**1. The interpretation of terms**

Given and Cartesian closed category, we define the interpretation of type expressions, typing contexts and terms 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 the typing derivation of.

**Base cases**

( )

Hence, .

**Inductive steps:**

**(1) Application**

By the inductive hypothesis,

Then,

( by )

Hence,

**(2) Lambda abstraction**

By the inductive hypothesis,

By lemma,

where .

Then,

( by induction hypothesis )

**3. Theorem (Soundness)** If , then every CCC satisfying also satisfies , i.e. the interpretations of and are the same morphism in every CCC satisfying .

**Proof**

It is proved by induction on equational proofs from .

The base cases include: , , , and .

The inductive steps include: , , and .

If just contains -, - and -equivalence, the soundness can be stated as follows:

Given and , with , then the interpretations of them equal, i.e. , in every CCC satisfying -equivalence.

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

The interpretations of the two typing contexts are same product object, i.e.  
.

**(2) -equivalence**

.

( by )

( by )

( by CCC Substitution Lemma )

Hence, .

**(3) -equivalence**

.

( by and )

( by )

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

**Proof**

This CCC is generated as follows:

The objects of are the lists of types of . The singleton lists are the atom objects while the lists of length greater than 1 are the product objects, i.e.  
  
And morphism from to is defined as follows:

We choose one variable of each type, and define the arrows from to using terms over the chosen free variable of type.

Then the operations in CCC can be defined as follows:

We write for the arrow of given by the term, i.e.  
. Then the operations in CCC can be defined as follows: