$3 :: \mathrm{Int}$

$\mathrm{add}\ 3\ 4 :: \mathrm{Int}$

$\mathrm{add}\ 3 :: \mathrm{Int} \to \mathrm{Int}$

$\mathrm{add} :: \mathrm{Int} \to \mathrm{Int} \to \mathrm{Int}$

$\mathrm{Int}, \mathrm{Bool}; \mathrm{Int} \to \mathrm{Bool}$

$(\lambda x : a \to a . x)(\lambda x : a . x)$

$id = \lambda x . x \quad :: \forall a . a \to a$

$id\ 3 :: \mathrm{Int}$

$id\ id :: \forall a . a \to a$

$id\ (id\ 3) :: \mathrm{Int}$

let $x = id\ id$ in $\longleftarrow$ $id :: \forall a . a \to a$

let $y = id\ 3$ in $\longleftarrow$ $id :: \mathrm{Int} \to \mathrm{Int}$

$x\ y$

Hindley Milner type system:

only values bound in let are polymorphic (can be instantiated)

$\lambda$-abs are monomorphic

$\lambda f . (f\ 0, f\ \mathrm{True})$ ✗ let $f = \lambda x . x$ in $(f\ 0, f\ \mathrm{True})$ ✓

Specialization: $\forall\alpha.\alpha\to\alpha \sqsubseteq Int \to Int$
$[\alpha \mapsto Int]$

$\sqsubseteq$ - p.o $\begin{pmatrix} \text{refl} \\ \text{antisymm} \\ \text{transit.} \end{pmatrix}$

$\forall\alpha.\alpha$ - smallest

$$\frac{\tau' = \{\alpha_i \mapsto \tau_i\}\,\tau \qquad \beta_i \notin free(\forall\alpha_1 \dots \alpha_n.\,\tau)}{\forall\alpha_1 \dots \alpha_n.\,\tau \sqsubseteq \forall\beta_1 \dots \beta_m.\,\tau'}$$

more general

when applying a subst, unbound vars are not to be replaced

Add the rules

$$\frac{\Gamma \vdash e : \sigma' \qquad \sigma' \sqsubseteq \sigma}{\Gamma \vdash e : \sigma}\ \text{Inst} \qquad\qquad \frac{\Gamma \vdash e : \sigma \qquad \alpha \notin free(\Gamma)}{\Gamma \vdash e : \forall\alpha.\,\sigma}\ \text{Gen}$$

$$\frac{x : \sigma \in \Gamma}{\Gamma \vdash x : \sigma}\ \text{Var} \qquad\qquad \frac{\Gamma \vdash e_0 : \tau \to \tau' \qquad \Gamma \vdash e_1 : \tau}{\Gamma \vdash e_0\, e_1 : \tau'}\ \text{App}$$

$$\frac{\Gamma, x : \tau \vdash e : \tau'}{\Gamma \vdash \lambda x.e : \tau \to \tau'}\ \text{Abs} \qquad\qquad \frac{\Gamma \vdash e_0 : \sigma \qquad \Gamma, x : \sigma \vdash e_1 : \tau}{\Gamma \vdash \text{let } x = e_0 \text{ in } e_1 : \tau}\ \text{Let}$$

monotype

polytype

$\Gamma = \{id : \forall\alpha.\,\alpha \to \alpha;\ n : Int\}$

$$\dfrac{\dfrac{}{\Gamma \vdash n : Int}\ \text{Var} \qquad \dfrac{\dfrac{}{\Gamma \vdash id : \forall\alpha.\,\alpha \to \alpha}\ \text{Var}}{\Gamma \vdash id : int \to int}\ \text{Inst}}{\Gamma \vdash id(n) : Int}\ \text{App}$$

$$\dfrac{\dfrac{\dfrac{x:\alpha \vdash x:\alpha}{\vdash \lambda x.x:\alpha\to\alpha}\,\text{Var}}{\vdash \lambda x.x:\forall\alpha.\alpha\to\alpha}\,\text{Gen} \qquad \dfrac{}{id:\forall\alpha.\alpha\to\alpha \vdash id:\forall\alpha.\alpha\to\alpha}\,\text{Var}}{\vdash \text{let } id=\lambda x.x \text{ in } id:\forall\alpha.\alpha\to\alpha}\,\text{Let}$$

$id : \forall\alpha.\alpha \to \alpha$

$\hookrightarrow id : w \to w$

$f : y \to y$

$x : \forall\alpha.\alpha$

$\hookrightarrow z$

## Algorithm W

$$\dfrac{x:\sigma \in \Gamma \qquad \tau = inst(\sigma)}{\Gamma \vdash x:\tau, \emptyset}\,\text{Var}$$

$$\dfrac{\Gamma \vdash e_0:\tau_0, S_0 \qquad S_0\Gamma \vdash e_1:\tau_1, S_1 \qquad \tau' = newvar \qquad S_2 = unify(S_1\tau_0, \tau_1 \to \tau')}{\Gamma \vdash e_0\, e_1 : S_2\tau', S_2 S_1 S_0}\,\text{App}$$

$$\dfrac{\tau = newvar \qquad \Gamma, x:\tau \vdash e:\tau', S}{\Gamma \vdash \lambda x.e : S\tau \to \tau', S}\,\text{Abs}$$

$$\dfrac{\Gamma \vdash e_0:\tau, S_0 \qquad S_0\Gamma, x:\overline{S_0\Gamma}(\tau) \vdash e_1:\tau', S_1}{\Gamma \vdash \text{let } x = e_0 \text{ in } e_1 : \tau', S_1 S_0}\,\text{let}$$

$\bar{\Gamma}(\tau) = \forall \bar{\alpha}.\tau$ , $\bar{\alpha} = free(\tau) \setminus free(\Gamma)$

Unification: unify(a, b) = $\Theta$ $\qquad a\Theta = b\Theta$

- a is Var $\Rightarrow$ subst $[a \mapsto b]$
- b is Var $\Rightarrow$ subst $[b \mapsto a]$
- structure matches $\Rightarrow$ unify substructures
- otherwise fail

Unification of functions: unify($a \to b$ , $c \to d$)

1) $s_1$ = unify (a , c)
2) $s_2$ = unify ( $b s_1$ , $d s_1$)
3) $s_1 + s_2$

Ex: unify($a \to b$ , $c \to a$)

unify(a, c) = $[a \mapsto c]$

unify( $b[a \mapsto c]$ , $a[a \mapsto c]$)

$\hookrightarrow$ unify (b, c) = $[b \mapsto c]$

result : $[a \mapsto c, b \mapsto c]$