INFINITE NORMAL FORMS FOR THE λ-CALCULUS AND SEMANTICS OF PROGRAMMING LANGUAGES

by

Reiji Nakajima

Memorandum No. ERL-M525

June 1975

ELECTRONICS RESEARCH LABORATORY

College of Engineering
University of California, Berkeley
94720

INFINITE NORMAL FORMS FOR THE λ-CALCULUS AND SEMANTICS OF PROGRAMMING LANGUAGES*

Reiji Nakajima

Computer Science Division
Department of Electrical Engineering and Computer Sciences
and the Electronics Research Laboratory
University of California at Berkeley

June 1975

Abstract

The semantics of programming languages is studied through the notion of infinite expansions of programs. By the infinite expansion of a program one means, for example, the thorough unwinding of the loops which are constituted by such control structures as go to's, while's and recursions. One can also view this infinite expansion as the executions for all possible inputs. One way to describe the meaning (or the semantics) of a program is to give its infinite expansion.

This idea is formalized on the domain of the λ-calculus. We define a mapping, from the λ-expressions to an algebraic domain, called C-function. The map of a λ-expression (program) by the C-function is the infinite expansion of the λ-expression which can be said to be a generalization of the normal forms for the λ-calculus. Böhm's Theorem on the normal λ-expression is extended to general λ-expressions via the C-function.

The main result of this thesis is that the semantics of the λ-expressions given by Scott's model D_∞ of λ-calculus is equivalent to the semantics of the λ-expressions given by their maps

*This work was partially supported by National Science Foundation grant GJ-34342X.

of the C-function. More precisely, the partial order among the λ-expressions in D_∞ is characterized by the partial order among their maps by the C-function in the algebraic domain that includes the image of the C-function. Extending the syntactical structure of the C-function, the λ-expressions are generalized to the infinite λ-expressions and the C-function is also extended to be defined on all the infinite λ-expressions. It is shown that the image of the infinite λ-expressions by the C-function forms a smooth structure of the partial order and its lattice topology is equivalent to the lattice topology of the λ-expressions induced by D_∞.

Utilizing this lattice topology, an attempt is made to give an axiomatization of the extensional model theory of the λ-calculus. Also, the formal idea described above is interpreted to realistic programming languages such as Algol-like programs and recursively defined programs.

Acknowledgment

I am deeply indebted to Dr. James Morris at Xerox Corporation for his whole-hearted support of my work, most of which went beyond his obligations and duties.

Special acknowledgment is due to Dr. Christopher Wadsworth who offered numerous helpful suggestions throughout my research.

I thank Professors Leon Henkin and Richard Karp for their useful comments on this thesis. I owe the beauty of this thesis to Ms. Ruth Suzuki's artistic talent.

A large number of people encouraged me 'unofficially' during my three year stay at Berkeley. Since such a large acknowledgment, however, does not suit this modest thesis and since any private feeling does not match the λ-calculus, I would rather express my appreciation to them by dedicating this thesis to:

Maka Hannya Haramita

INFINITE NORMAL FORMS FOR THE λ-CALCULUS

AND SEMANTICS OF PROGRAMMING LANGUAGES

Reiji Nakajima

Table of Contents

CHAPTER 1

INTRODUCTION

The aim of this dissertation is to study some properties of the λ-calculus as a computation model and contribute to a better understanding of the semantics of programming languages. The λ-calculus was originally introduced by Church as a logical system. A variety of formal theories on the λ-calculus were discussed by several mathematical logicians, e.g. [5].

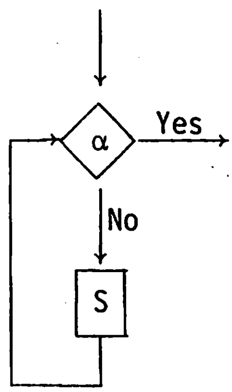
The λ-calculus has atracted some theoretical computer scientists since it can be regarded as a model of programming languages [6 , 7 ,19]. Many concepts of programming languages were analyzed through the corresponding concepts in the λ-calculus.

However, the sound understanding of the λ-calculus as a model of computation became possible only after Scott developed the theory of computation on lattice domain [14,15,18], in which he gave the construction of D_∞, the first semantic model for the λ-calculus [15,16]. On this domain, the model theory of the λ-calculus was developed by Wadsworth [21,22] and many interesting properties of the behavior of the λ-expressions in D_∞ were shown as we see in Chapter 2. In this thesis, we make efforts to develop further the theory on λ-expressions vs. D_∞. In [16], Scott gives an interesting lecture on the λ-calculus. There, he asserts that the interpretation of the λ-calculus via D_∞ gives a more essential meaning to the λ-calculus than the conversion rules. For example, Wadsworth proved that there exists a normal λ-expression which is equivalent to a non-normal expression in Scott's D_∞ although they cannot be converted to each other by the applications of conversion rules. In Chapter 3, we shall define the "infinite normal forms" for the λ-expressions, normal or non-normal. Then we shall show that two λ-expressions are equivalent under Scott's interpretation (i.e. D_∞) if they have the same infinite normal forms. The infinite normal form can be said to be the <u>infinite expansion</u> of a λ-expression.

We illustrate this idea of infinite expansion in the following discussion on flowchart programs.

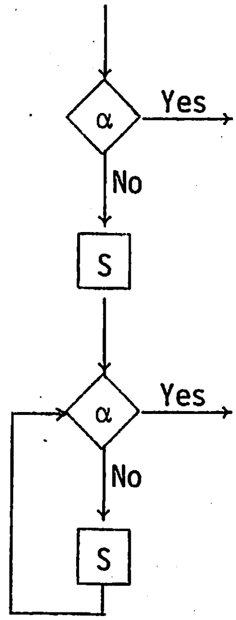
We are given a flow chart program:

A0:



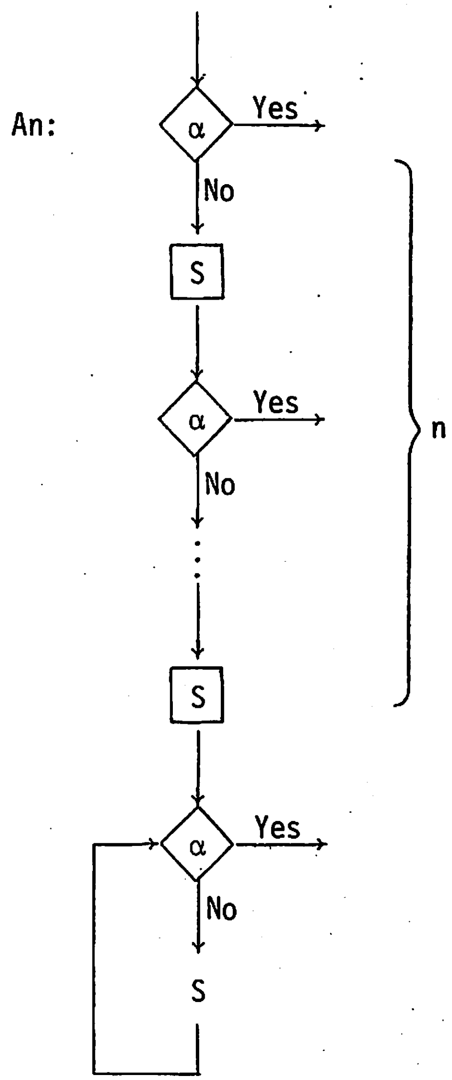
where α is a Boolean function and S is a statement (or a list of statements)

A1:



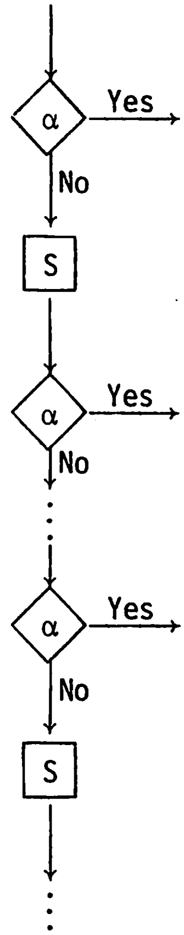
Since A1 is the result of unwinding the loop in A0 one time, A1 is equivalent to A0. From another point of view, the transformation A1 ↦ A2 can be regarded as the execution of A0 for one time under an unspecified input.

Applying this operation n times, we have



Letting $n \to \infty$, we have an infinitely sequential flowchart:

A∞:



A∞ can be regarded as the <u>infinite expansion of</u> A0 or the <u>result of execution of</u> A0 <u>under all possible inputs</u>.

It is possible to apply this idea to more complex program constructs. The infinite expansion of programs can be formalized in the following way: Let P be the domain of (some category of) programs and I be the domain of the infinite expansion of the programs which belong to P. The expansion is a mapping $E: P \to I$. (in the example, for $A0 \in P$, $E(A0) = A\infty$.) We raise the following questions:

1) How can we formalize I and E?

2) Can we say that the meaning (semantics) of a program P is given by $E(P)$? So, for instance, is P_1 equivalent to P_2 if and only if $E(P_1) = E(P_2)$?

3) What kind of structure does I have? Does it have, for instance, a lattice-like structure?

We shall answer these questions regarding λ-expressions as programs. Namely, we have the following correspondence:

$P \longleftrightarrow \Lambda$	(the λ-expressions)
$I \longleftrightarrow \mathbb{C}_{inf}$	(a partially ordered set defined in Chapter 6)
$E \longleftrightarrow C$	(C-function in Chapter 3 or infinite normal form)

Here, the transformation $An \mapsto A(n+1)$ corresponds to a β-reduction. C gives the map $A0 \mapsto A\infty$. On the other hand, the equivalence between two programs (i.e. λ-expressions) P_1 and P_2 is given by the equality as members of D_∞.

In Chapter 5 we shall try to bridge the gap between the λ-expressions and the real programming languages and show that, under some translation of programming languages, the infinite normal forms, in fact, correspond to the infinite expansion or execution of programs.

CHAPTER 2

PREPARATIONS

We make a review on the λ-calculus, Scott's lattice theoretic approach to computation and Wadsworth's model theory of the λ-calculus in D_∞, which constitute the prerequisite for this thesis.

§1. The λ-Calculus

We shall denote the set of the integers by $\mathbb{Z}$ and the set of the positive integers by $\mathbb{N}$ throughout this thesis.

2.1.1 Definition (λ-expression). Λ is the set of all of the expressions that are formed by the following rules:

Let U be the denumerable set of the variables. Assume that there is a numbering on the members of U, i.e. $U = \{v_1, v_2, \ldots\}$.

1) A variable $v \in U$ standing alone is in Λ.

2) (Application). If $x, y \in \Lambda$, so is $x(y)$.

3) (Abstraction). If $v \in U$ and $x \in \Lambda$ then $\lambda v.x \in \Lambda$.

2.1.2 Example. We list some λ-expressions:

$$I = \lambda v.v$$

$$K = \lambda x.\lambda y.x$$

$$H = \lambda x.\lambda y.y$$

$$\underline{spl} = (\lambda x.x(x))(\lambda x.x(x))$$

$$Y = \lambda f.(\lambda x.f(x(x)))(\lambda x.f(x(x)))$$ (Curry's Paradoxical Combinator)

$$J = Y(\lambda f.\lambda x.\lambda y.x(f(y)))$$ (Wadsworth)

2.1.3 Definition (Bound Variables). Given $x \in \Lambda$, we define $\mathcal{B}(x) \subset U$, the set of the bound variables in x.

1) $\mathcal{B}(v) = \emptyset$ for $v \in U$.

2) $\mathcal{B}(x(y)) = \mathcal{B}(x) \cup \mathcal{B}(y)$ for $x, y \in \Lambda$.

3) $\mathcal{B}(\lambda u.x) = \mathcal{B}(x) \cup \{u\}$.

In 3), x is said to be the scope of the bound variable v. If a variable v occurring in x is not bound in x, we say that v is free in x. If $x \in \Lambda$ has no free variables, x is said to be <u>closed</u>. We denote the set of all closed λ-expressions by $\underline{\Lambda_c}$.

2.1.4 <u>Definition</u> (Subexpression). Given $x, y \in \Lambda$, we say that x is a <u>subexpression</u> of x and write it as $x \lhd y$ if one of the following conditions holds:

1) $x = y$.
2) $x \lhd z$ and $y = w(z)$ or $z(w)$ for some $w \in \Lambda$.
3) $x \lhd z$ and $y = \lambda v.z$ for $v \in U$.

2.1.5 <u>Definition</u> (Simultaneous Substitution). The simultaneous substitution of $x_1, x_2, \ldots, x_n \in \Lambda$ for $u_1, u_2, \ldots, u_n \in U$ in $y \in \Lambda$, $\int_{x_1,x_2,\ldots,x_n}^{u_1,u_2,\ldots,u_n} y$, is defined inductively as:

Let $u = (u_1, u_2, \ldots, u_n)$ and $x = (x_1, x_2, \ldots, x_n)$.

1) If $y \in U$ and $y \neq u_i$ for all i then $\int_x^u y = y$.
2) If $y = u_i$ for $1 \leq i \leq n$, then $\int_x^u y = x_i$.
3) If $y = a(b)$ for $a, b \in \Lambda$, then $\int_x^u y = (\int_x^u a)(\int_x^u b)$.
4) If $y = \lambda v.z$ for $v \neq u_i$ $(i = 1, 2, \ldots, n)$, then if v is not free in any of the x_i's then $\int_x^u y = \lambda v.\int_x^u z$ otherwise $\int_x^u y = \lambda v'.\int_x^u (\int_{v'}^v z)$ where v' is the first variable, other than any u_i's or v in the enumeration of the variables in U such that v' does not occur free in y or z.
5) If $y = \lambda u_i.z$ for $1 \leq i \leq n$, then

$$\int_x^u y = \lambda u_i.\int_{x_1,x_2,\ldots,x_{i-1},x_{i+1},\ldots,x_n}^{u_1,u_2,\ldots,u_{i-1},u_{i+1},\ldots,u_n} y \quad .$$

2.1.6 Definition. Given $x, y, z \in \Lambda$, we say that x matches y except at occurrences of z in x if there exists $w \in \Lambda$ having free occurrences of $v \in U$ and $z_0 \in \Lambda$ such that

$$x = \int_z^v w \quad \text{and} \quad y = \int_{z_0}^v w \quad .$$

In this case, we say that z in x is homologous to z_0 in y.

Notational Convention. We will use the following notational abbreviation:

1) xy stands for $x(y)$.
2) $x_1x_2\cdots x_n$ stands for $((\cdots(x_1x_2)x_3)\cdots)x_n)$.
3) $\lambda s_1s_2\cdots s_n.x$ stands for $\lambda s_1.\lambda s_2.\cdots.\lambda s_n.x$.

So note that a λ-expression can generally be written as: $\lambda t_1t_2\cdots t_m.x_1x_2\cdots x_n$ for $t_1,t_2,\ldots,t_m \in U$ and $x_1,x_2,\ldots,x_n \in \Lambda$.

2.1.7 Definition (Conversion Rules). Let $\xi = \alpha$, β, η-red or η-ab. We will define $R_\xi \subseteq \Lambda \times \Lambda$ for each case of ξ. $(x,y) \in R_\xi$ is denoted by $x \xrightarrow{\xi} y$.

$(x,y) \in R_\xi$ if

I) a) (α-conversion) $\xi = \alpha$: $x = \lambda u.z$ and $y = \lambda v.\int_v^u z$ under the following restrictions:

i) v does not occur free in z.

ii) If $v \in \mathcal{B}(z)$, any free occurrence of u in z must not be in the scope of v.

b) (β-reduction) $\xi = \beta$: $x = (\lambda v.z)w$ and $y = \int_w^v z$.

c) (η-abstraction) $\xi = \eta$-ab: $y = \lambda v.xv$ where v does not occur free in x.

d) (η-reduction) $\xi = \eta$-red: $x = \lambda v.yv$ where v does not occur free in y.

or

II) y is derived from x by applying ξ-conversion (reduction) to a subexpression of x.

We define $\xrightarrow{CNV} \subseteq \Lambda \times \Lambda$ to be the reflective, transitive closure of $R_\alpha \cup R_\beta \cup R_{\eta\text{-red}} \cup R_{\eta\text{-ab}}$, i.e. $x \xrightarrow{CNV} y$ if and only if $x = y$ or there exist $x_1, x_2, \ldots, x_n \in \Lambda$ such that $x = x_1 \xrightarrow{\xi_1} x_2 \xrightarrow{\xi_2} \cdots \xrightarrow{\xi_{n-1}} x_n = y$ where $\xi_i = \alpha, \beta, \eta\text{-ab}$ or η-red.

2.1.8 <u>Definition</u>. a) A <u>β-redex</u> is a λ-expression in the form of $(\lambda v.x)y$. A λ-expression is said to have a β-redex if one of its subexpressions is a β-redex.

b) A β-redex y in $x \in \Lambda$ is said to be the <u>outermost-leftmost</u> <u>β-redex</u> if there is no β-redex w such that

i) $y \lhd w \lhd x$

or ii) $w \lhd a$, $y \lhd b$ and $ab \lhd x$.

c) Let $x \in \Lambda$ have $(\lambda v.y)z$ as its outermost-leftmost β-redex. The <u>outermost-leftmost</u> <u>β-reduction</u> to x is the replacement of $(\lambda v.y)z$ in x by $\int_z^v y$.

2.1.9 <u>Definition</u>. a) $y \in \Lambda$ is said to be in a <u>head</u> <u>normal</u> <u>form</u> if y is in the form of $\lambda s_1 s_2 \cdots s_m . v y_1 y_2 \cdots y_n$ for $s_1, s_2, \ldots, s_m, v \in U$ and $y_1, y_2, \ldots, y_m \in \Lambda$.

b) $y \in \Lambda$ is said to be <u>head normal</u> if there exists a sequence: $y = x_1 \xrightarrow{\beta} x_2 \xrightarrow{\beta} \cdots \xrightarrow{\beta} x_{n-1} \xrightarrow{\beta} x_n$ and x_n is in a head normal form. Here x_n is said to be a head normal form of y.

2.1.10 <u>Corollary</u>. Let $x \in \Lambda$ be head normal. If

$$x \xrightarrow{CNV} \lambda s_1 s_2 \cdots s_m . u x_1 x_2 \cdots x_n$$

and

$$x \xrightarrow{CNV} \lambda r_1 r_2 \cdots r_p . v y_1 y_2 \cdots y_q \quad ,$$

then 1) If u occurs free in x, then $u = v$.

2) If $u = s_i$ for $i \le m$, then $i \le p$ and $v = r_i$.

3) $m - n = p - q$.

<u>Proof</u>. See [21]. □

In Corollary 2.1.10 let

$$\text{head}(x) = \begin{cases} u \in U & \text{if } u \text{ is free in } x \\ i \in \mathbb{N} & \text{otherwise} \end{cases}$$

and

$$\text{index}(x) = m - n \ .$$

By the corollary, head(x) and index(x) are uniquely defined for $x \in \Lambda$ if x has a head normal form. We define the relationship $\sim \subseteq \Lambda \times \Lambda$ by:

<u>$x \sim y$</u> if

either neither x nor y has a head normal form

or index(x) = index(y) and head(x) = head(y).

2.1.11 Definition. $x \in \Lambda$ is said to be in a β-normal form if x has no β-redex as its subexpression. We say that $x \in \Lambda$ is β-normal if x can be reduced to a normal form by β-reductions.

Note that if a λ-expression is normal, it is also head-normal.

2.1.12 Theorem. If $x \in \Lambda$ is head-normal (normal), then there exists the following sequence:

$$x = x_0 \to x_1 \to x_2 \to \cdots \to x_n$$

where x_{i+1} is the result of the outermost-leftmost β-reduction applied to x_i for $n = 1,2,\ldots,n-1$ and x_n is in a head-normal form (normal form).

Proof. For the normal form case, see [5]. The proof is similar for the head normal case. □

To have a certain uniqueness for the head normal form, we define $x \xrightarrow[\beta h]{} y$ as follows: $x \xrightarrow[\beta h]{} y$ if there is a sequence $x = x_0 \to x_1 \to x_2 \to \cdots \to x_n = y$ such that x_{i+1} is the result of outermost-leftmost β-reduction to x_i. x_n is in a head normal form and x_i is not in a head normal form for $i \lneqq n$.

It is easy to see that if $x \xrightarrow[\beta h]{} y$ then y is uniquely determined by x.

2.1.13 Theorem (Scott). It is not decidable whether a λ-expression is normal or whether a λ-expression is head-normal.

Proof. See [7]. □

The following theorem is fundamental in the theory of the λ-calculus.

2.1.14 Theorem (Church-Rosser). Given $x, y_1, y_2 \in \Lambda$, if $x \xrightarrow{\text{CNV}} y_1$ and $x \xrightarrow{\text{CNV}} y_2$, then there exists $z \in \Lambda$ such that both $y_1 \xrightarrow{\text{CNV}} z$ and $y_2 \xrightarrow{\text{CNV}} z$.

Proof. See [2]. □

§2. Theory of Computation on Lattice Domains, D_∞ Model

Scott [14] proposed the following axioms that a mathematical model of computation ought to have:

Axiom 1: A domain D is a complete lattice. We denote $\cup^* D$ by $\top$ (top) and $\cup\emptyset$ by $\perp$ (bottom).

2.2.1 Definition. a) Let D be a partially ordered set. A subset $S \subseteq D$ is said to be directed if, for any finite subset F of S, there exists z in S such that

$$x \subseteq^* z \quad \text{for all} \quad x \in F .$$

b) A partially ordered set D is said to be directed-complete if all directed subsets of D have the least upper bound.

c) A function from a partially ordered set D_1 to another partially ordered set D_2 is said to be continuous if, for all directed sets $E \subseteq D_1$,

$$f(\cup E) = \cup\{f(x) \mid x \in E\} \quad .$$

f is said to be additive if

$$f(\cup S) = \cup\{f(x) \mid x \in S\}$$

for all subsets $S \subseteq D_1$.

As we see in Chapter 7, the completeness is not necessarily needed for the development of the theory in this thesis. At most, we would need a directed-complete partially ordered set with the least element $\perp$. Given two partially ordered sets D_1, D_2, we

*We use $\cup$ and $\subseteq$ instead of $\sqcup$ and $\sqsubseteq$ for typographical convenience.

denote the set of all continuous functions from D_1 to D_2 by $[D_1 \to D_2]$ (we denote all maps $D_1 \to D_2$ by $(D_1 \to D_2)$.)

2.2.2 Corollary (Scott). If D_1, D_2 are directed-complete (complete), then $[D_1 \to D_2]$ is also a directed complete (complete) latrice, where we define $\subseteq$ in $[D_1 \to D_2]$ by: $f \subseteq g$ if and only if $f(x) \subseteq g(x)$ for all $x \in D_1$.

Proof. See, for example, [12]. □

Axiom 2: A map from domain D_1 to domain D_2 is continuous.

2.2.3 Theorem (Scott). Let f be a continuous function over a directed-complete partially ordered set D. Then f has the least fixed point $\bigcup_{n=0}^{\infty} f^n(\perp)$.

Proof. See [12]. □

2.2.4 Definition. a) A subset G of a directed-complete subset D is said to be open if

1) For any $x \in G$, if $x \subseteq y$, then $y \in G$.

2) For any directed set $\mathcal{D} \subseteq D$, if $\cup\mathcal{D} \in G$ then $\mathcal{D} \cap G \neq \emptyset$.

b) For $x, y \in D$, we say $x \prec y$ (x is strictly less than y) if there is an open set G such that $y \in G$ and $G \subseteq \{z \mid x \subseteq z\}$.

c) A directed complete partially ordered set D is said to be continuous if, for all $x \in D$, $x = \cup\{y \mid y \prec x\}$.

Note that a domain has a T_0-topology induced by the open sets defined above. Continuous mappings are continuous in this topological sense.

Axiom 3: A domain is a continuous lattice.

The last axiom is on the computability:

Axiom 4: A domain D has a subset E of the following properties:

1) The cardinality of E is at most denumerable and the elements of E are recursively enumerable.

2) For any $x \in D$, $x = \cup\{y \in E \mid y \prec x\}$.

3) For all $e_1, e_2 \in E$, $e_1 \cup e_2$ and $e_1 \subseteq e_2$ are computable.

Next, we state the construction of D_∞-lattice. We shall confine ourselves to the description of the properties of D_∞ that are needed in our discussion in the subsequent chapters. For the complete presentation of D_∞, see [12].

Construction of D_∞

We want to have a lattice domain D with the property $D \approx [D \to D]$. Let D_0 be any complete lattice. (In fact, a directed-complete partially ordered set is good enough for our purpose, but for simplicity, we assume the completeness.)

Let $D_1 = [D_0 \to D_0]$, $D_2 = [D_1 \to D_1], \ldots, D_n = [D_{n-1} \to D_{n-1}], \ldots$. Note that each D_n is a complete lattice. We define (i_n, j_n) for each n such that

1) $i_n \colon D_n \to D_{n+1}$, $j_n \colon D_{n+1} \to D_n$

2) i_n, j_n are additive and $j_n \circ i_n = 1_{D_n}$ and $i_n \circ j_n \subseteq 1_{D_{n+1}}$ (so i_n is one-to-one and j_n is onto).

Definition of (i_n,j_n)

$$n = 0 \qquad \begin{array}{ll} i_o(a) = \lambda\beta \in D_o{:}a & \text{for each } a \in D_o \\ j_o(x) = x(\bot_{D_o}) & \text{for each } x \in D_1 \end{array}$$

Suppose that we have defined (i_n,j_n) for $n \leq k-1$ $(k \geq 1)$. We define (i_k,j_k)

$$D_{k-1} \underset{j_{k-1}}{\overset{i_{k-1}}{\rightleftarrows}} D_k \underset{j_k}{\overset{i_k}{\rightleftarrows}} D_{k+1}$$

$$i_k(x) = i_{k-1} \circ x \circ j_{k-1} \quad \text{for all} \quad x \in D_k$$

$$j_k(y) = j_{k-1} \circ y \circ i_{k-1} \quad \text{for all} \quad y \in D_{k+1}$$

It is easy to see that (i_k,j_k) satisfies the properties 1) and 2) by induction on k.

We define $D_\infty = \{(x_o,x_1,\ldots,x_n,\ldots) \mid x_n \in D_n,\ x_n = j_n(x_{n+1})\}$, where, for x, $y \in D_\infty$, $x \subseteq y$ if and only if $x_i \subseteq y_i$ for all i.

Embedding of D_n in D_∞

We define $\phi_{nm}\colon D_m \to D_n$ as follows:

$$\phi_{nm} = \begin{cases} i_{n-1} \circ i_{n-2} \circ \cdots \circ i_{m+1} \circ i_m & \text{if} \quad m < n \\ 1_{D_m} & \text{if} \quad m = n \\ j_n \circ j_{n-1} \circ \cdots \circ j_m \circ j_{m+1} & \text{if} \quad n < m\ . \end{cases}$$

Now we embed D_n into D_∞ by

$$E_n\colon D_n \to D_\infty \qquad x \mapsto \langle \phi_{On}(x), \phi_{1n}(x), \ldots, \phi_{(n-1)n}(x), x, \phi_{(n+1)n}(x), \ldots \rangle$$

By defining $\bar{D}_n = E_n(D_n) \subseteq D_\infty$,

1) $E_n\colon D_n \to \bar{D}_n$ is one-to-one and continuous

2) $\bar{D}_o \subseteq \bar{D}_1 \subseteq \bar{D}_2 \subseteq \cdots$

Conversely, the projection $\pi_n\colon D_\infty \to \bar{D}_n$ is defined by:

$$\pi_n\colon \langle x_0, x_1, x_2, \ldots, x_n, \ldots\rangle \in D_\infty$$
$$\mapsto \langle x_0, x_1, \ldots, x_n, \phi_{(n+1)n}(x_n), \phi_{(n+2)n}(x_n), \ldots\rangle \in \bar{D}_n.$$

Also we define $P_n\colon D_\infty \to D_n$ by

$$P_n\colon \langle x_0, x_1, \ldots, x_n, \ldots\rangle \mapsto x_n \quad .$$

It is easy to see that $x = \bigcup_{n=0}^{\infty} \pi_n(x)$ for all $x \in D_\infty$.

Isomorphism $D_\infty \approx [D_\infty \to D_\infty]$

We define

$$\Phi\colon D_\infty \to [D_\infty \to D_\infty]$$
$$\Psi\colon [D_\infty \to D_\infty] \to D_\infty$$

by: For all $x \in D_\infty$

$$\Phi(x)(y) = \bigcup_{n=0}^{\infty} E_n(P_{n+1}(x)(P_n(y))) \quad \text{for all} \quad y \in D_\infty \quad .$$

For all $f \in [D_\infty \to D_\infty]$

$$\Psi(f) = \langle f_0, f_1, f_2, \ldots, f_n, \ldots\rangle$$

where

$$f_0 = P_0(f(\bot))$$
$$f_n = \lambda x \in D_{n-1}\colon P_{n-1}(f(E_{n-1}(x))) \quad \text{for} \quad n \geqq 1 \ .$$

In a straightforward way, we can verify that Φ, Ψ are additive, $\Psi\circ\Phi = 1_{D_\infty}$ and $\Phi\circ\Psi = 1_{[D_\infty \to D_\infty]}$.

We can now define the application of x to y for $x, y \in D_\infty$ by $\Phi(x)y$. We denote this by $x(y)$. By the definition of Φ,

$$\Phi(x)(y) = \bigcup_{n=1}^{\infty} \pi_n(x)(\pi_{n-1}(y)) .$$

Lastly we list the important properties of D_∞ projections:

2.2.5 <u>Theorem</u> (Scott). 1) $\pi_m \subseteq \pi_n \subseteq 1_{D_\infty}$ for $m \leq n$

2) $\bigcup_{n=0}^{\infty} \pi_n = 1_{D_\infty}$

3) $\pi_n \circ \pi_m = \pi_{\min(n,m)}$

4) $\pi_n(x)(y) = \pi_n(x)(\pi_{n-1}(y)) = \pi_{n-1}(x(\pi_{n-1}(y)))$

5) $\pi_0(x)(y) = \pi_0(x) = \pi_0(x(\perp))$

§3. Wadworth's Model Theory of λ-Calculus in D_∞

In this section, we state the results due to Wadsworth [21,22]. As we have seen in the last section: $D_\infty \underset{\Psi}{\overset{\Phi}{\rightleftarrows}} [D_\infty \to D_\infty]$ for continuous Φ, Ψ satisfying $\Psi \circ \Phi = 1_{D_\infty}$ and $\Phi \circ \Psi = 1_{[D_\infty \to D_\infty]}$. This property of D_∞ can be characterized in the following way:

1) Extensionality: $x(z) \sqsubseteq y(z)$ for all $z \in D_\infty$ iff $x \sqsubseteq y$

so, particularly, $x(z) = y(z)$ for all $z \in D_\infty$ iff $x = y$

2) Comprehension: If $\cdots x \cdots$ is an expression taking values on D_∞ which is continuous in the variable x as x ranges over D_∞, then there is $f \in D_\infty$ such that

$$f(a) = \cdots a \cdots \quad \text{for all} \quad a \in D_\infty \quad .$$

2.3.1 Definition (Wadsworth). Let EN be the set of all mappings from the set of the variables U to D_∞. The semantic function $\mathbb{V}: \Lambda \to (EN \to D_\infty)$ is defined as follows:

1) For $v \in U$ and $\rho \in EN$, $\mathbb{V}[\![v]\!]\rho = \rho(v)$.

2) For $x(y) \in \Lambda$ and $\rho \in EN$, $\mathbb{V}[\![x(y)]\!]\rho = \mathbb{V}[\![x]\!]\rho(\mathbb{V}[\![y]\!]\rho)$.

3) For $\lambda v.x \in \Lambda$ and $\rho \in EN$,

$$\mathbb{V}[\![\lambda v.x]\!]\rho = \lambda\beta \in D_\infty : \mathbb{V}[\![x]\!]\rho[v/\beta]$$

where $\rho[v/\beta]$ is defined by

$$\rho[v/\beta](u) = \begin{cases} \rho(u) & \text{if } u \neq v \\ \beta & \text{if } u = v \ . \end{cases}$$

Since $\mathbb{V}[\![x]\!]\rho[v/\beta]$ is continuous in the variable β, $\lambda\beta \in D_\infty: \mathbb{V}[\![x]\!]\rho[v/\beta]$ is a member of D_∞ due to the comprehension of D_∞.

2.3.2 Proposition (Wadsworth). If $x \xrightarrow{CNV} y$ for $x, y \in \Lambda$, then $\mathbb{V}[\![x]\!]\rho = \mathbb{V}[\![y]\!]\rho$ for all $\rho \in EN$.

Proof. The result is obvious for the α-conversion and β-reduction. The η-conversions preserve the D_∞ value due to the extensionality of D_∞. □

2.3.3 Definition. We say $x \underset{D_\infty}{\subseteq} y$ for $x, y \in \Lambda$ if $\mathbb{V}[\![x]\!]\rho \subseteq \mathbb{V}[\![y]\!]\rho$ for all $\rho \in EN$. Similarly $x \underset{D_\infty}{=} y$ if $\mathbb{V}[\![x]\!]\rho = \mathbb{V}[\![y]\!]\rho$ for all $\rho \in EN$.

2.3.4 Corollary. $\underset{D_\infty}{\subseteq}$ is reflective and transitive.

Proof. Obvious. □

However $\underset{D_\infty}{\subseteq}$ is, obviously, not antisymmetric, so $\underset{D_\infty}{\subseteq}$ is not a partial ordering.

We first show that Λ is not trivial in D_∞, namely, Λ is not mapped into one element in D_∞.

2.3.5 Proposition. $K \underset{D_\infty}{\neq} H$ and $I \underset{D_\infty}{\neq} \bot$.

Proof. See [21]. □

2.3.6 Theorem (Wadsworth). Let $I = \lambda x.x$ and $J = Y(\lambda fxy.x(fy))$. Then $I \underset{D_\infty}{=} J$.

Proof. See [22] for the proof based on the type construction. Also see Example 4.3.4. □

Since I is normal and J is non-normal, I and J are not convertible to each other. So this shows that $\underset{D_\infty}{=}$ is strictly larger than $\xrightarrow{CNV}$, i.e. $\xrightarrow{CNV} \subsetneq \bar{\bar{D}}_\infty$.

The next theorem shows that Curry's Y gives the least fixed point operator in D_∞.

2.3.7 <u>Theorem</u> (Park). $Y \underset{D_\infty}{=} \lambda f \in D_\infty : \bigcup_{n=0}^{\infty} f^n(\perp)$.

<u>Proof</u>. See [12], also Corollary 4.2.3. $\square$

We can introduce $\cup, \cap$ operations in Λ as follows: Given $S \subseteq \Lambda$, $\cup S$ is a syntactic object with the semantic value in D_∞ of:

$$\mathbb{V}[\![\cup S]\!]\rho = \cup\{\mathbb{V}[\![x]\!]\rho \mid x \in S\}$$

for $\rho \in EN$.

We define $\cap S$ in the similar manner.

<u>λ-Ω-Calculus</u>

It is convenient to have a syntactical symbol in Λ that represents $\perp$. The λ-Ω-expressions, Λ_Ω, are formed according to the following rules:

0) Ω is in Λ_Ω.

1)-3) Same as Definition 2.1.1.

Semantic function $\mathbb{V}$ is $\perp$ on Ω, i.e.

$$\mathbb{V}[\![\Omega]\!]\rho = \perp \text{ for all } \rho \in EN .$$

We include two conversion rules for Λ_Ω in addition to those for Λ.

1) $\lambda v.\Omega \to \Omega$ for $v \in U$

2) $\Omega(x) \to \Omega$ for $x \in \Lambda_\Omega$

These rules are semantically sound since $\lambda\beta \in D_\infty: \perp = \perp = \perp(a)$ for $a \in D_\infty$.

Type Assignments of λ-Expressions

This part of the section is needed to prove Lemma 4.2.1. For the details of the discussion, we refer to [23].

As a member of D_∞, each λ-expression has a component in each $\tilde{D}_n$. The typed λ-expressions defined below are introduced to, in a sense, approximate the components of λ-expressions in D_∞.

2.3.8 Definition (Typed λ-expressions).

Syntax of Λ^t

The typed λ-expression, Λ^t, is the set of all expressions that are formed by the rules below:

1) For $v \in U$, $v^{(n)} \in \Lambda^t$ for each $n \in \mathbb{N}$.

2) If $x, y \in \Lambda^t$, $(x(y))^{(n)} \in \Lambda^t$ for $n \in \mathbb{N}$. $(x(y))^{(n)}$ is abbreviated as $(xy)^{(n)}$.

3) For $v \in U$, $x \in \Lambda^t$, $(\lambda v.x)^n \in \Lambda^t$ for $n \in \mathbb{N}$.

4) $\Omega^{(n)} \in \Lambda^t$ for $n \in \mathbb{N}$.

Semantics of Λ^t

The semantic function, $\mathbb{U}: \Lambda^t \to (EN \to D_\infty)$ is defined as:

1) $\mathbb{U}[\![v^{(n)}]\!]\rho = \pi^n(\rho(v))$

2) $\mathbb{U}[\![(xy)^{(n)}]\!]\rho = \pi^n(\mathbb{U}[\![x]\!]\rho(\mathbb{U}[\![y]\!]\rho))$

3) $\mathbb{U}[\![(\lambda v.x)^{(n)}]\!]\rho = \pi^n(\lambda\beta \in D_\infty: \mathbb{U}[\![x]\!]\rho[v/\beta])$

We define several auxiliary functions:

$\underline{type}$: $\Lambda^t \to \mathbb{N}$ is a mapping.

1) $\underline{type}(v^{(n)}) = n$

2) $\underline{type}((xy)^{(n)}) = n$

3) $\underline{type}((\lambda v.y)^{(n)}) = n$

$\underline{W}$: $\Lambda^t \to \Lambda_\Omega$ is a mapping defined as:

1) $\underline{W}(v^{(n)}) = v$

2) $\underline{W}((xy)^{(n)}) = \underline{W}(x)\underline{W}(y)$

3) $\underline{W}((\lambda v.z)^{(n)}) = \lambda v.\underline{W}(z)$

4) $\underline{W}(\Omega^{(n)}) = \Omega$

i.e. $\underline{W}(x)$ is the λ-Ω-expression obtained from $x \in \Lambda^t$ by deleting all type superfixes of x.

$\underline{T}$: $\Lambda_\Omega \to P(\Lambda^t)$ (power set of Λ^t) is defined by:

$$\underline{T}(x) = \{y \mid x = \underline{W}(y)\} \subseteq \Lambda^t \quad ,$$

i.e. $\underline{T}(x)$ is the set of all typed λ-expressions generated from x by putting a type superfix to each subexpression of x.

2.3.9 Lemma. $x \underset{D_\infty}{=} \cup \underline{T}(x)$.

Proof. See [23]. □

Notation

For $x \in \Lambda^t$ and $n \in \mathbb{N}$, let $[x]_n$ be the typed λ-expression determined by the following rule:

$$[x]_n = x \quad \text{if } \underline{type}(x) \leqq n$$

$$[x]_n = y^{(n)} \quad \text{if } \underline{type}(x) > n,$$

$$\text{where } x \equiv y^{(m)} \text{ for } m = \underline{type}(x) \ .$$

Typed Conversions

In the similar manner to the conversion rules in the ordinary λ-calculus, we define <u>typed</u> <u>conversion</u> <u>rules</u> for Λ^t.

2.3.10 <u>Definition</u> (Typed Substitution). For $v \in V$ and $x, y \in \Lambda^t$, we define $\int_y^v x$ to be:

1) If $x = u^{(n)}$ for $u \in V$, $u \neq v$, then $\int_y^v x = u^{(n)}$.

2) If $x = v^{(n)}$, then $\int_y^v x = [y]_n$.

3) If $x = \Omega^{(n)}$, then $\int_y^v x = \Omega^{(n)}$.

4) If $x = (ts)^{(n)}$, then $\int_y^v x = \left((\int_y^v t)(\int_y^v s)\right)^{(n)}$.

5) If $x = (\lambda u.w)^{(n)}$ for $u \neq v$, then $\int_y^v x = \left\{\lambda u.(\int_y^v w)\right\}^{(n)}$

if u does not occur free in w. If u occurs free in w, $\int_y^u x = \left\{\lambda u'.(\int_y^v w')\right\}^{(n)}$ where w' is w with each u replaced by u' and u' is the first variable other than u or v in U such that u' does not occur free in w or y.

6) If $x = (\lambda v.w)^{(n)}$, then $\int_y^v x = (\lambda v.w)^{(n)}$.

2.3.11 <u>Lemma</u> (Wadsworth). For $v \in V$, $x, y \in \Lambda^t$ and $\rho \in EN$,

$$\mathbb{U}[\![\int_y^v x]\!]\rho = \mathbb{U}[\![x]\!]\rho[v/\mathbb{U}[\![y]\!]\rho] \quad .$$

<u>Proof</u>. See [23]. □

Typed β-Reduction

$$((\lambda v.x)^{(i)}y)^{(j)} \xrightarrow{t\beta} [\int_{[y]_{i-1}}^v x]_{\min(i-1,j)} \quad \text{if } i > 0$$

$$((\lambda v.x)^{(0)}y)^{j} \xrightarrow{t\beta} [\int_{\Omega^{(0)}}^{v} x]_{0}$$

We extend $t\beta$ in the same manner as the non-typed case, i.e., $x \xrightarrow{t\beta} y$ if y is the result of applying several typed β-reductions to some subexpressions of x.

<u>Typed α-conversion</u>

Given $(\lambda v.x)^{(n)} \in \Lambda^{t}$ and let $v^{(n_1)}, v^{(n_2)}, \ldots, v^{(n_k)}$ be all the occurrences of v in x. Then

$$(\lambda v.x)^{(n)} \xrightarrow{t\alpha} \left(\lambda u. \int_{u^{(\max_i[n_i])}}^{v} x\right)^{(n)}$$

For $x, y \in \Lambda^{t}$, $x \xrightarrow{t\alpha} y$ if y is derived from x by applying several typed α-conversions to some subexpressions of x.

<u>Typed η-abstraction</u>

Let $\text{type}(x) = n$ for $x \in \Lambda^{t}$.

$$x \xrightarrow{t\eta\text{-ab}} (\lambda t.(xt^{(n-1)})^{(n-1)})^{(n)} \quad \text{if } n \geq 1$$
$$x \xrightarrow{t\eta\text{-ab}} (\lambda t.(x\Omega^{(0)})^{(0)})^{(0)} \quad \text{if } n = 0$$

For $x, y \in \Lambda^{t}$, $x \xrightarrow{t\eta} y$ if y is derived from x by applying several typed η-abstraction to some subexpressions of x.

2.3.12 <u>Theorem</u> (Wadsworth). For $x, y \in \Lambda^{t}$, if x and y are type-convertible to each other, then $x \underset{D_\infty}{=} y$.

<u>Proof</u>. See [23]. □

2.3.13 Lemma (Wadsworth). Given any $x \in \Lambda^t$, then there is a typed β-reduction sequence: $x = x_1 \to x_2 \to \cdots \to x_n$ such that x_n has no typed β-redex.

Proof. See [23]. □

2.3.14 Lemma (Wadsworth). Let $x_1 \to x_2 \to \cdots \to x_n$ be a sequence of typed β-reductions for $x_1, x_2, \ldots, x_n \in \Lambda^t$. Then there exists an ordinary β-reduction sequence $y_1 \to y_2 \to \cdots \to y_n$ where y_i matches $\underline{W}(x_i)$ except at occurrences of Ω in $\underline{W}(x_i)$.

Proof. See [23]. □

Notion of Reduced Approximant

Given $x \in \Lambda$, $\varepsilon \in \Lambda_\Omega$ is said to be a direct approximant of x if ε has no β-redex and ε matches x except at occurrences of Ω in ε. For example, a λ-Ω-expression, ε, that is obtained from x by replacing each β-redex in x by Ω is a direct approximant of x.

$\varepsilon \in \Lambda_\Omega$ is said to be a reduced approximant of x if ε is a direct approximant of x itself or of some $y \in \Lambda$ that is β-reducible from x (i.e. there is a β-reduction sequence from x to y).

For $x \in \Lambda$, we denote the set of all reduced approximants of x by $A(x)$.

2.3.15 Theorem (Wadsworth). For any $x \in \Lambda$, $x \underset{D_\infty}{=} \cup A(x)$.

Proof. See [23]. □

From this theorem, the following theorem is directly deduced.

2.3.16 <u>Theorem</u> (Wadsworth). If $x \in \Lambda$ is not head normal, $x \underset{D_\infty}{=} \perp$.

<u>Proof</u>. Since any $y \in \Lambda$ that is β-reducible from x is in a form $\lambda s_1 s_2 \cdots s_m.(\lambda v.w)x_1 x_2 \cdots x_n$, its direct approximant is:

$$\lambda s_1 s_2 \cdots s_m.\Omega x_2 \cdots x_n \to \Omega \quad .$$

So $A(x) = \{\Omega\}$. Thus $x \underset{D_\infty}{=} \perp$. □

Conversely

2.3.17 <u>Theorem</u> (Wadsworth). If $x \in \Lambda$ is head normal, $x \underset{D_\infty}{\neq} \perp$.

<u>Proof</u>. Let $\bar{x} = \lambda s_1 s_2 \cdots s_m.vx_1 x_2 \cdots x_n$ be a head normal form of x. Let $y = \lambda r_1 r_2 \cdots r_m.I$. Then

$$\int_y^v \bar{x} s_1 s_2 \cdots s_m \xrightarrow{\beta} I \underset{D_\infty}{\neq} \perp \quad .$$

So under some environment ρ, $\mathbb{V}[\![x]\!]\rho = \perp$. □

2.3.18 <u>Corollary</u> (Wadsworth). For $x \in \Lambda$, $x \underset{D_\infty}{=} \perp$ if and only if x has no head normal form. □

This corollary implies that we can replace any non-head normal subexpression of $x \in \Lambda$ by <u>spl</u> without affecting the D_∞-value of x since <u>spl</u> $\underset{D_\infty}{=} \Omega$. Hereafter we take the following convention. <u>If Ω is regarded as a member of Λ, it stands for spl.</u>

We introduce the following non-effective conversion rule to Λ.

Ω-conversion

$(x,y) \in R_\Omega$ or $x \xrightarrow{\Omega} y$ if y derives from x by replacing some subexpressions of x that have no head normal form by Ω.

We define $\approx$ as the reflective, transitive and symmetric closure of $R_\alpha \cup R_\beta \cup R_\xi \cup R_\Omega$. We conclude that

$$\xrightarrow{CNV} \subsetneq \approx \subsetneq \underset{D_\infty}{=} \quad .$$

$\xrightarrow{CNV} \subsetneq \approx$ since, for example, $(\lambda x.xx)(\lambda x.xx) \approx (\lambda x.xxx)(\lambda x.xxx)$ though it is not that $(\lambda x.xx)(\lambda x.xx) \xrightarrow{CNV} (\lambda x.xxx)(\lambda x.xxx)$.

On the other hand $\approx \subsetneq \underset{D_\infty}{=}$ since $I \underset{D_\infty}{=} J$ although it is not the case that $I \approx J$. $\underset{D_\infty}{=}$ will be characterized in Chapter 4.

CHAPTER 3

INFINITE NORMAL FORMS FOR λ-CALCULUS

We formalize the <Infinite Expansions> of programs in the domain of the λ-expression -- called C-function. We show that the C-function can be regarded as an extension of the conventional normal forms. Böhms Theorem on the normal expressions is extended to the general expressions via the C-function.

§1. Pedigree

First, we introduce an infinite set which will be used to characterize the behavior of λ-expressions.

3.1.1 Definition. Pedigree, Δ, is the set $\{0\} \cup \{(n_1, n_2, \ldots, n_k) \mid k, n_j \in \mathbb{N}\}$. There is a natural partial order, $<$, in Δ defined by:

for $\delta_1, \delta_2 \in \Delta$, $\delta_1 < \delta_2$ if and only if

either 1. $\delta_1 = 0$

or 2. $\delta_1 = (m_1, m_2, \ldots, m_i)$ and $\delta_2 = (n_1, n_2, \ldots, n_j)$

where $i < j$ and $m_1 = n_1, \ldots, m_i = n_i$.

We say $\delta_1 \leq \delta_2$ if $\delta_1 = \delta_2$ or $\delta_1 < \delta_2$, i.e. $\leq$ means "is a prefix of."

3.1.2 Definition. Given $\delta \in \Delta$, $|\delta|$, length of δ, is defined by:

$$|\delta| = 0 \text{ if } \delta = 0$$

$$|\delta| = k \text{ if } \delta = (n_1, n_2, \ldots, n_k) \quad .$$

3.1.3 Definition. Map $\mathrm{Pr}\colon \Delta \to \Delta$ is defined by:

$$\mathrm{Pr}(\delta) = \begin{cases} \text{undefined} & \text{if } \delta = 0 \\ 0 & \text{if } |\delta| = 1 \\ (n_1, n_2, \ldots, n_{k-1}) & \text{if } \delta = (n_1, n_2, \ldots, n_k) \end{cases} \quad .$$

3.1.4 Definition. Given $\delta \in \Delta$ and a positive integer m,

$$\delta \circ m = (m) \qquad \text{if } \delta = 0$$

$$\delta \circ m = (n_1, n_2, \ldots, n_k, m) \quad \text{if } \delta = (n_1, n_2, \ldots, n_k) \quad .$$

§2. Idea of Infinite Expansion

Before going to the formal definition of C-functions, we try here to illustrate informally the idea of infinite normal form.

3.2.1 Example. R is defined by $Y(\lambda f\lambda x\lambda y.x(fy)(fy((\lambda x.xx)(\lambda x.xxx))))$ where Y is the fixed point operator. Since we know that $(\lambda x.xx)(\lambda x.xxx)$ does not have a head normal form, we replace it by Ω. The recursive definition of R is:

$$R \xrightarrow{\beta} \lambda x\lambda y.x(Ry)(Ry\Omega) \quad .$$

Arrange it in the form:

$$\begin{array}{ccc} & \lambda x\lambda y.x & \\ / & & \backslash \\ Ry & & Ry\Omega \end{array}$$

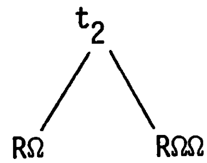
with operands below the leading operator. Renaming the bound variables according to their position:

$$\begin{array}{ccc} & \lambda t_1\lambda t_2.t_1 & \\ / & & \backslash \\ Rt_2 & & Rt_2\Omega \end{array}$$

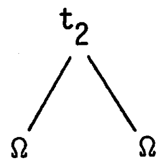
$Rt_2 \xrightarrow{\beta} \lambda y.t_2(Ry)(Ry\Omega)$ and, so, the left sub-tree is depicted as:

$$\begin{array}{ccc} & \lambda t_{11}.t_2 & \\ / & & \backslash \\ Rt_{11} & & Rt_{11}\Omega \end{array}$$

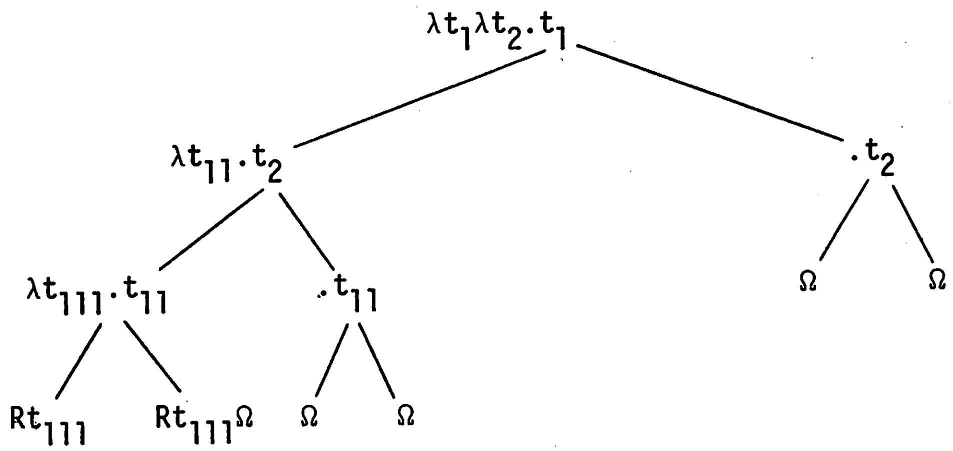
$Rt_2\Omega \xrightarrow{\beta} t_2(R\Omega)(R\Omega\Omega)$, so the right sub-tree is depicted as:



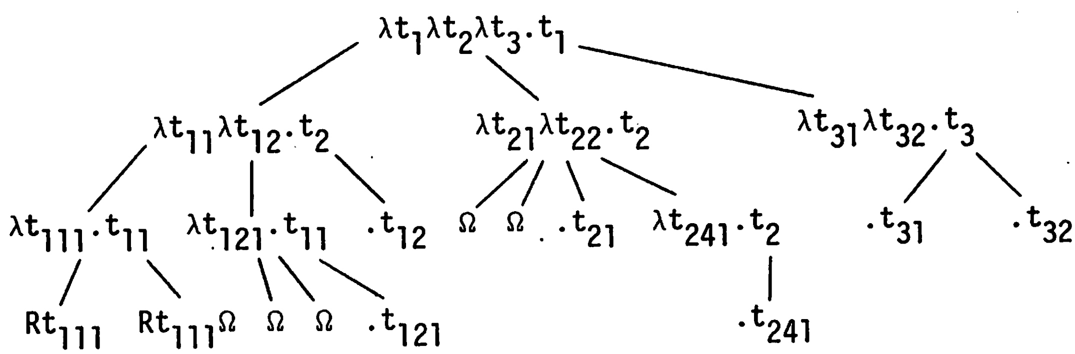
However, since $R\Omega \xrightarrow{\beta} \Omega$, this sub-tree becomes:



Applying β-reductions further, we have



Now, several applications of η-abstraction lead us to:



In this way, we can expand any expression infinitely by applying η-abstractions and β-reductions. In the illustrations, the arrangement of the head normal forms should be noted. Each head variable is situated higher than its operands since it is dominant over them. This situation is similar to the program of the form begin A;B end. A can be said to be dominant over B since execution may never reach B depending on the control structure in A. This point will be further discussed in Chapter 5.

Here note that there are four operations involved in the process of expansion:

a) β-reduction

b) Ω-conversion

c) Renaming the bound variables according to their position

d) η-abstraction

Also note that there are two important bases to consider this process.

1) We consider the λ-calculus with η-convertibility. Operation d) depends upon this assumption.

2) The head normality is an undecidable property. Thus operation b) is not effective and the functions and L, $\hat{C}$ and C to be defined in this chapter are non-computable.

§3. L-function

To make the argument easier in the rest of this chapter, we make the following conventions:

Let U be the enumerably infinite set of the variables. We take two mutually disjoint subsets F and T_Δ of U and set $V = F \cup T_\Delta$, where

and
$$F = \{f_i \mid i = 1,2,\ldots\}$$
$$T_\Delta = \{t_\delta \mid \delta \in \Delta - \{0\}\} \quad .$$

We assume, in the rest of this chapter, that <u>if any given expression has some occurrences of a free variable, it is one of the f_i's in</u> F. Our intention is to convert any given expression into one whose bound variables are in T_Δ by applying α-conversions. We will be using z to represent a variable which is either in F or T_Δ.

Let $\Sigma = \{(z,m,n) \mid z \in U,\ m,\ n \in \mathbb{N} \cup \{0\}\}$ and Ω be a symbol not in Σ.

An auxiliary function $h\colon \Lambda \to \Sigma \cup \{\Omega\}$ is defined by:

$$h(x) = \begin{cases} \Omega & \text{if } x \text{ has no head normal form} \\ (z,m,n) & \text{if } x \xrightarrow{\beta} \lambda x_1 \cdots x_m . zX_1 \cdots X_n \end{cases} \quad .$$

It is easy to see that h is well-defined. Note that h is not a computable function since the existence of a head normal form is not recursively decidable.

3.3.1 <u>Definition</u> (L-function). We define $L\colon \Lambda \to (\Delta \to \Lambda)$ inductively as follows:

Given $x \in \Lambda$, assume that any t_δ in T_Δ does not appear in x (by applying α-conversions if necessary).

Step 0.

$$L(x,0) = \begin{cases} \Omega & \text{if } h(x) = \Omega \quad \text{(Operation b)} \\ \lambda t_1 \cdots t_m . zX_1 \cdots X_n & \text{if } h(x) \neq \Omega \quad \text{(Operation a, c)} \end{cases}$$

and $x \xrightarrow[\beta h]{} \lambda s_1 s_2 \cdots s_m . wX_1' X_2' \cdots X_n'$

and $zX_1 X_2 \cdots X_n = \int_{t_1,t_2,\ldots,t_m}^{s_1,s_2,\ldots,s_m} wX_1' X_2' \cdots X_n'$

Step δ. Suppose that we have defined $L(x,\delta')$ for all $\delta' \in \Delta$ such that $\delta' \leq \delta$. We are to define $L(x,\delta \circ i)$ for each $i \in \mathbb{N}$.

Case I. If $L(x,\delta) = \Omega$ then $L(x,\delta\circ i) = \Omega$ for all $i \in \mathbb{N}$.

Case II. If $L(x,\delta) = \lambda t_{\delta\circ 1} t_{\delta\circ 2} \cdots \lambda t_{\delta\circ m} . zX_1 X_2 \cdots X_n$ then

(i) If $i \leqq n$ then

(a) $L(x,\delta\circ i) = \Omega$ if $h(X_i) = \Omega$ (Operation b)

(b) $L(x,\delta\circ i) = \lambda t_{\delta\circ i\circ 1} t_{\delta\circ i\circ 2} \cdots t_{\delta\circ i\circ p} . vY_1 Y_2 \cdots Y_q$ (Operation a, c)

if $X_i \xrightarrow[\beta h]{} \lambda r_1 r_2 \cdots r_p . uY_1' Y_2' \cdots Y_q'$

and $zY_1 Y_2 \cdots Y_q = \int_{t_{\delta\circ i\circ 1}, t_{\delta\circ i\circ 2}, \ldots, t_{\delta\circ i\circ p}}^{r_1, r_2, \ldots, r_p} uY_1' Y_2' \cdots Y_q'$

(ii) If $i > n$ then $L(x,\delta\circ i) = t_{\delta\circ(m-n+i)}$. (Operation d)

We should note that in (ii) of Case II above that we are applying η-abstractions. Also each head variable of $L(x,\delta)$ is in F if it is free in x or in T_Δ if it is bound in x.

3.3.2 Example. We look at $L(R,\delta)$ for R defined in §2.

$$L(R,0) = \lambda t_1 \lambda t_2 . t_1(Rt_2)(Rt_2\Omega)$$
$$L(R,1) = \lambda t_{11} . t_2(Rt_{11})(Rt_{11}\Omega)$$
$$L(R,2) = t_2(R\Omega)(R\Omega\Omega)$$
$$L(R,3) = t_3$$
$$\vdots$$
$$L(R,i) = t_i$$
$$\vdots$$
$$L(R,11) = \lambda t_{111} . t_{11}(Rt_{111})(Rt_{111}\Omega)$$
$$L(R,12) = t_{11}\Omega\Omega$$
$$L(R,13) = t_{12}$$
$$L(R,14) = t_{13}$$
$$\vdots \qquad \vdots$$
$$L(R,1i) = t_{1(i+1)}$$
$$\vdots$$
$$L(R,21) = \Omega$$
$$L(R,22) = \Omega$$
$$L(R,23) = t_{21}$$
$$\vdots$$
$$L(R,2i) = t_{2(i-2)}$$
$$\vdots$$

3.3.3 Corollary. If $x \approx y$, then $L(x,\delta) \approx L(y,\delta)$ for all $\delta \in \Delta$.

§4. C-functions

Now we are ready to present the definition of C-functions, $\hat{C}$ and C. $\hat{C}$ is not essentially necessary to state our result, but, $\hat{C}$ gives a way to simplify our discussion.

3.4.1 Definition ($\hat{C}$-function). Let $S = \{(z,k) \mid k \in \mathbb{Z},\ z \in F \cup T_\Delta\} \cup \{\omega\}$. We define $\hat{C}: \Lambda \to (\Delta \to S)$ by:

$$\hat{C}(x,\delta) = \begin{cases} \omega & \text{if } L(x,\delta) = \Omega \\ (z,k) & \text{if } h(L(x,\delta)) = (z,m,n) \text{ and } k = m-n . \end{cases}$$

We note that if $\hat{C}(x,\delta) = (z,k)$ for $x \in \Lambda$ and $\delta \in \Delta$, then there exists a positive integer M such that

$$\hat{C}(x,\delta \circ N) = (t_{N+k},0) \quad \text{for all } N \geqq M.$$

Although $\hat{C}$ is defined as above, is the second component, k, of $C(x,\delta)$ is not necessary to uniquely specify a λ-expression. Thus we define C, simplified version of $\hat{C}$.

3.4.2 Definition (C-function). C is a function $\Lambda \to (\Delta \to V \cup \{\omega\})$ defined by:

$$C(x,\delta) = \begin{cases} z & \text{if } \hat{C}(x,\delta) = (z,k) \\ \omega & \text{if } \hat{C}(x,\delta) = \omega . \end{cases}$$

3.4.3 Corollary. If $x \approx y$, then $\hat{C}(x) = \hat{C}(y)$ and $C(x) = C(y)$.

We will now state the theorem which plays the central role in this thesis. Proof of this theorem was essentially given in Böhm [4] (also in Wadsworth [21]), however since the arguments used in the proof are fundamental in this thesis and that it is not yet widely publicized, we present a complete proof for the theorem, hopefully, with notational improvements.

Firstly, the theorem is stated using $\hat{C}$ and thereafter, it will be modified for C.

3.4.5 Lemma. Given x and y in Λ, suppose that, for $\delta_0 \in \Delta$, $\hat{C}(x,\delta) = \hat{C}(y,\delta)$ for all δ such that $\delta < \delta_0$, and that

$$\hat{C}(x,\delta_0) = (u,i)$$
$$\hat{C}(y,\delta_0) = (v,j)$$

where $(u,i) \neq (v,j)$. Then for arbitrary a, b in D_∞ we can choose $e_1,e_2,\ldots,e_n \in \Lambda$ and an environment ρ for which

$$\mathbb{V}[\![xe_1e_2\cdots e_n]\!]\rho = a$$
$$\mathbb{V}[\![ye_1e_2\cdots e_n]\!]\rho = b \quad .$$

Moreover, if a, $b \in \Lambda_c$, we can choose ρ so that $\rho(V) \subset \Lambda_c$.

3.4.6 Lemma. Given x and y in Λ, suppose that, for $\delta_0 \in \Delta$, $\hat{C}(x,\delta) = \hat{C}(y,\delta)$ for all δ such that $\delta < \delta_0$ and that $\hat{C}(x,\delta_0) = \omega$ and $\hat{C}(y,\delta_0) \neq \omega$. Then, for arbitrary a in D_∞, we can choose $e_1,e_2,\ldots,e_n \in \Lambda$ and an environment ρ for which

$$\mathbb{V}[\![xe_1e_2\cdots e_n]\!]\rho = \perp$$
$$\mathbb{V}[\![ye_1e_2\cdots e_n]\!]\rho = a \quad .$$

Moreover, if $a \in \Lambda_c$, we can choose ρ so that $\rho(V) \subset \Lambda_c$.

We prove only Lemma 3.4.5. The proof for Lemma 3.4.6 is straightforward from the proof given for Lemma 3.4.5.

Proof of Lemma 3.4.5.

Case I. $\delta_0 = 0$: Since $(u,i) \neq (v,j)$, $u \neq v$ or $i \neq j$.

Case a. $u \neq v$: By the definition of $\hat{C}$, we conclude that

$$L(x,0) = \lambda t_1 t_2 \cdots t_m . uX_1 \cdots X_n$$
$$L(y,0) = \lambda t_1 t_2 \cdots t_p . vY_1 \cdots Y_q$$

Take a positive integer K such that $K > \max(m,p) + 1$. Then

$$L(x,0) t_1 t_2 \cdots t_K \xrightarrow{\text{CNV}} uX_1 X_2 \cdots X_n t_{m+1} \cdots t_K$$
$$L(y,0) t_1 t_2 \cdots t_K \xrightarrow{\text{CNV}} vY_1 Y_2 \cdots Y_q t_{p+1} \cdots t_K$$

Note that $u, v \in V = F \cup T$.

Now take an environment, ρ, so that

$$\rho(u) = \lambda s_1 s_2 \cdots s_{K-m+n} . s_{K-m+n}$$
$$\rho(v) = \lambda s_1 s_2 \cdots s_{K-p+q} . s_{K-p+q-1}$$
$$\rho(t_K) = a$$
$$\rho(t_{K-1}) = b$$

Then

$$\mathbb{V} [\![x t_1 t_2 \cdots t_K]\!] \rho = a$$
$$\mathbb{V} [\![y t_1 t_2 \cdots t_K]\!] \rho = b$$

Case b. $u = v$ and $i \neq j$: We can assume, without loss of generality, that $i < j$. As in Case a,

$$L(x,0) = \lambda t_1 t_2 \cdots t_m . uX_1 \cdots X_n \quad \text{and} \quad i = m - n$$
$$L(y,0) = \lambda t_1 t_2 \cdots t_p . uY_1 \cdots Y_q \quad \text{and} \quad j = p - q$$

Let K be a positive number such that

$$K > \max(m, p, j+n) \quad .$$

$$L(x,0) t_1 t_2 \cdots t_K \xrightarrow{\text{CNV}} uX_1 X_2 \cdots X_n t_{m+1} \cdots t_K$$
$$L(y,0) t_1 t_2 \cdots t_K \xrightarrow{\text{CNV}} uY_1 Y_2 \cdots Y_q t_{p+1} \cdots t_K$$

By substituting $\lambda s_1 s_2 \cdots s_{K-j} . s_{K-j}$ for u we have

$$L(x,0) t_1 t_2 \cdots t_K \xrightarrow{\text{CNV}} t_{K-j+i} t_{K-j+i+1} \cdots t_K$$
$$L(y,0) t_1 t_2 \cdots t_K \xrightarrow{\text{CNV}} t_K$$

since $K - j > n$.

Now we choose ρ such that

$$\rho(u) = \lambda s_1 s_2 \cdots s_{K-j} . s_{K-j}$$
$$\rho(t_K) = b$$
$$\rho(t_{K-j+i}) = \lambda s_1 s_2 \cdots s_{j-i} . a$$

and we have

$$\mathcal{V} [\![x t_1 t_2 \cdots t_K]\!] \rho = a$$
$$\mathcal{V} [\![y t_1 t_2 \cdots t_K]\!] \rho = b$$

<u>Case II</u>. $\delta_0 > 0$: Let $\delta_0 = (d_1, d_2, \ldots, d_L)$ and set $\delta^0 = 0, \delta^1 = (d_1), \ldots, \delta^\ell = (d_1, \ldots, d_\ell), \ldots, \delta^L = \delta_0$ (set $d_0 = 0$ for convenience). By the assumption of the theorem, for $0 \leq \ell \leq L-1$,

$$\hat{C}(x, \delta^\ell) = \hat{C}(y, \delta^\ell) = (z_\ell, k_\ell)$$

for some $(z_\ell, k_\ell) \in S$ and

$$\hat{C}(x,\delta_0) = (u,i)$$
$$\hat{C}(y,\delta_0) = (v,j) \quad .$$

The proof technique for Case II is the following:
Due to the fact that $\hat{C}(x,\delta) = \hat{C}(y,\delta)$ for $\delta < \delta_0$, we can choose a context under which $L(x,\delta_0)$ and $L(y,\delta_0)$ can be 'dug out' of x and y, respectively. Thereafter the problem is reduced to the difference between $L(x,\delta_0)$ and $L(y,\delta_0)$ which, applying the technique used in Case I, leads us to the conclusion of the Lemma.

Case II-1. Suppose that all z_ℓ's are distinct, i.e. for $\ell_1 \neq \ell_2$, $z_{\ell_1} \neq z_{\ell_2}$: Let, for $\ell = 0,\ldots,L-1$,

$$L(x,\delta^\ell) = \lambda t_{\delta^\ell \circ 1} t_{\delta^\ell \circ 2} \cdots t_{\delta^\ell \circ m_\ell} . z_\ell X_1 X_2 \cdots X_{n_\ell}$$

and

$$L(y,\delta^\ell) = \lambda t_{\delta^\ell \circ 1} t_{\delta^\ell \circ 2} \cdots t_{\delta^\ell \circ p_\ell} . z_\ell Y_1 Y_2 \cdots Y_{q_\ell}$$

where $m_\ell - n_\ell = p_\ell - q_\ell$ by the assumption of the lemma. Let $k_\ell = m_\ell - n_\ell$ and take a positive integer K such that

$$K > \max_{\ell=0,\ldots,L-1} (m_\ell, p_\ell, k_\ell + d_{\ell+1}) \quad .$$

If we substitute $\lambda s_1 s_2 \cdots s_{K-k_\ell} . s_{d_{\ell+1}}$ for z_ℓ

$$L(x,\delta^\ell) t_{\delta^\ell \circ 1} t_{\delta^\ell \circ 2} \cdots t_{\delta^\ell \circ K} \xrightarrow{\text{CNV}} L(x,\delta^{\ell+1})$$
$$L(y,\delta^\ell) t_{\delta^\ell \circ 1} t_{\delta^\ell \circ 2} \cdots t_{\delta^\ell \circ K} \xrightarrow{\text{CNV}} L(y,\delta^{\ell+1}) \quad .$$

Thus by induction on ℓ, if we substitute $Q = \lambda s_1 s_2 \cdots s_{K-k_\ell} \cdot s_{d_{\ell+1}}$ for z_ℓ for $\ell = 0,1,\ldots,L-1$, we have

$$\int_Q^{z_\ell} x t_1 t_2 \cdots t_K t_{\delta^1 \circ 1} t_{\delta^1 \circ 2} \cdots t_{\delta^1 \circ K} t_{\delta^2 \circ 1} \cdots t_{\delta^{L-1} \circ K} \xrightarrow{\text{CNV}} \int_Q^{z_\ell} L(x,\delta_o)$$

$$\int_Q^{z_\ell} y t_1 t_2 \cdots t_K t_{\delta^1 \circ 1} t_{\delta^1 \circ 2} \cdots t_{\delta^1 \circ K} t_{\delta^2 \circ 1} \cdots t_{\delta^{L-1} \circ K} \xrightarrow{\text{CNV}} \int_Q^{z_\ell} L(y,\delta_o)$$

Combining this result with Case I, we conclude that the lemma holds for Case II-1.

<u>Case II-2</u>. z_ℓ, $\ell = 0,1,2,\ldots,L-1$, are not necessarily distinct: We should note that the proof technique used in Case II-1 is no longer valid here since it is not possible to substitute different combinators for z_ℓ's. To get around this difficulty, we introduce a combinator $R_K = \lambda s_1 s_2 \cdots s_K . s_K s_1 s_2 \cdots s_{K-1}$. Roughly speaking, we will substitute R_K for each z_ℓ in x and y so that x and y will have distinct head variables after the substitutions.

Before we start working on x and y, we give two observations for R_K.

<u>Claim 1</u>. Let $S, T \in \Lambda$. If $S \sim T$, then, for a sufficiently large K, $\int_{R_K}^{z} S \sim \int_{R_K}^{z} T$ where z is a free variable appearing in S and T.

<u>Proof</u>. Suppose that neither S nor T has a head normal form. Then obviously neither $\int_R^z S$ nor $\int_R^z T$ has a head normal form.

Suppose S and T have a head normal form and (after several α-conversions)

$$S = \lambda r_1 r_2 \cdots r_m . w S_1 S_2 \cdots S_n$$
$$T = \lambda r_1 r_2 \cdots r_p . w T_1 T_2 \cdots T_q$$

where $m - n = p - q$.

<u>Case i</u>. $z \neq w$:

$$\int_{R_K}^{z} S = \lambda r_1 r_2 \cdots r_m . w S_1' S_2' \cdots S_n'$$

$$\int_{R_K}^{z} T = \lambda r_1 r_2 \cdots r_p . w T_1' T_2' \cdots T_q'$$

where $S_j' = \int_{R_K}^{z} S_j$ and $T_j' = \int_{R_K}^{z} T_j$. Thus $\int_{R_K}^{z} S \sim \int_{R_K}^{z} T$.

<u>Case ii</u>. $z = w$: We take K so that $K > \max(n,q)$. Then

$$\int_{R}^{z} S = \lambda r_1 r_2 \cdots r_m . (\lambda s_1 s_2 \cdots s_K . s_K s_1 s_2 \cdots s_{K-1}) S_1' S_2' \cdots S_n'$$
$$\xrightarrow{\text{CNV}} \lambda r_1 r_2 \cdots r_m s_{n+1} s_{n+2} \cdots s_K . s_K S_1' S_2' \cdots S_n' s_{n+1} \cdots s_{K-1} \quad .$$

In the same manner

$$\int_{R}^{z} T \xrightarrow{\text{CNV}} \lambda r_1 r_2 \cdots r_p s_{q+1} \cdots s_K . s_K T_1' T_2' \cdots T_q' s_{q+1} \cdots s_{K-1} \quad .$$

From $m - n = p - q$, we conclude that

$$(m + K - n) - (K - 1) = (p + K - q) - (K - 1)$$

Thus $\int_{R}^{z} S$ and $\int_{R}^{z} T$ have the same index. Also the heads in $\int_{R}^{z} S$ and $\int_{R}^{z} T$ are $K - m + n = K - p + q^{\text{th}}$ bound variable in both.

Thus $\int_{R_K}^{z} S \sim \int_{R_K}^{z} T$. □

Claim 2. If $S \nmid T$, then for a sufficiently large K, $\int_{R_K}^{z} S \nmid \int_{R_K}^{z} T$ for a free variable z in S and T.

Proof. Suppose that S has a head normal form and T does not. Then as we have seen in the proof of Claim 1, $\int_{R_K}^{z} S$ has a head normal form for a sufficiently large K, but $\int_{R_K}^{z} T$ does not have a head normal form. Thus

$$\int_{R_K}^{z} S \nmid \int_{R_K}^{z} T \quad .$$

On the other hand, suppose

$$S = \lambda r_1 r_2 \cdots r_m . w_1 S_1 S_2 \cdots S_n$$
$$T = \lambda r_1 r_2 \cdots r_p . w_2 T_1 T_2 \cdots T_q$$

where $(m-n, w_1) \neq (p-q, w_2)$. If $w_1 \neq z$ and $w_2 \neq z$, obviously $\int_{R_K}^{z} S \nmid \int_{R_K}^{z} T$ since the substitution does not alter the heads nor the indices. If $w_1 \neq w_2$ and $w_1 = z$, then we take K so that $K > \max(n, p+n-m)$. As above,

$$\int_{R_K}^{z} S \xrightarrow{CNV} \lambda r_1 r_2 \cdots r_m s_{n+1} \cdots s_K . s_K S_1' S_2' \cdots S_n' s_{n+1} \cdots s_{K-1}$$

while

$$\int_{R_K}^{z} T = \lambda r_1 r_2 \cdots r_p . w_2 T_1' T_2' \cdots T_q' \quad .$$

The head s_K in $\int_{R_K}^{z} S$ is the $K\text{-}n\text{+}m^{th}$ bound variable but since $K\text{-}n\text{+}m > p$, w_2 cannot be $K\text{-}n\text{+}m^{th}$ bound variable in $\int_{R_K}^{z} T$. Thus

$$\int_{R_K}^{z} S \nmid \int_{R_K}^{z} T \quad .$$

On the other hand, if $w_1 = w_2 = z$ and $m - n \neq p - q$, then we take K such that $K > \max(n,q)$. It is easy to see that $\int_{R_K}^{z} S$ and $\int_{R_K}^{z} T$ are different in their indices. $\square$

Now we are ready to prove our lemma for Case II-2. We determine K of R_K in the following way: Let

$$L(x,\delta^{\ell}) = \lambda t_{\delta^{\ell}\circ 1} t_{\delta^{\ell}\circ 2} \cdots t_{\delta^{\ell}\circ m_{\ell}} . u_{\ell} X_1 X_2 \cdots X_{n_{\ell}}$$

$$L(y,\delta^{\ell}) = \lambda t_{\delta^{\ell}\circ 1} t_{\delta^{\ell}\circ 2} \cdots t_{\delta^{\ell}\circ p_{\ell}} . v_{\ell} Y_1 Y_2 \cdots Y_{q_{\ell}}$$

for $\ell = 0,1,\ldots,L$. We take K such that $K > \max\limits_{\ell=0,1,\ldots,L} (n_{\ell},q_{\ell},d_{\ell}) + 1$.

<u>Step 0</u>. We want a context to select $L(x,\delta^1)$ and $L(y,\delta^1)$ out of x and y, respectively. Let

$$L(x,0) = \lambda t_1 t_2 \cdots t_{m_0} . z_0 X_1 X_2 \cdots X_{n_0}$$

$$L(y,0) = \lambda t_1 t_2 \cdots t_{p_0} . z_0 Y_1 Y_2 \cdots Y_{q_0}$$

where $m_0 - n_0 = p_0 - q_0$. Substituting R_K for z_0 in

$L(x,0)t_1t_2\cdots t_H$ and $L(y,0)t_1t_2\cdots t_H$ where $\max(m_o,p_o,d_1+m_o-n_o) < H < K+m_o-n_o$, we have

$$X_o^* = \lambda s_{H-m_o+n_o-1}s_{H-m_o+n_o+2}\cdots s_K\cdot s_K X_1'X_2'\cdots X_{n_o}' t_{m_o+1}\cdots t_H$$

$$s_{H-m_o+n_o+1}\cdots s_{K-1} \qquad \text{from } L(x,0)t_1t_2\cdots t_H$$

$$Y_o^* = \lambda s_{H-p_o+q_o+1}s_{H-p_o+q_o+2}\cdots s_K\cdot s_K Y_1'Y_2'\cdots Y_{q_o}' t_{p_o+1}\cdots t_H$$

$$s_{H-p_o+q_o+1}\cdots s_{K-1} \qquad \text{from } L(y,0)t_1t_2\cdots t_H$$

where $X_j' = \int_{R_K}^{z_o} X_j$ and $Y_j' = \int_{R_K}^{z_o} Y_j$.

Now substituting $\lambda r_1r_2\cdots r_{K-1}\cdot r_{d_1}$ for s_K in both $X_o^* s_{H-m_o+n_o+1}s_{H-m_o+n_o+1}\cdots s_K$ and $Y_o^* s_{H-p_o+q_o+1}s_{H-p_o+q_o+2}\cdots s_K$, we have

$$X^* s_{H-m_o+n_o+1}\cdots s_K \xrightarrow{\text{CNV}} \int_{R_K}^{z_o} L(x,\delta^1)$$

$$Y^* s_{H-p_o+q_o+1}\cdots s_K \xrightarrow{\text{CNV}} \int_{R_K}^{z_o} L(y,\delta^1) \quad .$$

<u>Step ℓ</u>. Suppose that, under some context, we have selected $\int_{R_KR_K\cdots R_K}^{z_oz_1\cdots z_{\ell-1}} L(x,\delta^\ell)$ and $\int_{R_KR_K\cdots R_K}^{z_oz_1\cdots z_{\ell-1}} L(y,\delta^\ell)$ out of x and y respectively. Let

$$L(x,\delta^\ell) = \lambda t_{\delta^\ell\circ 1} t_{\delta^\ell\circ 2}\cdots t_{\delta^\ell\circ m_\ell}\cdot z_\ell X_1X_2\cdots X_{n_\ell}$$

$$L(y,\delta^\ell) = \lambda t_{\delta^\ell\circ 1} t_{\delta^\ell\circ 2}\cdots t_{\delta^\ell\circ p_\ell}\cdot z_\ell Y_1Y_2\cdots Y_{q_\ell} \quad .$$

We take a positive integer, H, such that

$$\max(m_\ell, p_\ell, d_{\ell+1}+m_\ell-n_\ell) < H < K+m_\ell-n_\ell$$

and, substituting R_K for z_ℓ in

$$L(x,\delta^\ell)t_{\delta^\ell\circ 1}t_{\delta^\ell\circ 2}\cdots t_{\delta^\ell\circ H} \quad\text{and}\quad L(y,\delta^\ell)t_{\delta^\ell\circ 1}t_{\delta^\ell\circ 2}\cdots t_{\delta^\ell\circ H}$$

(if $z_\ell \neq z_k$ for any $k = 0,1,\ldots,\ell-1$, z_ℓ is already substituted by R_K), we have

$$X_\ell^* = \lambda s_{H-m_\ell+n_\ell+1}\cdots s_K \cdot s_K X_1' X_2' \cdots X_{n_\ell}' t_{m_\ell+1}\cdots t_H s_{H-m_\ell+n_\ell+1}\cdots s_{K-1}$$

$$Y_\ell^* = \lambda s_{H-p_\ell+q_\ell+1}\cdots s_K \cdot s_K Y_1' Y_2' \cdots Y_{q_\ell}' t_{p_\ell+1}\cdots t_H s_{H-p_\ell+q_\ell+1}\cdots s_{K-1}$$

where

$$X_j' = \int_{R_K R_K \cdots R_K}^{z_1 z_2 \cdots z_\ell} X_j$$

$$Y_j' = \int_{R_1 R_2 \cdots R_K}^{z_1 z_2 \cdots z_\ell} Y_j$$

Now, following exactly the same procedure as in Step 0, we can select $\int_{R_K R_K \cdots R_K}^{z_0 z_1 \cdots z_\ell} L(x,\delta^{\ell+1})$ and $\int_{R_K R_K \cdots R_K}^{z_0 z_1 \cdots z_\ell} L(y,\delta^{\ell+1})$ out of X_ℓ^* and Y_ℓ^* respectively.

By the discussion above we conclude that under some appropriate context, we can select

$$\int_{R_K,R_K,\ldots,R_K}^{z_0,z_1,\ldots,z_{L-1}} L(x,\delta_0) \quad\text{and}\quad \int_{R_K,R_K,\ldots,R_K}^{z_0,z_1,\ldots,z_{L-1}} L(y,\delta_0)$$

out of X and Y respectively.

Applying Claim 2 repeatedly, we conclude that

$$\int_{R_K, R_K, \ldots, R_K}^{z_0, z_1, \ldots, z_{L-1}} L(x,\delta_0) \nmid \int_{R_K, R_K, \ldots, R_K}^{z_0, z_1, \ldots, z_{L-1}} L(y,\delta_0)$$

where neither of them is without a head normal form. By the result of Case I, we have proved the lemma for Case II-2. □

3.4.7 <u>Theorem</u>. Given x, y in Λ:

1. If there exists $\delta \in \Delta$ such that, for different u, v in V,

$$C(x,\delta) = u \quad \text{and} \quad C(y,\delta) = v$$

then, for arbitrarily given a, b in D_∞, we can choose $e_1, e_2, \ldots, e_n$ in Λ and an environment ρ for which

$$\mathbb{V}[\![xe_1e_2\cdots e_n]\!]\rho = a$$
$$\mathbb{V}[\![ye_1e_2\cdots e_n]\!]\rho = b \quad .$$

Moreover, if $a, b \in \Lambda_c$, we can choose ρ such that $\rho(V) \subset \Lambda_c$.

2. If there exists $\delta \in \Delta$ such that, for all δ_0 satisfying $|\delta_0| < |\delta|$, $C(x,\delta_0) = C(y,\delta_0)$

and that $\quad C(x,\delta) = u \in V, \quad C(y,\delta) = \omega$

then, for arbitrarily given a in D_∞, we can choose $e_1, e_2, \ldots, e_n$ in Λ and an environment ρ for which

$$\mathbb{V}[\![xe_1e_2\cdots e_n]\!]\rho = a$$
$$\mathbb{V}[\![ye_1e_2\cdots e_n]\!]\rho = \perp \quad .$$

Moreover, if $a \in \Lambda_c$, we can choose ρ so that $\rho(\Lambda) \subseteq \Lambda_c$.

<u>Proof</u>. Straightforwardly deduced from Lemma 3.4.5 and Lemma 3.4.6, using a technique similar to the proof of Corollary 3.4.9.

3.4.8 <u>Definition</u>. Let

$$\mathbb{C} = \{c \mid c \in \Delta \to V \cup \{\omega\}\}$$

and

$$\hat{\mathbb{C}} = \{c \mid c \in \Delta \to S\} \quad .$$

We define relation $\leq$ over $\mathbb{C}$ and $\hat{\mathbb{C}}$ as follows:

For $c_1, c_2 \in \mathbb{C}$ $(\hat{\mathbb{C}})$, $c_1 \leq c_2$ if and only if, for all $\delta \in \Delta$, $c_1(\delta) = \omega$ or $c_1(\delta) = c_2(\delta)$.

3.4.9 <u>Corollary</u>. For x, y in Λ, $C(x) \leq C(y)$ if and only if $\hat{C}(x) \leq \hat{C}(y)$.

<u>Proof</u>. If $\hat{C}(x) \leq \hat{C}(y)$, $C(x) \leq C(y)$ is immediate from the definition of C.

On the other hand, assume that $\hat{C}(x) \leq \hat{C}(y)$ does not hold. This means that there is $\delta \in \Delta$ such that

either <u>Case 1</u>. $\hat{C}(x,\delta) = (u,i)$ and $\hat{C}(y,\delta) = (v,j)$ where $(u,i) \neq (v,j)$

or <u>Case 2</u>. $\hat{C}(x,\delta) \neq \omega$ and $\hat{C}(y,\delta) = \omega$.

$C(x) \not\leq C(y)$ is immediate from Case 2 by the definition of C.

For Case 1, let δ be such that, for any δ' satisfying $\delta' < \delta$,

$$\hat{C}(x,\delta') = \hat{C}(y,\delta')$$

and

$$\hat{C}(x,\delta) = (u,i)$$
$$\hat{C}(y,\delta) = (v,j) \quad .$$

If $u \neq v$, $C(x) \not\leq C(y)$ is immediate.

Suppose $u = v$ and $i \neq j$. Let

$$L(x,\delta) = \lambda t_{\delta\circ 1} t_{\delta\circ 2} \cdots t_{\delta\circ m} . u X_1 X_2 \cdots X_n$$
$$L(y,\delta) = \lambda t_{\delta\circ 1} t_{\delta\circ 2} \cdots t_{\delta\circ p} . u Y_1 Y_2 \cdots Y_q$$

where $m - n = i$ and $p - q = j$. If we take K larger than n and q,

$$C(x,\delta\circ K) = L(x,\delta\circ K) = t_{\delta\circ(K-n+m)}$$
$$C(y,\delta\circ K) = L(x,\delta\circ K) = t_{\delta\circ(K-q+p)} \ .$$

Since $i \neq j$, $K - n + m \neq K - q + p$. Thus $C(x,\delta) \not\leq C(y,\delta)$. □

3.4.10 Corollary. For x, y in Λ, $C(x) = C(y)$ if and only if $\hat{C}(x) = \hat{C}(y)$. □

3.4.11 Corollary. For $x, y \in \Lambda$, if $x \underset{D_\infty}{\subseteq} y$, then $C(x) \leq C(y)$.

Proof. Let us negate that $C(x) \leq C(y)$. Then there must exist $\delta \in \Delta$ such that, for some $u, v \in V$,

either $\quad \begin{aligned} C(x,\delta) &= u \\ C(y,\delta) &= v \end{aligned} \quad$ where $u \neq v$

or $\quad \begin{aligned} C(x,\delta) &= u \\ C(y,\delta) &= \omega \end{aligned} \quad .$

In either case, there must be at least one $\delta \in \Delta$ for which the condition of part 1 or part 2 of Theorem 3.4.7 holds. Thus by the conclusion of the theorem, there exist $e_1, e_2, \ldots, e_n \in \Lambda$ and an environment ρ such that,

$$\text{either} \quad \begin{array}{l} \mathbb{V}[\![xe_1e_2\cdots e_n]\!]\rho = K \\ \text{and} \\ \mathbb{V}[\![ye_1e_2\cdots e_n]\!]\rho = H \end{array}$$

$$\text{or} \quad \begin{array}{l} \mathbb{V}[\![xe_1e_2\cdots e_n]\!]\rho = K \\ \text{and} \\ \mathbb{V}[\![ye_1e_2\cdots e_n]\!]\rho = \perp \end{array} ,$$

which contradicts $x \underset{D_\infty}{\subseteq} y$ by Proposition 2.3.5 and Corollary 2.3.18.

□

3.4.12 <u>Corollary</u>. For $x, y \in \Lambda$, if $x \underset{D_\infty}{=} y$, then $C(x) = C(y)$.

<u>Proof</u>. Similar to the proof of Corollary 3.4.11. □

In fact, the converses of Corollary 3.4.11 and Corollary 3.4.12 are also true and will be proved in Chapter 4.

We should note here that we can further formalize C-functions. Since each variable z in V is in F or T_Λ, we encode z as follows:

If $z = f_i$ in F, $En(z) = i \in \mathbb{N}$.

If $z = t_\delta$ in T_Δ, $En(z) = \delta \in \Delta$.

Now the new version of C, $\bar{C}: \Lambda \to (\Delta \to \Delta \cup \mathbb{N} \cup \{\omega\})$ is defined by:

$$\text{For } x \in \Lambda, \quad \begin{array}{ll} \bar{C}(x,\delta) = En(z) & \text{if } C(x,\delta) = z \in V \\ \bar{C}(x,\delta) = \omega & \text{if } C(x,\delta) = \omega \end{array} .$$

Thus we can discard the notion of variables. We do not take this convention since this does not provide us with any substantial improvement other than formalism.

§5. C-function as Infinite Normal Form -- Extension of Böhm's Theorem

Theorem 3.4.7 can be regarded as an extension of Böhm's Theorem [4] which is stated as follows:

3.5.1 Böhm's Theorem. Let x, y in Λ. If x and y have different normal forms, then, for any two variables $u, v \in V$, we can choose λ-expressions $e_1, e_2, \ldots, e_n \in \Lambda$, variables $z_1, z_2, \ldots, z_m \in V$ and closed λ-expressions $h_1, h_2, \ldots, h_m \in \Lambda_c$ such that

$$\left(\int_{h_1,h_2,\ldots,h_m}^{z_1,z_2,\ldots,z_m} x\right) e_1 e_2 \cdots e_n \xrightarrow{\text{CNV}} u$$

and

$$\left(\int_{h_1,h_2,\ldots,h_m}^{z_1,z_2,\ldots,z_m} y\right) e_1 e_2 \cdots e_n \xrightarrow{\text{CNV}} v \quad . \quad \square$$

If we translate Theorem 3.4.7 into one stated in pure λ-calculus language:

3.5.2 Theorem. Let x, y in Λ. If $C(x) \neq C(y)$, then, for any $u, v \in V$, we can choose λ-expressions, $e_1, e_2, \ldots, e_n \in \Lambda$, variables $z_1, z_2, \ldots, z_m \in V$ and closed λ-expressions $h_1, h_2, \ldots, h_m \in \Lambda_c$ so that one of the following 1), 2) and 3) holds: Let

$$x^* = \left(\int_{h_1,h_2,\ldots,h_m}^{z_1,z_2,\ldots,z_m} x\right) e_1 e_2 \cdots e_n$$

and

$$y^* = \left(\int_{h_1,h_2,\ldots,h_m}^{z_1,z_2,\ldots,z_m} y\right) e_1 e_2 \cdots e_n \quad .$$

1) $x^* \xrightarrow{\text{CNV}} u$ and $y^* \xrightarrow{\text{CNV}} v$.

2) $x^* \xrightarrow{\text{CNV}} u$ and y^* has no head normal form.

3) x^* has no head normal form and $y^* \xrightarrow{\text{CNV}} u$. □

The main point of the extension of Böhm's Theorem is that we are no longer concerned with conventional normal forms. Theorem 3.5.2 is a statement about general λ-expressions no matter whether or not they are normal. In this respect, we might as well call C-functions <infinite normal form> or generalized normal form. Refer to [21] for an alternative extension of Böhm's Theorem.

CHAPTER 4

CHARACTERIZATION OF THE D_∞-VALUES OF THE λ-EXPRESSIONS

The main result in this chapter is to characterize, using the C-function, the partial ordering among the λ-expressions that is induced by D_∞. Namely, it is shown that, given two λ-expressions x, y, the relation $x \subseteq y$ in D_∞ is equivalent to $C(x) \leq C(y)$ in the algebraic domain which includes the range of C.

§1. Structural Approximation

In Chapter 2, we defined the notion of approximant of λ-expressions. Roughly speaking, the approximant of $x \in \Lambda$ is obtained from x by replacing the β-redexes in x by $\perp$.

In this section, we define another notion of approximation which is more closely related to the C-function. For this purpose, we need a class of subsets of Δ, called Δ-trees.

4.1.1 Definition (Δ-trees). A Δ-tree, T, is an infinite subset of Δ such that

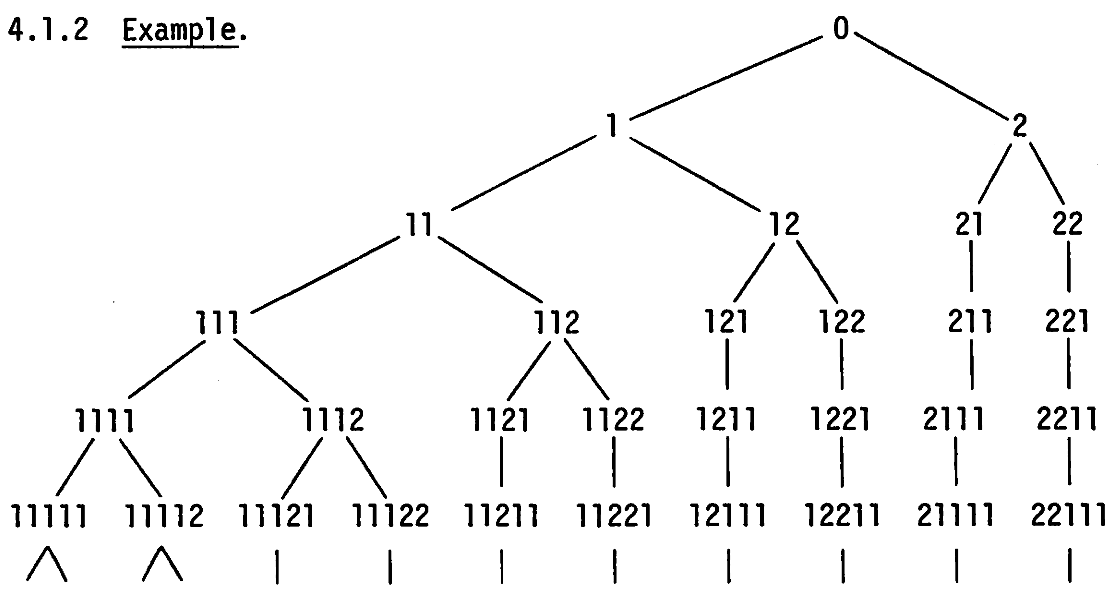
1) $0 \in T$.

2) If $\delta \in T$, then $Pr(\delta) \in T$.

3) For all $\delta \in T$, there exists a positive integer N such that $\delta\circ 1, \delta\circ 2, \ldots, \delta\circ N \in T$ and $\delta\circ K \notin T$ for all $K > N$.

For a Δ-tree, T, we call N in 3) above $\gamma_T(\delta)$, i.e. $\gamma_T(\delta) = \#\{\delta' \mid \delta' \in T,\ Pr(\delta') = \delta\}$.

4.1.2 Example.



Let T be $\{\alpha \in \{1,2\}^* \mid \alpha = \beta,\ \beta 2\gamma \text{ or } \beta 22\gamma \text{ for } \beta, \gamma \in \{1\}^*\} \cup \{0\}$ $(= 0 \cup 1^* \cup 1^*21^* \cup 1^*221^*)$.

Obviously, T is a Δ-tree. Here, for example, $\gamma_T(11) = 2$, $\gamma_T(121) = 1$.

In Chapter 3, we formulated the expansion of λ-expressions. Δ-trees give in a way the opposite operation. Suppose, given the C-function of a certain λ-expression, we want to synthesize the original λ-expression from the C-function. Since the C-function contains arbitrary numbers of η-abstractions, we should restrict our attention to finite and meaningful parts of the C-function, which are represented by a Δ-tree.

4.1.3 <u>Definition</u>. Given a Δ-tree, T, $n \in \mathbb{N}$ and x in Λ, we define <u>$T^n(x)$</u> to be $T^0_{x,n}$ where $T^\delta_{x,n}$ is defined for $\delta \in T$ by:

1) If $|\delta| < n$, then
   a) if $\hat{C}(x,\delta) = \omega$, then $T^\delta_{x,n} = \Omega$.
   b) if $\hat{C}(x,\delta) = (z,k)$, then

$$T^\delta_{x,n} = \lambda t_{\delta\circ 1} t_{\delta\circ 2} \cdots t_{\delta\circ[\gamma_T(\delta)+k]} . z T^{\delta\circ 1}_{x,n} T^{\delta\circ 2}_{x,n} \cdots T^{\delta\circ\gamma_T(\delta)}_{x,n} .$$

2) If $|\delta| = n$, then $T^\delta_{x,n} = L(x,\delta)$,

where [ ] is the Gauss notation.

4.1.4 <u>Definition</u>. A Δ-tree, T, is said to be <u>admissible</u> to x in Λ if and only if, for all $n \in \mathbb{N}$,

$$x \xrightarrow[\substack{\alpha \\ \beta \\ \eta\text{-ab} \\ \Omega}]{} T^n(x) .$$

Intuitively, a Δ-tree, T, is admissible to x in Λ if it is wide enough to cover the whole significant portion of x, i.e.

the significant parts which were derived by β-reductions rather than by η-abstractions.

4.1.5 <u>Definition</u>. Let x be in Λ. We define <u>$N(x)$</u> to be a subset of Δ such that

1) $0 \in N(x)$

2) Let $\delta \in N(x)$. Then $\delta \circ i \in N(x)$ if and only if

$$L(x,\delta) \xrightarrow{\beta} \lambda s_1 s_2 \cdots s_m . z X_1 X_2 \cdots X_n$$

for $i \leqq n$.

We should note that $N(x)$ may be infinite or finite. For example, if x is normal, or x is not head normal, $\#(N(x))$ is finite. We give $L(x,\delta)$ a special name <u>$LT(x,\delta)$</u> if $\delta \in N(x)$.

4.1.6 <u>Corollary</u>. Let x be in Λ and T be a Δ-tree. Then T is admissible to x if and only if $N(x) \subseteq T$. So, x has at least one admissible Δ-tree. □

4.1.7 <u>Corollary</u>. Let T_1, T_2 be Δ-trees. If $T_1 \subseteq T_2$ and T_1 is admissible to x, so is T_2. □

4.1.8 <u>Corollary</u>. If T is a Δ-tree admissible to x, then

$$T^n(x) \underset{D_\infty}{=} x \text{ for any } n. \quad \square$$

4.1.9 <u>Corollary</u>. If T is an admissible Δ-tree to $x \in \Lambda$, then $LT(x,\delta) \lhd T^n(x)$ for all $LT(x,\delta)$ such that $|\delta| = n$. □

4.1.10 <u>Example</u>. Let us look at T in Example 4.1.2. It is easy to see that T is admissible to R in Example 3.3.2.

$$T^3(R) = \lambda t_1 t_2 . t_1(\lambda t_{11} . t_2(\lambda t_{111} . t_{11}(Rt_{111})(Rt_{111}\Omega))(t_{11}\Omega\Omega))(t_2\Omega\Omega)$$
$$\{\delta \mid \delta \in N(R),\ |\delta| = 3\} = \{111,112,121,122\} \quad ,$$

so

$$LT(R,111) = Rt_{111}$$
$$LT(R,112) = Rt_{111}\Omega$$
$$LT(R,121) = \Omega$$
$$LT(R,122) = \Omega \quad .$$

Let us see how $T^n(x)$ is generated from x. First apply to x all the β-reductions that were necessary to generate all $LT(x,\delta)$'s for $|\delta| \leqq n$ $(x \xrightarrow{\beta} Z_1)$. So, for each $LT(x,\delta)$ such that $|\delta| = n$, $LT(x,\delta) \lhd Z_1$. Now, apply to Z_1 α-conversions so that each bound variable, which occurs outside of all $LT(x,\delta)$ with $|\delta| = n$, will be renamed and belong to T_Δ $(Z_1 \xrightarrow{\alpha} Z_2)$. Now apply Ω-conversions and replace each subexpression, which is not head normal, by Ω $(Z_2 \xrightarrow{\Omega} Z_3)$. Finally apply η-abstractions for each node in $\{\delta \mid \delta \in T - N(x),\ |\delta| \leqq n\}$ $(Z_3 \xrightarrow{\eta\text{-ab}} T^n(x))$. Note that no η-abstraction is made inside of $LT(x,\delta)$ with $|\delta| = n$. We state the process above as a corollary:

4.1.11 <u>Corollary</u>. Let x be in Λ and T be a Δ-tree admissible to x. For $n \in \mathbb{N}$, there exist $Z_1, Z_2, Z_3 \in \Lambda$ such that

$$x \xrightarrow{\beta} Z_1 \xrightarrow{\alpha} Z_2 \xrightarrow{\Omega} Z_3 \xrightarrow{\eta\text{-ab}} T^n(x)$$

where Z_2 matches Z_3 except at occurrences of Ω in Z_3 and if any of the η-abstractions applied in $Z_3 \xrightarrow{\eta\text{-ab}} T^n(x)$ is of the form $A \to \lambda s.As$, A cannot be a proper subexpression of one of $LT(x,\delta)$'s, $|\delta| = n$. Moreover, if Z_2 has any β-redex, it must be contained in either $LT(x,\delta)$ with $|\delta| = n$, or in one of the subexpressions which were converted to Ω in $Z_2 \xrightarrow{\Omega} Z_3$. □

4.1.12 Definition (Structural Approximation). Given x in Λ, $n \in \mathbb{N}$ and a Δ-tree, T, which is admissible to x, $A_p^n(x,T)$, structural approximation of x, of order n, with respect to T, is defined by: $A_p^n(x,T) = A_{x,n}^{0}$ where $A_{x,n}^{\delta}$ is recursively defined for $\delta \in T$ by:

1) If $|\delta| < n$, then

   a) if $\hat{C}(x,\delta) = \omega$ then $A_{x,n}^{\delta} = \Omega$

   b) if $\hat{C}(x,\delta) = (z,k)$, then

$$A_{x,n}^{\delta} = \lambda t_{\delta\circ 1} t_{\delta\circ 2} \cdots t_{\delta\circ(k+\gamma_T(\delta))} . zA_{x,n}^{\delta\circ 1} A_{x,n}^{\delta\circ 2} \cdots A_{x,n}^{\delta\circ\gamma_T(\delta)} .$$

2) If $|\delta| = n$, then $A^{\delta} = \Omega$.

4.1.13 Corollary. For $x \in \Lambda$, $n \in \mathbb{N}$ and Δ-tree, T, admissible to x,

1) $A_p^n(x,T)$ is obtained from $T^n(x)$ by replacing, by Ω, each $L(x,\delta)$ in $T^n(x)$ such that $|\delta| = n$.

2) $A_p^n(x,T) \underset{D_\infty}{\subseteq} x$

3) If $m \leq n$, then $A_p^m(x,T) \underset{D_\infty}{\subseteq} A_p^n(x,T)$. □

4.1.14 Example. For T in Example 4.1.2 and R in Example 3.3.2

$$A_p^3(x,T) = \lambda t_1 t_2 . t_1(\lambda t_{11} . t_2(\lambda t_{111} . t_{11}\Omega\Omega)(t_{11}\Omega\Omega))(t_2\Omega\Omega) \quad .$$

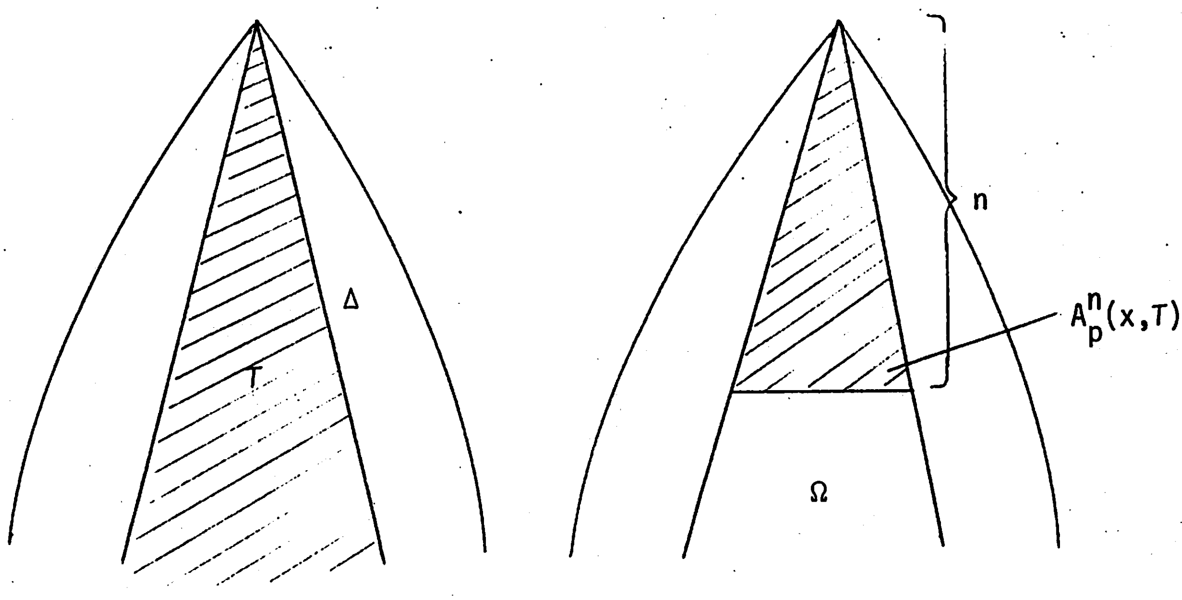
The following corollary characterizes the relation between the C-function and the structural approximation.

4.1.15 Corollary. Let x be in Λ and T be a Δ-tree admissible to x. Then, for each $n \in \mathbb{N}$,

$$C(A_p^n(x,T),\delta) = \begin{cases} C(x,\delta) & \text{if } |\delta| < n \text{ or } \delta \notin T \\ \omega & \text{if } |\delta| \geqq n \text{ and } \delta \in T \end{cases} \quad .$$

The assertion remains valid when we replace C by $\hat{C}$.

Proof. Since $x \approx T^n(x)$, $L(x,\delta) \approx L(T^n(x),\delta)$ for all δ and $C(x) = C(T^n(x))$. The conclusion of the corollary is immediate since, by Corollary 4.1.13-1, $A_p^n(x,T)$ matches $T^n(x)$ except at occurrences of Ω in $A_p^n(x,T)$. $\square$



§2. Convergence Lemma

Our objective here is to prove Lemma 4.2.2 which is fundamental throughout the rest of this thesis. We use the proof technique of typed λ-calculus in Chapter 2.

4.2.1 Lemma. Let $x \in \Lambda$ and T be a Δ-tree admissible to x. For any x^t in $\underline{T}(x)$, there is a sufficiently large n such that, $x^t \underset{D_\infty}{\subseteq} A_p^n(x,T)$.

Since the proof for Lemma 4.2.1 is very long with a high complexity, we first give the outline of the proof in order to ease the unreadability of the full proof.

1: There exists a typed β-reduction sequence $x^t \to x_p^t$ so that x_p^t has no typed β-redex (by Lemma 2.3.13).

2: There exists a (usual) β-reduction sequence $x \to x_p$ so that $\underline{W}(x_p^t)$ is a reduced approximant of x_p (by Lemma 2.3.14).

3: For $k \in \mathbb{N}$, there are Q_1^k, Q_2^k and $W^k \in \Lambda$, so that $x_p \xrightarrow{\beta} Q_1^k \xrightarrow{\alpha} Q_2^k \xrightarrow{\eta\text{-ab}} W^k$ and W^k matches $T^k(x)$ except at occurrences of Ω and $LT(x,\delta)$ with $|\delta| = k$ in $T^n(x)$ (by Corollary 4.1.11 and the Church-Rosser Theorem). The parts in W^k to match Ω in $T^k(x)$ are non-head normal.

4: Correspondingly, there is a typed conversion sequence $x_p^t \xrightarrow{t\beta} y_q^t \xrightarrow{t\alpha} y^t \xrightarrow{*} Y^t$ where $y_q^t \in \underline{T}(Q_1^k)$, $y^t \in \underline{T}(Q_2^k)$ and $Y^t \in \underline{T}(W^k)$ ($*$ is 'modified' $t\eta$-ab) and $x_p^t \underset{D_\infty}{\subseteq} Y^t$.

5: By the definition of $T^k(x)$, $T^k(x)$ contains each variable z^δ such that $C(x,\delta) = z^\delta$ for each $\delta \in T$ with $|\delta| = k-1$.

6: By 3, W^k contains each z^δ in 5 and, by 4, Y^t contains a typed variable $t[z^\delta] \in \underline{T}(z^\delta)$ for each z^δ in 5.

7: We consider the process in which z^δ in W^k is derived:

[1] z^δ already occurs in x_p.

[2] z^δ is derived in $x_p \xrightarrow{\beta} Q_1^k$.

[3] z^δ is derived in $Q_2^k \xrightarrow{\eta\text{-ab}} W^k$.

8: If we take k large enough, we can reassign type 0 to each $t[z^\delta]$ in Y^t so that $|\delta| = k-1$ and still have $x_p^t \underset{D_\infty}{\subseteq} Y^t$ valid, for,

a. [1] cannot occur for z^δ if k is large enough.

b. In [2], z^δ is from a β-redex in x_p, but x_p^t has no typed β-redex. (The β-redexes have degenerated to Ω in x_p^t.)

c. In [3], note that typed η-abstraction reduces the type of the variable by 1 (i.e. $A^{(n)} \to \lambda s.(A^{(n)}s^{(n-1)})^{(n-1)}$).

So, if k is large enough, $t[z^\delta]$ is of type 0.

9: Since we have reassigned 0 to each $t[z^\delta]$ in Y^t with $|\delta| = n-1$, we can replace the subexpressions applied to $t[z^\delta]$ by $\Omega^{(0)}$ and still have $x_p^t \underset{D_\infty}{\subseteq} Y^t$ (by Theorem 2.2.5). Furthermore, we replace, by $\Omega^{(0)}$, each subexpression in Y^t that corresponds to a non-head normal subexpression in W^k (pointed out in 3).

10: Since W^k matches $T^k(x)$ except at $LT(x,\delta)$ ($|\delta| = n$) and occurrences of Ω in $T^k(x)$, and $A_p^k(x,T)$ is derived from $T^k(x)$ by replacing $L(x,\delta)$ ($|\delta| = k$) by Ω, we conclude that Y^t obtained in 9 is in $\underline{T}(A_p^k(x,T))$. So, $x^t \underset{D_\infty}{=} x_p^t \underset{D_\infty}{\subseteq} Y^t \underset{D_\infty}{\subseteq} A_p^k(x,T)$.

The following is the reason why the proof is so difficult: The transformation $x \to \mathcal{T}^n(x)$ involves both β-reductions and η-abstractions. The structure, $\mathcal{T}$, is arbitrarily given regardless of the structure of x. In addition, the transformation $\mathcal{T}^n(x) \to A_p^n(x,\mathcal{T})$ is rather an artificial deformation and the parts in $\mathcal{T}^n(x)$ that are replaced by Ω include variables added by η-abstractions as well as subexpressions generated by β-reductions.

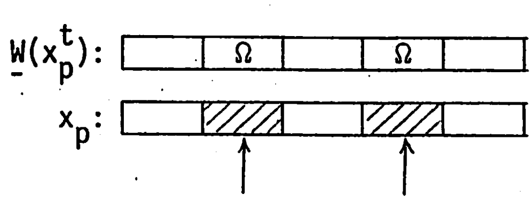
Proof. By Lemma 2.3.13, there exists the following typed β-reduction sequence:

$$x^t \to x_1^t \to x_2^t \to \cdots \to x_p^t$$

where $x_i^t \in \Lambda^t$ and x_{i+1}^t derives from x_i^t by one application of typed β-reduction and x_p^t has no typed β-redex. Correspondingly, by Lemma 2.3.14, there is a β-reduction sequence:

$$x \to x_1 \to x_2 \to \cdots \to x_p$$

where x_{i+1} derives from x_i by one application of β-reduction and x_i matches $\underline{W}(x_i^t)$ except at occurrences of Ω in $\underline{W}(x_i^t)$. Since x_p^t has no β-redex, $\underline{W}(x_p^t)$ is a reduced approximant of x, and so, every β-redex in x_p is contained in a part of x_p which has no corresponding part in $\underline{W}(x_p^t)$ except Ω.



a β-redex can exist only in these areas.

Given any k in $\mathbb{N}$, by Corollary 4.1.11 there exist $Z_1^k, Z_2^k, Z_3^k \in \Lambda$ such that

$$x \xrightarrow{\beta} Z_1^k \xrightarrow{\alpha} Z_2^k \xrightarrow{\Omega} Z_3^k \xrightarrow{\eta\text{-ab}} T^k(x)$$

where Z_2^k matches Z_3^k except at occurrences of Ω in Z_3^k and every β-redex in Z_2^k is contained in a part which corresponds to Ω in Z_3 or in $LT(x,\delta)$ for $|\delta| = k$. Also, if $A \to \lambda s.As$ is made in $Z_3 \to T^k(x)$, A is not a proper subexpression of $LT(x,\delta)$. By the Church-Rosser Theorem, there exist $Q_k^1, Q_k^2 \in \Lambda$ such that

$$\begin{tikzcd} x \arrow[r, "\beta"] \arrow[d, "\beta"'] & x_p \arrow[r, "\beta"] & Q_1^k \arrow[d, "\alpha"] \\ Z_1^k \arrow[r, "\alpha"'] & Z_2^k \arrow[r, "\beta"'] & Q_2^k \end{tikzcd} .$$

So

$$\begin{tikzcd} x \arrow[r, "\beta"] \arrow[d, "\beta"'] & x_p \arrow[r, "\beta"] & Q_1^k \arrow[d, "\alpha"] & \\ Z_1^k \arrow[r, "\alpha"'] & Z_2^k \arrow[r, "\beta"'] \arrow[d, "\Omega"'] & Q_2^k & \\ & Z_3^k \arrow[r, "\eta\text{-ab}"] & T^k(x) & \end{tikzcd} .$$

Since every β-redex in Z_2^k is either in a part which corresponds to no part except Ω in Z_3 or in $LT(x,\delta)$ for $|\delta| = k$, it follows that Z_3^k matches Q_2^k except at occurrences of Ω and $LT(x,\delta)$ in Z_3^k.

On the other hand, since all the η-abstractions in

$Z_3^k \xrightarrow{\eta\text{-ab}} T^k(x)$ are made externally to $LT(x,\delta)$ with $|\delta| = k$, we conclude that there are η-abstractions which, applied to Q_2^k, yield $W^k \in \Lambda$ which matches $T^k(x)$ except at occurrences of Ω and $LT(x,\delta)$ in $T^k(x)$ such that $|\delta| = n$. Thus we have:

$$x_p \xrightarrow{\beta} Q_1^k \xrightarrow{\alpha} Q_2^k \xrightarrow{\eta\text{-ab}} W^k \quad .$$

It follows that the structure of W^k is described as follows: $W^k = W_0^k$ where W_δ^k $(\delta \in T)$ is of the form:

1) if $|\delta| = k$, then $W_\delta^k = w(\delta)$ for some $w(\delta) \in \Lambda$ such that $w(\delta) \approx L(x,\delta)$

2) if $|\delta| < k$, then

a) if $\hat{C}(x,\delta) = \omega$, then W_δ^k is a certain expression which has no head normal form

b) if $\hat{C}(x,\delta) = (z^\delta, r^\delta)$, then

$$W_\delta^k = \lambda t_{\delta\circ 1} t_{\delta\circ 2} \cdots t_{\delta\circ(\gamma_T(\delta)+r^\delta)} . z^\delta W_{\delta\circ 1}^k W_{\delta\circ 2}^k \cdots W_{\delta\circ\gamma_T(\delta)}^k \quad .$$

We examine each z^δ which appears in W^k. z^δ in W^k must satisfy one of the following conditions:

[1] This occurrence of z^δ is homologous to one in x_p (i.e. it occurs already in x_p).

[2] It was derived in the process of $x_p \xrightarrow{\beta} Q_1^k$.

[3] It was derived in the process of $Q_3^k \xrightarrow{\eta\text{-ab}} W^k$.

(In any case (1), (2) and (3) above, z^δ may have been renamed in $Q_2^k \xrightarrow{\alpha} Q_3^k$.)

Let $j = 1, 2$ or 3. We define $\eta_j^k(i)$ to be the subset of T determined by:

$\eta_j^k(i) = \{\delta \mid \delta \in T,\ |\delta| = i,\ C(x,\delta) \neq \omega$ and z^δ in W^k is derived as in [j]$\}$.

Since x_p is a finite expression, there is m_1 in $\mathbb{N}$ for which [1] above cannot occur for z^δ if $|\delta| \geqq m_1$. Let m_2 be the maximum type among the types that were assigned to the components of x^t. We set $n = m_1 + m_2$. (To simplify the description of the proof, we set $n = m_1 + 1$ if $m_2 = 0$.) We set $k = n$ and develop $x \xrightarrow{\beta} x_p \xrightarrow{\beta} Q_1^n$ into

$$x = x_o \to x_1 \to x_2 \to \cdots \to x_p \to x_{p+1} \to \cdots \to x_q = Q_1^n$$

where x_{i+1} is the result of an application of β-reduction to x_i. Correspondingly, we define a sequence typed λ-expressions:

$$y_o^t \to y_1^t \to y_2^t \to \cdots \to y_p^t \to y_{p+1}^t \to \cdots \to y_q^t$$

in the following way:

<u>Step 1</u>: Set $y_o^t = x^t \in \underline{T}(x_o)$.

<u>Step i</u>: Suppose y_{i-1}^t is in $\underline{T}(x_{i-1})$ and that the redex which is reduced in $x_{i-1} \to x_i$ is $(\lambda s.M)N \to \int_N^s M$ and that the corresponding occurrence of $(\lambda s.M)N$ in y_{i-1}^t is of the form $((\lambda s.M^t)^{(h_1)} N^t)^{(h_2)}$ where $M^t \in \underline{T}(M)$ and $N^t \in \underline{T}(N)$. We define y_i^t as follows:

<u>Case 1</u>. If $h_1 > 0$, replace $((\lambda s.M^t)^{(h_1)} N^t)^{(h_2)}$ in y_{i-1}^t by

$$[\int_{[N^t]_{h_1-1}}^s M^t]_{\min(h_1-1,h_2)} .$$

<u>Case 2</u>. If $h_1 = 0$, replace $((\lambda s.M^t)^{(h_1)} N^t)^{(h_2)}$ by

$$[\int_{[N^t]_0}^{s} M^t]_0 \quad .$$

Obviously y_i^t defined above is in $\underline{I}(x_i)$.

Moreover,

$$y_{i-1}^t \underset{D_\infty}{\subseteq} y_i^t \quad ,$$

for, if Case 1 occurs, then it is exactly typed β-reduction. So by Theorem 2.3.12,

$$y_{i-1}^t \underset{D_\infty}{=} y_i^t \quad .$$

On the other hand, in Case 2, we could have replaced $((\lambda s.M^t)^{(0)}N^t)^{(h_2)}$ in y_{i-1}^t by

$$[\int_{\Omega(0)}^{s} M^t]_0$$

without changing D_∞-value of y_{i-1}^t. Since

$$[\int_{\Omega(0)}^{s} M^t]_0 \underset{D_\infty}{\subseteq} [\int_{[N^t]_0}^{s} M^t]_0 \quad ,$$

$$y_{i-1}^t \underset{D_\infty}{\subseteq} y_i^t \quad .$$

By induction, we conclude that, for any i such that $1 \leqq i \leqq q$,

$$y_i^t \in \underline{I}(x_i) \text{ and } y_{i-1}^t \subseteq y_i^t \quad .$$

Thus we have proved that there exists a typed λ-expression y_q^t in $\underline{T}(Q_1^n)$ such that $x^t \underset{D_\infty}{\subseteq} y_q^t$. Also it is easy to see that, for i such that $1 \leqq i \leqq p$, y_i^t matches x_i^t except at occurrences of Ω in x_i^t.

Now we apply typed α-conversions to y_q^t which correspond to $Q_1^n \xrightarrow{\alpha} Q_2^n$. Let this result be $y^t \in \underline{T}(Q_2^n)$. (So, $y_q^t \xrightarrow{t\alpha} y^t$.)

Next, let us develop $Q_2^n \xrightarrow{\eta\text{-ab}} W^n$ into the sequence:

$$Q_1^n = Y_1 \to Y_2 \to \cdots \to Y_{d-1} \to Y_d = W^n$$

where Y_i is derived from Y_{i-1} by an application of η-abstraction.

Correspondingly, we define a sequence of typed λ-expression

$$Y_1^t \to Y_2^t \to \cdots \to Y_d^t$$

as follows:

<u>Step 1</u>: $Y_1^t = y^t$

<u>Step i</u>: Suppose that $Y_{i-1} \xrightarrow{\eta\text{-ab}} Y_i$ is the replacement of A in Y_{i-1} by $\lambda s.As$ and that Y_{i-1}^t is in $\underline{T}(Y_{i-1})$. Let $A^t \in \underline{T}(A)$ be the corresponding occurrence of A in Y_{i-1}^t. We replace it with $(\lambda s.(A^t s^{([j-1])})^{([j-1])})^{(j)}$ to have Y_i^t. Now it is easy to see that Y_i^t is in $\underline{T}(Y_i)$ and $Y_{i-1}^t \underset{D_\infty}{=} Y_{i-1}^t$. We set $Y^t = Y_d^t$.

We have proved here that there exists a typed expression Y^t in $\underline{T}(W^n)$ so that $x^t \underset{D_\infty}{\subseteq} Y^t$. Since Y^t is in $\underline{T}(W^n)$, we name each component of Y^t via the corresponding component of W^n, that is,

$$\begin{array}{llllll} & \underline{t[z^{\delta}]} & \text{in} & Y^t & \text{corresponds to} & z^{\delta} \text{ in } W^n \\ \text{and} & & & & & \\ & \underline{t[W^n_{\delta}]} & \text{in} & Y^t & \text{corresponds to} & W^n_{\delta} \text{ in } W^n \quad . \end{array}$$

Our next stage is to transform Y^t into another typed expression in $\underline{T}(A^n_p(x,T))$ without violating $x^t \underset{D_\infty}{\subseteq} Y^t$. Consider all $t[z^{\delta}]$ in Y^t such that $|\delta| = m_1$. As we have observed as for W^n,

$$\text{either} \quad \delta \in \eta^n_1(m_1), \quad \eta^n_2(m_1) \quad \text{or} \quad \eta^n_3(m_1) \quad ,$$

but it is impossible that $\delta \in \eta^n_1(m_1)$ because of the definition of m_1. Let $\delta \in \eta^n_2(m_1)$. Then z^{δ} has been derived in $x_p \xrightarrow{\beta} Q^n_1$.

Since x^t_p matches y^t_p except at occurrences of Ω in x^t_p and x^t_p has no β-redex, every β-redex of y^t_p is in a part which has no corresponding part in x^t_p except Ω. Thus, by replacing each β-redex in y^t_p by $\Omega^{(0)}$, $x^t_p \underset{D_\infty}{\subseteq} y^t_p$ still holds. Since $t[z^{\delta}]$ in Y^t was derived from some β-redexes in y^t_p, we can reassign to $t[z^{\delta}]$ the minimum type 0 and still have $x^t = x^t_p \subseteq Y^t$.

For each δ in $\eta^n_2(m_1)$, $t[W^n_{\delta}]$ is as

$$(\cdots((t[z^{\delta}]t[W^n_{\delta\circ 1}])^{(*)}t[W^n_{\delta\circ 2}])^{(*)}\cdots)^{(*)}t[W^n_{\delta\circ\gamma_T(\delta)}])^{(*)}$$

in Y^t where $(*)$'s are types.

Since $t[z^{\delta}]$ is now of type 0, we can replace each $t[W^n_{\delta\circ i}]$ $(i = 1,2,\ldots,\gamma_T(\delta))$ by $\Omega^{(0)}$ without affecting the D_∞-value of Y^t. (If $a \in \pi_0(D_\infty)$, $(\cdots((ab_1)b_2)\cdots)b_n = (\cdots((a\bot)\bot)\cdots)\bot$ by Theorem 2.2.5.) So, at least, we can reassign the minimum type 0 to each component of $t[W^n_{\delta\circ i}]$ in Y^t and still have $x^t \underset{D_\infty}{\subseteq} Y^t$.

Especially, for $\delta' \in T$ with $|\delta'| = n-1$ for which there exists $\delta \in \eta_2^n(m_1)$ such that $\delta < \delta'$, $t[z^{\delta'}]$ is of type 0.

Next, we consider δ such that $\delta \in \eta_3^n(m_1)$. Then z^{δ} was derived in $Q_2^n \xrightarrow{\eta\text{-ab}} W^n$. Since the highest type among those that are attached to the components of x^t is m_2 and the conversion $x^t \to y^t \in \underline{T}(Q_2^n)$ does not increase any type, the highest type in y^t is not more than m_2. This means that $t[z^{\delta}]$ is of type at most $m_2 - 1$ by the way the sequence $Y_1^t \to Y_2^t \to \cdots \to Y_d^t$ is defined.

On the other hand, if $t[W_\delta^n]$ is as

$$(\cdots((t[z^{\delta}]t[W^n_{\delta\circ 1}])^{(*)}t[W^n_{\delta\circ 2}])^{(*)}\cdots)^{(*)}t[W^n_{\delta\circ\gamma_T(\delta)}])^{(*)}$$

in Y^t, since z^{δ} comes from η-abstraction, so does each $W^n_{\delta\circ i}$ $(i = 1,2,\ldots,\delta\circ\gamma_T(\delta))$ and, so, $W^n_{\delta\circ i}$ is, in fact, the variable $z^{\delta\circ i}$. Since $t[z^{\delta}]$ is of type at most $[m_2-1]$,

$$\begin{aligned} &t[z^{\delta\circ 1}] \text{ is of type at most } [m_2-2] \\ &\qquad\vdots \\ &t[z^{\delta\circ i}] \text{ is of type at most } [m_2-(i+1)] \quad , \end{aligned}$$

again by the way the sequence $Y_1^t \to Y_2^t \to \cdots \to Y_d^t$ is defined. This indicates that if we take $\delta' \in T$ with $|\delta'| = m_1 + s$ for which there is $\delta \in \eta_3^n(m_1)$ such that $\delta < \delta'$, then $t[z^{\delta'}]$ is of type at most $[m_2-(s+1)]$. Especially, if $s = m_2 - 1$, (i.e. $|\delta'| = n-1$) then $t[z^{\delta'}]$ is of type at most 0.

What we have proved is that we have transformed Y^t so that, for any $\delta \in T$ such that $|\delta| = n-1$, the type of $t[z^{\delta}]$ in Y^t is 0. So we can replace each $t[W_\delta^n]$ with $|\delta| = n$ by $\Omega^{(0)}$ without affecting the D_∞-value of Y^t.

Finally, we replace $t[W_\delta^n]$ in Y^t by $\Omega^{(0)}$ if $C(x,\delta) = \omega$. Since W_δ^n in W^n has no head normal form, this transformation does not affect the D_∞-value of Y^t, either.

We remember that W^n matches $T^n(x)$ except at occurrences of Ω and $LT(x,\delta)$ $(|\delta| = n)$ in $T^n(x)$. So W^n matches $A_p^n(x,T)$ except at occurrences of Ω in $A_p^n(x,T)$ since each $L(x,\delta)$ $(|\delta| = n)$ in $T^n(x)$ is replaced by Ω in $A_p^n(x,T)$.

It follows that $Y^t \in \underline{T}(A_p^n(x,T))$.

We conclude that

$$x^t \underset{D_\infty}{\subseteq} Y^t \underset{D_\infty}{\subseteq} A_p^n(x,T) \quad . \quad \square$$

4.2.2 <u>Lemma</u> (Convergence Lemma). Let x be in Λ and T be a Δ-tree admissible to x. Then $x \underset{D_\infty}{=} \bigcup_{n=0}^{\infty} A_p^n(x,T)$.

<u>Proof</u>. Since $A_p^n(x,T) \underset{D_\infty}{\subseteq} T^n(x) \underset{D_\infty}{=} x$, for any n,

$$\bigcup_{n=0}^{\infty} A_p^n(x,T) \underset{D_\infty}{\subseteq} x \quad .$$

On the other hand, by Lemma 4.2.1, for all $x^t \in \underline{T}(x)$, there exists n such that $x^t \underset{D_\infty}{\subseteq} A_p^n(x,T)$. By Lemma 2.3.9, $x \underset{D_\infty}{=} \bigcup \underline{T}(x)$. Thus $x \underset{D_\infty}{\subseteq} \bigcup_{n=0}^{\infty} A_p^n(x,T)$. $\square$

4.2.3 <u>Corollary</u> (Park). Let Y be the fixed point operator. Then $Yf \underset{D_\infty}{=} \bigcup_{n=0}^{\infty} f^n(\perp)$ for $f \in \Lambda$.

Proof. Let $T = \{\delta \mid \delta = (\underbrace{111\cdots1}_{n})$ for some $n \in \mathbb{N}\} \cup \{0\}$. Then $A_p^n(Yf,T) = \underbrace{f(f(f(\cdots(}_{n}f(\Omega)\cdots))))$. The result is immediate from Lemma 4.2.2. $\square$

§3. Characterization of D_∞-Value of λ-Expressions

In this section, we prove the converse of Corollary 3.4.11 and 3.4.12 using the Convergence Lemma 4.2.2.

4.3.1 Theorem. Let x, y be in Λ. If $C(x) \leq C(y)$, then $x \underset{D_\infty}{\subseteq} y$.

Proof. Suppose $C(x) \leq C(y)$. By Corollary 3.4.9, $\hat{C}(x) \leq \hat{C}(y)$. We take a sufficiently large Δ-tree, T, which is admissible to both x and y. We compare $A_p^n(x,T)$ and $A_p^n(y,T)$. Since $\hat{C}(x) \leq \hat{C}(y)$, $A_p^n(x,T)$ matches $A_p^n(y,T)$ except at occurrences of Ω in $A_p^n(x,T)$ by Definition 4.1.12. So

$$A_p^n(x,T) \underset{D_\infty}{\subseteq} A_p^n(y,T) \quad .$$

By Lemma 4.2.2, $x \underset{D_\infty}{\subseteq} y$. □

4.3.2 Theorem.* For x, y in Λ, $x \underset{D_\infty}{\subseteq} y$ if and only if $C(x) \leq C(y)$, and so, $x \underset{D_\infty}{=} y$ if and only if $C(x) = C(y)$.

Proof. By Corollary 3.4.11 and Theorem 4.3.1. □

4.3.3 Example. Let $Y_0 = Y$, the fixed point operator, and define inductively $Y_i = Y_{i-1}G$ where $G = \lambda x \lambda f.f(xf)$. We can show that $Y_i \underset{D_\infty}{=} Y_j$ for any pair (i,j). For example, we prove $C(Y_0) = C(Y_1)$.

$$Y_0 = Y \xrightarrow{\beta} \lambda f.f((\lambda h.f(hh))(\lambda h.f(hh)))$$
$$\xrightarrow{\beta} \lambda f.f^n((\lambda h.f(hh))(\lambda h.f(hh))) \quad .$$

*Refer to [24] for an alternative characterization of $\underset{D_\infty}{\subseteq}$.

Let G^* denote $(\lambda h.G(hh))(\lambda h.G(hh))$. Note that $G^* \xrightarrow{\beta} G((\lambda h.G(hh))(\lambda h.G(hh)))$.

$$\begin{aligned} Y_1 = YG \xrightarrow{\beta} G^* \xrightarrow{\beta} GG^* \xrightarrow{\beta} \lambda f.f(G^*f) \xrightarrow{\beta} \lambda f.f(GG^*f) \\ \xrightarrow{\beta} \lambda f.f(f(G^*f)) \xrightarrow{\beta} \lambda f.f^n(G^*f) \quad . \end{aligned}$$

Now it is obvious that $C(Y_0) = C(Y_1)$. We can see the proof for $Y_i \not\Leftrightarrow Y_j$ $(i \neq j)$ in [3].

4.3.4 <u>Example</u> (Wadsworth). Let $F = \lambda f\lambda x\lambda y.x(fy)$ and $J = YF$. So

$$J \xrightarrow{\beta} \lambda x\lambda y.x(Jy) \quad .$$

Let $I = \lambda x.x$. Then it is easy to show that $C(I) = C(J)$. Thus $I \underset{D_\infty}{=} J$. Obviously $I \not\Leftrightarrow J$. It was a surprising fact that a normal expression I is equal to a non-normal expression J. J might be considered to be an infinite computation process.
Given an input, it returns the computation result little by little taking an infinite amount of time. The limit of this infinite computation turns out to be equal to the computation of I. The conversion rules alone cannot describe the outcome of this infinite computation. It is possible only after Λ is mapped into a lattice space such as D_∞ where the limit of such infinite computation can exist. As Scott claims in [16], $\underset{D_\infty}{=}$ is a more essential relation than the convertibilities. Further discussion on computational interpretations of normality, non-normality, head-normality of λ-expressions will be given in Chapter 5.

§4. Further Properties of Λ Mapped in D_∞

In this section, we state further properties of Λ mapped in D_∞ lattice. These properties will be the basis of the theory on lattice Λ^∞ which will be introduced in Chapter 6.

4.4.1 Theorem. Let x be in Λ. Suppose $C(x,\delta) \neq \omega$ for any $\delta \in \Lambda$, then x is maximal in Λ, that is, there is no y in Λ such that $x \underset{D_\infty}{\subsetneq} y$.

Proof. If $x \underset{D_\infty}{\subseteq} y$ for some y in Λ, it must be that $C(x) \leq C(y)$ by Theorem 4.3.2. Since there is no $\delta \in \Delta$ such that $C(x,\delta) = \omega$, $x \underset{D_\infty}{=} y$. □

4.4.2 Corollary. Let x be in Λ. If x has a normal form, then x is maximal in Λ.

Proof. If x is normal, $C(x,\delta) \neq \omega$ for any $\delta \in \Delta$. □

4.4.3 Definition.

1. Let $\mathcal{D}$ be a subset of Λ. $\mathcal{D}$ is said to be directed (with respect to D_∞ partial order) if $\mathcal{D}$ satisfies the following property: For F any finite subset of $\mathcal{D}$, there exists an element z of $\mathcal{D}$ such that, for each $x \in F$,

$$\mathbb{V}[\![x]\!]\rho \subseteq \mathbb{V}[\![z]\!]\rho$$

for all environments ρ.

2. Let $\mathcal{D} \subset \Lambda$ be directed. $\mathcal{D}$ is said to be interesting if there is no x in $\mathcal{D}$ for which

$$\mathbb{V}[\![x]\!]\rho = \cup\{\mathbb{V}[\![y]\!]\rho \mid y \in \mathcal{D}\}$$

for all environments ρ.

The following theorem is a generalization of Lemma 4.2.2.

4.4.4 <u>Theorem</u> (General Convergence Lemma). Let $\mathcal{D}$ be a directed subset of Λ. We define $c_{\mathcal{D}} \in \mathbb{C}$ by:

$$c_{\mathcal{D}}(\delta) = \begin{cases} \omega & \text{if } C(y,\delta) = \omega \text{ for all } y \in \mathcal{D} \\ z \in V & \text{if } C(y,\delta) = z \text{ for some } y \in \mathcal{D}. \end{cases}$$

Then $c_{\mathcal{D}} = C(x)$ for $x \in \Lambda$ if and only if $x \underset{D_\infty}{=} \cup\mathcal{D}$.

<u>Proof</u>. To prove that $c_{\mathcal{D}}$ is well defined, assume that, for some $y_1, y_2 \in \mathcal{D}$ and $\delta \in \Delta$, $C(y_1,\delta) \neq \omega$, $C(y_2,\delta) \neq \omega$ and $C(y_1,\delta) \neq C(y_2,\delta)$. Since $\mathcal{D}$ is directed, there must be z in $\mathcal{D}$ for which both $y_1 \underset{D_\infty}{\subseteq} z$ and $y_2 \underset{D_\infty}{\subseteq} z$, but this is impossible by Theorem 4.3.2. So given any $\delta \in \Delta$, either $C(y,\delta) = \omega$ for all $y \in \mathcal{D}$ or there is $v \in V$ such that $C(y,\delta) = v$ for all $y \in \mathcal{D}$ such that $C(y,\delta) \neq \omega$.

Let T be a Δ-tree which is admissible to x. Now suppose $c_{\mathcal{D}} = c(x)$. For any $\delta \in T$, there is at least one y in $\mathcal{D}$ such that $C(x,\delta) = C(y,\delta)$. Given n in $\mathbb{N}$, since $\#\{\delta \mid |\delta| < n, \delta \in T\}$ is finite, there is a finite subset, F, of $\mathcal{D}$ such that, for any $\delta \in T$ with $|\delta| < n$, there is at least one y in F for which $C(x,\delta) = C(y,\delta)$. By directedness of $\mathcal{D}$, there is z in $\mathcal{D}$ such that, for any $y \in F$, $y \underset{D_\infty}{\subseteq} z$. It follows that $C(A_p^n(x,T)) \leq C(z)$. So, by Theorem 4.3.1, $A_p^n(x,T) \underset{D_\infty}{\subseteq} z$. By

Lemma 4.2.2, $x = \bigcup_{D_\infty}\bigcup_{n=0}^{\infty} A_p^n(x,T)$. Thus $x \sqsubseteq_{D_\infty} \bigcup\mathcal{D}$. On the other hand, since $C(y) \leq C(x)$ for any $y \in \mathcal{D}$, $\bigcup\mathcal{D} \sqsubseteq_{D_\infty} x$.

Conversely, suppose that $x =_{D_\infty} \bigcup c_{\mathcal{D}}$. By Theorem 4.3.1, $C(y) \leq C(x)$ for all $y \in c_{\mathcal{D}}$. Assume, for some $\delta \in \Delta$, $C(x,\delta) \neq \omega$ and $C(y,\delta) = \omega$ for all $y \in \mathcal{D}$. Using the fact that $c_{\mathcal{D}}$ is directed, a discussion similar to the proof of Lemmas 3.4.5 and 3.4.6 leads us to prove that there exists an environment ρ and $e_1, e_2, \ldots, e_n \in \Lambda_c$ such that

$$\perp = \mathbb{V}[\![ye_1e_2\cdots e_n]\!]\rho \subsetneq \mathbb{V}[\![xe_1e_2\cdots e_n]\!]\rho$$

for all $y \in c_{\mathcal{D}}$. So, under ρ,

$$(\cup c_{\mathcal{D}})e_1e_2\cdots e_n = \perp \subsetneq xe_1e_2\cdots e_n \ .$$

This means that $\cup c_{\mathcal{D}} \subsetneq x$ contradicting the assumption. $\square$

However, it is not always the case that a directed subset of Λ has a least upper bound in Λ. But as we see in the next theorem, every element of Λ that is not the bottom is the least upper bound of a directed subset of Λ which does not include the original element. In Chapter 6, this situation will be discussed more uniformly.

4.4.5 <u>Definition</u>. Let D be a directed-complete lattice and F be a directed subset of D. Then F is said to be <u>interesting</u> if F does not contain its own least upper bound, i.e. $\cup F \notin F$.

Note that any finite directed subset is not interesting and that any infinite non-interesting directed subset can become interesting by removing its least upper bound.

4.4.6 <u>Theorem</u>. Let x be in Λ. If x has a head normal form, then there is a subset $\mathcal{D}$ of Λ which is an interesting directed set such that

$$x \underset{D_\infty}{=} \cup \mathcal{D} \quad .$$

<u>Proof</u>. We take a sufficiently large Δ-tree T so that

1) T is admissible to x.

2) For any $n > 0$, there exists at least one $\delta \in T$ with $|\delta| = n$, for which $C(x,\delta) \neq \omega$.

For example, T as defined below satisfies 1) and 2) above: Take any Δ-tree T' admissible to x. Let T be a Δ-tree which includes $T' \cup \{\delta \circ (\gamma_{T'}(\delta)+1) \mid \delta \in T\}$. In the first place, it is obvious that such a Δ-tree exists. In the second place T is admissible to x. Thirdly, since T' is admissible, each $L(x,\delta)$ with $\delta \in T - T'$ is obtained by η-abstraction, and so, in fact, $L(x,\delta)$ is $t_{\delta \circ k}$ for some $k > 0$.

Since $\{\delta |\ |\delta| = n,\ \delta \in T - T'\} \neq \emptyset$ for each n by the definition of T, it follows that T satisfies 2), too. Since there is at least one $\delta \in T$ with $|\delta| = n$ for each n such that $C(x,\delta) \neq \omega$ and $C(A_p^n(x,\delta),T) = \omega$, we conclude that $A_p^n(x,T) \underset{D_\infty}{\subsetneq} x$ for all n by Theorem 4.3.2.

On the other hand, $x \underset{D_\infty}{=} \bigcup_{n=0}^{\infty} A_p^n(x,T)$ by Lemma 4.2.2. Thus,

$$\mathcal{D} = \{A_p^0(x,T), A_p^1(x,T), A_p^2(x,T), \ldots\}$$

is an interesting directed set whose least upper limit is x. □

4.4.7 <u>Theorem</u>. Given $x, y \in \Lambda$ such that $x \underset{D_\infty}{\subsetneq} y$. Then there is $z \in \Lambda$ with $x \underset{D_\infty}{\subsetneq} z \underset{D_\infty}{\subsetneq} y$.

<u>Proof</u>. Let T be a Δ-tree which is admissible to both x and y. Let $n \in \mathbb{N}$ be such that there exists δ with $|\delta| = n$ for which $C(x,\delta) = \omega$ and $C(y,\delta) \neq \omega$. By Definition 4.1.3, $T^n(y)$ contains $L(y,\delta)$ as a subexpression. Since $C(y,\delta) \neq \omega$,

$$L(y,\delta) \xrightarrow[\beta]{} \lambda s_1 s_2 \cdots s_p . z Y_1 Y_2 \cdots Y_q \quad .$$

Let z be derived from $T^n(y)$ by replacing $L(y,\delta)$ in $T^n(y)$ by $\lambda s_1 s_2 \cdots s_p s_{p+1} . z Y_1 Y_2 \cdots Y_q \Omega$ where s_{p+1} does not appear free in $z Y_1 Y_2 \cdots Y_q$. Now it is easy to see

$$x \underset{D_\infty}{\subsetneq} z \underset{D_\infty}{\subsetneq} T^n(y) \underset{D_\infty}{=} y \quad . \quad □$$

The following fact is interesting in relation to ω-completeness discussions in Barendregt [2] and Plotkin [11].

4.4.8 Theorem. Let $x, y \in \Lambda$. If, for any z in Λ_c, $xz \underset{D_\infty}{=} yz$, then $x \underset{D_\infty}{=} y$.

Proof. Suppose $x \underset{D_\infty}{\neq} y$. By Theorem 4.3.2, $C(x) \neq C(y)$. By Theorem 3.4.7, there exist $e_1, e_2, \ldots, e_n \in \Lambda$ and an environment ρ such that

$$\mathbb{V}[\![xe_1e_2\cdots e_n]\!]\rho \underset{D_\infty}{\neq} \mathbb{V}[\![ye_1e_2\cdots e_n]\!]\rho \quad . \qquad (*)$$

Since, by Proposition 2.3.5 and Corollary 2.3.18, the $\underset{D_\infty}{\neq}$ can be realized with both sides being in Λ_c, so we can choose ρ such that $\rho(V) \subset \Lambda_c$.

Let

$$\bar{x} = \int_{\rho(u_1),\rho(u_2),\ldots,\rho(u_p)}^{u_1,u_2,\ldots,u_p} x \quad ,$$

$$\bar{y} = \int_{\rho(v_1),\rho(v_2),\ldots,\rho(v_q)}^{v_1,v_2,\ldots,v_q} y \quad ,$$

and

$$\bar{e}_i = \int_{\rho(w_1^i),\rho(w_2^i),\ldots,\rho(w_{m(i)}^i)}^{w_1^i,w_2^i,\ldots,w_{m(i)}^i} e_i$$

where $u_1, u_2, \ldots, u_p$ are the free variables occurring in x,

$v_1, v_2, \ldots, v_q$ are the free variables in y and $w_1^i, w_2^i, \ldots, w_{m(i)}^i$ are the free variables in e_i for $i = 1,2,\ldots,n$.

Now the inequality (*) is written as:

$$\bar{x}\bar{e}_1\bar{e}_2\cdots\bar{e}_n \underset{D_\infty}{\neq} \bar{y}\bar{e}_1\bar{e}_2\cdots\bar{e}_n$$

where $\bar{x}$, $\bar{y}$, $\bar{e}_i \in \Lambda_c$. By extensionality of D_∞, we conclude that

$$\bar{x}\bar{e}_1\cdots\bar{e}_{n-1} \underset{D_\infty}{\neq} \bar{y}\bar{e}_1\cdots\bar{e}_{n-1}$$

and so

$$\bar{x}\bar{e}_1\cdots\bar{e}_{n-2} \underset{D_\infty}{\neq} \bar{y}\bar{e}_1\cdots\bar{e}_{n-2}$$
$$\vdots$$
$$\bar{x}\bar{e}_1 \underset{D_\infty}{\neq} \bar{y}\bar{e}_1 \quad .$$

This indicates that if $x \underset{D_\infty}{\neq} y$, there exists $\bar{e}_1 \in \Lambda_c$ for which

$$x\bar{e}_1 \underset{D_\infty}{\neq} y\bar{e}_1 \quad . \quad \square$$

We translate this theorem into one which is stated by C-functions:

4.4.9 <u>Corollary</u>. Let x, y be in Λ. If, for any $z \in \Lambda_c$, $C(xz) = C(yz)$, then $C(x) = C(y)$. $\square$

Theorem 4.3.5 is obvious if we replace $z \in \Lambda_c$ by $z \in D_\infty$ since D_∞ is extensional. The theorem says that the extensionality holds in Λ_c modulo $\underset{D_\infty}{=}$.

CHAPTER 5

INFINITE EXPANSIONS IN REAL PROGRAMMING LANGUAGES

We discuss informally how the concepts and formulations introduced for the λ-calculus in the previous chapters are applied to more realistic programming languages such as recursively defined programs and Algol-like programs. We present algorithms to translate a program written in these languages into a λ-expression. Using this translation, we show that the C-function for the λ-expressions, in fact, corresponds to the infinite expansion or the executions on all possible inputs for the programs in realistic languages. Most of the results here are not essentially new.

§1. Recursively Defined Programs

We consider the programs which are defined by recursive equations. (Discussions on this type of program are found, for example, in [20]). The following is the syntax of the language $\mathcal{R}$.

Syntax of $\mathcal{R}$

Elements

1. $A_1, A_2, \ldots$: Symbols for constants
2. $X_1, X_2, \ldots, X_\ell$: Symbols for variables
3. $G_1, G_2, \ldots, G_m$: Symbols for known functions
4. $F_1, F_2, \ldots, F_n$: Symbols for unknown functions

$$
\begin{aligned}
\langle \text{term} \rangle ::= \; & A_1 | A_2 | \cdots \\
& | X_1 | X_2 | \cdots | X_\ell | \\
& | G_1(\langle \text{term } 1 \rangle, \ldots, \langle \text{term } k_1 \rangle) \\
& \;\vdots \\
& | G_m(\langle \text{term } 1 \rangle, \ldots, \langle \text{term } k_m \rangle) \\
& | F_1(\langle \text{term } 1 \rangle, \ldots, \langle \text{term } p_1 \rangle) \\
& \;\vdots \\
& | F_n(\langle \text{term } 1 \rangle, \ldots, \langle \text{term } p_n \rangle)
\end{aligned}
$$

$$
\langle \text{program} \rangle ::= \begin{cases} F_1(X_1, X_2, \ldots, X_{p_1}) \leftarrow \langle \text{term } 1 \rangle \\ \quad\quad \vdots \\ F_n(X_1, X_2, \ldots, X_{p_n}) \leftarrow \langle \text{term } n \rangle \end{cases}
$$

where we assume that <term k> does not contain any variable symbol other than $X_1, X_2, \ldots, X_{p_k}$ for $k = 1, 2, \ldots, n$.

For example

$$n = 1: \quad F(X) \leftarrow F(F(X))$$

$$n = 2: \begin{cases} F_1(X_1,X_2) \leftarrow G_1(X_2,F_2(X_1)) \\ F_2(X_1) \leftarrow G_2(F_2(X_1),F_1(X_1)) \end{cases}$$

are programs of $\mathcal{R}$.

<u>Computation of Terms</u>

Given a program ξ where ξ is:

$$\begin{cases} F_1(X_1,X_2,\dots,X_{k_1}) \leftarrow \Psi_1(X_1,X_2,\dots,X_{k_1}) \\ \vdots \\ F_n(X_1,X_2,\dots,X_{k_n}) \leftarrow \Psi_n(X_1,X_2,\dots,X_{k_n}) \end{cases}$$

where $\Psi_i(X_1,\dots,X_{k_i})$ is a term with occurrences of $X_1,X_2,\dots,X_{k_i}$ for $i = 1,2,\dots,n$. For a term T, a <u>computation of</u> T <u>according to</u> ξ is defined to be a sequence of terms:

$$T_1,T_2,\dots,T_n$$

where $T_1 = T$ and T_i is obtained from T_{i-1} by replacing an occurrence of $F_j(s_1,s_2,\dots,s_{k_j})$ by $\Psi_j(s_1,s_2,\dots,s_{k_j})$ where s_i's are terms. We write $T \xrightarrow[\xi]{} T_n$.

<u>Translation of $\mathcal{R}$ to Λ</u>

Given a term T and a program ξ, we want to synthesize a λ-expression $\Sigma_\xi(T)$ such that each computation of T according to ξ corresponds to a β-reduction sequence from $\Sigma_\xi(T)$.

Let $a_1,a_2,\ldots,\ x_1,x_2,\ldots,x_\ell,\ g_1,g_2,\ldots,g_m,\ \phi_1,\phi_2,\ldots,\phi_n$ be distinct variables in Λ.

<u>Algorithm</u>

(a) If T is A_i,

$$\Sigma'_\xi(T) = a_i \ .$$

(b) If T is X_i, then $\Sigma'_\xi(T) = x_i$.

(c) If T is $G_i(s_1,s_2,\ldots,s_{p_i})$ where $\Sigma'_\xi(s_j) = S_j$ for $j = 1,2,\ldots,p_i$, then $\Sigma'_\xi(T) = g_i(S_1)(S_2)\cdots(S_{p_i})$. The parentheses are omitted if S_j is a variable.

(d) If T is $F_i(s_1,s_2,\ldots,s_{k_i})$ then $\Sigma'_\xi(T) = \phi_i(S_1)(S_2)\cdots(S_{k_i})$.

By applying the transformations (a), (b), (c), we obtain a λ-expression $\Sigma'_\xi(T) \in \{a_i,g_j,\phi_k\}^*$ for a term T. Next we substitute a λ-expression for $\phi_1,\phi_2,\ldots,\phi_n$ in $\Sigma'(T)$. For example we see what to substitute for ϕ_1.

Let

$$\begin{aligned}
y_n &= Y\left(\lambda\phi_n.\ \Sigma'_\xi(\Psi_n)\right)\\
y_{n-1} &= Y\left(\lambda\phi_{n-1}.\int_{y_n}^{\phi_n}\Sigma'_\xi(\Psi_{n-1})\right)\\
&\ \ \vdots\\
y_i &= Y\left(\lambda\phi_i.\int_{y_{i+1},y_{i+2},\ldots,y_n}^{\phi_{i+1},\phi_{i+2},\ldots,\phi_n}\Sigma'_\xi(\Psi_i)\right)\\
&\ \ \vdots\\
y_1 &= Y\left(\lambda\phi_1.\int_{y_2,y_3,\ldots,y_n}^{\phi_2,\phi_3,\ldots,\phi_n}\Sigma'_\xi(\Psi_1)\right)
\end{aligned}$$

Then $Y_1 = y_1$ is the λ-expression we want to substitute for ϕ_1. In the similar manner, we synthesize a λ-expression Y_i to substitute for ϕ_i $(i = 1,2,\ldots,n)$. Now

$$\Sigma_\xi(T) = \int_{Y_1,Y_2,\ldots,Y_n}^{\phi_1,\phi_2,\ldots,\phi_n} \Sigma'_\xi(T) \quad .$$

It is easy to see that:

5.1.1 <u>Theorem</u>. For terms T_1, T_2 and a program ξ, $T_1 \xrightarrow[\xi]{} T_2$ if and only if $\Sigma_\xi(T_1) \xrightarrow{\beta} \Sigma_\xi(T_2)$. □

To translate the result of Chapter 4 to R, we introduce the notion of semantics to R.

Theorem 4.3.2 can be read as:

"Given λ-expressions (programs) x, y, if x and y have the same C-function (infinite expansion), then x is equivalent to y under the interpretation of the D_∞-semantics."

Here, instead of D_∞, we use general domains to specify the semantics of R (as in [20]).

<u>Semantics of R</u>

We define an <u>interpretation</u>, I, of R to be the pair $(\mathcal{D}_I, v_I)$ where $\mathcal{D}_I$ is a directed complete partially ordered set with $\perp = \cap\mathcal{D}_I$ and v_I is the semantic function which maps: the constant symbols A_i to elements a_i, the variable symbols X_i to variables x_i which range over $\mathcal{D}_I$, the known function symbols

G_i to $g_i \in [\mathcal{D}_I \to \mathcal{D}_I]$. v_I is extended in the obvious manner to the terms generated from G_i, A_j and X_k. Now a program ξ of $\mathcal{R}$ can be translated via v_I into an equation ξ^* with the unknown functions F_i's over $\mathcal{D}_I$. By Scott's fixed point theorem (Theorem 2.2.3) we can conclude that there exist continuous functions

$$f_1, f_2, \ldots, f_n \in [\mathcal{D}_I \to \mathcal{D}_I]$$

which satisfy ξ^*. By way of the correspondence $F_i \mapsto f_i$ we extend the definition of v_I onto all terms of $\mathcal{R}$. We denote this extension by v_I^{ξ}.

Now we have the following fact which corresponds to Theorem 4.3.2:

"Given programs ξ_1, ξ_2 and terms T_1, T_2, then $v_I^{\xi_1} [\![T_1]\!] = v_I^{\xi_2} [\![T_2]\!]$ for all interpretation I if and only if

$$C(\Sigma_{\xi_1}(T_1)) = C(\Sigma_{\xi_2}(T_2)) \text{ ,"}$$

which says the semantic equivalence can be described by the equivalence of the infinite expansion (C-function). It is easy to see that "given a program ξ and a term T, $v_I^{\xi} [\![T]\!] = \bot_{\mathcal{D}_I}$ for all interpretations I if and only if $\Sigma_{\xi}(T) \in \Lambda$ has no head normal form."

On the other hand, in a straightforward application of the formal language theory, we see that the property:

$$P(\xi,T) \equiv \text{"} v_I^\xi [\![T]\!] = \bot_{\mathcal{D}_I} \text{ for all interpretations } I\text{"}$$

is decidable. So the head normality is a decidable property on R.

However, we do not know the answer to the following* question:

"Given programs ξ_1, ξ_2 and terms T_1, T_2, is it decidable whether or not

$$C(\Sigma_{\xi_1}(T_1)) = C(\Sigma_{\xi_2}(T_2)) \text{ ?"}$$

which is equivalent to:

"Given programs ξ_1, ξ_2 and terms T_1, T_2, is it decidable whether or not for <u>all</u> interpretations I

$$v_I^{\xi_1} [\![T_1]\!] = v_I^{\xi_2} [\![T_2]\!] \text{ ? "}$$

This property is, of course, undecidable on Λ, for, the head normality is already undecidable on Λ.

*This problem is equivalent to the equivalence problem of the deterministic pushdown automata. Refer to B. Courcelle, "Recursive schemes, algebraic trees, deterministic languages," <u>Proceedings of the 15th SWAT Symposium</u> (1974).

§2. Algol-like Language

We take the translation algorithm based on the continuation technique in [1]. The continuation is explained as follows:

Given a program:

$$S \equiv S_1;S_2;\dots;S_n$$

where S_i's are a (block of) statement(s).

We can regard S as a function over the program domain D and ';' is understood in the following two ways:

(I) Each block S_i is a function over D. Thus ';' is the composition of two functions. Let f_{S_i} be the λ-expression that corresponds to S_i. Then the translation of S is:

$$\lambda x.f_{S_n}(f_{S_{n-1}}(\cdots(f_{S_1}x))) \quad .$$

(II) Consider $S;S'$. Let f' be the function over D which is defined by S'. We regard S as a functional Φ_S which, applied f', yields a new function. So the translation of $S;S'$ is

$$\lambda x.(\Phi_S(f'))(x) \quad .$$

If S' is null, f' is $I = \lambda x.x$. So, for example, the translation of $S_1;S_2$ is

$$\lambda x.(\Phi_{S_1}(\Phi_{S_2}(I)))(x) \quad .$$

For $S = S_1;S_2;\dots;S_n$, we give

$$\lambda x.(\Phi_{S_1}(\Phi_{S_2}(\cdots(\Phi_{S_n}(I))\cdots)(x) \quad .$$

(I) is not accurate when the program contains such statements as goto or halt since, in that case, execution of the program is not necessarily sequential.

Here we show how some of the program constructs of Algol can be translated into λ-expressions based on (II).

Given a program

$$S \equiv S_1;S_2;\ldots;S_n \quad ,$$

each S_i is translated into a λ-expression of the form:

$$s_i \equiv \lambda\phi x_1 x_2 \cdots x_m . \mathcal{S}_i(x_1,x_2,\ldots,x_m,\phi)$$

where $\mathcal{S}_i$ is a λ-expression that contains the variables $x_1,x_2,\ldots,x_m,\phi$. The x_i's are the program variables and ϕ is called the continuation variable and stands for the remaining part of the program execution that follows the execution of S_i.

Now S is translated into:

$$s_1(s_2(\cdots(s_n I))\cdots) \quad .$$

Algorithm. We state the translation algorithm in [1] for some of the important program constructs. For the complete and detailed description, we refer to [1]. For simplicity, we do not consider the block structures and the program is assumed to have the global variables $x_1,x_2,\ldots,x_n$.

i) Assignment: $x_j \leftarrow f(x_1,x_2,\ldots,x_n)$ is translated as:

$$\lambda\phi x_1 x_2 \cdots x_n . \phi x_1 x_2 \cdots x_{i-1} f(x_1,x_2,\ldots,x_n) x_{i+1} \cdots x_n$$

ii) Conditional Statement: if α then S_1 else S_2 where $\alpha \equiv \alpha(x_1,x_2,\ldots,x_n)$ is a Boolean expression. Let s_1

and s_2 be the translation of S_1 and S_2, respectively. Let $<\alpha> \equiv <\alpha(x_1,x_2,\ldots,x_n)>$ be the translation of α such that

$$<\alpha>AB \xrightarrow{\beta} A \quad \text{if} \quad \alpha(x_1,x_2,\ldots,x_n) = \underline{\text{true}}$$
$$<\alpha>AB \xrightarrow{\beta} B \quad \text{if} \quad \alpha(x_1,x_2,\ldots,x_n) = \underline{\text{false}} \quad .$$

Then <u>if</u> α <u>then</u> S_1 <u>else</u> S_2 is translated as

$$\lambda\phi x_1x_2\cdots x_n.<\alpha>((s_1\phi)x_1x_2\cdots x_n)((s_2\phi)x_1x_2\cdots x_n)$$

iii) <u>goto</u> ℓ: We associate a certain part of the program P to each label ℓ. Let m be the label which is defined in P next to ℓ and ℓ and m occur in P as

$$\ell:S_1;S_2;\ldots;S_q;m:S_{q+1} \quad .$$

Then we associate $S_1;S_2;\ldots;S_q$ to ℓ. So the translation of ℓ is:

$$[\ell] \equiv s_1(s_2(\cdots(s_q([m])\cdots)$$

where [m] is the translation of m and s_i is the translation of S_i for $1 \leq i \leq q$. If no label appears after ℓ,

$$[\ell] \equiv s_1(s_2(\cdots(s_q(I))\cdots) \quad .$$

Now <u>goto</u> ℓ is translated as:

$$\lambda\phi x_1x_2\cdots x_n.[\ell]x_1x_2\cdots x_n \quad .$$

Since <u>goto</u> ℓ forgets the statements following itself, ϕ does not occur in $[\ell]x_1x_2\cdots x_n$.

iv) while α do S: This statement can be regarded as W which is recursively defined as:

$$W \equiv \underline{\text{if}}\ \alpha\ \underline{\text{then}}\ \underline{\text{begin}}\ S;\ W\ \underline{\text{end}}\ \underline{\text{else}}\ \text{no action} \quad .$$

Since $\underline{\text{if}}\ \alpha\ \underline{\text{then}}\ \underline{\text{begin}}\ S;\ W\ \underline{\text{end}}$ is translated into

$$\lambda\phi x_1 x_2 \cdots x_n . <\alpha>((s(w\phi))x_1 x_2 \cdots x_n)(\phi x_1 x_2 \cdots x_n) \quad ,$$

the translation of W, w satisfies the equation:

$$w \xrightarrow{\beta} \lambda\phi x_1 x_2 \cdots x_n . <\alpha>((s(w\phi))x_1 x_2 \cdots x_n)(\phi x_1 \cdots x_n) \quad .$$

So

$$w = Y(\lambda f\phi x_1 \cdots x_n . <\alpha>((s(f\phi))x_1 \cdots x_n)(\phi x_1 \cdots x_n) \quad .$$

Example. Consider the following program:

```
begin
   input(x,y);                    (1)
      i := x;                     (2)
      while i>0 do                (3)
         begin y := y^2;          (4)
            if y>x^2 then goto ℓ  (5)
            i := i-1;             (6)
         end;
ℓ: end                            (7)
```

Translation:

(1): We regard the input as an assignment and have

$A \equiv \lambda\phi xyi.\phi abi$

(2): $B \equiv \lambda\phi xyi.\phi xyx$

(4): $C \equiv \lambda\phi xyi.\phi xy^2 i$

$$(5):\quad D \equiv \lambda\phi xyi.\langle y>x^2\rangle([\ell]xyi)(\phi xyi)$$

$$(6):\quad E \equiv \lambda\phi xyi{:}xy\underline{i-1}$$

$$(7):\quad [\ell] \equiv I$$

$$(3):\quad F \equiv Y(\lambda f\phi xyi.\langle i>0\rangle((G(f\phi))xyi)(\phi xyi)) \quad \text{where}$$

$$G \equiv \lambda\phi.C(D(E\phi)).$$

Now the whole program $P \equiv A(B(FI))$. It is easy to see that

$$P \xrightarrow{\beta} \lambda xyi.\langle a>0\rangle(\langle b^2>a^2\rangle(ab^2a)(\langle a-1>0\rangle(\langle b^4>a^2\rangle(ab^4\underline{a-1})(\cdots$$

$$\cdots)ab^2\underline{a-1})aba$$

which shows all possible executions for arbitrary inputs or the infinite expansion of the program.

In the example, one might see the correspondence between β-reductions and program execution.

Next we ask to what programming concept the head-normality and the normality correspond under this translation. If we assume that the computation of each Boolean function terminates, the following is the answer:

"A integral part of a program is translated to a non-head normal λ-expression if and only if under any assignment of the Boolean values (i.e. <u>true</u> and <u>false</u>) to the Boolean functions occurring in the part, execution can never leave the part once it enters it." (Note that this property is, obviously, decidable.)

For example,

$$\ell{:}\ \underline{\text{goto}}\ \ell \quad .$$

This <u>goto</u> statement is translated into

$G \equiv \lambda\phi x_1 x_2 \cdots x_n . [\ell] x_1 x_2 \cdots x_n$ and $[\ell] = \lambda\phi . G(\cdots)$. Obviously, G has no head normal form. On the other hand, a normal expression corresponds to a program that has no loop in it, i.e. no <u>while</u>, no <u>goto</u> that makes a loop. Thus, a normal expression is a program which terminates upon all inputs. On the other hand, a non-normal, head-normal expression corresponds to a program that may or may not terminate depending on the input condition.

Now we have the following observations. Although the discussion to support these conclusions is informal and rather shallow, they might give some intuitive insight and understanding to the formal argument in the rest of the chapters.

(i) The process to generate $C(x)$ from $x \in \Lambda$ corresponds to program expansion or execution upon arbitrary inputs.

(ii) The Ω-conversion corresponds to removal of meaningless parts in the program (such as ℓ: goto ℓ).

(iii) Theorem 4.3.2 is understood as "two programs have the same meaning if (and only if) they have the same infinite expansion."

CHAPTER 6

GENERALIZED λ-EXPRESSIONS,
A NATURAL LATTICE STRUCTURE OF THE λ-CALCULUS

We generalize λ-expressions to the infinite λ-expressions. The results on the λ-expressions in D_∞ are extended to the infinite λ-expressions. It is shown that the lattice structure of the infinite λ-expressions (including the conventional λ-expressions) induced by the D_∞-partial order is equivalent to a directed complete partially ordered set $\mathbb{C}_{inf}$, which can be regarded as the domain of all the infinite expansion of the λ-expressions. Since $\mathbb{C}_{inf}$ is defined independent of D_∞, $\mathbb{C}_{inf}$ can be said to give a natural lattice structure of the λ-calculus.

§1. Infinite Programs

How can a program be infinite? Probably in three ways.

1) non-termination, i.e. run time is infinite.

2) infinite work area, e.g. a Turing Machine is an <infinite-program> in the sense that it has an infinite storage.

3) infinitely many commands, e.g. an Algol program which is textually infinite.

Here, by an infinite program, we mean one in the category 3) above.

However, one may ask how such a program can be realized. In [12], Reynolds presents the following programming environment.

Let us imagine an interactive situation in which a person is programming in front of a terminal. He builds up his program in such a way that some of the integral parts (e.g., inside of a <u>begin</u>-<u>end</u> block, a procedure body, or simply a statement) are left unspecified. He can let the system execute this program. When it turns out that the system needs the specification of an undefined part of the program to continue execution, the programmer is requested to fill it with a code which could have several unspecified parts, too. The programmer meets this request probably considering the outcome of execution he has obtained so far. This process of programming can continue infinitely. Since a person with free will takes part in this process, it can become a non-recursively enumerable object.

If we are to formalize this idea of <infinite programs>, we shall probably have <infinite λ-expressions>. Then, what do infinite λ-expressions look like?

A "λ-like-expression" can be infinite in two ways:

1) Infinitely wide expressions

a) number of applications: We define x by

$$x := (\cdots((A_1A_2)A_3)\cdots)A_n)A_{n+1})\cdots \quad ,$$

that is, x is the outcome of infinite applications

$$\begin{aligned} x_1 &:= A_1 \\ x_2 &:= x_1A_2 \\ &\vdots \\ x_n &:= x_{n-1}A_{n-1} \\ &\vdots \end{aligned}$$

b) number of abstractions: Let x be an expression of the form

$$x := \lambda v_1v_2v_3\cdots v_n\cdots.w \quad ,$$

that is, x is a computation process which, given an infinite sequence of inputs, $\{A_1,A_2,\ldots,A_n,\ldots\}$, returns

$$xA_1 \to \lambda v_2v_3\cdots v_n\cdots. \int_{A_1}^{v_1} w$$

$$xA_1A_2 \to \lambda v_3v_4\cdots v_n\cdots. \int_{A_1,A_2}^{v_1,v_2} w$$

$$\vdots$$

$$xA_1A_2\cdots A_k \to \lambda v_{k+1}v_{k+2}\cdots. \int_{A_1,A_2,\ldots,A_k}^{v_1,v_2,\ldots,v_k} w \, .$$

$$\vdots$$

c) combination of a) and b): For example,

$$x := \lambda v_1 v_2 \cdots v_n \cdots . x_1 x_2 x_3 \cdots \quad ,$$

that is,

$$x := \lambda v_1 v_2 \cdots v_n \cdots . w$$

where $w := (\cdots(x_1 x_2) x_3) x_4) \cdots) x_n) \cdots$.

In a sense, the C-function has this structure. To apply infinitely many η-abstraction is to have an infinite expression:

$$\lambda t_1 t_2 t_3 \cdots . X t_1 t_2 t_3 \cdots$$

from $X \in \Lambda$.

2) Infinitely deep expressions: Consider such an expression as

$$x := x_1(x_2(x_3(\cdots(x_n(\cdots))\cdots) \quad .$$

x would be the outcome of the application

$$x_1 y_2$$

where y_2 would be the outcome of the application

$$x_2 y_3$$

where y_3 $\cdots$

$$\cdots$$

where y_n would be the outcome of the application

$$x_n y_{n+1}$$

where y_{n+1} $\cdots$

$$\cdots \quad .$$

We will mainly study this <infintely deep λ-expression> in this chapter.

It is more likely that <infinitely deep λ-expressions> reflect the infinity of Reynold's infinite program. Let us take the following sequence of Algol-like commands:

$$S \equiv \underline{\text{begin}}\ S_1;S_2;\cdots;S_n\ \underline{\text{end}} \quad .$$

As we saw in Chapter 5, there are two methods to translate S into a λ-expression.

1) Regard S_i $(i = 1,2,\ldots,n)$ as a function: $\mathbb{D} \to \mathbb{D}$. Let s_i be the λ-expression translated from S_i. Since S is the map: $\mathbb{D} \to \mathbb{D}$ which is the composition of all S_i's, the translation of S is:

$$\lambda v.s_n(s_{n-1}(\cdots(s_1(v))\cdots)) \quad .$$

2) Regard S_i $(i = 1,2,\ldots,n)$ as a functional: $(\mathbb{D} \to \mathbb{D}) \to (\mathbb{D} \to \mathbb{D})$. Let s_i be the λ-expression translated from S_i. Using the technique of continuation in Chapter 5, the translation of S is:

$$\lambda v.s_1(s_2(\cdots(s_n(I))\cdots)(v)$$

where I is $\lambda x.x$.

In both 1) and 2), we would have an infinitely deep expression letting $n \to \infty$. (However, here, note that 2) is more appropriate as we see in the following discussion.)

Given an infinite program:

$$S_1;S_2;,\ldots;S_n;\ldots ,$$

this program will probably be the limit of the sequence:

$$\begin{array}{l} S_1;\perp \\ S_1;S_2;\perp \\ \quad\vdots \\ S_1;S_2;\ldots;S_n;\perp \quad . \end{array}$$

Using 2), we have

$$\begin{array}{l} \lambda v.(s_1(\perp))v \\ \lambda v.(s_1(s_2(\perp)))v \\ \quad\vdots \\ \lambda v.(s_1(s_2(\cdots(s_n(\perp))\cdots)v \\ \quad\vdots \end{array}$$

So, probably, the infinite program above will be translated as:

$$\bigcup_{n=0}^{\infty} \lambda v.(s_1(s_2(\cdots(s_n(\perp))\cdots)v \quad .$$

We will formalize this idea in §3.

§2. Characterization of C-functions

As in Chapter 3, let $\mathbb{C} = \{c \mid c \in \Delta \to V \cup \{\omega\}\}$. The C-function is a map: $\Lambda \to \mathbb{C}$. It is easy to see that the range of C is only a proper subset of $\mathbb{C}$, i.e., $C(\Lambda) \subsetneq \mathbb{C}$, but what sort of subset is $C(\Lambda)$?

In fact, $\mathbb{C}$ is of too arbitrary structure to attract any interest. The following conditions characterize the hierarchy of some interesting subclasses of $\mathbb{C}$.

Given $c \in \mathbb{C}$.

<u>Condition 1</u>: If $c(\delta) = z \in V$, then either z is free or $z = t_{\delta'}$ for $\delta' \in \Delta$ where $\delta' \leq \delta$ or $\delta' = \delta \circ m$ for some $m \in \mathbb{N}$ (i.e. if a variable is bound, it must be so in an outer context).

<u>Condition 2</u>: If $c(\delta) = \omega$ for some $\delta \in \Delta$, $c(\delta') = \omega$ for any $\delta' \in \Delta$ with $\delta < \delta'$ (i.e. once a subexpression turns out to be bottom, any of its decendants must be bottom, too).

<u>Condition 3</u>: If $c(\delta) \neq \omega$, there exists an integer k_δ^c and a positive integer N_δ^c such that, for all $n > N_\delta^c$, $c(\delta \circ n) = t_{\delta \circ (n+k_\delta^c)}$ and $c(\delta \circ n \circ \delta') = t_{\delta \circ n \circ \delta'}$ for all $\delta' \in \Delta$ (i.e. c is 'finitely wide').

<u>Condition 4</u>: Let $Fr(c) = \{z \mid z \in F,\ c(\delta) = z \text{ for some } \delta \in \Delta\}$. Then $\#(Fr(c)) < \infty$ (i.e. the number of the distinct variables which occur in $\{c(\delta) \mid \delta \in \Delta\}$ is finite).

<u>Condition 5</u>: There are partially computable functions $\phi_c\colon \Delta \to \mathbb{N}$ and $\psi_c\colon \Delta \to V$ such that $\phi_c(\delta) = N_\delta^c$ and $\psi_c(\delta) = z$ if $c(\delta) = z \in V$, $\phi_c(\delta)$ and $\psi_c(\delta)$ are undefined if $c(\delta) = \omega$ (i.e. $\{c(\delta) \mid c(\delta) \neq \omega, \delta \in \Delta\}$ is a recursively enumerable object and the width in Condition 3 is also partially computable).

6.2.1 Theorem. Each element of $C(\Lambda)$ satisfies Conditions 1-5.

Proof. Conditions 1 and 2 are obviously satisfied by the definition of C. Let x be one of the λ-expression such that $C(x) = c$.

Condition 3: Since $C(x,\delta) \neq \omega$, $L(x,\delta) \neq \Omega$. Let

$$L(x,\delta) \xrightarrow{\alpha\beta} \lambda t_{\delta\circ 1} t_{\delta\circ 2} \cdots t_{\delta\circ p} . z X_1 X_2 \cdots X_q \quad .$$

Now set $N_\delta^c = q$ and $k_\delta^c = p - q$.

Condition 4: Since x is finite, x can contain at most finite number of distinct free variables.

Condition 5: Obvious from the definition of C. □

The converse of Theorem 6.2.1 is true as demonstrated in Theorem 6.2.2.

$\mathbb{C}_{fin}$ and $\mathbb{C}_{inf}$ are subclasses of $\mathbb{C}$ defined as follows:

$$\mathbb{C}_{fin} = \{c \mid c \in \mathbb{C}, \text{ c satisfies Conditions 1-5}\}$$
$$\mathbb{C}_{inf} = \{c \mid c \in \mathbb{C}, \text{ c satisfies Conditions 1-3}\} \quad .$$

We have the sequence: $\mathbb{C}_{fin} \subsetneq \mathbb{C}_{inf} \subsetneq \mathbb{C}$. The smallest class $\mathbb{C}_{fin}$ is, in fact, the same as $C(\Lambda)$ as proved in the following theorem.

6.2.2 Theorem. Let c be in $\mathbb{C}$. If c satisfies Conditions 1-5, then there exists a λ-expression x such that $C(x) = c$.

Proof. We give effective codings of $\mathbb{Z}$, Δ and V into Λ as:

$$\text{En:}\quad \begin{array}{l} n \in \mathbb{Z} \mapsto \bar{n} \in \Lambda \\ \delta \in \Delta \mapsto \tilde{\delta} \in \Lambda \\ t_\delta \in T_\Delta \mapsto \hat{t}_\delta \in \Lambda \\ f_i \in F \mapsto \hat{f}_i = f_i \in \Lambda \end{array} \quad .$$

We assume that $\mathrm{En}(\mathbb{Z})$, $\mathrm{En}(\Delta)$, $\mathrm{En}(T_\Delta)$ and $\mathrm{En}(F) = F$ are mutually disjoint.

Given $c \in \mathbb{C}_{fin}$, let Δ_c be the subset of Δ consisting of all δ such that $\phi_c(\delta)$ is defined. Obviously, Δ_c is recursively enumerable.

In the rest of the proof, we depend on the following fact due to Kleene:

"For each partial recursive function $\Phi: \mathbb{N} \to \mathbb{N}$, there exists a λ-expression $\tilde{\Phi} \in \Lambda$ such that

$\tilde{\Phi}\bar{n} \xrightarrow{\beta} \bar{m}$ if $\Phi(n) = m$.

$\tilde{\Phi}\bar{n}$ has no head normal form if $\Phi(n)$ is undefined.

where $\bar{m}$, $\bar{n}$ are the encodings of $m, n \in \mathbb{N}$ in Λ."

(For the proof of the proposition above, see, for example, [2].)

We define $\pi_c \in \Lambda$ by:

$$\pi_c\tilde{\delta} \xrightarrow{\text{CNV}} \begin{cases} \text{a } \lambda\text{-expression without a head normal form} \\ \qquad \text{if } \delta \notin \Delta_c \\ \lambda x.x \quad \text{if } \delta \in \Delta_c \end{cases}$$

A partially computable function $M_c: \Delta \to \mathbb{N}$ is defined by:

$$M_c(\delta) = \begin{cases} \text{undefined} & \text{if } c(\delta) = \omega \\ k_\delta^c + \phi_c(\delta) & \text{if } c(\delta) \neq \omega \ . \end{cases}$$

$P \in \Lambda$ is defined by:

$$P\bar{i}\tilde{\delta} \xrightarrow{\text{CNV}} \widetilde{\delta \circ i} \quad \text{for} \quad i \in \mathbb{N} \quad \text{and} \quad \delta \in \Delta \quad .$$

Finally we define $\Theta_c \in \Lambda$ by the following recursive equation:

$$\Theta_c \tilde{\delta} e \xrightarrow{\text{CNV}} \pi_c \tilde{\delta}(f\tilde{\delta}\bar{0}\overline{M_c(\delta)}\,\overline{\phi_c(\delta)}\,\widehat{c(\delta)}e)$$

where $f \in \Lambda$ is defined by:

$$f\tilde{\delta}\bar{i}\bar{m}\bar{n}\hat{z}e \xrightarrow{\text{CNV}} \begin{cases} g\tilde{\delta}\bar{0}\bar{n}\hat{z}e & \text{if} \quad i = m \\ \lambda s.f\tilde{\delta}\overline{i+1}\bar{m}\bar{n}\hat{z}(N\tilde{\delta}\overline{i+1}e) & \text{otherwise} \end{cases} \qquad (*)$$

where $g \in \Lambda$ is given by

$$g\tilde{\delta}\bar{j}\bar{n}\hat{z}e \xrightarrow{\text{CNV}} \begin{cases} e\hat{z} & \text{if} \quad j = n \\ g\tilde{\delta}\overline{j+1}\bar{n}\hat{z}e(\Theta_c(P\overline{n-j}\tilde{\delta})e) & \text{otherwise} \end{cases}$$

and $N \in \Lambda$ is given by:

$$N\tilde{\delta}\bar{i}e\hat{z} \xrightarrow{\text{CNV}} \begin{cases} s & \text{if} \quad z = t_{\delta \circ i} \\ e\hat{z} & \text{otherwise .} \end{cases} \qquad (**)$$

Note that s at $(**)$ is the same as the bound variable at $(*)$.

Now we assert that $C(\Theta_c\tilde{0}I) = c$. To prove this, we show that there exists $e_\delta \in \Lambda$ for each $\delta \in \Delta_c$ such that:

(1) $\delta = 0$:

(a) If $c(0) = \omega$, then $\Theta_c\tilde{0}I$ is non-head normal.

(b) If $c(0) = v \in V$, $\phi_c(0) = q$ and $q + k_\delta^c = p$,

$$\Theta_c\tilde{0}I \xrightarrow{\text{CNV}} \lambda s_1 s_2 \cdots s_p . v(\Theta_c\tilde{1}e_0)(\Theta_c\tilde{2}e_0)\cdots(\Theta_c\tilde{q}e_0) \ .$$

(2) $\delta = \delta' \circ d \neq 0$

(a) If $c(\delta) = \omega$, then $\Theta_c \tilde{\delta} e_{\delta'}$ is non-head normal.

(b) If $c(\delta) = v \in V$, $\phi_c(\delta) = q$ and $q + k_\delta^c = p$, then

$$\Theta_c \tilde{\delta} e_{\delta'} \xrightarrow{\text{CNV}} \lambda r_1 r_2 \cdots r_p . v(\Theta_c \widetilde{\delta \circ 1} e_\delta)(\Theta_c \widetilde{\delta \circ 2} e_\delta) \cdots (\Theta_c \widetilde{\delta \circ q} e_\delta) \quad .$$

We only prove (1)-(a) and (b). (2)-(a) and (b) are proven similarly.

(1)-(a): Since $c(0) = \omega$, $0 \notin \Delta_c$ and so, $\pi_c \tilde{0}$ is non-head normal. Thus,

$$\Theta_c \tilde{0} I \xrightarrow{\text{CNV}} \pi_c \tilde{0}(\cdots)\text{: non-head normal}$$

(1)-(b): Since $c(0) \neq \omega$, $\pi_c \tilde{0} \xrightarrow{\text{CNV}} I$

$$\begin{aligned}
\Theta_c \tilde{0} I &\xrightarrow{\text{CNV}} I(\cdots) \\
&\xrightarrow{\text{CNV}} f\tilde{0}\bar{0}\bar{p}\bar{q}\hat{v}I \\
&\xrightarrow{\text{CNV}} \lambda s_1 . f\tilde{0}\bar{1}\bar{p}\bar{q}\hat{v}I(N\tilde{0}\bar{1}I) \\
&\quad \vdots \\
&\xrightarrow{\text{CNV}} \lambda s_1 s_2 \cdots s_p . f\tilde{0}\bar{p}\bar{p}\bar{q}\hat{v}(N\tilde{0}\bar{p}(N\tilde{0}\overline{p-1}(\cdots(N\tilde{0}\bar{1}I))\cdots)
\end{aligned}$$

(Set $e_0 = N\tilde{0}\bar{p}(N\tilde{0}\overline{p-1}(\cdots(N\tilde{0}\bar{1}I))\cdots)$.)

$$\begin{aligned}
&\xrightarrow{\text{CNV}} \lambda s_1 s_2 \cdots s_p . g\tilde{0}\bar{0}\bar{q}\hat{v}e_0 \\
&\xrightarrow{\text{CNV}} \lambda s_1 s_2 \cdots s_p . g\tilde{0}\bar{1}\bar{q}\hat{v}e_0(\Theta_c(P\bar{q}\tilde{0})e_0) \\
&\xrightarrow{\text{CNV}} \lambda s_1 s_2 \cdots s_p . g\tilde{0}\bar{1}\bar{q}\hat{v}e_0(\Theta_c \widetilde{0 \circ q} e_0) \\
&\quad \vdots \\
&\xrightarrow{\text{CNV}} \lambda s_1 s_2 \cdots s_p . g\tilde{0}\bar{q}\bar{q}\hat{v}e_0(\Theta_c \widetilde{0 \circ 1} e_0)(\Theta_c \widetilde{0 \circ 2} e_0) \cdots (\Theta_c \widetilde{0 \circ q} e_0) \\
&\xrightarrow{\text{CNV}} \lambda s_1 s_2 \cdots s_q . e_0 \hat{v}(\Theta_c \widetilde{0 \circ 1} e_0)(\Theta_c \widetilde{0 \circ 2} e_0) \cdots (\Theta_c \widetilde{0 \circ q} e_0) \\
&\xrightarrow{\text{CNV}} \lambda s_1 s_2 \cdots s_q . v(\Theta_c \tilde{1} e_0)(\Theta_c \tilde{2} e_0) \cdots (\Theta_c \tilde{q} e_0) \quad .
\end{aligned}$$

This completes the proof for (1)-(b). $\square$

Probably $\theta\colon \mathbb{C}_{fin} \to \Lambda$ $(\theta\colon c \mapsto \Theta_c)$ corresponds to the universal Turing machine.

6.2.3 <u>Corollary</u>. $\mathbb{C}_{fin} = C(\Lambda)$. □

Now Theorem 4.3.2 can be stated as:

$$\mathbb{C}_{fin} \approx \Lambda/=_{D_\infty} \quad .$$

In the rest of this chapter, we shall mainly study $\mathbb{C}_{inf}$.

§3. Infinite λ-expressions

In this section, we formalize the idea of infinitely-deep λ-expressions. We utilize the process of generating infinite programs given in §1 to define the infinite λ-expressions Λ^∞.

6.3.1 Definition.

a. $\Lambda_\square$ is the set of the expressions to be defined by:

1) A variable $v \in U$ alone is in $\Lambda_\square$.

2) $\square \in \Lambda_\square$.

3) If ξ, ζ are in $\Lambda_\square$, so is $\xi(\zeta)$.

4) If η is in $\Lambda_\square$ and $v \in U$ is a variable, then $\lambda v.\eta$ is in $\Lambda_\square$.

b. Let $\xi, \zeta \in \Lambda_\square$. We say that ζ is a specification of ξ if either $\xi = \zeta$ or ζ derives from ξ by replacing some occurrences of $\square$ in ξ with elements in $\Lambda_\square$. (We write as ζ spec ξ.)

c. Given $\zeta \in \Lambda_\square$, we define ζ^* in Λ to be the λ-expression which is derived from ζ by replacing each $\square$ in ζ by Ω.

d. Λ^∞, infinite λ-expressions, is the set of all sequences

$$(\zeta_1, \zeta_2, \ldots, \zeta_n, \ldots)$$

where $\zeta_i \in \Lambda_\square$ and ζ_{i+1} spec ζ_i for each $i = 1,2,\ldots,$ that is,

$$\Lambda^\infty = \{\eta \mid \eta = (\zeta_1, \zeta_2, \ldots),\ \zeta_i \in \Lambda_\square \text{ and } \zeta_{i+1} \text{ spec } \zeta_i \text{ for each } i = 1,2,\ldots\} .$$

Since Λ can be regarded as a subset of $\Lambda_\square$ by the obvious injection: $\Lambda \to \Lambda_\square$, we can embed Λ into Λ^∞ as follows: Let $x \in \Lambda$, $\iota: x \mapsto (x,x,x,x,\ldots) \in \Lambda^\infty$ by $\iota: \Lambda \to \Lambda^\infty$, we regard as $\Lambda \subseteq \Lambda^\infty$. We define Λ_c^∞ to be the set of the infinite λ-expressions which do not contain any free variables, i.e.

$$\Lambda_c^\infty = \{\zeta \mid \zeta = (\zeta_1,\zeta_2,\ldots,\zeta_n) \in \Lambda_c^\infty \text{ where } \zeta_i \text{ has no free variables for each } i\} \quad .$$

The restriction of ι to Λ_c, $\iota|_{\Lambda_c}$ gives the inclusion: $\Lambda_c \subset \Lambda_c^\infty$.

Given $\eta = (\zeta_1,\zeta_2,\ldots,\zeta_n,\ldots)$ in Λ^∞, each ζ_i can be looked upon as a program which has some unspecified parts. $\square$'s occurring in ζ_i are the unspecified parts. ζ_{i+1} is obtained by filling $\square$ in ζ_i with another ξ of $\Lambda_\square$. This process eventually leads us to the infinite λ-expression η.

As we mapped Λ into D_∞ through $\mathbb{V}$, we are to define a semantic function $\mathbb{V}_\infty$ of Λ^∞ into D_∞.

6.3.2 <u>Definition</u> (Semantics $\mathbb{V}_\infty$ of Λ^∞). U is the set of all variables of discourse and Env is the set of all functions: $U \to D_\infty$.

Now, $\mathbb{V}_\infty$: EN $\to (\Lambda^\infty \to D_\infty)$ is the following map: Given $\eta = (\zeta_1,\zeta_2,\ldots) \in \Lambda^\infty$ and $\rho \in$ EN.

$$\mathbb{V}_\infty [\![\eta]\!] \rho = \bigcup_{i=1}^{\infty} \mathbb{V} [\![\zeta_i^*]\!] \rho$$

We should note that, in Definition 6.3.2,

$$\mathbb{V} [\![\zeta_i^*]\!] \rho \subseteq \mathbb{V} [\![\zeta_{i+1}^*]\!] \rho \quad \text{for each } i.$$

So, $\mathbb{V}_\infty [\![\eta]\!] \rho$ is the least upper bound of a directed sequence in D_∞.

6.3.3 <u>Corollary</u>. Let x be in Λ. Then

$$\mathbb{V} [\![x]\!] \rho = \mathbb{V}_\infty [\![\iota(x)]\!] \rho \quad . \quad \square$$

6.3.4 <u>Definition</u>. Given ξ and ζ in Λ^∞, we define the <u>application</u> of ξ to ζ, $\xi(\zeta) \in \Lambda^\infty$, as follows: Let

$$\xi = (\xi_1, \xi_2, \xi_3, \ldots)$$
$$\zeta = (\zeta_1, \zeta_2, \zeta_3, \ldots) \quad .$$

Then

$$\xi(\zeta) = (\xi_1(\zeta_1), \xi_2(\zeta_2), \xi_3(\zeta_3), \ldots) \quad .$$

It is easy to verify that

$$\mathbb{V}_\infty [\![\xi(\zeta)]\!] \rho = \mathbb{V}_\infty [\![\xi]\!] \rho (\mathbb{V}_\infty [\![\zeta]\!] \rho)$$

for ξ, ζ in Λ^∞.

Now, we are ready to consider the correspondence between Λ^∞ and $\mathbb{C}_{inf}$. In fact, the similar relation holds between Λ^∞ and $\mathbb{C}_{inf}$ to that between Λ and $\mathbb{C}_{fin}$.

The following lemma is necessary to prove part 3 of Theorem 6.3.6.

6.3.5 <u>Lemma</u>.

1. Let x and y be λ-expressions which satisfy the condition of Theorem 3.4.7-1. Then, by the theorem, given any $a, b \in D_\infty$, we can choose $e_1, e_2, \ldots, e_n \in \Lambda$ and an environment

ρ for which

$$\mathbb{V}[\![xe_1e_2\cdots e_n]\!]\rho = a$$

and

$$\mathbb{V}[\![ye_1e_2\cdots e_n]\!]\rho = b \quad .$$

Here if, for x_1 and $y_1 \in \Lambda$, $x \underset{D_\infty}{\subseteq} x_1$ and $y \underset{D_\infty}{\subseteq} y_1$, then

$$\mathbb{V}[\![x_1e_1e_2\cdots e_n]\!]\rho = a$$

and

$$\mathbb{V}[\![y_1e_1e_2\cdots e_n]\!]\rho = b \quad .$$

2. Let x and y be in Λ. Assume that, for $\delta \in \Delta$, x and y satisfy the condition of Theorem 3.4.7-2. Then, by the theorem, given any $a, b \in D_\infty$, we can choose $e_1, e_2, \ldots, e_n \in \Lambda$ and an environment ρ for which

$$\mathbb{V}[\![xe_1e_2\cdots e_n]\!]\rho = a$$

and

$$\mathbb{V}[\![ye_1e_2\cdots e_n]\!]\rho = \bot \quad .$$

Here, if, for $x_1, y_1 \in \Lambda$, $x \underset{D_\infty}{\subseteq} x_1$, $y \underset{D_\infty}{\subseteq} y_1$ and $C(y_1, \delta) = \omega$, then

$$\mathbb{V}[\![x_1e_1e_2\cdots e_n]\!]\rho = a$$

and

$$\mathbb{V}[\![y_1e_1e_2\cdots e_n]\!]\rho = \bot \quad .$$

<u>Proof</u>. We prove only part 1 of the lemma. By the assumption of the lemma, there exists $\delta \in \Delta$ which satisfies the following:

(*) For any δ' with $\delta' < \delta$, $\hat{C}(x,\delta') = \hat{C}(y,\delta')$ and

$$\hat{C}(x,\delta) = (u,i)$$

and

$$\hat{C}(y,\delta) = (v,j)$$

where $(u,i) \neq (v,j)$.

Since $\hat{C}(x) \leq \hat{C}(x_1)$ and $\hat{C}(y) \leq \hat{C}(y_1)$,

$$\hat{C}(x,\delta') = \hat{C}(x_1,\delta') \neq \omega$$

and

$$\hat{C}(y,\delta') = \hat{C}(y_1,\delta') \neq \omega$$

for each δ' with $\delta' \leq \delta$.

So, (*) still holds if we replace x by x_1 and y by y_1.

As is seen in the proof of Lemma 3.4.5, the choice of $e_1, e_2, \ldots, e_n$ and ρ depends upon only $C(x,\delta')$ and $C(y,\delta')$ for $\delta' \leq \delta$, from which our assertion follows immediately. □

6.3.6 Definition. $C_\infty\colon \Lambda^\infty \to \mathbb{C}$ is the following map: Given $\zeta = (\zeta_1, \zeta_2, \ldots) \in \Lambda^\infty$,

$$C_\infty(\zeta,\delta) = \begin{cases} \omega & \text{if } C(\zeta_i^*,\delta) = \omega \text{ for all } i = 1,2,\ldots \\ z \in V & \text{if } C(\zeta_i^*,\delta) = z \text{ for some } i. \end{cases}$$

C_∞ is well defined since $\zeta_i^* \underset{D_\infty}{\subseteq} \zeta_{i+1}^*$ and so $C(\zeta_i^*) \leq C(\zeta_{i+1}^*)$.

6.3.7 Theorem.

1. $C_\infty\big|_\Lambda = C$
2. $C_\infty(\Lambda^\infty) = \mathbb{C}_{\mathrm{inf}}$

3. For all $\xi, \zeta \in \Lambda^{\infty}$,

$$\xi \underset{D_\infty}{\subseteq} \zeta \text{ iff } C_\infty(\xi) \leq C_\infty(\zeta)$$

(where $\xi \underset{D_\infty}{\subseteq} \zeta$ means $\mathbb{V}_\infty[\![\xi]\!]\rho \subseteq \mathbb{V}_\infty[\![\zeta]\!]\rho$ for all $\rho \in EN$ and $\leq$ is the partial order over $\mathbb{C}$ as in Chapter 3).

So, $\mathbb{C}_{inf} \approx \Lambda^{\infty}/\underset{D_\infty}{=}$.

Proof. It is obvious that $C_\infty|_\Lambda = C$, and $C_\infty(\Lambda^\infty) \subseteq \mathbb{C}_{inf}$.

To prove that C_∞ is surjective, we need a similar concept to admissible Δ-trees in Chapter 4.

Let $c \in \mathbb{C}_{inf}$. We say that a Δ-tree, T, is admissible to c if T satisfies the following:

> If $\delta \in T$, then $\delta\circ 1, \delta\circ 2, \ldots, \delta\circ N_\delta^c$ are also in T where N_δ^c is as in Condition 3 of $\mathbb{C}_{inf}$.

Now we define $\alpha^n(c,T) \in \Lambda_\square$ for $c \in \mathbb{C}_{inf}$, a Δ-tree, T, admissible to c, and $n \in \mathbb{N}$ (similar to $A_p^n(x,T)$ in Chapter 4).

$\alpha^n(c,T) = \alpha_c^0$ where α_c^δ $(\delta \in T)$ is defined by:

1) If $|\delta| = n$, $\alpha_c^\delta = \square$.

2) If $|\delta| < n$ and $c(\delta) = \omega$, $\alpha_c^\delta = \Omega$.

3) If $|\delta| < n$ and $c(\delta) = z$,

$$\alpha_c^\delta = \lambda t_{\delta\circ 1} t_{\delta\circ 2} \cdots t_{\delta\circ(\gamma_T(\delta)+k_\delta^c)} . z\alpha_c^{\delta\circ 1}\alpha_c^{\delta\circ 2}\cdots\alpha_c^{\delta\circ\gamma_T(\delta)}$$

where k_δ^c is as in Condition 3 of $\mathbb{C}_{inf}$.

Now it is easy to see that

$$\alpha^n(c,T) \in \Lambda_\square \quad \text{and} \quad \alpha^{n+1}(c,T) \ \underline{\text{spec}}\ \alpha^n(c,T)$$

for $n = 0,1,2,\ldots,$ and, letting $\alpha = (\alpha^0(c,T),\alpha^1(c,T),\alpha^2(c,T),\ldots)$, $\alpha \in \Lambda^\infty$ and $C_\infty(\alpha) = c$.

To prove the last part of the theorem, let us assume that $C_\infty(\xi) \not\leq C_\infty(\zeta)$ for given $\xi = (\xi_1,\xi_2,\ldots)$ and $\zeta = (\zeta_1,\zeta_2,\ldots)$ in Λ^∞. By the definition of C_∞, there exist $i, j \in \mathbb{N}$ and $\delta \in \Delta$ for which

$$\begin{aligned} &\text{either} \quad C(\xi_i^*,\delta) = u \quad \text{and} \quad C(\zeta_j^*,\delta) = v \quad \text{with} \quad u \neq v \\ &\qquad \text{or} \quad C(\xi_i^*,\delta) \neq \omega \quad \text{and} \quad C(\zeta_k^*,\delta) = \omega \quad \text{for all} \quad k \in \mathbb{N}. \end{aligned}$$

Since $\xi_k^* \underset{D_\infty}{\subseteq} \xi_{k+1}^*$ and $\zeta_k^* \underset{D_\infty}{\subseteq} \zeta_{k+1}^*$ for each k, by Lemma 6.3.5, there exist $e_1,e_2,\ldots,e_n \in \Lambda$ and an environment ρ for which there exists $K > 0$ such that for all $m > K$,

$$\begin{aligned} &\mathbb{V}[\![\xi_m^* e_1 e_2 \cdots e_n]\!]\rho = K \\ &\mathbb{V}[\![\zeta_m^* e_1 e_2 \cdots e_n]\!]\rho = H \end{aligned}$$

or

$$\begin{aligned} &\mathbb{V}[\![\xi_m^* e_1 e_2 \cdots e_n]\!]\rho = K \\ &\mathbb{V}[\![\zeta_m^* e_1 e_2 \cdots e_n]\!]\rho = \bot \quad . \end{aligned}$$

Since $\mathbb{V}_\infty[\![\xi]\!]\rho = \bigcup_{k=1}^{\infty} \mathbb{V}[\![\xi_k^*]\!]\rho$ and $\mathbb{V}_\infty[\![\zeta]\!]\rho = \bigcup_{k=1}^{\infty} \mathbb{V}[\![\zeta_k^*]\!]\rho$, by the definition of $\mathbb{V}_\infty$, we conclude that $\xi \underset{D_\infty}{\not\subseteq} \zeta$.

On the other hand, let us assume that $C_\infty(\xi) \leq C_\infty(\zeta)$. We take a Δ-tree, T, which is admissible to both $C_\infty(\xi)$ and $C_\infty(\zeta)$.

We define $\beta(\eta,i)$ for $\eta = \xi$ or ζ and $i > 0$ as follows: $\beta(\eta,i) = \beta^0(\eta,i)$ where $\beta^\delta(\eta,i)$ $(\delta \in T)$ is defined by:

1) If $|\delta| = i$, then $\beta^\delta(\eta,i) = \Omega$.

2) If $|\delta| < i$ and $C(\eta_i^*,\delta) = \omega$, then $\beta^\delta(\eta,i) = \Omega$.

3) If $|\delta| < i$ and $\hat{C}(\eta_i^*,\delta) = (z,k)$, then

$$\beta^\delta(\eta,i) = \lambda t_{\delta\circ 1} t_{\delta\circ 2} \cdots t_{\delta\circ(\gamma_T(\delta)+k)} \cdot z\beta^{\delta\circ 1}(\eta,i)\beta^{\delta\circ 2}(\eta,i)\cdots\beta^{\delta\circ\gamma_T(\delta)}(\eta,i) \quad .$$

In the first place,

$$\beta(\xi,i) \underset{D_\infty}{\subseteq} \xi_i^* \quad \text{and} \quad \beta(\zeta,i) \underset{D_\infty}{\subseteq} \zeta_i^* \quad .$$

In the second place, since

$$\xi_i^* = \bigcup_{n=0}^{\infty} A_p^n(\xi_i^*,T)$$

and

$$\zeta_i^* = \bigcup_{n=0}^{\infty} A_p^n(\xi_i^*,T) \quad ,$$

and

$$A_p^n(\xi_i^*,T) \underset{D_\infty}{\subseteq} \beta(\xi,j)$$

and $\qquad\qquad$ for $j > \max(i,n)$

$$A_p^n(\zeta_i^*,T) \underset{D_\infty}{\subseteq} \beta(\zeta,j)$$

we conclude that

$$(*) \qquad \zeta \underset{D_\infty}{=} \bigcup_{i=1}^{\infty} \beta(\xi,i) \quad \text{and} \quad \xi \underset{D_\infty}{=} \bigcup_{i=1}^{\infty} \beta(\zeta,i) \quad .$$

Since $C_\infty(\xi) \leq C_\infty(\zeta)$, by the definition of C_∞, it follows that, for any $p > 0$, there exists a sufficiently large $Q > 0$ such that

$$C(\beta(\xi,p)) \leq C(\beta(\zeta,Q)) \quad ,$$

that is,

$$(**) \qquad \beta(\xi,p) \underset{D_\infty}{\subseteq} \beta(\zeta,Q) \quad .$$

From (*) and (**), $\xi \underset{D_\infty}{\subseteq} \zeta$ is immediate. □

6.3.8 <u>Definition</u>. Given a complete lattice D, let E be a subset of D.

The <u>directed completion</u> of E in D is the set

$$\{a \mid \text{there is a directed set } F \subseteq E \text{ such that } a = \cup F\} \quad .$$

6.3.9 <u>Theorem</u>. $\{\mathbb{V}_\infty [\![\xi]\!] \mid \xi \in \Lambda_c^\infty\} \subseteq D_\infty$ is the directed completion of $\{\mathbb{V} [\![x]\!] \mid x \in \Lambda_c\} \subseteq D_\infty$. Thus $\{\mathbb{V}_\infty [\![\xi]\!] \mid \xi \in \Lambda_c^\infty\}$ is a directed complete subset of D_∞.

(Note that the elements of Λ_c^∞ and Λ_c do not have any free variables. So, their values in D_∞ do not depend on the environment $\rho \in EN$.)

<u>Proof</u>. The argument for the proof is similar to Theorem 6.3.7.

□

Theorem 6.3.9 shows that the relation between Λ_c^∞ and Λ_c is similar to that between the real numbers and the rational numbers.

Each real number is defined as the limit of a non-decreasing sequence of rational numbers. In this way, we may well regard Λ^∞ as the generalization of Λ.

6.3.10 <u>Corollary</u>. The cardinality of D_∞ is strictly larger than denumerable if $D_0 \neq \{\perp\}$.

<u>Proof</u>. Obviously, $C_\infty(\Lambda_c^\infty)$ $(\subseteq \mathbb{C}_{inf})$ has a cardinality strictly larger than denumerable. Since ξ and ζ in Λ_c^∞ are mapped to different elements in D_∞ if $C_\infty(\xi) \neq C_\infty(\zeta)$, D_∞ must have a cardinality strictly larger than denumerable. $\square$

§4. Lattice Structure of $\mathbb{C}_{fin}$ and $\mathbb{C}_{inf}$

In the previous section, we generalized Λ to Λ^{∞}. Here we shall show that the lattice structure in Λ and Λ^{∞} induced by the D_{∞} partial order is equivalent to the lattice structure of $\mathbb{C}_{inf}$ which is a directed-complete partially ordered set. In addition, we shall examine the structure of $\mathbb{C}_{inf}$ and $\mathbb{C}_{fin}$.

6.4.1 Proposition. $\mathbb{C}_{fin}$ and $\mathbb{C}_{inf}$ are partially ordered sets with the order $\subseteq$ defined by: For $a, b \in \mathbb{C}_{inf}$, $a \subseteq b$ if and only if, for all $\delta \in \Delta$,

$$\begin{array}{ll} \text{either} & a(\delta) = \omega \\ \text{or} & a(\delta) = b(\delta) \quad . \end{array} \qquad \square$$

6.4.2 Proposition. $\mathbb{C}_{fin}$ and $\mathbb{C}_{inf}$ lower semi-lattices. More precisely, we define $a \cap b \in \mathbb{C}_{inf}$ $(\mathbb{C}_{fin})$, $a, b \in \mathbb{C}_{inf}$ $(\mathbb{C}_{fin})$ as follows: Let $m_{\delta} = \max(N_{\delta}^{a}, N_{\delta}^{b}) + 1$ where N_{δ}^{a} and N_{δ}^{b} are as in Condition 3 of $\mathbb{C}_{inf}$. Define $c = a \cap b \in \mathbb{C}_{inf}$ $(\mathbb{C}_{fin})$ by:

1) $$c(0) = \begin{cases} a(0) & \text{if } a(0) = b(0) \neq \omega \text{ and } a(m_0) = b(m_0) \\ \omega & \text{otherwise.} \end{cases}$$

2) Let $\delta = \delta' \circ n$ and suppose that $c(\delta')$ is already defined.

a) If $c(\delta') = \omega$ then $c(\delta) = \omega$.

b) If $c(\delta') \neq \omega$ then

$$c(\delta) = \begin{cases} a(\delta) & \text{if } a(\delta) = b(\delta) \neq \omega \text{ and } a(\delta \circ m_{\delta}) = b(\delta \circ m_{\delta}) \\ \omega & \text{otherwise.} \end{cases}$$

Proof. If $a, b \in \mathbb{C}_{inf}$, $c \in \mathbb{C}_{inf}$ since, obviously, c as defined above satisfies Conditions 1 to 3 of $\mathbb{C}_{inf}$. In case $a, b \in \mathbb{C}_{fin}$, we only have to show that c satisfies Conditions 4 and 5 of $\mathbb{C}_{fin}$ to prove that $c \in \mathbb{C}_{fin}$. Since c cannot contain any free variable that does not appear in a or b, c satisfies Condition 4. On the other hand, $c(\delta)$ for each $\delta \in \Delta$ is computed in the following way: If $\psi_a(\delta)$ and $\psi_b(\delta)$ are both defined and $\psi_a(\delta) = \psi_b(\delta)$ and if $\psi_a(\delta\circ(\max(\phi_a(\delta),\phi_b(\delta))+1)) = \psi_b(\delta\circ(\max(\phi_a(\delta),\phi_b(\delta))+1))$ then $c(\delta) = \psi_a(\delta)$. Otherwise $c(\delta) = \omega$.

This statement guarantees that there exists a partially computable function $\psi_c\colon \Delta \to V$ that satisfies Condition 5. Also $\phi_c\colon \Delta \to \mathbb{N}$ is defined by:

$$\phi_c(\delta) = \begin{cases} \text{undefined} & \text{if } \psi_c \text{ is undefined} \\ \max(\phi_a(\delta),\phi_b(\delta)) & \text{otherwise.} \end{cases}$$

On the other hand it is easy to see that there is no $d \in \mathbb{C}_{inf}$ $(\mathbb{C}_{fin})$ such that $c \subsetneq d$ and both $d \subseteq a$ and $d \subseteq b$. $\square$

6.4.3 Corollary. Given $x, y \in \Lambda$, there exists a λ-expression $z \in \Lambda$ such that

$$C(z) = C(x) \cap C(y) \quad .$$

Proof. Immediate from Proposition 6.4.2 and Theorems 6.2.1 and 6.2.2. $\square$

Given $x, y, z \in \Lambda$, if $C(z) = C(x) \cap C(y)$, $z \underset{D_\infty}{\subseteq} x \cap y$ by Theorem 4.3.2.

However it is not generally the case that $z \underset{D_\infty}{=} x \cap y$.

6.4.4 Counterexample. Let

$$X = \lambda xyz.x\Omega z$$
$$Y = \lambda xyz.xy\Omega$$
$$Z = \lambda xyz.x\Omega\Omega \quad .$$

Obviously, $C(Z) = C(X) \cap C(Y)$, but when D_∞ is continuous,*
$(u \cap v)(w) = u(w) \cap v(w)$ for $u, v, w \in D_\infty$, and so,

$$Z(\lambda ab.a \cup \lambda ab.b)II \xrightarrow{\beta} \Omega$$
$$X(\lambda ab.a \cup \lambda ab.b)II \xrightarrow{\beta} I$$
$$Y(\lambda ab.a \cup \lambda ab.b)II \xrightarrow{\beta} I \quad .$$

So

$$Z \underset{D_\infty}{\subsetneq} X \cap Y \quad .$$

6.4.5 Proposition. Any directed subset of $\mathbb{C}_{inf}$ has its least upper bound in $\mathbb{C}_{inf}$. So $\mathbb{C}_{inf}$ is directed-complete.

Proof. Let $\mathcal{D}$ be any directed set of $\mathbb{C}_{inf}$. $d \in \mathbb{C}_{inf}$ as defined below gives the least upper bound of $\mathcal{D}$:

$$d(\delta) = \begin{cases} \omega & \text{if } c(\delta) = \omega \text{ for all } c \in \mathcal{D} \\ z & \text{if } c(\delta) = z \text{ for some } c \in \mathcal{D} \, . \end{cases}$$

d is well defined by Theorem 6.3.7-3 since $\mathcal{D}$ is directed. It is easy to see that d satisfies Conditions 1-3 of $\mathbb{C}_{inf}$. □

The following proposition shows that the lattice topology of Λ and Λ^∞ induced by D_∞ partial order is, in fact, equivalent to the lattice topology of $\mathbb{C}_{inf}$.

---

*Refer to [15] or [12] for the details.

6.4.6 Theorem. For $\xi \in \Lambda^{\infty}(\Lambda)$ and a directed set $\mathcal{D} \subset \Lambda^{\infty}(\Lambda)$, $\xi \underset{D_\infty}{=} \cup\mathcal{D}$ if and only if $C(\xi) = \cup\{C(\zeta) \mid \zeta \in \mathcal{D}\}$ in $\mathbb{C}_{inf}$ $(\mathbb{C}_{fin})$.

Proof. For the case of $\mathcal{D} \subset \Lambda$, we proved in Theorem 4.4.4. Since the D_∞-value of each element of Λ^{∞} is defined as the limit of a directed sequence of members of Λ, this result is extended to the case of $\mathcal{D} \subset \Lambda^{\infty}$ in a straightforward manner. □

It would be interesting to ask if we could remove the condition of directedness from Theorems 4.4.4 and 6.4.6. The answer is 'no' by the following argument.

6.4.7 Definition. 1) For $a, b \in \mathbb{C}_{inf}$, we say a and b are compatible if there is no $\delta \in \Delta$ such that

$$a(\delta) \neq \omega$$

$$b(\delta) \neq \omega$$

and $$a(\delta) \neq b(\delta)\ .$$

2) For $S \subseteq \mathbb{C}_{inf}$, S is said to be compatible if any two elements of S are compatible.

6.4.8 Corollary. For $\mathcal{D} \subset \mathbb{C}_{inf}$, if $\mathcal{D}$ is directed then $\mathcal{D}$ is compatible. □

By Theorem 4.3.2 and Theorem 6.3.7-3, if $C(\xi) = \cup\{C(\zeta) \mid \zeta \in S\}$ for $\xi \in \Lambda^{\infty}$ and $S \subseteq \Lambda^{\infty}$ such that $\{C(\zeta) \mid \zeta \in S\} \subseteq \mathbb{C}_{inf}$ is compatible, then $\cup S \underset{D_\infty}{\subseteq} \xi$. However it is not always the case that $\cup S \underset{D_\infty}{=} z$.

6.4.9 Counterexample. Let

$$X = \lambda zxy.z\Omega y$$

$$Y = \lambda zxy.zx\Omega$$

$$Z = \lambda zxy.zxy \quad .$$

Obviously, X and Y are compatible and $C(Z) = C(X) \cup C(Y)$, but

$$Z(\lambda ab.ba)II \xrightarrow{\beta} I$$

$$X(\lambda ab.ba)II \xrightarrow{\beta} \Omega$$

$$Y(\lambda ab.ba)II \xrightarrow{\beta} \Omega$$

so

$$X \cup Y \underset{D_\infty}{\subsetneq} Z \quad .$$

We note that $\mathbb{C}_{fin}$ and $\mathbb{C}_{inf}$ have the least element $\bar{\Omega}$ which satisfies

$$\bar{\Omega}(\delta) = \omega \text{ for all } \delta \in \Delta \quad .$$

6.4.10 Proposition. For all elements $a \in \mathbb{C}_{inf}$ $(\mathbb{C}_{fin})$ except $\bar{\Omega}$, there exists an interesting directed set $\mathcal{D} \subseteq \mathbb{C}_{inf}$ $(\mathbb{C}_{fin})$ whose least upper bound is a, i.e. $x \subsetneq a$ for all $x \in \mathcal{D}$ and $a = \cup\mathcal{D}$.

Proof. Similar to Theorem 4.4.6. $\square$

6.4.11 Proposition. Given $a, b \in \mathbb{C}_{inf}$ $(\mathbb{C}_{fin})$ such that $a \subsetneq b$, then there exists $c \in \mathbb{C}_{inf}$ $(\mathbb{C}_{fin})$ such that

$$a \subsetneq c \subsetneq b \quad .$$

Proof. Similar to Theorem 4.4.7. □

Here we state the negative result that $\mathbb{C}_{inf}$ is not continuous.* The following example shows that $\mathbb{C}_{inf}$ is not continuous.

Let

$$a_n = C(\lambda xyz.z(F^n(\Omega)y)x$$
$$b = C(\lambda xyz.zy\Omega)$$
$$c = C(\lambda xyz.zyx)$$

where $F = \lambda fxy.x(fy)$. Since $I_{D_\infty} = J_{D_\infty} = \bigcup_{n=0}^{\infty} F^n(\Omega)$,

$$c = \bigcup_{n=0}^{\infty} a_n .$$

Also, $b \subseteq c$, but it is not the case that $b \subseteq a_n$ for any $n \geqq 1$. (The topological order, $\prec$, on $\mathbb{C}_{inf}$ is trivial in a sense that $a \prec b$ for $a, b \in \mathbb{C}_{inf}$ if and only if $a = \perp$.)

6.4.12 Proposition. Given $a, b \in \mathbb{C}_{inf}$ ($\mathbb{C}_{fin}$), we define $a(b)$, application of a to b, as follows:

$$a(b) = C_\infty(\xi(\zeta))$$

where $\xi \in C_\infty^{-1}(a)$ and $\zeta \in C_\infty^{-1}(b)$. Then $\Phi_a: \mathbb{C}_{inf} \to \mathbb{C}_{inf}$ (: $\mathbb{C}_{fin} \to \mathbb{C}_{fin}$) for each $a \in \mathbb{C}_{inf}$ ($\mathbb{C}_{fin}$), defined by $\Phi_a(b) = a(b)$,

*This fact was suggested with the example to the author by Christopher Wadsworth.

is continuous and $\mathbb{C}_{inf}$ ($\mathbb{C}_{fin}$) is extensional with respect to the application, i.e. given any $a, b \in \mathbb{C}_{inf}$ ($\mathbb{C}_{fin}$) if $a(c) = b(c)$ for all $c \in \mathbb{C}_{inf}$ ($\mathbb{C}_{fin}$) then $a = b$.

Proof. The continuity of Φ_a is immediate from the definition and the fact that the application of λ-expressions is continuous in D_∞. The extensionality can be shown in a similar way to Theorem 4.4.8. □

We now note that $\mathbb{C}_{inf}$ can become a complete lattice by adding T (top) to $\mathbb{C}_{inf}$. Namely, we define $a \cup b \in \{T\} \cup \mathbb{C}_{inf}$ for $a, b \in \{T\} \cup \mathbb{C}_{inf}$ as follows:

1) If $a = T$, $b = T$ or a and b are not compatible then $a \cup b = T$.

2) If a and b are compatible, determine $c = a \cup b$ by:

$$c(\delta) = \begin{cases} a(\delta) & \text{if } b(\delta) = \omega \\ b(\delta) & \text{if } a(\delta) = \omega \\ \omega & \text{if } a(\delta) = b(\delta) = \omega \end{cases} .$$

6.4.13 Corollary. $\{T\} \cup \mathbb{C}_{inf}$ is a complete lattice. □

However, Counterexample 6.4.9 shows that $\cup S$ reflects the reality only if $S \subseteq \mathbb{C}_{inf}$ is directed. Also, the definition of $\cup$ above is artificially too strong. For example, take $\lambda x.x$ and $\lambda x.xx$. Since they are not compatible, by the definition above $\lambda x.x \cup \lambda x.xx = T$, but since

$$\begin{aligned}(\mathbb{V}[\![\lambda x.xx]\!] \cup \mathbb{V}[\![\lambda x.x]\!])(\mathbb{V}[\![\lambda x.xx]\!]) &= \mathbb{V}[\![\lambda x.xx]\!] \cup \mathbb{V}[\![(\lambda x.xx)(\lambda x.xx)]\!] \\ &= \mathbb{V}[\![\lambda x.xx]\!] \neq T .\end{aligned}$$

So $\mathbb{W}[\![\lambda x.xx]\!] \cup \mathbb{W}[\![\lambda x.x]\!] \neq T$. However, can any interesting theory be built if we allow $\cup$ of mutual elements such as $\lambda x.xx$ and $\lambda x.x$ not be T? I leave this question open here.

CHAPTER 7

SUMMARY, CONCLUSIONS, PROSPECTS

Summarizing the results obtained in the previous chapters, we show that only some abstract properties of D_∞ are needed to deduce these results. This fact makes it possible to give an axiomatization of the extensional model theory of λ-calculus. Some prospects for future research are given.

§1. Summary

We start this chapter with the summary of the results in Chapters 3, 4 and 6. The following diagram is illustrative:

7.1.1 Diagram.

$$\begin{tikzcd}
\Lambda \arrow[dd, "C"'] \arrow[dr, "\mathbb{V}"] & \subseteq & \Lambda^{\infty} \arrow[dl, "\mathbb{V}_{\infty}"'] \arrow[dd, "C_{\infty}"] \\
 & {[EN \to D_{\infty}]} & \\
\mathbb{C}_{fin} \arrow[ur, "d"'] & \subseteq & \mathbb{C}_{inf} \arrow[ul, "d_{\infty}"]
\end{tikzcd}$$

where d_∞ and d are defined as follows:

For $c \in \mathbb{C}_{inf}$, $\rho \in EN$,

$d_\infty [\![c]\!] \rho = V_\infty [\![\xi]\!] \rho$ for $\xi \in \Lambda^\infty$ such that $c = C_\infty(\xi)$.

For $c \in \mathbb{C}_{fin}$, $\rho \in EN$,

$d [\![c]\!] \rho = V [\![z]\!] \rho$ for $z \in \Lambda$ such that $C(z) = c$.

These definitions of d and d_∞ are valid, since, by Theorems 4.3.2 and 6.3.7-3,

$$\xi \underset{D_\infty}{=} \eta \text{ iff } C_\infty(\xi) = C_\infty(\eta) \quad .$$

We list the results we have reached:

1) C and C_∞ are surjective.

2) d and d_∞ are injective.

3) $C = C_\infty|_\Lambda$ and $d = d_\infty|_{\mathbb{C}_{fin}}$.

4) The diagram is commutative, i.e. $\mathbb{V} = d \circ C$ and $\mathbb{V}_\infty = d_\infty \circ C_\infty$.

5) $d_\infty \colon \mathbb{C}_{inf} \to [EN \to D_\infty]$ is a continuous map.

(So d_∞ is monotonic, too.)

Let us prove 5). We regard $[EN \to D_\infty]$ as a lattice by the partial order induced by D_∞, i.e. given $\alpha, \beta \in [EN \to D_\infty]$, $\alpha \subseteq \beta$ if and only if $\alpha(\rho) \subseteq \beta(\rho)$ for all $\rho \in EN$. Given any directed set $\mathcal{D} \subseteq \mathbb{C}_{inf}$, let $\bar{\mathcal{D}} = \{\xi \in \Lambda^\infty \mid C_\infty(\xi) \in \mathcal{D}\}$. Then $\bar{\mathcal{D}}$ is directed in D_∞. Let $\eta = \cup\bar{\mathcal{D}} \in \Lambda^\infty$. By Theorem 6.4.6, $C_\infty(\eta) = \cup\{C_\infty(\xi) \mid \xi \in \bar{\mathcal{D}}\}$. Now, given any $\rho \in EN$,

$$\begin{aligned}
d_\infty [\![\cup\mathcal{D}]\!] \rho &= d_\infty [\![C_\infty(\eta)]\!] \rho \\
&= \mathbb{V}_\infty [\![\eta]\!] \rho && \text{by the definition of } d_\infty \\
&= \mathbb{V}_\infty [\![\cup\bar{\mathcal{D}}]\!] \rho \\
&= \cup\{\mathbb{V}_\infty [\![\xi]\!] \rho \mid \xi \in \bar{\mathcal{D}}\} \\
&= \cup\{d_\infty [\![d]\!] \rho \mid d \in \mathcal{D}\} && \text{by the definition of } d_\infty .
\end{aligned}$$

This proves that d_∞ is continuous. □

Since d_∞ is 1 to 1, continuous, we can say that $\mathbb{C}_{inf}$ give the lattice structure of Λ and Λ^∞ induced by D_∞. Since $\mathbb{C}_{inf}$ can be defined naturally from Λ, independent of D_∞, the lattice structure of $\mathbb{C}_{inf}$ can be said to be the inherent structure of Λ.

§2. Universality of $\mathbb{C}_{inf}$ over Λ

We may ask what properties of D_∞ are essential to have the theory summarized in the previous section. Namely, for what kind of model of the λ-calculus, could we draw Diagram 7.1.1 such that the properties 1) to 5) may hold? This speculation will probably lead us to a more general theory of λ-calculus models. To make the description of this section as self-contained as possible, we start with the definitions of the Δ-trees and their admissibility in Chapter 4.

7.2.1 Definition. An infinite subset T of the pedigree Δ is said to be a Δ-tree if

1) $0 \in \Delta$

2) If $\delta \in T$, then there exists $N \in \mathbb{N}$ such that $\delta\circ 1, \delta\circ 2, \ldots, \delta\circ N \in T$ and $\delta\circ k \notin T$ for all $k > N$.

For a Δ-tree T and $\delta \in T$, define $\gamma_T(\delta)$ to be N in (2), i.e.

$$\gamma_T(\delta) = \#\{\delta' \in T \mid \delta' = \delta\circ m \text{ for some } m \in \mathbb{N}\} \quad .$$

We redefine the notion of admissibility as follows:

7.2.2 Definition. Given a Δ-tree T and $c \in \mathbb{C}_{inf}$, T is said to be admissible to c if, for all $\delta \in T$, $\gamma_T(\delta) \geqq N_\delta^c$ where N_δ^c is as in Condition 3 of $\mathbb{C}_{inf}$.

7.2.3 Definition (Structural Approximation). Given $c \in \mathbb{C}_{inf}$ and a Δ-tree T which is admissible to c, we define $A_p^n(c,T) \in \Lambda_\square$ in the following way: $A_p^n(c,T) = A^0(c,T,n)$ where $A^\delta(c,T,n)$ is defined for each $\delta \in T$ inductively as:

1) If $|\delta| \leq n$,

(i) if $c(\delta) = \omega$ then $A^{\delta}(c,T,n) = \Omega$

(ii) if $c(\delta) = z$, then

$$A^{\delta}(c,T,n) = \lambda t_{\delta\circ 1} t_{\delta\circ 2} \cdots t_{\delta\circ(\gamma_T(\delta)+k^c_\delta)} . zA^{\delta\circ 1}(c,T,n) A^{\delta\circ 2}(c,T,n) \cdots \cdots A^{\delta\circ\gamma_T(\delta)}(c,T,n)$$

where k^c_δ is as in Condition 3 of $\mathbb{C}_{inf}$.

2) If $|\delta| = n$, then $A^{\delta}(c,T,n) = \square$.

Lemma 4.2.2 is rewritten as follows:

7.2.4 Lemma. Given $x \in \Lambda$ and a Δ-tree T, if T is admissible to $C(x)$, then

$$x \underset{D_\infty}{=} \bigcup_{n=1}^{\infty} (A^n_p(C(x),T))^*$$

where $*: \Lambda_\square \to \Lambda$ is as in Definition 6.3.1. $\square$

Obviously $A^{n+1}_p(c,T)$ spec $A^n_p(c,T)$ for $c \in \mathbb{C}_{inf}$ and a Δ-tree T admissible to c. So $\xi_c = (A^1_p(c,T), A^2_p(c,T), \ldots)$ is in Λ^∞. This ξ_c gives a decoding of c in Λ^∞, i.e. $c = C_\infty(\xi_c)$.

7.2.5 Definition. We say that a domain D is a reasonable extensional model for the λ-calculus if D satisfies the following conditions:

Axiom 1. D is a directed-complete partially ordered set with the least element $\perp = \cap D$ and $D \neq \{\perp\}$.

<u>Axiom 2</u>. There is the following pair of maps (ϕ,ψ) that are bijective and continuous

$$D \underset{\psi}{\overset{\phi}{\rightleftarrows}} [D \to D] \quad .$$

<u>Axiom 3</u>. We map Λ into D by the semantic function $\mathbb{V}$ as follows: Let $EN = (U \to D)$ for the set of variables U. $\mathbb{V}: \Lambda \to [EN \to D]$ is defined as:

1) For $v \in U$ and $\rho \in EN$, $\mathbb{V}[\![v]\!]\rho = \rho(v)$.

2) For $x, y \in \Lambda$, $\rho \in EN$, $\mathbb{V}[\![x(y)]\!]\rho = \phi(\mathbb{V}[\![x]\!]\rho)(\mathbb{V}[\![y]\!]\rho)$.

3) For $v \in U$, $x \in \Lambda$ and $\rho \in EN$,

$$\mathbb{V}[\![\lambda v.x]\!]\rho = \psi(\lambda\beta \in D\colon \mathbb{V}[\![x]\!]\rho[v/\beta])$$

where

$$\rho[v/\beta](u) = \begin{cases} \rho(u) & \text{if } u \neq v \\ \beta & \text{if } u = v \,. \end{cases}$$

Then the following two properties hold:

a) For each $x \in \Lambda$, $\mathbb{V}[\![x]\!]\rho = \bot$ for all $\rho \in EN$ if x has no head normal form.

b) For each $x \in \Lambda$ and a Δ-tree T which is admissible to $C(x)$,

$$\mathbb{V}[\![x]\!]\rho = \cup\{\mathbb{V}[\![A_p^n(C(x),T)]\!]\rho \mid n \in \mathbb{N}\}$$

for each $\rho \in EN$.

7.2.6 <u>Theorem</u> (Universality of $\mathbb{C}_{inf}$ over Λ). If D is any reasonable extensional model for Λ, then Diagram 7.1.1 is valid even if we replace D_∞ by D.

Proof. This theorem asserts that all the theory developed in Chapters 4, 5 and 6 depend on only the properties of D in Definition 7.2.5. The proof is done by a careful inspection on what properties of D_∞ are used to prove the validity of the diagram. □

Here we note that Wadsworth's theorem on reduced approximants also holds on D as defined in 7.2.5.

7.2.7 Proposition. Let $A(x)$ be the set of all reduced approximants of $x \in \Lambda$. Then in a reasonable extensional model D,

$$x \underset{D}{=} \cup A(x) \quad .$$

Proof. Let T be a Δ-tree admissible to $C(x)$. We show that given $A_p^n(C(x),T)$ for any $n \in \mathbb{N}$ there is $\varepsilon \in A(x)$ such that

$$A_p^n(C(x),T) \underset{D}{\subseteq} \varepsilon \quad .$$

i) If x has no head normal form, then $x \underset{D}{=} \perp$ and $A(x) = \{\Omega\}$, so $x \underset{D}{=} \cup A(x)$.

ii) If x has a head normal form, let us consider $A_p^n(C(x),T)$. By Corollary 4.1.13, there is $T^n(x)$ such that

$$x \xrightarrow{CNV} T^n(x)$$

and $A_p^n(C(x,T))$ is obtained from $T^n(x)$ by replacing each $L(x,\delta)$ in $T^n(x)$ by Ω for each $\delta \in T$ such that $|\delta| = n$. Let $x \xrightarrow{\beta} x' \xrightarrow{CNV} T^n(x)$ be the sequence of reductions so that

$x' \to T^n(x)$ does not contain any β-reductions. Since α, η, Ω-conversions do not increase β-redexes, this resolution is possible. Then let ε be the direct approximant of x'. Then

$$A_p^n(C(x),T) \underset{D}{\subseteq} \varepsilon$$

because all the β-redexes in x' are in $T^n(x)$ and they are in the parts of $T^n(x)$ which have no corresponding part in $A_p^n(C(x),T)$ except Ω. Since $x \underset{D}{=} \bigcup_{n=0}^{\infty} A_p^n(C(x),T)$, $x \underset{D}{=} \cup A(x)$. □

However Axiom 3b cannot be deduced from Axiom 3a or $x \underset{D}{=} \cup A(x)$ or the combination of both. One may ask if we could reduce Axiom 3b to a simpler condition. This does not seem to be possible if we consider Park's pathological model [10]. Park showed that if a different (ϕ_0,ψ_0) is adopted to construct D_∞, then $Y \underset{D_\infty}{\neq} \lambda f. \bigcup_{n=0}^{\infty} f^n(\perp)$ for the Curry's pathological combinator Y.

As Corollary 4.2.3, Axiom 3b implies $Y \underset{D}{=} \lambda f. \bigcup_{n=0} f^n(\perp)$. This implies that Axiom 3b is not true in Park's model. This indicates that this axiom cannot be deduced from such a simpler condition as the continuity of D_∞. (We note that the proof of Axiom 3b in D_∞ depends on the type construction of D_∞.) Probably, Axiom 3b must be proved for each model D directly from its construction.

By Proposition 7.2.7, we can conclude that, if D satisfies Axioms 1-3, all the results on D_∞ vs. Λ due to Wadsworth [21,22] are valid on D vs. Λ. (So, $I \underset{D}{=} J$, $Y \underset{D}{=} \lambda f. \bigcup_{n=0}^{\infty} f^n(\perp)$, etc.)

§3. Prospects

We have completely ignored the other models of the λ-calculus. E_∞ in [21] and $P\omega$ in [17] are examples of non-extensional models, on which the formulation of the infinite normal form developed in this thesis is no longer valid.

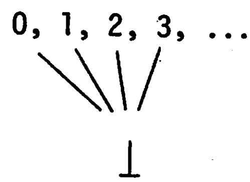
It would be possible to formulate the infinite normal forms on these non-extensional models. However, it does not seem possible to give such a clean theory as is possible on D_∞. Many interesting algebraic properties of $\mathbb{C}_{inf}$ are possible due to the extensionality of D_∞.

Another point to note is the problems of the compatibility and inconsistency among the λ-expressions. We say that two λ-expressions x, y are <u>inconsistent</u> if there exists $\delta \in \Delta$ such that $C(x,\delta) \neq \omega$, $C(y,\delta) \neq \omega$ and $C(x,\delta) \neq C(y,\delta)$. Some of the problems caused by the introduction of $\cup$ into the λ-calculus were discussed in §4, Chapter 6.

I am not sure at this point whether or not we could develop an interesting theory by introducing the $\cup$ operation into the λ-calculus.

Lastly in this section, we focus our attention on Λ^∞. One may ask what Λ^∞ is in actuality. We can say that Λ^∞ is arbitrary if we confine ourselves to the function over the integers. For example, if we consider the flat space of:

$N \equiv$



Λ^∞ gives all continuous functions from N to N.* Let us consider the following sequence of mappings over N. Let

$$f_0 : N \to N \text{ be } f(n) = \begin{cases} a_0 & \text{if } n = 0 \\ \perp & \text{if } n > 0 \end{cases}$$

$$f_1 : N \to N \text{ be } f(n) = \begin{cases} a_0 & \text{if } n = 0 \\ a_1 & \text{if } n = 1 \\ \perp & \text{if } n > 0 \end{cases}$$

$$\vdots$$

$$f_p : N \to N \text{ be } f(n) = \begin{cases} a_i & \text{if } n = i \leqq p \\ \perp & \text{if } n > p \end{cases}$$

$$\vdots$$

Then $f_0 \subseteq f_1 \subseteq \cdots \subseteq f_n \subseteq \cdots$. Since the choice of a_i is arbitrary, $f = \bigcup_{i=0}^{\infty} f_i$ can be the arbitrary continuous function $N \to N$.

This shows that to give the smooth property to Λ, we must

*This fact was suggested to the author by Manuel Blum.

inevitably include a rather non-computable structure. This is similar to extending the rational number to the real number. The real numbers contain all the arbitrary transcendental numbers, but we cannot discuss any problem in the elementary calculus excluding these objects. Another more hopeful view is that Λ^{∞} may give a certain significant proper subset of the continuous functions if the domain's lattice structure is more complex and has the continuous cardinality. For example, $R \equiv \{[a,b] \mid a \leqq b \text{ are real}\} \cup \{\emptyset\}$ with the partial order $\alpha < \beta$ if $\beta \subseteq \alpha$ and $\perp = \emptyset \in R$. (If LAMDA^{∞} is generated from LAMDA [17] in the same manner as Λ^{∞} is generated from Λ, it will probably be the case that $P\omega = \text{LAMDA}^{\infty}$, for any recursively enumerable set is in LAMDA and any set of integers is the limit of an increasing sequence of recursively enumerable sets.)

REFERENCES

[1] Abdali, K., "A simple λ-calculus model of programming languages," AEC Research and Development Report, Courant Institute for Mathematical Sciences (1973).

[2] Barendregt, H.P., "Some extensional term models for combinatory logic," Thesis, Utrecht (1971).

[3] Böhm, C., "The CUCH as a formal and description language," in Formal Language Description Language for Computer Programming (ed. Steel), Amsterdam (1966).

[4] ______, "Alcune proprieta della forme β-η-normali del λ-κ-calcolo," Publicazioni dell'Instituto per Applicazioni del Calcolo, No. 696, Rome (1968).

[5] Curry, H.B. and Fey, B., Combinatory Logic, Vol. 1, Amsterdam (1958).

[6] Landin, J.R. "A correspondence between Algol 60 and Church's λ-notation," Communications of the Association for Computing Machinery, vol. 8, nos. 2-3 (1965).

[7] Morris, J.H., "λ-calculus models of programming languages," Project MAC Report MAC-TR-56, M.I.T., Cambridge (1968).

[8] Morris, J.H. and Nakajima, R., "Mechanical characterization of the partial order in lattice model, D_∞, of the λ-calculus," Technical Report No. 18, Department of Computer Science, University of California, Berkeley (1973).

[9] Nakajima, R., "Infinite normal forms for the λ-calculus," Symposium on λ-calculus and Computer Science Theory, Springer Lecture Note Series in Computer Science (to appear).

[10] Park, D. "The Y combinator in Scott's λ-calculus models," Symposium on Theory of Programming, University of Warwick (1970).

[11] Plotkin, C.D., "The λ-calculus is ω-incomplete," SAI-RM-2, School of Artificial Intelligence, University of Edinburgh (1971).

[12] Reynolds, J., "Lattice theoretic approach to theory of computation," Lecture note, Syracuse University (1971).

[13] Scott, D., "A system of functional abstraction," unpublished note, Stanford University (1963).

[14] ______, "Outline of a mathematical theory of computation," Oxford Monograph PRG-2, Oxford University (1970).

[15] ______, "Continuous lattices," Oxford Monograph PRG-17, Oxford University (1972).

[16] ______, "Lattice theory, data types, and semantics," in Formal Semantics of Programming Languages (ed. Rustin), Courant Computer Science Symposium 2 (1970).

[17] ______, "λ-calculus and recursion theory," unpublished note, Oxford (1973).

[18] ______, "The lattice of flow diagrams," in Semantics of Algorithmic Languages (ed. Engeler), Springer Lecture Notes in Mathematics, vol. 188 (1971).

[19] Strachey, C., "Toward a formal semantics," in Formal Language Description Languages (ed. Steel), Amsterdam (1966).

[20] Vuillemin, J., "Correct and optimal implementation of recursion in a simple programming language," Journal of Computer and System Sciences, vol. 9, no. 3 (1975).

[21] Wadsworth, C.P., "Semantics and pragmatics of the λ-calculus," Ph.D. Thesis, Oxford University (1971).

[22] ______, "The relation between λ-expressions and their denotations in Scott's model for the λ-calculus," SIAM Journal of Computing (to appear).

[23] ______, "Approximate reduction and λ-calculus models," SIAM Journal of Computing (to appear).

[24] ______, "A general form of theorem of Böhm and its application to Scott's model of the λ-calculus," (in preparation).