generate-constraint :: Expr → Set<Constraints>

solve :: Set<Constraint> → Set<Constraint>

$$\frac{CHS}{CHS} = \frac{CHS}{\dfrac{\begin{matrix}type(e) \\ id\text{-}type(x)\end{matrix} = \begin{matrix}base \\ con\end{matrix}}{\text{Shape of output constraints}}}$$

CHS:

type (Expr expr)

base (n) (n = "Num", "Bool")

id-type(x) (x = "n", "x")

con (name, pieces) name "Arrow" con("Arrow", [list: base("Num"), base("Bool")])
"List"
CHS

$$\frac{5x}{} = \frac{7y}{}$$

$$\frac{4+5y}{} = \frac{4z}{}$$

$$\frac{\quad}{} = \frac{\quad}{\quad}$$

Unify :: (Constrs :: Set<Constraint>, subst :: Set<Constraint>) → Set<Constraint>

if is-empty(constrs): subst

else

Consider elt of constrs

- type(e) = CHS

  if type(e)=CHS' in subst

    add CHS = CHS' to constrs and recur

    else

      add type(e) = CHS to subst and recur

      replace type(e) with CHS in subst

if type(e) is inside CHS, error

extend subst with type(e)=CHS (need to do more than just this)

- id-type(x) = CHS

  same as above

- base(n) = base(n') if n=n' : recur else: error

  base(n) = con(...) error

  base(n) = CHS add CHS = base(n) to constrs, recur

- con(n, ps) = basic error
  con(n, ps) = CHS add CHS = con(n, ps) and recur
  con(n, ps) = con(n', ps')
    if n ≠ n' error
      add $p_1 = p_1'$ for p_i, p_i' from ps, ps'
      $p_2 = p_2'$
      ⋮
      $p_n = p_n'$
      to consts and recur

1. Terminates because finite types (finite con(...))

2. (lam (x:ANY) x) = e

a → a

type(e) = con(→, id(x), id(x))

type(x) = id(x)

Underconstrained System

3. (lam (x) (x x)) = e ω

type(e) = con(→, id(x), type((x x)))

type(x) = con(→, type(x), type((x x)))

Occurs CHECK

a → b found a and "a occurs in b"

4. Principal types

Robin Milner
ML
Turing Award