Oishi Type System に対する型推論アルゴリズム T_2

2016/1/15

1 型推論用の型システム T_2

(var0)

$$\frac{(x : t') \in \Gamma}{\Gamma \vdash x : t;\ \sigma}\ Constr;\ \Gamma \models t \geq t'$$

(var1)

$$\frac{(u : t)^{\gamma'} \in \Gamma}{\Gamma \vdash^{\gamma} u : t;\ \sigma}\ Constr;\ \Gamma \models \gamma \geq \gamma'$$

(add)

$$\frac{\Gamma \vdash x : \mathtt{int}; \sigma \quad \Gamma \vdash y : \mathtt{int}; \sigma}{\Gamma \vdash x + y : t; \sigma}\ Constr;\ t = \mathtt{int}$$

(code-add)

$$\frac{\Gamma \vdash u : \langle \mathtt{int} \rangle^{\gamma}; \sigma \quad \Gamma \vdash w : \langle \mathtt{int} \rangle^{\gamma}; \sigma}{\Gamma \vdash u \mathbin{\underline{+}} w : t; \sigma}\ Constr;\ \Gamma \models t \geq \langle \mathtt{int} \rangle^{\gamma}$$

(const)

$$\frac{}{\Gamma \vdash^{L} c : t;\ \sigma}\ Constr;\ \Gamma \models t \geq t^{c}$$

(app0)

$$\frac{\Gamma \vdash^{\gamma} e_1 : t_2 \xrightarrow{\sigma} t_1;\ \sigma \quad \Gamma \vdash^{\gamma} e_2 : t_2;\ \sigma}{\Gamma \vdash^{\gamma} e_1\, e_2 : t;\ \sigma}\ Constr;\ \Gamma \models t \geq t_1$$

(app1)

$$\frac{\Gamma \vdash e_1 : t_2 \to t_1;\ \sigma \quad \Gamma \vdash e_2 : t_2;\ \sigma}{\Gamma \vdash e_1\, e_2 : t;\ \sigma}\ Constr;\ \Gamma \models t \geq t_1$$

(lambda0)

$$\frac{\Gamma,\ x : t_1 \vdash e : t_2;\ \sigma'}{\Gamma \vdash \lambda x.e : t;\ \sigma}\ Constr;\ t = t_1 \xrightarrow{\sigma'} t_2,\ \Gamma \models \sigma \geq \sigma'$$

(lambda1)

$$\frac{\Gamma,\ (u : t_1)^{\gamma} \vdash^{\gamma} e : t_2;\ \sigma}{\Gamma \vdash^{\gamma} \lambda u.e : t;\ \sigma}\ Constr;\ t = t_1 \to t_2$$

(if)

$$\frac{\Gamma \vdash^L e_1 : \texttt{Bool};\ \sigma \quad \Gamma \vdash^L e_2 : t;\ \sigma \quad \Gamma \vdash^L e_3 : t;\ \sigma}{\Gamma \vdash^L \textbf{if}\ e_1 \textbf{then}\ e_2 \textbf{else}\ e_3\ :\ t;\ \sigma}\ Constr;\ (none)$$

(code-lambda)

$$\frac{\Gamma, \gamma' \geq \gamma, x : \langle t_1 \rangle^{\gamma'} \vdash e : \langle t_2 \rangle^{\gamma'};\ \sigma}{\Gamma \vdash \underline{\lambda} x.e\ :\ t;\ \sigma}\ Constr;\ \Gamma \models t \geq \langle t_1 \to t_2 \rangle^{\gamma}$$

(code-let)

$$\frac{\Gamma \vdash e_0 : \langle t' \rangle^{\gamma_0}; \sigma \quad \Gamma, \gamma_1 \geq \gamma_0, x : \langle t' \rangle^{\gamma_1} \vdash e_1 : \langle t' \rangle^{\gamma_1}; \sigma}{\Gamma \vdash \underline{\textbf{let}}\ x = e_0\ \underline{\textbf{in}}\ e_1 : t; \sigma}\ Constr;\ \Gamma \models t \geq \langle t' \rangle^{\gamma_0}$$

(reset0)

$$\frac{\Gamma \vdash e : \langle t' \rangle^{\gamma};\ \langle t' \rangle^{\gamma}, \sigma}{\Gamma \vdash \textbf{reset0}\ e\ :\ t;\ \sigma}\ Constr;\ \Gamma \models t \geq \langle t' \rangle^{\gamma}$$

(shift0)

$$\frac{\Gamma,\ k : \langle t_1 \rangle^{\gamma_1} \overset{\sigma}{\Rightarrow} \langle t_0 \rangle^{\gamma_0} \vdash e : \langle t_0 \rangle^{\gamma_0}; \sigma}{\Gamma \vdash \textbf{shift0}\ k \to e : t\ ;\ t_2, \sigma}\ Constr;\ \Gamma \models t \geq \langle t_1 \rangle^{\gamma_1},\ t_2 = \langle t_0 \rangle^{\gamma_0},\ \Gamma \models \gamma_1 \geq \gamma_0$$

(throw0)

$$\frac{\Gamma,\ k : t' \vdash v : \langle t_1 \rangle^{\gamma'}; \sigma}{\Gamma,\ k : t' \vdash \textbf{throw}\ k\ v : t; \sigma}\ Constr;\ \Gamma \models t \geq \langle t_0 \rangle^{\gamma_2},\ (\Gamma,\ k{:}\ t') \models \gamma_2 \geq \gamma_0,\ t' = \langle t_1 \rangle^{\gamma_1} \overset{\sigma}{\Rightarrow} \langle t_0 \rangle^{\gamma_0},\ \Gamma \models \gamma_1 \cup \gamma_2 \geq \gamma'$$

(code)

$$\frac{\Gamma \vdash^{\gamma} e : t_1;\ \sigma}{\Gamma \vdash \texttt{<}e\texttt{>} : t;\ \sigma}\ Constr;\ t \geq \langle t_1 \rangle^{\gamma}$$

1.1 bug ってそうなところ

(code-let)
(add)
(code-add)
の規則を追加したのですが，あってるのかわからない．

1.2 bug ってそうな箇所

実装は “study/src/metaS0/typeinf.ml” にあります．

```
let rec gen_constr (tyct: tycntxtT list) (lv: lvT) (e: expr) (t: tyT)
                   (sgm: sgmT) (cnstl: constrT list) : constrT list =
  match e,lv with
  | (Var x), L0 ->
    let (t', L0) = lookup_tycntxt x tyct in
    (* print_ty' t'; print_lv' l; *)
    let c1 = CModelGtt(tyct, (t, t')) in
    let new_cnstl = c1 :: cnstl in
    new_cnstl
...
```

のところで，let (t', l) = lookup_tycntxt x tyct in とすると，以下の 1.2.1 の例は動きます．(あってるかはちょっと不明なのですが).

let (t', L0) = lookup_tycntxt x tyct in とすると，動きません．

理由としては，型文脈 tyct の x は レベルが L0 でなく，レベル γ であるからです．パターンマッチとしては， x のレベルは L0 であるべきはずなのですが...

1.2.1 動いてほしい例

```
# pp_cnstrl @@ Let_("x", Code(Int 1), PrimOp2("Add_", Code(Int 1), Var "x"))

# pp_cnstrl @@ R0(Let_("x", Code(Int 1), R0(Let_("y", Code(Int 2),
                  S0("k", Let_("z", Var "x", T0("k", App(Var "k", Var "z")))))))
```