**1. The interpretation of terms**

Given a lambda calculus and a CCC, we define the interpretation of type expressions, typing contexts and terms of in.

The interpretation of type expression is defined as follows:



The interpretation of typing context is defined by induction on the length of the context:



Both type expressions and typing contexts are interpreted as objects in.

The interpretation of a well-typed term is a morphism from to . It is defined by induction on the proof of the typing judgement:

* ;
* ;
* ;
* ;
* ;
* ;
* where is the  
  ,-function such that and thus is a morphism from to .

**2. Lemma (Substitution)** If and are well-typed terms, then  
.

**Proof**

The proof is carried out by induction on typing derivation.

**Base cases**

(by)

**Inductive steps**

**(1) Projection**

Induction hypothesis:

**(2)**

Induction hypothesis:

(by)

**(3) Application**

Induction hypothesis:

**(4) Lambda abstraction**

Induction hypothesis:

(by Lemma 1)

(by)

Hence,

□

**3. Theorem (Soundness)** Given any well-typed terms and with , then the interpretations of them are same, i.e. , in every CCC.

**Proof**

**(1) -equivalence** If and are well-typed, with, then

Let,, and , then the two terms can be written as and.

According to Lemma 2, we get and .

**(2) -equivalence**

(by)

(by)

(by Substitution Lemma)

**(3) -equivalence**

(by and )

(by)

Hence,

□

**4. Theorem (Completeness)** Given any well-typed terms and , there exists a CCC such that if, then .

**Proof**

The category is generated by:

The objects of are the types over the signature of and the morphisms are equivalence classes of terms. We choose one variable of each type, and define the morphisms from to using terms over the chosen free variables of types and.

We write for the morphism of given by the term, i.e.  
.

The identity morphism for object is

.

The composition of and is given by

.

**(1) is cartesian closed**

The cartesian closed structure of is obtained as follows:

**i)** Terminal object and unique morphism:

The terminal object in is the terminal type in .

The morphism is obtained as follows:

.

Since -term is unique, morphism is unique for every type.

**ii)** Product objects, function with projection morphisms

The product objects in are the product types in .

The function is obtained as follows:

.

The projection morphisms are obtained as follows:

.

CCCs should satisfy and. By using the definitions of and, with composition defined at the beginning, these equations can be obtained:

For any and

Similarly, we get.

For any

**iii)** Function objects, function and morphism

The function objects in are the function types in .

The function is obtained as follow:

.

The morphism is obtained as follow:

.

CCCs should satisfy these two equations: and . By using the definitions of, , , and composition, these equations can be obtained:

For any

Since

Then

For any

Hence, is cartesian closed.

**(2)**

If, then by using Lemma 2, we get  
  
and  
According to (1), and are in the same equivalence class, i.e.  
  
By the -equivalence, and are in the same equivalence class, i.e.

Otherwise, is empty. According to the definition, .

□

**Lemma 1**  where is any -function, contains all free variables of, and .

**Proof**

**Lemma 2** Suppose is an ordered type assignment and is an assignment with product type , then let be the substitution

For any , .

**Proof**

The proof is carried out by induction on term.

The interpretations of and are same, i.e.

**Base case** ()

Then,

**Inductive steps**

**(1) Projections**

Induction hypothesis:

Then,

**(2)**

Induction hypothesis:

Then,

**(3) Application**

Induction hypothesis:

Then,

**(4)** **Abstraction**

Induction hypothesis:

where , and

Then,

.

Hence,

□