL : $(\lambda n : bool , \Omega)$

case inR : $(\lambda p : (\alpha \times List[\alpha]) , fst\ p)$

$T = B \mid T \to T \mid \forall x . T \mid X \mid T + T \mid T \times T \mid \mu X . T$

$|Bool| = 2$ $\quad |Bool| + |Bool| = 2 + 2 = 4$

$|Bool| \times |Bool| = 4$ $\quad |Bool| + |Fool| = 5$

$|Bool \to Bool| = ?$ interesting $\quad |Bool| \times |Fool| = 6$

want a T, $|T| = 0$ $\longrightarrow$ call T "0" "zero"

want a T, $|T| = 1$ $\longrightarrow$ call T "1" "unit"

$tt : 1$

Bool $\simeq$ $1 + 1$?

true, false $\quad$ inL tt, inR tt

Nat $\simeq$ $\mu N . \ 1 + N$

3, 4, 5 $\quad$ either 0 or succ of another number

$0 =$ inL tt

$3 =$ inR (inR (inR (inL tt)))

Algebraic Data Types (types formed from $\overrightarrow{0, 1, +, \times}$ and μ)

24-4/

$\delta_x 0 = 0$

$\delta_x 1 = 0$

$\delta_x x = 1$

$\delta_x y = 0$

$\delta_x A + B = \delta_x A + \delta_x B$

$\delta_x A \times B = \delta_x A \times B + \cancel{\delta_x} A \times \delta_x B$

$\delta_\alpha \text{List}[\alpha]$?

$$
\begin{aligned}
& \delta_\alpha \text{List}[\alpha] \\
&= \delta_\alpha (1 + (\alpha \times \text{List}[\alpha])) \\
&= \delta_\alpha 1 + \delta_\alpha (\alpha \times \text{List}[\alpha]) \\
&= 0 + (\delta_\alpha \alpha) \times \text{List}[\alpha] + (\alpha \times \delta_\alpha \text{List}[\alpha]) \\
&= \text{List}[\alpha] + (\alpha \times \delta_\alpha \text{List}[\alpha])
\end{aligned}
$$

$Z[\alpha] = \text{List}[\alpha] + (\alpha \times Z[\alpha])$

A Z of numbers is either
a list of numbers
or, a number and a Z of numbers

$(1, (2, (3, [4, 5, 6]))) \in Z$

Z is a position in a list
1, 2, 3, 4, 5, 6

Zipper
a.k.a gap buffer (zipper of list of array of bytes)

$f(5) = 7$

$f(6) = ?$

$$\underbrace{f(5)}_{\text{old value}} + \underbrace{\overset{+1}{f'(\cancel{5})}}_{\text{change in value}} = f(6)$$

$$\int_{x=5}^{6} f'(x)$$

Incremental Computation