Subtyping / Type inference / Bidirectional typing

2015-10-02

Review

⇒ Bidirectional typing ⇐

(typeof tc e) $\Gamma \vdash e : A$ Given Γ, (tc) e, return A (or #false)

(synth tc e) $\Gamma \vdash e \Rightarrow A$ Given Γ, (tc) e, return A (or #false)

(check tc e B) $\Gamma \vdash e \Leftarrow B$ Given Γ, (tc), e, B, return #true (or #false)

(typeof

Subtyping:

$$\frac{\Gamma \vdash e : A \quad A <: B}{\Gamma \vdash e : B}\ \text{Type-sub}$$

)

$$\frac{\Gamma \vdash e \Rightarrow A \quad A <: B}{\Gamma \vdash e \Leftarrow B}\ \text{Check-sub}$$

```
both = (lam f : Bool→Rat
          (lam p : Bool * Bool
            (pair (app f (fst p))
                  (app f (snd p))))))
```

Type-app

$$\frac{\Gamma \vdash \text{both} : (\text{Bool} \to \text{Rat}) \to (\text{Bool} * \text{Bool}) \to (\text{Rat} * \text{Rat}) \qquad \Gamma \vdash (\text{lam } x\ (\text{ite} \cdots)) : \cdots}{\Gamma \vdash (\text{app both } (\text{lam } x\ (\text{ite } x\ (\text{num } 1)\ (\text{num } 0)))) : \cdots}$$

$x : \ldots, \Gamma \vdash (\text{ite} \cdots) : \ldots$

Bool

Needs to have type Bool→Rat.

So x : Bool.

But typeof can't relay that fact.

$$\frac{\Gamma \vdash e1 \Rightarrow (A1 \to A2) \quad \Gamma \vdash e2 \Leftarrow A1}{\Gamma \vdash (\text{app } e1\ e2) \Rightarrow A2}\ \text{Synth-app} \qquad \frac{x{:}A, \Gamma \vdash e \Leftarrow B}{\Gamma \vdash (\text{lam } x\ e) \Leftarrow A \to B}\ \text{Checks lam}$$

$$\frac{\Gamma \vdash \text{both} \overset{\text{id}}{\Rightarrow} (\text{Bool} \to \text{Rat}) \to (\text{Bool} * \text{Bool}) \to (\text{Rat} * \text{Rat}) \qquad \dfrac{x{:}\text{Bool}, \Gamma \vdash (\text{ite} \cdots) \Leftarrow \text{Rat}}{\Gamma \vdash (\text{lam } x\ (\text{ite} \cdots)) \Leftarrow (\text{Bool} \to \text{Rat})}}{\Gamma \vdash (\text{app } \overset{\text{id}}{\text{both}}\ (\text{lam } x\ (\text{ite } \overset{\text{id}}{x}\ (\text{num } 1)\ (\text{num } 0)))) \Rightarrow}\ \text{Synth-app}$$

The branch for app in synth can relay the knowledge that (lam x ...) should have type Bool→Rat by calling check, passing it (Bool→Rat).