Language theory of lambda-calculus with recursive types

Language Theory of a λ–calculus with Recursive Types

by

H.M.M. ten Eikelder

R.H. Mak

88/14

june 1988

Eindhoven University of Technology
Department of Mathematics and Computing Science
P.O. Box 513
5600 MB EINDHOVEN
The Netherlands
All rights reserved
Editors:

0.INTRODUCTION

It is not uncommon to design a programming language by regarding the kind of computations one would like to perform and to decide on a style of notation. Thus one arrives at a syntactic definition of the language which in general contains a large number of constructs and which, for the purpose of expressing ones computations, is usually very satisfying. However, when it comes to assigning a precise meaning to the syntactic constructs thus arrived at, the problems soon become tremendous. Therefore it seems more appropriate to investigate what the proper mathematical abstractions are to model ones computations with and to see in which way they should be manipulated. Thus a carefully chosen (preferably small) number of semantic constructs should dictate the basic syntactic ingredients of a kernel language. Ease of programming can be obtained by adding an additional layer of syntactic sugar to this kernel language. Since the latter is defined in terms of the basic syntactic constructs, it is not hard to define its semantics. Our ultimate goal is to design a language along these lines. Our interest is not so much in the resulting language, however, but rather in the design process itself. As the kernel for our language we have opted for the lambda–calculus, because of its simple nature, extended with a rich type structure, that should allow for instance polymorphism and recursively defined types. There are several approaches known in the literature such as languages with implicit types like ML [HMcQM86] or languages with explicit types as described in [Re85]. In this report we make a start towards the latter in the sense that the language we define does contain recursive types and what is known as a polymorphic let–construct. It does not contain, however, expressions which are 'type–abstractions'. We have chosen this cautious approach, since the semantics of second order lambda calculi with recursive types is not yet well understood, although various results are known [McQPS86,McC79,Me86,Mi71]. Therefore we study this relatively simple case in great detail before we turn our attention towards 'full' polymorphism. Moreover, we have included both strict and non–strict versions of our type constructors. Investigation of their semantic properties will enable us to make the proper choice in a latter stage when we design the actual language.

CHAPTER 0

The structure of this report is as follows. In chapter 1 the language is given and the meaning of its constructs is briefly explained. A comprehensive and formal semantics is given in chapters 4 and 5 for the type expressions and expressions proper respectively. In chapter 2 a type deduction system is given that enables us to keep the type information within expressions to a minimum. Chapter 3 states a set of reduction rules whose soundness is proven in chapter 6. These rules can be viewed as an operational semantics of our language. Finally, in chapter 7, it is shown that a typed version of the Curry fixed point combinator [Ba81,HiSe86] can be defined in the language.

1.SYNTAX OF TYPE EXPRESSIONS AND EXPRESSIONS

The language we consider consists of expressions that contain type information. Its formal syntax is given by two kinds of expressions, type expressions and expressions proper. Let Tvar be a countable infinite set of variables. Elements of Tvar will be called type variables. Type expressions are generated by the following rules.

T1. Texp ::= Ω▪

T2. Texp ::= Tvar▪

T3. Texp ::= $\uparrow$Texp▪

T4.1. Texp ::= (Texp + Texp)▪

T4.2. Texp ::= (Texp $\oplus$ Texp)▪

T5.1. Texp ::= (Texp $\times$ Texp)▪

T5.2. Texp ::= (Texp $\otimes$ Texp)▪

T6.1. Texp ::= (Texp $\longrightarrow$ Texp)▪

T6.2. Texp ::= (Texp $\ominus$ Texp)▪

T7. Texp ::= ν(Λ Tvar | Texp)▪

A formal semantics, which associates a domain (c.p.o.) to every type expression, will be defined in section 4. We now give an informal description of the domains corresponding to type expressions generated by T1 – T7. The type expression Ω corresponds to the one point domain. The symbol $\uparrow$ is used to denote lifting of the domain, i.e. appending a fresh bottom element. Further $+$, $\times$, $\longrightarrow$ correspond to the disjoint sum, cartesian product and function space domain constructors, whereas $\oplus$, $\otimes$, $\ominus$ correspond to their strict versions, i.e. the coalesced sum, smash product and space of strict functions. A type expression of the form $\nu(\Lambda\, t \,|\, te)$ describes a recursively defined type. For instance the type expression $\nu(\Lambda\, t \,|\, (t + t))$ corresponds to a domain D such that D is isomorphic to the disjoint sum of D and D ; the type expression $\nu(\Lambda\, t \,|\, (\uparrow\Omega \oplus t))$ describes the flat domain of natural numbers. Whether an actual programming language should contain all the type constructors above remains to be seen. However, it is precisely the intention of this paper to investigate the properties of the various constructs in order to allow a deliberate choice.

Let Var be a countable infinite set of variables such that Var ∩ Tvar = ϕ . The syntax of expressions is given by the following rules.

E1. Exp ::= (**btm**|Texp)▪

E2. Exp ::= Var▪

E3.1. Exp ::= (**up** Exp)▪

E3.2. Exp ::= (**down** Exp)▪

E4.1.1. Exp ::= (**inl** Exp | Texp)▪

E4.1.2. Exp ::= (**inr** Texp | Exp)▪

E4.1.3. Exp ::= (**sum** Exp Exp)▪

E4.2.1. Exp ::= (**inls** Exp | Texp)▪

E4.2.2. Exp ::= (**inrs** Texp | Exp)▪

E4.2.3. Exp ::= (**sums** Exp Exp)▪

E5.1.1. Exp ::= (**prol** Exp)▪

E5.1.2. Exp ::= (**pror** Exp)▪

E5.1.3. Exp ::= (**prod** Exp Exp)▪

E5.2.1. Exp ::= (**prols** Exp)▪

E5.2.2. Exp ::= (**prors** Exp)▪

E5.2.3. Exp ::= (**prods** Exp Exp)▪

E6.1.1. Exp ::= (**λ** Var:Texp | Exp)▪

E6.1.2. Exp ::= (**appl** Exp Exp)▪

E6.2.1. Exp ::= (**λs** Var:Texp | Exp)▪

E6.2.2. Exp ::= (**appls** Exp Exp)▪

E7.1. Exp ::= (**intro** ν(**Λ** Tvar|Texp) | Exp)▪

E7.2. Exp ::= (**elim** ν(**Λ** Tvar|Texp) | Exp)▪

E8. Exp ::= (**Λ** Tvar|Exp) Texp▪

In chapter 2 we give a type deduction system that defines the well typed expressions. Furthermore it will be shown that every well typed expression has exactly one type (up to α–conversion). In chapter 5 we define the semantics of a well typed expression and show that the value of an expression is an element of the domain corresponding to its type. An operational semantics in terms of reduction rules is given in section 3.

In the rest of this chapter we give an informal description of the expressions introduced above. Let te be a type expression. The expression (btm | te) stands for a nonterminating computation which does not yield any information. The expressions generated by E3 are used in connection with the lifting of domains. In particular the (**up** e) construct is used to postpone reductions inside the expression e (see also chapter 3). The expressions defined by E4.1 are related to the disjoint sum of domains: (**inl** e | te) and (**inr** te | e) denote the injection of e in the left respectively right part of a sum domain. If e1 and e2 denote two functions with the same range, then (**sum** e1 e2) denotes a function whose domain is the disjoint sum of the domains of e1 and e2 and whose range is the common range of e1 and e2 . The expressions defined by E4.2 are the strict versions of those given in E4.1, they correspond to the strict sum of domains ($\oplus$). E5.1 generates expressions which are related to the product of domains. The first two rules correspond to the left and right projection, whereas E5.1.3 corresponds to the pair construction. Again E5.2 gives the strict versions. E6.1 (and E6.2) describe (strict) lambda abstraction and application. To understand E7 consider a recursively defined type expression, for instance $\nu(\Lambda\, t \,|\, t + t)$. The domain D which will be associated to this type expression (see chapter 4) is isomorphic to the disjoint sum of D and D . The two expressions given by E7 are the syntactic representants of these kinds of isomorphism and its inverse. Finally E8 gives the possibility of building a context of type variables which are bound to type expressions.

Next we introduce some notations which will be used frequently in this report. The mapping FV : Exp $\longrightarrow$ Var yields the free variables of an expression. The mapping FTV : Exp $\cup$ Texp $\longrightarrow$ Tvar gives the free type variables of an expression or a type expression. Recursive definitions of FV and FTV can easily be given, but we shall not do so here. In the sequel we shall encounter three kinds of substitution. The substitution of type expressions for type variables can be performed in type expressions and in expressions. The substitution of expressions for variables can only take place in expressions. Apart from the case of (type) expressions with bounded (type) variables the definition of substitution is straightforward. In case of substitution for a type variable in a (type) expression with a bounded type variable or substitution for a variable in an expression with a bounded variable name clashes may occur. In that case the bounded (type) variable is <u>always</u> replaced by the first appropriate free (type) variable. We list the instances where this happens. Let $s,t \in \text{Tvar}$, $x,y \in \text{Var}$, $te,te1,te2 \in \text{Texp}$ and $e,e1,e2 \in \text{Exp}$. Then

- $(\nu(\Lambda\, t \,|\, te1))^{s}_{te2} = \nu(\Lambda\, u \,|\, (te1^{t}_{u})^{s}_{te2})$,

  where u is the first type variable such that $u \not\equiv s$ and $u \notin FTV(te1) \cup FTV(te2)$.
- $((\lambda\, x{:}te \,|\, e1)^{y}_{e2} = (\lambda\, z{:}te \,|\, (e1^{x}_{z})^{y}_{e2})$,

  where z is the first variable such that $z \not\equiv y$ and $z \notin FV(e1) \cup FV(e2)$.
- $((\Lambda\, t \,|\, e)\; te1\,)^{s}_{te2} = ((\Lambda\, u \,|\, (e^{t}_{u})^{s}_{te2})\; te1^{s}_{te2}$,

  where u is the first type variable such that $u \not\equiv s$ and $u \notin FTV(e) \cup FTV(te) \cup FTV(te2)$.

  Here te is the type expression which will be associated to e by the type inference system given in the next chapter (hence substitution is only defined for well–typed expressions).

Note that our definition of substitution implies that bound variables will also be renamed in cases where this is in fact not necessary. The reason for choosing this definition, instead of a more usual one which considers several cases [Ba81], is to reduce the case analysis in the proofs further on. Finally we mention that the symbol $\equiv$ will be used to denote the syntactic equality of (type) expressions, whereas $\equiv_{\alpha}$ will be used for the equality of (type) expressions up to renaming of the bound variables (α–conversion).

2.TYPE INFERENCE

2.1.Introduction.

In this chapter we demonstrate that the kernel language introduced in the previous chapter is an explicitly typed language in the sense of Reynolds[Re85]. That is, given an expression and a sequence of assumptions regarding the free variables and free type variables occurring in that expression it is possible to assert at most one type for that expression. By a **type** we mean a class of type expressions that are equal up to α–conversion. In chapter 4 it is shown that all type expressions in such a class denote the same domain.

2.2.Formal type inference system.

Formula's of the type inference system will be called typings and they are constructed according to the following grammar rules:

I1.	Typing	::= Assumptions ▸ Consequences ▪
I2.1.	Assumptions	::= ▪
I2.2.	Assumptions	::= Assumption Rest ▪
I3.1.	Assumption	::= Type assignment ▪
I3.2.	Assumption	::= Tvar ▪
I4.1.	Rest	::= ▪
I4.2.	Rest	::= ; Assumption Rest ▪
I5.1.	Consequences	::= Consequences , Consequences ▪
I5.2.	Consequences	::= Type assertion ▪
I5.3.	Consequences	::= Texp ▪
I6.	Type assignment	::= Var : Texp ▪
I7.	Type assertion	::= Exp : Texp ▪

For instance, the typing $t;x{:}t \blacktriangleright$ (**inl** x | t) : t+t states that under the assumptions that (there exists a context in which) first of all a type t is introduced and secondly a variable x of type t , one may assert that the expression (**inl** x | t) is of type t+t . As usual we prefix a typing with the symbol $\vdash$ to indicate that it is derivable.

Let A $\in$ Assumptions. The set FTV(A) of free type variables of A is the set of type variables that occur as subassumptions in A (cf. I3.2). Hence for x:t an assumption $t \notin$ FTV(x:t) ! The set FV(A) of free variables of A is the set of variables that occur in any left–hand side of any type assignment in A (cf. I3.1 and I6).

Let C $\in$ Consequences . The set FTV(C) is the set of type variables occurring free in any expression or type expression contained in C (cf. I5.3 and I7). In particular FTV(e:te) = FTV(e) $\cup$ FTV(te) . Hence if x:t is a consequence then $t \in$ FTV(x:t) (cf. above)! Similarly, FV(C) is the set of free variables occurring in any expression contained in C (cf. I7). In particular FV(e:te) = FV(e) .

Let $A, A_1, A_2 \in$ Assumptions ; $C_1, C_2 \in$ Consequences ; $t \in$ Tvar ; tx,te,te1,te2 $\in$ Texp ; x $\in$ Var and e,e1,e2,f,f1,f2 $\in$ Exp. Then the inference rules for type deduction are :

TR1. $$\frac{}{A \blacktriangleright \Omega}$$

TR2. $$\frac{}{A_1;t;A_2 \blacktriangleright t}$$

TR3. $$\frac{A \blacktriangleright te}{A \blacktriangleright \uparrow te}$$

TR4. $$\frac{A \blacktriangleright te1, te2}{\begin{array}{l} A \blacktriangleright te1 + te2 \\ A \blacktriangleright te1 \oplus te2 \end{array}}$$

TR5. $$\frac{A \blacktriangleright te1, te2}{\begin{array}{l} A \blacktriangleright te1 \times te2 \\ A \blacktriangleright te1 \otimes te2 \end{array}}$$

TR6. $$\frac{A \blacktriangleright te1, te2}{\begin{array}{l} A \blacktriangleright te1 \rightarrow te2 \\ A \blacktriangleright te1 \ominus te2 \end{array}}$$

TR7. $$\frac{A;t \blacktriangleright te}{A \blacktriangleright v(\Lambda\ t \mid te)}$$

ER1.1. $$\frac{A \blacktriangleright te}{A \blacktriangleright (\mathbf{btm} \mid te) : te}$$

ER2. $$\frac{A_1 \blacktriangleright tx}{A_1;x{:}tx;A_2 \ \blacktriangleright\ x{:}tx}\ , \qquad \text{provided } x \notin FV(A_2) \text{ and } FTV(tx) \cap FTV(A_2) = \phi$$

ER3.1. $$\frac{A \blacktriangleright e : te}{A \blacktriangleright (\mathrm{up}\ e) : \uparrow te}$$

ER3.2. $$\frac{A \blacktriangleright e : \uparrow te}{A \blacktriangleright (\mathrm{down}\ e) : te}$$

ER4.1. $$\frac{A \blacktriangleright e1 : te1\ ,\ te2}{\begin{array}{l} A \blacktriangleright (\mathrm{inl}\ e1 \mid te2) : te1 + te2 \\ A \blacktriangleright (\mathrm{inls}\ e1 \mid te2) : te1 \oplus te2 \end{array}}$$

ER4.2. $$\frac{A \blacktriangleright te1\ ,\ e2 : te2}{\begin{array}{l} A \blacktriangleright (\mathrm{inr}\ te1 \mid e2) : te1 + te2 \\ A \blacktriangleright (\mathrm{inrs}\ te1 \mid e2) : te1 \oplus te2 \end{array}}$$

ER4.3.1 $$\frac{A \blacktriangleright f1 : te1 \longrightarrow te\ ,\ f2 : te2 \longrightarrow te}{A \blacktriangleright (\mathrm{sum}\ f1\ f2) : (te1 + te2) \longrightarrow te}$$

ER4.3.2 $$\frac{A \blacktriangleright f1 : te1 \ominus te\ ,\ f2 : te2 \ominus te}{A \blacktriangleright (\mathrm{sums}\ f1\ f2) : (te1 \oplus te2) \ominus te}$$

ER5.1. $$\frac{A \blacktriangleright e : te1 \times te2}{\begin{array}{l} A \blacktriangleright (\mathrm{prol}\ e) : te1 \\ A \blacktriangleright (\mathrm{pror}\ e) : te2 \end{array}}$$

ER5.2. $$\frac{A \blacktriangleright e : te1 \otimes te2}{\begin{array}{l} A \blacktriangleright (\mathrm{prols}\ e) : te1 \\ A \blacktriangleright (\mathrm{prors}\ e) : te2 \end{array}}$$

ER5.3. $$\frac{A \blacktriangleright e1 : te1\ ,\ e2 : te2}{\begin{array}{l} A \blacktriangleright (\mathrm{prod}\ e1\ e2) : te1 \times te2 \\ A \blacktriangleright (\mathrm{prods}\ e1\ e2) : te1 \otimes te2 \end{array}}$$

ER6.1. $$\frac{\begin{array}{l} A \blacktriangleright tx\ ,\ te \\ A;x{:}tx \blacktriangleright e : te \end{array}}{\begin{array}{l} A \blacktriangleright (\lambda\ x{:}tx \mid e) : tx \longrightarrow te \\ A \blacktriangleright (\lambda s\ x{:}tx \mid e) : tx \ominus te \end{array}}$$

ER6.2. $$\frac{A \blacktriangleright f : te \longrightarrow te1\ ,\ e : te}{A \blacktriangleright (\mathrm{appl}\ f\ e) : te1}$$

ER6.3. $$\frac{A \blacktriangleright f : te \ominus te1\ ,\ e : te}{A \blacktriangleright (\mathrm{appls}\ f\ e) : te1}$$

ER7.1. $$\frac{A \blacktriangleright e : \nu(\Lambda\ t \,|\, te)}{A \blacktriangleright (\textbf{intro}\ \nu(\Lambda\ t \,|\, te)\ \mid\ e)\ :\ te^{t}_{\nu(\Lambda\ t \,|\, te)}}$$

ER7.2. $$\frac{A \blacktriangleright e : te^{t}_{\nu(\Lambda\ t \,|\, te)}\ ,\ \nu(\Lambda\ t \,|\, te)}{A \blacktriangleright (\text{elim}\ \nu(\Lambda\ t \,|\, te)\ \mid\ e)\ :\ \nu(\Lambda\ t \,|\, te)}$$

ER8. $$\frac{\begin{array}{l} A \blacktriangleright te1 \\ A;t \blacktriangleright e : te \end{array}}{A \blacktriangleright (\Lambda\ t \,|\ e)te1\ :\ te^{t}_{te1}}$$

ER9. $$\frac{A \blacktriangleright e : te1}{A \blacktriangleright e : te2},$$ provided $te1 \equiv_{\alpha} te2$

ER10.1. $$\frac{\begin{array}{l} A \blacktriangleright C_1 \\ A \blacktriangleright C_2 \end{array}}{A \blacktriangleright C_1\ ,\ C_2}$$

ER10.2. $$\frac{A \blacktriangleright C_1\ ,\ C_2}{\begin{array}{l} A \blacktriangleright C_1 \\ A \blacktriangleright C_2 \end{array}}$$

Notice that to each T– and E–rule of chapter 1 there corresponds exactly one inference rule. The additional rule ER9 signifies that we are only interested in type expressions up to α–conversion. The reason for this is that type expressions that are equal up to α–conversion denote the same domain. Rules ER10 are not essential. They merely allow us the notational convenience of typings containing more than one consequence. Therefore we shall leave applications of these rules implicit in the derivation of typings.

Most proofs given below rely on the fact that given a typing we are able to determine the last inference rule of its derivation. In the absence of rule ER9 this last rule would be uniquely identifiable from the structure of the expression. Derivations of typings in which the expressions contain bound type variables, however, can always end with one or more applications of rule ER9 . In order to avoid these trivial but cumbersome details we assume in all proofs, and without loss of generality, that no derivation ends with an application of rule ER9 .

2.3.Explicit typing.

Our type inference system has been designed to ensure that under any given sequence of assumptions each expression has at most one type, which is, if it exists, derivable from the types of its constituting parts (Recall that a type is an equivalence class of type expressions under α–conversion.) In Reynolds[Re85] this property is called explicit typing.

Theorem 2.3.1.[Explicit typing theorem]

Let $A \in$ Assumptions ; te1,te2 $\in$ Texp and $e \in$ Exp . If both $\vdash A \blacktriangleright e : te1$ and $\vdash A \blacktriangleright e : te2$ then $te1 \equiv_{\alpha} te2$.

Sketch of proof. By induction on the structure of expression e . Note that to each of the rules E1.1 thru E8 to construct expressions there corresponds exactly one inference rule that enables us to assert a type for the expressions produced by that rule. Therefore the induction is straightforward.

□

As stated above it is necessary to provide expressions with a certain amount of type information to obtain an explicitly typed language. The need for additional type information in expressions produced by E4.1 , E4.2 and E6.1 is rather obvious (see p.e. [Re85]). The reader may wonder, however, about the necessity of the type information contained in rules E7.1 and E7.2 . Therefore let us assume, for the sake of the argument, that rule E7.2 is simplified to Exp ::= (**elim** Exp) and that inference rule ER7.2 is accordingly modified to

$$\frac{A \blacktriangleright e : te^{t}_{\nu(\Lambda\ t|te)}\ ,\ \nu(\Lambda\ t|te)}{A \blacktriangleright (\textbf{elim}\ e)\ : \nu(\Lambda\ t|te)} \qquad (*)$$

Let $A \in$ Assumptions be such that $\vdash A \blacktriangleright e : t^{t}_{\nu(\Lambda\ t|t)}\ ,\ \nu(\Lambda\ t|t)$. Then we may assert on account of (*) that (elim | e) is of type $\nu(\Lambda\, t|t)$. However, since $t^{t}_{\nu(\Lambda\, t|t)} \equiv_{\alpha}$ $\nu(\Lambda\, t|t)^{s}_{\nu(\Lambda\, s|\nu(\Lambda\, t|t))}$ we may apply rule ER9 before applying (*) and assert that (elim e) has type $\nu(\Lambda\, s|\nu(\Lambda\, t|t))$ as well. Therefore the type information $\nu(\Lambda\, \mathrm{Tvar}|\mathrm{Texp})$ is absolutely essential in rule E7.2 to obtain explicit typing. For reasons of symmetry the same type information has been added to rule E7.1 , although one can show that explicit typing can be obtained without it.

Given an assumption A we define the set WTV(A) (WTE(A)) of well–typed variables (expressions) under A by

$$\mathrm{WTV(A)} = \{\, x \in \mathrm{Var} \mid (\exists\, te \in \mathrm{Texp} \mid \vdash A \blacktriangleright x : te\,)\,\} \tag{2.3.2}$$

$$\mathrm{WTE(A)} = \{\, e \in \mathrm{Exp} \mid (\exists\, te \in \mathrm{Texp} \mid \vdash A \blacktriangleright e : te\,)\,\} \tag{2.3.3}$$

On account of the explicit typing theorem one can also define for each assumption A a function τ_A that assigns to each expression $e \in \mathrm{WTE(A)}$ an arbitrary, but fixed, type expression te such that $\vdash A \blacktriangleright e : te$. We shall take care that whenever τ_A is used, the particular te chosen for $\tau_A(e)$ is irrelevant, i.e. may be replaced by any type expression te1 such that $te1 \equiv_{\alpha} te$.

2.4.Elementary properties.

Before we state the fundamental properties of our type inference system, viz. inference rules for substitution and α–conversion, we first list some elementary properties of typings.

Property 2.4.1.[Introduction of type variables]

For $A \in$ Assumptions and $te \in$ Texp :

$$\vdash A \blacktriangleright te \quad \text{iff} \quad \mathrm{FTV}(te) \subseteq \mathrm{FTV}(A)$$

□

This property expresses that all free type variables of a type expression should be properly introduced.

Property 2.4.2.[Additional inference rules]

The following additional inference rules are derivable from the ones given in section 2.2 :

Rules to extend assumptions

ER11.1. $\dfrac{A \blacktriangleright C}{A;t \blacktriangleright C}$, provided $t \notin FTV(C)$

ER11.2. $\dfrac{A \blacktriangleright tx \;,\; C}{A;x:tx \blacktriangleright C}$, provided $x \notin FV(C)$

Rules to reorder assumptions

ER12.1. $\dfrac{A_1;s;t;A_2 \blacktriangleright C}{A_1;t;s;A_2 \blacktriangleright C}$

ER12.2. $\dfrac{A_1;x:tx;y:ty;A_2 \blacktriangleright C}{A_1;y:ty;x:tx;A_2 \blacktriangleright C}$, provided $x \not\equiv y \vee tx \equiv_\alpha ty$

ER12.3. $\dfrac{A_1;x:tx;t;A_2 \blacktriangleright C}{A_1;t;x:tx;A_2 \blacktriangleright C}$

ER12.4. $\dfrac{A_1;t;x:tx;A_2 \blacktriangleright C}{A_1;x:tx;t;A_2 \blacktriangleright C}$, provided $t \notin FTV(tx)$

□

2.5. Substitution and α–conversion.

As indicated in chapter 1 three kinds of substitution can be performed. For each kind we present a corresponding inference rule. Likewise three kinds of α–conversion can be performed. Three additional inference rules state that each kind of α–conversion leaves the types of expressions invariant. In chapters 4 and 5 we shall demonstrate that α–conversion neither changes the meaning op type expressions nor the meaning of expressions.

Theorem 2.5.1.[Substitution of type expressions for type variables in type expressions]

Let $A_1,A_2 \in$ Assumptions ; $t \in$ Tvar and te,te1 $\in$ Texp . Then the following inference rule can be derived.

$$\text{ER13.}\quad \frac{\begin{array}{l} A_1 \blacktriangleright te1 \\ A_1; t ; A_2 \blacktriangleright te \end{array}}{A_1;{A_2}^{t}_{te1} \blacktriangleright te^{t}_{te1}}, \qquad \text{provided } t \notin FTV(A_2)$$

Proof. By induction on the structure of type expression te . All other cases being trivial we only consider the case $te \equiv \nu(\Lambda\, s \,|\, tf)$.

Assume $\quad \vdash A_1 \blacktriangleright te1$ (*)

$\vdash A_1;t;A_2 \blacktriangleright te$ (**)

$t \notin FTV(A_2)$ (***)

1. Let $te \equiv \nu(\Lambda\, s \,|\, tf)$
2. $\vdash A_1;t;A_2;s \blacktriangleright tf$ [(**),TR7]
3. Let u be the first type variable such that
   $u \not\equiv t \wedge u \notin FTV(tf) \wedge u \notin FTV(te1)$
4. $\vdash A_1;t;A_2;s;u \blacktriangleright tf$ [(2),(3),ER11.1]
5. $\vdash A_1;t;A_2;u \blacktriangleright u$ [TR2]
6. $\vdash A_1;t;A_2;u;s \blacktriangleright tf$ [(4),ER12.1]
7. $\vdash A_1;t;A_2;u \blacktriangleright tf^{s}_{u}$ [(5),(6),IH]
8. $t \notin FTV(A_2;u) = FTV(A_2) \cup \{u\}$ [(***),(3)]
9. $\vdash A_1;(A_2;u)^{t}_{te1} \blacktriangleright (tf^{s}_{u})^{t}_{te1}$ [(*),(7),(8),IH]
10. $\vdash A_1;{A_2}^{t}_{te1};u \blacktriangleright (tf^{s}_{u})^{t}_{te1}$ [(3),(9)]
11. $\vdash A_1;{A_2}^{t}_{te1} \blacktriangleright \nu(\Lambda\, u \,|\, (tf^{s}_{u})^{t}_{te1})$ [(10),TR7]
12. $\vdash A_1;{A_2}^{t}_{te1} \blacktriangleright \nu(\Lambda\, s \,|\, tf)^{t}_{te1}$ [(3),(11),subst.]
13. $\vdash A_1;{A_2}^{t}_{te1} \blacktriangleright te^{t}_{te1}$ [(1),(12)]

□

Theorem 2.5.2.[Substitution of type expressions for type variables in expressions]

Let $A_1;A_2 \in$ Assumptions ; $t \in$ Tvar ; te,te1 $\in$ Texp and $e \in$ Exp. Then the following inference rule can be derived

ER14. $$\frac{\begin{array}{l} A_1 \blacktriangleright \text{te1} \\ A_1;t;A_2 \blacktriangleright e : \text{te} \end{array}}{A_1;A_2{}^{t}_{\text{te1}} \blacktriangleright e^{t}_{\text{te1}} : \text{te}^{t}_{\text{te1}}},$$ provided $FTV(A_1;t) \cap FTV(A_2) = \phi$

Proof. By induction on the structure of expression e . We prove only a few cases. The remaining cases are trivial.

Assume $\vdash A_1 \blacktriangleright \text{te1}$ (*)

$\vdash A_1;t;A_2 \blacktriangleright e : \text{te}$ (**)

$FTV(A_1;t) \cap FTV(A_2) = \phi$ (***)

1.1. Let $e \equiv x$ and $x \notin FV(A_2)$, hence $x \in FV(A_1)$

1.2. Let $A_3,A_4 \in$ Assumptions be such that [(**),ER2]

a) $A_1 = A_3;x:\text{te};A_4$

b) $\vdash A_3 \blacktriangleright \text{te}$

c) $x \notin FV(A_4;t;A_2)$

d) $FTV(\text{te}) \cap FTV(A_4;t;A_2) = \phi$

1.3. $x \notin FV(A_4;A_2{}^{t}_{\text{te1}}) = FV(A_4;t;A_2)$ [(1.2c)]

1.4. $FTV(\text{te}) \cap FTV(A_4;A_2{}^{t}_{\text{te1}}) = \phi$ [(***),(1.2d)]

1.5. $\vdash A_3;x:\text{te};A_4;A_2{}^{t}_{\text{te1}} \blacktriangleright x : \text{te}$ [(1.2b),(1.3),(1.4),ER2]

1.6. $t \notin FTV(\text{te})$ [(1.2d)]

1.7. $\text{te} \equiv_{\alpha} \text{te}^{t}_{\text{te1}}$ [(1.6)]

1.8. $\vdash A_1;A_2{}^{t}_{\text{te1}} \blacktriangleright x : \text{te}$ [(1.2a),(1.5)]

1.9. $\vdash A_1;A_2{}^{t}_{\text{te1}} \blacktriangleright x^{t}_{\text{te1}} : \text{te}^{t}_{\text{te1}}$ [(1.7),(1.8),ER9]

1.10. $\vdash A_1;A_2{}^{t}_{\text{te1}} \blacktriangleright e^{t}_{\text{te1}} : \text{te}^{t}_{\text{te1}}$ [(1.1),(1.9)]

2.1.	Let $e \equiv x$ and $x \in FV(A_2)$	
2.2.	Let $A_3,A_4 \in$ Assumptions be such that	[(**),ER2]
	a) $A_2 = A_3;x{:}te;A_4$	
	b) $\vdash A_1;t;A_3 \blacktriangleright te$	
	c) $x \notin FV(A_4)$	
	d) $FTV(te) \cap FTV(A_4) = \phi$	
2.3.	a) $t \notin FTV(A_3)$	[(***),(2.2a)]
	b) $t \notin FTV(A$	[(***),(2.2a)]
2.4.	$\vdash A_1;A_3{}^{t}_{te1} \blacktriangleright te^{t}_{te1}$	[(*),(2.2b),(2.3),thm.2.5.1]
2.5.	$x \notin FV(A_4{}^{t}_{te1}) = FV(A_4)$	[(2.2c)]
2.6.	$FTV(te^{t}_{te1}) \cap FTV(A_4{}^{t}_{te1})$	
	$= FTV(te^{t}_{te1}) \cap FTV(A_4)$	[(2.3b)]
	$= ((FTV(te) \setminus \{t\}) \cup FTV(te1)) \cap FTV(A_4)$	
	$= FTV(te1) \cap FTV(A_4)$	[(2.2d)]
	$\subseteq FTV(A_1) \cap FTV(A_4)$	[prop.2.4.1]
	$= \phi$	[(***),(2.2a)]
2.7.	$\vdash A_1;A_3{}^{t}_{te1};x{:}te^{t}_{te1};A_4{}^{t}_{te1} \blacktriangleright x : te^{t}_{te1}$	[(2.4),(2.5),(2.6),ER2]
2.8.	$\vdash A_1;A_2{}^{t}_{te1} \blacktriangleright x^{t}_{te1} : te^{t}_{te1}$	[(2.2a),(2.7),subst.]
2.9.	$\vdash A_1;A_2{}^{t}_{te1} \blacktriangleright e^{t}_{te1} : te^{t}_{te1}$	[(2.1),(2.8)]
3.1.	Let $e \equiv (\lambda\, y{:}ty \mid f)$	
3.2.	Let $tf \in$ Texp be such that	[(**),ER6.1]
	a) $\vdash A_1;t;A_2 \blacktriangleright ty,tf$	
	b) $\vdash A_1;t;A_2;y{:}ty \blacktriangleright f : tf$	
	c) $\vdash te \equiv_{\alpha} ty \longrightarrow tf$	
3.3.	$FTV(A_1;t) \cap FTV(A_2;y{:}ty) = \phi$	[(***)]
3.4.	$\vdash A_1;(A_2;y{:}ty)^{t}_{te1} \blacktriangleright f^{t}_{te1} : tf^{t}_{te1}$	[(*),(3.2b),(3.3),IH]

3.5. $\vdash A_1;A2^t_{te1} \blacktriangleright ty^t_{te1}, tf^t_{te1}$ [(*),(***),(3.2a),thm2.5.1]

3.6. $\vdash A_1;A2^t_{te1};y{:}ty^t_{te1} \blacktriangleright f^t_{te1} : tf^t_{te1}$ [(3.4)]

3.7. $\vdash A_1;A2^t_{te1} \blacktriangleright (\lambda\, y{:}ty^t_{te1} \mid f^t_{te1}) : ty^t_{te1} \rightarrow tf^t_{te1}$ [(3.5),(3.6),ER6.1]

3.8. $\vdash A_1;A2^t_{te1} \blacktriangleright e^t_{te1} : te^t_{te1}$ [(3.1),(3.2c),(3.7)]

4.1. Let $e \equiv (\text{elim}\ \nu(\Lambda\, s \mid tf) \mid f)$

4.2. $\vdash A_1;t;A_2 \blacktriangleright f : tf^s_{\nu(\Lambda\, s \mid tf)}, \nu(\Lambda\, s \mid tf)$ [(**),ER7.2]

4.3. $\vdash A_1;A2^t_{te1} \blacktriangleright f^t_{te1} : (tf^s_{\nu(\Lambda\, s \mid tf)})^t_{te1}$ [(*),(4.2),(***),IH]

4.4. Let r be the first type variable such that

$r \not\equiv t \ \wedge\ r \notin FTV(tf) \ \wedge\ r \notin FTV(te1)$

4.5. $\nu(\Lambda\, s \mid tf)^t_{te1} \equiv \nu(\Lambda\, r \mid (tf^s_r)^t_{te1})$ [(4.4),subst]

4.6. $(tf^s_{\nu(\Lambda\, s \mid tf)})^t_{te1} \equiv_\alpha ((tf^s_r)^t_{te1})^r_{\nu(\Lambda\, r \mid (tf^s_r)^t_{te1})}$ [(4.4)]

4.7. $\vdash A_1;A2^t_{te1} \blacktriangleright f^t_{te1} : ((tf^s_r)^t_{te1})^r_{\nu(\Lambda\, r \mid (tf^s_r)^t_{te1})}$ [(4.3),(4.6),ER9]

4.8. $\vdash A_1;A2^t_{te1} \blacktriangleright \nu(\Lambda\, s \mid tf)^t_{te1}$ [(*),(4.2),(***),thm2.5.1]

4.9. $\vdash A_1;A2^t_{te1} \blacktriangleright \nu(\Lambda\, r \mid (tf^s_r)^t_{te1})$ [(4.4),(4.8),subst]

4.10. $\vdash A_1;A2^t_{te1} \blacktriangleright (\text{elim}\ \nu(\Lambda\, r \mid (tf^s_r)^t_{te1}) \mid f^t_{te1}) : \nu(\Lambda\, r \mid (tf^s_r)^t_{te1})$ [(4.7),(4.9),ER7.2]

4.11. $\vdash A_1;A2^t_{te1} \blacktriangleright (\text{elim}\ \nu(\Lambda\, s \mid tf) \mid f)^t_{te1} : \nu(\Lambda\, s \mid tf)^t_{te1}$ [(4.10),subst]

4.12. $te \equiv_\alpha \nu(\Lambda\, s \mid tf)$ [(**),ER7.2]

4.13. $\vdash A_1;A2^t_{te1} \blacktriangleright e^t_{te1} : te^t_{te1}$ [(4.1),(4.11),(4.12)]

5.1. Let $e \equiv (\Lambda\, s \mid f)tf1$

5.2. $\vdash A_1;t;A_2 \blacktriangleright tf1$ [(**),ER8]

5.3. Let $tf \in Texp$ be such that [(**),ER8]

a) $\vdash A_1;t;A_2;s \blacktriangleright f{:}tf$

b) $te \equiv_\alpha tf^s_{tf1}$

5.4. Let r be the first type variable such that

a) $r \notin FTV(f{:}tf)$

b) $r \not\equiv t \;\wedge\; r \notin FTV(te1)$

5.5. $\vdash A_1;t;A_2;s;r \blacktriangleright f{:}tf$ [(5.3a),(5.4a),ER11.1]

5.6. $\vdash A_1;t;A_2;r;s \blacktriangleright f{:}tf$ [(5.5),ER12.1]

5.7. $\vdash A_1;t;A_2;r \blacktriangleright r$ [TR2]

5.8. $\vdash A_1;t;A_2;r \blacktriangleright f^s_r : tf^s_r$ [(5.6),(5.7),IH]

5.9. $t \notin FTV(A_2;r)$ [(***),(5.4b)]

5.10. $\vdash A_1;(A_2;r)^t_{te1} \blacktriangleright (f^s_r)^t_{te1} : (tf^s_r)^t_{te1}$ [(*),(5.8),(5.9),IH]

5.11. $\vdash A_1;A2^t_{te1};r \blacktriangleright (f^s_r)^t_{te1} : (tf^s_r)^t_{te1}$ [(5.10)]

5.12. $\vdash A_1;A2^t_{te1} \blacktriangleright tf1^t_{te1}$ [(*),(5.2),(***),thm2.5.1]

5.13. $\vdash A_1;A2^t_{te1} \blacktriangleright (\Lambda\, r \,|\, (f^s_r)^t_{te1})tf1^t_{te1} : ((tf^s_r)^t_{te1})^r_{tf1^t_{te1}}$ [(5.11),(5.12),E8]

5.14. $((tf^s_r)^t_{te1})^r_{tf1^t_{te1}}$

$\equiv_\alpha\; (tf^s_{tf1})^t_{te1}$

$\equiv_\alpha\; te^t_{te1}$ [(5.3b)]

5.15. $\vdash A_1;A2^t_{te1} \blacktriangleright (\Lambda\, r \,|\, (f^s_r)^t_{te1})tf1^t_{te1} : te^t_{te1}$ [(5.13),(5.14),ER9]

5.16. $\vdash A_1;A2^t_{te1} \blacktriangleright e^t_{te1} : te^t_{te1}$ [(5.1),(5.15)]

□

Theorem 2.5.3.[renaming bound type variables]

Let $A \in$ Assumptions ; $s,t \in$ Tvar ; $te,te1,te2 \in$ Texp and $e \in$ Exp. Then the following inference rules can be derived:

ER15.1. $$\frac{A \blacktriangleright \nu(\Lambda\ t \,|\, te)}{A \blacktriangleright \nu(\Lambda\ s \,|\, te^t_s)},$$ provided $s \notin FTV(te)$

ER15.2. $$\frac{A \blacktriangleright (\Lambda\ t \,|\ e)te1\ : te2}{A \blacktriangleright (\Lambda\ s \,|\ e^t_s)te1 : te2},$$ provided $s \notin FTV(e{:}\tau_{A;t}(e))$

Proof.

Assume $s \notin FTV(te)$ (*)

$\vdash A \blacktriangleright \nu(\Lambda\, t \,|\, te)$ (**)

1. $\vdash A;t \blacktriangleright te$ [(**),TR7]
2. $\vdash A;t;s \blacktriangleright te$ [(*),(1),ER11.1]
3. $\vdash A;s;t \blacktriangleright te$ [(2),ER12.1]
4. $\vdash A;s \blacktriangleright s$ [TR2]
5. $\vdash A;s \blacktriangleright te^{t}_{s}$ [(3),(4),thm2.5.1]
6. $\vdash A \blacktriangleright \nu(\Lambda\, s \,|\, te^{t}_{s})$ [(5),TR7]

Hence rule ER15.1 is derivable.

Assume $s \notin FTV(e{:}te)$ (*)

$\vdash A \blacktriangleright (\Lambda\, t \,|\, e)te1 : te2$ (**)

1. $\vdash A \blacktriangleright te1$ [(**),ER8]
2. Let $te \in Texp$ be such that [(**),ER8]

   a) $\vdash A;t \blacktriangleright e : te$

   b) $te2 \equiv_{\alpha} te^{t}_{te1}$
3. $\vdash A;t;s \blacktriangleright e : te$ [(*),(2a),ER11.1]
4. $\vdash A;s;t \blacktriangleright e : te$ [(3),ER12.1]
5. $\vdash A;s \blacktriangleright s$ [TR2]
6. $\vdash A;s \blacktriangleright e^{t}_{s} : te^{t}_{s}$ [(4),(5),thm2.5.2]
7. $\vdash A \blacktriangleright (\Lambda\, s \,|\, e^{t}_{s})te1 : (te^{t}_{s})^{s}_{te1}$ [(1),(6),ER8]
8. $\vdash A \blacktriangleright (\Lambda\, s \,|\, e^{t}_{s})te1 : te^{t}_{te1}$ [(*),(7),ER9,subst]
9. $\vdash A \blacktriangleright (\Lambda\, s \,|\, e^{t}_{s})te1 : te2$ [(2b),(8),ER9]

Hence rule ER15.2 is derivable.

□

Theorem 2.5.4.[Substitution of expressions for variables in expressions]

Let $A \in$ Assumptions ; te,te1 $\in$ Texp ; $x \in$ Var and e,e1 $\in$ Exp. Then the following inference rule is derivable :

$$\text{ER16.} \quad \frac{\begin{array}{l} A \blacktriangleright e1 : te1 \\ A;x{:}te1 \blacktriangleright e : te \end{array}}{A \blacktriangleright e^{x}_{e1} : te}$$

Proof. By induction on the structure of expression e. We consider only a few cases. The other cases are trivial.

Assume $\vdash A \blacktriangleright e1 : te1$ (*)

$\vdash A;x{:}te1 \blacktriangleright e : te$ (**)

1.1. Let $e \equiv x$

1.2. $\vdash A \blacktriangleright te1$ [(*)]

1.3. $\vdash A;x{:}te1 \blacktriangleright e : te1$ [(1.1),(1.2),ER2]

1.4. $te \equiv_{\alpha} te1$ [(**),(1.3),thm2.3.1]

1.5. $\vdash A \blacktriangleright x^{x}_{e1} : te1$ [(*),subst]

1.6. $\vdash A \blacktriangleright x^{x}_{e1} : te$ [(1.4),(1.5),ER9]

1.7. $\vdash A \blacktriangleright e^{x}_{e1} : te$ [(1.1),(1.6)]

2.1. Let $e \equiv y \ \wedge \ y \not\equiv x$

2.2. Let A_1 , $A_2 \in$ Assumptions be such that [(2.1),(**),ER2]

a) $A = A_1;y{:}te;A_2$

b) $\vdash A_1 \blacktriangleright te$

c) $y \notin FV(A_2;x{:}te1)$

d) $FTV(te) \cap FTV(A_2;x{:}te1) = \phi$

2.3. a) $y \notin FV(A_2)$ [(2.1),(2.2c)]

b) $FTV(te) \cap FTV(A_2) = \phi$ [(2.2d)]

2.4. $\vdash A \blacktriangleright y : te$ [(2.2a),(2.3),ER2]

2.5.	$\vdash A \blacktriangleright y^{x}_{e1} : te$	[(2.1),(2.4)]
2.6.	$\vdash A \blacktriangleright e^{x}_{e1} : te$	[(2.1),(2.4)]
3.1.	Let $e \equiv (\lambda\, y{:}te2 \mid f)$	
3.2.	Let $tf \in Texp$ be such that	[(**),ER6.1]
	a) $\vdash A;x{:}te1 \blacktriangleright te2 , tf$	
	b) $\vdash A;x{:}te1;y{:}te2 \blacktriangleright f : tf$	
	c) $te \equiv_{\alpha} te2 \longrightarrow tf$	
3.3.	Let z be the first variable such that	
	$z \not\equiv x \;\wedge\; z \notin FV(f) \;\wedge\; z \notin FV(e1)$	
3.4.	$\vdash A;x{:}te1;y{:}te2 \blacktriangleright te2$	[(3.2a),ER11,2]
3.5.	$\vdash A;x{:}te1;y{:}te2;z{:}te2 \blacktriangleright f : tf$	[(3.2b),(3.4),(3.3),ER11.2]
3.6.	$\vdash A;z{:}te2;x{:}te1;y{:}te2 \blacktriangleright f : tf$	[(3.5),ER12.2]
3.7.	$\vdash A;x{:}te1;z{:}te2 \blacktriangleright z : te2$	[(3.2a),ER2]
3.8.	$\vdash A;z{:}te2;x{:}te1 \blacktriangleright z : te2$	[(3.3),(3.7),ER12.2]
3.9.	$\vdash A;z{:}te2;x{:}te1 \blacktriangleright f^{y}_{z} : tf$	[(3.6),(3.8),IH]
3.10.	$\vdash A \blacktriangleright te2 , tf$	[(3.2a)]
3.11.	$\vdash A;z{:}te2 \blacktriangleright e1 : te1$	[(*),(3.10),(3.3),ER11.2]
3.12.	$\vdash A;z{:}te2 \blacktriangleright (f^{y}_{z})^{x}_{e1} : tf$	[(3.9),(3.11),IH]
3.13.	$\vdash A \blacktriangleright (\lambda\, z{:}te2 \mid (f^{y}_{z})^{x}_{e1}) : te2 \longrightarrow tf$	[(3.10),(3.12),ER6.1]
3.14.	$\vdash A \blacktriangleright (\lambda\, y{:}te2 \mid f)^{x}_{e1} : te2 \longrightarrow tf$	[(3.3),(3.13)]
3.15.	$\vdash A \blacktriangleright e^{x}_{e1} : te$	[(3.1),(3.2c)(3.14),ER9]
4.1.	Let $e \equiv (\Lambda\, s \mid f)tf1$	
4.2.	$\vdash A;x{:}te1 \blacktriangleright tf1$	[(**),ER8]
4.3.	Let $tf \in Texp$ be such that	[(**),ER8]
	a) $\vdash A;x{:}te1;s \blacktriangleright f : tf$	
	b) $te \equiv_{\alpha} tf^{s}_{tf1}$	

4.4. Assume without loss of generality that [thm2.5.3]

$s \notin FTV(e1:te1)$

4.5. $\vdash A;s \blacktriangleright e1 : te1$ [(*),(4.4),ER11.1]

4.6. $\vdash A;s;x:te1 \blacktriangleright f : tf$ [(4.3a),ER12.3]

4.7. $\vdash A;s \blacktriangleright f^{x}_{e1} : tf$ [(4.5),(4.6),IH]

4.8. $\vdash A \blacktriangleright tf1$ [(4.2)]

4.9. $\vdash A \blacktriangleright (\Lambda\, s \mid f^{x}_{e1})tf1 : tf^{s}_{tf1}$ [(4.7),(4.8),ER8]

4.10. $\vdash A \blacktriangleright e^{x}_{e1} : te$ [(4.1),(4.3b),(4.9),ER9]

□

Theorem 2.5.5.[renaming bound variables]

Let $A \in$ Assumptions ; $te1,te2 \in$ Texp ; $x,y \in$ Var and $e \in$ Exp. Then the following inference rules can be derived:

ER17.1. $$\frac{A \blacktriangleright (\lambda\ x:te1 \mid e) : te2}{A \blacktriangleright (\lambda\ y:te1 \mid e^{x}_{y}) : te2},$$ provided $y \notin FV(e)$

ER17.2. $$\frac{A \blacktriangleright (\lambda s\ x:te1 \mid e) : te2}{A \blacktriangleright (\lambda s\ y:te1 \mid e^{x}_{y}) : te2},$$ provided $y \notin FV(e)$

Proof.

Assume $y \notin FV(e)$ (*)

$\vdash A \blacktriangleright (\lambda\, x:te1 \mid e) : te2$ (**)

1. Let $te \in$ Texp be such that [(**),ER6.1]

a) $\vdash A \blacktriangleright te1$, te

b) $\vdash A;x:te1 \blacktriangleright e : te$

c) $te2 \equiv_{\alpha} te1 \longrightarrow te$

2. $\vdash A;x:te1 \blacktriangleright te1$ [(1a),ER11.2]

3. $\vdash A;y:te1 \blacktriangleright y : te1$ [(1a),ER2]

4. $\vdash$ A;x:te1;y:te1 $\blacktriangleright$ e : te [(2),(1b),ER11.2]

5. $\vdash$ A;y:te1;x:te1 $\blacktriangleright$ e : te [(4),ER12.2]

6. $\vdash$ A;y:te1 $\blacktriangleright$ e^{x}_{y} : te [(3),(5),thm2.5.4]

7. $\vdash$ A $\blacktriangleright$ (λ y:te1 | e^{x}_{y}) : te1 $\longrightarrow$ te [(1a),(6),ER6.1]

8. $\vdash$ A $\blacktriangleright$ (λ y:te1 | e^{x}_{y}) : te2 [(1c),(7),ER9]

Hence rule ER17.1 is derivable. Similarly it can be shown that rule ER17.2 is derivable.

□

CHAPTER 3

3.REDUCTION

3.1. Introduction.

In this chapter a reduction relation » on expressions is defined that provides an operational semantics for our kernel language. We shall present this reduction relation in the form of a formal theory (cf. Hindley and Seldin [HiSe86]). Besides reduction rules that deal with expressions having function types, which are familiar from the lambda calculus, the theory contains reduction rules for expressions having sum, product or recursive types.

In order to present this theory we need the notion of a context. Suppose we take an expression and replace some of its subexpressions by the fresh symbol $. The resulting term is called a context. Actually we think of a context as an expression with some holes in it. The symbol $ merely enables us to give a proper syntactic definition. To that end replace in rules E1 – E8 of chapter 1 the nonterminal Exp by C_and_E and add the rule C_and_E ::= $. Let Exp be the subset of sentences of C_and_E that contain zero occurrences of the symbol $, and let Context be the subset of sentences that contain at least one occurrence of $. Notice that substituting an expression for $ describes the process of filling in the holes of a context.

3.2. The theory of reduction.

The theory of reduction consists of formula's of the form Exp » Exp and the following rules :

(ν) (**btm** | te) » (**btm** | te)

(δ) (**down** (**up** e)) » e

(σ_1) (**appl** (**sum** f1 f2) (**inl** e1 | te2)) » (**appl** f1 e1)

(σ_2) (**appl** (**sum** f1 f2) (**inr** te1 | e2)) » (**appl** f2 e2)

(σ_3) (**appls** (**sums** f1 f2) (**inls** e1 | te2)) » (**appls** f1 e1)

(σ_4) (**appls** (**sums** f1 f2) (**inrs** te1 | e2)) » (**appls** f2 e2)

(σ_5) (sums (λs x:te1 | (appls f (inls x | te2)))
(λs x:te2 | (appls f (inrs te1 | x)))) » f, $x \notin FV(f)$

(π_1) (prol (prod e1 e2)) » e1

(π_2) (pror (prod e1 e2)) » e2

(π_3) (prod (prol e) (pror e)) » e

(π_4) (prols (prods e1 e2)) » e1 , provided e2 in normal form

(π_5) (prors (prods e1 e2)) » e2 , provided e1 in normal form

(π_6) (prods (prols e) (prors e)) » e

(ε_1) (elim ν(Λ t|te) | (intro ν(Λ t|te) | e)) » e

(ε_2) (intro ν(Λ t|te) | (elim ν(Λ t|te) | e)) » e

(β_1) (appl (λ x:tx | e) e1) » e^{x}_{e1}

(β_2) (appls (λs x:tx | e) e1) » e^{x}_{e1} , provided e1 in normal form

(β_3) (Λ t| e)te1 » e^{t}_{te1}

(η_1) (λ x:tx | (appl f x)) » f , $x \notin FV(f)$

(η_2) (λs x:tx | (appls f x)) » f , $x \notin FV(f)$

(ρ) e » e reflexivity

(τ) $$\frac{e1 \gg e2 \qquad e2 \gg e3}{e1 \gg e3}$$ transitivity

(ψ) $$\frac{e1 \gg e2}{c^{\$}_{e1} \gg c^{\$}_{e2}},$$ provided there exist no contexts c1 and c2 such that $c \equiv_\alpha c1^{\$}_{\mathrm{up}\ c2}$

Rule ψ expresses the substitutivity property (or compatibility property as it is called in Barendregt [Ba81]) of » . It states, however, one exception, viz. subexpressions appearing in an up–context can not be reduced. Hence » is the reflexive, transitive and (almost) substitutive closure of the one–step reduction relation defined by rules ν thru η . The left–hand side of any of these rules is called a redex. An expression in which all redices, if any, appear inside an up–context is called a normal form.

Notice that the notions redex and normal form are actually defined by mutual recursion, on account of the constraints in rules π_4 , π_5 and β_2 . In particular **(btm** | te) is not a normal form. This is proper, since it corresponds to a nonterminating computation that yields no information at all. On the other hand, any **up**–expression is in normal form.

Up–expressions can be used to enforce lazy evaluation. Consider the two expressions

(appl (**λ** x:tx | (**inl** x | te2)) e)

and

(appl (**λ** x:↑tx | (**inl** (**down** x) | te2)) (**up** e)

If e » e1 then (**appl** (**λ** x:tx | (**inl** x | te2)) e) » (**inl** e1 | te2) in two distinct ways, viz. applying rule β_1 before rule ψ , which is called lazy evaluation or applying rule ψ and then rule β_1 , which is called eager evaluation. Likewise (**appl** (**λ** x:↑tx | (**inl** (**down** x) | te2)) (**up** e) » (**inl** e1 | te2) , but the order in which the rules are applied has to be first β_1 then δ and finally ψ .

One would expect that reduction does not change the type of an expression. This is indeed the case, if renaming of bound variables is ignored. Of course type expressions that differ only in the names of their bound variables have the same semantics. Hence, if we are a little more liberal and consider a type to be a class of type expressions that are equal up to α–conversion then we can say that types are invariant under reduction.

Theorem 3.2.1.

Let A ∊ Assumptions and e1,e2 ∊ Exp.

If e1 ∊ WTE(A) and e1 » e2

Then e2 ∊ WTE(A) and $\tau_A(e1) \equiv_\alpha \tau_A(e2)$.

Proof. With the exception of the β–rules this follows for each of the remaining rules ν thru η by a straightforward calculation. Rules β_1 and β_2 preserve types on account of theorem 2.5.3. Rule β_3 preserves types on account of theorem 2.5.4.

□

Remark. For reductions $e1 \gg e2$ that do not comprise rule β_3 one can prove that $\tau_A(e1) \equiv \tau_A(e2)$.

□

CHAPTER 4

4.SEMANTICS OF TYPE EXPRESSIONS

4.1.Introduction.

In this chapter we show how a complete partial order (c.p.o.) can be associated to every type expression. The c.p.o.'s corresponding to recursively defined types, i.e. type expressions of the form $\nu(\Lambda\ t \mid te)$, are found using the inverse limit construction. The use of this technique to solve recursive domain equations has been described by Smyth & Plotkin [SP82], Lehmann & Smyth [LS81] and others. A detailed description (for the case of the category of c.p.o.'s with embedding–projection pairs as morphisms) can be found in Bos & Hemerik [BH88]. For general aspects of category theory we refer to Herrlich & Strecker [HeStr73] or Maclane[McL71].

In this section we introduce some notations and conventions. Some elementary properties of the concepts introduced in this section are given in section 4.2. The actual semantics of type expressions is given in section 4.3. We first associate a certain functor with every type expression. The c.p.o. corresponding to a type expression is then found by applying that functor to an object, called the type environment. Finally in section 4.4. some elementary properties of the semantics of type expressions are given.

Let $s,t \in Tvar$. In the sequel we shall use the following notations.

- $C = \underline{CPO}_{PR}$, the category of c.p.o.'s with embedding/projection pairs as morphisms
- $\Pi C = \prod_{t \in Tvar} \underline{CPO}_{PR}$.
- $P_t : \Pi C \to C$, the projection functor on component t.
- If $A \in obj(\Pi C)$, then $A_t = P_t(A)$.
- If $f \in mor(\Pi C)$, then $f_t = P_t(f)$.
- If $A \in obj(\Pi C)$, $B \in obj(C)$, then $A[B/t] \in obj(\Pi C)$ is defined by

$$A[B/t]_s = \begin{cases} A_s & \text{if } s \not\equiv t \\ B & \text{if } s \equiv t \end{cases}$$

If $f \in mor(\Pi C)$, $g \in mor(C)$, then $f[g/t] \in mor(\Pi C)$ is defined by

$$f[g/t]_s = \begin{cases} f_s & \text{if } s \not\equiv t \\ g & \text{if } s \equiv t \end{cases}.$$

– Consider the functors $F : \Pi C \longrightarrow \Pi C$ and $G : \Pi C \longrightarrow C$. Then the functor $F[G/t] : \Pi C \longrightarrow \Pi C$ is defined by

$$P_s \circ F[G/t] = \begin{cases} P_s \circ F & \text{if } s \not\equiv t \\ G & \text{if } s \equiv t \end{cases}.$$

– $Id : \Pi C \longrightarrow \Pi C$, the identity functor.

– $id_A : A \longrightarrow A$, the identity morphism on object A.

– Consider the functor $F : \Pi C \longrightarrow C$. The functor $abstr_t F : \Pi C \longrightarrow (C \longrightarrow C)$ is defined in the following way:

i) For $A \in obj(\Pi C)$ is $abstr_t F(A)$ the object in the category $C \longrightarrow C$ (i.e. the functor $C \longrightarrow C$) defined by

$$abstr_t F(A)\ (B) = F(A[B/t]) \quad \text{for } b \in obj(C) ,$$

$$abstr_t F(A)\ (g) = F(id_A[g/t]) \quad \text{for } g \in mor(C) .$$

ii) For $f \in mor(\Pi C)$ is $abstr_t F(f)$ the morphism in the category $C \longrightarrow C$ (i.e. the natural transformation) defined by

$$(abstr_t F(f))_B = F(f[id_B/t]) \quad \text{for } B \in obj(C)$$

– Suppose D is an arbitrary category. A functor $F: \Pi C \longrightarrow D$ will be called independent of t if

$$F = F \circ Id[G/t] \quad \text{for all functors } G: \Pi C \longrightarrow C.$$

– We shall use the following functors.

$CONST_A : \Pi C \longrightarrow C$, the constant functor corresponding to an object $A \in obj(C)$,

$LIFT : C \longrightarrow C$, the lifting functor,

$DS : C \times C \longrightarrow C$, the disjoint sum functor,

$CP : C \times C \longrightarrow C$, the cartesian product functor,

$FS : C \times C \longrightarrow C$, the function space functor,

$CS : C \times C \longrightarrow C$, the coalesced sum functor,

$SP : C \times C \longrightarrow C$, the smash product functor,

$SF: C \times C \longrightarrow C$, the strict function space functor,

$IFP : [C \longrightarrow C] \longrightarrow C$, the initial fixed point functor.

The formal definition of these functors can be found in Bos & Hemerik [BH88] or Smyth and Plotkin [SP82].

4.2.Elementary properties.

The following properties of the concepts introduced in the preceding section can easily be shown. Let $F,G : \Pi C \longrightarrow C$, $H : C \longrightarrow D$ and $t,u \in Tvar$. Then

- $F = P_t \circ Id[F/t]$, (4.2.1)
- if $t \not\equiv u$ then P_u is independent of t, (4.2.2)
- $abstr_u F$ is independent of u, (4.2.3)
- if F is independent of t, then $abstr_u F$ is independent of t, (4.2.4)
- if F is independent of u, then $abstr_u(F \circ Id[P_u/t]) = abstr_t F$, (4.2.5)
- if G is independent of u and $t \not\equiv u$, then

  $abstr_u(F \circ Id[G/t]) = (abstr_u F) \circ Id[G/t]$. (4.2.6)

4.3. Definition of semantics of type expression.

We first show that with every type expression an ω – continuous functor $\Pi C \longrightarrow C$ can be associated. Define $\mathcal{F}$: Texp $\longrightarrow [\Pi C \longrightarrow C]$ by

–	$\mathcal{F}[\![\Omega]\!]$	$= \mathrm{CONST}_A$, where A is the one–point c.p.o.
–	$\mathcal{F}[\![t]\!]$	$= P_t$,
–	$\mathcal{F}[\![\uparrow te]\!]$	$= \mathrm{LIFT} \circ \mathcal{F}[\![te]\!]$,
–	$\mathcal{F}[\![te1 + te2]\!]$	$= \mathrm{DS} \circ < \mathcal{F}[\![te1]\!] , \mathcal{F}[\![te2]\!] >$,
–	$\mathcal{F}[\![te1 \times te2]\!]$	$= \mathrm{CP} \circ < \mathcal{F}[\![te1]\!] , \mathcal{F}[\![te2]\!] >$,
–	$\mathcal{F}[\![te1 \longrightarrow te2]\!]$	$= \mathrm{FS} \circ < \mathcal{F}[\![te1]\!] , \mathcal{F}[\![te2]\!] >$,
–	$\mathcal{F}[\![te1 \oplus te2]\!]$	$= \mathrm{CS} \circ < \mathcal{F}[\![te1]\!] , \mathcal{F}[\![te2]\!] >$,
–	$\mathcal{F}[\![te1 \otimes te2]\!]$	$= \mathrm{SP} \circ < \mathcal{F}[\![te1]\!] , \mathcal{F}[\![te2]\!] >$,
–	$\mathcal{F}[\![te1 \ominus te2]\!]$	$= \mathrm{SF} \circ < \mathcal{F}[\![te1]\!] , \mathcal{F}[\![te2]\!] >$,
–	$\mathcal{F}[\![\nu(\Lambda\, t \mid te)]\!]$	$= \mathrm{IFP} \circ (\mathrm{abstr}_t \mathcal{F}[\![te]\!])$.

The constant and projection functors are trivially ω – continuous. The ω – continuity of the functors DS , CP , FS , CS , SP and SF follows from the local continuity of the corresponding functors on $\underline{\mathrm{CPO}} \times \underline{\mathrm{CPO}}$ respectively $\underline{\mathrm{CPO}}_\perp \times \underline{\mathrm{CPO}}_\perp$, see for instance Smyth & Plotkin [SP82] or Bos & Hemerik [BH88]. The ω continuity of the functor LIFT follows from the local continuity of the corresponding functor $\underline{\mathrm{CPO}} \longrightarrow \underline{\mathrm{CPO}}_\perp$, see also [SP82] or [BH88]. Further if $F : [\Pi C \longrightarrow C]$, then also $\mathrm{abstr}_t F : [\Pi C \longrightarrow [C \longrightarrow C]]$, see for instance Herrlich & Strecker [HeStr73, th.15.9]. The ω – continuity of the initial fixed point functor IFP is shown in Lehmann & Smyth [LS81]. Now using the property that the composition of two ω – continuous functors is again ω – continuous (see Mac Lane [McL71]), it is easily shown by induction on the structure of te that $\mathcal{F}[\![te]\!]$ is an ω – continuous functor for every type expression te.

Define Tenv = obj(ΠC) . Elements of Tenv will be called type environments. If $\rho \in$ Tenv , then $\rho_t = P_t(\rho)$ is the c.p.o. associated to $t \in$ Tvar by the type environment ρ. The c.p.o.

corresponding to a type expression te in the environment ρ is given by $\mathcal{F}[\![te]\!]\rho$.

4.4.Properties of the type semantics.

We now describe some properties of the semantics of type expressions. Theorem 4.4.4. shows that the functor associated to a type expression te depends only on the type variables which appear freely in te . Hence the c.p.o. which corresponds to te in an environment ρ depends only on the values of ρ on FTV(te) .

Theorem 4.4.1.

Let $te \in$ Texp and $t \in$ Tvar. If $t \notin$ FTV(te) then $\mathcal{F}[\![te]\!]$ is independent of t .

Proof. The theorem is easily proved using induction on the structure of te.

i) te = Ω , then $\mathcal{F}[\![te]\!]$ = $CONST_A$, where A is the one–point c.p.o. Clearly this functor is independent of t.

ii) te = u $\in$ Tvar with u $\not\equiv$ t . Then $\mathcal{F}[\![te]\!] = P_u$, which by property (4.4.2) is independent of t.

iii) te = $\uparrow$te1 , te = te1 + te2 , te = te1 x te2 ,te = te1 $\longrightarrow$ te2 , te = te1 $\oplus$ te2 , te = te1 $\otimes$ te2 and te = te1 $\ominus$ te2 . These cases are easily handled using the induction hypothesis that $\mathcal{F}[\![te1]\!]$ respectively $\mathcal{F}[\![te1]\!]$ and $\mathcal{F}[\![te2]\!]$ are independent of t.

iv) te = $\nu(\Lambda\ u\,|\,te)$. Then $\mathcal{F}[\![\nu(\Lambda\ u\,|\,te)]\!]$ = IFP $\circ$ (abstr$_u\mathcal{F}[\![te]\!]$) . If t $\equiv$ u the result follows from property (4.2.3). If t $\not\equiv$ u then t $\notin$ FTV(te1) and the theorem follows from the induction assumption and property (4.2.4).

□

The next theorem gives the behaviour of $\mathcal{F}[\![te]\!]$ under substitution in te .

Theorem 4.4.2. [substitution in type expressions]

Let te1 , te2 $\in$ Texp and t $\in$ Tvar . Then $\mathcal{F}[\![\mathrm{te1}^{t}_{\mathrm{te2}}]\!] = \mathcal{F}[\![\mathrm{te1}]\!] \circ \mathrm{Id}[\mathcal{F}[\![\mathrm{te2}]\!] \,/\, t]$.

Proof. The proof is done by induction on the structure of te1 .

i) te1 = Ω or te1 = s with s $\in$ Tvar and s $\not\equiv$ t . In these cases t $\notin$ FTV(te1) and the theorem follows from theorem 4.4.1.

ii) te1 = t . A simple calculation yields that

$$\begin{aligned}
&\mathcal{F}[\![t^{t}_{\mathrm{te2}}]\!] \\
=\ &\mathcal{F}[\![\mathrm{te2}]\!] \\
=\ &P_t \circ \mathrm{Id}[\mathcal{F}[\![\mathrm{te2}]\!] \,/\, t] && \text{[property 4.2.1]} \\
=\ &\mathcal{F}[\![t]\!] \circ \mathrm{Id}[\mathcal{F}[\![\mathrm{te2}]\!] \,/\, t] \,.
\end{aligned}$$

iii) te1 = $\uparrow$te . Then we have

$$\begin{aligned}
&\mathcal{F}[\![(\uparrow\mathrm{te})^{t}_{\mathrm{te2}}]\!] \\
=\ &\mathcal{F}[\![\uparrow(\mathrm{te}^{t}_{\mathrm{te2}})]\!] \\
=\ &\mathrm{LIFT} \circ \mathcal{F}[\![\mathrm{te}^{t}_{\mathrm{te2}}]\!] \\
=\ &\mathrm{LIFT} \circ \mathcal{F}[\![\mathrm{te}]\!] \circ \mathrm{Id}[\mathcal{F}[\![\mathrm{te2}]\!] \,/\, t] && \text{[induction hypothesis]} \\
=\ &\mathcal{F}[\![\uparrow\mathrm{te}]\!] \circ \mathrm{Id}[\mathcal{F}[\![\mathrm{te2}]\!] \,/\, t] \,.
\end{aligned}$$

iv) te1 = te3 & te4 where & = +, x, $\longrightarrow$, $\oplus$, $\otimes$, $\ominus$ corresponds to respectively FU = DS, CP, FS, CS, SP, SF. The result follows from the following computation.

$$\begin{aligned}
&\mathcal{F}[\![(\mathrm{te3}\ \&\ \mathrm{te4})^{t}_{\mathrm{te2}}]\!] \\
=\ &\mathcal{F}[\![\mathrm{te3}^{t}_{\mathrm{te2}}\ \&\ \mathrm{te4}^{t}_{\mathrm{te2}}]\!] \\
=\ &\mathrm{FU} \circ < \mathcal{F}[\![\mathrm{te3}^{t}_{\mathrm{te2}}]\!] \,,\, \mathcal{F}[\![\mathrm{te4}^{t}_{\mathrm{te2}}]\!] > \\
=\ &\mathrm{FU} \circ < \mathcal{F}[\![\mathrm{te3}]\!] \circ \mathrm{Id}[\mathcal{F}[\![\mathrm{te2}]\!] \,/\, t] \,,\, \mathcal{F}[\![\mathrm{te4}]\!] \circ \mathrm{Id}[\mathcal{F}[\![\mathrm{te2}]\!] \,/\, t] > && \text{[induction hypothesis]} \\
=\ &\mathrm{FU} \circ < \mathcal{F}[\![\mathrm{te3}]\!] \,,\, \mathcal{F}[\![\mathrm{te4}]\!] > \circ\, \mathrm{Id}[\mathcal{F}[\![\mathrm{te2}]\!] \,/\, t] && [\ <F1 \circ F \,,\, F2 \circ F> = <F1,F2> \circ F\ ] \\
=\ &\mathcal{F}[\![\mathrm{te3}\ \&\ \mathrm{te4}]\!] \circ \mathrm{Id}[\mathcal{F}[\![\mathrm{te2}]\!] \,/\, t]
\end{aligned}$$

v) te1 = $\nu(\Lambda\ s \mid \mathrm{te})$. Let u be the first variable such that u $\not\equiv$ t and u $\notin$ FTV(te) $\cup$ FTV(te2) . The result now follows from the following calculation.

$\mathcal{F}[\![(\nu(\Lambda\ s\,|\,te))^{t}_{te2}]\!]$

$= \mathcal{F}[\![\nu(\Lambda\ u\,|\,(te^{s}_{u})^{t}_{te2}]\!]$ [def. of substitution]

$= \mathrm{IFP} \circ (\mathrm{abstr}_u \mathcal{F}[\![(te^{s}_{u})^{t}_{te2}]\!])$

$= \mathrm{IFP} \circ (\mathrm{abstr}_u(\mathcal{F}[\![te^{s}_{u}]\!] \circ \mathrm{Id}[\mathcal{F}[\![te2]\!]\ /\ t]\ )\ )$ [induction hypothesis]

$= \mathrm{IFP} \circ (\mathrm{abstr}_u \mathcal{F}[\![te^{s}_{u}]\!]\ ) \circ \mathrm{Id}[\mathcal{F}[\![te2]\!]\ /\ t]$ [$\mathcal{F}[\![te2]\!]$ is independent of u , property (4.2.6)]

$= \mathrm{IFP} \circ (\mathrm{abstr}_u(\mathcal{F}[\![te]\!] \circ \mathrm{Id}[P_u/s])\ ) \circ \mathrm{Id}[\mathcal{F}[\![te2]\!]\ /\ t]$ [induction hypothesis, $\mathcal{F}[\![u]\!] = P_u$]

$= \mathrm{IFP} \circ (\mathrm{abstr}_s \mathcal{F}[\![te]\!]) \circ \mathrm{Id}[\mathcal{F}[\![te2]\!]\ /\ t]$ [$\mathcal{F}[\![te]\!]$ is independent of u , property (4.2.5)]

$= \mathcal{F}[\![(\nu(\Lambda\ s\,|\,te)]\!] \circ \mathrm{Id}[\mathcal{F}[\![te2]\!]\ /\ t]$.

□

As a consequence of theorem 4.4.2 we have

$$\mathcal{F}[\![te1^{t}_{te2}]\!]\ \rho\ = \mathcal{F}[\![te1]\!]\ (\rho[\mathcal{F}[\![te2]\!]\rho\ /t]) \tag{4.4.3}$$

for all te1,te2 ∊ Texp , t ∊ Tvar and ρ ∊ Tenv . This relation shows that substitution in a type expressions can be replaced by substitution in the type environment.

As expected, the semantics of a recursively defined type does not depend on the name of the bound variable.

Theorem 4.4.4.

Let te ∊ Texp and t,u ∊ Tvar . If u ∉ FTV(te) , then

$$\mathcal{F}[\![\nu(\Lambda\ t\,|\,te)]\!] = \mathcal{F}[\![\nu(\Lambda\ u\,|\,te^{t}_{u})]\!].$$

Proof. Using the previous theorem this result can be proved by a straightforward calculation.

$\mathcal{F}[\![\nu(\Lambda\ t\,|\,te^{t}_{u})]\!]$

$= \mathrm{IFP} \circ \mathrm{abstr}_u \mathcal{F}[\![te^{t}_{u}]\!]$

$= \mathrm{IFP} \circ \mathrm{abstr}_u(\mathcal{F}[\![te]\!] \circ \mathrm{Id}[P_u/t]\ )$ [theorem 4.4.2.]

$= \mathrm{IFP} \circ \mathrm{abstr}_t \mathcal{F}[\![\mathrm{te}]\!]$ [$\mathcal{F}[\![\mathrm{te}]\!]$ is independent of u , property 4.2.5.)]

$= \mathcal{F}[\![\mathrm{v}(\Lambda\ \mathrm{t} \,|\, \mathrm{te})]\!]$.

□

Finally we mention a technical result which will be used in section 5. From part v) of the proof of theorem 4.4.2. we infer that if $u \not\equiv t$ and $t \notin \mathrm{FTV(te)} \cup \mathrm{FTV(te2)}$, then

$$\mathrm{abstr}_u \mathcal{F}[\![(\mathrm{te}^{s}_{u})^{t}_{\mathrm{te2}}]\!] = (\ \mathrm{abstr}_s \mathcal{F}[\![\mathrm{te}]\!]\ ) \circ \mathrm{Id}[\mathcal{F}[\![\mathrm{te2}]\!] \,/\, t].$$

Hence we see that under the same assumptions

$$(\ \mathrm{abstr}_u \mathcal{F}[\![(\mathrm{te}^{s}_{u})^{t}_{\mathrm{te2}}]\!]\ )\ \rho\ =\ \mathrm{abstr}_s \mathcal{F}[\![\mathrm{te}]\!]\ )\ (\rho[\mathcal{F}[\![\mathrm{te2}]\!]\rho \,/\, t]) \qquad (4.4.5)$$

for every type assignment ρ.

CHAPTER 5

5.SEMANTICS OF EXPRESSIONS

5.1.States.

The value of an expression $e \in \mathrm{WTE}(A)$ depends on the values of the free variables occurring in it. The function that defines these values is called a state. Hence a state maps each free variable of an expression to an element of a specific c.p.o. . Which c.p.o. that is depends on the assumption A and the type environment ρ . Therefore we define for $A \in \mathrm{Assumptions}$ and $\rho \in \mathrm{Tenv}$

$$\mathrm{ST}_{\rho,A} = \Pi \{ \mathcal{T}[\![\tau_A(x)]\!]\rho \mid x \in \mathrm{WTV}(A) \} \tag{5.1.1}$$

i.e. the set of functions σ such that $\sigma(x) \in \mathcal{T}[\![\tau_A(x)]\!]\rho$ for all $x \in \mathrm{WTV}(A)$. Elements of $\mathrm{ST}_{\rho,A}$ are called **states** .

Definition 5.1.2.
Let $A \in \mathrm{Assumptions}$ and $\rho \in \mathrm{Tenv}$. Moreover, let $x \in \mathrm{Var}$ and $tx \in \mathrm{Texp}$ such that $\vdash A \blacktriangleright tx$ and let $d \in \mathcal{T}[\![tx]\!]\rho$. Then for $\sigma \in \mathrm{ST}_{\rho,A}$ we define the function $\sigma[d/x] \in \mathrm{ST}_{\rho,A;x:tx}$ by :

$$\sigma[d/x](y) = \underline{\mathrm{if}}\ y \equiv x \longrightarrow d \ \square\ y \not\equiv x \longrightarrow \sigma(y)\ \underline{\mathrm{fi}}$$

Moreover, for $A_1 \in \mathrm{Assumptions}$ and $\rho_1 \in \mathrm{Tenv}$ such that $\mathrm{WTV}(A_1) \subseteq \mathrm{WTV}(A)$ and $\mathcal{T}[\![\tau_{A_1}(x)]\!]\rho_1 = \mathcal{T}[\![\tau_A(x)]\!]\rho$ for all $x \in \mathrm{WTV}(A_1)$ we define the restriction $\sigma \upharpoonright \mathrm{WTV}(A_1) \in \mathrm{ST}_{\rho_1,A_1}$ by :

$$(\sigma \upharpoonright \mathrm{WTV}(A_1))\ (x) = \sigma(x)$$

Note that if also $\vdash A_1 \blacktriangleright tx$ then

$$\sigma[d/x] \upharpoonright (\mathrm{WTV}(A_1;x:tx)) = (\sigma \upharpoonright \mathrm{WTV}(A_1))\ [d/x] \tag{5.1.3}$$

□

5.2. Semantic mappings

The meaning of an expression e is given by a family of mappings $\mathcal{E} = <\mathcal{E}_{\rho,A} \mid \rho \in$ Tenv , A $\in$ Assumptions> such that for ρ and A the domain of $\mathcal{E}_{\rho,A}$ is WTE(A) and for all expressions e $\in$ WTE(A) we have $\mathcal{E}_{\rho,A}[\![e]\!] \in ST_{\rho,A} \longrightarrow \mathcal{F}[\![\tau_A(e)]\!]\rho$. Hence given a state $\sigma \in ST_{\rho,A}$, $\mathcal{E}_{\rho,A}[\![e]\!]\sigma$ indeed yields a value in the domain $\mathcal{F}[\![\tau_A(e)]\!]\rho$.

Definition 5.2.1.[Semantic mapping $\mathcal{E}_{\rho,A}$]
Let $\rho \in$ Tenv and A $\in$ Assumptions . For all t,tx $\in$ Tvar; te,te1 $\in$ Texp; x $\in$ Var; e,e1,e2,f1,f2 $\in$ Exp and $\sigma \in ST_{\rho,A}$ the mapping $\mathcal{E}_{\rho,A} \in \Pi\ \{ST_{\rho,A} \longrightarrow \mathcal{F}[\![\tau_A(e)]\!]\rho \mid e \in WTE(A)\}$ is defined by :

1. $\mathcal{E}_{\rho,A}[\![(\text{btm} \mid \text{te})]\!]\sigma = \bot_D$
   where $D = \mathcal{F}[\![\text{te}]\!]\rho$

2. $\mathcal{E}_{\rho,A}[\![x]\!]\sigma = \sigma(x)$

3.1. $\mathcal{E}_{\rho,A}[\![(\text{up } e)]\!]\sigma = <0,\mathcal{E}_{\rho,A}[\![e]\!]\sigma>_{\uparrow D}$
   where $D = \mathcal{F}[\![\tau_A(x)]\!]\rho$

3.2. $\mathcal{E}_{\rho,A}[\![(\text{down } e)]\!]\sigma =$

```
    if $\mathcal{E}_{\rho,A}[\![e]\!]\sigma = \bot_{\uparrow D}$ $\longrightarrow \bot_D$
    [] $\mathcal{E}_{\rho,A}[\![e]\!]\sigma = <0,d>_{\uparrow D}$ $\longrightarrow d$
    fi
```

   where $D = \mathcal{F}[\![\tau_A(x)]\!]\rho$

4.1. $\mathcal{E}_{\rho,A}[\![(\text{inl } e1 \mid \text{te2})]\!]\sigma = <1,\mathcal{E}_{\rho,A}[\![e1]\!]\sigma>_{D_1+D_2}$

   $\mathcal{E}_{\rho,A}[\![(\text{inls } e1 \mid \text{te2})]\!]\sigma = <1,\mathcal{E}_{\rho,A}[\![e1]\!]\sigma>_{D_1 \oplus D_2}$
   where $D_1 = \mathcal{F}[\![\tau_A(e1)]\!]\rho$, $D_2 = \mathcal{F}[\![\text{te2}]\!]\rho$

4.2. $\mathcal{E}_{\rho,A}[\![(\text{inr te1} \mid \text{e2})]\!]\sigma = \langle 2,\mathcal{E}_{\rho,A}[\![\text{e2}]\!]\sigma\rangle_{D_1+D_2}$

$\mathcal{E}_{\rho,A}[\![(\text{inrs te1} \mid \text{e2})]\!]\sigma = \langle 2,\mathcal{E}_{\rho,A}[\![\text{e2}]\!]\sigma\rangle_{D_1 \oplus D_2}$

where $D_1 = \mathcal{F}[\![\text{te1}]\!]\rho$, $D_2 = \mathcal{F}[\![\tau_A(\text{e2})]\!]\rho$

note that $\langle 1,\perp_{D_1}\rangle_{D_1 \oplus D_2} = \perp_{D_1 \oplus D_2} = \langle 2,\perp_{D_2}\rangle_{D_1 \oplus D_2}$.

4.3. $\mathcal{E}_{\rho,A}[\![(\text{sum f1 f2})]\!]\sigma =$

```
(λ d ∈ $D_1+D_2$
|if d = $\perp_{D_1+D_2}$            ⟶ $\perp_D$
[] d = $\langle 1,d_1\rangle_{D_1+D_2}$      ⟶ $(\mathcal{E}_{\rho,A}[\![f1]\!]\sigma)(d_1)$
[] d = $\langle 2,d_2\rangle_{D_1+D_2}$      ⟶ $(\mathcal{E}_{\rho,A}[\![f2]\!]\sigma)(d_2)$
fi
)
```

where $D_1 \longrightarrow D = \mathcal{F}[\![\tau_A(\text{f1})]\!]\rho$, $D_2 \longrightarrow D = \mathcal{F}[\![\tau_A(\text{f2})]\!]\rho$

$\mathcal{E}_{\rho,A}[\![(\text{sums f1 f2})]\!]\sigma =$

```
(λ d ∈ $D_1 \oplus D_2$
|if d = $\langle 1,d_1\rangle_{D_1 \oplus D_2}$  ⟶ $(\mathcal{E}_{\rho,A}[\![f1]\!]\sigma)(d_1)$
[] d = $\langle 2,d_2\rangle_{D_1 \oplus D_2}$  ⟶ $(\mathcal{E}_{\rho,A}[\![f2]\!]\sigma)(d_2)$
fi
)
```

where $D_1 \longrightarrow D = \mathcal{F}[\![\tau_A(\text{f1})]\!]\rho$, $D_2 \longrightarrow D = \mathcal{F}[\![\tau_A(\text{f2})]\!]\rho$

5.1. $\mathcal{E}_{\rho,A}[\![(\text{prol e})]\!]\sigma = \pi_1(\mathcal{E}_{\rho,A}[\![\text{e}]\!]\sigma)$

$\mathcal{E}_{\rho,A}[\![(\text{pror e})]\!]\sigma = \pi_2(\mathcal{E}_{\rho,A}[\![\text{e}]\!]\sigma)$

where $\pi_1 = (\lambda \langle d_1,d_2\rangle_{D_1 \times D_2} \in D_1 \times D_2 \mid d_1)$

and $\pi_2 = (\lambda \langle d_1,d_2\rangle_{D_1 \times D_2} \in D_1 \times D_2 \mid d_2)$

and $D_1 \times D_2 = \mathcal{F}[\![\tau_A(\text{e})]\!]\rho$

5.2. $\mathcal{E}_{\rho,A}[\![(\text{prols e})]\!]\sigma = \psi_1(\mathcal{E}_{\rho,A}[\![e]\!]\sigma)$

$\mathcal{E}_{\rho,A}[\![(\text{prors e})]\!]\sigma = \psi_2(\mathcal{E}_{\rho,A}[\![e]\!]\sigma)$

where $\psi_1 = (\lambda <d_1,d_2>_{D_1 \otimes D_2} \in D_1 \otimes D_2 \mid \underline{\text{if}}\ d_2 = \perp_{D_2} \longrightarrow \perp_{D_1} [\,]\ d_2 \neq \perp_{D_2} \longrightarrow d_1\ \underline{\text{fi}})$

and $\psi_2 = (\lambda <d_1,d_2>_{D_1 \otimes D_2} \in D_1 \otimes D_2 \mid \underline{\text{if}}\ d_1 = \perp_{D_1} \longrightarrow \perp_{D_2} [\,]\ d_1 \neq \perp_{D_1} \longrightarrow d_2\ \underline{\text{fi}})$

and $D_1 \otimes D_2 = \mathcal{F}[\![\tau_A(e)]\!]\rho$

note that $<d_1,\perp_{D_2}>_{D_1 \otimes D_2} = \perp_{D_1 \otimes D_2} = <\perp_{D_1},d_2>_{D_1 \otimes D_2}$

5.3. $\mathcal{E}_{\rho,A}[\![(\text{prod e1 e2})]\!]\sigma = <\mathcal{E}_{\rho,A}[\![e1]\!]\sigma,\mathcal{E}_{\rho,A}[\![e2]\!]\sigma>_{D_1 \times D_2}$

$\mathcal{E}_{\rho,A}[\![(\text{prods e1 e2})]\!]\sigma = <\mathcal{E}_{\rho,A}[\![e1]\!]\sigma,\mathcal{E}_{\rho,A}[\![e2]\!]\sigma>_{D_1 \otimes D_2}$

6.1. $\mathcal{E}_{\rho,A}[\![(\lambda\ x{:}tx \mid e)]\!]\sigma = (\lambda\ d \in D \mid \mathcal{E}_{\rho,A_1}[\![e]\!]\sigma[d/x])$

$\mathcal{E}_{\rho,A}[\![(\lambda s\ x{:}tx \mid e)]\!]\sigma =$

```
    (λ d ∈ D
    |if d = $\perp_D$ ⟶ $\perp_E$
    [] d ≠ $\perp_D$ ⟶ $\mathcal{E}_{\rho,A_1}[\![e]\!]\sigma[d/x]$
    fi
    )
```

where $A_1 = A;x{:}tx$, $D = \mathcal{F}[\![tx]\!]\rho$, $E = \mathcal{F}[\![\tau_{A_1}(e)]\!]\rho$

6.2. $\mathcal{E}_{\rho,A}[\![(\text{appl f e})]\!]\sigma = \mathcal{E}_{\rho,A}[\![f]\!]\sigma\ (\mathcal{E}_{\rho,A}[\![e]\!]\sigma)$

6.3. $\mathcal{E}_{\rho,A}[\![(\text{appls f e})]\!]\sigma = \mathcal{E}_{\rho,A}[\![f]\!]\sigma\ (\mathcal{E}_{\rho,A}[\![e]\!]\sigma)$

7. $\mathcal{E}_{\rho,A}[\![(\text{intro}\ \nu(\Lambda\ t \mid te) \mid e)]\!]\sigma = \alpha^R(\mathcal{E}_{\rho,A}[\![e]\!]\sigma)$

$\mathcal{E}_{\rho,A}[\![(\text{elim}\ \nu(\Lambda\ t \mid te) \mid e)]\!]\sigma = \alpha^L(\mathcal{E}_{\rho,A}[\![e]\!]\sigma)$

where $(D,(\alpha^L,\alpha^R))$ is the initial fixed point of the endofunctor $F = (\text{abstr}_t\ \mathcal{F}[\![te]\!])\rho$ on the category $C = \underline{CPO}_{PR}$ obtained by applying the inverse limit construction to the ω – chain $< F^n\perp_C , F^n u \mid 0 \leq n >$ with u the unique morphism from $\perp_C$ to $F(\perp_C)$. Note that $D = \mathcal{F}[\![\nu(\Lambda\ t \mid te)]\!]\rho$, $F(D) = \mathcal{F}[\![te^t_{\nu(\Lambda\ t \mid te)}]\!]\rho$, $\alpha^L \in \text{Hom}(F(D),D)$

and $\alpha^R \in \mathrm{Hom}(D,F(D))$, cf[BH88,SP82]

8. $\mathcal{E}_{\rho,A}[\![(\Lambda\ t|\ e)te1]\!]\sigma = \mathcal{E}_{\rho_1,A_1}[\![e]\!](\sigma\upharpoonright WTV(A_1))$
where $\rho_1 = \rho[\mathcal{F}[\![te1]\!]\rho\ /\ t]$, $A_1 = A;t$

Remark. All clauses of definition 2.5.2 are of the form

$$\mathcal{E}_{\rho,A}[\![e]\!]\sigma = \phi(\ \mathcal{E}_{\rho_1,A_1}[\![e_1]\!]\sigma_1,\ \ ,\mathcal{E}_{\rho_n,A_n}[\![e_n]\!]\sigma_n)\ ,\ 0 \le n$$

where $e_1,\ldots,e_n$ are the constituting subexpressions of e , and ϕ is some function. This is a proper definition iff

— if $e \in WTE(A)$ then $e_i \in WTE(A_i)$, for $1 \le i \le n$

— $\phi : \mathcal{F}[\![\tau_{A_1}(e_1)]\!]\rho_1 \times \ldots \times \mathcal{F}[\![\tau_{A_n}(e_n)]\!]\rho_n \longrightarrow \mathcal{F}[\![\tau_A(e)]\!]\rho$

For all clauses but 7 and 8 this is trivial. For clause 7 we consider the case $(\mathbf{elim}\ \nu(\Lambda\ t|te)\ |\ e)$ only. The case $(\mathbf{intro}\ \nu(\Lambda\ t|te)\ |\ e)$ will then be evident. For all $A \in$ Assumptions such that $(\mathbf{elim}\ \nu(\Lambda\ t|te)\ |\ e) \in WTE(A)$:

(i)
$$\begin{aligned}
&\mathcal{F}[\![\tau_A((\mathbf{elim}\ \nu(\Lambda\ t|te)\ |\ e))]\!]\rho\\
&= \mathcal{F}[\![\nu(\Lambda\ t|te)]\!]\rho\\
&= (IFP \circ (abstr_t\ \mathcal{F}[\![te]\!]))\rho\\
&= IFP((abstr_t\ \mathcal{F}[\![te]\!])\rho)\\
&= IFP(F)
\end{aligned}$$

By rule ER7.2 it follows that $e \in WTE(A)$ and, moreover,

(ii)
$$\begin{aligned}
&\mathcal{F}[\![\tau_A(e)]\!]\rho\\
&= \mathcal{F}[\![te^{t}_{\nu(\Lambda\ t|te)}]\!]\rho\\
&= \mathcal{F}[\![te]\!]\rho[\mathcal{F}[\![\nu(\Lambda\ t|te)]\!]\rho\ /\ t]\\
&= \mathcal{F}[\![te]\!]\rho[IFP(F)\ /\ t]\\
&= ((abstr_t\ \mathcal{F}[\![te]\!])\rho)IFP(F)\\
&= F(IFP(F))
\end{aligned}$$

Since α^L is an embedding from F(IFP(F)) into IFP(F) it follows that clause 7 is a proper definition. From rule ER8 it follows that if $(\Lambda\ t|\ e)te1 \in WTE(A)$ then $e \in WTE(A_1)$. Since the introduction of the rightmost type variable t in A_1 invalidates type assignments for variables in which the type expression depends on t and that occur to the left of it (see rule ER2), it follows that $WTV(A_1) \subseteq WTV(A)$. Moreover, for $x \in WTV(A_1)$ it holds that $\tau_{A_1}(x) = \tau_A(x)$ and that $t \notin FTV(\tau_{A_1}(x))$. Hence

$$\begin{aligned} & \mathcal{T}[\![\tau_{A_1}(x)]\!]\rho_1 \\ = \ & \mathcal{T}[\![\tau_{A_1}(x)]\!]\rho[\mathcal{T}[\![te1]\!]\rho \,/\, t] && \text{[thm.4.4.1]} \\ = \ & \mathcal{T}[\![\tau_{A_1}(x)]\!]\rho \\ = \ & \mathcal{T}[\![\tau_{A}(x)]\!]\rho \end{aligned}$$

and therefore $\sigma \restriction WTV(A_1) \in ST_{\rho_1,A_1}$ is properly defined.

□

In the sequel we shall frequently need to compare the meanings (values) of a single expression under similar assumptions and in similar states. The following property indicates that if these similarities are strong enough the respective values are equal.

Property 5.2.2.

For all $A_1,A_2 \in$ Assumptions ; $\rho \in$ Tenv ; $e \in$ Exp ; $\sigma_1 \in ST_{\rho,A_1}$ and $\sigma_2 \in ST_{\rho,A_2}$:

If $\vdash A_1 \blacktriangleright e : te$

$\vdash A_2 \blacktriangleright e : te$

$\sigma_1 \restriction WTV(A_2) = \sigma_2 \restriction WTV(A_1)$

Then $\mathcal{E}_{\rho,A_1}[\![e]\!]\sigma_1 = \mathcal{E}_{\rho,A_2}[\![e]\!]\sigma_2$

□

5.3. Substitution and α–conversion.

In order to prove the soundness of the β–reduction rules (see chapter 6) we have to determine the meaning of expressions containing substitutions. For each of the two kinds of substitutions in expressions (see chapter 1) we present a substitution theorem.

Theorem 5.3.1.[Modification of type environment]

For all $A \in$ Assumptions ; $\rho \in$ Tenv ; $t \in$ Tvar ; $te \in$ Texp ; $e \in$ Exp ; $D \in \mathrm{Obj}(\underline{\mathrm{CPO}}_{\mathrm{PR}})$ and $\sigma \in ST_{\rho,A}$:

If $\quad \vdash A \blacktriangleright e:te$ (*)

$\quad t \notin FTV(e:te)$ (**)

Then $\quad \mathcal{E}_{\rho,A}[\![e]\!]\sigma = \mathcal{E}_{\rho_1,A_1}[\![e]\!]\sigma_1$

where $\quad \rho_1 = \rho[D/t]$, $A_1 = A;t$ and $\sigma_1 = \sigma \restriction WTV(A_1)$

Proof. By induction on the structure of expression e . We prove only a limited number of difficult cases. Assume (*) and (**).

1.1. Let $e \equiv x$

1.2. $x \in WTV(A_1)$ [(*),(**),ER11.1]

1.3. $\mathcal{E}_{\rho,A}[\![e]\!]\sigma = \sigma(x) = \sigma_1(x) = \mathcal{E}_{\rho_1,A_1}[\![x]\!]\sigma_1$ [(1.2),def.$\mathcal{E}$]

2.1. Let $e \equiv (\lambda\, y:ty \mid f)$

2.2. Let $d \in \mathcal{F}[\![ty]\!]\rho$. Moreover, let $A_2 \in$ Assumptions and $\sigma_2 \in ST_{\rho,A_2}$ be such that

$A_2 = A;y:ty \qquad \sigma_2 = \sigma[d/y]$

2.3. Let $tf \in$ Texp be such that [(*),(2.2),ER6.1]

a) $\vdash A \blacktriangleright ty , tf$

b) $\vdash A_2 \blacktriangleright f : tf$

c) $te \equiv_\alpha ty \longrightarrow tf$

2.4. $FTV(e:te) = FTV(ty) \cup FTV(f) \cup FTV(tf)$ [(2.1),(2.3c)]

2.5. a) $t \notin FTV(ty)$ [(**),(2.4)]

b) $t \notin FTV(f{:}tf)$

2.6. $\vdash A_2;t \blacktriangleright f : tf$ [(2.3b),(2.5b),ER11.1]

2.7. $\vdash A_1;y{:}ty \blacktriangleright f : tf$ [(2.6),ER12.3]

2.8. $WTV(A_2;t) = WTV(A_1;y{:}ty)$ [ER12.3]

2.9. $(\sigma_2 \upharpoonright WTV(A_2;t)) \upharpoonright WTV(A_1;y{:}ty)$

$= (\sigma[d/y] \upharpoonright WTV(A_2;t)) \upharpoonright WTV(A_1;y{:}ty)$

$= (\sigma[d/y] \upharpoonright (WTV(A_1;y{:}ty)) \upharpoonright WTV(A_2;t)$ [(2.8)]

$= (\sigma \upharpoonright WTV(A_1))[d/y] \upharpoonright WTV(A_2;t)$ [(5.1.3)]

$= \sigma_1[d/y] \upharpoonright WTV(A_2;t)$

2.10. $\mathcal{E}_{\rho,A}[\![(\lambda\ y{:}ty \mid f)]\!]\sigma$

$= (\lambda\ d \in \mathcal{F}[\![ty]\!]\rho \mid \mathcal{E}_{\rho,A_2}[\![f]\!]\sigma_2\ )$ [def.$\mathcal{E}$]

$= (\lambda\ d \in \mathcal{F}[\![ty]\!]\rho \mid \mathcal{E}_{\rho_1,A_2;t}[\![f]\!]\sigma_2 \upharpoonright WTV(A_2;t)\ )$ [(2.3b),(2.5b),IH]

$= (\lambda\ d \in \mathcal{F}[\![ty]\!]\rho \mid \mathcal{E}_{\rho_1,A_1;y:ty}[\![f]\!]\sigma_1[d/y]\ )$ [(2.6),(2.7),(2.9),prop.5.2.2]

$= \mathcal{E}_{\rho_1,A_1}[\![(\lambda\ y{:}ty \mid f)]\!]\sigma_1$ [def.$\mathcal{E}$]

3.1. Let $e \equiv (\mathbf{elim}\ \nu(\Lambda\ s \mid tf) \mid f)$

3.2. a) $\vdash A \blacktriangleright f : tf^{s}_{\nu(\Lambda\ s \mid tf)}$ [(*),ER7.2]

b) $te \equiv_{\alpha} \nu(\Lambda\ s \mid tf)$

3.3 $FTV(f{:}tf^{s}_{\nu(\Lambda\ s \mid tf)})$

$= FTV(f) \cup (FTV(tf) \setminus \{s\}) \cup FTV(\nu(\Lambda\ s \mid tf))$

$= FTV(f) \cup FTV(\nu(\Lambda\ s \mid tf))$

$= FTV((\mathbf{elim}\ \nu(\Lambda\ s \mid tf) \mid f) : \nu(\Lambda\ s \mid tf))$

$= FTV(e{:}te)$

3.4. $t \notin FTV(f{:}tf^{s}_{\nu(\Lambda\ s \mid tf)}) \wedge t \notin \nu(\Lambda\ s \mid tf)$ [(**),(3.2),(3.3)]

3.5. Let $(A,(\alpha^L,\alpha^R))$ be the unique IFP resulting from the inverse limit construction with functor $(abstr_s\ \mathcal{F}[\![(te)]\!])\rho$

3.6. Let $(B,(\beta^L,\beta^R))$ be the unique IFP resulting from the inverse limit construction with functor $(\mathrm{abstr}_s\ \mathcal{F}[\![(te)]\!])\rho[D/t]$

3.7. Since $t \notin FTV(\nu(\Lambda\ s\,|\,tf))$, by (3.4), it follows that the functor $(\mathrm{abstr}_s\ \mathcal{F}[\![(tf)]\!])\rho$ is independent of t , i.e. $(\mathrm{abstr}_s\ \mathcal{F}[\![(tf)]\!])\rho = (\mathrm{abstr}_s\ \mathcal{F}[\![(tf)]\!])\rho[D/t]$

3.8. $\alpha^L = \beta^L$ [(3.5),(3.6),(3.7)]

3.9. $\mathcal{E}_{\rho,A}[\![(\mathrm{elim}\ \nu(\Lambda\ s\,|\,tf)\ \mid\ f)]\!]\sigma$

$= \alpha^L\ (\mathcal{E}_{\rho,A}[\![f]\!]\sigma)$ [def.$\mathcal{E}$]

$= \beta^L\ (\mathcal{E}_{\rho,A}[\![f]\!]\sigma)$ [(3.8)]

$= \beta^L\ (\mathcal{E}_{\rho_1,A_1}[\![f]\!]\sigma_1)$ [(3.2),(3.4),IH]

$= \mathcal{E}_{\rho_1,A_1}[\![(\mathrm{elim}\ \nu(\Lambda\ s\,|\,tf)\ \mid\ f)]\!]\sigma_1$ [def.$\mathcal{E}$]

4.1. Let $e \equiv (\Lambda\ s|\ f)tf1\ \wedge\ s \not\equiv t$

4.2. Let $A_2 \in$ Assumptions ; $\rho_2 \in$ Tenv and $\sigma_2 \in ST_{\rho_2,A_2}$ be such that

$A_2 = A;s$ $\qquad \rho_2 = \rho[\mathcal{F}[\![tf1]\!]\rho\ /\ s]$ $\qquad \sigma_2 = \sigma \restriction WTV(A_2)$

4.3. Let $tf \in$ Texp be such that [(*),ER8]

a) $\vdash A_2 \blacktriangleright f : tf$

b) $te \equiv_\alpha tf^s_{tf1}$

4.4. $FTV(e{:}te)$

$= FTV(e)\ \cup\ FTV(tf^s_{tf1})$

$= (FTV(f) \setminus \{s\})\ \cup\ FTV(tf1)\ \cup\ (FTV(tf)\setminus\{s\})$

$= (FTV(f{:}tf) \setminus \{s\})\ \cup\ FTV(tf1)$

4.5. $t \notin FTV(f{:}tf)$ [(**),(4.1),(4.4)]

4.6. $\rho_2[D/t] = \rho_1[\mathcal{F}[\![tf1]\!]\rho\ /\ s]$ [$s\not\equiv t$,def.ρ_1,ρ_2]

4.7. $\vdash A_2;t \blacktriangleright f : tf$ [(4.3a),(4.5),ER11.1]

4.8. $\vdash A_1;s \blacktriangleright f : tf$ [(4.7),ER12.1]

4.9. Since $WTV(A_2;t) = WTV(A_1;s)$ it follows that

$(\sigma_2 \restriction WTV(A_2;t)) \restriction WTV(A_1;s)\ =\ (\sigma_1 \restriction WTV(A_1;s)) \restriction WTV(A_2;t)$

4.10. $\mathcal{E}_{\rho,A}[\![(\Lambda\, s|\ f)tf1]\!]\sigma$

$= \mathcal{E}_{\rho_2,A_2}[\![f]\!]\sigma_2$ [def.$\mathcal{E}$]

$= \mathcal{E}_{\rho_2[D/t],A_2;t}[\![f]\!]\sigma_2{\restriction}WTV(A_2;t)$ [(4.3a),(4.5),IH]

$= \mathcal{E}_{\rho_1[\mathcal{F}[\![tf1]\!]\rho\ /\ s],A_2;t}[\![f]\!]\sigma_2{\restriction}WTV(A_2;t)$ [(4.6)]

$= \mathcal{E}_{\rho_1[\mathcal{F}[\![tf1]\!]\rho\ /\ s],A_1;s}[\![f]\!]\sigma_1{\restriction}WTV(A_1;s)$ [(4.7),(4.8),(4.9),prop.5.2.2]

$= \mathcal{E}_{\rho_1,A_1}[\![(\Lambda\, s|\ f)tf1]\!]\sigma_1$ [def.$\mathcal{E}$]

5.1. Let $e \equiv (\Lambda\, t|\ f)tf1$

5.2. Let $tf \in Texp$ be such that [(*),ER8.1]

a) $\vdash A_1 \blacktriangleright f : tf$

b) $te \equiv_\alpha tf^{t}_{tf1}$

5.3. $\vdash A_1;t \blacktriangleright f : tf$ [(5.2a)]

5.4. Since $WTV(A_1) = WTV(A_1;t)$ it follows that

$\sigma_1 \restriction WTV(A_1;t) = (\sigma_1 \restriction WTV(A_1;t)) \restriction WTV(A_1)$

5.5. $t \notin FTV(tf1)$ [(**)]

5.6. $\rho[\mathcal{F}[\![tf1]\!]\rho\ /\ t]$

$= \rho[D/t][\mathcal{F}[\![tf1]\!]\rho\ /\ t]$

$= \rho_1[\mathcal{F}[\![tf1]\!]\rho\ /\ t]$

$= \rho_1[\mathcal{F}[\![tf1]\!]\rho[D/t]\ /\ t]$ [(5.5),thm4.4.1]

$= \rho_1[\mathcal{F}[\![tf1]\!]\rho_1\ /\ t]$

5.7. $\mathcal{E}_{\rho,A}[\![(\Lambda\, t|\ f)tf1]\!]\sigma$

$= \mathcal{E}_{\rho[\mathcal{F}[\![tf1]\!]\rho\ /\ t],A_1}[\![f]\!]\sigma_1$ [def.$\mathcal{E}$]

$= \mathcal{E}_{\rho_1[\mathcal{F}[\![tf1]\!]\rho_1\ /\ t],A_1}[\![f]\!]\sigma_1$ [(5.6)]

$= \mathcal{E}_{\rho_1[\mathcal{F}[\![tf1]\!]\rho_1\ /\ t],A_1;t}[\![f]\!]\sigma_1{\restriction}WTV(A_1;t)$ [(5.2a),(5.3),(5.4),prop.5.2.2]

$= \mathcal{E}_{\rho_1,A_1}[\![(\Lambda\, t|\ f)tf1]\!]\sigma_1$ [def.$\mathcal{E}$]

□

Theorem 5.3.2.[Substitution of type expressions for type variables in expressions]

For all $\rho \in$ Tenv ; $A_1,A_2 \in$ Assumptions ; $t \in$ Tvar ; te,te1 $\in$ Texp ; $e \in$ Exp and $\sigma \in ST_{\rho,A_1;A_2{}^t_{te1}}$:

If $\vdash A_1 \blacktriangleright te1$ (*)

$\vdash A_1;t;A_2 \blacktriangleright e{:}te$ (**)

$FTV(A_1;t) \cap FTV(A_2) = \phi$ (***)

Then $\mathcal{E}_{\rho,A_1;A_2{}^t_{te1}}[\![e^t_{te1}]\!]\sigma = \mathcal{E}_{\rho_1,A_1;t;A_2}[\![e]\!]\sigma_1$

where $\rho_1 = \rho[\mathcal{F}[\![te1]\!]\rho\,/\,t]$ and $\sigma_1 = \sigma\upharpoonright WTV(A_1;t;A_2)$

Proof. By induction on the structure of expression e . We prove only a limited number of difficult cases .Assume (*),(**) and (***).

1.1. Let $e \equiv x$

1.2. $x \in WTV(A_1;t;A_2)$ [(**)]

1.3. $\mathcal{E}_{\rho,A_1;A_2{}^t_{te1}}[\![x^t_{te1}]\!]\sigma$

$= \mathcal{E}_{\rho,A_1;A_2{}^t_{te1}}[\![x]\!]\sigma$ [subst]

$= \sigma(x)$ [def.$\mathcal{E}$]

$= \sigma_1(x)$ [(1.2)]

$= \mathcal{E}_{\rho_1,A_1;t;A_2}[\![x]\!]\sigma_1$ [def.$\mathcal{E}$]

2.1. Let $e \equiv (\lambda\ y{:}ty \mid f)$

2.2. Let $tf \in$ Texp be such that [(**),ER6.1]

a) $\vdash A_1;t;A_2 \blacktriangleright ty\ ,\ tf$

b) $\vdash A_1;t;A_2;y{:}ty \blacktriangleright f : tf$

2.3. $t \notin FTV(A_2;y{:}ty) = FTV(A_2)$ [(***)]

2.4. $\mathcal{F}[\![ty^t_{te1}]\!]\rho = \mathcal{F}[\![ty]\!]\rho_1$ [(4.4.3)]

2.5. Let $d \in \mathcal{F}[\![ty]\!]\rho_1$. Moreover let $A_3 \in$ Assumptions and $\sigma_3 \in ST_{\rho_1,A_1;t;A_3}$ be such that

$A_3 = A_2;y{:}ty$ $\sigma_3 = \sigma_1[d/y]$

2.6. $\sigma[d/y] \upharpoonright \mathrm{WTV}(A_1;t;A_3)$

$= (\sigma \upharpoonright \mathrm{WTV}(A_1;t;A_2))[d/y]$ [(5.1.3)]

$= \sigma_1[d/y]$

$= \sigma_3$

2.7. $\mathcal{E}_{\rho,A_1;A_3{}^t_{te1}}[\![f^t_{te1}]\!]\sigma[d/y]$

$= \mathcal{E}_{\rho_1,A_1;t;A_3}[\![f]\!](\sigma[d/y] \upharpoonright \mathrm{WTV}(A_1;t;A_3))$ [(*),(2.2b),(2.3),IH]

$= \mathcal{E}_{\rho_1,A_1;t;A_3}[\![f]\!]\sigma_3$ [(2.6)]

2.8. $\mathcal{E}_{\rho,A_1;A_2{}^t_{te1}}[\![(\lambda\, y{:}ty \mid f)^t_{te1}]\!]\sigma$

$= \mathcal{E}_{\rho,A_1;A_2{}^t_{te1}}[\![(\lambda\, y{:}ty^t_{te1} \mid f^t_{te1})]\!]\sigma$ [subst.]

$= (\lambda\, d \in \mathcal{F}[\![ty^t_{te1}]\!]\rho \mid \mathcal{E}_{\rho,A_1;A_3{}^t_{te1}}[\![f^t_{te1}]\!]\sigma[d/y]\,)$ [def.$\mathcal{E}$]

$= (\lambda\, d \in \mathcal{F}[\![ty]\!]\rho_1 \mid \mathcal{E}_{\rho_1,A_1;t;A_3}[\![f]\!]\sigma_3\,)$ [(2.4),(2.7)]

$= \mathcal{E}_{\rho_1,A_1;t;A_2}[\![(\lambda\, y{:}ty \mid f)]\!]\sigma_1$ [def.$\mathcal{E}$]

3.1. Let $e \equiv (\mathrm{elim}\ \nu(\Lambda\, s \mid tf) \mid f)$

3.2. $\vdash A_1;t;A_2 \blacktriangleright f : tf^s_{\nu(\Lambda\, s \mid tf)}$ [ER7.2]

3.3. Let r be the first type variable such that

$r \not\equiv t \ \wedge\ r \notin \mathrm{FTV}(tf) \ \wedge\ r \notin \mathrm{FTV}(te1)$

3.4. Let $(A,(\alpha^L,\alpha^R))$ be the unique initial fixed point resulting from the inverse limit construction with functor $(\mathrm{abstr}_r\ \mathcal{F}[\![(tf^s_r)^t_{te1}]\!])\rho$

3.5. Let $(B,(\beta^L,\beta^R))$ be the unique initial fixed point resulting from the inverse limit construction with functor $(\mathrm{abstr}_s\ \mathcal{F}[\![tf]\!])\rho_1$

3.6. $(\mathrm{abstr}_r\ \mathcal{F}[\![(tf^s_r)^t_{te1}]\!])\rho = (\mathrm{abstr}_s\ \mathcal{F}[\![tf]\!])\rho_1$ [(4.5.2)]

3.7. $\alpha^L = \beta^L$ [(3.4)–(3.6)]

3.8. $\mathcal{E}_{\rho,A_1;A_2{}^t_{te1}}[\![(\mathrm{elim}\ \nu(\Lambda\, s \mid tf) \mid f)^t_{te1}]\!]\sigma$

$= \mathcal{E}_{\rho,A_1;A_2{}^t_{te1}}[\![(\mathrm{elim}\ \nu(\Lambda\, r \mid (tf^s_r)^t_{te1} \mid f^t_{te1})]\!]\sigma$ [(3.1),subst]

$= \alpha^{L}(\mathcal{E}_{\rho,A_1;A_2{}^{t}_{te1}}[\![f^{t}_{te1}]\!]\sigma)$ [def.$\mathcal{E}$]

$= \alpha^{L}(\mathcal{E}_{\rho_1,A_1;t;A_2}[\![f]\!]\sigma_1)$ [(*),(3.2),(***),IH]

$= \beta^{L}(\mathcal{E}_{\rho_1,A_1;t;A_2}[\![f]\!]\sigma_1)$ [(3.7)]

$= \mathcal{E}_{\rho_1,A_1;t;A_2}[\![(\text{elim } \nu(\Lambda\ s \mid tf) \mid f)]\!]\sigma_1$ [def.$\mathcal{E}$]

4.1. Let $e \equiv (\Lambda\ s \mid f)tf1$

4.2. Let $tf \in$ Texp be such that [ER8]

a) $\vdash A_1;t;A_2;s \blacktriangleright f : tf$

b) $te \equiv_{\alpha} tf^{s}_{tf1}$

4.3. Let r be the first type variable such that

$r \not\equiv t \ \wedge\ r \notin FTV(te1) \ \wedge\ r \notin FTV(f{:}tf)$

4.4. Let $\rho_n \in$ Tenv and $A_n \in$ Assumptions and $\sigma_n \in ST_{\rho_n,A_n}$, $2 \le n \le 5$, be such that

	$\rho_2 = \rho_1[\mathcal{F}[\![tf1]\!]\rho_1 / s]$	$\sigma_2 = \sigma_1 \upharpoonright WTV(A_1;t;A_2;s)$
$A_3 = (A_2;r)^{t}_{te1}$	$\rho_3 = \rho[\mathcal{F}[\![tf1^{t}_{te1}]\!]\rho / r]$	$\sigma_3 = \sigma \upharpoonright WTV(A_1;A_3)$
$A_4 = A_2;r$	$\rho_4 = \rho_3[\mathcal{F}[\![te1]\!]\rho_3 / t]$	$\sigma_4 = \sigma_3 \upharpoonright WTV(A_1;t;A_4)$
$A_5 = A_4;s$	$\rho_5 = \rho_4[\mathcal{F}[\![r]\!]\rho_4 / s]$	$\sigma_5 = \sigma_4 \upharpoonright WTV(A_1;t;A_5)$

4.5. $\vdash A_1;t;A_2;s;r \blacktriangleright f : tf$ [(4.2a),(4.3),ER11.1]

4.6. $\vdash A_1;t;A_4 \blacktriangleright r$ [TR2]

4.7. $\vdash A_1;t;A_5 \blacktriangleright f : tf$ [(4.5),ER12.1]

4.8. $\vdash A_1;t;A_4 \blacktriangleright f^{s}_{r} : tf^{s}_{r}$ [(4.6),(4.7),ER14]

4.9. $t \notin FTV(A_2;r)$ [(***),(4.3)]

4.10. ρ_4

$= \rho_3[\mathcal{F}[\![te1]\!]\rho_3 / t]$ [def.ρ_4]

$= \rho_3[\mathcal{F}[\![te1]\!]\rho[\mathcal{F}[\![tf1^{t}_{te1}]\!]\rho / r] / t]$ [def.ρ_3]

$= \rho_3[\mathcal{F}[\![te1]\!]\rho / t]$ [(4.3),thm.4.4.1]

$= \rho[\mathcal{F}[\![tf1^{t}_{te1}]\!]\rho / r][\mathcal{F}[\![te1]\!]\rho / t]$ [def.ρ_3]

$= \rho[\mathcal{F}[\![tf1]\!]\rho_1 / r][\mathcal{F}[\![te1]\!]\rho / t]$ [(4.4.3),def.ρ_1]

$= \rho[\mathcal{F}[\![te1]\!]\rho / t][\mathcal{F}[\![tf1]\!]\rho_1 / r]$ [(4.3)]

$= \rho_1[\mathcal{F}[\![tf1]\!]\rho_1 /r]$ [def.ρ_1]

4.11. ρ_5

$= \rho_4[\mathcal{F}[\![r]\!]\rho_4 / s]$ [def.ρ_5]

$= \rho_4[\rho_4(r) / s]$ [def.$\mathcal{F}$]

$= \rho_4[(\rho_1[\mathcal{F}[\![tf1]\!]\rho_1 / r])(r) / s]$ [(4.10)]

$= \rho_4[\mathcal{F}[\![tf1]\!]\rho_1 /s]$

$= \rho_1[\mathcal{F}[\![tf1]\!]\rho_1 / r][\mathcal{F}[\![tf1]\!]\rho_1 / s]$ [(4.10)]

$= \rho_1[\mathcal{F}[\![tf1]\!]\rho_1 / s][\mathcal{F}[\![tf1]\!]\rho_1 / r]$

$= \rho_2[\mathcal{F}[\![tf1]\!]\rho_1 / r]$ [def.ρ_2]

4.12. $\sigma_5 = \sigma \restriction WTV(A_1;t;A_2;r;s) = \sigma_2 \restriction WTV(A_1;t;A_2;s;r)$

4.13. $\mathcal{E}_{\rho,A_1;A_2{}^{t}_{te1}}[\![((\Lambda\, s|\ f)tf1)^{t}_{te1}]\!]\sigma$

$= \mathcal{E}_{\rho,A_1;A_2{}^{t}_{te1}}[\![(\Lambda\, r|(f^{s}_{r})^{t}_{te1})tf1^{t}_{te1}]\!]\sigma$ [subst.,(3.1)]

$= \mathcal{E}_{\rho_3,A_1;A_3}[\![(f^{s}_{r})^{t}_{te1}]\!]\sigma_3$ [def.σ_3,def.$\mathcal{E}$]

$= \mathcal{E}_{\rho_4,A_1;t;A_4}[\![f^{s}_{r}]\!]\sigma_4$ [(*),(4.8),(4.9),IH]

$= \mathcal{E}_{\rho_5,A_1;t;A_5}[\![f]\!]\sigma_5$ [(4.6),(4.7),IH]

$= \mathcal{E}_{\rho_5,A_1;t;A_2;s;r}[\![f]\!]\sigma_2 \restriction WTV(A_1;t;A_2;s;r)$ [(4.5),(4.7),(4.12),prop.5.2.2]

$= \mathcal{E}_{\rho_2,A_1;t;A_2;s}[\![f]\!]\sigma_2$ [(4.2),(4.3),(4.11),thm5.3.1]

$= \mathcal{E}_{\rho_1,A_1;t;A_2}[\![(\Lambda\, s|\ f)tf1]\!]\sigma_1$ [def.σ_2,def.$\mathcal{E}$]

□

Theorem 5.3.3 [Renaming a bound type variable]

For all $\rho \in$ Tenv ; $A \in$ Assumptions ; $s,t \in$ Tvar ; $te,te1 \in$ Texp ; $e \in$ Exp and $\sigma \in ST_{\rho,A}$:

If $\vdash A \blacktriangleright te1$ (*)

$\vdash A;t \blacktriangleright e{:}te$ (**)

$s \notin FTV(e{:}te)$ (***)

Then $\mathcal{E}_{\rho,A}[\![(\Lambda\ t|\ e)te1]\!]\sigma = \mathcal{E}_{\rho,A}[\![(\Lambda\ s|\ e^t_s)te1]\!]\sigma$

Proof. Assume (*),(**) and (***).

1. Let $\rho_1,\rho_2 \in$ Tenv ; $A_1,A_2 \in$ Assumptions and $\sigma_1 \in ST_{\rho_1,A_1}$, $\sigma_2 \in ST_{\rho_2,A_2}$ be such that

$A_1 = A;t \qquad \rho_1 = \rho[\mathcal{F}[\![te1]\!]\rho\ /\ t] \qquad \sigma_1 = \sigma{\restriction}WTV(A_1)$

$A_2 = A;s \qquad \rho_2 = \rho[\mathcal{F}[\![te1]\!]\rho\ /\ s] \qquad \sigma_2 = \sigma{\restriction}WTV(A_2)$

2. $\vdash A_2 \blacktriangleright s$ [(1),TR2]

3. $\vdash A_1;s \blacktriangleright e : te$ [(1),(**),(***),ER11.1]

4. Since $FTV(A) \subseteq FTV(A_2)$ it follows from (*) and prop.2.4.1 that

$\vdash A_2 \blacktriangleright te1$

5. $\vdash A_2;t \blacktriangleright e : te$ [(3),ER12.1]

6. $(\sigma_2 \restriction WTV(A_2;t)) \restriction WTV(A_1;s)$

$= (\sigma \restriction WTV(A_2;t)) \restriction WTV(A_1;s)$

$= (\sigma \restriction WTV(A_1;s)) \restriction WTV(A_2;t)$

$= (\sigma_1 \restriction WTV(A_1;s)) \restriction WTV(A_2;t)$

7. $\rho_2[\mathcal{F}[\![s]\!]\rho_2\ /\ t]$

$= \rho_2[\rho_2(s)\ /\ t]$ [def.$\mathcal{F}$]

$= \rho_2[\mathcal{F}[\![te1]\!]\rho\ /\ t]$ [(1)]

$= \rho[\mathcal{F}[\![te1]\!]\rho\ /\ s][\mathcal{F}[\![te1]\!]\rho\ /\ t]$ [(1)]

$= \rho[\mathcal{F}[\![te1]\!]\rho\ /\ t][\mathcal{F}[\![te1]\!]\rho\ /\ s]$

$= \rho_1[\mathcal{F}[\![te1]\!]\rho\ /\ s]$ [(1)]

8. $\mathcal{E}_{\rho,A}[\![(\Lambda\ s|\ e^t_s)te1]\!]\sigma$

$= \mathcal{E}_{\rho_2,A_2}[\![e^t_s]\!]\sigma_2$ [def.$\mathcal{E}$]

$= \mathcal{E}_{\rho_2[\mathcal{F}[\![s]\!]\rho_2\ /\ t],A_2;t}[\![e]\!]\sigma_2{\restriction}WTV(A_2;t)$ [(2),(5),thm5.3.2]

$= \mathcal{E}_{\rho_2[\mathcal{F}[\![s]\!]\rho_2\ /\ t],A_1;s}[\![e]\!]\sigma_1{\restriction}WTV(A_1;s)$ [(3),(5),(6),prop.5.2.2]

$= \mathcal{E}_{\rho_1[\mathcal{F}[\![te1]\!]\rho\ /\ s],A_1;s}[\![e]\!]\sigma_1{\restriction}WTV(A_1;s)$ [(7)]

$= \mathcal{E}_{\rho_1,A_1}[\![e]\!]\sigma_1$ [(**),(***),thm5.3.1]

$= \mathcal{E}_{\rho,A}[\![(\Lambda\, t \mid e)te1]\!]\sigma$ [def.$\mathcal{E}$]

□

Theorem 5.3.4.[State modification]

For all $A \in$ Assumptions ; $\rho \in$ Tenv ; te,te1 $\in$ Texp ; $x \in$ Var ; $e \in$ Exp ; $\sigma \in ST_{\rho,A}$ and $d_1 \in \mathcal{T}[\![te1]\!]\rho$:

If $\vdash A \blacktriangleright te1$ (*)

$\vdash A \blacktriangleright e : te$ (**)

$x \notin FV(e)$ (***)

Then $\mathcal{E}_{\rho,A}[\![e]\!]\sigma = \mathcal{E}_{\rho,A_1}[\![e]\!]\sigma_1$

where $A_1 = A;x{:}te1$ and $\sigma_1 = \sigma[d_1/x]$

Proof. By induction to the structure of expression e . We prove only a limited number of difficult cases. Assume (*) and (**).

1.1. Let $e \equiv y$

1.2. $y \not\equiv x$ [(**),(1.1)]

1.3. $\mathcal{E}_{\rho,A}[\![y]\!]\sigma = \sigma(y) = \sigma_1(y) = \mathcal{E}_{\rho,A_1}[\![y]\!]\sigma_1$ [(1.2)]

2.1. Let $e \equiv (\lambda\, y{:}te2 \mid f) \wedge x \not\equiv y$

2.2. Let $d_2 \in \mathcal{T}[\![te2]\!]\rho$ and let $A_2 \in$ Assumptions and $\sigma_2 \in ST_{\rho,A_2}$ be such that

$A_2 = A;y{:}te2$ $\sigma_2 = \sigma[d_2/y]$

2.3. Let tf $\in$ Texp be such that [(**),ER6.1]

a) $\vdash A \blacktriangleright te2 , tf$

b) $\vdash A_2 \blacktriangleright f : tf$

c) $te \equiv_\alpha te2 \longrightarrow tf$

2.4. $\vdash A_2 \blacktriangleright te1$ [(*),ER11.2]

2.5. $x \notin FV(f)$ [(***),(2.1)]

2.6. $\mathcal{E}_{\rho,A_2}[\![f]\!]\sigma_2 = \mathcal{E}_{\rho,A_2;x:te1}[\![f]\!]\sigma_2[d_1/x]$ [(2.4),(2.3b),(2.5),IH]

2.7. $\vdash A_2;x:te1 \blacktriangleright f : tf$ [(2.3),(2.4),ER11.2]

2.8. $\vdash A_1;y:te2 \blacktriangleright f : tf$ [(2.5),ER12.2]

2.9. $\sigma_2[d_1/x] \restriction WTV(A_1;y:te2) = \sigma_1[d_2/y] \restriction WTV(A_2;x:te1)$ [$x \not\equiv y$]

2.10. $\mathcal{E}_{\rho,A}[\![(\lambda\ y:te2 \mid f)]\!]\sigma$

$= (\lambda\ d_2 \in \mathcal{F}[\![te2]\!]\rho \mid \mathcal{E}_{\rho,A_2}[\![f]\!]\sigma_2\ )$ [def.$\mathcal{E}$]

$= (\lambda\ d_2 \in \mathcal{F}[\![te2]\!]\rho \mid \mathcal{E}_{\rho,A_2;x:te1}[\![f]\!]\sigma_2[d_1/x]\ )$ [(2.6)]

$= (\lambda\ d_2 \in \mathcal{F}[\![te2]\!]\rho \mid \mathcal{E}_{\rho,A_1;y:te2}[\![f]\!]\sigma_1[d_2/y]\ )$ [(2.7),(2.8),(2.9),prop.5.2.2]

$= \mathcal{E}_{\rho,A_1}[\![(\lambda\ y:te2 \mid f)]\!]\sigma_1$ [def.$\mathcal{E}$]

3.1. Let $e \equiv (\lambda\ x:te2 \mid f)$

3.2. Let $tf \in Texp$ be such that [(**),ER6.1]

a) $\vdash A \blacktriangleright te2\ ,\ tf$

b) $\vdash A;x:te2 \blacktriangleright f : tf$

c) $te \equiv_\alpha ty \longrightarrow tf$

3.3. $\vdash A_1;x:te2 \blacktriangleright f : tf$ [(3.2b)]

3.4. $\sigma[d/x] \restriction WTV(A_1;x:te2) = \sigma_1[d/x] \restriction WTV(A;x:te2)$

3.5. $\mathcal{E}_{\rho,A}[\![(\lambda\ y:ty \mid f)]\!]\sigma$

$= (\lambda\ d \in \mathcal{F}[\![ty]\!]\rho \mid \mathcal{E}_{\rho,A;y:ty}[\![f]\!]\sigma[d/y]\ )$ [def.$\mathcal{E}$]

$= (\lambda\ d \in \mathcal{F}[\![ty]\!]\rho \mid \mathcal{E}_{\rho,A_1;y:ty}[\![f]\!]\sigma_1[d/y]\ )$ [(3.3),(3.4),prop5.2.2]

$= \mathcal{E}_{\rho,A_1}[\![(\lambda\ y:ty \mid f)]\!]\sigma_1$ [def.$\mathcal{E}$]

4.1. Let $e \equiv (\Lambda\ s \mid f)tf1$

4.2. Assume without loss of generality that [thm.5.3.3]

$s \notin FTV(te1)$

4.3. Let $A_2 \in$ Assumptions and $\sigma_2 \in ST_{\rho_2,A_2}$ be such that

$A_2 = A;s$ $\rho_2 = \rho[\mathcal{F}[\![tf1]\!]\rho\ /\ s]$ $\sigma_2 = \sigma \restriction WTV(A_2)$

4.4. $\vdash A_2 \blacktriangleright te1$ [(*),(4.2),ER11.1]

4.5. Let tf $\in$ Texp be such that [ER8]

$\vdash A_2 \blacktriangleright f : tf$

4.6. $x \notin FV(f)$ [(4.1),(**)]

4.7. $\vdash A_2;x{:}te1 \blacktriangleright f : tf$ [(4.5),(4.6),ER11.2]

4.8. $\vdash A_1;s \blacktriangleright f : tf$ [(4.2),(4.7),ER12.4]

4.9. $\sigma_2[d_1/x] \upharpoonright WTV(A_1;s)$

$= (\sigma \upharpoonright WTV(A_2))[d_1/x] \upharpoonright WTV(A_1;s)$

$= (\sigma[d_1/x] \upharpoonright WTV(A_2;x{:}te1)) \upharpoonright WTV(A_1;s)$ [(5.1.3)]

$= (\sigma_1 \upharpoonright WTV(A_2;x{:}te1)) \upharpoonright WTV(A_1;s)$

$= (\sigma_1 \upharpoonright WTV(A_1;s)) \upharpoonright WTV(A_2;x{:}te1)$

4.10. $\mathcal{E}_{\rho,A}[\![(\Lambda\ s \mid f)tf1]\!]\sigma$

$= \mathcal{E}_{\rho_2,A_2}[\![f]\!]\sigma_2$ [def.$\mathcal{E}$]

$= \mathcal{E}_{\rho_2,A_2;x:te1}[\![f]\!]\sigma_2[d_1/x]$ [(4.4),(4.5),(4.6),IH]

$= \mathcal{E}_{\rho_2,A_1;s}[\![f]\!]\sigma_1 \upharpoonright WTV(A_1;s)$ [(4.7),(4.8),(4.9),prop5.2.2]

$= \mathcal{E}_{\rho,A_1}[\![(\Lambda\ s \mid f)tf1]\!]\sigma_1$ [def.$\mathcal{E}$]

□

Theorem 5.3.5.[Substitution of expressions for variables in expressions]

For all $\rho \in$ Tenv ; A $\in$ Assumptions ; te1 $\in$ Texp ; x $\in$ Var ; e,e1 $\in$ Exp and $\sigma \in ST_{\rho,A}$:

If $\vdash A \blacktriangleright e1 : te1$ (*)

$\vdash A_1 \blacktriangleright e : te$ (**)

Then $\mathcal{E}_{\rho,A}[\![e^{x}_{e1}]\!]\sigma = \mathcal{E}_{\rho,A_1}[\![e]\!]\sigma_1$

where $A_1 = A;x{:}te1$ and $\sigma_1 = \sigma[\mathcal{E}_{\rho,A}[\![e1]\!]\sigma / x]$

Proof. By induction on the structure of expression e . We prove only a few difficult cases.

Assume (*) and (**).

1.1. Let $e \equiv x$

1.2. $\mathcal{E}_{\rho,A}[\![x^{x}_{e1}]\!]\sigma$

$= \mathcal{E}_{\rho,A}[\![e1]\!]\sigma$

$= \sigma[\mathcal{E}_{\rho,A}[\![e1]\!]\sigma\ /\ x]\ (x)$

$= \mathcal{E}_{\rho,A_1}[\![x]\!]\sigma_1$

2.1. Let $e \equiv y\ \wedge\ y \not\equiv x$

2.2. $\mathcal{E}_{\rho,A}[\![y^{x}_{e1}]\!]\sigma\ =\ \mathcal{E}_{\rho,A}[\![y]\!]\sigma\ =\ \sigma(y)\ =\ \sigma_1(y)\ =\ \mathcal{E}_{\rho,A_1}[\![y]\!]\sigma_1$

3.1. Let $e \equiv (\lambda\ y{:}te2\ |\ f)$

3.2. Let z be the first variable such that

$z \not\equiv x\ \wedge\ z \notin FV(f)\ \wedge\ z \notin FV(e1)$

3.3. Let $D = \mathcal{T}[\![te2]\!]\rho$ and $d \in D$

3.4. Let $A_n \in$ Assumptions and $\sigma_n \in ST_{\rho,A_n}$, $2 \le n \le 5$, be such that

$A_2 = A_1;y{:}te2$	$\sigma_2 = \sigma_1[d/y]$
$A_3 = A;z{:}te2$	$\sigma_3 = \sigma[d/z]$
$A_4 = A_3;x{:}te1$	$\sigma_4 = \sigma_3[\mathcal{E}_{\rho,A_3}[\![e1]\!]\sigma_3\ /\ x]$
$A_5 = A_4;y{:}te2$	$\sigma_5 = \sigma_4[\mathcal{E}_{\rho,A_4}[\![z]\!]\sigma_4\ /\ y]$

3.5. $\vdash A_3 \blacktriangleright e1 : te1$ [(3.10) in thm.2.5.4]

3.6. $\vdash A_4 \blacktriangleright f^{y}_{z} : tf$ [(3.8) in thm.2.5.4]

3.7. $\vdash A_4 \blacktriangleright z : te2$ [(3.7) in thm.2.5.4]

3.8. $\vdash A_5 \blacktriangleright f : tf$ [(3.6) in thm.2.5.4]

3.9. $\mathcal{E}_{\rho,A_3}[\![e1]\!]\sigma_3 = \mathcal{E}_{\rho,A}[\![e1]\!]\sigma$ [(*),(3.2),thm.5.3.4]

3.10. $\vdash A_2;z{:}te2 \blacktriangleright f : tf$ [(3.5) in thm.2.5.4]

3.11. σ_5

$= \sigma_4[\mathcal{E}_{\rho,A_4}[\![z]\!]\sigma_4\ /\ y]$ [def.σ_5]

$= \sigma_4[\ \sigma_4(z)\ /\ y]$ [def.$\mathcal{E}$]

$= \sigma_4[(\sigma[d/z][\mathcal{E}_{\rho,A_3}[\![e1]\!]\sigma_3\ /\ x])(z)\ /\ y]$ [(3.4)]

$= \sigma_4[d/y]$ [$z \not\equiv x$]

$= \sigma[d/z][\mathcal{E}_{\rho,A_3}[\![e1]\!]\sigma_3 / x])[d/y]$ [(3.4)]

$= \sigma[\mathcal{E}_{\rho,A_3}[\![e1]\!]\sigma_3 / x][d/y][d/z]$ [$z \not\equiv x$]

$= \sigma[\mathcal{E}_{\rho,A}[\![e1]\!]\sigma / x][d/y][d/z]$ [(3.9)]

$= \sigma_1[d/y][d/z]$

$= \sigma_2[d/z]$

3.12. $\mathcal{E}_{\rho,A}[\![(\lambda\ y{:}te2 \mid f)^x_{e1}]\!]\sigma$

$= \mathcal{E}_{\rho,A}[\![(\lambda\ z{:}te2 \mid (f^y_z)^x_{e1})]\!]\sigma$ [(3.1)]

$= (\lambda\ d \in D \mid \mathcal{E}_{\rho,A_3}[\![(f^y_z)^x_{e1}]\!]\sigma_3)$ [def.$\mathcal{E}$]

$= (\lambda\ d \in D \mid \mathcal{E}_{\rho,A_4}[\![f^y_z]\!]\sigma_4)$ [(*),(3.6),IH]

$= (\lambda\ d \in D \mid \mathcal{E}_{\rho,A_5}[\![f]\!]\sigma_5)$ [(3.7),(3.8),IH]

$= (\lambda\ d \in D \mid \mathcal{E}_{\rho,A_2;z:te2}[\![f]\!]\sigma_2[d/z])$ [(3.8),(3.10),(3.11),prop.5.2.2]

$= (\lambda\ d \in D \mid \mathcal{E}_{\rho,A_2}[\![f]\!]\sigma_2)$ [thm5.3.4]

$= \mathcal{E}_{\rho,A_1}[\![(\lambda\ y{:}te2 \mid f)]\!]\sigma_1$ [def.$\mathcal{E}$]

4.1. Let $e \equiv (\Lambda\ s \mid f)tf1$

4.2. Assume without loss of generality that [thm5.3.3]

$s \notin FTV(e1{:}te1)$

4.3. Let $A_2 \in$ Assumptions $\rho_2 \in$ Tenv and $\sigma_2 \in ST_{\rho,A_2}$ be such that

$A_2 = A;s$ $\quad\quad$ $\rho_2 = \rho[\mathcal{T}[\![tf1]\!]\rho / s]$ $\quad\quad$ $\sigma_2 = \sigma \restriction WTV(A_2)$

4.4. Let $tf \in$ Texp be such that [(**),ER8]

$\vdash A_1;s \blacktriangleright f : tf$

4.5. $\vdash A_2 \blacktriangleright e1 : te1$ [(4.5) in thm.2.5.4]

4.6. $\vdash A_2;x{:}te1 \blacktriangleright f : tf$ [(4.6) in thm.2.5.4]

4.7. $\sigma \restriction WTV(A_2)$

$= (\sigma \restriction WTV(A_2)) \restriction WTV(A)$

$= \sigma_2 \restriction WTV(A)$

4.8. $\mathrm{WTV}(A_1;s) \subseteq \mathrm{WTV}(A_2;x:te1)$ [ER12.3]

4.9. $(\sigma_2[\mathcal{E}_{\rho,A_2}[\![e1]\!]\sigma_2 / x]) \restriction \mathrm{WTV}(A_1;s)$

$= (\sigma_2[\mathcal{E}_{\rho,A}[\![e1]\!]\sigma / x]) \restriction \mathrm{WTV}(A_1;s)$ [(*),(4.5),(4.7),prop.5.2.2]

$= ((\sigma \restriction \mathrm{WTV}(A_2))[\mathcal{E}_{\rho,A}[\![e1]\!]\sigma / x]) \restriction \mathrm{WTV}(A_1;s)$

$= (\sigma[\mathcal{E}_{\rho,A}[\![e1]\!]\sigma / x] \restriction \mathrm{WTV}(A_2;x:tx)\ ) \restriction \mathrm{WTV}(A_1;s)$

$= (\sigma_1 \restriction \mathrm{WTV}(A_2;x:tx)\ ) \restriction \mathrm{WTV}(A_1;s)$

$= \sigma_1 \restriction \mathrm{WTV}(A_1;s)$ [(4.8)]

4.10. $\mathcal{E}_{\rho,A}[\![((\Lambda\ s \mid f)tf1)^{x}_{e1}]\!]\sigma$

$= \mathcal{E}_{\rho,A}[\![(\Lambda\ s \mid f^{x}_{e1})tf1]\!]\sigma$ [subst]

$= \mathcal{E}_{\rho_2,A_2}[\![f^{x}_{e1}]\!]\sigma_2$ [def.$\mathcal{E}$]

$= \mathcal{E}_{\rho_2,A_2;x:te1}[\![f]\!]\sigma_2[\mathcal{E}_{\rho,A}[\![e1]\!]\sigma / x]$ [(4.5),(4.6),IH]

$= \mathcal{E}_{\rho_2,A_1;s}[\![f]\!]\sigma_1 \restriction \mathrm{WTV}(A_1;s)$ [(4.4),(4.6),(4.9),prop5.2.2]

$= \mathcal{E}_{\rho,A_1}[\![(\Lambda\ s \mid f)tf1]\!]\sigma_1$

□

Renaming bound variables should and indeed does not alter the meaning of an expression.

Theorem 5.3.6.[Renaming a bound variable]

For all $\rho \in \mathrm{Tenv}$; $A \in \mathrm{Assumptions}$; $\sigma \in \mathrm{ST}_{\rho,A}$; $te,te1 \in \mathrm{Texp}$; $x,y \in \mathrm{Var}$ and $e \in \mathrm{Exp}$:

If $\vdash A \blacktriangleright te1$ (*)

$\vdash A;x:te1 \blacktriangleright e : te$ (**)

$y \notin \mathrm{FV}(e)$ (***)

Then $\mathcal{E}_{\rho,A}[\![(\lambda\ x:te1 \mid e)]\!]\sigma = \mathcal{E}_{\rho,A}[\![(\lambda\ y:te1 \mid e^{x}_{y})]\!]\sigma$

and $\mathcal{E}_{\rho,A}[\![(\lambda s\ x:te1 \mid e)]\!]\sigma = \mathcal{E}_{\rho,A}[\![(\lambda s\ y:te1 \mid e^{x}_{y})]\!]\sigma$

Proof. Assume (*),(**) and (***).

1. Let $D = \mathcal{F}[\![\mathrm{te1}]\!]\rho$ and $d \in D$

2. Let $A_n \in$ Assumptions and $\sigma_n \in ST_{\rho,A_n}$, $0 \le n \le 4$, be such that

   $A_1 = A;x:\mathrm{te1}$ $\quad\quad$ $\sigma_1 = \sigma[d/x]$

   $A_2 = A;y:\mathrm{te1}$ $\quad\quad$ $\sigma_2 = \sigma[d/y]$

   $A_3 = A_2;x:\mathrm{te1}$ $\quad\quad$ $\sigma_3 = \sigma_2[\mathcal{E}_{\rho,A_2}[\![y]\!]\sigma_2 \,/\, x]$

   $A_4 = A_1;y:\mathrm{te1}$ $\quad\quad$ $\sigma_4 = \sigma_1[d/y]$

3. $\vdash A_1 \blacktriangleright \mathrm{te1}$ $\quad$ [(*),ER11.2]

4. $\vdash A_2 \blacktriangleright y : \mathrm{te1}$ $\quad$ [(*),ER2]

5. $\vdash A_4 \blacktriangleright e : \mathrm{te}$ $\quad$ [(**),(***),(4),ER11.2]

6. $\vdash A_3 \blacktriangleright e : \mathrm{te}$ $\quad$ [(5),ER12.2]

7. σ_3

   $= \sigma_2[\mathcal{E}_{\rho,A_2}[\![y]\!]\sigma_2 \,/\, x]$

   $= \sigma_2[\sigma_2(y) \,/\, x]$

   $= \sigma_2[d/x]$

   $= \sigma_1[d/y]$

   $= \sigma_4$

8. $\mathcal{E}_{\rho,A_2}[\![e^x_y]\!]\sigma_2$

   $= \mathcal{E}_{\rho,A_3}[\![e]\!]\sigma_3$ $\quad$ [(4),(5),thm.5.3.5]

   $= \mathcal{E}_{\rho,A_4}[\![e]\!]\sigma_4$ $\quad$ [(5),(6),(7),prop.5.2.2]

   $= \mathcal{E}_{\rho,A_1}[\![e]\!]\sigma_1$ $\quad$ [(3),(**),(***),thm.5.3.4]

9. $\mathcal{E}_{\rho,A}[\![(\lambda\, y:\mathrm{te1} \mid e^x_y)]\!]\sigma$

   $= (\lambda\, d \in D \mid \mathcal{E}_{\rho,A_2}[\![e^x_y]\!]\sigma_2)$ $\quad$ [def.$\mathcal{E}$]

   $= (\lambda\, d \in D \mid \mathcal{E}_{\rho,A_1}[\![e]\!]\sigma_1)$ $\quad$ [(8)]

   $= \mathcal{E}_{\rho,A}[\![(\lambda\, x:\mathrm{te1} \mid e)]\!]\sigma$ $\quad$ [(2),def.$\mathcal{E}$]

The case of strict λ–abstraction is proved similarly.

□

CHAPTER 6

6.SOUNDNESS OF REDUCTION

6.1.Introduction.

In chapter 3 we have introduced a set of reduction rules for expressions. Furthermore, we have shown that for expressions e1 and e2 such that e1 » e2 their values $\mathcal{E}_{\rho,A}[\![e1]\!]\sigma$ and $\mathcal{E}_{\rho,A}[\![e2]\!]\sigma$ are members of the same domain $\mathcal{F}[\![\tau_A(e1)]\!]\rho = \mathcal{F}[\![\tau_A(e2)]\!]\rho$. In this chapter we prove that the reduction rules of chapter 3 are sound, i.e. reducing an expression yields an expression with the same value. In order to prove this result we need some elementary properties.

Property 6.1.1.[strictness]

For all $A \in$ Assumptions ; $\rho \in$ Tenv ; $\sigma \in ST_{\rho,A}$; te,te1 $\in$ Texp and f $\in$ Exp :

If $\vdash A \blacktriangleright f : te \ominus te1$

Then $(\mathcal{E}_{\rho,A}[\![f]\!]\sigma)\ (\perp_{\mathcal{F}[\![te]\!]\rho}) = \perp_{\mathcal{F}[\![te1]\!]\rho}$

□

Property 6.1.2.[normal form]

For all A $\in$ Assumptions ; $\rho \in$ Tenv ; $\sigma \in ST_{\rho,A}$ and e $\in$ Exp :

If e is in normal form

and $(\forall\ x \in WTV(A) \mid \sigma(x) \neq \perp_{\mathcal{F}[\![\tau_A(x)]\!]\rho})$

Then $\mathcal{E}_{\rho,A}[\![e]\!]\sigma \neq \perp_{\mathcal{F}[\![\tau_A(e)]\!]\rho}$.

□

6.2.Soundness.

Theorem 6.2.1.[soundness]

For all A $\in$ Assumptions ; $\rho \in$ Tenv ; $\sigma \in ST_{\rho,A}$ and e1,e2 $\in$ WTE(A) :

If $\quad$ e1 » e2

and $\quad$ $(\forall\, x \in \mathrm{WTV(A)} \mid \sigma(x) \neq \bot_{\mathcal{F}[\![\tau_A(x)]\!]\rho})$

Then $\quad$ $\mathcal{E}_{\rho,A}[\![e1]\!]\sigma = \mathcal{E}_{\rho,A}[\![e2]\!]\sigma$

Proof. It is sufficient to prove the soundness of each of the rules ν thru η . Apart from rules σ_5, π_4, π_5, β_1, β_2 and β_3 this is a trivial exercise using definition 5.2.1.

Rule σ_5: $\quad$ (**sums** f1 f2) » f , $\quad$ provided $x \notin \mathrm{FV(f)}$

where $\quad$ f1 = (**λs** x:te1 | (**appls** f (**inls** x | te2))) $\quad$ (*)

and $\quad$ f2 = (**λs** x:te2 | (**appls** f (**inrs** te1 | x))) $\quad$ (**)

Assume $\quad$ $x \notin \mathrm{FV(f)}$ $\quad$ (***)

1. The lefthandside of σ_5 is an element of WTE(A) iff

   a) $\vdash A \blacktriangleright$ te1,te2

   b) there exists a type expression te $\in$ Texp such that $\vdash A \blacktriangleright f : te1 \oplus te2 \ominus te$

2. Let $D_1 = \mathcal{F}[\![te1]\!]\rho$, $D_2 = \mathcal{F}[\![te2]\!]\rho$ and $D = \mathcal{F}[\![te]\!]\rho$. Moreover, let

   A_1 = A;x:te1 and A_2 = A;x:te2 .

3. Since $x \notin \mathrm{FV(f)}$ it follows by (1) , (***) and rule ER11.2

   that $\vdash A_1 \blacktriangleright f : (te1 \oplus te2) \ominus te$ and $\vdash A_2 \blacktriangleright f : (te1 \oplus te2) \ominus te$.

   Hence $\mathcal{F}[\![\tau_{A_1}(f)]\!]\rho = \mathcal{F}[\![\tau_{A_2}(f)]\!]\rho = \mathcal{F}[\![\tau_A(f)]\!]\rho$.

4. For $d \in D_1$:

$$\begin{aligned}
&\mathcal{E}_{\rho,A_1}[\![(\textbf{appls}\ f\ (\textbf{inls}\ x \mid te2))]\!]\sigma[d/x] \\
&= \mathcal{E}_{\rho,A_1}[\![f]\!]\sigma[d/x]\ (\mathcal{E}_{\rho,A_1}[\![(\textbf{inls}\ x \mid te2)]\!]\sigma[d/x]) \\
&= \mathcal{E}_{\rho,A_1}[\![f]\!]\sigma[d/x]\ (<1,\mathcal{E}_{\rho,A_1}[\![x]\!]\sigma[d/x]>_{D_1 \oplus D_2}) \\
&= \mathcal{E}_{\rho,A_1}[\![f]\!]\sigma[d/x]\ (<1,d>_{D_1 \oplus D_2}) \\
&= \mathcal{E}_{\rho,A}[\![f]\!]\sigma\ (<1,d>_{D_1 \oplus D_2}) \qquad [(1a),(1b),(***),\text{thm}5.3.4]
\end{aligned}$$

Similarly one proves for $d \in D_2$:

$\mathcal{E}_{\rho,A_2}[\![(\text{appls f (inrs te1} \mid \text{x}))]\!]\sigma[d/x]$

$= \mathcal{E}_{\rho,A}[\![f]\!]\sigma\ (<2,d>_{D_1 \oplus D_2})$

5. $\mathcal{E}_{\rho,A}[\![f1]\!]\sigma$

$= (\lambda\ d \in D_1$

$\mid \underline{\text{if}}\ d = \perp_{D_1} \longrightarrow \perp_D$

$[\,]\ d \neq \perp_{D_1} \longrightarrow \mathcal{E}_{\rho,A_1}[\![(\text{appls f (inls x} \mid \text{te2}))]\!]\sigma[d/x]$

$\underline{\text{fi}}$

$)$

$= (\lambda\ d \in D_1$

$\mid \underline{\text{if}}\ d = \perp_{D_1} \longrightarrow \mathcal{E}_{\rho,A}[\![f]\!]\sigma\ (<1,\perp_{D_1}>_{D_1 \oplus D_2})$ [prop.6.1.1]

$[\,]\ d \neq \perp_{D_1} \longrightarrow \mathcal{E}_{\rho,A}[\![f]\!]\sigma\ (<1,d>_{D_1 \oplus D_2})$ [(4)]

$\underline{\text{fi}}$

$)$

$= (\lambda\ d \in D_1 \mid \mathcal{E}_{\rho,A}[\![f]\!]\sigma\ (<1,d>_{D_1 \oplus D_2})\ )$

Similarly one proves that

$\mathcal{E}_{\rho,A}[\![f2]\!]\sigma = (\lambda\ d \in D_2 \mid \mathcal{E}_{\rho,A}[\![f]\!]\sigma\ (<2,d>_{D_1 \oplus D_2})\ )$

6. $\mathcal{E}_{\rho,A}[\![(\text{sums f1 f2})]\!]\sigma$

$= (\lambda\ d \in D_1 \oplus D_2$

$\mid \underline{\text{if}}\ d = <1,d_1>_{D_1 \oplus D_2} \longrightarrow (\mathcal{E}_{\rho,A}[\![f1]\!]\sigma)\ (d_1)$

$[\,]\ d = <2,d_2>_{D_1 \oplus D_2} \longrightarrow (\mathcal{E}_{\rho,A}[\![f2]\!]\sigma)\ (d_2)$

$\underline{\text{fi}}$

$)$

$= (\lambda\, d \in D_1 \oplus D_2$

$\quad |\ \underline{\text{if}}\ d = \langle 1,d_1\rangle_{D_1 \oplus D_2} \longrightarrow \mathcal{E}_{\rho,A}[\![f]\!]\sigma\, (\langle 1,d_1\rangle_{D_1 \oplus D_2})$ [(5)]

$\quad \square\ d = \langle 2,d_2\rangle_{D_1 \oplus D_2} \longrightarrow \mathcal{E}_{\rho,A}[\![f]\!]\sigma\, (\langle 2,d_2\rangle_{D_1 \oplus D_2})$ [(5)]

$\quad \underline{\text{fi}}$

$\quad)$

$= (\lambda\, d \in D_1 \oplus D_2 \mid \mathcal{E}_{\rho,A}[\![f]\!]\sigma(d))$

$= \mathcal{E}_{\rho,A}[\![f]\!]\sigma$

Rule π_4 : (prols (prods e1 e2)) » e1 , provided e2 in normal form

Assume e2 is in normal form. (*)

$(\forall\, x \in \text{WTV}(A) \mid \sigma(x) \neq \bot_{\mathcal{F}[\![\tau_A(e)]\!]\rho})$ (**)

1. Let $D_1 = \mathcal{F}[\![\tau_A(\text{e1})]\!]\rho$ and $D_2 = \mathcal{F}[\![\tau_A(\text{e2})]\!]\rho$

2. $\mathcal{E}_{\rho,A}[\![(\text{prols (prods e1 e2)})]\!]\sigma$

$= \psi_1(\langle \mathcal{E}_{\rho,A}[\![\text{e1}]\!]\sigma , \mathcal{E}_{\rho,A}[\![\text{e2}]\!]\sigma \rangle_{D_1 \otimes D_2})$

$= \underline{\text{if}}\ \mathcal{E}_{\rho,A}[\![\text{e2}]\!]\sigma = \bot_{D_2} \longrightarrow \bot_{D_1}$

$\quad \square\ \mathcal{E}_{\rho,A}[\![\text{e2}]\!]\sigma \neq \bot_{D_2} \longrightarrow \mathcal{E}_{\rho,A}[\![\text{e1}]\!]\sigma$

$\quad \underline{\text{fi}}$

$= \mathcal{E}_{\rho,A}[\![\text{e1}]\!]\sigma$ [(*),(**),prop.6.1.2]

Similarly one proves the soundness of rule π_5 .

Rules β_1 , β_2 , β_3 :

The soundness of rules β_1 and β_2 follows from a simple computation using theorem 5.3.2.

The soundness of rule β_3 follows from theorem 5.3.1.

□

CHAPTER 7

7.A TYPED FIXED POINT COMBINATOR

7.1.Syntax.

In the type free lambda calculus every term can be considered as a function. Moreover, every term (function) has a fixed point which can be computed using a fixed point combinator. The most well known fixed point combinator in the type free lambda calculus is the Curry combinator

$$\mu = (\lambda\, f \mid (\lambda\, x \mid f(xx))\, (\lambda\, x \mid f(xx))\,)$$

A simple calculation shows that for every term g the terms μg and $g(\mu g)$ are convertible; so μg can be considered as a fixed point of g . It can be shown that in the D_∞ model of the type free lambda calculus μ corresponds to the least fixed point operator, see for instance Wadsworth [Wa76].

In this chapter we show that similar results hold for the typed language described in this report. Let te be an arbitrary type expression. In this section we shall describe an expression $(\mu \mid te)$ with type $(te \longrightarrow te) \longrightarrow te$,which can be considered as a typed version of the Curry combinator. In $(\mu \mid te)$ a recursively defined type will be used. Some properties associated with the corresponding domain (found by the inverse limit construction) are given in section 7.2. Finally in section 7.3 we show that in the appropriate domain $(\mu \mid te)$ corresponds to the least fixed point operator.

In this chapter we use the following abbreviations

$$w \;=\; \nu(\Lambda\, t \mid t \longrightarrow te) \qquad (7.1.1)$$

where t is the first type variable such that $t \notin FTV(te)$ and

$$g \;=\; (\lambda\, x{:}w \mid (\textbf{appl}\ f\ (\textbf{appl}\ (\textbf{intro}\ w \mid x)\ x)\,)\,. \qquad (7.1.2)$$

A typed version of the Curry combinator is then given by

$$(\mu \mid te) = (\, f : te \longrightarrow te \mid (\textbf{appl}\ g\ (\textbf{elim}\ w \mid g))\,)\,. \qquad (7.1.3)$$

It is an elementary exercise to show that the following type inference rule holds

$$\frac{A \blacktriangleright \text{te}}{A \blacktriangleright (\mu\,|\,\text{te}) : (\text{te} \longrightarrow \text{te}) \longrightarrow \text{te}}$$

The following theorem states that $(\mu\,|\,\text{te})$ is a syntactic fixed point combinator.

Theorem 7.1.4.

Let $f : \text{te} \longrightarrow \text{te}$. Then **(appl** $(\mu\,|\,\text{te})$ f) and **(appl** f **(appl** $(\mu\,|\,\text{te})$ f)) have a common reduct.

Proof. The theorem is easily proved by the following computations.

	(appl (μ | te) f)	
»	**(appl** g **(elim** w | g))	[β_1]
»	**(appl** (**λ** x:w | **(appl** f **(appl (intro** w | x) x))) **(elim** w | g))	[(7.1.2)]
=	**(appl** f **(appl (intro** w | **(elim** w | g)) **(elim** w | g)))	[β_1]
»	**(appl** f **(appl** g **(elim** w | g)))	[ε_2]

Also

	(appl f **(appl** (μ | te) f))	
»	**(appl** f **(appl** g **(elim** w | g)))	[β_1]

which proves the theorem.

□

Note that, although **(appl** $(\mu\,|\,\text{te})$ f) and **(appl** f **(appl** $(\mu\,|\,\text{te})$ f)) have a common reduct, it is not possible to reduce one of these terms to the other. The same property holds for the untyped Curry combinator. For the untyped lambda calculus there exists another fixed point combinator, the Turing combinator μ' , such that $\mu'f$ reduces to $f(\mu'f)$. A typed version of the Turing combinator, with similar properties as described in this report, can also be given, see for instance Struik[St88].

7.2. Technical results.

In the construction of $(\mu|te)$ we used the recursive type $w = \nu(\Lambda\ t|t \longrightarrow te)$. In a type environment ρ the corresponding domain W is obtained in the following way. Let the functor $F : C \longrightarrow C$ be given by

$$F = (\mathrm{abstr}_t\, \mathcal{F}[\![t \longrightarrow te]\!])(\rho)$$

Then, following the semantics of type expressions as described in section 4, we get

$$W = \mathcal{F}[\![\nu(\Lambda\ t|t \longrightarrow te)]\!]\rho = \mathrm{IFP}(F). \tag{7.2.1}$$

A simple computation (using the definition of abstr_t given in section 4.1) yields that

$$F = FS \circ \langle I, C_B \rangle : C \longrightarrow C$$

where $I, C_B : C \longrightarrow C$ are respectively the identity functor and the constant functor corresponding to the domain $B = \mathcal{F}[\![te]\!]\rho$. Recall that in the category C (= $\underline{CPO}_{PR}$) a morphism $\alpha \in \mathrm{Hom}(A_1,A_2)$ is a pair $\alpha = (\alpha^L,\alpha^R)$, where $\alpha^L : A_1 \longrightarrow A_2$ is an embedding and $\alpha^R : A_2 \longrightarrow A_1$ is a projection, i.e.

$$\alpha^L \circ \alpha^R \sqsubseteq \mathrm{id}_{A_2} \quad \text{and} \quad \alpha^R \circ \alpha^L = \mathrm{id}_{A_1} \tag{7.2.2}$$

If α is an isomorphism, then in the first relation equality holds and

$$(\alpha^{-1})^L = \alpha^R \quad , \ (\alpha^{-1})^R = \alpha^L .$$

The composition of the morphisms $\alpha = (\alpha^L,\alpha^R) \in \mathrm{Hom}(A_1,A_2)$ and $\beta = (\beta^L,\beta^R) \in \mathrm{Hom}(A_2,A_3)$ is given by

$$\beta \circ \alpha = (\beta^L \circ \alpha^L , \alpha^R \circ \alpha^L) .$$

From the definition of the function space functor FS (see for instance [BH88], where this functor is called A) it follows that (recall $B = \mathcal{F}[\![te]\!]\rho$)

– if $A \in \mathrm{obj}(C)$, then $F(A) = [A \longrightarrow B]$,

– if $\alpha \in \mathrm{Hom}(A_1,A_2)$, then $F(\alpha) \in \mathrm{Hom}([A_1 \longrightarrow B],[A_2 \longrightarrow B])$ is defined by

$$F(\alpha)^L(\xi) = \xi \circ \alpha^R \text{ for } \xi \in [A_1 \longrightarrow B] , \tag{7.2.3}$$

$F(\alpha)^R(\eta) = \eta \circ \alpha^L$ for $\eta \in [A_2 \rightarrow B]$.

The object W is constructed in the following way. Let D_0 be the initial object in the category C, i.e. D_0 is the one point c.p.o. Let $D_k = F^k(D_0)$ for $k \geq 1$. Since D_0 is initial, there exists a unique morphism $\psi_0 \in \mathrm{Hom}(D_0,D_1)$. Let $\psi_k = F^k(\psi_0)$ for $k \geq 1$. Then $\Delta = <(D_i,\psi_i)>_{i=0}^{\infty}$ is an ω–chain in C. Since C is an ω–category this ω–chain has a colimit (W,α). This defines the (an) object W. To see that W is a fixed point of the functor F, we consider the ω–chain $\Delta' = <(F(D_i),F(\psi_i))>_{i=0}^{\infty} = <(D_{i+1},\psi_{i+1})>_{i=0}^{\infty}$. Since F is an ω–continuous functor it preserves colimits. So $(F(W),F(\alpha))$ (where $F(\alpha)$ stands for $<F(\alpha_i)>_{i=0}^{\infty}$) is a colimit of the ω–chain Δ'. Apart from the first element of Δ, the chains Δ and Δ' are identical. Thus (W,α) and $(F(W),F(\alpha))$ are both colimits of the same ω–chain, which implies that there exists an isomorphism $\beta \in \mathrm{Hom}(F(W),W)$. The situation may be elucidated by the following figure.

$$\begin{tikzcd}
\Delta : D_0 \arrow[r, "\psi_0"] & D_1 \arrow[r, "\psi_1"] & \ldots \arrow[r] & D_k \arrow[rr, "\psi_k"] \arrow[dr] & & D_{k+1} \arrow[dl] \\
 & & & & W \arrow[d, shift right, "\beta"'] & \\
 & & & & F(W) \arrow[u, shift right, "\beta^{-1}"'] & \\
\Delta' : F(D_0) \arrow[r, "F(\psi_0)"'] & F(D_1) \arrow[r, "F(\psi_1)"'] & \ldots \arrow[r] & F(D_k) \arrow[rr, "F(\psi_k)"'] \arrow[ur] & & F(D_{k+1}) \arrow[ul]
\end{tikzcd}$$

Now the following properties hold (see for instance Smyth and Plotkin[SP82] or Bos and Hemerik [BH88]).

$$\alpha_k^L \circ \alpha_k^R \sqsubseteq id_W \qquad (7.2.4)$$

$$\alpha_k^L \circ \alpha_k^R \sqsubseteq id_{D_k} \qquad (7.2.5)$$

$$\beta^{-1} \circ \alpha_{k+1} = F(\alpha_k) \qquad (7.2.6)$$

$$\beta \circ F(\alpha_k) = \alpha_{k+1} \qquad (7.2.7)$$

$$\bigsqcup_{k=0}^{\infty} \alpha_k^L \circ \alpha_k^R = \mathrm{id}_W \tag{7.2.8}$$

Define the mapping $P_k : W \longrightarrow W$ by $P_k = \alpha_k^L \circ \alpha_k^R$. Then (7.2.4) and (7.2.8) can be written as

$$P_k \sqsubseteq \mathrm{id}_W \tag{7.2.9}$$

and

$$\bigsqcup_{k=0}^{\infty} P_k = \mathrm{id}_W \tag{7.2.10}$$

Since D_0 is the one–point c.p.o. and α_0^L is strict, we have

$$P_0(x) = \bot_W \qquad\qquad \text{for all } x \in W \tag{7.2.11}$$

In the remainder of this section we give some technical lemmas, which will be used in section 7.3.

Lemma 7.2.12.

If $\ell \geq k \geq 0$ then $\alpha_k^R \circ P_\ell = \alpha_k^R$ and $P_\ell \circ \alpha_k^L = \alpha_k^L$.

Proof. We prove the first relation for fixed ℓ by induction with respect to k. If $k = \ell$ the result follows from (7.2.5). Next suppose $\alpha_k^R \circ P_\ell = \alpha_k^R$ and $\ell \geq k \geq 1$. Since (W,α) is a cocone for Δ , we have $\alpha_{k-1} = \alpha_k \circ \psi_{k-1}$, so $\alpha_{k-1}^R = \psi_{k-1}^R \circ \alpha_k^R$. Then using the induction hypothesis, we get

$$\alpha_{k-1}^R \circ P_\ell = \psi_{k-1}^R \circ \alpha_k^R \circ P_\ell = \psi_{k-1}^R \circ \alpha_k^R = \alpha_{k-1}^R \ .$$

The second part of the lemma can be proved in a similar way.

□

Lemma 7.2.13.

If $k \geq 0$ and $\ell \geq 0$ then $P_k \circ P_\ell = P_{\min(k,\ell)}$.

Proof. The lemma follows immediately from the definition of P_k and lemma 7.2.12.

□

Note that if $y \in W$, then $\beta^R(y) \in F(W) = [W \longrightarrow B]$. In the case that $y \in P_{k+1}(W)$ the mapping $\beta^R(y) : [W \longrightarrow B]$ has a special property.

Lemma 7.2.14.

Let $x \in W$ and $\ell \geq k \geq 0$. Then

$$\beta^R(P_{k+1}(x)) = \beta^R(P_{k+1}(x)) \circ P_\ell .$$

Proof. The lemma follows from the following computation.

$$\begin{aligned}
&\beta^R(P_{k+1}(x)) \\
=\ & F(\alpha_k)^L (\alpha^R_{k+1}(x)) && [\text{L component of (7.2.6)}, (\beta^{-1})^L = \beta^R] \\
=\ & (\alpha^R_{k+1}(x)) \circ \alpha^R_k && [(7.2.3)] \\
=\ & (\alpha^R_{k+1}(x)) \circ \alpha^R_k \circ P_\ell && [\text{lemma.7.2.12}]
\end{aligned}$$

□

7.3. **Semantics.**

We now show that the semantics of $(\mu \,|\, te)$ is the least fixed point operator in the appropriate c.p.o. The computation given here, is similar to the computation of the semantics of the untyped Curry fixed point combinator as given in Wadsworth [Wa76].

Suppose that $\vdash A \blacktriangleright te$ and let $\rho \in$ Tenv. We introduce the following abbreviations :

$w = \nu(\Lambda\, t \,|\, t \longrightarrow te)$,

$g = (\lambda\, x{:}w \mid (\text{appl } f\ (\text{appl } (\text{intro } w \,|\, x)\ x)))$,

$W = \mathcal{F}[\![\nu(\Lambda\, t \,|\, t \longrightarrow te)]\!]\rho$,

$B = \mathcal{F}[\![te]\!]\rho$,

$\chi = \mathcal{E}_{\rho,A_1}[\![g]\!]\sigma[\phi/f]$ where $A_1 = A;f{:}te \longrightarrow te$

From the definition of the semantics of expressions, see [def. 5.2.1 case 7], it follows that

$$\chi \;=\; \mathcal{E}_{\rho,A_1}[\![g]\!]\sigma[\phi/f] \;=\; (\lambda\, d \in W \mid \phi(\,(\beta^R(d))\; d)) \tag{7.3.1}$$

where β is the isomorfism between $F(W) = [W \longrightarrow B]$ and W . From (7.1.3) we get

$$\mathcal{E}_{\rho,A}[\![(\mu \mid te)]\!]\sigma \;=\; (\lambda\, \phi \in [B \longrightarrow B] \mid \chi(\beta^L(\chi))\,) \tag{7.3.2}$$

The following theorem shows that $\mathcal{E}_{\rho,A}[\![(\mu \mid te)]\!]\sigma$ is a fixed point operator for the domain B.

Theorem 7.3.3.

Let $\phi \in [B \longrightarrow B]$. Then for all states $\sigma \in ST_{\rho,A}$

$$(\,\mathcal{E}_{\rho,A}[\![(\mu \mid te)]\!]\sigma\,)\;\phi \;=\; \phi\,((\,\mathcal{E}_{\rho,A}[\![(\mu \mid te)]\!]\sigma\,)\;\phi\,)$$

Proof. The theorem follows from the following computation.

$$\begin{aligned}
& (\,\mathcal{E}_{\rho,A}[\![(\mu \mid te)]\!]\sigma\,)\;\phi \\
=\;& \chi\,(\beta^L(\chi)) && [(7.3.2)] \\
=\;& \phi\,(\,(\beta^R(\beta^L(\chi))\,)\,(\beta^L(\chi))\,) && [(7.3.1)] \\
=\;& \phi\,(\chi\,(\beta^L(\chi))\,) && [(7.2.2)] \\
=\;& \phi\,(\,(\mathcal{E}_{\rho,A}[\![(\mu \mid te)]\!]\sigma)\;\phi\,) && [(7.3.2)]
\end{aligned}$$

□

Since

$$\mu_B \;=\; (\lambda\, \phi \in [B \longrightarrow B] \mid \bigsqcup_{k=0}^{\infty} \phi^k(\bot_B)\,) \tag{7.3.4}$$

is the least fixed point operator in the c.p.o. B, we now have

$$\mu_B \sqsubseteq \mathcal{E}_{\rho,A}[\![(\mu \mid te)]\!]\sigma \;. \tag{7.3.5}$$

The following theorem shows that in (7.3.5) equality holds.

Theorem 7.3.6.

For all states $\sigma \in ST_{\rho,A}$

$$\mathcal{E}_{\rho,A}[\![(\mu \mid te)]\!]\sigma \;=\; \mu_B \;.$$

Proof. We first show by induction with respect to k that for $k \geq 0$

$$\chi\,(P_k\,(\beta^L(\chi))\,)\;\sqsubseteq\;\phi^{k+1}(\perp_B) \qquad (7.3.7)$$

– Induction basis, k = 0. Then

$$\begin{aligned}
&\chi\,(P_0\,(\beta^L(\chi))\,) \\
=\;&\chi\,(\perp_W) && [(7.2.11)] \\
=\;&\phi\,(\,(\beta^R(\perp_W))\,\perp_W\,) && [(7.3.1)] \\
=\;&\phi\,(\perp_B)\,. && [\beta^R \text{ is strict}]
\end{aligned}$$

– Induction step. Suppose (7.3.7) holds. Then

$$\begin{aligned}
&\chi\,(P_{k+1}\,(\beta^L(\chi))\,) \\
=\;&\phi\,(\,(\beta^R\,(P_{k+1}\,(\beta^L(\chi))\,)\,)\,(P_{k+1}\,(\beta^L(\chi))\,)\,) && [(7.3.1)] \\
=\;&\phi\,(\,(\beta^R\,(P_{k+1}\,(\beta^L(\chi))\,)\circ P_k\,)\,(P_{k+1}\,(\beta^L(\chi))\,)\,) && [\text{lemma 7.2.14 with } \ell = k \text{ and } x = \beta^L(\chi)] \\
=\;&\phi\,(\,(\beta^R\,(P_{k+1}\,(\beta^L(\chi))\,)\,)\,(P_k\,(\beta^L(\chi))\,)\,) && [\text{lemma 7.2.13}] \\
\sqsubseteq\;&\phi\,(\chi\,(P_k\,(\beta^L(\chi))\,)\,) && [(7.2.9),\ \beta^R \text{ and } \phi \text{ are monotonic, } (7.2.2)] \\
\sqsubseteq\;&\phi\,(\phi^{k+1}\,(\perp_B)\,) && [\text{induction hypothesis (7.3.7), } \phi \text{ is monotonic}]
\end{aligned}$$

This proves (7.3.7) for all $k \geq 0$. The theorem now follows from (7.3.5) and the following computation.

$$\begin{aligned}
&\mathcal{E}_{\rho,A}[\![(\mu \mid te)]\!]\sigma \\
=\;&(\lambda\,\phi \in [B \longrightarrow B] \mid \chi(\,(\,\bigsqcup_{k=0}^{\infty} P_k)(\beta^L(\chi))\,)\,) && [(7.3.2),(7.2.10)]
\end{aligned}$$

$$= (\lambda\, \phi \in [B \longrightarrow B] \mid \bigsqcup_{k=0}^{\infty} (\chi\, (P_k(\beta^L(\chi)))\,)\,) \qquad [\chi \text{ is continuous}]$$

$$\sqsubseteq (\lambda\, \phi \in [B \longrightarrow B] \mid \bigsqcup_{k=0}^{\infty} \phi^{k+1}(\bot_B)\,) \qquad [(7.3.7)]$$

$$= \mu_B \qquad [(7.3.4)]$$

□

Thus we have shown that the semantics of $(\mu \mid te)$ is the least fixed point operator in the domain B corresponding to the type expression te. This result means that it is not necessary to add recursion explicitly to the language given in chapters 1 and 2. Recursion can be performed using the typed fixed point combinator $(\mu \mid te)$, which can be written in terms of the already defined language. Note that the presence of recursively defined types is essential for the construction of $(\mu \mid te)$.

REFERENCES

Ba81 Barendregt, H. P., *The lambda calculus*, North Holland,Amsterdam, (1981).

BH88 Bos, R. and Hemerik, C. , *An introduction to the category theoretical solution of recursive domain equations*, to appear.

HMcQM86 Harper, R., MacQueen, D. and Milner, R. , *Standard ML*, LFCS Report Series,ECS LFCS–86–2, University of Edinburgh (1986).

HeStr73 Herrlich, H. and Strecker, G. E., *Category Theory*, Allyn and Bacon inc, Boston (1973).

HiSe86 Hindley, J. R. and Seldin, J. P., *Introduction to Combinators and λ–Calculus*, Cambridge, University Press (1986).

LS81 Lehmann, D. J. and Smyth, M. B., *Algebraic Specification of Data Types: A Synthetic Approach*, Math. Systems Theory 14 (1981) 97–139.

McL71 Mac Lane, S., *Categories for the Working Mathematician*, Graduate Texts in Mathematics,Springer, New York, (1971).

McQPS86 MacQueen, D.,Plotkin, G. and Sethi, R., *An Ideal Model for Polymorphic Types*, Inf. and Contr. **71** (1986) 95–130.

McC79 McCracken, N., *An Investigation of a Programming language with a Polymorphic Type Structure*, Ph. D. dissertation, Syracuse University, New York (1979).

Me86 Mendler, N. P., *First– and Second–Order Lambda Calculi with Recursive Types*, Technical Report, Department of Computer Science, Cornell University, Ithaca, New York (1986).

Mi71 Mitchell, J.C., Semantic Models for Second–Order Lambda Calculus, in *25th Annual Symposium on Foundations of Computer Science*, IEEE Computer Society, New York (1984).

Re85 Reynolds, J. C., Three approaches to type structure, in *Mathematical Foundations of software development*, LNCS 185, Springer, New York, (1985).

REFERENCES

SP82 Smyth, M. B. and Plotkin, G. D., *The category–theoretic solution of recursive domain equations*, SIAM J. Comput. 11 (1982) 761–783.

St87 Struik, M., *Some Relations between Operational and Denotational Semantics of a Typed Lambda Calculus*, Master's Thesis, Department of Mathematics and Computing Science, Eindhoven University of Technology, the Netherlands (1988).

Wa76 Wadsworth, C. P., *The relation between computational and denotational properties for Scott's D_∞ Models of the lambda calculus*, SIAM J. Comput. 5 (1976) 488–521.