Subtyping

STFbR

For subtyping: create a relation of form $\tau <: \tau$.

Meaning: $\tau_1 <: \tau_2$ means "anywhere τ_2 is expected, τ_1 can be used"

$\leq$

Refl. $$\frac{}{\tau <: \tau}$$

Trans $$\frac{\tau_1 <: \tau_2 \quad \tau_2 <: \tau_3}{\tau_1 <: \tau_3}$$

Record $$\frac{\tau_1 <: \tau_1' \quad \cdots \quad \tau_n <: \tau_n'}{\{l_1 : \tau_1, \ldots, l_n : \tau_n, \ldots, l_m : \tau_m\} <: \{l_1 : \tau_1', \ldots, l_n : \tau_n'\}}$$

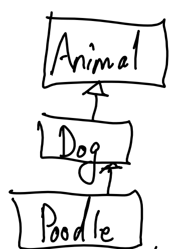
Fun $$\frac{\tau_1' <: \tau_1 \quad \tau_2 <: \tau_2'}{\tau_1 \to \tau_2 <: \tau_1' \to \tau_2'}$$

$\{a : Int, b : Bool\}$

$\{a = 5, b = True\}$

$\{a : Int\}$

Function $x \to x.a + 1$



Animal :> Dog

Dog <: Animal

$\tau \to \tau'$

$\forall v \in \tau.\ \exists v' \in \tau'$. this function takes v and returns v'

request

have f: ~~Dog~~ → Dog
want Animal → Dog (f myCat)
X

have f: Animal → Dog
want Dog → Dog (f myDog) ✓

promise

have Dog → Animal
wants Dog → Dog
X

have Dog → Dog
want Dog → Animal ✓

Dog → Dog <: Dog → Animal

function subtyping is
<u>covariant</u> on output
<u>contravariant</u> on input

as a function becomes a subtype, output moves toward subtype and input moves toward a supertype

Animal ⟶ Poodle <: Dog → Dog <: Poodle → Animal

Animal → Poodle <: Poodle → Animal

Type Inference

Equational constraints to infer types of Fb expressions.

OCaml	fun x → x + 1	: int → int
TFb	Function x : Int → x+1	: Int → Int
EFb	Function x → x+1	: Int → Int

fun x → x : 'a → 'a

EFb is an extension of Fb.

e grammar is the same
v grammar is the same.

$\tau ::= \mathrm{Int} \mid \mathrm{Bool} \mid \tau \to \tau \mid \alpha$ ← type variable

$\alpha ::= \text{'a} \mid \text{'b} \mid \ldots$

$E ::= \{\tau = \tau, \ldots\}$

Doing type inference in EFb

1. Build an inference proof to get a type and some equations
2. Deductively close equation set
3. Check consistency
4. Generate output type

$e \Rightarrow v$	evaluation	(Fb)
$\Gamma \vdash e : \tau$	typing	(TFb)
$\tau <: \tau$	subtyping	(STFbR)
$\Gamma \vdash e : \tau \setminus E$	inference	(EFb)

$\underset{\text{assumptions}}{\Gamma} \vdash \underset{\text{expr}}{e} : \underset{\text{type}}{\tau} \setminus \underset{\text{equations learned}}{E}$

Int $\dfrac{}{\Gamma \vdash \cancel{n} : \mathrm{Int} \setminus \emptyset}$

True $\dfrac{}{\Gamma \vdash \mathrm{True} : \mathrm{Bool} \setminus \emptyset}$

Plus $\dfrac{\Gamma \vdash e_1 : \tau_1 \setminus E_1 \qquad \Gamma \vdash e_2 : \tau_2 \setminus E_2}{\Gamma \vdash e_1 + e_2 : \mathrm{Int} \setminus E_1 \cup E_2 \cup \{\tau_1 = \mathrm{Int}, \tau_2 = \mathrm{Int}\}}$

Equality $\dfrac{\Gamma \vdash e_1 : \tau_1 \setminus E_1 \qquad \Gamma \vdash e_2 : \tau_2 \setminus E_2}{\Gamma \vdash e_1 = e_2 : \mathrm{Bool} \setminus E_1 \cup E_2 \cup \{\tau_1 = \mathrm{Int}, \tau_2 = \mathrm{Int}\}}$

Function $\dfrac{\Gamma, x : \alpha \vdash e : \tau \setminus E \qquad \alpha \text{ fresh}}{\Gamma \vdash \text{Function } x \to e : \alpha \to \tau \setminus E}$

Var $\dfrac{x : \tau \in \Gamma}{\Gamma \vdash x : \tau \setminus \emptyset}$

$$\text{Fun}\ \dfrac{\text{Plus}\ \dfrac{\text{Var}\ \dfrac{}{\{n:\text{'a}\} \vdash n : \text{'a} \setminus \emptyset} \qquad \text{Int}\ \dfrac{}{\{n:\text{'a}\} \vdash 1 : \mathrm{Int} \setminus \emptyset}}{\{n:\text{'a}\} \vdash n+1 : \mathrm{Int} \setminus \{\text{'a} = \mathrm{Int}, \mathrm{Int} = \mathrm{Int}\}}}{\emptyset \vdash \text{Function } n \to n+1 : \text{'a} \to \mathrm{Int} \setminus \{\text{'a} = \mathrm{Int}, \mathrm{Int} = \mathrm{Int}\}}$$

Section 6.6

$$\text{Plus}\ \dfrac{\text{Int}\ \dfrac{}{\emptyset \vdash 1 : \mathrm{Int} \setminus \emptyset} \qquad \text{True}\ \dfrac{}{\emptyset \vdash \mathrm{True} : \mathrm{Bool} \setminus \emptyset}}{\emptyset \vdash 1 + \mathrm{True} : \mathrm{Int} \setminus \{\mathrm{Int} = \mathrm{Int}, \mathrm{Bool} = \mathrm{Int}\}}$$

"1+True has type Int if Int=Int and <u>Bool=Int</u>"