The Singleton Kind Calculus

February 2, 2017

Abstract

In this section we develop the singleton kind calculus. A singleton kind $S(c)$ is the kind of all constructors that are equivalent to c. The addition of these new kinds will be useful to explain module signatures later on.

1 Syntax

The singleton kind calculus is built on top of souped-up F_ω.

$$\begin{array}{lcl} k & ::= & \texttt{type} \mid k \to k \mid k * k \mid S(c) \mid \Pi\alpha : k.\ k \mid \Sigma\alpha : k.\ k \\ c & ::= & \alpha \mid c \to c \mid \forall\alpha : k.c \mid \lambda\alpha : k.c \mid c\ c \mid \langle c, c\rangle \mid \pi_1\ c \mid \pi_2\ c \\ e & ::= & x \mid \lambda x : c.e \mid e\ e \mid \Lambda\alpha : k.e \mid e[c] \\ \Gamma & ::= & \cdot \mid \Gamma, x : c \mid \Gamma, \alpha : k \end{array}$$

2 Motivation

Consider the following ML signature.

```
sig
    type t
    type 'a u
    type ('a, 'b) v
    type w = int
end
```

The first three types can be assigned kinds in F_ω in a straight forward way.

$$\begin{array}{l} \texttt{t} : \texttt{type} \\ \texttt{u} : \texttt{type} \to \texttt{type} \\ \texttt{v} : \texttt{type} \times \texttt{type} \to \texttt{type} \end{array}$$

But how do we kind `w`? Remember, `int` is not a kind, so it doesn't make sense to say `w : int`. But it's not quite right to say `w : type` either, because

$\texttt{w}$ cannot stand for arbitrary types. We therefore write $\texttt{w} : S(\texttt{int})$: $S(\texttt{int})$ is the kind containing exactly `int` and all those types equivalent to `int`, such as $(\lambda\alpha : \texttt{type}.\ \alpha)\ \texttt{int}$. The other new kind constructs, $\Pi\alpha : k.\ k$ and $\Sigma\alpha : k.\ k$ (which are called dependent function spaces and dependent sums respectively), exist to solve the analogous problem for kinding assignments to polymorphic types in signatures. Specifically, consider this example:

```
sig
    type 'a t = 'a list
end
```

We can say that

$$\texttt{t} : \Pi\alpha : \texttt{type}.\ \mathcal{S}(\alpha)$$

Σ kinds are used to kind types that are assigned to abstract types. In the signature

```
    type t
    type s = t
```

we say that

$$\begin{aligned}&\texttt{t} : \texttt{type}\\ &\texttt{s} : \Sigma\alpha : \texttt{type}.\ \mathcal{S}(\alpha)\end{aligned}$$

3 Definitions

In this section the following judgements will be defined.

Judgement	Description
$\Gamma \vdash k : \texttt{kind}$	k is a kind
$\Gamma \vdash k \equiv k' : \texttt{kind}$	kind equivalence
$\Gamma \vdash k \leq k'$	subkinding
$\Gamma \vdash c : k$	c has kind k
$\Gamma \vdash c \equiv c' : k$	constructor equivalence
$\Gamma \vdash e : \tau$	e has type τ

A complete list would also include the judgement $\Gamma \vdash \tau : \texttt{type}$ but these rules are exactly the same as in F_ω so we will omit them.

3.1 Well-Formed Kinds

3A
$$\frac{}{\Gamma \vdash \tau : \mathtt{kind}}$$

3B
$$\frac{\Gamma \vdash c : \tau}{\Gamma \vdash S(c) : \mathtt{kind}}$$

3C
$$\frac{\Gamma \vdash k_1 : \mathtt{kind} \quad \Gamma, k_1 : \mathtt{kind} \vdash k_2 : \mathtt{kind}}{\Gamma \vdash \Pi\alpha : k_1.\ k_2 : \mathtt{kind}}$$

3D
$$\frac{\Gamma \vdash k_1 : \mathtt{kind} \quad \Gamma, k_1 : \mathtt{kind} \vdash k_2 : \mathtt{kind}}{\Gamma \vdash \Sigma\alpha : k_1.\ k_2 : \mathtt{kind}}$$

3.2 Definitional Equality of Kinds

3E
$$\frac{\Gamma \vdash k : \mathtt{kind}}{\Gamma \vdash k \equiv k : \mathtt{kind}}$$

3F
$$\frac{\Gamma \vdash k_1 \equiv k_2 : \mathtt{kind}}{\Gamma \vdash k_2 \equiv k_1 : \mathtt{kind}}$$

3G
$$\frac{\Gamma \vdash k_1 \equiv k_2 : \mathtt{kind} \quad \Gamma \vdash k_2 \equiv k_3 : \mathtt{kind}}{\Gamma \vdash k_1 \equiv k_3 : \mathtt{kind}}$$

3H
$$\frac{\Gamma \vdash c \equiv c' : \mathtt{type}}{\Gamma \vdash S(c) \equiv S(c') : \mathtt{kind}}$$

3I
$$\frac{\Gamma \vdash k_1 \equiv k_1' : \mathtt{kind} \quad \Gamma, \alpha : k_1 \vdash k_2 \equiv k_2' : \mathtt{kind}}{\Gamma \vdash \Pi\alpha : k_1.\ k_2 \equiv \Pi\alpha : k_1'.\ k_2 : \mathtt{kind}}$$

3J
$$\frac{\Gamma \vdash k_1 \equiv k_1' : \mathtt{kind} \quad \Gamma, \alpha : k_1 \vdash k_2 \equiv k_2' : \mathtt{kind}}{\Gamma \vdash \Sigma\alpha : k_1.\ k_2 \equiv \Sigma\alpha : k_1'.\ k_2 : \mathtt{kind}}$$

3.3 Kind Membership

This judgement defines which constructors belong to a given kind.

3K
$$\frac{\alpha : k \in \Gamma}{\Gamma \vdash \alpha : k}$$

3L
$$\frac{\Gamma \vdash c_1 : \texttt{type} \quad \Gamma \vdash c_2 : \texttt{type}}{\Gamma \vdash c_1 \to c_2 : \texttt{type}}$$

3M
$$\frac{\Gamma \vdash k : \texttt{kind} \quad \Gamma, \alpha : k \vdash c : \texttt{type}}{\Gamma \vdash \forall \alpha : k.\ c : \texttt{type}}$$

3N
$$\frac{\Gamma \vdash k_1 : \texttt{kind} \quad \Gamma, \alpha : k_1 \vdash c : k_2}{\Gamma \vdash \lambda \alpha : k_1.\ c : \Pi \alpha : k_1.\ k_2}$$

3O
$$\frac{\Gamma \vdash c_1 : \Pi \alpha : k.\ k' \quad \Gamma \vdash c_2 : k}{\Gamma \vdash c_1\ c_2 : [c_2/\alpha]k'}$$

3P
$$\frac{\Gamma \vdash c_1 : k_1 \quad \Gamma \vdash c_2 : [c_1/\alpha]k_2 \quad \Gamma, \alpha : k_1 \vdash k_2 : \texttt{kind}}{\Gamma \vdash \langle c_1, c_2 \rangle : \Sigma \alpha : k_1.\ k_2}$$

3Q
$$\frac{\Gamma \vdash c : \Sigma \alpha : k_1.\ k_2}{\Gamma \vdash \pi_1\ c : k_1}$$

3R
$$\frac{\Gamma \vdash c : \Sigma \alpha : k_1.\ k_2}{\Gamma \vdash \pi_2\ c : [\pi_1 c/\alpha]k_2}$$

3S
$$\frac{\Gamma \vdash c : \texttt{type}}{\Gamma \vdash c : S(c)}$$

Notice that even though $\Sigma\alpha : k_1.\ k_2$ is called a dependent sum, it behaves like a product. To avoid the obvious naming confusion here, we will try our best to call $\Sigma\alpha : k_1.\ k_2$ a "dependent sum" and $\Pi\alpha : k_1.\ k_2$ a "dependent function space" or sometimes just a "dependent function" because the former is a bit of a mouthful.

3.4 Subkinding

3T
$$\frac{\Gamma \vdash k : \texttt{kind}}{\Gamma \vdash k \leq k}$$

3U
$$\frac{\Gamma \vdash k_1 \leq k_2 \quad \Gamma \vdash k_2 \leq k_3}{\Gamma \vdash k_1 \leq k_3}$$

3V
$$\frac{\Gamma \vdash c : \texttt{type}}{\Gamma \vdash S(c) \leq \texttt{type}}$$

3W
$$\frac{\Gamma \vdash c \equiv c' : \texttt{type}}{\Gamma \vdash S(c) \leq S(c') : \texttt{type}}$$

3X
$$\frac{\Gamma \vdash k_1' \leq k_1 \quad \Gamma\alpha : k_1 \vdash k_2 : \texttt{kind} \quad \Gamma, \alpha : k_1' \vdash k_2 \leq k_2'}{\Gamma \vdash \Pi\alpha : k_1.\ k_2 \leq \Pi\alpha : k_1'.\ k_2'}$$

3Y
$$\frac{\Gamma \vdash k_1 \leq k_1' \quad \Gamma, \alpha : k_1 \vdash k_2 \leq k_2' \quad \Gamma, \alpha : k_1' \vdash k_2' : \texttt{kind}}{\Gamma \vdash \Sigma\alpha : k_1.\ k_2 \leq \Sigma\alpha : k_1'.\ k_2'}$$

3Z
$$\frac{\Gamma \vdash c : k \quad \Gamma \vdash k \leq k'}{\Gamma \vdash c : k'}$$

3AA
$$\frac{\Gamma \vdash k \equiv k'}{\Gamma \vdash k \leq k'}$$

3.5 Definitional Equality of Constructors

3AB
$$\frac{\Gamma \vdash c : k}{\Gamma \vdash c \equiv c : k}$$

3AC
$$\frac{\Gamma \vdash c_1 \equiv c_2 : k}{\Gamma \vdash c_2 \equiv c_1 : k}$$

3AD
$$\frac{\Gamma \vdash c_2 : k \quad \Gamma, \alpha : k \vdash c_1 : k'}{\Gamma \vdash (\lambda\alpha : k.\ c_1)\ c_2 \equiv [c_2/\alpha]c_1 : [c_2/\alpha]k'}$$

3AE
$$\frac{\Gamma \vdash c_1 : k_1 \quad \Gamma \vdash c_2 : k_2}{\Gamma \vdash \pi_1\langle c_1, c_2\rangle \equiv c_1}$$

3AF
$$\frac{\Gamma \vdash c_1 : k_1 \quad \Gamma \vdash c_2 : k_2}{\Gamma \vdash \pi_2\langle c_1, c_2\rangle \equiv c_2}$$

3AG
$$\frac{\Gamma \vdash c_1 \equiv c_2 : k \quad \Gamma \vdash k \leq k'}{\Gamma \vdash c_1 \equiv c_2 : k'}$$

3AH
$$\frac{\Gamma \vdash c \equiv c' : \texttt{type}}{\Gamma \vdash c \equiv c' : S(c)}$$

3AI
$$\frac{\Gamma \vdash c : S(c')}{\Gamma \vdash c \equiv c' : \texttt{type}}$$

3AJ

$$\frac{\Gamma \vdash k_1 \equiv k_2 : \texttt{kind} \quad \Gamma, \alpha : k_1 \vdash c \equiv c' : k_2}{\Gamma \vdash \lambda\alpha : k_1.\ c \equiv \lambda\alpha : k_1'.\ c' : \Pi\alpha : k_1.\ k_2}$$

3AK

$$\frac{\Gamma \vdash c \equiv c' : \Pi\alpha : k.\ k' \quad \Gamma \vdash c_2 \equiv c_2' : k}{\Gamma \vdash c_1\ c_2 \equiv c_1'\ c_2' : [c_2/\alpha]k'}$$

3AL

$$\frac{\Gamma \vdash c_1 \equiv c_1' : k_1 \quad \Gamma \vdash c_2 \equiv c_2' : [c_1/\alpha]k_2 \quad \Gamma, \alpha : k_1 \vdash k_2 : \texttt{kind}}{\Gamma \vdash \langle c_1, c_2\rangle \equiv \langle c_1', c_2'\rangle : \Sigma\alpha : k_1.\ k_2}$$

3AM

$$\frac{\Gamma \vdash c \equiv c' : \Sigma\alpha : k_1.\ k_2}{\Gamma \vdash \pi_1\ c \equiv \pi_1\ c' : k_1}$$

3AN

$$\frac{\Gamma \vdash c \equiv c' : \Sigma\alpha : k_1.\ k_2}{\Gamma \vdash \pi_2\ c \equiv \pi_2\ c' : [\pi_1\ c/\alpha]k_2}$$

3AN

$$\frac{\Gamma \vdash c_1 \equiv c_1' : \texttt{type} \quad \Gamma \vdash c_2 \equiv c_2' : \texttt{type}}{\Gamma \vdash c_1 \to c_2 \equiv c_1' \to c_2' : \texttt{type}}$$

3AO

$$\frac{\Gamma \vdash k \equiv k' : \texttt{kind} \quad \Gamma, \alpha : k \vdash c \equiv c' : \texttt{type}}{\Gamma \vdash \forall\alpha : k.\ c \equiv \forall\alpha : k'.c' : \texttt{type}}$$

3AP

$$\frac{\Gamma, \alpha : k_1 \vdash c\ \alpha \equiv c'\ \alpha : k_2 \quad \Gamma \vdash c : \Pi\alpha : k_1.\ k_2 \Gamma \vdash c' : \Pi\alpha : k_1.\ k_2''}{\Gamma \vdash c \equiv c' : \Pi\alpha : k_1.\ k_2}$$

3AQ

$$\frac{\Gamma \vdash \pi_1\ c \equiv \pi_1\ c' \quad \Gamma, \alpha : k_1 \vdash k_2 : \texttt{kind} \quad \Gamma \vdash \pi_2\ c \equiv \pi_2\ c' : [\pi_1\ c/\alpha]k_2}{\Gamma \vdash c \equiv c' : \Sigma\alpha : k_1.\ k_2}$$