Motivating the PT algorithm

11.4.2025.

$PT(\lambda x.P)$ is easy.

Consider $PT(PQ)$.

- let $PT(P) \equiv \rho \to \sigma$ $PT(Q) \equiv \tau$

let $PT(PQ)$ has no free variables,

we need to make ρ and τ same in order to apply $(\to E)$.

let S_1, S_2 be substitutions s.t.

$$\frac{P: \Gamma \vdash S_1(\rho) \to S_1(\sigma) \qquad Q: \Gamma \vdash S_2(\tau)}{\Gamma \vdash \cancel{S}\, PQ : S_1(\sigma)}\ (\to E)$$

$$S_1(\rho) \equiv S_2(\tau)$$

Then $PT(PQ) \equiv S_1(\sigma)$ by $(\to E)$

Thus the problem of deciding whether PQ is typable reduces to that of finding S_1 and S_2 s.t. $S_1(\rho) \equiv S_2(\tau)$.

Definition (Common instance (c.i))

~~(i)~~ Iff $v \equiv S_1(\rho) \equiv S_2(\tau)$ we call v a c.i. of the pair (ρ, τ) and we call (S_1, S_2) a pair of converging substitutions for (ρ, τ).

~~(ii)~~

e.g.

1. $(a,\ b \to b)$ $S_1 \equiv [b \to b/a]$ $S_2 \equiv [b/b]$
2. $(f(1, y),\ f(x, 2))$ $S_1 \equiv [2/y]$ $S_2 = [1/x]$

   constant   variable

3. $(foo(a, x),\ foo(y, b))$ $S_1 \equiv [b/x]$, $S_2 \equiv [a/y]$

   variable

4. $(foo(a, x, c),\ foo(y, 2))$ no substitution exists.

Defn. Most general c.i (mgci) for (ρ, τ) is a c.i. v_0 s.t. every other c.i. v is an instance of v_0.

1. $S_2 \equiv$ empty   2. mgci   3. mgci

— x —

11.2.2025

The principal-type algorithm (PT-algorithm) (type checking algorithm).

$id = \lambda x.x$, $id : a \to a$ same as $id : b \to b$ *

The type $(a \to a)$ is called the principal type of id (simplest form).

No other type that can be assigned to id ~~has~~ is simpler than $(a \to a)$. For eg. $(a \to a) \to (a \to a)$, Infact all other types for id are substitution instances of the basic type $(a \to a)$.

Definition (Type-Substitution) :-

A type substitution $\mathbf{S}$ (bold face S) is any expression

$[\sigma_1/a_1, \ldots, \sigma_n/a_n] \leftarrow$ simultaneously substituting σ_i for a_i

a_i's : type variables - distinct - σ_i : any type

(i) $\mathbf{S}(a_i) \equiv \sigma_i$

(ii) $\mathbf{S}(b) \equiv b$ if b is an atom $\notin \{a_1 \ldots a_n\}$

(iii) $\mathbf{S}(\rho \to \tau) \equiv \mathbf{S}(\rho) \to \mathbf{S}(\tau)$

We call $\mathbf{S}(\tau)$ an instance of τ. (substitution-instance of τ)

Definition (PT) A PT of a term M (in TA_λ) is a type τ s.t.

(i) $\Gamma \vdash_\lambda M : \tau$ for some Γ

(ii) if $\Gamma' \vdash_\lambda M : \sigma$ for some Γ' and σ then σ is an instance of τ.

Note : • A type τ is a PT of M iff,

for all types σ, $(\exists \Gamma'.\ \Gamma' \vdash_\lambda M : \sigma) \iff \sigma$ is an instance of τ.

• A PT of a term is unique. PT(M) - denotes PT of M.

---

* alphabetic variants.