

# BACHELOR'S THESIS COMPUTING SCIENCE



RADBOUD UNIVERSITY NIJMEGEN

---

**Thesis Title**

---

*Subtitle if you like*

*Author:*  
Floris Reuvers  
s1096976

*First supervisor/assessor:*  
Dr. Niels van der Weide

*Second assessor:*  
Dr. Engelbert Hubbers

March 31, 2025

### **Abstract**

Brief outline of research questions, results. (The preferred size of an abstract is one paragraph or one page of text.)

# Contents

<b>1</b>	<b>Introduction</b>	<b>2</b>
<b>2</b>	<b>Rules</b>	<b>3</b>
2.1	Rules Intuitionistic Logic . . . . .	3
2.2	Rules Intuitionistic Logic with Types . . . . .	4
2.3	Rules Linear Logic . . . . .	5
2.4	Rules Linear Logic with Types . . . . .	7
<b>3</b>	<b>Lambda Calculus</b>	<b>8</b>
3.1	Introduction to the $\lambda$ -calculus . . . . .	8
3.2	$\beta$ -Reduction . . . . .	8
3.2.1	Variables . . . . .	9
3.2.2	Substitution . . . . .	10
3.2.3	$\alpha$ equivalence . . . . .	11
3.2.4	$\beta$ -Reduction . . . . .	11
3.3	Call-by-name calculus . . . . .	12
3.4	Call-by-value calculus . . . . .	12
<b>4</b>	<b>Call-by-box Lambda Calculus</b>	<b>14</b>
4.1	Specification of $\lambda_b$ . . . . .	14
4.2	Embedding of $\lambda_n$ into $\lambda_b$ . . . . .	15
4.3	Embedding of $\lambda_v$ into $\lambda_b$ . . . . .	16
<b>5</b>	<b>Related Work</b>	<b>17</b>
<b>6</b>	<b>Conclusions</b>	<b>18</b>
<b>A</b>	<b>Appendix</b>	<b>20</b>

# Chapter 1

## Introduction

The introduction of your bachelor thesis introduces the research area, the research hypothesis, and the scientific contributions of your work. A good narrative structure is the one suggested by Simon Peyton Jones [1]:

- describe the problem / research question
- motivate why this problem must be solved
- demonstrate that a (new) solution is needed
- explain the intuition behind your solution
- motivate why / how your solution solves the problem (this is technical)
- explain how it compares with related work

Close the introduction with a paragraph in which the content of the next chapters is briefly mentioned (one sentence per chapter).

Starting a new paragraph is done by inserting an empty line like this.

## Chapter 2

# Rules

### 2.1 Rules Intuitionistic Logic

$$\begin{array}{c}
 \frac{}{A \vdash A} \text{Id} \qquad \frac{\Gamma, \Delta \vdash A}{\Delta, \Gamma \vdash A} \text{Exchange} \\
 \\
 \frac{\Gamma, A, A \vdash B}{\Gamma, A \vdash B} \text{Contraction} \qquad \frac{\Gamma \vdash B}{\Gamma, A \vdash B} \text{Weakening} \\
 \\
 \frac{\Gamma, A \vdash B}{\Gamma \vdash A \rightarrow B} \rightarrow\text{-I} \qquad \frac{\Gamma \vdash A \rightarrow B \quad \Delta \vdash A}{\Gamma, \Delta \vdash B} \rightarrow\text{-E} \\
 \\
 \frac{\Gamma \vdash A \quad \Delta \vdash B}{\Gamma, \Delta \vdash A \times B} \times\text{-I} \qquad \frac{\Gamma \vdash A \times B \quad \Delta, A, B \vdash C}{\Gamma, \Delta \vdash C} \times\text{-E} \\
 \\
 \frac{\Gamma \vdash A}{\Gamma \vdash A + B} +\text{-I}_1 \qquad \frac{\Gamma \vdash B}{\Gamma \vdash A + B} +\text{-I}_2 \\
 \\
 \frac{\Gamma, A \vdash C \quad \Delta, A \vdash C \quad \Delta, B \vdash C}{\Gamma, \Delta \vdash C} +\text{-E}
 \end{array}$$

Figure 2.1: Multiple Proof Rules in Columns

## 2.2 Rules Intuitionistic Logic with Types

$$\begin{array}{c}
\frac{}{x : A \vdash x : A} \text{Id} \qquad \frac{\Gamma, \Delta \vdash t : A}{\Delta, \Gamma \vdash t : A} \text{Exchange} \\
\\
\frac{\Gamma, y : A, z : A \vdash u : B}{\Gamma, x : A \vdash u[y \mapsto x][z \mapsto x] : B} \text{Contraction} \\
\\
\frac{\Gamma \vdash u : B}{\Gamma, x : A \vdash u : B} \text{Weakening} \\
\\
\frac{\Gamma, x : A \vdash t : B}{\Gamma \vdash \lambda x. t : A \rightarrow B} \rightarrow\text{-I} \\
\\
\frac{\Gamma \vdash s : A \rightarrow B \quad \Delta \vdash t : A}{\Gamma, \Delta \vdash s(t) : B} \rightarrow\text{-E} \\
\\
\frac{\Gamma \vdash t : A \quad \Delta \vdash u : B}{\Gamma, \Delta \vdash (t, u) : A \times B} \times\text{-I} \\
\\
\frac{\Gamma \vdash s : A \times B \quad \Delta, x : A, y : B \vdash v : C}{\Gamma, \Delta \vdash \text{case } s \text{ of } (x, y) \rightarrow v : C} \times\text{-E} \\
\\
\frac{\Gamma \vdash t : A}{\Gamma \vdash \text{inl}(t) : A + B} +\text{-I}_1 \qquad \frac{\Gamma \vdash u : B}{\Gamma \vdash \text{inr}(u) : A + B} +\text{-I}_2 \\
\\
\frac{\Gamma \vdash s : A + B \quad \Delta, x : A \vdash v : C \quad \Delta, y : B \vdash w : C}{\Gamma, \Delta \vdash \text{case } s \text{ of } \text{inl}(x) \rightarrow v; \text{inr}(y) \rightarrow w : C} +\text{-E}
\end{array}$$

Figure 2.2: Multiple Proof Rules in Columns

## 2.3 Rules Linear Logic

$$\begin{array}{c}
\frac{}{\langle A \rangle \vdash A} \langle \text{Id} \rangle \qquad \frac{}{[A] \vdash A} [\text{Id}] \qquad \frac{\Gamma, \Delta \vdash A}{\Delta, \Gamma \vdash A} \text{Exchange} \\
\\
\frac{\Gamma, [A], [A] \vdash B}{\Gamma, [A] \vdash B} \text{Contraction} \qquad \frac{\Gamma \vdash B}{\Gamma, [A] \vdash B} \text{Weakening} \\
\\
\frac{[\Gamma] \vdash A}{[\Gamma] \vdash !A} !\text{-I} \qquad \frac{\Gamma \vdash !A \quad \Delta, [A] \vdash B}{\Gamma, \Delta \vdash B} !\text{-E} \\
\\
\frac{\Gamma, \langle A \rangle \vdash B}{\Gamma \vdash A \multimap B} \multimap\text{-I} \qquad \frac{\Gamma \vdash A \multimap B \quad \Delta \vdash A}{\Gamma, \Delta \vdash B} \multimap\text{-E} \\
\\
\frac{\Gamma \vdash A \quad \Delta \vdash B}{\Gamma, \Delta \vdash A \otimes B} \otimes\text{-I} \\
\\
\frac{\Gamma \vdash A \otimes B \quad \Delta, \langle A \rangle, \langle B \rangle \vdash C}{\Gamma, \Delta \vdash C} \otimes\text{-E} \\
\\
\frac{\Gamma \vdash A \quad \Gamma \vdash B}{\Gamma \vdash A \& B} \&\text{-I} \\
\frac{\Gamma \vdash A \& B}{\Gamma \vdash A} \&\text{-E}_1 \qquad \frac{\Gamma \vdash A \& B}{\Gamma \vdash B} \&\text{-E}_2 \\
\\
\frac{\Gamma \vdash A}{\Gamma \vdash A \oplus B} \oplus\text{-I}_1 \qquad \frac{\Gamma \vdash B}{\Gamma \vdash A \oplus B} \oplus\text{-I}_2 \\
\\
\frac{\Gamma \vdash A \oplus B \quad \Delta, \langle A \rangle \vdash C \quad \Delta, \langle B \rangle \vdash C}{\Gamma, \Delta \vdash C} \oplus\text{-E}
\end{array}$$

Figure 2.3: Multiple Proof Rules in Columns





## 2.4 Rules Linear Logic with Types

$$\begin{array}{c}
\frac{}{\langle x : A \rangle \vdash x : A} \langle \text{Id} \rangle \qquad \frac{}{[x : A] \vdash x : A} [\text{Id}] \\
\\
\frac{\Gamma, \Delta \vdash t : A}{\Delta, \Gamma \vdash t : A} \text{Exchange} \qquad \frac{\Gamma \vdash u : B}{\Gamma, [x : A] \vdash u : B} \text{Weakening} \\
\\
\frac{\Gamma, [y : A], [z : A] \vdash u : B}{\Gamma, [x : A] \vdash u[y \mapsto x][z \mapsto x] : B} \text{Contraction} \\
\\
\frac{[\Gamma] \vdash t : A}{[\Gamma] \vdash !t : !A} !\text{-I} \\
\\
\frac{\Gamma \vdash s : !A \quad \Delta, [x : A] \vdash u : B}{\Gamma, \Delta \vdash \text{case } s \text{ of } !x \rightarrow u : B} !\text{-E} \\
\\
\frac{\Gamma, \langle x : A \rangle \vdash u : B}{\Gamma \vdash \lambda \langle x \rangle. u : A \multimap B} \multimap\text{-I} \\
\\
\frac{\Gamma \vdash s : A \multimap B \quad \Delta \vdash t : A}{\Gamma, \Delta \vdash s \langle t \rangle : B} \multimap\text{-E} \\
\\
\frac{\Gamma \vdash t : A \quad \Delta \vdash u : B}{\Gamma, \Delta \vdash \langle t, u \rangle : A \otimes B} \otimes\text{-I} \\
\\
\frac{\Gamma \vdash s : A \otimes B \quad \Delta, \langle x : A \rangle, \langle y : B \rangle \vdash v : C}{\Gamma, \Delta \vdash \text{case } s \text{ of } \langle x, y \rangle \rightarrow v : C} \otimes\text{-E} \\
\\
\frac{\Gamma \vdash t : A \quad \Gamma \vdash u : B}{\Gamma \vdash \langle \langle t, u \rangle \rangle : A \& B} \&\text{-I} \\
\frac{\Gamma \vdash s : A \& B}{\Gamma \vdash \text{fst} \langle s \rangle : A} \&\text{-E}_1 \qquad \frac{\Gamma \vdash s : A \& B}{\Gamma \vdash \text{snd} \langle s \rangle : B} \&\text{-E}_2 \\
\\
\frac{\Gamma \vdash t : A}{\Gamma \vdash \text{inl} \langle t \rangle : A \oplus B} \oplus\text{-I}_1 \qquad \frac{\Gamma \vdash u : B}{\Gamma \vdash \text{inr} \langle u \rangle : A \oplus B} \oplus\text{-I}_2 \\
\\
\frac{\Gamma \vdash s : A \oplus B \quad \Delta, \langle x : A \rangle \vdash v : C \quad \Delta, \langle y : B \rangle \vdash w : C}{\Gamma, \Delta \vdash \text{case } s \text{ of } \text{inl} \langle x \rangle \xrightarrow{7} v; \text{inr} \langle y \rangle \rightarrow w : C} \oplus\text{-E}
\end{array}$$

Figure 2.4: Multiple Proof Rules in Columns

## Chapter 3

# Lambda Calculus

This section provides a detailed and formal description of the  $\lambda$ -calculus. We define a formal grammar of the  $\lambda$ -calculus and give examples of some terms in the  $\lambda$ -calculus. Then we explain  $\beta$ -reduction, after which we define call-by-name evaluation and call-by-value evaluation.

### 3.1 Introduction to the $\lambda$ -calculus

The grammar of the lambda calculus is as follows:

$$M, N, P, Q ::= x \mid \lambda x.M \mid MN$$

Terms are denoted by  $M, N, P$  or  $Q$  and can either be of the form  $x$ ,  $\lambda x.M$  or  $MN$ :

- $x$  is a variable, which is a symbol that represents an input.
- $\lambda x.M$  is an abstraction. An abstraction is an anonymous function, where  $x$  is the parameter and  $M$  is the body of the function.
- $MN$  is a function application, where  $M$  and  $N$  are terms.

The  $\lambda$ -term  $\lambda x.x$  is an abstraction. This specific abstraction is called the identity function and has one input parameter, namely  $x$ , and returns the input  $x$ . The body of the function is also  $x$  in this case and the function is often abbreviated as **I**. The  $\lambda$ -term  $(\lambda x.x)y$  is a function application. So the identity function is applied to the variable  $y$  and the  $\lambda$ -term reduces to the variable  $y$ .

### 3.2 $\beta$ -Reduction

Before we discuss  $\beta$ -reduction, we first define bound and free variables and substitution.

### 3.2.1 Variables

Let us first define free variables. Let  $P$  be any term in the  $\lambda$ -calculus and  $FV(P)$  be the set that contains all free variables in  $P$ . We define  $FV$  inductively on  $P$ . Since  $P$  can only be a variable, an abstraction or a function application,  $P$  can only be of the form  $x$ ,  $\lambda x.M$  and  $MN$ . Therefore, we define  $FV$  as follows:

$$\begin{aligned} FV(x) &= \{x\} \\ FV(\lambda x.M) &= FV(M) \setminus \{x\} \\ FV(MN) &= FV(M) \cup FV(N) \end{aligned}$$

The variable in a  $\lambda$ -term that solely consists of a variable is free. All free variables in an abstraction are those that are not the parameter of the abstraction. For example, the  $\lambda$ -term  $\lambda x.yx$  has one free variable,  $y$ . We can use the following reasoning:

$$\begin{aligned} FV(\lambda x.yx) &= FV(yx) \setminus \{x\} \\ &= (FV(x) \cup FV(y)) \setminus \{x\} \\ &= (\{x\} \cup \{y\}) \setminus \{x\} \\ &= \{x, y\} \setminus \{x\} \\ &= \{y\} \end{aligned}$$

Let us now define bound variables. Again, let  $P$  be any term in the  $\lambda$ -calculus and  $BV(P)$  be the set of all bound variables in  $P$ . We define  $BV$  inductively on  $P$  as follows:

$$\begin{aligned} BV(x) &= \emptyset \\ BV(\lambda x.M) &= BV(M) \cup \{x\} \\ BV(MN) &= BV(M) \cup BV(N) \end{aligned}$$

We say that the abstraction  $\lambda x.M$  binds the variable  $x$ . Using our previous example of the  $\lambda$ -term  $\lambda x.yx$ , we can now reason that  $BV(\lambda x.yx) = \{x\}$ .

$$\begin{aligned} BV(\lambda x.xy) &= BV(xy) \cup \{x\} \\ &= (BV(x) \cup BV(y)) \cup \{x\} \\ &= (\emptyset \cup \emptyset) \cup \{x\} \\ &= \{x\} \end{aligned}$$

Let us consider a more tricky example:  $\lambda x.(\lambda x.x)x$ . Using the definitions of  $FV$  and  $BV$ , we can reason that  $FV(\lambda x.(\lambda x.x)x) = \{\emptyset\}$  and  $BV(\lambda x.(\lambda x.x)x) = \{x\}$ . However,  $FV(\lambda x.xx) = \{x\}$  and  $BV(\lambda x.xx) =$

$\{x\}$ , which means that the variable  $x$  is both free and bound. This may not be very intuitive, so we would prefer to write  $\lambda$ -term as  $\lambda y.(\lambda x.x)y$  instead, where  $y$  is a new variable. These  $\lambda$ -terms are equivalent by the notion of *alpha equivalence*, which states that  $\lambda$ -terms that differ only by the names of bound variables are considered the same. We will discuss this in more detail after we define substitution.

### 3.2.2 Substitution

Keeping the definitions of free and bound variables in mind, we now define substitution. In this research, substitution is often denoted as  $P[x \mapsto Q]$  and defined inductively on  $P$  by:

$$\begin{aligned} x[x \mapsto Q] &= Q \\ y[x \mapsto Q] &= y \text{ if } (x \neq y) \\ (MN)[x \mapsto Q] &= M[x \mapsto Q]N[x \mapsto Q] \\ (\lambda x.M)[x \mapsto Q] &= \lambda x.M \\ (\lambda y.M)[x \mapsto Q] &= \lambda z.M[y \mapsto z][x \mapsto Q] \text{ if } (x \neq y) \end{aligned}$$

with  $z$  a variable defined by:

1. If  $x \notin FV(N)$  or  $y \notin FV(M)$  then  $z = y$
2. Otherwise,  $z$  can be any variable such that  $z \notin FV(N)$  or  $z \notin FV(M)$

The variable ' $z$ ' always exists. Choosing a fresh variable for  $z$  will always satisfy condition two. It might not be clear why need to use the variable  $z$  and one can expect that the following rule is good enough:

$$(\lambda y.M)[x \mapsto Q] = \lambda z.M[x \mapsto Q] \text{ if } (x \neq y)$$

However, the following example will illustrate why the more complicated rule is necessary. Consider the substitution  $(\lambda y.xy)[x \mapsto y]$ . The free variable of  $\lambda$ -term  $\lambda y.xy$  is  $x$  and the bound variable is  $y$ . If we use the simple rule above, we get that  $(\lambda y.xy)[x \mapsto y] = \lambda y.yy$ , the meaning of the  $\lambda$ -term changes. However, the free variable we substituted, is not free anymore. The variable  $y$  is not a free variable in the term that we substituted  $y$  in. Therefore, we change the bound variable  $y$  in  $\lambda y.xy$  to a new variable  $z$ . We can pick  $z$  as  $z$  is not a free variable in  $xy$ . Following the more complex rule, we get:

$$\begin{aligned} (\lambda y.xy)[x \mapsto y] &= \lambda z.(xy)[y \mapsto z][x \mapsto y] \\ &= \lambda z.yz \end{aligned}$$

### 3.2.3 $\alpha$ equivalence

Here, we define  $\alpha$  equivalence. The idea is that renaming bound variables does not change the meaning of a  $\lambda$ -term. For example, the  $\lambda$ -terms  $\lambda x.x$  and  $\lambda y.y$  are equivalent ( $\lambda x.x \equiv_a \lambda y.y$ ), since renaming all bound variables can result in identical  $\lambda$ -terms. The  $\lambda$ -terms  $\lambda x.x$  and  $\lambda y.y$  represent the same idea. We can define the relation  $P \equiv_a P'$  inductively on  $P$  and  $P'$  as follows:

$$\begin{aligned} x \equiv_a y & \quad \text{if } x = y \\ MN \equiv_a M'N' & \quad \text{if } M \equiv_a M' \text{ and } N \equiv_a N' \\ \lambda x.M \equiv_a \lambda y.M' & \quad \text{if } M \equiv_a M'[y \mapsto x] \text{ with } x = y \text{ or } x \notin FV(M') \end{aligned}$$

The last rule states that abstractions  $\lambda x.M$  and  $\lambda y.M'$  are equivalent if  $M$  and  $M'$  are equivalent after substituting  $y$  for  $x$  in  $M'$ . However,  $x$  should be equal to  $y$  or  $x$  should not be a free variable in  $M'$ .

- $\lambda x.zx$  is equivalent to  $\lambda y.zy$ , since  $zy[y \mapsto x] = zx \equiv_a zx$  and  $x$  is not a free variable in  $zy$ .
- $\lambda z.zx$  is not to  $\lambda y.zy$ , since  $z$  is a free variable in  $zy$  and  $x$  is not equal to  $y$ .

### 3.2.4 $\beta$ -Reduction

Now, we are ready to discuss  $\beta$ -reduction. In the  $\lambda$ -calculus, there is one way to simplify terms, which is  $\beta$ -reduction. We can define:

$$(\lambda x.M)N \rightarrow M[x \mapsto N] \quad (\beta)$$

So a  $\lambda$ -term of the form  $(\lambda x.M)N$  can be reduced by substituting  $x$  by  $N$  in  $M$ . However, without additional rules, we cannot apply  $\beta$ -reduction to subterms. For example, the following reduction would not be possible:  $\lambda x.(\lambda y.y)x \rightarrow \lambda x.x$ . Therefore, we can define the following rules:

$$\frac{MN \rightarrow M'N}{M \rightarrow M'} (\mu) \quad \frac{MN \rightarrow MN'}{N \rightarrow N'} (\nu) \quad \frac{\lambda x.M \rightarrow \lambda x.M'}{M \rightarrow M'} (\xi)$$

Now, we define  $\rightarrow_\beta$  as  $\beta$  closed under  $\mu$ ,  $\nu$  and  $\xi$ . So with  $\rightarrow_\beta$  we can use  $\beta$ -reduction on all subterms if the subterm is of the form  $(\lambda x.M)N$ . A subterm of the form  $(\lambda x.M)N$  is called a  $\beta$ -redex. In one reduction, we may need to use multiple rules. The  $\beta$ -redex that is reduced is underlined. For instance, we need to use the  $\mu$  and  $\xi$  rule in the last example of the following reductions:

$$\begin{array}{ll}
(\underline{\mathbf{II}})(\mathbf{II}) \rightarrow_{\beta} \mathbf{I}(\mathbf{II}) & \beta \text{ with } \mu \text{ rule} \\
(\mathbf{II})(\underline{\mathbf{II}}) \rightarrow_{\beta} (\mathbf{II})\mathbf{I} & \beta \text{ with } \nu \text{ rule} \\
\lambda x. \underline{\mathbf{I}x} \rightarrow_{\beta} \lambda x. x & \beta \text{ with } \xi \text{ rule} \\
(\mathbf{I}(\lambda x. \underline{\mathbf{I}x}))(\mathbf{II}) \rightarrow_{\beta} (\mathbf{I}(\lambda x. x))(\mathbf{II}) & \beta \text{ with } \mu \text{ and } \xi \text{ rule}
\end{array}$$

### 3.3 Call-by-name calculus

In this section, we discuss the call-by-name  $\lambda$ -calculus. The reduction rule that is used in this calculus is the same as the  $\beta$  reduction rule. However, we name the rule  $\beta_n$  to make it clear that it is the reduction rule of the call-by-name  $\lambda$ -calculus.

$$(\lambda x. M)N \rightarrow M[x \mapsto N] \quad \beta_n$$

We can define weak reduction,  $\rightarrow_w$ , as  $\beta_n$  closed under  $\mu$  and  $\nu$ . So  $\rightarrow_w$  has the only restriction that it cannot reduce under  $\lambda$ 's. The relation  $\rightarrow_n$  is defined as  $\beta_n$  closed under  $\mu$ . So using name evaluation, we can not evaluate the argument of a function. Call-by-name evaluation is defined as  $\rightarrow_n^*$ . In the following example, we give the call-by-name evaluation of the  $\lambda$ -term  $(\mathbf{I}(\lambda x. \mathbf{I}x))(\mathbf{II})$ . The  $\beta$ -redex that is reduced is underlined. Note that  $\rightarrow_n$  is deterministic, so there is only one way to apply  $\rightarrow_n$  on a  $\lambda$ -term.

$$\begin{array}{l}
(\underline{\mathbf{I}(\lambda x. \mathbf{I}x)})(\mathbf{II}) \rightarrow_n (\lambda x. \underline{\mathbf{I}x})(\mathbf{II}) \\
\rightarrow_n \underline{\mathbf{I}(\mathbf{II})} \\
\rightarrow_n \underline{\mathbf{II}} \\
\rightarrow_n \mathbf{I}
\end{array}$$

### 3.4 Call-by-value calculus

Before we can define the call-by-value  $\lambda$ -calculus (abbreviated as  $\lambda_v$ ), we first define what we mean by values. Values are all  $\lambda$ -terms that are either a variable or an abstractions. Values are usually denoted by  $V$  or  $W$ .

$$V ::= x \mid \lambda x. M$$

In the  $\lambda_v$ , we can only reduce if the argument is a value. Therefore, the reduction rule is as follows:

$$(\lambda x.M)V \rightarrow M[x \mapsto V] \quad \beta_v$$

In order to define the  $\rightarrow_v$  relation, we first define the rule  $\nu_{<}$ .

$$\frac{N \rightarrow N'}{VN \rightarrow VN'} \nu_{<}$$

The relation  $\rightarrow_v$  can be defined as  $\beta_v$ , closed under  $\mu$  and  $\nu_{<}$ . The  $\nu_{<}$  forces evaluation from left to right and  $\beta_v$  makes sure that the argument needs to be a value. In the following example, we give the call-by-value evaluation of the  $\lambda$ -term  $(\mathbf{I}(\lambda x.\mathbf{I}x))(\mathbf{II})$ . The  $\beta$ -redex that is reduced is underlined. Note that  $\rightarrow_v$  is deterministic, so there is only one way to apply  $\rightarrow_v$  on a  $\lambda$ -term.

$$\begin{aligned} (\mathbf{I}(\lambda x.\mathbf{I}x))(\mathbf{II}) &\rightarrow_v (\lambda x.\mathbf{I}x)(\mathbf{II}) \\ &\rightarrow_v \frac{(\lambda x.\mathbf{I}x)\mathbf{I}}{\mathbf{II}} \\ &\rightarrow_v \mathbf{II} \\ &\rightarrow_v \mathbf{I} \end{aligned}$$

## Chapter 4

# Call-by-box Lambda Calculus

### 4.1 Specification of $\lambda_b$

The terms of  $\lambda_b$  are given by:

$$M, N, P, Q ::= \varepsilon(x) \mid \lambda x.M \mid MN \mid \text{box}(N)$$

Types are given by:

$$A, B ::= X \mid A \rightarrow B \mid \Box A$$

The typing rules of  $\lambda_b$  are give by:

$$\begin{array}{c} \frac{}{\Gamma, x : \Box A \vdash \varepsilon(x) : A} \text{Id}_{\Box} \\[10pt] \frac{\Gamma, x : A \vdash M : B}{\Gamma \vdash \lambda x.M : A \rightarrow B} \rightarrow\text{-I} \\[10pt] \frac{\Gamma \vdash M : A \rightarrow B \quad \Delta \vdash N : A}{\Gamma, \Delta \vdash MN : B} \rightarrow\text{-E} \\[10pt] \frac{\Gamma \vdash M : A}{\Gamma \vdash \text{box}(M) : \Box A} \Box\text{-I} \end{array}$$

Since variables always occur inside  $\varepsilon$ , we need to redefine free and bound variables and substitution. For  $\lambda_b$ , we have:

$$\begin{array}{llll} FV_b(\varepsilon(x)) & = \{\varepsilon(x)\} & BV(\varepsilon(x)) & = \emptyset \\ FV_b(\lambda x.M) & = FV(M) \setminus \{\varepsilon(x)\} & BV(\lambda x.M) & = BV(M) \cup \{\varepsilon(x)\} \\ FV_b(MN) & = FV(M) \cup FV(N) & BV(MN) & = BV(M) \cup BV(N) \end{array}$$



Thus, for substitution, we get:

$$\begin{aligned}
\varepsilon(x)[\varepsilon(x) \mapsto Q] &= Q \\
\varepsilon(y)[\varepsilon(x) \mapsto Q] &= \varepsilon(y) \text{ if } (x \neq y) \\
(MN)[\varepsilon(x) \mapsto Q] &= M[x \mapsto Q]N[x \mapsto Q] \\
(\lambda x.M)[\varepsilon(x) \mapsto Q] &= \lambda x.M \\
(\lambda y.M)[\varepsilon(x) \mapsto Q] &= \lambda z.M[\varepsilon(y) \mapsto \varepsilon(z)][\varepsilon(x) \mapsto Q] \text{ if } (x \neq y)
\end{aligned}$$

with  $z$  a variable defined by:

1. If  $\varepsilon(x) \notin FV_b(N)$  or  $\varepsilon(y) \notin FV_b(M)$  then  $z = y$
2. Otherwise,  $z$  can be any variable such that  $\varepsilon(z) \notin FV(N)$  or  $\varepsilon(z) \notin FV(M)$

The reduction rule we can use for this  $\lambda$ -calculus is:

$$(\lambda x.M)\text{box}(N) \rightarrow M[\varepsilon(x) \mapsto N] \quad (\beta_b)$$

We define two relations:  $\rightarrow_{b_n}$  and  $\rightarrow_{b_v}$ . The relation  $\rightarrow_{b_n}$  will be used for cbn evaluation for embeddings of  $\lambda_n$  in  $\lambda_b$ . The relation  $\rightarrow_{b_v}$  will be used for cbn evaluation for embeddings of  $\lambda_v$  in  $\lambda_b$ .

## 4.2 Embedding of $\lambda_n$ into $\lambda_b$

Let  $\Lambda_{CBN}$  be the set containing all  $\lambda$ -terms in the  $\lambda_n$  and let  $\Lambda_{CBB}$  be the set containing all  $\lambda$ -terms in the  $\lambda_b$ . The set  $\mathcal{T}_{CBN}$  contains all types of the  $\lambda_n$  and  $\mathcal{T}_{CBB}$  contains all types of the  $\lambda_b$ . Now we define functions from  $\Lambda_{CBN}$  to  $\Lambda_{CBB}$  for term translation and from  $\mathcal{T}_{CBN}$  to  $\mathcal{T}_{CBB}$  for type translation. These functions represent an embedding of the  $\lambda_n$  into the  $\lambda_b$ .

$$\begin{array}{llll}
T_t & : \mathcal{T}_{CBN} \rightarrow \mathcal{T}_{CBB} & T & : \Lambda_{CBN} \rightarrow \Lambda_{CBB} \\
T_t(X) & = X & T(x) & = \varepsilon(x) \\
T_t(A \rightarrow B) & = \Box T_t(A) \rightarrow T_t(B) & T(\lambda x.M) & = \lambda x.T(M) \\
& & T(MN) & = T(M)\text{box}(T(N))
\end{array}$$

To clarify, we give the embedding of  $\mathbf{I}x$  into  $\lambda_b$  after which we apply  $\rightarrow_{b_n}$ . It is clear that the result obtained after  $\rightarrow_{b_n}$  is equal to the embedding of the result of applying  $\rightarrow_{b_n}$  on  $\mathbf{I}x$  in the  $\lambda_n$ , which is  $T(x) = \varepsilon(x)$ .

$$\begin{aligned}
T(\mathbf{I}x) &= (T(\lambda x.x))\text{box}(T(x)) \\
&= (\lambda x.T(x))\text{box}(T(x)) \\
&= (\lambda x.\varepsilon(x))\text{box}(\varepsilon(x)) \\
&\rightarrow_{b_n} (\lambda x.\varepsilon(x))[\varepsilon(x) \mapsto \varepsilon(x)] \\
&= \varepsilon(x)
\end{aligned}$$

### 4.3 Embedding of $\lambda_v$ into $\lambda_b$

For the embedding of the  $\lambda_v$  into the  $\lambda_b$ , we make use of the abbreviation "raise" defined by:

$$\text{raise}(M) := \lambda z.\varepsilon(z)M$$

Let  $\Lambda_{CBV}$  be the set containing all  $\lambda$ -terms in the  $\lambda_v$  and let  $\Lambda_{CBB}$  be the set containing all  $\lambda$ -terms in the  $\lambda_b$ . The set  $\mathcal{T}_{CBV}$  contains all types of the  $\lambda_v$  and  $\mathcal{T}_{CBB}$  contains all types of the  $\lambda_b$ . Now we define functions from  $\Lambda_{CBV}$  to  $\Lambda_{CBB}$  for term translation and from  $\mathcal{T}_{CBV}$  to  $\mathcal{T}_{CBB}$  for type translation.

$$\begin{array}{ll}
T_t & : \mathcal{T}_{CBV} \rightarrow \mathcal{T}_{CBB} \\
T_t(X) & = \Box X \\
T_t(A \rightarrow B) & = \Box(\Box T_t(A) \rightarrow \Box T_t(B)) \\
T & : \Lambda_{CBN} \rightarrow \Lambda_{CBB} \\
T(x) & = \text{box}(\varepsilon(x)) \\
T(\lambda x.M) & = \text{box}(\lambda x.T(M)) \\
T(MN) & = \text{raise}(T(N))T(M)
\end{array}$$

To clarify, we give the embedding of  $\mathbf{I}x$  into  $\lambda_b$  after which we apply  $\rightarrow_{b_v}$  twice. It is clear that the result obtained after  $\rightarrow_{b_v}$  is equal to the embedding of the result of applying  $\rightarrow_v$  on  $\mathbf{I}x$  in the  $\lambda_v$ , which is  $T(x) = \text{box}(\varepsilon(x))$ .

$$\begin{aligned}
T(\mathbf{I}x) &= \text{raise}(T(x))T(\mathbf{I}) \\
&= \text{raise}(T(x))T(\lambda x.x) \\
&= \text{raise}(\text{box}(\varepsilon(x)))\text{box}(\lambda x.T(x)) \\
&= \text{raise}(\text{box}(\varepsilon(x)))\text{box}(\lambda x.\text{box}(\varepsilon(x))) \\
&= (\lambda z.\varepsilon(z)\text{box}(\varepsilon(x)))\text{box}(\lambda x.\text{box}(\varepsilon(x))) \\
&\rightarrow_{b_v} (\lambda x.\text{box}(\varepsilon(x)))\text{box}(\varepsilon(x)) \\
&\rightarrow_{b_v} \text{box}(\varepsilon(x))
\end{aligned}$$

## Chapter 5

# Related Work

In this chapter you demonstrate that you are sufficiently aware of the state-of-art knowledge of the problem domain that you have investigated as well as demonstrating that you have found a *new* solution / approach / method.

## Chapter 6

# Conclusions

In this chapter you present all conclusions that can be drawn from the preceding chapters. It should not introduce new experiments, theories, investigations, etc.: these should have been written down earlier in the thesis. Therefore, conclusions can be brief and to the point.

# Bibliography

- [1] Simon Peyton Jones. How to write a good research paper, 2004. Presentation at Technical University of Vienna, <http://research.microsoft.com/en-us/um/people/simonpj/papers/giving-a-talk/writing-a-paper-slides.pdf>.

## Appendix A

# Appendix

Appendices are *optional* chapters in which you cover additional material that is required to support your hypothesis, experiments, measurements, conclusions, etc. that would otherwise clutter the presentation of your research.