การคณนาได้โดยแคลคูลัสแลมบ์ดาที่มีแบบรูป

นาย บดินทร์ สกุลเกียรติ

วิทยานิพนธ์นี้เป็นส่วนหนึ่งของการศึกษาตามหลักสูตรปริญญาวิทยาศาสตรมหาบัณฑิต
สาขาวิชาคณิตศาสตร์ ภาควิชาคณิตศาสตร์
คณะวิทยาศาสตร์ จุฬาลงกรณ์มหาวิทยาลัย
ปีการศึกษา 2552

COMPUTABILITY VIA THE LAMBDA-CALCULUS WITH PATTERNS

Mr. Bodin Skulkiat

A Thesis Submitted in Partial Fulfillment of the Requirements
for the Degree of Master of Science Program in Mathematics
Department of Mathematics
Faculty of Science
Chulalongkorn University
Academic Year 2009

Thesis Title	COMPUTABILITY VIA THE LAMBDA-CALCULUS WITH PATTERNS
By	Mr. Bodin Skulkiat
Field of Study	Mathematics
Thesis Advisor	Assistant Professor Pimpen Vejjajiva, Ph.D.
Thesis Co-Advisor	Associate Professor Mark Hall, Ph.D.

Accepted by the Faculty of Science, Chulalongkorn University in Partial Fulfillment of the Requirements for the Master's Degree

.... Dean of the Faculty of Science
(Professor Supot Hannongbua, Dr.rer.nat.)

THESIS COMMITTEE

.... Chairman
(Associate Professor Ajchara Harnchoowong, Ph.D.)

.... Thesis Advisor
(Assistant Professor Pimpen Vejjajiva, Ph.D.)

.... Thesis Co-Advisor
(Associate Professor Mark Hall, Ph.D.)

.... Examiner
(Assistant Professor Jaruloj Chongstitvatana, Ph.D.)

.... External Examiner
(Professor Chaweewan Ratanaprasert, Ph.D.)

บดินทร์ สกุลเกียรติ : การคณนาได้โดยแคลคูลัสแลมบ์ดาที่มีแบบรูป. (COMPUTABILITY VIA THE LAMBDA-CALCULUS WITH PATTERNS) อ.ที่ปรึกษาวิทยานิพนธ์หลัก : ผศ. ดร. พิมพ์เพ็ญ เวชชาชีวะ อ.ที่ปรึกษาวิทยานิพนธ์ร่วม : ASSOC. PROF. MARK HALL, Ph.D., 72 หน้า.

เรานำเสนอแนวคิดเรื่อง **การคณนาได้สัมพัทธ์กับโครงสร้าง** ซึ่งระบุว่าฟังก์ชันใดบนโดเมนของโครงสร้างอันดับที่หนึ่งสามารถคณนาได้โดยใช้แคลคูลัสแลมบ์ดาที่มีแบบรูป ในการนี้เรานิยาม **การสมภาคในโครงสร้าง** $\equiv_{\mathfrak{A}}$ เพื่อบ่งชี้ว่าสองพจน์ใดแทนสมาชิกเดียวกันในโดเมน เราแสดงให้เห็นว่าถึงแม้เราเพิ่มการสมภาคแบบใหม่แต่สมบัติพื้นฐานต่าง ๆของแคลคูลัสแลมบ์ดาที่มีแบบรูปดั้งเดิมยังคงอยู่ทุกประการ รวมทั้งสอดคล้องกับทฤษฎีบทเชอร์ช-รอสเซอร์ด้วย

เพื่อแสดงว่าการใช้คำว่า "การคณนาได้" นั้นสมเหตุผล เราพิสูจน์ว่า ถ้าฟังก์ชันบน $\mathbb{N}$ เป็นฟังก์ชันเวียนเกิดแล้ว ฟังก์ชันนั้นก็จะคณนาได้สัมพัทธ์กับ $\mathfrak{N}$ ซึ่งเป็นโครงสร้างมาตรฐานของ $\mathbb{N}$ สำหรับบทกลับของทฤษฎีบทนี้ เราสร้างรหัสแบบเกอเดิลสำหรับพจน์ต่าง ๆในแคลคูลัสแลมบ์ดาที่มีแบบรูปพร้อมทั้งศึกษาขั้นตอนวิธีในการลดรูปพจน์ดังเกล่าโดยใช้ฟังก์ชันเวียนเกิด

ภาควิชา คณิตศาสตร์	ลายมือชื่อนิสิต ......บดินทร์ สกุลเกียรติ
สาขาวิชา คณิตศาสตร์	ลายมือชื่ออ.ที่ปรึกษาวิทยานิพนธ์หลัก Pimpen Vejjajiva
ปีการศึกษา 2552	ลายมือชื่ออ.ที่ปรึกษาวิทยานิพนธ์ร่วม Mark E. Hall..

5072332623 : MAJOR MATHEMATICS
KEYWORDS: COMPUTABILITY / LAMBDA CALCULUS / LAMBDA CALCULUS WITH PATTERNS / CONGRUENCE RELATIVE TO A STRUCTURE

BODIN SKULKIAT: COMPUTABILITY VIA THE LAMBDA-CALCULUS WITH PATTERNS. THESIS ADVISOR: ASST. PROF. PIMPEN VEJJAJIVA, Ph.D., THESIS CO-ADVISOR: ASSOC. PROF. MARK HALL, Ph.D., 72 pp.

We introduce a concept of **computability relative to a structure**, which specifies which functions on the domain of a first-order structure are computable, using the lambda calculus with patterns. In doing so, we add a new congruence, $\equiv_{\mathfrak{A}}$, called a **congruence in a structure** to identify two syntactically different terms which represent the same element of the domain. We then show that, with the introduction of the new congruence, all the basic properties of the original lambda calculus with patterns still hold, including the Church-Rosser theorem.

To justify the word "computable", we first prove that if a total function on $\mathbb{N}$ is recursive then it is computable relative to $\mathfrak{N}$, the standard structure for $\mathbb{N}$. For the converse, we construct a Gödel coding for terms in the lambda calculus with patterns and investigate how to perform various steps in the reduction of an encoded term using recursive functions.

Department: Mathematics	Student's Signature บดินทร์ สกุลเกียรติ
Field of Study: Mathematics	Advisor's Signature Pimpen Vejjajiva
Academic Year: 2009	Co-Advisor's Signature Mark E. Hall

ACKNOWLEDGEMENTS

I would like to express my deep and sincere gratitude to my advisor, Assistant Professor Dr. Pimpen Vejjajiva, for her guidance and continued support, despite the difficult time.

I am greatly indebted to my co-advisor, Associate Professor Dr. Mark Edwin Hall, for the original idea and numerous advices in preparation of this thesis.

I also owe my gratitude to my dear friend, Assistant Professor Dr. Chariya Uiyyasathian, who introduced me to the beauty of mathematics in the first place.

I would like to thank all those who supported me in any respect during the three years period of this thesis.

Lastly, I am heartily thankful for my wife, whose encouragement and understanding made this thesis possible.

CONTENTS

CHAPTER I

INTRODUCTION

When discussing the computability of a function on $\mathbb{N}$, the standard definition to use is that of recursiveness, which we will quickly review. We begin with these three **initial functions**:

i. The **zero function**: $g(a) = 0$ for all $a \in \mathbb{N}$,

ii. The **successor function**: $g(a) = a + 1$ for all $a \in \mathbb{N}$,

iii. The **projection function**: $g_i^n(a_1, \ldots, a_n) = a_i$ for all $1 \leq i \leq n$ and $a_1, \ldots, a_n \in \mathbb{N}$,

and the following rules for obtaining new functions from given functions:

iv. **composition**: given functions $h(y_1, \ldots, y_m), k_1(x_1, \ldots, x_n), \ldots, k_m(x_1, \ldots, x_n)$, obtain the function g satisfying

$$g(x_1, \ldots, x_n) = h(k_1(x_1, \ldots, x_n), \ldots, k_m(x_1, \ldots, x_n)),$$

v. **primitive recursion**: given functions $h(x_1, \ldots, x_n)$ and $k(x_1, \ldots, x_{n+2})$, obtain the function g satisfying

$$g(x_1, \ldots, x_n, 0) = h(x_1, \ldots, x_n)$$

$$g(x_1, \ldots, x_n, y + 1) = k(x_1, \ldots, x_n, y, g(x_1, \ldots, x_n, y)),$$

vi. **restricted minimization**: given a function $h(x_1, \ldots, x_n, y)$ such that for any $x_1, \ldots, x_n$ there exists a y such that $h(x_1, \ldots, x_n, y) = 0$, obtain the function g satisfying

$$g(x_1, \ldots, x_n) = \mu y(h(x_1, \ldots, x_n, y) = 0),$$

where $\mu y(h(x_1, \ldots, x_n, y) = 0)$ denote the least y such that $h(x_1, \ldots, x_n, y) = 0$.

A function is said to be **recursive** if and only if it can be obtained from the initial functions by any finite number of applications of composition, primitive recursion, and minimization.

If we wish to extend this definition of recursiveness, a very general target to consider is to extend it to a first-order structure. Notice that there is no obvious way to do so, since both primitive recursion and restricted minimization depend on certain properties of the natural numbers. Alternatively a computable function is one for which we can write a “program” to compute. A good, mathematically rigorous “programming language” is the lambda calculus. Since the lambda calculus only deals with symbols, without any assumptions about their meanings, it is a good tool to help us extend the concept of computability to functions on an arbitrary first-order structure. To gain greater expressive power, we will use a lambda calculus with patterns, created by Pimpen Vejjajiva [5][6], which we will briefly describe.

Assume there are given an infinite sequence of distinct symbols, called **variables**, and a set of symbols which are distinct from the variables, called **constants**. The set of **patterns** is defined inductively as follows.

P1. Each variable and constant is a pattern.

P2. If P_1 is a pattern which is not a variable, P_2 is any pattern, and no variable occurs in both P_1 and P_2, then (P_1P_2) is a pattern.

Then, the set of **terms** is defined inductively as follows.

T1. Each variable and constant is a term, called an **atom**.

T2. If M and N are any terms, (MN) is a term, called an **application**.

T3. If P is any pattern and Q is any term, $(\lambda P.Q)$ is a term, called a **simple abstraction**.

T4. If P is any pattern, Q is any term, and A is any abstraction, $((\lambda P.Q) \mid A)$ is a term, called a **compound abstraction**.

An abstraction $(\lambda x.M)$ represents a function $f : x \mapsto M$. For example, $(\lambda x.x)$ represents an identity function. An application (MN) represents applying a function represented by M to an argument represented by N. For example, if we let $\mathsf{0}$ be a constant representing the natural number 0, $((\lambda x.x)\mathsf{0})$ represents applying an identity function to 0, which would result in 0. Avoiding complex technical details for the moment, we will use the symbol $\triangleright$ to represent the idea of "computing". In this notation the preceding example can be written as $((\lambda x.x)\mathsf{0}) \triangleright \mathsf{0}$. Here is a more involved example. If we let S be a constant representing the successor function and $\bar{a}$ be a constant representing any natural number a, then $((\lambda \mathsf{0}.\mathsf{0}) \mid (\lambda \mathsf{S}x.x))$ represents a predecessor function which maps $0 \mapsto 0$, i.e. $\big(((\lambda \mathsf{0}.\mathsf{0}) \mid (\lambda \mathsf{S}x.x))\mathsf{0}\big) \triangleright \mathsf{0}$, and maps $(a+1) \mapsto a$, i.e. $\big(((\lambda \mathsf{0}.\mathsf{0}) \mid (\lambda \mathsf{S}x.x))\mathsf{S}\bar{a}\big) \triangleright \bar{a}$.

The general idea of how to extend the concept of computable functions to a first-order structure $\mathfrak{A}$ for a language $\mathcal{L}$ is as follows. For each element $a \in |\mathfrak{A}|$, let $\bar{a}$ be a distinct symbol that does not occur in $\mathcal{L}$. Define patterns and terms as in the lambda calculus with patterns, using as constants all of the symbols in $\mathcal{L}$ together with all of the symbols $\bar{a}$. Then an n-ary function g on $|\mathfrak{A}|$ is **computable relative to** $\mathfrak{A}$ if and only if there is a term G such that for all $a_1, \ldots, a_n, a \in |\mathfrak{A}|$ we have $G\bar{a}_1 \ldots \bar{a}_n \triangleright \bar{a}$, whenever $g(a_1, \ldots, a_n) = a$. Informally speaking, a function on $|\mathfrak{A}|$ is computable relative to $\mathfrak{A}$ if and only if it can be represented by a term which captures all its functionalities. The interpretations of the elements of $\mathcal{L}$ in the structure $\mathfrak{A}$ are captured by adding a new congruence, $\equiv_{\mathfrak{A}}$, called congruence in a structure, to identify two syntactically different terms that represent the same element of the domain $|\mathfrak{A}|$. For example, $\mathsf{S}\bar{0} \equiv_{\mathfrak{A}} \bar{1}$, since they both represent 1 in $\mathbb{N}$.

The remainder of this thesis is organized as follows. In Chapter II, we begin with definition of λP-term and preliminary lemmas from previous work. Chapter III concerns definitions of the new congruence and the computability relative to a structure, and proofs of all basic properties. Chapter IV shows that our extension satisfies all the basic properties of the original lambda calculus with patterns, including the Church-Rosser theorem. To help justify the word "computable", we will lay the groundwork for a proof that a function on the natural numbers $\mathbb{N}$ is recursive if and only if it is computable relative to $\mathfrak{N}$, the standard structure for $\mathbb{N}$. We will show that every recursive total function on $\mathbb{N}$ is computable relative to $\mathfrak{N}$ in Chapter V. In preparation for proving the converse, in Chapter VI, we will construct a Gödel coding for terms in the lambda calculus with patterns and investigate how to perform various steps in the reduction of an encoded term using recursive functions.

CHAPTER II

λP-TERMS AND PRELIMINARY LEMMAS

All definitions in this chapter are based on the lambda calculus with patterns [5][6] with some adjustments. Let $\mathcal{L}$ be a first-order language and $\mathfrak{A}$ a structure for $\mathcal{L}$. We use $|\mathfrak{A}|$ to denote the domain of $\mathfrak{A}$.

2.1 λP-terms

Definition 2.1.1. For each element a in $|\mathfrak{A}|$, let $\bar{a}$ be a distinct symbol that is not in $\mathcal{L}$. We call all the nonlogical symbols in $\mathcal{L}$ together with all of the symbols $\bar{a}$ and two additional distinct symbols T and F **constants**. Assume also that an infinite sequence of distinct symbols $v_1, v_2, \ldots$, called **variables** is given. **Patterns** and **$\boldsymbol{\lambda}$P-terms** are expressions constructed using these symbols, as follows.

The set of **patterns** is the smallest set of expressions satisfying the following.

P1. All variables are patterns.

P2. The two constant symbols T and F, and all constant symbols in $\mathcal{L}$ are patterns.

P3. All function symbols f in $\mathcal{L}$ such that $f^{\mathfrak{A}}$ is one-to-one are patterns.

P4. If P is a pattern that is not a variable, Q is any pattern, and no variable occurs in both P and Q, then (PQ) is a pattern.

The set of **$\boldsymbol{\lambda}$P-terms** is divided into sets of atoms, applications, and abstractions, and is defined to be the smallest set of expressions satisfying the following.

T1. All variables and constants are λP-terms (these are the **atoms**).

T2. If P and Q are any λP-terms, then (PQ) is a λP-term (these are the **applications**).

T3. If P is any pattern and Q is any λP-term, then $(\lambda P.Q)$ is a λP-term (called a **simple abstraction**).

T4. If P is any pattern, Q is any λP-term, and A is any abstraction, then $((\lambda P.Q) \mid A)$ is a λP-term (called a **compound abstraction**).

An **abstraction** is either a simple abstraction or a compound abstraction.

Notation.

i. Parentheses will be omitted by using the convention of association to the left.

ii. $\lambda P.MN$ will abbreviate $(\lambda P.(MN))$.

iii. We may simply write "terms" for "λP-terms".

iv. Syntactic identity of expressions will be denoted by $\equiv$. That is, $M \equiv N$ if and only if M is exactly the same string of symbols as N.

Definition 2.1.2. An occurrence of a variable x in a term M is **bound** if it is in a subterm of M of the form $\lambda P.Q$ and it occurs in P; otherwise it is **free**. If x has at least one free occurence in M, it is called a **free variable** of M; the set of all such variables is denoted by $FV(M)$.

Definition 2.1.3. Let M and $\underline{N} = N_1, \ldots, N_k, k \geq 1$, be terms and $\underline{x} = x_1, \ldots, x_k$ be distinct variables. The result of substituting N_i for all free occurrences of $x_i, i = 1, 2, \ldots, k$, in M, denoted by $[N_1/x_1, \ldots, N_k/x_k]M$ or $[\underline{N}/\underline{x}]M$, is defined as follows.

a. $[\underline{N}/\underline{x}]x_i \equiv N_i$ for all $1 \leq i \leq k$;

b. $[\underline{N}/\underline{x}]a \equiv a$ for all atoms a such that $a \notin \{x_1, \ldots, x_k\}$;

c. $[\underline{N}/\underline{x}](PQ) \equiv ([\underline{N}/\underline{x}]P[\underline{N}/\underline{x}]Q)$;

d. $$[\underline{N}/\underline{x}](\lambda P.Q) \equiv \begin{cases} \lambda P.Q & \text{if } \{x_1, \ldots, x_k\} \cap FV(\lambda P.Q) = \varnothing; \\ [N_{i_1}/x_{i_1}, \ldots, N_{i_m}/x_{i_m}](\lambda P.Q) & \\ & \text{if } \{x_1, \ldots, x_k\} \cap FV(\lambda P.Q) = \{x_{i_1}, \ldots, x_{i_m}\}; \end{cases}$$

e. $[\underline{N}/\underline{x}](\lambda P.Q) \equiv \lambda P.[\underline{N}/\underline{x}]Q$
if $\{x_1, \ldots, x_k\} \subseteq FV(\lambda P.Q)$ and $FV(P) \cap FV(N_1 \ldots N_k) = \varnothing$;

f. $[\underline{N}/\underline{x}](\lambda P.Q) \equiv [\underline{N}/\underline{x}](\lambda[z/y]P.[z/y]Q)$
if $\{x_1, \ldots, x_k\} \subseteq FV(\lambda P.Q)$ and $FV(P) \cap FV(N_1 \ldots N_k) \neq \varnothing$, where y is the first variable in $FV(P) \cap FV(N_1 \ldots N_k)$ and z is chosen to be the first variable which is not in $FV(PQN_1 \ldots N_k)$;

g. $[\underline{N}/\underline{x}](\lambda P.Q \mid A) \equiv ([\underline{N}/\underline{x}](\lambda P.Q) \mid [\underline{N}/\underline{x}]A)$.

Definition 2.1.4. Let A be an occurrence of a simple abstraction $\lambda P.Q$ in a term M. Let $x \in FV(P)$ and $y \notin FV(PQ)$. The act of replacing A by $\lambda[y/x]P.[y/x]Q$ is called a **change of bound variable** or an **α-step** in M.

We say M **1α-converts** to a term N, denoted by $M \equiv_{1\alpha} N$, if N is obtained from M by a single-step change of bound variable.

We say M is **congruent** to N, or M **α-converts** to N, denoted by $M \equiv_\alpha N$, if N is obtained from M by a finite (possibly empty) sequence of changes of bound variables.

2.2 Preliminary Lemmas from Previous Work

The following lemmas and notes are from [5], of which the corresponding result number will be included in brackets for the ease of reading.

Lemma 2.2.1. [Corollary 2.1.12] *Let* $\underline{x} = x_1, \ldots, x_k, k \geq 1$, *be distinct variables,* M, $\underline{N} = N_1, \ldots, N_k$ *be terms, and* $\lambda P.Q$ *be a simple abstraction.*

a. If $\{x_1, \ldots, x_k\} \cap FV(M) = \{x_{i_1}, \ldots, x_{i_m}\}$, *then* $[N_1/x_1, \ldots, N_k/x_k]M \equiv [N_{i_1}/x_{i_1}, \ldots, N_{i_m}/x_{i_m}]M$.

b. If $FV(P) \cap FV(x_1 \ldots x_k N_1 \ldots N_k) = \varnothing$, *then* $[\underline{N}/\underline{x}](\lambda P.Q) \equiv \lambda P.[\underline{N}/\underline{x}]Q$.

Lemma 2.2.2. [Lemma 2.2.4] *Let* M *and* N *be terms such that* $M \equiv_\alpha N$.

a. If $M \equiv M_1M_2$, *then* $N \equiv N_1N_2$ *for some terms* N_1 *and* N_2, *where* $M_i \equiv_\alpha N_i$, $i = 1, 2$;

b. if $M \equiv \lambda P.Q$, *and no variable in* P *has been changed, then* $N \equiv \lambda P.Q'$ *for some term* Q' *such that* $Q \equiv_\alpha Q'$;

c. if $M \equiv (\lambda P.Q \mid A)$ *then* $N \equiv (\lambda P'.Q' \mid A')$ *for some abstractions* $\lambda P'.Q'$ *and* A' *where* $\lambda P.Q \equiv_\alpha \lambda P'.Q'$ *and* $A \equiv_\alpha A'$.

Lemma 2.2.3. [Lemma 2.2.5]

a. For any terms M *and* N, *if* $M \equiv_\alpha N$, *then* $FV(M) = FV(N)$.

b. For any term M, *any variables* $x_1, \ldots, x_n, n \geq 1$, *there exists a term* M' *such that* $M \equiv_\alpha M'$ *and none of* $x_1, \ldots, x_n$ *is bound in* M'.

Lemma 2.2.4. [Lemma 2.2.6] *Let* x *and* v *be distinct variables, and* V *and* M *be terms. If* $v \notin FV(M)$, *then* $[V/v][v/x]M \equiv_\alpha [V/x]M$.

Lemma 2.2.5. [Lemma 2.2.7] *Let* $\underline{x} = x_1, \ldots, x_k, k \geq 1$ *be distinct variables, and* $\underline{N} = N_1, \ldots, N_k$*,* $\underline{N}' = N'_1, \ldots, N'_k$ *be terms such that* $N_i \equiv_\alpha N'_i$ *for all* $1 \leq i \leq k$*. For any terms* M *and* M'*, if* $M \equiv_\alpha M'$*, then* $[\underline{N}/\underline{x}]M \equiv_\alpha [\underline{N}'/\underline{x}]M'$*.*

CHAPTER III

COMPUTABILITY RELATIVE TO A STRUCTURE

In this chapter, we will define congruence in a structure and computability relative to a structure, and prove all the basic properties.

3.1 Congruence in a Structure

Definition 3.1.1. Single-Step Congruence in $\mathfrak{A}$, denoted by $\equiv_{1\mathfrak{A}}$, is defined as follows.

C1. For any constant symbol c in $\mathcal{L}$ and any a in $|\mathfrak{A}|$,

$$c \equiv_{1\mathfrak{A}} \bar{a} \text{ if } c^{\mathfrak{A}} = a.$$

C2. For any n-ary function symbol f in $\mathcal{L}$ and any $a, a_1, \ldots, a_n$ in $|\mathfrak{A}|$,

$$f\bar{a}_1 \ldots \bar{a}_n \equiv_{1\mathfrak{A}} \bar{a} \text{ if } f^{\mathfrak{A}}(a_1, \ldots, a_n) = a.$$

C3. For any n-ary relation symbol r in $\mathcal{L}$ and any $a_1, \ldots, a_n$ in $|\mathfrak{A}|$,

$$r\bar{a}_1 \ldots \bar{a}_n \equiv_{1\mathfrak{A}} \begin{cases} \mathsf{T} & \text{if } (a_1, \ldots, a_n) \in r^{\mathfrak{A}}, \\ \mathsf{F} & \text{otherwise.} \end{cases}$$

C4. For any terms M and N,

$$M \equiv_{1\mathfrak{A}} N \text{ if } N \equiv_{1\mathfrak{A}} M \text{ by C1, C2, or C3.}$$

C5. Let P be any pattern; A be any abstraction; and M, N, and Q be any terms such that $M \equiv_{1\mathfrak{A}} N$. Then

i. $MQ \equiv_{1\mathfrak{A}} NQ$.

ii. $QM \equiv_{1\mathfrak{A}} QN$.

iii. $\lambda P.M \equiv_{1\mathfrak{A}} \lambda P.N$.

iv. $(\lambda P.M \mid A) \equiv_{1\mathfrak{A}} (\lambda P.N \mid A)$.

v. $(\lambda P.Q \mid M) \equiv_{1\mathfrak{A}} (\lambda P.Q \mid N)$ if M and N are abstractions.

For any terms M and N, we write $M \equiv^0_{1\mathfrak{A}} N$ if $M \equiv_{1\mathfrak{A}} N$ by C1, C2, C3, or C4.

If L is an occurence of a term M in a term Q and $M \equiv^0_{1\mathfrak{A}} N$, the act of replacing L by N is called a **$1\mathfrak{A}$-conversion** in Q.

Note 3.1.2.

a. If $M \equiv_{1\mathfrak{A}} N$ where M and N are terms which are not atomic, then $M \not\equiv^0_{1\mathfrak{A}} N$.

b. If $M \equiv_{1\mathfrak{A}} N$ and $FV(M) \cup FV(N) \neq \varnothing$, then $M \not\equiv^0_{1\mathfrak{A}} N$.

c. If $M \equiv_{1\mathfrak{A}} N$ but $M \not\equiv^0_{1\mathfrak{A}} N$, then M and N are of the same form.

d. For any variable x and any term M, $x \not\equiv_{1\mathfrak{A}} M$.

Definition 3.1.3. For any terms M and N, we say M is **congruent in $\mathfrak{A}$** to N, denoted by $M \equiv_{\mathfrak{A}} N$, if there exists a sequence of terms $M \equiv M_1, \ldots, M_n \equiv N, n \geq 1$, such that for each $1 \leq i < n$, $M_i \equiv_{1\mathfrak{A}} M_{i+1}$.

If L is an occurence of a term M in a term Q and $M \equiv_{\mathfrak{A}} N$, the act of replacing L with N is called an **$\mathfrak{A}$-conversion** in Q.

Note 3.1.4.

a. If $M \equiv_{\mathfrak{A}} N$ and M contains an abstraction then M and N are of the same form.

b. If $M_1M_2 \equiv_{\mathfrak{A}} N_1N_2$ with no $\equiv^0_{1\mathfrak{A}}$ in the sequence of congruences, then $M_1 \equiv_{\mathfrak{A}} N_1$ and $M_2 \equiv_{\mathfrak{A}} N_2$.

Lemma 3.1.5. *For any terms M and N, if $M \equiv_{1\mathfrak{A}} N$, then $N \equiv_{1\mathfrak{A}} M$.*

Proof. Let M and N be terms. We induct on M. From Definition 3.1.1, we can see that $M \equiv_{1\mathfrak{A}} N$ by C1, C2, or C3 if and only if $N \equiv_{1\mathfrak{A}} M$ by C4. Thus it remains to show only the induction step. Suppose $M \equiv_{1\mathfrak{A}} N$ by C5. We will give a proof only for the following case, since the remaining are similar. Assume $M \equiv M_1Q$ and $N \equiv N_1Q$ for some terms M_1, N_1, and Q such that $M_1 \equiv_{1\mathfrak{A}} N_1$. By the induction hypothesis we have $N_1 \equiv_{1\mathfrak{A}} M_1$. So $N \equiv N_1Q \equiv_{1\mathfrak{A}} M_1Q \equiv M$. □

Corollary 3.1.6. *For any terms M and N, if $M \equiv_{\mathfrak{A}} N$ then $N \equiv_{\mathfrak{A}} M$.*

Proof. This follows directly from Lemma 3.1.5. □

Corollary 3.1.7. *The relation $\equiv_{\mathfrak{A}}$ is an equivalence relation.*

Proof. It is clear that $\equiv_{\mathfrak{A}}$ is reflexive and transitive. By Corollary 3.1.6 we have that $\equiv_{\mathfrak{A}}$ is symmetric. Hence $\equiv_{\mathfrak{A}}$ is an equivalence relation. □

Remark. Note that $\equiv_{1\mathfrak{A}}$ is symmeric, but neither reflexive nor transitive.

Proposition 3.1.8. *If $M \equiv_{1\mathfrak{A}} N$, FV(M) = FV(N).*

Proof. This can be easily proved by induction. □

Lemma 3.1.9. *Let f be a k-ary function symbol and $M_1, \ldots, M_k$, N be terms. If $fM_1 \ldots M_k \equiv_{1\mathfrak{A}} N$ but $fM_1 \ldots M_k \not\equiv^0_{1\mathfrak{A}} N$, then $N \equiv fN_1 \ldots N_k$ for some terms N_i such that either $M_i \equiv N_i$ or $M_i \equiv_{1\mathfrak{A}} N_i$, $1 \leq i \leq k$.*

Proof. Assume $fM_1 \ldots M_k \equiv_{1\mathfrak{A}} N$ and $fM_1 \ldots M_k \not\equiv^0_{1\mathfrak{A}} N$. Induct on k. If $k = 1$ then $fM_1 \equiv_{1\mathfrak{A}} N \equiv fN_1$ for some term N_1 such that $M_1 \equiv_{1\mathfrak{A}} N_1$. Suppose $k > 1$. Then $(fM_1 \ldots M_{k-1})M_k \equiv_{1\mathfrak{A}} N \equiv N'N_k$ for some terms N' and N_k.

Case 1. $N' \equiv (fM_1 \ldots M_{k-1})$.

Then $M_k \equiv_{1\mathfrak{A}} N_k$, so $N \equiv (fM_1 \ldots M_{k-1})N_k$.

Case 2. $N_k \equiv M_k$. Then $fM_1 \ldots M_{k-1} \equiv_{1\mathfrak{A}} N'$.

By induction, $N' \equiv fN_1 \ldots N_{k-1}$ for some terms N_i such that either $M_i \equiv N_i$ or $M_i \equiv_{1\mathfrak{A}} N_i$, $1 \le i \le k-1$, so $N \equiv fN_1 \ldots N_k$. □

Lemma 3.1.10. *Let f be a k-ary function symbol and $M_1, \ldots, M_k$, N be terms. If $fM_1 \ldots M_k \equiv_{\mathfrak{A}} N$ with no $\equiv^0_{1\mathfrak{A}}$ in the sequence of congruences, then $N \equiv fN_1 \ldots N_k$ for some terms N_i such that $M_i \equiv_{\mathfrak{A}} N_i$, $1 \le i \le k$.*

Proof. This follows directly from Lemma 3.1.9. □

Lemma 3.1.11. *For any a, b in $|\mathfrak{A}|$, if $\bar{a} \equiv_{\mathfrak{A}} \bar{b}$ by a sequence of terms $\bar{a} \equiv M_1, \ldots, M_k \equiv \bar{b}$, $k \ge 1$, then k is odd.*

Proof. We will prove this by contradiction. Let k be the least even number such that $\bar{a} \equiv M_1, \ldots, M_k \equiv \bar{b}$ for some a, b in $|\mathfrak{A}|$. Consider $M_{k-1} \equiv_{1\mathfrak{A}} M_k \equiv \bar{b}$.

Case 1. $M_{k-1} \equiv c$ for some constant symbol c in $\mathcal{L}$.

Since $M_{k-1} \equiv c \not\equiv \bar{a} \equiv M_1$, $k-1 \ne 1$. In fact $k > 2$. Now consider $M_{k-2} \equiv_{1\mathfrak{A}} M_{k-1} \equiv c$. We must have $M_{k-2} \equiv \bar{a}_1$ for some a_1 in $|\mathfrak{A}|$. Thus $\bar{a} \equiv M_1, \ldots, M_{k-2} \equiv \bar{a}_1$. This contradicts the fact that k is the least such even number.

Case 2. $M_{k-1} \equiv f\bar{b}_1 \ldots \bar{b}_n$ for some n-ary function symbol f and some $b_1, \ldots, b_n$ in $|\mathfrak{A}|$.

Since $\bar{a}$ is not of the same form as $f\bar{b}_1 \ldots \bar{b}_n$, by Note 3.1.2, $M_j \equiv^0_{1\mathfrak{A}} M_{j+1}$ for some $1 \le j < k-1$. Let m be the largest such j. Since $M_m \equiv^0_{1\mathfrak{A}} M_{m+1}$ and $M_{m+1} \equiv_{\mathfrak{A}} M_{k-1} \equiv f\bar{b}_1 \ldots \bar{b}_n$ with no $\equiv^0_{1\mathfrak{A}}$ in the sequence of congruence, by Lemma 3.1.10, $M_{m+1} \equiv f\bar{a}_1 \ldots \bar{a}_n$ and $M_m \equiv \bar{a}_0$ for some $a_0, a_1, \ldots, a_n$ in $|\mathfrak{A}|$ such that $\bar{b}_i \equiv_{\mathfrak{A}} \bar{a}_i$, $1 \le i \le n$. Since $\bar{a} \equiv M_1, \ldots, M_m \equiv \bar{a}_0$ and $m < k$, m must be odd. Let $K_j \equiv M_{j+m-1}$ for $1 \le j \le k-m+1$. Then

$\bar{a}_0 \equiv M_m \equiv K_1, \ldots, K_{k-m+1} \equiv M_k \equiv \bar{b}$. Since $k - m + 1$ is even, this contradicts the fact that k is the least such even number. □

Lemma 3.1.12. *For any a, b in $|\mathfrak{A}|$, if $\bar{a} \equiv_{\mathfrak{A}} \bar{b}$ then $a = b$.*

Proof. Let a, b in $|\mathfrak{A}|$ be such that $\bar{a} \equiv_{\mathfrak{A}} \bar{b}$ by a sequence of terms $\bar{a} \equiv M_1, \ldots, M_k \equiv \bar{b}$, $k \geq 1$. Induct on k. If $k = 1$ then $\bar{a} \equiv M_1 \equiv \bar{b}$ and we are done. Suppose $k > 1$. In fact, by Lemma 3.1.11, $k \geq 3$. Consider $M_{k-1} \equiv_{1\mathfrak{A}} M_k \equiv \bar{b}$.

Case 1. $M_{k-1} \equiv c$ for some constant symbol c in $\mathcal{L}$.

Since $M_{k-2} \equiv_{1\mathfrak{A}} M_{k-1} \equiv c$, $M_{k-2} \equiv \bar{a}_0$ for some a_0 in $|\mathfrak{A}|$. Thus $\bar{a} \equiv M_1, \ldots, M_{k-2} \equiv \bar{a}_0 \equiv_{1\mathfrak{A}} c \equiv_{1\mathfrak{A}} \bar{b}$. By induction we have $a = a_0 = b$.

Case 2. $M_{k-1} \equiv f\bar{b}_1 \ldots \bar{b}_n$ for some n-ary function symbol f and some $b_1, \ldots, b_n$ in $|\mathfrak{A}|$. Since $\bar{a}$ is not of the same form as $f\bar{b}_1 \ldots \bar{b}_n$, by Note 3.1.2, $M_j \equiv^0_{1\mathfrak{A}} M_{j+1}$ for some $1 \leq j < k-1$. Let m be the largest such j. Since $M_m \equiv^0_{1\mathfrak{A}} M_{m+1}$ and $M_{m+1} \equiv_{\mathfrak{A}} M_{k-1} \equiv f\bar{b}_1 \ldots \bar{b}_n$ with no $\equiv^0_{1\mathfrak{A}}$ in the sequence of congruence, by Lemma 3.1.10, $M_{m+1} \equiv f\bar{a}_1 \ldots \bar{a}_n$ and $M_m \equiv \bar{a}_0$ for some $a_0, a_1, \ldots, a_n$ in $|\mathfrak{A}|$ such that $\bar{b}_i \equiv_{\mathfrak{A}} \bar{a}_i$, $1 \leq i \leq n$.

(2.1) $m = 1$.

Then $\bar{a} \equiv^0_{1\mathfrak{A}} f\bar{a}_1 \ldots \bar{a}_n \equiv M_2, \ldots, M_{k-1} \equiv f\bar{b}_1 \ldots \bar{b}_n \equiv^0_{1\mathfrak{A}} \bar{b}$ with no other $\equiv^0_{1\mathfrak{A}}$ in the sequence of congruence. Since $\bar{a}_i \equiv_{\mathfrak{A}} \bar{b}_i$, by induction we have $a_i = b_i$ for all $1 \leq i \leq n$, so $a = f^{\mathfrak{A}}(a_1, \ldots, a_n) = f^{\mathfrak{A}}(b_1, \ldots, b_n) = b$.

(2.2) $m > 1$.

Then $\bar{a} \equiv M_1, \ldots, M_m \equiv \bar{a}_0 \equiv^0_{1\mathfrak{A}} f\bar{a}_1 \ldots \bar{a}_n \equiv M_{m+1}, \ldots, M_k \equiv \bar{b}$. By induction we have $a = a_0 = b$. □

Lemma 3.1.13. *Let P be a pattern with $FV(P) = \{x_1, \ldots, x_k\}, k \geq 1$, and $\underline{U} = U_1, \ldots, U_k, \underline{V} = V_1, \ldots, V_k$ be terms. Let $\underline{x} = x_1, \ldots, x_k$. If $[\underline{U}/\underline{x}]P \equiv_{\mathfrak{A}} [\underline{V}/\underline{x}]P$,* *then* $U_i \equiv_{\mathfrak{A}} V_i$ *for all* $1 \leq i \leq k$.

Proof. Assume $[\underline{U}/\underline{x}]P \equiv_{\mathfrak{A}} [\underline{V}/\underline{x}]P$. Induct on P.

Case 1. $P \equiv x_1$.

Then $U_1 \equiv [U_1/x_1]P \equiv_{\mathfrak{A}} [V_1/x_1]P \equiv V_1$.

Case 2. $P \equiv P_1P_2$.

Let $[\underline{U}/\underline{x}]P \equiv_{\mathfrak{A}} [\underline{V}/\underline{x}]P$ by a sequence of terms $[\underline{U}/\underline{x}]P \equiv K_1, \ldots, K_s \equiv [\underline{V}/\underline{x}]P$, $s \geq 1$.

(2.1) $K_i \not\equiv^0_{1\mathfrak{A}} K_{i+1}$ for all $1 \leq i < s$.

By Note 3.1.4(b) $[\underline{U}/\underline{x}]P_1 \equiv_{\mathfrak{A}} [\underline{V}/\underline{x}]P_1$ and $[\underline{U}/\underline{x}]P_2 \equiv_{\mathfrak{A}} [\underline{V}/\underline{x}]P_2$. By induction, $U_i \equiv_{\mathfrak{A}} V_i$ for all $1 \leq i \leq k$.

(2.2) $K_i \equiv^0_{1\mathfrak{A}} K_{i+1}$ for some $1 \leq i < s$.

Let n be the first such i. Since $K_n \equiv^0_{1\mathfrak{A}} K_{n+1}$, $K_n \equiv f\bar{a}_1 \ldots \bar{a}_q$ and $K_{n+1} \equiv \bar{a}$ for some function symbol f, and some $a, a_1, \ldots, a_q \in |\mathfrak{A}|$, $q \geq 1$. Since $K_{n+1} \equiv_{\mathfrak{A}} K_s \equiv [\underline{V}/\underline{x}]P$ and K_{n+1} is not of the same form as $[\underline{V}/\underline{x}]P$, by Note 3.1.2 $K_j \equiv^0_{1\mathfrak{A}} K_{j+1}$ for some $n+1 \leq j < s$. Let m be the most such j. Then we have $K_m \equiv \bar{b}$ and $K_{m+1} \equiv g\bar{b}_1 \ldots \bar{b}_r$ for some function symbol g, and some $b, b_1, \ldots, b_r \in |\mathfrak{A}|, r \geq 1$. Since a pattern cannot begin with a variable and $[\underline{U}/\underline{x}]P \equiv_{\mathfrak{A}} K_n \equiv f\bar{a}_1 \ldots \bar{a}_q$ with $K_j \not\equiv^0_{1\mathfrak{A}} K_{j+1}$ for all $1 \leq j < n$, by induction on q, the pattern P must begin with f. Similarly for $g\bar{b}_1 \ldots \bar{b}_r \equiv K_{m+1} \equiv_{\mathfrak{A}} [\underline{V}/\underline{x}]P$, the pattern P must begin with g. Therefore $f \equiv g$. Since $f\bar{a}_1 \ldots \bar{a}_q \equiv^0_{1\mathfrak{A}} \bar{a} \equiv K_{n+1} \equiv_{\mathfrak{A}} K_m \equiv \bar{b} \equiv^0_{1\mathfrak{A}} g\bar{b}_1 \ldots \bar{b}_r \equiv f\bar{b}_1 \ldots \bar{b}_r$, by Lemma 3.1.12, we have $f^{\mathfrak{A}}(a_1, \ldots, a_q) = a = b = f^{\mathfrak{A}}(b_1, \ldots, b_r)$. Since f is in a pattern,

$f^{\mathfrak{A}}$ is one-to-one, so $q = r$ and $a_j = b_j$ for all $1 \le j \le q$. Then

$$[\underline{U}/\underline{x}]P \equiv K_1 \equiv_{\mathfrak{A}} K_n \equiv f\bar{a}_1 \ldots \bar{a}_q$$
$$\equiv g\bar{b}_1 \ldots \bar{b}_r \equiv K_{m+1} \equiv_{\mathfrak{A}} K_s \equiv [\underline{V}/\underline{x}]P,$$

with $K_j \not\equiv^0_{1\mathfrak{A}} K_{j+1}$ for all $1 \le j < n$ and $m+1 \le j < s$. Hence by Case 2.1 $U_i \equiv_{\mathfrak{A}} V_i$ for all $1 \le i \le k$. $\square$

Lemma 3.1.14. *Let Q and Q' be terms, x and y be variables. If $y \notin FV(Q)$ and $[y/x]Q \equiv_{1\mathfrak{A}} Q'$, then $Q \equiv_{1\mathfrak{A}} Q''$ for some term Q'' such that $Q'' \equiv_\alpha [x/y]Q'$.*

Proof. Assume $y \notin FV(Q)$ and $[y/x]Q \equiv_{1\mathfrak{A}} Q'$.

Case 1. $x \notin FV(Q)$.

Since $y \notin FV(Q)$ and $Q \equiv [y/x]Q \equiv_{1\mathfrak{A}} Q'$, by Proposition 3.1.8, we have $y \notin FV(Q')$. Then $Q \equiv [y/x]Q \equiv_{1\mathfrak{A}} Q' \equiv [x/y]Q'$.

Case 2. $x \in FV(Q)$.

We will induct on Q.

(2.1) Q is atomic.

Since $x \in FV(Q)$, $Q \equiv x$. Then $y \equiv [y/x]x \equiv [y/x]Q \equiv_{1\mathfrak{A}} Q'$, which is impossible. Therefore this case cannot occur.

(2.2) $Q \equiv Q_1Q_2$ for some terms Q_1 and Q_2.

Then $[y/x]Q_1[y/x]Q_2 \equiv [y/x]Q \equiv_{1\mathfrak{A}} Q'$. Since $x \in FV(Q), y \in FV([y/x]Q)$, so Q' is not atomic. Without loss of generality, assume $Q' \equiv Q'_1[y/x]Q_2$ for some term Q'_1 such that $[y/x]Q_1 \equiv_{1\mathfrak{A}} Q'_1$. Then by induction, $Q_1 \equiv_{1\mathfrak{A}} Q''_1 \equiv_\alpha [x/y]Q'_1$ for some term Q''_1. Since $y \notin FV(Q)$, we have $y \notin FV(Q_2)$.

$$\begin{aligned} \text{Then } Q \equiv Q_1Q_2 \equiv_{1\mathfrak{A}} Q_1''Q_2 \equiv_\alpha {} & [x/y]Q_1'Q_2 \\ \equiv_\alpha {} & [x/y]Q_1'[x/y][y/x]Q_2 \quad \text{(by Lemma 2.2.4)} \\ \equiv {} & [x/y](Q_1'[y/x]Q_2) \\ \equiv {} & [x/y]Q'. \end{aligned}$$

We are done with $Q'' \equiv Q_1''Q_2$.

(2.3) $Q \equiv \lambda P.Q_1$ for some pattern P and some term Q_1.

Since $x \in FV(Q), x \notin FV(P)$ and $x \in FV(Q_1)$. Then $[y/x]Q \equiv [y/x]\lambda P.Q_1 \equiv \lambda[z/y]P.[y/x][z/y]Q_1$ where $z \equiv y$ if $y \notin FV(P)$, otherwise z is the first variable not in $FV(PQ_1)$. So $Q' \equiv \lambda[z/y]P.Q_1'$ where $[y/x][z/y]Q_1 \equiv_{1\mathfrak{A}} Q_1'$. By induction, we have $[z/y]Q_1 \equiv_{1\mathfrak{A}} Q_1'' \equiv_\alpha [x/y]Q_1'$ for some term Q_1'', and $Q_1 \equiv_{1\mathfrak{A}} Q_1''' \equiv_\alpha [y/z]Q_1''$ for some term Q_1'''. Note that $\lambda P.[y/z][x/y]Q_1' \equiv_\alpha \lambda[z/y]P.[z/y][y/z][x/y]Q_1'$ because $z \equiv y$ if $y \notin FV(P)$, and otherwise $z \notin FV(P[y/z][x/y]Q_1')$. (1)

$$\begin{aligned} \text{Then } Q & \equiv \lambda P.Q_1 \\ & \equiv_{1\mathfrak{A}} \lambda P.Q_1''' \\ & \equiv_\alpha \lambda P.[y/z]Q_1'' \\ & \equiv_\alpha \lambda P.[y/z][x/y]Q_1' \\ & \equiv_\alpha \lambda[z/y]P.[z/y][y/z][x/y]Q_1' && \big(\text{by } (1)\big) \\ & \equiv_\alpha \lambda[z/y]P.[x/y]Q_1' && \text{(by Lemma 2.2.4)} \\ & \equiv [x/y]\lambda[z/y]P.Q_1' && \big(\text{since } \{x,y\} \cap FV([z/y]P) = \varnothing\big) \\ & \equiv [x/y]Q' \end{aligned}$$

We are done with $Q'' \equiv \lambda P.Q_1'''$.

(2.4) $Q \equiv (\lambda P.Q_1 \mid A)$ for some pattern P, some term Q_1, and some abstraction A.

The proof for this case is similar to Case 2.2. □

Lemma 3.1.15. *Let M, N, and N' be terms. If $M \equiv_{1\alpha} N \equiv_{1\mathfrak{A}} N'$, then $M \equiv_{1\mathfrak{A}} M' \equiv_\alpha N'$ for some term M'.*

Proof. Assume $M \equiv_{1\alpha} N \equiv_{1\mathfrak{A}} N'$. Let $\lambda P.Q$ be the simple abstraction in M which gets replaced by $\lambda[y/x]P.[y/x]Q$ when M 1α-converts to N, where $x \in FV(P)$ and $y \notin FV(PQ)$. We will induct on M. Since M contains a simple abstraction, M cannot be an atom.

Case 1. $M \equiv \lambda P.Q$.

Then $\lambda[y/x]P.[y/x]Q \equiv N \equiv_{1\mathfrak{A}} N'$. Thus $N' \equiv \lambda[y/x]P.Q'$ form some term Q' where $[y/x]Q \equiv_{1\mathfrak{A}} Q'$. Note that since $x \in FV(P)$ and $y \notin FV(P[x/y]Q')$, we have $\lambda P.[x/y]Q' \equiv_{1\alpha} \lambda[y/x]P.[y/x][x/y]Q'$. Since $x \notin FV([y/x]Q)$ and $[y/x]Q \equiv_{1\mathfrak{A}} Q'$, by Proposition 3.1.8, we have $x \notin FV(Q')$. Then by Lemma 2.2.4, $[y/x][x/y]Q' \equiv_\alpha [y/y]Q' \equiv Q'$. Since $y \notin FV(Q)$ and $[y/x]Q \equiv_{1\mathfrak{A}} Q'$, by Lemma 3.1.14, we have $Q \equiv_{1\mathfrak{A}} Q''$ for some term Q'' such that $Q'' \equiv_\alpha [x/y]Q'$. Let $M' \equiv \lambda P.Q''$. Then $M \equiv \lambda P.Q \equiv_{1\mathfrak{A}} \lambda P.Q'' \equiv M'$, and

$$\begin{aligned} M' \equiv \lambda P.Q'' &\equiv_\alpha \lambda P.[x/y]Q' \\ &\equiv_{1\alpha} \lambda[y/x]P.[y/x][x/y]Q' \\ &\equiv_\alpha \lambda[y/x]P.Q' \\ &\equiv N'. \end{aligned}$$

Case 2. $M \equiv \lambda L.M_1$ for some pattern L and some term M_1.

Since $M \equiv_{1\alpha} N$, $N \equiv \lambda L.N_1$ for some term N_1 such that $M_1 \equiv_{1\alpha} N_1$. Since

$N \equiv_{1\mathfrak{A}} N'$, $N' \equiv \lambda L.N_1'$ for some term N_1' such that $N_1 \equiv_{1\mathfrak{A}} N_1'$. Since $M_1 \equiv_{1\alpha} N_1 \equiv_{1\mathfrak{A}} N_1'$, by induction we have $M_1 \equiv_{1\mathfrak{A}} M_1' \equiv_\alpha N_1'$ for some term M_1'. Then $M \equiv \lambda L.M_1 \equiv_{1\mathfrak{A}} \lambda L.M_1' \equiv_\alpha \lambda L.N_1' \equiv N_1'$. Choose $M' \equiv \lambda L.M_1'$.

Case 3. $M \equiv M_1 M_2$ for some terms M_1 and M_2.

Without loss of generality, assume $\lambda P.Q$ is in M_1. Since $M_1 M_2 \equiv M \equiv_{1\alpha} N$, $N \equiv N_1 M_2$ for some term N_1 such that $M_1 \equiv_{1\alpha} N_1$. Note that N_1 contains an abstraction, hence N does as well, and thus $N \not\equiv^0_{1\mathfrak{A}} N'$. Since $N_1 M_2 \equiv N \equiv_{1\mathfrak{A}} N'$, N' can only be one of the two following cases.

(3.1) $N' \equiv N_1' M_2$ for some term N_1' such that $N_1 \equiv_{1\mathfrak{A}} N_1'$.

Since $M_1 \equiv_{1\alpha} N_1 \equiv_{1\mathfrak{A}} N_1'$, by induction we have $M_1 \equiv_{1\mathfrak{A}} M_1' \equiv_\alpha N_1'$ for some term M_1'. Then $M \equiv M_1 M_2 \equiv_{1\mathfrak{A}} M_1' M_2 \equiv_\alpha N_1' M_2 \equiv N'$. Choose $M' \equiv M_1' M_2$.

(3.2) $N' \equiv N_1 M_2'$ for some term M_2' such that $M_2 \equiv_{1\mathfrak{A}} M_2'$.

Then $M \equiv M_1 M_2 \equiv_{1\mathfrak{A}} M_1 M_2' \equiv_{1\alpha} N_1 M_2' \equiv N'$. Choose $M' \equiv M_1 M_2'$.

Case 4. $M \equiv (\lambda L.M_1 \mid A)$ for some pattern L, some term M_1, and some abstraction A.

This can be proved in the same way as Case 3. □

Corollary 3.1.16. *Let M, N, and N' be terms. If $M \equiv_\alpha N \equiv_{\mathfrak{A}} N'$ then $M \equiv_{\mathfrak{A}} M' \equiv_\alpha N'$ for some term M'.*

Proof. This follows directly from Lemma 3.1.15. □

Lemma 3.1.17. *Let $\underline{x} = x_1, \ldots, x_k, k \geq 1$, be distinct variables and M, N, and $\underline{U} = U_1, \ldots, U_k$ be terms. If $M \equiv_{1\mathfrak{A}} N$ then $[\underline{U}/\underline{x}]M \equiv_{1\mathfrak{A}} N'$ for some term N' such that $N' \equiv_\alpha [\underline{U}/\underline{x}]N$.*

Proof. Assume $M \equiv_{1\mathfrak{A}} N$. If $\{\underline{x}\} \cap FV(M) = \varnothing$, by Proposition 3.1.8 $\{\underline{x}\} \cap FV(N) = \varnothing$, so $[\underline{U}/\underline{x}]M \equiv M \equiv_{1\mathfrak{A}} N \equiv [\underline{U}/\underline{x}]N$, and we are finished. Now assume $\{\underline{x}\} \cap FV(M) \neq \varnothing$, and in fact, by Corollary 2.2.1(a) we may assume that $\{\underline{x}\} \subseteq FV(M)$. We will induct on M. Note that since $M \equiv_{1\mathfrak{A}} N$ and $FV(M) \neq \varnothing$, by Note 3.1.2(d) M is not atomic.

Case 1. $M \equiv \lambda P.M_1$ for some pattern P and some term M_1.

Then $N \equiv \lambda P.N_1$ for some term N_1 such that $M_1 \equiv_{1\mathfrak{A}} N_1$. Then by induction $[\underline{U}/\underline{x}]M_1 \equiv_{1\mathfrak{A}} N_1' \equiv_\alpha [\underline{U}/\underline{x}]N_1$ for some term N_1'. Let $m = \left|FV(P) \cap FV(U_1 \ldots U_k)\right|$ and induct on m. If $m = 0$, then

$$\begin{aligned}[\underline{U}/\underline{x}]M &\equiv \lambda P.[\underline{U}/\underline{x}]M_1 \\ &\equiv_{1\mathfrak{A}} \lambda P.N_1' \\ &\equiv_\alpha \lambda P.[\underline{U}/\underline{x}]N_1 \\ &\equiv [\underline{U}/\underline{x}]\lambda P.N_1 \\ &\equiv [\underline{U}/\underline{x}]N.\end{aligned}$$

Now assume $m > 0$. Let y be the first variable in $FV(P) \cap FV(U_1 \ldots U_k)$ and z be the first variable which is not in $FV(PM_1\underline{U})$. Note that z is also the first variable which is not in $FV(PN_1\underline{U})$ since $M_1 \equiv_{1\mathfrak{A}} N_1$, so $FV(M_1) \equiv FV(N_1)$. By the main induction hypothesis, $[z/y]M_1 \equiv_{1\mathfrak{A}} N_1'' \equiv_\alpha [z/y]N_1$ for some term N_1''. Then by the subsidiary induction hypothesis, $[\underline{U}/\underline{x}]\lambda[z/y]P.[z/y]M_1 \equiv_{1\mathfrak{A}} N' \equiv_\alpha [\underline{U}/\underline{x}]\lambda[z/y]P.N_1''$ for some term

N'. Hence

$$\begin{aligned}[\underline{U}/\underline{x}]M &\equiv [\underline{U}/\underline{x}]\lambda[z/y]P.[z/y]M_1 \\ &\equiv_{1\mathfrak{A}} N' \\ &\equiv_\alpha [\underline{U}/\underline{x}]\lambda[z/y]P.N_1'' \\ &\equiv_\alpha [\underline{U}/\underline{x}]\lambda[z/y]P.[z/y]N_1 \\ &\equiv [\underline{U}/\underline{x}]\lambda P.N_1 \\ &\equiv [\underline{U}/\underline{x}]N\end{aligned}$$

Case 2. $M \equiv M_1M_2$ for some terms M_1 and M_2.

Then $N \equiv N_1N_2$ for some terms N_1 and N_2. Without loss of generality, assume $M_1 \equiv_{1\mathfrak{A}} N_1$ and $M_2 \equiv N_2$. By induction we have $[\underline{U}/\underline{x}]M_1 \equiv_{1\mathfrak{A}} N_1' \equiv_\alpha [\underline{U}/\underline{x}]N_1$ for some term N_1'. Hence

$$\begin{aligned}[\underline{U}/\underline{x}]M &\equiv [\underline{U}/\underline{x}]M_1[\underline{U}/\underline{x}]M_2 \\ &\equiv_{1\mathfrak{A}} N_1'[\underline{U}/\underline{x}]M_2 \\ &\equiv_\alpha [\underline{U}/\underline{x}]N_1[\underline{U}/\underline{x}]M_2 \\ &\equiv [\underline{U}/\underline{x}]N\end{aligned}$$

The case where M is a compound abstraction is similar. □

Corollary 3.1.18. *Let $\underline{x} = x_1, \ldots, x_k, k \geq 1$, be distinct variables and M, N, and $\underline{U} = U_1, \ldots, U_k$ be terms. If $M \equiv_{\mathfrak{A}} N$ then $[\underline{U}/\underline{x}]M \equiv_{\mathfrak{A}} N'$ for some term N' such that $N' \equiv_\alpha [\underline{U}/\underline{x}]N$.*

Proof. This follows from Corollary 3.1.16 and Lemma 3.1.17. □

3.2 Contractions and Reductions

Most of the definitions and lemmas in this section are based on the lambda calculus with patterns [5] with some adjustments. Most of the lemmas are unaffected by the new congruence $\equiv_{\mathfrak{A}}$ and for these proofs will not be given. Only a few need some small changes in the statement or proof; for these we will show the details of those parts that differ. Again, for the ease of reading, the corresponding result number from [5] will be included in brackets.

Definition 3.2.1. For any pattern P with k free variables $\underline{x} = x_1, \ldots, x_k, k \geq 1$ (respectively P has no free variables), and any term N, if there exist terms $\underline{N} = N_1, \ldots, N_k$ such that $[\underline{N}/\underline{x}]P \equiv N$ (respectively $P \equiv N$), then for any term Q, $(\lambda P.Q)N$ is called a ***β*-redex** and the corresponding term $[\underline{N}/\underline{x}]Q$ (respectively Q) is called its ***β*-contractum**.

Let R be an occurrence of a β-redex in a term M. If we replace R by its β-contractum, and the result is the expression M', then we say M ***β*-contracts** to M', which we denote by $M \triangleright_{1\beta} M'$.

We extend the definitions of reductions by adding the new congruence '$\equiv_{\mathfrak{A}}$'.

Definition 3.2.2. For any terms M and M', we say M ***β*-reduces** to M', denoted by $M \triangleright_{\beta} M'$, if there exists a sequence of terms $M \equiv M_1, \ldots, M_n \equiv M', n \geq 1$, such that for each $1 \leq i < n$, $M_i \triangleright_{1\beta} M_{i+1}$, $M_i \equiv_{\alpha} M_{i+1}$, or $M_i \equiv_{\mathfrak{A}} M_{i+1}$.

Definition 3.2.3. Let $(\lambda P.Q \mid A)$ be a compound abstraction and N a term with m free variables $\underline{y} = y_1, \ldots, y_m, m \geq 1$ (respectively N has no free variables). We will call $(\lambda P.Q \mid A)N$ a ***γ*-redex** with ***γ*-contractum** S if one of the following two conditions holds:

a. the term $(\lambda P.Q)N$ is a β-redex, in which case $S \equiv (\lambda P.Q)N$; or

b. for all terms $\underline{U} = U_1, \ldots, U_m$ and all terms N' such that $[\underline{U}/\underline{y}]N \rhd_\beta N'$ (respectively $N \rhd_\beta N'$), the term $(\lambda P.Q)N'$ is not a β-redex, in which case $S \equiv AN$.

Let R be an occurrence of a γ-redex in a term M. If we replace R by its γ-contractum, and the result is the expression M', then we say M **$\boldsymbol{\gamma}$-contracts** to M', which we denote by $M \rhd_{1\gamma} M'$.

Definition 3.2.4. For any terms M and M', we say M **$\boldsymbol{\beta\gamma}$-reduces** to M', denoted by $M \rhd_{\beta\gamma} M'$, if there exists a sequence of terms $M \equiv M_1, \ldots, M_n \equiv M', n \geq 1$, such that for each $1 \leq i < n$, $M_i \rhd_{1\beta} M_{i+1}, M_i \rhd_{1\gamma} M_{i+1}$, $M_i \equiv_\alpha M_{i+1}$, or $M_i \equiv_{\mathfrak{A}} M_{i+1}$.

Definition 3.2.5. Let $(\lambda P.Q \mid A)$ be a compound abstraction and N a term with m free variables $\underline{y} = y_1, \ldots, y_m, m \geq 1$ (respectively N has no free variables). We will call $(\lambda P.Q \mid A)N$ a **$\boldsymbol{\delta}$-redex** with **$\boldsymbol{\delta}$-contractum** S if one of the following two conditions holds:

a. the term $(\lambda P.Q)N$ is a β-redex, in which case $S \equiv (\lambda P.Q)N$; or

b. for all terms $\underline{U} = U_1, \ldots, U_m$ and all terms N' such that $[\underline{U}/\underline{y}]N \rhd_{\beta\gamma} N'$ (respectively $N \rhd_{\beta\gamma} N'$), the term $(\lambda P.Q)N'$ is not a β-redex, in which case $S \equiv AN$.

Let R be an occurrence of a δ-redex in a term M. If we replace R by its δ-contractum, and the result is the expression M', then we say M **$\boldsymbol{\delta}$-contracts** to M', which we denote by $M \rhd_{1\delta} M'$.

Definition 3.2.6. For any terms M and M', we say M **$\boldsymbol{\beta\delta}$-reduces** to M', denoted by $M \rhd_{\beta\delta} M'$, if there exists a sequence of terms $M \equiv M_1, \ldots, M_n \equiv M', n \geq 1$, such that for each $1 \leq i < n$, $M_i \rhd_{1\beta} M_{i+1}$, $M_i \rhd_{1\delta} M_{i+1}$, $M_i \equiv_\alpha M_{i+1}$, or $M_i \equiv_{\mathfrak{A}} M_{i+1}$. We call such a sequence of terms a **$\boldsymbol{\beta\delta}$-reduction**.

Definition 3.2.7. For any abstraction A and any term N, AN is called a **potential redex**.

Definition 3.2.8. For any potential redex R, R is called a **contractible redex** if R is either a β-redex or a δ-redex.

Note 3.2.9. For any terms M and M', if $M \triangleright_{\beta\delta} M'$ and M contains no abstraction, then $M \equiv_{\mathfrak{A}} M'$.

Remark. Unless explicitly specified otherwise, a "reduction" means a "$\beta\delta$-reduction".

Notation. The expression $M \triangleright_{1\beta 1\delta} N$ will mean "$M \triangleright_{1\beta} N$ or $M \triangleright_{1\delta} N$".

Note 3.2.10. [**Note 2.3.14**] For any terms M and N, if $M \triangleright_{1\beta 1\delta} N$ and R is the occurence of a potential redex which is contracted when $M \triangleright_{1\beta 1\delta} N$, then

a. if $M \equiv M_1 M_2$ and $M \not\equiv R$ then $N \equiv N_1 N_2$ for some terms N_1 and N_2 such that either $M_1 \triangleright_{1\beta 1\delta} N_1$ and $M_2 \equiv N_2$ or $M_1 \equiv N_1$ and $M_2 \triangleright_{1\beta 1\delta} N_2$;

b. if $M \equiv \lambda P.Q$ then $N \equiv \lambda P.Q'$ for some term Q' such that $Q \triangleright_{1\beta 1\delta} Q'$;

c. if $M \equiv (\lambda P.Q \mid A)$ then $N \equiv (\lambda P.Q' \mid A')$ for some term Q' and some abstraction A' such that either $Q \triangleright_{1\beta 1\delta} Q'$ and $A \equiv A'$ or $Q \equiv Q'$ and $A \triangleright_{1\beta 1\delta} A'$.

Corollary 3.2.11. [**Corollary 2.3.15**] *For any term M, if $M \triangleright_{\beta\delta} N$, then N is a term and*

a. if $M \equiv M_1 M_2$ and $M \triangleright_{\beta\delta} N$ by a sequence of terms $M \equiv K_1, \ldots, K_n \equiv N$, $n \geq 1$, such that for each $1 \leq i < n$, K_i is not the potential redex which is contracted and $K_i \not\equiv^0_{1\mathfrak{A}} K_{i+1}$ then $N \equiv N_1 N_2$ for some terms N_1 and N_2 such that $M_i \triangleright_{\beta\delta} N_i$, $i = 1, 2$;

b. if $M \equiv \lambda P.Q$, and no variable in P has been changed when $M \triangleright_{\beta\delta} N$ then $N \equiv \lambda P.Q'$ for some term Q' such that $Q \triangleright_{\beta\delta} Q'$;

c. if $M \equiv (\lambda P.Q \mid A)$ *then* $N \equiv (\lambda P'.Q' \mid A')$ *for some abstractions* $\lambda P'.Q'$ *and* A' *such that* $\lambda P.Q \rhd_{\beta\delta} \lambda P'.Q'$ *and* $A \rhd_{\beta\delta} A'$.

Proof. This follows from Lemma 2.2.2, Notes 3.1.2, and Notes 3.2.10, . □

Lemma 3.2.12. [Lemma 3.1.1] *Let* $\underline{x} = x_1, \ldots, x_k, k \geq 1$*, be distinct variables,* $\underline{N} = N_1, \ldots, N_k$ *be terms, and* P *be a pattern. If* $[\underline{N}/\underline{x}]P$ *is a potential redex, then* $P \equiv x_t$ *for some* $1 \leq t \leq k$.

Lemma 3.2.13. *Let* $\underline{x} = x_1, \ldots, x_k, k \geq 1$*, be distinct variables,* $\underline{N} = N_1, \ldots, N_k$ *be terms,* P *be a pattern, and* S *be a potential redex. If* S *is in* $[\underline{N}/\underline{x}]P$*, then* S *is in* N_t *for some* $1 \leq t \leq k$.

Proof. Assume S is in $[\underline{N}/\underline{x}]P$. We will induct on P. Note that since a pattern cannot contain an abstraction, $\{\underline{x}\} \cap FV(P) \neq \varnothing$, otherwise S is in $[\underline{N}/\underline{x}]P \equiv P$, a contradiction. In fact, by Corollary 2.2.1(a) we may assume that $\{\underline{x}\} \subseteq FV(P)$.

Case 1. $P \equiv x_1$.

Then $[\underline{N}/\underline{x}]P \equiv N_1$. So S is in N_1 and we are finished.

Case 2. $P \equiv P_1 P_2$.

Then $[\underline{N}/\underline{x}]P \equiv [\underline{N}/\underline{x}]P_1[\underline{N}/\underline{x}]P_2$. Since P is not a variable, by Lemma 3.2.12, any substitution of P is not a potential redex. Hence S is either in $[\underline{N}/\underline{x}]P_1$ or $[\underline{N}/\underline{x}]P_2$. In either case, by induction S is in N_t for some $1 \leq t \leq k$. □

Lemma 3.2.14. [Lemma 3.1.2] *Let* $\lambda P.Q$ *be a simple abstraction with* $FV(P) = \{x_1, \ldots, x_k\}$*,* $k \geq 1$*, and* N *be a term such that* $\lambda P.Q \equiv_\alpha N$*. Then* $N \equiv \lambda[y_1/x_1, \ldots, y_k/x_k]P.Q'$ *for some distinct variables* $y_1, \ldots, y_k$ *and some term* Q' *such that* $\{y_1, \ldots, y_k\} \cap FV(\lambda P.Q) = \varnothing$ *and* $Q' \equiv_\alpha [y_1/x_1, \ldots, y_k/x_k]Q$.

Lemma 3.2.15. [Lemma 3.1.4] *Let P and P' be patterns with $FV(P) \subseteq \{x_1, \ldots, x_k\}$, $k \geq 1$, and $P' \equiv [y_1/x_1, \ldots, y_k/x_k]P$ for some distinct variables $y_1, \ldots, y_k$ and let Q and N be terms. If $(\lambda P.Q)N$ is a β-redex, then $(\lambda P'.Q')[U_1/u_1, \ldots, U_m/u_m]N$ is also a β-redex for any distinct variables $u_1, \ldots, u_m$, $m \geq 1$, and any terms $Q', U_1, \ldots, U_m$.*

Lemma 3.2.16. [Lemma 3.1.6] *Let $R \equiv (\lambda P.Q)N$ be a β-redex, $\underline{x} = x_1, \ldots, x_k$, $k \geq 1$, be distinct variables, and $S, \underline{U} = U_1, \ldots, U_k$ be terms. If $R \triangleright_{1\beta} S$, then $[\underline{U}/\underline{x}]R \triangleright_{\beta} [\underline{U}/\underline{x}]S$. To be precise, if $R \triangleright_{1\beta} S$, then $[\underline{U}/\underline{x}]R \triangleright_{1\beta} S^*$ for some term S^*, where $S^* \equiv_{\alpha} [\underline{U}/\underline{x}]S$.*

Lemma 3.2.17. [Lemma 3.1.7] *Let $R \equiv (\lambda P.Q \mid A)N$ be a δ-redex, $\underline{x} = x_1, \ldots, x_k, k \geq 1$, be distinct variables, and $\underline{U} = U_1, \ldots, U_k$ be terms. If $R \triangleright_{1\delta} S$, then $[\underline{U}/\underline{x}]R \triangleright_{1\delta} [\underline{U}/\underline{x}]S$.*

Lemma 3.2.18. [Corollary 3.1.8] *Let $\underline{x} = x_1, \ldots, x_k, k \geq 1$, be distinct variables and M, M', $\underline{U} = U_1, \ldots, U_k$ be terms.*

a. If $M \triangleright_{\beta\delta} M'$, then $[\underline{U}/\underline{x}]M \triangleright_{\beta\delta} [\underline{U}/\underline{x}]M'$.

b. If R is a contractible redex, then so is $[\underline{U}/\underline{x}]R$.

Lemma 3.2.19. [Lemma 3.1.9] *Let A be an abstraction, A' and N be terms such that $A \triangleright_{1\beta 1\delta} A'$. If AN is a contractible redex, then so is $A'N$.*

Lemma 3.2.20. [Lemma 3.1.10] *Let P be a pattern with $FV(P) = \{x_1, \ldots, x_k\}$, $k \geq 1$, and $N, \underline{U} = U_1, \ldots, U_k$ be terms. Let $\underline{x} = x_1, \ldots, x_k$. If $[\underline{U}/\underline{x}]P \triangleright_{\beta\delta} N$, then $N \equiv_{\mathfrak{A}} [\underline{V}/\underline{x}]P$ for some terms $\underline{V} = V_1, \ldots, V_k$ such that $U_i \triangleright_{\beta\delta} V_i$ for all $1 \leq i \leq k$.*

Proof. Assume $[\underline{U}/\underline{x}]P \triangleright_{\beta\delta} N$. Induct on P.

Case 1. $P \equiv x_1$.

Let $V_1 \equiv N$. Then $N \equiv V_1 \equiv [V_1/x_1]P$ and $U_1 \equiv [U_1/x_1]P \triangleright_{\beta\delta} N \equiv V_1$.

Case 2. $P \equiv P_1P_2$.

Let $[\underline{U}/\underline{x}]P \triangleright_{\beta\delta} N$ by a sequence of terms $[\underline{U}/\underline{x}]P \equiv K_1, \ldots, K_n \equiv N$, $n \geq 1$.

(2.1) $K_i \not\equiv^0_{1\mathfrak{A}} K_{i+1}$ for all $1 \leq i < n$.

By Lemma 3.2.12, any substitution of P is not a potential redex. Since $[\underline{U}/\underline{x}]P_1[\underline{U}/\underline{x}]P_2 \equiv [\underline{U}/\underline{x}]P \triangleright_{\beta\delta} N$, by Corollary 3.2.11, $N \equiv N_1N_2$ for some terms N_1 and N_2, where $[\underline{U}/\underline{x}]P_i \triangleright_{\beta\delta} N_i, i = 1, 2$. Since $FV(P) = \{x_1, \ldots, x_k\}$, $FV(P_1) \neq \varnothing$ or $FV(P_2) \neq \varnothing$. Without loss of generality, assume $FV(P_1) \neq \varnothing$.

(2.1.1) $FV(P_2) = \varnothing$.

Then $FV(P_1) = \{x_1, \ldots, x_k\}$. Since $[\underline{U}/\underline{x}]P_1 \triangleright_{\beta\delta} N_1$, by induction $N_1 \equiv_{\mathfrak{A}} [\underline{V}/\underline{x}]P_1$ for some terms $\underline{V} = V_1, \ldots, V_k$, where $U_i \triangleright_{\beta\delta} V_i$ for all $1 \leq i \leq k$. Since $FV(P_2) \equiv \varnothing$, $P_2 \equiv [\underline{U}/\underline{x}]P_2 \triangleright_{\beta\delta} N_2$. In fact $P_2 \equiv_{\mathfrak{A}} N_2$, since P_2 contains no abstractions. Hence

$$\begin{aligned} N &\equiv N_1N_2 \\ &\equiv_{\mathfrak{A}} ([\underline{V}/\underline{x}]P_1)P_2 \\ &\equiv [\underline{V}/\underline{x}]P_1[\underline{V}/\underline{x}]P_2 \\ &\equiv [\underline{V}/\underline{x}](P_1P_2) \\ &\equiv [\underline{V}/\underline{x}]P. \end{aligned}$$

(2.1.2) $FV(P_2) = \{x_{j_1}, \ldots, x_{j_p}\}$.

Since $FV(P) = \{x_1, \ldots, x_k\}$ and no variable occurs in both P_1 and P_2, $FV(P_1) = \{x_{i_1}, \ldots, x_{i_m}\}$, where $\{i_1, \ldots, i_m\} \cup \{j_1, \ldots, j_p\} = \{1, \ldots, k\}$ and $\{i_1, \ldots, i_m\} \cap \{j_1, \ldots, j_p\} = \varnothing$. By Corollary 2.2.1(a), $[\underline{U}/\underline{x}]P_1 \equiv [U_{i_1}/x_{i_1}, \ldots, U_{i_m}/x_{i_m}]P_1$ and $[\underline{U}/\underline{x}]P_2 \equiv$ $[U_{j_1}/x_{j_1}, \ldots, U_{j_p}/x_{j_p}]P_2$. By induction, $N_1 \equiv_{\mathfrak{A}} [V_{i_1}/x_{i_1}, \ldots, V_{i_m}/x_{i_m}]P_1$ and $N_2 \equiv_{\mathfrak{A}} [V_{j_1}/x_{j_1}, \ldots, V_{j_p}/x_{j_p}]P_2$ for some terms $V_{i_1}, \ldots, V_{i_m}$,

$V_{j_1}, \ldots, V_{j_p}$, where $U_r \rhd_{\beta\delta} V_r$ for all $1 \leq r \leq k$. Let $\underline{V} = V_1, \ldots, V_k$. Hence

$$
\begin{aligned}
N &\equiv N_1 N_2 \\
&\equiv_{\mathfrak{A}} [V_{i_1}/x_{i_1}, \ldots, V_{i_m}/x_{i_m}]P_1[V_{j_1}/x_{j_1}, \ldots, V_{j_p}/x_{j_p}]P_2 \\
&\equiv [\underline{V}/\underline{x}]P_1[\underline{V}/\underline{x}]P_2 \\
&\equiv [\underline{V}/\underline{x}]P.
\end{aligned}
$$

(2.2) $K_i \equiv^0_{1\mathfrak{A}} K_{i+1}$ for some $1 \leq i < n$.

Let k be the first such i. Then $[\underline{U}/\underline{x}]P \rhd_{\beta\delta} K_k$ with $K_j \not\equiv^0_{1\mathfrak{A}} K_{j+1}$ for all $1 \leq j < k$. By Case 2.1 we have $K_k \equiv_{\mathfrak{A}} [\underline{V}/\underline{x}]P$ for some terms $\underline{V} = V_1, \ldots, V_k$ such that $U_j \rhd_{\beta\delta} V_j$ for all $1 \leq j \leq k$. Since $K_k \equiv^0_{1\mathfrak{A}} K_{k+1}$, K_k contains no abstraction. Then, since $K_k \rhd_{\beta\delta} N$, by Note 3.2.9, $K_k \equiv_{\mathfrak{A}} N$. Hence $N \equiv_{\mathfrak{A}} K_k \equiv_{\mathfrak{A}} [\underline{V}/\underline{x}]P$. □

Lemma 3.2.21. [Lemma 3.1.12] *Let A be an abstraction, and N be a term such that AN is a contractible redex.*

a. For any term N' such that $N \equiv_\alpha N'$, AN' is a contractible redex.

b. For any term N' such that $N \rhd_{1\beta 1\delta} N'$, AN' is a contractible redex.

Proof. Both are special cases of Lemma 3.1.12 in [5]. □

Lemma 3.2.22. [Lemma 3.1.12] *Let A be an abstraction, and N and N' be terms such that $N \rhd_{\beta\delta} N'$. If AN is a contractible redex, then there exists a term N'' such that $N' \equiv_{\mathfrak{A}} N''$ and AN'' is a contractible redex.*

Proof. Assume AN is contractible. Let $N \rhd_{\beta\delta} N'$ by a sequence of terms $N \equiv N_1, \ldots, N_k \equiv N'$, $k \geq 1$. We will induct on k. If $k = 1$ then $N \equiv N_1 \equiv N'$, so AN' is contractible and we are finished. Now assume $k > 1$. Then by induction there exists a term N'_{k-1} such that $N_{k-1} \equiv_{\mathfrak{A}} N'_{k-1}$ and AN'_{k-1} is a contractible redex.

Case 1. $N_{k-1} \equiv_{\mathfrak{A}} N'$.

Since $N'_{k-1} \equiv_{\mathfrak{A}} N_{k-1} \equiv_{\mathfrak{A}} N'$, and AN'_{k-1} is contractible, we are finished with $N'' \equiv N'_{k-1}$.

Case 2. $N_{k-1} \equiv_{\alpha} N'$.

Since $N'_{k-1} \equiv_{\mathfrak{A}} N_{k-1} \equiv_{\alpha} N'$, by Corollary 3.1.16, $N'_{k-1} \equiv_{\alpha} N''_{k-1} \equiv_{\mathfrak{A}} N'$ for some term N''_{k-1}. Then, since AN'_{k-1} is contractible, by Lemma 3.2.21(a), so is AN''_{k-1}. Thus we are finished with $N'' \equiv N''_{k-1}$.

Case 3. $N_{k-1} \triangleright_{1\beta 1\delta} N'$.

Then N_{k-1} contains an abstraction. Since $N_{k-1} \equiv_{\mathfrak{A}} N'_{k-1}$, by Note 3.1.4(a), N'_{k-1} also contains an abstraction.

(3.1) $A \equiv \lambda P.Q.$ for some pattern P and some term Q.

(3.1.1) $FV(P) = \varnothing$.

Since $(\lambda P.Q)N'_{k-1}$ is contractible, $P \equiv N'_{k-1}$. This is a contradiction since a pattern cannot contain an abstraction.

(3.1.2) $FV(P) = \{x_1, \ldots, x_n\}$ for some variables $\underline{x} = x_1, \ldots, x_n$.

Since $(\lambda P.Q)N'_{k-1}$ is contractible, $[\underline{U}/\underline{x}]P \equiv N'_{k-1}$ for some terms $\underline{U} = U_1, \ldots, U_n$. Since $[\underline{U}/\underline{x}]P \equiv N'_{k-1} \equiv_{\mathfrak{A}} N_{k-1} \triangleright_{1\beta 1\delta} N'$, by Lemma 3.2.20 $N' \equiv_{\mathfrak{A}} [\underline{V}/\underline{x}]P$ for some terms $\underline{V} = V_1, \ldots, V_n$. Let $N'' \equiv [\underline{V}/\underline{x}]P$. Then $N' \equiv_{\mathfrak{A}} N''$ and $AN'' \equiv (\lambda P.Q)([\underline{V}/\underline{x}]P)$ is contractible.

(3.2) $A \equiv (\lambda P.Q \mid B)$ for some pattern P, some term Q, and some abstraction B.

If AN_{k-1} is contractible, since $N_{k-1} \rhd_{1\beta 1\delta} N'$, by Lemma 3.2.21(b), AN' is contractible and we are finished. Now suppose $(\lambda P.Q \mid B)N_{k-1}$ is not contractible. Then since $N'_{k-1} \equiv_{\mathfrak{A}} N_{k-1}$, by Corollary 3.1.18, $(\lambda P.Q \mid B)N'_{k-1} \not\rhd_{1\delta} BN'_{k-1}$. Since $(\lambda P.Q \mid B)N'_{k-1}$ is contractible, $(\lambda P.Q)N'_{k-1}$ is contractible, and the proof can be finished much like in Case 3.1. □

Lemma 3.2.23. [Lemma 3.1.13] *Let R be a contractible redex, and R' and S be terms such that $R \equiv_\alpha R'$. If $R \rhd_{1\beta} S$ (respectively $R \rhd_{1\delta} S$), then $R' \rhd_{1\beta} S'$ (respectively $R' \rhd_{1\delta} S'$) for some term S', where $S' \equiv_\alpha S$.*

Proof. We use Lemma 3.2.21(a) instead of Lemma 3.1.12 in the original proof in [5]. The rest of the proof remains unchanged. □

Lemma 3.2.24. [Corollary 3.1.14]

a. *Let M, M', and N be terms such that $M \equiv_\alpha M'$. If $M \rhd_{1\beta} N$ (respectively $M \rhd_{1\delta} N$), then $M' \rhd_{1\beta} N'$ (respectively $M' \rhd_{1\delta} N'$) for some term N', where $N' \equiv_\alpha N$.*

b. *If R is a contractible redex and R' is a term such that $R \equiv_\alpha R'$, then R' is also a contractible redex.*

3.3 Computability Relative to a Structure

Definition 3.3.1. Let g be an n-ary function on $|\mathfrak{A}|$. We say g is **computable relative to $\mathfrak{A}$** if and only if there is a term G, using only variables and symbols in $\mathcal{L}$ together with T and F, such that for all $a_1, \ldots, a_n, a \in |\mathfrak{A}|$, we have

$$G\bar{a}_1 \ldots \bar{a}_n \rhd_{\beta\delta} \bar{a}$$

whenever $g(a_1, \ldots, a_n) = a$.

Definition 3.3.2. Let r be an n-ary relation on $|\mathfrak{A}|$. We say r is **computable relative to** $\mathfrak{A}$ if and only if there is a term R, using only variables and symbols in $\mathcal{L}$ together with T and F, such that for all $a_1, \ldots, a_n \in |\mathfrak{A}|$, we have

$$R\bar{a}_1 \ldots \bar{a}_n \rhd_{\beta\delta} \mathsf{T} \text{ if } (a_1, \ldots, a_n) \in r, \text{ and}$$

$$R\bar{a}_1 \ldots \bar{a}_n \rhd_{\beta\delta} \mathsf{F} \text{ otherwise.}$$

CHAPTER IV

THE CHURCH-ROSSER THEOREM

4.1 Minimal Complete Developments

The definition for *minimal complete development (MCD)* is slightly modified from the original one to allow the new congruence $\equiv_{\mathfrak{A}}$.

Definition 4.1.1. Let R and S be occurrences of contractible redexes in a term M. When R is contracted, let M change to M'.

The **contraction-residuals** of S (with respect to R) are occurrences of potential redexes in M', defined as follows.

Case 1. R and S are non-overlapping parts of M.

Then contracting R leaves S unchanged. This unchanged S in M' is the contraction-residual of S.

Case 2. $R \equiv S$

Then contracting R is the same as contracting S. We say S has no contraction-residuals in M'.

Case 3. R is part of S and $R \not\equiv S$.

Since S is a potential redex, $S \equiv AN$ for some abstraction A, and some term N. So R is either in A or in N. Then contracting R changes S to S', where $S' \equiv A'N'$ for some abstraction A' and some term N' such that either $A \rhd_{1\beta 1\delta} A'$ and $N \equiv N'$ or $A \equiv A'$ and $N \rhd_{1\beta 1\delta} N'$. This S' is the contraction-residual of S.

Case 4. S is part of R and $S \not\equiv R$.

There are cases and subcases as follows.

(4.1) $R \equiv (\lambda P.Q)N$.

(4.1.1) $FV(P) = \varnothing$.

Since R is a β-redex, $P \equiv N$ and $R \triangleright_{1\beta} Q$. Since S is a potential redex in R, S is in Q. Since $R \triangleright_{1\beta} Q$, contracting R leaves S unchanged in M'; this is the contraction-residual of S.

(4.1.2) $FV(P) = \{x_1, \ldots, x_k\}, k \geq 1$.

Then $[\underline{N}/\underline{x}]P \equiv N$ for some terms $\underline{N} = N_1, \ldots, N_k$ and $R \triangleright_{1\beta} [\underline{N}/\underline{x}]Q$.

(4.1.2.1) S is in Q.

Then S changes to S', where S' is either S or some substitution of S. This S' is the contraction-residual of S.

(4.1.2.2) S is in N.

Then S is in $[\underline{N}/\underline{x}]P$. By Lemma 3.2.13, S is in N_t for some $1 \leq t \leq k$. Hence there is an occurrence of S in each N_t substituted for an occurrence of x_t in Q. These are the contraction-residuals of S. (Note that S may have many or no contraction-residuals.)

(4.2) $R \equiv (\lambda P.Q \mid A)N$.

(4.2.1) $R \triangleright_{1\delta} (\lambda P.Q)N$.

If S is in Q or N, then contracting R leaves S unchanged, and this is the contraction-residual of S in M'. If S is in A, then S has no contraction-residuals in M'.

(4.2.2) $R \triangleright_{1\delta} AN$.

If S is in A or N, then this unchanged S in A or N is the contraction-residual of S in M'. If S is in Q, then S has no contraction-residuals in M'.

Note 4.1.2.

a. Except in the Case 4.1.2.2, S has at most one contraction-residual.

b. Each contraction-residual is a contractible redex. (The contraction-residual in Case 3 is contractible by Lemmas 3.2.19 and 3.2.21(b), and the contraction-residual in Case 4.1.2.1 is contractible by Corollary 3.2.18(b)).

Definition 4.1.3. Let R be an occurrence of a contractible redex in a term M. The $1\mathfrak{A}$**-conversion-residuals** of R (with respect to M) when $M \equiv_{1\mathfrak{A}} M'$ are occurrences of potential redexes in M', defined inductively as follows. Note that since M contains an abstraction, $M \not\equiv^0_{1\mathfrak{A}} M'$, so M and M' are of the same form.

Case 1. $M \equiv R$.

If M' is a contractible redex then this M' is the $1\mathfrak{A}$-conversion-residual of R, otherwise R has no $1\mathfrak{A}$-conversion-residuals in M'.

Case 2. $M \not\equiv R$.

(2.1) R is unchanged in M'.

This unchanged R is the $1\mathfrak{A}$-conversion-residual of R.

(2.2) R is changed in M'.

(2.2.1) $M \equiv M_1M_2$ for some term M_1 and M_2.

Then R is in M_i for some $i \in \{1, 2\}$. The $1\mathfrak{A}$-conversion-residual of R with respect to M_i is the $1\mathfrak{A}$-conversion-residual of R with respect to M.

(2.2.2) $M \equiv \lambda P.N$ for some pattern P and some term N.

Then R is in N. The $1\mathfrak{A}$-conversion-residual of R with respect to N is the $1\mathfrak{A}$-conversion-residual of R with respect to M.

(2.2.3) $M \equiv (A_1 \mid A_2)$ for some abstractions A_1 and A_2.

Then R is in A_i for some $i \in \{1, 2\}$. The $1\mathfrak{A}$-conversion-residual of R with respect to A_i is the $1\mathfrak{A}$-conversion-residual of R with respect to M.

Note 4.1.4.

a. R has at most one $1\mathfrak{A}$-conversion-residual.

b. Each $1\mathfrak{A}$-conversion-residual is a contractible redex.

Remark. We may simply use "residual" to abbreviate either a "contraction-residual" or a "$1\mathfrak{A}$-conversion-residual", where there is no ambiguity.

Definition 4.1.5. If $\mathscr{R} = \{R_i \mid 1 \leq i \leq n\}, n \geq 0$, is a set of occurrences of potential redexes in a term M, then an R_i is called **minimal** (with respect to $\mathscr{R}$) if it properly contains no other $R_j \in \mathscr{R}$. (Note that if $n = 0$ then $\mathscr{R} = \varnothing$, i.e., M contains no potential redex.)

Let $\mathscr{R}_M = \{R_i \mid 1 \leq i \leq n\}, n \geq 0$, be a set of occurrences of contractible redexes in a term M. For any terms M^* and M' such that $M^* \equiv_{\mathfrak{A}} M$, we say M' is obtained from M^* by a **minimal complete development (MCD)** of $\mathscr{R}_M$, denoted by $M^* \rhd_{mcd} M'$ (of $\mathscr{R}_M$), if M' is obtained from M by the following process.

First contract any minimal R_i; without loss of generality let $i = 1$. By Definition 4.1.1, this leaves $n-1$ contraction-residuals, $R'_2, R'_3, \ldots, R'_n$. Then make as many $1\mathfrak{A}$-conversions as you like (possibly none), this leaves at most $n-1$ $1\mathfrak{A}$-conversion-residuals among $R''_2, R''_3, \ldots, R''_n$. Again, contract any minimal R''_t and make $1\mathfrak{A}$-conversions. This leaves at most $n-2$ residuals. Repeat this process until no contraction-residuals are left. Then make as many $1\mathfrak{A}$-conversions as you like. Finally, make as many α-steps as you like.

Note 4.1.6. [Note 3.2.4]

a. Each MCD is a $\beta\delta$-reduction.

b. For any contractible redex L, if $L \triangleright_{mcd} M$ of $\mathscr{R}_L$, without α-steps, where $L \notin \mathscr{R}_L$, and $M \triangleright_{1\beta 1\delta} N$, with M being the potential redex contracted, then $L \triangleright_{mcd} N$ of $\mathscr{R}_L \cup \{L\}$, without α-steps. In fact, for any term L' such that $L' \equiv_{\mathfrak{A}} L$, $L' \triangleright_{mcd} N$ of $\mathscr{R}_L \cup \{L\}$, without α-steps.

Proposition 4.1.7. *Let M, N and N' be terms. If $M \triangleright_{mcd} N \equiv_{\mathfrak{A}} N'$ then $M \triangleright_{mcd} N'$.*

Proof. This follows directly from Corollary 3.1.16. □

Lemma 4.1.8. [Lemma 3.2.5] *For any term M, if $M \triangleright_{mcd} N$, then N is a term and*

a. if $M \equiv M_1M_2$ and $M \triangleright_{mcd} N$ by a sequence of terms $M \equiv K_1, \ldots, K_n \equiv N$, $n \geq 1$, such that for each $1 \leq i < n$, K_i is not the potential redex which is contracted and $K_i \not\equiv^0_{1\mathfrak{A}} K_{i+1}$ then $N \equiv N_1N_2$ for some terms N_1 and N_2 such that $M_i \triangleright_{mcd} N_i, i = 1, 2$;

b. if $M \equiv \lambda P.Q$, and no variable in P has been changed when $M \triangleright_{mcd} N$ then $N \equiv \lambda P.Q'$ for some term Q' such that $Q \triangleright_{mcd} Q'$;

c. if $M \equiv (\lambda P.Q \mid A)$ then $N \equiv (\lambda P.Q' \mid A')$ for some abstractions $\lambda P'.Q'$ and A' such that $\lambda P.Q \triangleright_{mcd} \lambda P'.Q'$ and $A \triangleright_{mcd} A'$.

Proof. This follows from Notes 3.1.2, 3.2.10, and Lemma 2.2.2. □

Lemma 4.1.9. [Lemma 3.2.6] *Let P be a pattern with $FV(P) = \{x_1, \dots, x_k\}, k \geq 1$, and N, $\underline{U} = U_1, \dots, U_k$ be terms. Let $\underline{x} = x_1, \dots, x_k$. If $[\underline{U}/\underline{x}]P \triangleright_{mcd} N$, then $N \equiv_{\mathfrak{A}} [\underline{V}/\underline{x}]P$ for some terms $\underline{V} = V_1, \dots, V_k$ such that $U_i \triangleright_{mcd} V_i$ for all $1 \leq i \leq k$.*

Proof. This can be proved in the same way as Lemma 3.2.20. $\square$

Lemma 4.1.10. [Lemma 3.2.7] *For any terms M, N, and M', if $M \triangleright_{mcd} N$ and $M \equiv_\alpha M'$, then $M' \triangleright_{mcd} N$.*

Lemma 4.1.11. [Lemma 3.2.8] *For any distinct variables $\underline{x} = x_1, \dots, x_k, k \geq 1$, and any terms M, N, $\underline{U} = U_1, \dots, U_k, \underline{V} = V_1, \dots, V_k$, if $M \triangleright_{mcd} N$ and $U_i \triangleright_{mcd} V_i$ for all $1 \leq i \leq k$, then $[\underline{U}/\underline{x}]M \triangleright_{mcd} [\underline{V}/\underline{x}]N$.*

Proof. As in the original proof of these two lemmas (Lemma 3.2.7 and 3.2.8 in [5]), they are proved simultaneously by induction on M, and additionally we may assume that the MCD $M \triangleright_{mcd} N$ has no α-steps and $\{\underline{x}\} \subseteq FV(M)$. The proof remains unchanged except for the case where $M \equiv M_1 M_2$, which is rewritten as follows. Let $M \triangleright_{mcd} N$ of $\mathscr{R}$ by a sequence of terms $M \equiv K_1, \dots, K_n \equiv N$, $n \geq 1$.

Case 1. $K_i \not\equiv^0_{1\mathfrak{A}} K_{i+1}$ for all $1 \leq i < n$.

(1.1) $M \notin \mathscr{R}$.

This can be proved in the same way as the case when M is a compound abstraction (See Case iii. of the original proof in [5]).

(1.2) $M \in \mathscr{R}$.

Since $M \in \mathscr{R}$ and $M \triangleright_{mcd} N$, without α-steps, $M \triangleright_{mcd} M_1^0 M_2^0$ for some terms M_1^0 and M_2^0 such that $M_1 \triangleright_{mcd} M_1^0$ and $M_2 \triangleright_{mcd} M_2^0$, both without α-steps, and $M_1^0 M_2^0 \triangleright_{1\beta 1\delta} N^0$ with $M_1^0 M_2^0$ being the potential redex contracted, for some term N^0 such that $N^0 \equiv_{\mathfrak{A}} N$.

Proof of 4.1.10.

Since $M \equiv_\alpha M'$, we have that $M' \equiv M_1'M_2'$ for some terms M_1' and M_2' such that $M_i' \equiv_\alpha M_i, i = 1, 2$. By induction, $M_i' \triangleright_{mcd} M_i^0, i = 1, 2$. Hence $M_1' \triangleright_{mcd} M_1^*$ and $M_2' \triangleright_{mcd} M_2^*$, both without α-steps, for some terms M_1^* and M_2^*, where $M_i^* \equiv_\alpha M_i^0, i = 1, 2$. Since $M_1^*M_2^* \equiv_\alpha M_1^0M_2^0$ and $M_1^0M_2^0 \triangleright_{1\beta 1\delta} N^0$, by Lemma 3.2.23 $M_1^*M_2^* \triangleright_{1\beta 1\delta} M^*$ for some term M^*, where $M^* \equiv_\alpha N^0$. Hence $M' \equiv M_1'M_2' \triangleright_{mcd} M_1^*M_2^* \triangleright_{1\beta 1\delta} M^* \equiv_\alpha N^0$. Since $M \equiv_\alpha M'$, by Corollary 3.2.24(b) M' is contractible. By Note 4.1.6(b), $M' \triangleright_{mcd} N^0$. Since $N^0 \equiv_{\mathfrak{A}} N$, by Proposition 4.1.7 $M' \triangleright_{mcd} N$.

Proof of 4.1.11.

Since $M_1 \triangleright_{mcd} M_1^0$ and $M_2 \triangleright_{mcd} M_2^0$, by induction $[\underline{U}/\underline{x}]M_i \triangleright_{mcd} [\underline{V}/\underline{x}]M_i^0, i = 1, 2$. Hence $[\underline{U}/\underline{x}]M_i \triangleright_{mcd} M_i^*$, without α-steps, for some term M_i^* such that $M_i^* \equiv_\alpha [\underline{V}/\underline{x}]M_i^0, i = 1, 2$. Since $M_1^0M_2^0 \triangleright_{1\beta 1\delta} N^0$, by Lemmas 3.2.16 and 3.2.17 $[\underline{V}/\underline{x}](M_1^0M_2^0) \triangleright_{1\beta 1\delta} N^*$ for some term N^*, where $N^* \equiv_\alpha [\underline{V}/\underline{x}]N^0$. Since $M_1^*M_2^* \equiv_\alpha [\underline{V}/\underline{x}](M_1^0M_2^0)$, by Lemma 3.2.23 $M_1^*M_2^* \triangleright_{1\beta 1\delta} M^*$ for some term M^* such that $M^* \equiv_\alpha N^*$.

$$
\begin{aligned}
\text{Hence } [\underline{U}/\underline{x}]M &\equiv [\underline{U}/\underline{x}]M_1[\underline{U}/\underline{x}]M_2 \\
&\triangleright_{mcd} M_1^*M_2^* \\
&\triangleright_{1\beta 1\delta} M^* \\
&\equiv_\alpha N^* \\
&\equiv_\alpha [\underline{V}/\underline{x}]N^0.
\end{aligned}
$$

Since M is contractible, by Corollary 3.2.18(b), $[\underline{U}/\underline{x}]M$ is contractible. By Note 4.1.6(b) $[\underline{U}/\underline{x}]M \rhd_{mcd} [\underline{V}/\underline{x}]N^0$. Since $N^0 \equiv_{\mathfrak{A}} N$, by Proposition 3.1.18 $[\underline{V}/\underline{x}]N^0 \equiv_{\mathfrak{A}} N'$ for some term N' such that $N' \equiv_\alpha [\underline{V}/\underline{x}]N$. Then by Proposition 4.1.7 $[\underline{U}/\underline{x}]M \rhd_{mcd} [\underline{V}/\underline{x}]N$.

Case 2. $K_i \equiv^0_{1\mathfrak{A}} K_{i+1}$ for some $1 \leq i < n$.

Let k be the first such i. Then $M \rhd_{mcd} K_k$ with $K_j \not\equiv^0_{1\mathfrak{A}} K_{j+1}$ for all $1 \leq j < k$. Since $K_k \equiv^0_{1\mathfrak{A}} K_{k+1}$, K_k contains no abstractions. Then, since $K_k \rhd_{mcd} N$, it must be that $K_k \equiv_{\mathfrak{A}} N$.

Proof of 4.1.10.

Since $M \equiv_\alpha M'$, by Case 1 $M' \rhd_{mcd} K_k$. Then by Proposition 4.1.7 $M' \rhd_{mcd} N$.

Proof of 4.1.11.

By Case 1 we have $[\underline{U}/\underline{x}]M \rhd_{mcd} [\underline{V}/\underline{x}]K_k$. Since $K_k \equiv_{\mathfrak{A}} N$, by Proposition 3.1.18 $[\underline{V}/\underline{x}]K_k \equiv_{\mathfrak{A}} N'$ for some term N' such that $N' \equiv_\alpha [\underline{V}/\underline{x}]N$. Then by Proposition 4.1.7 $[\underline{U}/\underline{x}]M \rhd_{mcd} [\underline{V}/\underline{x}]N$.

□

4.2 The Church-Rosser Theorem for $\beta\delta$-Reduction

We first prove the Church-Rosser theorem for MCD's, where most of the work is done, then use it to prove the Church-Rosser theorem for $\beta\delta$-reduction.

Theorem 4.2.1. ***(The Church-Rosser Theorem for MCD's)***
For any terms L, M, and N, if $L \rhd_{mcd} M$ and $L \rhd_{mcd} N$, then there exists a term T such that $M \rhd_{mcd} T$ and $N \rhd_{mcd} T$.

Proof. Let L, M, and N be terms such that $L \rhd_{mcd} M$ and $L \rhd_{mcd} N$. Then M (respectively N) is obtained from L by the given MCD of a set $\mathscr{R}_M$ (respectively

$\mathscr{R}_N$). By Lemma 4.1.10, it is sufficient to consider the case in which the given MCD's have no α-steps. Induct on L.

i. L is an atom.

Since $L \triangleright_{mcd} M$ and $L \triangleright_{mcd} N$, it must be that $M \equiv_{\mathfrak{A}} L \equiv_{\mathfrak{A}} N$ and we are finished by choosing $T \equiv M$.

ii. $L \equiv \lambda P.Q$.

Since $L \triangleright_{mcd} M$ and $L \triangleright_{mcd} N$, both without α-steps, $M \equiv \lambda P.Q^M$ and $N \equiv \lambda P.Q^N$ for some terms Q^M and Q^N such that $Q \triangleright_{mcd} Q^M$ and $Q \triangleright_{mcd} Q^N$. By induction, there exists a term Q^* such that $Q^M \triangleright_{mcd} Q^*$ and $Q^N \triangleright_{mcd} Q^*$. Let $T \equiv \lambda P.Q^*$. Then $M \equiv \lambda P.Q^M \triangleright_{mcd} \lambda P.Q^* \equiv T$ and, similarly, $N \triangleright_{mcd} T$.

iii. $L \equiv (\lambda P.Q \mid A)$

Since $L \triangleright_{mcd} M$ and $L \triangleright_{mcd} N$, both without α-steps, $M \equiv (\lambda P.Q^M \mid A^M)$ and $N \equiv (\lambda P.Q^N \mid A^N)$ for some terms Q^M and Q^N and some abstractions A^M and A^N such that $Q \triangleright_{mcd} Q^M$, $Q \triangleright_{mcd} Q^N$, $A \triangleright_{mcd} A^M$, and $A \triangleright_{mcd} A^N$. By induction, there exist terms Q^* and A^* such that $Q^M \triangleright_{mcd} Q^*$, $Q^N \triangleright_{mcd} Q^*$, $A^M \triangleright_{mcd} A^*$, and $A^N \triangleright_{mcd} A^*$. By Lemma 4.1.8, A^* is also an abstraction. Let $T \equiv (\lambda P.Q^* \mid A^*)$. Then $M \equiv (\lambda P.Q^M \mid A^M) \triangleright_{mcd} (\lambda P.Q^* \mid A^*) \equiv T$ and, similarly, $N \triangleright_{mcd} T$.

iv. $L \equiv L_1 L_2$

Case 1. $L \notin \mathscr{R}_M$ and $L \notin \mathscr{R}_N$.

This case can be proved in the same way as (iii).

Case 2. $L \in \mathscr{R}_M$ or $L \in \mathscr{R}_N$.

Without loss of generality, assume that $L \in \mathscr{R}_M$. There are cases and subcases as follows.

(2.1) $L_1 \equiv \lambda P.Q$.

Since $L \in \mathscr{R}_M$ and $(\lambda P.Q)L_2 \equiv L \triangleright_{mcd} M$, without α-steps, $L \triangleright_{mcd}$

$(\lambda P.Q^M)L_2^M$ for some terms Q^M and L_2^M such that $Q \triangleright_{mcd} Q^M$, $L_2 \triangleright_{mcd} L_2^M$, and $(\lambda P.Q^M)L_2^M \triangleright_{1\beta} M^0 \equiv_{\mathfrak{A}} M$, with $(\lambda P.Q^M)L_2^M$ being the β-redex contracted, for some term M^0.

(2.1.1) $L \in \mathscr{R}_N$.

Similar to the above, $L \triangleright_{mcd} (\lambda P.Q^N)L_2^N$ for some terms Q^N and L_2^N, such that $Q \triangleright_{mcd} Q^N$, $L_2 \triangleright_{mcd} L_2^N$, and $(\lambda P.Q^N)L_2^N \triangleright_{1\beta} N^0 \equiv_{\mathfrak{A}} N$, with $(\lambda P.Q^N)L_2^N$ being the β-redex contracted, for some term N^0. By induction, there exist terms Q^* and L_2^* such that $Q^M \triangleright_{mcd} Q^*$, $Q^N \triangleright_{mcd} Q^*$, $L_2^M \triangleright_{mcd} L_2^*$, and $L_2^N \triangleright_{mcd} L_2^*$.

(2.1.1.1) $FV(P) = \varnothing$.

Since $(\lambda P.Q^M)L_2^M \triangleright_{1\beta} M^0$ and $(\lambda P.Q^N)L_2^N) \triangleright_{1\beta} N^0$, $M^0 \equiv Q^M$ and $N^0 \equiv Q^N$. Hence $M \equiv_{\mathfrak{A}} M^0 \equiv Q^M \triangleright_{mcd} Q^*$. So $M \triangleright_{mcd} Q^*$. Similarly for N, we have $N \triangleright_{mcd} Q^*$. So we are finished with $T \equiv Q^*$.

(2.1.1.2) $FV(P) = \{x_1, \ldots, x_k\}$.

Since $(\lambda P.Q^M)L_2^M \triangleright_{1\beta} M^0$ and $(\lambda P.Q^N)L_2^N \triangleright_{1\beta} N^0$, there exist terms $\underline{U} = U_1, \ldots, U_k, \underline{V} = V_1, \ldots, V_k$ such that $[\underline{U}/\underline{x}]P \equiv L_2^M$, $[\underline{V}/\underline{x}]P \equiv L_2^N$, $M^0 \equiv [\underline{U}/\underline{x}]Q^M$, and $N^0 \equiv [\underline{V}/\underline{x}]Q^N$. Since $L_2^M \triangleright_{mcd} L_2^*$ and $L_2^N \triangleright_{mcd} L_2^*$, by Lemma 4.1.9 $L_2^* \equiv_{\mathfrak{A}} [\underline{U'}/\underline{x}]P$ and $L_2^* \equiv_{\mathfrak{A}} [\underline{V'}/\underline{x}]P$ for some terms $\underline{U'} = U_1', \ldots, U_k'$, $\underline{V'} = V_1', \ldots, V_k'$ such that $U_i \triangleright_{mcd} U_i'$, and $V_i \triangleright_{mcd} V_i'$ for all $1 \leq i \leq k$. Since $[\underline{U'}/\underline{x}]P \equiv_{\mathfrak{A}} L_2^* \equiv_{\mathfrak{A}} [\underline{V'}/\underline{x}]P$, by Lemma 3.1.13, for each $1 \leq i \leq k$, $U_i' \equiv_{\mathfrak{A}} V_i'$. Then $U_i \triangleright_{mcd} U_i'$ and since $V_i \triangleright_{mcd} V_i' \equiv_{\mathfrak{A}} U_i'$, by Proposition 4.1.7, $V_i \triangleright_{mcd} U_i'$ for all $1 \leq i \leq k$. Let $\underline{W} = U_1', \ldots, U_k'$. Thus by Lemma 4.1.11 $M \equiv_{\mathfrak{A}} M^0 \equiv [\underline{U}/\underline{x}]Q^M \triangleright_{mcd} [\underline{W}/\underline{x}]Q^*$ and $N \equiv_{\mathfrak{A}} N^0 \equiv [\underline{V}/\underline{x}]Q^N \triangleright_{mcd} [\underline{W}/\underline{x}]Q^*$. So we are finished with $T \equiv [\underline{W}/\underline{x}]Q^*$.

(2.1.2) $L \notin \mathscr{R}_N$.

Since $(\lambda P.Q)L_2 \equiv L \triangleright_{mcd} N$, without α-steps, $N \equiv_{\mathfrak{A}} (\lambda P.Q^N)L_2^N$ for some terms Q^N and L_2^N such that $Q \triangleright_{mcd} Q^N$ and $L_2 \triangleright_{mcd} L_2^N$. By induction, there exist terms Q^* and L_2^* such that $Q^N \triangleright_{mcd} Q^*$ and $L_2^N \triangleright_{mcd} L_2^*$, both without α-steps, and $Q^M \triangleright_{mcd} Q^*$ and $L_2^M \triangleright_{mcd} L_2^*$.

(2.1.2.1) $FV(P) = \varnothing$.

Since $L \equiv (\lambda P.Q)L_2$ is contractible, $P \equiv L_2$. Then L_2 contains no bound variables. Since $L_2 \triangleright_{mcd} L_2^N$, actually $L_2 \equiv_{\mathfrak{A}} L_2^N$, and

$$
\begin{aligned}
N &\equiv_{\mathfrak{A}} (\lambda P.Q^N)L_2^N \\
&\equiv_{\mathfrak{A}} (\lambda P.Q^N)L_2 \\
&\triangleright_{mcd} (\lambda P.Q^*)L_2 \\
&\triangleright_{1\beta} Q^*.
\end{aligned}
$$

Since $(\lambda P.Q^N)L_2$ is contractible, by Note 4.1.6(b) $N \triangleright_{mcd} Q^*$. Also, since $(\lambda P.Q^M)L_2^M \triangleright_{1\beta} M^0$, $M \equiv_{\mathfrak{A}} M^0 \equiv Q^M \triangleright_{mcd} Q^*$, so we are finished with $T \equiv Q^*$.

(2.1.2.2) $FV(P) = \{x_1, \ldots, x_k\}$.

Since $(\lambda P.Q^M)L_2^M \triangleright_{1\beta} M^0$, there exist terms $\underline{U} = U_1, \ldots, U_k$ such that $[\underline{U}/\underline{x}]P \equiv L_2^M$ and $M^0 \equiv [\underline{U}/\underline{x}]Q^M$. Since $L_2^M \triangleright_{mcd} L_2^*$, by Lemma 4.1.9, $L_2^* \equiv_{\mathfrak{A}} [\underline{V}/\underline{x}]P$ for some terms $\underline{V} = V_1, \ldots, V_k$ such that $U_i \triangleright_{mcd} V_i$ for all $1 \leq i \leq k$. Since $L \equiv (\lambda P.Q)L_2$ is contractible, there exist terms $\underline{W} = W_1, \ldots, W_k$ such that $[\underline{W}/\underline{x}]P \equiv L_2$. Then since $L_2 \triangleright_{mcd} L_2^N$, by Lemma 4.1.9, $L_2^N \equiv_{\mathfrak{A}} [\underline{W'}/\underline{x}]P$ for some terms $\underline{W'} = W'_1, \ldots, W'_k$. Again since $L_2^N \triangleright_{mcd} L_2^*$, we have $L_2^* \equiv_{\mathfrak{A}} [\underline{W''}/\underline{x}]P$ for some terms

$\underline{W}'' = W_1'', \ldots, W_k''$ such that $W_i' \rhd_{mcd} W_i''$ for all $1 \leq i \leq k$. Since $[\underline{W}''/\underline{x}]P \equiv_{\mathfrak{A}} L_2^* \equiv_{\mathfrak{A}} [\underline{V}/\underline{x}]P$, by Lemma 3.1.13, $W_i'' \equiv_{\mathfrak{A}} V_i$ for all $1 \leq i \leq k$. Then

$$
\begin{aligned}
N &\equiv (\lambda P.Q^N)L_2^N \\
&\equiv_{\mathfrak{A}} (\lambda P.Q^N)([\underline{W}'/\underline{x}]P) \\
&\rhd_{mcd} (\lambda P.Q^*)([\underline{W}''/\underline{x}]P) \\
&\equiv_{\mathfrak{A}} (\lambda P.Q^*)([\underline{V}/\underline{x}]P) \\
&\rhd_{1\beta} [\underline{V}/\underline{x}]Q^*.
\end{aligned}
$$

Since $(\lambda P.Q^N)([\underline{W}'/\underline{x}]P)$ is contractible, by Note 4.1.6(b) and Proposition 4.1.7, $N \rhd_{mcd} [\underline{V}/\underline{x}]Q^*$. Also, by Lemma 4.1.11, we have $M \equiv_{\mathfrak{A}} M^0 \equiv [\underline{U}/\underline{x}]Q^M \rhd_{mcd} [\underline{V}/\underline{x}]Q^*$, so we are finished with $T \equiv [\underline{V}/\underline{x}]Q^*$.

(2.2) $L_1 \equiv (\lambda P.Q \mid A)$.

Since $L \in \mathscr{R}_M$ and $(\lambda P.Q \mid A)L_2 \equiv L \rhd_{mcd} M$, without α-steps, $L \rhd_{mcd} (\lambda P.Q^M \mid A^M)L_2^M$ for some terms Q^M and L_2^M and some abstraction A^M such that $Q \rhd_{mcd} Q^M$, $A \rhd_{mcd} A^M$, $L_2 \rhd_{mcd} L_2^M$, and $(\lambda P.Q^M \mid A^M)L_2^M \rhd_{1\delta} M^0 \equiv_{\mathfrak{A}} M$, with $(\lambda P.Q^M \mid A^M)L_2^M$ being the δ-redex contracted, for some term M^0.

(2.2.1) $L \in \mathscr{R}_N$.

Similar to the above, $L \rhd_{mcd} (\lambda P.Q^N \mid A^N)L_2^N$ for some terms Q^N and L_2^N and some abstraction A^N such that $Q \rhd_{mcd} Q^N$, $A \rhd_{mcd} A^N$, $L_2 \rhd_{mcd} L_2^N$, and $(\lambda P.Q^N \mid A^N)L_2^N \rhd_{1\delta} N^0 \equiv_{\mathfrak{A}} N$, with $(\lambda P.Q^N \mid A^N)L_2^N$ being the δ-redex contracted, for some term N^0. By induction, there exist terms Q^*, A^* and L_2^* such that $Q^M \rhd_{mcd} Q^*$, $Q^N \rhd_{mcd} Q^*$, $A^M \rhd_{mcd} A^*$, $A^N \rhd_{mcd} A^*$, $L_2^M \rhd_{mcd} L_2^*$, and $L_2^N \rhd_{mcd} L_2^*$.

(2.2.1.1) $(\lambda P.Q^M \mid A^M)L_2^M \triangleright_{1\delta} (\lambda P.Q^M)L_2^M$.

Then $(\lambda P.Q^M)L_2^M$ is a β-redex and $M^0 \equiv (\lambda P.Q^M)L_2^M$. Since $L_2^M \triangleright_{mcd} L_2^*$, by Lemmas 3.2.15 and 3.2.22 and Note 4.1.6(a) there exists a term L_2^0 such that $L_2^* \equiv_{\mathfrak{A}} L_2^0$ and $(\lambda P.Q^N)L_2^0$ is a β-redex. Since $L_2^N \triangleright_{mcd} L_2^* \equiv_{\mathfrak{A}} L_2^0$, we have that $L_2^N \triangleright_{\beta\delta} L_2^0$, and so $L_2^N \triangleright_{\beta\gamma} L_2^0$. Hence $(\lambda P.Q^N \mid A^N)L_2^N \not\triangleright_{1\delta} A^N L_2^N$. Since $(\lambda P.Q^N \mid A^N)L_2^N \triangleright_{1\delta} N^0$, it must be that $N^0 \equiv (\lambda P.Q^N)L_2^N$. Thus $M \equiv_{\mathfrak{A}} M^0 \equiv (\lambda P.Q^M)L_2^M \triangleright_{mcd} (\lambda P.Q^*)L_2^*$ and $N \equiv_{\mathfrak{A}} N^0 \equiv (\lambda P.Q^N)L_2^N \triangleright_{mcd} (\lambda P.Q^*)L_2^*$ so we are finished with $T \equiv (\lambda P.Q^*)L_2^*$.

(2.2.1.2) $(\lambda P.Q^M \mid A^M)L_2^M \triangleright_{1\delta} A^M L_2^M$.

Suppose $(\lambda P.Q^N)L_2^N$ is a β-redex. Since $L_2^N \triangleright_{mcd} L_2^*$, an argument similar to the one above shows that $(\lambda P.Q^M)L_2^0$ is a β-redex for some term L_2^0 such that $L_2^* \equiv_{\mathfrak{A}} L_2^0$. Since $L_2^M \triangleright_{mcd} L_2^* \equiv_{\mathfrak{A}} L_2^0$, we have that $L_2^M \triangleright_{\beta\delta} L_2^0$, and thus $L_2^M \triangleright_{\beta\gamma} L_2^0$. Hence $(\lambda P.Q^M \mid A^M)L_2^M \not\triangleright_{1\delta} A^M L_2^M$, a contradiction. So $(\lambda P.Q^N)L_2^N$ is not a β-redex. Since $(\lambda P.Q^N \mid A^N)L_2^N \triangleright_{1\delta} N$, $N \equiv A^N L_2^N$. Thus $M \equiv A^M L_2^M \triangleright_{mcd} A^* L_2^*$ and $N \equiv A^N L_2^N \triangleright_{mcd} A^* L_2^*$ so we are finished with $T \equiv A^* L_2^*$.

(2.2.2) $L \notin \mathscr{R}_N$.

Since $(\lambda P.Q \mid A)L_2 \equiv L \triangleright_{mcd} N$, without α-steps, $N \equiv_{\mathfrak{A}} (\lambda P.Q^N \mid A^N)L_2^N$ for some terms Q^N and L_2^N and some abstraction A^N such that $Q \triangleright_{mcd} Q^N, A \triangleright_{mcd} A^N$, and $L_2 \triangleright_{mcd} L_2^N$. By induction, there exist terms Q^*, A^*, and L_2^*, such that $Q^N \triangleright_{mcd} Q^*, A^N \triangleright_{mcd} A^*$, and $L_2^N \triangleright_{mcd} L_2^*$, all without α-steps, and $Q^M \triangleright_{mcd} Q^*, A^M \triangleright_{mcd} A^*$, and $L_2^M \triangleright_{mcd} L_2^*$. Note that A^* is an abstraction by Lemmas 3.2.14 and

4.1.8.

(2.2.2.1) $(\lambda P.Q^M \mid A^M)L_2^M \rhd_{1\delta} (\lambda P.Q^M)L_2^M$.

Then $(\lambda P.Q^M)L_2^M$ is a β-redex and $M^0 \equiv (\lambda P.Q^M)L_2^M$. Since $L_2^M \rhd_{mcd} L_2^*$, there exists a term L_2^0 such that $L_2^* \equiv_{\mathfrak{A}} L_2^0$ and $(\lambda P.Q^N)L_2^0$ is a β-redex. So

$$\begin{aligned} M &\equiv_{\mathfrak{A}} M^0 \\ &\equiv (\lambda P.Q^M)L_2^M \\ &\rhd_{mcd} (\lambda P.Q^*)L_2^* \\ &\equiv_{\mathfrak{A}} (\lambda P.Q^*)L_2^0 \end{aligned}$$

and

$$\begin{aligned} N &\equiv_{\mathfrak{A}} (\lambda P.Q^N \mid A^N)L_2^N \\ &\rhd_{mcd} (\lambda P.Q^* \mid A^*)L_2^* \\ &\equiv_{\mathfrak{A}} (\lambda P.Q^* \mid A^*)L_2^0 \\ &\rhd_{1\delta} (\lambda P.Q^*)L_2^0. \end{aligned}$$

Then by Proposition 4.1.7,$M \rhd_{mcd} (\lambda P.Q^*)L_2^0$ and $N \rhd_{mcd} (\lambda P.Q^*)L_2^0$, so we are finished with $T \equiv (\lambda P.Q^*)L_2^0$.

(2.2.2.2) $(\lambda P.Q^M \mid A^M)L_2^M \rhd_{1\delta} A^M L_2^M$.

Suppose $(\lambda P.Q^* \mid A^*)L_2^* \not\rhd_{1\delta} A^* L_2^*$. Then $(\lambda P.Q^*)L_2^+$ is a β-redex for some term L_2^+ such that $[\underline{U}/\underline{x}]L_2^* \rhd_{\beta\gamma} L_2^+$ for some distinct variables $\underline{x} = x_1, \ldots, x_k, k \geq 1$, and some terms $\underline{U} = U_1, \ldots, U_k$. Since $L_2^M \rhd_{mcd} L_2^*$, we have that $L_2^M \rhd_{\beta\delta} L_2^*$. By Corollary 3.2.18(a), $[\underline{U}/\underline{x}]L_2^M \rhd_{\beta\delta} [\underline{U}/\underline{x}]L_2^*$, so $[\underline{U}/\underline{x}]L_2^M \rhd_{\beta\gamma} [\underline{U}/\underline{x}]L_2^*$. Since the relation $\rhd_{\beta\gamma}$ is transitive, $[\underline{U}/\underline{x}]L_2^M \rhd_{\beta\gamma} L_2^+$. Since $(\lambda P.Q^*)L_2^+$ is a β-redex, $(\lambda P.Q^M)L_2^+$ is also a β-redex. Hence

$(\lambda P.Q^M \mid A^M)L_2^M \not\triangleright_{1\delta} A^M L_2^M$, a contradiction. Thus $(\lambda P.Q^* \mid A^*)L_2^* \triangleright_{1\delta} A^* L_2^*$. Hence $M \equiv_{\mathfrak{A}} M^0 \equiv A^M L_2^M \triangleright_{mcd} A^* L_2^*$ and $N \equiv (\lambda P.Q^N \mid A^N)L_2^N \triangleright_{mcd} (\lambda P.Q^* \mid A^*)L_2^* \triangleright_{1\delta} A^* L_2^*$ so we are finished with $T \equiv A^* L_2^*$. □

Theorem 4.2.2. ***(The Church-Rosser Theorem for $\beta\delta$-reduction)*** *For any terms L, M, and N, if $L \triangleright_{\beta\delta} M$ and $L \triangleright_{\beta\delta} N$, then there exists a term T such that $M \triangleright_{\beta\delta} T$ and $N \triangleright_{\beta\delta} T$.*

Proof. Using the fact that our new $\triangleright_{\beta\delta}$ allows $\equiv_{\mathfrak{A}}$, and a single $\equiv_{\mathfrak{A}}$ is an $\triangleright_{mcd}$, the proof remains the same as in [5]. □

4.3 $\boldsymbol{\beta\delta}$-Normal Form and $\boldsymbol{\beta\delta}$-Equality

Definition 4.3.1. For any abstraction A and any term N, AN is called a **contractible redex** if AN is either a β-redex or a δ-redex.

Definition 4.3.2. A term M which contains no contractible redexes is called a **$\boldsymbol{\beta\delta}$-normal form**. For any terms M and N, if $M \triangleright_{\beta\delta} N$ and N is a $\beta\delta$-normal form, then N is called a **$\boldsymbol{\beta\delta}$-normal form** of M.

Lemma 4.3.3. [**Lemma 3.1.15**] *For any $\beta\delta$-normal form M and any term N, if $M \triangleright_{\beta\delta} N$, then $M \equiv_{\mathfrak{A}} M' \equiv_{\alpha} N$ for some term M'.*

Proof. The essence of the proof remains unchanged. By inducting on the length of the sequence of reduction, we show that there is no $\triangleright_{1\beta 1\delta}$ in the sequence. Then by Corollary 3.1.16, we can rearrange $\equiv_{\mathfrak{A}}$'s and $\equiv_{\alpha}$'s in any order. □

Corollary 4.3.4. [**Corollary 3.3.3**] *For any term L, if L has $\beta\delta$-normal forms M and N, then $M \equiv_{\mathfrak{A}} M' \equiv_{\alpha} N$ for some term M'.*

Proof. Let L, M, and N be terms such that $L \triangleright_{\beta\delta} M$ and $L \triangleright_{\beta\delta} N$, and M and N are $\beta\delta$-normal forms. By Theorem 4.2.2, there exists a term T such that $M \triangleright_{\beta\delta} T$ and $N \triangleright_{\beta\delta} T$. Then by Lemma 4.3.3, $M \equiv_{\mathfrak{A}} M' \equiv_{\alpha} T \equiv_{\alpha} N' \equiv_{\mathfrak{A}} N$ for some terms M' and N'. So by Corollary 3.1.16, we have $M \equiv_{\mathfrak{A}} T' \equiv_{\alpha} N$ for some term T'. □

All other theorems and lemmas about $\beta\delta$-normal forms and $\beta\delta$-equality in [5] can be stated and proved in a similar fashion, by using the fact that the new $\triangleright_{\beta\delta}$ allows $\equiv_{\mathfrak{A}}$, and a single $\equiv_{\mathfrak{A}}$ is an $\triangleright_{mcd}$. The proofs may require some minor modifications to accommodate the new congruence $\equiv_{\mathfrak{A}}$.

CHAPTER V

RECURSIVENESS AND COMPUTABILITY RELATIVE TO A STRUCTURE

To justify the word "computable", we need to show that our new computability relative to a structure is equivalent to recursiveness. Given the standard structure $\mathfrak{N} = (\mathbb{N}, \{\mathsf{S}^{\mathfrak{N}}\}, \{\mathsf{0}^{\mathfrak{N}}\})$ for the language of natural numbers $\mathcal{L} = \{\mathsf{S}, \mathsf{0}\}$, we will show that every total recursive function on $\mathbb{N}$ is **computable relative to** $\mathfrak{N}$. The proof of the converse will be discussed in the next chapter. We begin by first reviewing the definition of recursive function.

5.1 Recursive Functions

The definitions concerning recursive functions in this section are summarized from [3].

Definition 5.1.1. The following functions are called **initial functions**.

i. The **zero function**, $g(a) = 0$ for all $a \in \mathbb{N}$.

ii. The **successor function**, $g(a) = a + 1$ for all $a \in \mathbb{N}$.

iii. The **projection functions**, $g_i^n(a_1, \ldots, a_n) = a_i$ for all $1 \leq i \leq n$ and $a_1, \ldots, a_n \in \mathbb{N}$.

Definition 5.1.2. The function g is said to be obtained by **composition** from the functions $h(y_1, \ldots, y_m), k_1(x_1, \ldots, x_n), \ldots, k_m(x_1, \ldots, x_n)$ if

$$g(x_1, \ldots, x_n) = h(k_1(x_1, \ldots, x_n), \ldots, k_m(x_1, \ldots, x_n)).$$

Definition 5.1.3. The function g is said to be obtained by **primitive recursion** from the functions $h(x_1, \ldots, x_n)$ and $k(x_1, \ldots, x_{n+2})$ if

$$g(x_1, \ldots, x_n, 0) = h(x_1, \ldots, x_n)$$

$$g(x_1, \ldots, x_n, y+1) = k(x_1, \ldots, x_n, y, g(x_1, \ldots, x_n, y)).$$

Definition 5.1.4. If $h(x_1, \ldots, x_n, y)$ is a functions such that for any $x_1, \ldots, x_n$ there exists a y such that $h(x_1, \ldots, x_n, y) = 0$, then we denote the least y such that $h(x_1, \ldots, x_n, y) = 0$ by $\mu y(h(x_1, \ldots, x_n, y) = 0)$.

The function g is said to be obtained by **restricted minimization** from $h(x_1, \ldots, x_n, y)$ if $g(x_1, \ldots, x_n) = \mu y(h(x_1, \ldots, x_n, y) = 0)$.

Definition 5.1.5. A function is said to be **recursive** if and only if it can be obtained from the initial functions by any finite number of applications of composition, primitive recursion, and restricted minimization.

5.2 Recursiveness Implies Computability Relative to $\mathfrak{N}$

In order to show that a total recursive function on $\mathbb{N}$ is **computable relative to** $\mathfrak{N}$, we will first show that the initial functions on $\mathbb{N}$ are computable relative to $\mathfrak{N}$, and then that the above rules for obtaining new recursive functions preserve the computability relative to $\mathfrak{N}$.

5.2.1 Initial Functions

i. The **zero function**, $g(a) = 0$ for all $a \in \mathbb{N}$.

Consider the term $G \equiv \lambda x.\mathbf{0}$. Since $\mathbf{0}^{\mathfrak{N}} = 0$, for any $a \in \mathbb{N}, G\bar{a} \equiv (\lambda x.\mathbf{0})\bar{a} \triangleright_{1\beta} \mathbf{0} \equiv_{\mathfrak{N}} \bar{0}$, so we have $G\bar{a} \triangleright_{\beta\delta} \bar{0}$. Hence the zero function is computable relative to $\mathfrak{N}$. $\square$

ii. The **successor function**, $g(a) = a + 1$ for all $a \in \mathbb{N}$.

Consider the term S. Since for any $a \in \mathbb{N}, \mathsf{S}^{\mathfrak{N}}(a) = a+1$, we have that $\mathsf{S}\bar{a} \equiv_{\mathfrak{N}} \overline{a+1}$, and thus $\mathsf{S}\bar{a} \rhd_{\beta\delta} \overline{a+1}$. Hence the successor function is computable relative to $\mathfrak{N}$. $\square$

iii. The **projection function**, $g_i^n(a_1, \ldots, a_n) = a_i$ for all $a_1, \ldots, a_n \in \mathbb{N}$.

Consider the term $G \equiv \lambda x_1 \ldots . \lambda x_n . x_i$. Since for all $a_1, \ldots a_n \in \mathbb{N}$,

$$\begin{aligned} G\bar{a}_1 \ldots \bar{a}_n &\equiv (\lambda x_1 \ldots . \lambda x_n . x_i)\bar{a}_1 \ldots \bar{a}_n \\ &\rhd_{1\beta} [\bar{a}_1/x_1](\lambda x_2 \ldots . \lambda x_n . x_i)\bar{a}_2 \ldots \bar{a}_n \\ &\ \vdots \\ &\rhd_{1\beta} [\bar{a}_n/x_n] \ldots [\bar{a}_1/x_1]x_i \\ &\equiv \bar{a}_i, \end{aligned}$$

we have $G\bar{a}_1\bar{a}_2 \ldots \bar{a}_n \rhd_{\beta\delta} \bar{a}_i$. Hence the projection function is computable relative to $\mathfrak{N}$. $\square$

5.2.2 Composition

Let $g(x_1, \ldots x_n)$ be a total function on $\mathbb{N}$ obtained by composition from the functions $h(y_1, \ldots, y_m)$ and $k_i(x_1, \ldots, x_n), 1 \le i \le m$ as follows,

$$g(x_1, \ldots, x_n) = h\big(k_1(x_1, \ldots, x_n), \ldots, k_m(x_1, \ldots, x_n)\big)$$

where h and k_i are computable relative to $\mathfrak{N}$ for all $1 \le i \le m$. Then there exist terms H and K_i corresponding to the functions h and k_i for all $1 \le i \le m$ respectively. Let $a_1, \ldots, a_n \in \mathbb{N}$, and $a = g(a_1, \ldots, a_n)$. Then $h\big(k_1(a_1, \ldots, a_n), \ldots, k_m(a_1, \ldots, a_n)\big) = a$. Suppose $k_i(a_1, \ldots, a_n) = b_i$ for some $b_i \in \mathbb{N}, 1 \le i \le m$. Then $K_i\bar{a}_1 \ldots \bar{a}_n \rhd_{\beta\delta} \bar{b}_i$ for all $1 \le i \le m$. Also we have

$h(b_1, \ldots, b_m) = a$, so $H\bar{b}_1 \ldots \bar{b}_m \triangleright_{\beta\delta} \bar{a}$. Now consider the term $G \equiv \lambda x_1 . \ldots \lambda x_n . H(K_1 x_1 \ldots x_n) \ldots (K_m x_1 \ldots x_n)$. Since

$$
\begin{aligned}
G\bar{a}_1 \ldots \bar{a}_n \equiv & \big(\lambda x_1 . \ldots \lambda x_n . H(K_1 x_1 \ldots x_n) \ldots (K_m x_1 \ldots x_n)\big)\bar{a}_1 \ldots \bar{a}_n \\
& \triangleright_{1\beta} [\bar{a}_1/x_1]\big(\lambda x_2 . \ldots \lambda x_n . H(K_1 x_1 \ldots x_n) \ldots (K_m x_1 \ldots x_n)\big)\bar{a}_2 \ldots \bar{a}_n \\
& \quad \vdots \\
& \triangleright_{1\beta} [\bar{a}_n/x_n] \ldots [\bar{a}_1/x_1]\big(H(K_1 x_1 \ldots x_n) \ldots (K_m x_1 \ldots x_n)\big) \\
& \equiv H(K_1 \bar{a}_1 \ldots \bar{a}_n) \ldots (K_m \bar{a}_1 \ldots \bar{a}_n) \\
& \triangleright_{\beta\delta} H\bar{b}_1 \ldots \bar{b}_m \\
& \triangleright_{\beta\delta} \bar{a},
\end{aligned}
$$

we have $G\bar{a}_1\bar{a}_2 \ldots \bar{a}_n \triangleright_{\beta\delta} \bar{a}$. Hence composition preserves computability relative to $\mathfrak{N}$. $\square$

5.2.3 Primitive Recursion

Let $g(x_1, \ldots, x_{n+1})$ be a total function on $\mathbb{N}$ obtained by primitive recursion from the functions $h(x_1, \ldots, x_n)$ and $k(x_1, \ldots, x_{n+2})$ as follows:

$$
\begin{aligned}
g(x_1, \ldots, x_n, 0) &= h(x_1, \ldots, x_n) \\
g(x_1, \ldots, x_n, y+1) &= k(x_1, \ldots, x_n, y, g(x_1, \ldots, x_n, y)),
\end{aligned}
$$

where h and k are computable relative to $\mathfrak{N}$. Then there exist terms H and K corresponding to the functions h and k respectively. Let $a_1, \ldots, a_n, m \in \mathbb{N}$, and $a = g(a_1, \ldots, a_n, m)$. Consider the term $G \equiv \mathsf{Y}P$ where Y is a fixed-point combinator* and $P \equiv \lambda f . \lambda x_1 . \ldots \lambda x_n . \big(\lambda \mathbf{0} . H x_1 \ldots x_n \mid \lambda \mathsf{S} y . K x_1 \ldots x_n y (f x_1 \ldots x_n y)\big)$.

*A fixed-point combinator Y is a term such that $\mathsf{Y}X \triangleright_\beta X(\mathsf{Y}X)$ for any term X. An example of such term by Alan M. Turing is $\mathsf{Y}_{\text{Turing}} = ZZ$ where $Z \equiv \lambda zx . x(zzx)$. See [1].

Then for any term M we have

$$\begin{aligned} G\bar{a}_1\ldots\bar{a}_nM &\equiv \mathsf{Y}P\bar{a}_1\ldots\bar{a}_nM \rhd_\beta P(\mathsf{Y}P)\bar{a}_1\ldots\bar{a}_nM \equiv PG\bar{a}_1\ldots\bar{a}_nM \\ &\equiv \Big(\lambda f.\lambda x_1.\ldots\lambda x_n.\big(\lambda\mathbf{0}.Hx_1\ldots x_n \mid \\ &\quad \lambda\mathsf{S}y.Kx_1\ldots x_ny(fx_1\ldots x_ny)\big)\Big)G\bar{a}_1\ldots\bar{a}_nM \\ &\rhd_\beta [\bar{a}_n/x_n]\ldots[\bar{a}_1/x_1][G/f]\big(\lambda\mathbf{0}.Hx_1\ldots x_n \mid \\ &\quad \lambda\mathsf{S}y.Kx_1\ldots x_ny(fx_1\ldots x_ny)\big)M \\ &\equiv \big(\lambda\mathbf{0}.H\bar{a}_1\ldots\bar{a}_n \mid \lambda\mathsf{S}y.K\bar{a}_1\ldots\bar{a}_ny(G\bar{a}_1\ldots\bar{a}_ny)\big)M. \end{aligned}$$

If $m = 0$, since H corresponds to h and $h(a_1,\ldots,a_n) = g(a_1,\ldots,a_n,0) = a$, we have $H\bar{a}_1\ldots\bar{a}_n \rhd_{\beta\delta} \bar{a}$. Hence

$$\begin{aligned} G\bar{a}_1\ldots\bar{a}_n\bar{0} &\equiv_{\mathfrak{N}} G\bar{a}_1\ldots\bar{a}_n\mathbf{0} \\ &\rhd_\beta \big(\lambda\mathbf{0}.H\bar{a}_1\ldots\bar{a}_n \mid \lambda\mathsf{S}y.K\bar{a}_1\ldots\bar{a}_ny(G\bar{a}_1\ldots\bar{a}_ny)\big)\mathbf{0} \\ &\rhd_{1\delta} \big(\lambda\mathbf{0}.H\bar{a}_1\ldots\bar{a}_n\big)\mathbf{0} \\ &\rhd_{1\beta} H\bar{a}_1\ldots\bar{a}_n \\ &\rhd_{\beta\delta} \bar{a}, \end{aligned}$$

so we have $G\bar{a}_1\ldots\bar{a}_n\bar{0} \rhd_{\beta\delta} \bar{a}$. Suppose $m = p+1$ for some $p \in \mathbb{N}$. Let $b = g(a_1,\ldots,a_n,p)$, so $k(a_1,\ldots,a_n,p,b) = k(a_1,\ldots,a_n,p,g(a_1,\ldots,a_n,p)) = g(a_1,\ldots,a_n,p+1) = a$. Since K corresponds to k, we have $K\bar{a}_1\ldots\bar{a}_n\bar{p}\bar{b} \rhd_{\beta\delta} \bar{a}$. Also, by the induction hypothesis, we have $G\bar{a}_1 \dots \bar{a}_n \bar{p} \rhd_{\beta\delta} \bar{b}$. Now since

$$\begin{aligned}
G\bar{a}_1 \dots \bar{a}_n \bar{m} = G\bar{a}_1 \dots \bar{a}_n \overline{p+1} &\equiv_{\mathfrak{N}} G\bar{a}_1 \dots \bar{a}_n (\mathsf{S}\bar{p}) \quad \big(\because \mathsf{S}^{\mathfrak{N}}(p) = p+1\big) \\
&\rhd_{\beta} \big(\lambda \mathbf{0}.H\bar{a}_1 \dots \bar{a}_n \mid \lambda \mathsf{S}y.K\bar{a}_1 \dots \bar{a}_n y(G\bar{a}_1 \dots \bar{a}_n y)\big)(\mathsf{S}\bar{p}) \\
&\rhd_{1\delta} \big(\lambda \mathsf{S}y.K\bar{a}_1 \dots \bar{a}_n y(G\bar{a}_1 \dots \bar{a}_n y)\big)(\mathsf{S}\bar{p}) \\
&\rhd_{1\beta} [\bar{p}/y]\big(K\bar{a}_1 \dots \bar{a}_n y(G\bar{a}_1 \dots \bar{a}_n y)\big) \\
&\equiv K\bar{a}_1 \dots \bar{a}_n \bar{p}(G\bar{a}_1 \dots \bar{a}_n \bar{p}) \\
&\rhd_{\beta\delta} K\bar{a}_1 \dots \bar{a}_n \bar{p}\bar{b} \\
&\rhd_{\beta\delta} \bar{a},
\end{aligned}$$

we have $G\bar{a}_1 \dots \bar{a}_n \bar{m} \rhd_{\beta\delta} \bar{a}$. By induction, we conclude that $G\bar{a}_1 \dots \bar{a}_n \bar{m} \rhd_{\beta\delta} \bar{a}$ for all m. Hence primitive recursion preserves computability relative to $\mathfrak{N}$ $\square$

5.2.4 The Restricted Minimization

Let $g(x_1, \dots, x_n)$ be a total function on $\mathbb{N}$ obtained by restricted minimization from $h(x_1, \dots, x_n, y)$ as follows:

$$g(x_1, \dots, x_n) = \mu y(h(x_1, \dots, x_n, y) = 0),$$

where h is computable relative to $\mathfrak{N}$. Then there exists a term H corresponding to the function h. Let $a_1, \dots, a_n \in \mathbb{N}$. Let $a = g(a_1, \dots, a_n)$. Consider the term $G \equiv G'\mathbf{0}$ where $G' \equiv \mathsf{Y}P$ and

$$P \equiv \lambda f.\lambda y.\lambda x_1. \dots \lambda x_n.\big(\lambda \mathbf{0}.y \mid \lambda z.f(\mathsf{S}y)x_1 \dots x_n\big)\big(Hx_1 \dots x_n y\big).$$

Case 1. $a = 0$.

Then $h(a_1, \dots, a_n, 0) = 0$. Since H corresponds to h, we have $H\bar{a}_1 \dots \bar{a}_n \bar{0} \rhd_{\beta\delta}$

$\bar{0} \equiv_{\mathfrak{N}} \mathbf{0}$. Then

$$
\begin{aligned}
G\bar{a}_1 \dots \bar{a}_n &\equiv G'\mathbf{0}\bar{a}_1 \dots \bar{a}_n \\
&\equiv_{\mathfrak{N}} G'\bar{0}\bar{a}_1 \dots \bar{a}_n \\
&\equiv \mathsf{Y}P\bar{0}\bar{a}_1 \dots \bar{a}_n \rhd_\beta P(\mathsf{Y}P)\bar{0}\bar{a}_1 \dots \bar{a}_n \equiv PG'\bar{0}\bar{a}_1 \dots \bar{a}_n \\
&\equiv \Big(\lambda f.\lambda y.\lambda x_1.\dots\lambda x_n.\big(\lambda\mathbf{0}.y \mid \\
&\quad \lambda z.f(\mathbf{S}y)x_1 \dots x_n\big)\big(Hx_1 \dots x_n y\big)\Big)G'\bar{0}\bar{a}_1 \dots \bar{a}_n \\
&\rhd_\beta [\bar{a}_n/x_n] \dots [\bar{a}_1/x_1][\bar{0}/y][G'/f]\Big(\big(\lambda\mathbf{0}.y \mid \\
&\quad \lambda z.f(\mathbf{S}y)x_1 \dots x_n\big)\big(Hx_1 \dots x_n y\big)\Big) \\
&\equiv \big(\lambda\mathbf{0}.\bar{0} \mid \lambda z.G'(\mathbf{S}\bar{0})\bar{a}_1 \dots \bar{a}_n\big)\big(H\bar{a}_1 \dots \bar{a}_n\bar{0}\big) \\
&\rhd_{\beta\delta} \big(\lambda\mathbf{0}.\bar{0} \mid \lambda z.G'(\mathbf{S}\bar{0})\bar{a}_1 \dots \bar{a}_n\big)\mathbf{0} \\
&\rhd_{1\delta} \big(\lambda\mathbf{0}.\bar{0}\big)\mathbf{0} \\
&\rhd_{1\beta} \bar{0} = \bar{a},
\end{aligned}
$$

so we have $G\bar{a}_1 \dots \bar{a}_n \rhd_{\beta\delta} \bar{a}$.

Case 2. $a \neq 0$.

Then $a = b+1$ for some $b \in \mathbb{N}$. Since $g(a_1, \dots, a_n) = a$, we have $h(a_1, \dots, a_n, a) = 0$, and for all $x \leq b, h(a_1, \dots, a_n, x) \neq 0$. Since H corresponds to h, we have $H\bar{a}_1 \dots \bar{a}_n\bar{a} \rhd_{\beta\delta} \bar{0} \equiv_{\mathfrak{N}} \mathbf{0}$, and by the Church-Rosser theorem for $\beta\delta$-reduction,

$H\bar{a}_1 \ldots \bar{a}_n \bar{x} \not\triangleright_{\beta\delta} \bar{0} \equiv_{\mathfrak{N}} \mathbf{0}$ for all $x \leq b$. Then

$$
\begin{aligned}
G\bar{a}_1 \ldots \bar{a}_n &\equiv G'\mathbf{0}\bar{a}_1 \ldots \bar{a}_n \\
&\equiv_{\mathfrak{N}} G'\bar{0}\bar{a}_1 \ldots \bar{a}_n \\
&\triangleright_{\beta} \big(\lambda\mathbf{0}.\bar{0} \mid \lambda z.G'(\mathbf{S}\bar{0})\bar{a}_1 \ldots \bar{a}_n\big)\big(H\bar{a}_1 \ldots \bar{a}_n\bar{0}\big) \\
&\triangleright_{1\delta} \big(\lambda z.G'(\mathbf{S}\bar{0})\bar{a}_1 \ldots \bar{a}_n\big)\big(H\bar{a}_1 \ldots \bar{a}_n\bar{0}\big) \quad (\because 0 \leq b) \\
&\triangleright_{1\beta} G'(\mathbf{S}\bar{0})\bar{a}_1 \ldots \bar{a}_n \\
&\equiv_{\mathfrak{N}} G'\bar{1}\bar{a}_1 \ldots \bar{a}_n \\
&\quad \vdots \\
&\triangleright_{\beta\delta} G'\bar{b}\bar{a}_1 \ldots \bar{a}_n \\
&\triangleright_{\beta} \big(\lambda\mathbf{0}.\bar{b} \mid \lambda z.G'(\mathbf{S}\bar{b})\bar{a}_1 \ldots \bar{a}_n\big)\big(H\bar{a}_1 \ldots \bar{a}_n\bar{b}\big) \\
&\triangleright_{1\delta} \big(\lambda z.G'(\mathbf{S}\bar{b})\bar{a}_1 \ldots \bar{a}_n\big)\big(H\bar{a}_1 \ldots \bar{a}_n\bar{b}\big) \quad (\because b \leq b) \\
&\triangleright_{1\beta} G'(\mathbf{S}\bar{b})\bar{a}_1 \ldots \bar{a}_n \\
&\equiv_{\mathfrak{N}} G'\bar{a}\bar{a}_1 \ldots \bar{a}_n \\
&\triangleright_{\beta} \big(\lambda\mathbf{0}.\bar{a} \mid \lambda z.G'(\mathbf{S}\bar{a})\bar{a}_1 \ldots \bar{a}_n\big)\big(H\bar{a}_1 \ldots \bar{a}_n\bar{a}\big) \\
&\triangleright_{\beta\delta} \big(\lambda\mathbf{0}.\bar{a} \mid \lambda z.G'(\mathbf{S}\bar{a})\bar{a}_1 \ldots \bar{a}_n\big)\mathbf{0} \\
&\triangleright_{1\delta} \big(\lambda\mathbf{0}.\bar{a}\big)\mathbf{0} \\
&\triangleright_{1\beta} \bar{a},
\end{aligned}
$$

so we have $G\bar{a}_1 \ldots \bar{a}_n \triangleright_{\beta\delta} \bar{a}$.

Hence restricted minimization preserves computability relative to $\mathfrak{N}$. □

5.2.5 Recursive Functions

Since the initial functions are all computable relative to $\mathfrak{N}$ and the applications of composition, primitive recursion, and restricted minimization to total functions all preserve computability relative to $\mathfrak{N}$, we have that every total recursive function on $\mathbb{N}$ is computable relative to $\mathfrak{N}$.

CHAPTER VI

ARITHMETIZATION

In the previous chapter, we showed that every total recursive function on $\mathbb{N}$ is **computable relative to $\mathfrak{N}$**. Unfortunately, although we believe the converse of the theorem holds, we have not been able to prove it yet. Nevertheless, the proof will most likely employ the technique of arithmetization, i.e., Gödel coding of terms and reductions (as in done in the proof of Gödel's Incompleteness Theorem, see [3]). Therefore, in this chapter, we will construct a Gödel coding for each element of the lambda calculus with patterns and define some auxiliary relations and functions. Then we will show a partial proof of the converse and point out where the problems are. Let $\mathfrak{N} = (\mathbb{N}, \{\mathtt{S}^{\mathfrak{N}}\}, \{\mathtt{0}^{\mathfrak{N}}\})$ be the standard structure for the language of arithmetic $\mathcal{L} = \{\mathtt{S},\ \mathtt{0}\}$.

6.1 Gödel Coding

In order to code reduction sequences, we start by assigning an odd positive integer to each symbol, then code terms and reductions.

6.1.1 Symbols

For each symbol u of our λP-calculus, the code for u is called the **Gödel number** of u, represented by $g(u)$.

Case 1. Basic symbols:

$$g(\,(\,) = 3, g(\,)\,) = 5, g(\,,\,) = 7, g(\cdot) = 9, g(\lambda) = 11, g(|) = 13, g(\varnothing) = 15.$$

Case 2. Contraction symbols:

$g(\triangleright_{1\beta}) = 17, g(\triangleright_{1\gamma}) = 19, g(\triangleright_{1\delta}) = 21.$

Case 3. Congruence symbols:

$g(\equiv_{1\alpha}) = 23, g(\equiv_{1\mathfrak{N}}) = 25.$

Case 4. Constants:

$g(\mathsf{0}) = 31, g(\mathsf{S}) = 39, g(\mathsf{T}) = 47, g(\mathsf{F}) = 55,$

and $g(\overline{k}) = 7 + 8(k + 7) = 63 + 8k$ where $k \geq 0$.

Case 5. Variables: $g(v_k) = 5 + 8(k + 2) = 21 + 8k$ where $k \geq 1$.

Then the Gödel numbers of all symbols are odd positive integers. Moreover, when divided by 8, $g(u)$ leaves a remainder of 5 when u is a variable, and a remainder of 7 when u is an individual constant. Note that there is no specific reason for choosing the number 8 other than to follow Gödel's convention. We could have choosen the Gödel numbers of variables and constants such that when divided by 4, $g(u)$ leaves a remainder of 1 when u is a variable, and a remainder of 3 when u is an individual constant, and the essence of the proof remains unaffected.

Also by Gödel's convention, we coded every symbol as an odd positive integer, and will code expressions and sequences of expressions (in our case, reductions) as even positive integers with different exponent of 2 in their prime power factorization to ensure the uniqueness of the code.

6.1.2 Expressions

We code an expression $M \equiv u_1 u_2 \ldots u_k$ by

$$g(M) = 2^{g(u_1)} 3^{g(u_2)} \ldots p_k^{g(u_k)}$$

where each u_i is a symbol, and p_k denotes the k^{th} prime number. Note that the Gödel number of an expression is an even positive integer and the exponent of 2 in its prime power factorization is odd.

6.1.3 Terms

For convenience in coding terms, we will rearrage the symbols of a term as a rooted binary tree. We first let $\varnothing$ denote an empty tree, then represent a nonempty tree by the expression (M, L, R), where M is its root and L and R are the left and right subtrees of the tree, respectively. Note that a leaf is represented by a tree with empty left and right subtrees, i.e., $(M, \varnothing, \varnothing)$.

A term can be represented as follows.

Case 1. Atom: t

Represented by the leaf $(t, \varnothing, \varnothing)$ where t is a variable or a constant.

Case 2. Application: (MN)

Represented by $(\cdot, m, n)$ where m and n are the tree representations for the terms M and N respectively.

Case 3. Simple abstraction: $(\lambda P.Q)$

Represented by (λ, p, q) where p is the tree representation for the pattern P and q is the tree representation for the term Q.

Case 4. Compound abstraction: $(M \mid A)$

Represented by $(|, m, a)$ where m is the tree representation for the simple abstraction M and a is the tree representation for the abstraction A.

The Gödel number of a nonempty tree can be defined inductively as follows:

$$g((u, L, R)) = 2^{g(u)} 3^{g(L)} 5^{g(R)}.$$

We then code a term by coding its tree representation. Note that the legal symbols for the root of a non-empty tree are a constant, a variable, $\cdot$, λ, or $|$.

6.1.4 Reductions

A reduction is a sequence of terms connected by contraction and congruence symbols. We can code a reduction $M_1 u_1 M_2 u_2 \ldots u_{k-1} M_k$ by

$$2^{g(M_1)} 3^{g(u_1)} 5^{g(M_2)} 7^{g(u_2)} \ldots p_{2k-2}^{g(u_{k-1})} p_{2k-1}^{g(M_k)}$$

where each M_i is a term, u_i is a contraction or congruence symbol, and p_k denotes the k^{th} prime number. The Gödel number of a reduction is an even positive integer, but unlike for a term, the exponent of 2 in its prime power factorization is even.

6.2 Primitive Recursive Relations and Functions

Using basic arithmetic, propositional connectives, and bounded quantifiers, which are all known to be primitive recursive, we define some auxiliary relations and functions and show that they are also primitive recursive.

6.2.1 Relations and Functions from Previous Work

Recall that a function is said to be **primitive recursive** if and only if it can be obtained from the initial functions by any finite number of applications of composition and primitive recursion. A relation is said to be primitive recursive if and only if its characteristic function is primitive recursive [3]. Each of the following relations and functions is primitive recursive (see [3] for proofs). We repeat the definitions here for reference.

(a) $x + y$

(b) $x \cdot y$

(c) x^y

(d) $x!$

(e) The **bounded μ-operator** is defined as follows.

$\mu y < zR(x_1, \ldots, x_n, y)$

$$= \begin{cases} \text{the least } y < z \text{ for which } R(x_1, \ldots, x_n, y) \text{ holds} & \text{if such } y \text{ exists,} \\ z & \text{otherwise.} \end{cases}$$

Also, we define $\mu y \leq zR(x_1, \ldots, x_n, y)$ to be $\mu y < (z+1)R(x_1, \ldots, x_n, y)$.

(f) Let $p(x)$ be the x_{th} prime number in ascending order. We shall write p_x instead of $p(x)$. Then $p_0 = 2, p_1 = 3, p_2 = 5$, and so on.

(g) Every positive integer x has a unique factorization into prime powers: $x = p_0^{a_0} p_1^{a_1} \ldots p_k^{a_k}$. Let $(x)_j$ denote the exponent a_j in this factorization. If $x = 1, (x)_j = 0$ for all j. If $x = 0$, we arbitrarily let $(x)_j = 1$ for all j.

(h) If the number $x = 2^{a_0} 3^{a_1} \ldots p_k^{a_k}$ is used to represent the sequence of positive integers $a_0, a_1, \ldots, a_k$, and $y = 2^{b_0} 3^{b_1} \ldots p_m^{b_m}$ represents the sequence of positive integers $b_0, b_1, \ldots, b_m$ then the number

$$x * y = 2^{a_0} 3^{a_1} \ldots p_k^{a_k} p_{k+1}^{b_0} p_{k+2}^{b_1} \ldots p_{k+1+m}^{b_m}$$

represents the new sequence $a_0, a_1, \ldots, a_k, b_0, b_1, \ldots, b_m$ obtained by **juxtaposing** the two sequences. The function $*$ is primitive recursive, called the **juxtaposition** function.

(i) Relations obtained from primitive recursive relations by means of the propositional connectives and the bounded quantifiers are also primitive recursive.

6.2.2 Auxiliary Relations and Functions

In this subsection we will define various functions and relations that will ultimately be used to prove that every total function on $\mathbb{N}$ computable relative to $\mathfrak{N}$ is recursive. For that proof we need only one relation, `IsRedOf`. However, the simplest way to define `IsRedOf` and prove that it is recursive is to define a sequence of auxiliary functions and relations, each of which is relatively simple to define and easily seen to be recursive. For each function or relation we will give a verbal description, followed by a symbolic definition, from which recursiveness will be clear.

`IsVar(x):` x is the Gödel number of a variable.

: $\exists k < x(x = 29 + 8k)$

`IsConst(x):` x is the Gödel number of a constant.

: $\exists k < x(x = 31 + 8k)$

`Num(x):` The Gödel number of a constant $\bar{x}$ corresponding to $x \in \mathbb{N}$.

: $71 + 8x$

`IsSym(x):` x is the Gödel number of a symbol.

: $x = g(\texttt{(}) \vee x = g(\texttt{)}) \vee x = g(\texttt{,}) \vee x = g(\cdot) \vee x = g(\lambda) \vee x = g(|) \vee x = g(\varnothing)$
$\vee\ \texttt{IsVar}(x) \vee \texttt{IsConst}(x)$

`IsROp(x):` x is the Gödel number of a 1α-conversion or a contraction symbol.

: $x = g(\equiv_{1\alpha}) \vee x = g(\rhd_{1\beta}) \vee x = g(\rhd_{1\gamma}) \vee x = g(\rhd_{1\delta})$.

`IsTreeRoot(x):` x is the Gödel number of a symbol that can be the root of a tree.

: $x = g(\cdot) \vee x = g(\lambda) \vee x = g(|) \vee \texttt{IsVar}(x) \vee \texttt{IsConst}(x)$

`IsTree(x)`: x is the Gödel number of a tree.

: $x = g(\varnothing) \vee \exists u, l, r < x\big(x = 2^u 3^l 5^r \wedge \texttt{IsTreeRoot}(u) \wedge \texttt{IsTree}(l) \wedge \texttt{IsTree}(r)\big)$

`Root(x)`: The Gödel number of the root of a tree with Gödel number x.

: $(x)_0$

Remark. If x is $\varnothing$ or x is not a tree, `Root(x)` is still defined but its value is of no interest, similarly for `LSubT(x), RSubT(x)`, and `Tree(u,l,r)`.

`LSubT(x)`: The Gödel number of the left subtree of a nonempty tree with Gödel number x.

: $(x)_1$

`RSubT(x)`: The Gödel number of the right subtree of a nonempty tree with Gödel number x.

: $(x)_2$

`IsLeaf(x)`: x is the Gödel number of a leaf.

: $\texttt{IsTree}(x) \wedge \texttt{LSubT}(x) = g(\varnothing) = \texttt{RSubT}(x)$

`IsSubT(x,y)`: x is the Gödel number of a subtree of a tree with Gödel number y.

: $\texttt{IsTree}(x) \wedge \texttt{IsTree}(y)$

$\wedge \big[x = y \vee \texttt{IsSubT}\big(x, \texttt{LSubT}(y)\big) \vee \texttt{IsSubT}\big(x, \texttt{RSubT}(y)\big)\big]$

`Tree(u,l,r)`: The Gödel number of a tree for which the Gödel numbers of its root, left and right subtrees are u, l, and r respectively.

: $2^u 3^l 5^r$

`IsVarTerm(x)`: x is the Gödel number of a term consisting of a single variable.

: $\texttt{IsLeaf}(x) \wedge \texttt{IsVar}(\texttt{Root}(x))$

`IsConstTerm(x):` x is the Gödel number of a term consisting of a single constant.

$: \texttt{IsLeaf}(x) \wedge \texttt{IsConst}(\texttt{Root}(x))$

`IsAtom(x):` x is the Gödel number of an atomic term.

$: \texttt{IsVarTerm}(x) \vee \texttt{IsConstTerm}(x)$

`IsVnTree(x,y):` x is the Gödel number of a variable occuring in a tree with Gödel number y.

$: \texttt{IsVar}(x) \wedge \texttt{IsTree}(y)$

$\wedge\Big(\big(\texttt{IsVarTerm}(y) \wedge \texttt{Root}(y) = x\big) \vee \exists m < y\big(\texttt{IsSubT}(m, y) \wedge \texttt{IsVnTree}(x, m)\big)\Big)$

`IsPat(x):` x is the Gödel number of a pattern.

$: \texttt{IsAtom}(x)$

$\vee\ [\texttt{IsTree}(x) \wedge \texttt{Root}(x) = g(\cdot)$

$\wedge\ \texttt{IsPat}(\texttt{LSubT}(x)) \wedge \neg\texttt{IsVarTerm}(\texttt{LSubT}(x))$

$\wedge\ \texttt{IsPat}(\texttt{RSubT}(x))$

$\wedge\ \neg\exists n < x\big(\texttt{IsVnTree}(n, \texttt{LSubT}(x)) \wedge \texttt{IsVnTree}(n, \texttt{RSubT}(x))\big)]$

The relations `IsApp`, `IsSAbst`, `IsCAbst`, and `IsTerm` are defined recursively and simultaneously as follows.

`IsApp(x):` x is the Gödel number of an application.

$: \texttt{IsTree}(x) \wedge \texttt{Root}(x) = g(\cdot)$

$\wedge\ \texttt{IsTerm}(\texttt{LSubT}(x)) \wedge \texttt{IsTerm}(\texttt{RSubT}(x))$

`IsSAbst(x):` x is the Gödel number of a simple abstraction.

$: \texttt{IsTree}(x) \wedge \texttt{Root}(x) = g(\lambda)$

$\wedge\ \texttt{IsPat}(\texttt{LSubT}(x)) \wedge \texttt{IsTerm}(\texttt{RSubT}(x))$

`IsCAbst(x):` x is the Gödel number of a compound abstraction.

$: \texttt{IsTree}(x) \wedge \texttt{Root}(x) = g(|)$

$\wedge\ \mathtt{IsSAbst}(\mathtt{LSubT}(x))$

$\wedge\ \big(\mathtt{IsSAbst}(\mathtt{RSubT}(x)) \vee \mathtt{IsCAbst}(\mathtt{RSubT}(x))\big)$

`IsTerm(x):` x is the Gödel number of a term.

: $\mathtt{IsAtom}(x) \vee \mathtt{IsApp}(x) \vee \mathtt{IsSAbst}(x) \vee \mathtt{IsCAbst}(x)$

`IsAbst(x):` x is the Gödel number of an abstraction.

: $\mathtt{IsSAbst}(x) \vee \mathtt{IsCAbst}(x)$

`IsFV(x,n):` x is the Gödel number of a free variable of a term with Gödel number n.

: $\mathtt{IsVar}(x) \wedge \mathtt{IsTerm}(n)$

$\wedge\ \big\{[\mathtt{Root}(n) = x]$

$\vee\ [\mathtt{IsSAbst}(n) \wedge \neg\mathtt{IsFV}(x, \mathtt{LSubT}(n)) \wedge \mathtt{IsFV}(x, \mathtt{RSubT}(n))]$

$\vee\ [\big(\mathtt{IsApp}(n) \vee \mathtt{IsCAbst}(n)\big)$

$\wedge\ \big(\mathtt{IsFV}(x, \mathtt{LSubT}(n)) \vee \mathtt{IsFV}(x, \mathtt{RSubT}(n))\big)]\big\}$

`IsSubst(y,n,x,m):` y is the Gödel number of the result of substituting a term with Gödel number n for all free occurences of a variable with Gödel number x in a term with Gödel number m. This is done under the assumption that n is free for x in m.

: $\mathtt{IsTerm}(y) \wedge \mathtt{IsTerm}(n) \wedge \mathtt{IsVar}(x) \wedge \mathtt{IsTerm}(m)$

$\wedge\ \Big\{\big[\neg\mathtt{IsFV}(x, m) \wedge y = m\big]$

$\vee\ \big[\mathtt{IsFV}(x, m) \wedge \{[\mathtt{IsAtom}(m) \wedge y = n]$

$\vee\ [\neg\mathtt{IsAtom}(m) \wedge \mathtt{Root}(y) = \mathtt{Root}(m)$

$\wedge\ \mathtt{IsSubst}(\mathtt{LSubT}(y), n, x, \mathtt{LSubT}(m)\big)$

$\wedge\ \mathtt{IsSubst}(\mathtt{RSubT}(y), n, x, \mathtt{RSubT}(m)\big)]\}\big]\Big\}$

`Subst(n,x,m):` The Gödel number of the result of substituting a term with Gödel number n for all free occurences of a variable with Gödel number x in a term with Gödel number m.

: $\mu y < (p_{nm}!)^{nm}(\texttt{IsSubst}(y, n, x, m))$

`IsOneA(m,n):` m is the Gödel number of a term which is obtained from a term with Gödel number n by a single α-step.

$$
\begin{aligned}
&: \texttt{IsTerm}(m) \wedge \texttt{IsTerm}(n) \wedge (\texttt{Root}(m) = \texttt{Root}(n))\\
&\wedge \big\{\big[\big(\texttt{IsApp}(n) \vee \texttt{IsCAbst}(n)\big)\\
&\qquad \wedge \big(\{\texttt{LSubT}(m) = \texttt{LSubT}(n) \wedge \texttt{IsOneA}(\texttt{RSubT}(m), \texttt{RSubT}(n))\}\\
&\qquad\qquad \vee \{\texttt{IsOneA}(\texttt{LSubT}(m), \texttt{LSubT}(n)) \wedge \texttt{RSubT}(m) = \texttt{RSubT}(n)\}\big)\big]\\
&\qquad \vee \big[\texttt{IsSAbst}(n)\\
&\qquad\qquad \wedge \big(\{\texttt{LSubT}(m) = \texttt{LSubT}(n) \wedge \texttt{IsOneA}(\texttt{RSubT}(m), \texttt{RSubT}(n))\}\\
&\qquad\qquad\qquad \vee \{\exists x < n \exists y < m[\texttt{IsFV}(x, \texttt{LSubT}(n))\\
&\qquad\qquad\qquad\qquad\qquad \wedge \texttt{IsVar}(y)\\
&\qquad\qquad\qquad\qquad\qquad \wedge \neg\texttt{IsFV}(y, \texttt{LSubT}(n))\\
&\qquad\qquad\qquad\qquad\qquad \wedge \neg\texttt{IsFV}(y, \texttt{RSubT}(n))\\
&\qquad\qquad\qquad\qquad\qquad \wedge \texttt{IsFreeFor}(y, x, \texttt{RSubT}(n))\\
&\qquad\qquad\qquad\qquad\qquad \wedge \texttt{IsSubst}(\texttt{LSubT}(m), y, x, \texttt{LSubT}(n)),\\
&\qquad\qquad\qquad\qquad\qquad \wedge \texttt{IsSubst}(\texttt{RSubT}(m), y, x, \texttt{RSubT}(n))]\}\big)\big]\big\}
\end{aligned}
$$

`IsOneACon(x):` x is the Gödel number of a single step α-conversion.

: $\exists u, v < x(x = 2^u * 2^{g(\equiv_{1\alpha})} * 2^v) \wedge \texttt{IsOneA}(v, u)$

`IsOneB(m,n):` m is the Gödel number of a term which is obtained from a term with Gödel number n by a β-contraction.

: A definition showing that `IsOneB` is primitive recursive has not yet been found.

`IsOneBCon(x):` x is the Gödel number of a β-contraction.

: $\exists u, v < x(x = 2^u * 2^{g(\triangleright_{1\beta})} * 2^v) \wedge \texttt{IsOneB}(v, u)$

`IsOneG(m,n)`: m is the Gödel number of a term which is obtained from a term with Gödel number n by a γ-contraction.

: A definition showing that `IsOneG` is primitive recursive has not yet been found.

`IsOneGCon(x)`: x is the Gödel number of a γ-contraction.

: $\exists u, v < x(x = 2^u * 2^{g(\triangleright_{1\gamma})} * 2^v) \wedge \texttt{IsOneG}(v, u)$

`IsOneD(m,n)`: m is the Gödel number of a term which is obtained from a term with Gödel number n by a δ-contraction.

: A definition showing that `IsOneD` is primitive recursive has not yet been found.

`IsOneDCon(x)`: x is the Gödel number of a δ-contraction.

: $\exists u, v < x(x = 2^u * 2^{g(\triangleright_{1\delta})} * 2^v) \wedge \texttt{IsOneD}(v, u)$

`IsOneN(m,n)`: m is the Gödel number of a term which is obtained from a term with Gödel number n by a single step $\mathfrak{N}$-conversion.

: A definition showing that `IsOneN` is primitive recursive has not yet been found.

`IsOneNCon(x)`: x is the Gödel number of a single step $\mathfrak{N}$-conversion.

: $\exists u, v < x(x = 2^u * 2^{g(\equiv_{1\mathfrak{N}})} * 2^v) \wedge \texttt{IsOneN}(v, u)$

`IsOneRed(x)`: x is the Gödel number of a “single step” reduction.

: $\texttt{IsOneACon}(x) \vee \texttt{IsOneBCon}(x) \vee \texttt{IsOneGCon}(x) \vee \texttt{IsOneDCon}(x) \vee \texttt{IsOneNCon}(x)$

`IsRedOf(x,m,n)`: x is the Gödel number of a reduction from a term with Gödel number m to a term with Gödel number n.

: $\texttt{IsTerm}(m) \wedge \texttt{IsTerm}(n)$

$\wedge \exists u, y < x\big(x = 2^m * 2^u * y \wedge \texttt{IsROp}(u)$

$$\wedge \{[\texttt{IsOneRed}(x) \wedge y = 2^n]$$
$$\vee [\exists k < y(\texttt{IsRedOf}(y,k,n) \wedge \texttt{IsOneRed}(2^m * 2^u * 2^k))]\}\big)$$

6.3 Computability Relative to $\mathfrak{N}$ Implies Recursiveness

Notice that definitions showing the recursiveness of some key relations, namely `IsOneB`, `IsOneG`, `IsOneD`, and `IsOneN`, are missing in the previous section. Of these, we expect that the ones for γ-contraction and δ-contraction, `IsOneG` and `IsOneD`, will be the most challenging to find, but it is likely that they will also be quite similar. If we can find definitions showing that all four of these relations are recursive then we can prove our main theorem, as follows.

Theorem 6.3.1. *If an n-ary total function g on $\mathbb{N}$ is computable relative to $\mathfrak{N}$, then g is recursive.*

Proof. Assume that g is computable relative to $\mathfrak{N}$. Let G be a term representing g and let v be the Gödel number of G. Define the $n+2$-ary relation R_G on $\mathbb{N}$ by

$R_G(x_1, x_2, \ldots, x_n, y, z)$ iff z is the Gödel number of the reduction $G\bar{x}_1\bar{x}_2 \ldots \bar{x}_n \rhd_{\beta\delta} \bar{y}$.

Then

$$R_G(x_1, x_2, \ldots, x_n, y, z) \equiv \texttt{IsRedOf}\big(z, \big(v*\texttt{Num}(x_1)*\texttt{Num}(x_2)*\cdots*\texttt{Num}(x_n)\big), \texttt{Num}(y)\big),$$

so R_G is recursive. Let $u_1, u_2, \ldots, u_n \in \mathbb{N}$. Suppose $g(u_1, u_2, \ldots, u_n) = u$ for some $u \in \mathbb{N}$. Since g is computable relative to $\mathfrak{N}$, we have $G\bar{u}_1\bar{u}_2 \ldots \bar{u}_n \rhd_{\beta\delta} \bar{u}$. Let s be the Gödel number of the above reduction. Then $R_G(u_1, u_2, \ldots, u_n, u, s)$ holds. Hence for any $x_1, x_2, \ldots, x_n$ there exists $y \in \mathbb{N}$ such that $R_G(x_1, x_2, \ldots, x_n, (y)_0, (y)_1)$ holds. Since $g(x_1, x_2, \ldots, x_n) = \big(\mu y[R_G(x_1, x_2, \ldots, x_n, (y)_0, (y)_1)]\big)_0$ and $\mu y[R_G(x_1, x_2, \ldots, x_n, (y)_0, (y)_1)]$ is recursive, we see that g is recursive. $\square$

CHAPTER VII

CONCLUSION

We have extended the concept of computability to functions on an arbitrary first-order structure using the lambda calculus with patterns. In doing so, we added a new congruence, congruence in a structure, which we proved to preserve all the basic properties of the original lambda calculus including the Church-Rosser theorem. It is interesting to notice that, when defining patterns using the non-logical symbols from a language, only the function symbols which represent one-to-one functions are allowed in a pattern. Such a constraint is neccessary for the validity of the Church-Rosser theorem. For example, if we were allowed to use the symbol A, which represents the addition function on the natural numbers, in patterns, then $(\lambda \mathsf{A}xy.x)\bar{2} \equiv_{\mathfrak{A}} (\lambda \mathsf{A}xy.x)(\mathsf{A}\bar{1}\bar{1}) \triangleright_\beta \bar{1}$ and $(\lambda \mathsf{A}xy.x)\bar{2} \equiv_{\mathfrak{A}} (\lambda \mathsf{A}xy.x)(\mathsf{A}\bar{0}\bar{2}) \triangleright_\beta \bar{0}$, but $\bar{1}$ and $\bar{0}$ do not reduce to anything in common, so the Church-Rosser theorem would fail to hold.

For the standard structure $\mathfrak{N}$ for the natural numbers, we have shown that every recursive total function on $\mathbb{N}$ is computable relative to $\mathfrak{N}$, in other words, it can be represented by a λP-term. So a question arises, how do we represent a recursive partial function? One possibility is through definition by cases. Since we may add a dummy symbol, say ∞, to the definition of the lambda calculus with patterns by calling it another constant, we can define a term to represent a recursive partial function if at all inputs for which the function value is defined the term applied to those inputs reduces to the corresponding result, and at all inputs for which the function value is not defined the term applied to those inputs reduces to ∞. We can do this by adding the undefined input case as the last case of a compound abstraction, i.e., $(\lambda P_1.Q_1 \mid (\lambda P_2.Q_2 \mid (\cdots \mid (\lambda P_n.Q_n \mid \lambda x.\infty))))$. Of course, this idea needs proper definitions and further investigation to verify.

As we have explained in Section 6.3, another challenging task that remains is to find a recursive relation that identifies delta contractions. Suppose we have a compound abstraction $\big((\lambda P.Q) \mid A\big)$ and a term M. When trying to decide whether $(\lambda P.Q)M$ reduces to a contractible redex, we must find a way to tell when we can stop and conclude that the compound abstraction reduces to (AM). For example, if we can prove that it is sufficient to try contracting only a finite number of times, for a given potential redex, then we have an upper bound for our search. Such a bound, the maximum number of contractions needed, would surely depend on P and M. Due to the simple structure of patterns and the limited number of non-logical symbols in the language of arithmetic, i.e., only $\mathbf{0}$ and $\mathbf{S}$, it may be possible to find a formula (to be precise, a recursive function of the Gödel codings of P and M) for calculating such a maximum number. The readers are encouraged to attempt finding this formula, which would enable us to finish the Gödel coding of the delta contraction, which in turn would complete our proof of the equivalence of the recursiveness and computability relative to a structure.

REFERENCES

[1] Hindley, J.R., and Seldin, J.P. *Introduction to Combinators and λ-calculus.* Cambridge: Cambridge University Press, 1986.

[2] Kechris, A.S. *Mathematical Logic and Axiomatic Set Theory.* (Lecture Notes).

[3] Mendelson, E. *Introduction to Mathematical Logic.* Florida: Chapman & Hall/CRC, 1997.

[4] Rogers, H. *Theory of Recursive Functions and Effective Computability.* New York: McGraw-Hill, 1967.

[5] Vejjajiva, P. *A Lambda-Calculus with Patterns.* Master's Thesis. Department of Mathematics, Chulalongkorn University, 1997.

[6] Vejjajiva, P., and Hall, M. E. A Lambda-Calculus with Patterns. *Proceedings of the International Conference on Algebra and Its Applications (ICAA 2002)*, 266-277 (2002).

VITA

Name: Mr. Bodin Skulkiat

Degree: *Bachelor of Engineering in Computer Engineering*, 1997
Chulalongkorn University, Bangkok, Thailand.

Master of Science in Electrical and Computer Engineering, 1998
University of Colorado at Boulder, Colorado, USA.

Master of Science in Real-Estate Business, 2003
Thammasat University, Bangkok, Thailand.