$F^{\omega}_{<:}$

Luca Cardelli
DEC SRC, 130 Lytton Avenue, Palo Alto CA 94301
luca@src.dec.com

Syntax

$K ::=$	Kinds
$\quad Ty$	types
$\quad K \Rightarrow K'$	operator kinds
$A ::=$	Families (types and operators)
$\quad X$	type variable
$\quad Top$	the supertype of all types
$\quad A \rightarrow A'$	function space
$\quad \forall(X<:A::K)A'$	bounded quantification
$\quad \lambda(X::K)A'$	operators
$\quad A(A')$	operator application
$a ::=$	Values
$\quad x$	value variable
$\quad top$	the canonical value of type Top
$\quad \lambda(x:A)a$	function
$\quad a(a')$	application
$\quad \lambda(X<:A::K)a$	family function
$\quad a(A)$	family application

Notation: $\forall(X)A \equiv \forall(X<:Top::Ty)A;\quad \lambda(X)a \equiv \lambda(X<:Top::Ty)a$

Judgments

$\vdash E\ env$	E is a well-formed environment
$E \vdash K\ kind$	K is a kind
$E \vdash A :: K$	A has kind K
$E \vdash A \leftrightarrow A'$	A and A' are equivalent families
$E \vdash A <: A' :: K$	A is a subfamily of A'
$E \vdash a : A$	a has type A
$E \vdash a \leftrightarrow a' : A$	a and a' are equivalent values in A

Environments

(Env ∅)

$$\frac{}{\vdash \varnothing\ env}$$

(Env x:A)

$$\frac{E \vdash A :: Ty \quad x \notin dom(E)}{\vdash E, x{:}A\ env}$$

(Env X<:A::K)

$$\frac{E \vdash A :: K \quad X \notin dom(E)}{\vdash E, X{<:}A{::}K\ env}$$

(Env X::K)

$$\frac{E \vdash K\ kind \quad X \notin dom(E)}{\vdash E, X{::}K\ env}$$

Families

(Fam X<:A::K)

$$\frac{\vdash E, X{<:}A{::}K, E'\ env}{E, X{<:}A{::}K, E' \vdash X :: K}$$

(Fam X::K)

$$\frac{\vdash E, X{::}K, E'\ env}{E, X{::}K, E' \vdash X :: K}$$

(Fam Top)

$$\frac{\vdash E\ env}{E \vdash Top :: Ty}$$

(Fam →)

$$\frac{E \vdash A :: Ty \quad E \vdash B :: Ty}{E \vdash A \rightarrow B :: Ty}$$

(Fam ∀)

$$\frac{E, X{<:}A{::}K \vdash B :: Ty}{E \vdash \forall(X{<:}A{::}K)B :: Ty}$$

(Fam fun)

$$\frac{E, X{::}K \vdash B :: L}{E \vdash \lambda(X{::}K)B : K \Rightarrow L}$$

(Fam appl)

$$\frac{E \vdash B : K \Rightarrow L \quad E \vdash A :: K}{E \vdash B(A) : L}$$

Subfamilies

(Sub Refl)

$$\frac{E \vdash A :: K}{E \vdash A <: A :: K}$$

(Sub Trans)

$$\frac{E \vdash A{<:}B{::}K \quad E \vdash B{<:}C{::}K}{E \vdash A <: C :: K}$$

(Sub X)

$$\frac{\vdash E, X{<:}A{::}K, E'\ env}{E, X{<:}A{::}K, E' \vdash X <: A :: K}$$

(Sub Top)

$$\frac{E \vdash A :: Ty}{E \vdash A <: Top :: Ty}$$

(Sub →)

$$\frac{E \vdash A'{<:}A{::}Ty \quad E \vdash B{<:}B'{::}Ty}{E \vdash A \rightarrow B <: A' \rightarrow B' :: Ty}$$

(Sub ∀)

$$\frac{E \vdash A'{<:}A{::}K \quad E, X{<:}A'{::}K \vdash B{<:}B'{::}Ty}{E \vdash \forall(X{<:}A{::}K)B <: \forall(X{<:}A'{::}K)B' :: Ty}$$

(Sub fun)

$$\frac{E, X{::}K \vdash B{<:}B'{::}L}{E \vdash \lambda(X{::}K)B <: \lambda(X{::}K)B' :: K \Rightarrow L}$$

(Sub appl)

$$\frac{E \vdash B{<:}B'{::}K \Rightarrow L \quad E \vdash A{::}K}{E \vdash B(A) <: B'(A) :: L}$$

Etc.

Natural extensions

We can place bounds on operator parameters:

$$\lambda(X{::}K)B : K{\Rightarrow}L \quad \leadsto \quad \lambda(X{<:}A{::}K)B : \Pi(X{<:}A{::}K)B$$

here kinds become dependend on types, and we must introduce a notion of subkind. At this point we may as well go all the way to powerfamilies:

$$\lambda(X{<:}A{::}K)B : \Pi(X{<:}A{::}K)B \quad \leadsto \quad \lambda(X{::}\mathcal{P}_K(A))B : \Pi(X{::}\mathcal{P}_K(A))B$$

which make the syntax much more uniform.

Orthogonally, we may introduce monotonic and antimonotonic operators:

$$\lambda(X{::}K)B : K{\Rightarrow}L \quad \leadsto \quad \lambda(X{::}K)B : K{\Rightarrow}^{+}L,\ \lambda(X{::}K)B : K{\Rightarrow}^{-}L$$

this produces additional, more symmetrical, rules for (Sub appl) where the operator arguments are assumed to be in subfamily relations.