A Study of the Explicit and Implicit Modal λ-Calculi, Their Non-Contextual Translations and the Contextual Explicit Modal λ-Calculus

Gulliver Häger

August 11, 2022

Abstract

In this report, a survey of the dual-zone modal λ-calculus and the Kripke-style modal λ-calculus introduced by Davies and Pfenning has been undertaken. A complete technical presentation of some of their proofs regarding the translations between the two systems have also been included. In addition, the contextual modal λ-calculus by Nanevski, Pfenning and Pientka has been studied and the notion of raising and lowering in this system have been further developed.

Supervised by
Brigitte Pientka

School of Computer Science
McGill University
Montréal, QC, Canada

Contents

1 Introduction

In this section, I give some brief context of how this project was undertaken, what the goal of the project was and where the project ultimately concluded.

Together with Professor Pientka at the School of Computer Science at McGill University, I applied for the NSERC Undergraduate Student Research Award (USRA) February 2022 and some months later I received the award. Together with some funding on Pientka's part, I was able to commit my full attention during the summer of 2022 towards this project which taught me valuable research skills and working more self-sufficiently. For this I want to thank both NSERC and Pientka.

At the time of conceiving of a research project together with Professor Pientka, I had taken her course COMP302, Programming Languages and Paradigms, and was taking her course COMP523, Language Based Security. While the former is an undergraduate course aimed to introduce students to functional programming languages, the latter is a graduate level course meant to give students a solid foundation in functional programming language design. At the time of approaching Pientka, I was sure I wanted to explore research in functional programming language design, but was not knowledgeable enough to suggest a topic of my own. When I approached her, she suggested the topic of meta-programming which was related to her current research with Hu[HP22], and which sounded interesting to me.

At the beginning of this project, Professor Pientka and I had a clear idea of what I was to study and pursue during the duration of the project. This idea is best summarized from the "student's role" section of the NSERC USRA application which we submitted in February 2022 and is shown below.

> "The student will first design a Kripke-style formulation of modal lambda-calculus that supports open code building on recent work by Hu and Pientka and Nanevski, Pfenning, and Pientka. The student will then study substitution properties and design a translation of this Kripke-stye formulation to the dual-zone modal lambda-calculus where evaluation order is explicit and prove soundness and completeness of that translation."

The first half of the project was spent studying the material already published on the topic, such as the paper by Davies and Pfenning[DP01] first introducing the idea of modelling meta-programs using modal logic.

Towards the second half of the summer, the area of focus of the project shifted and the original goal of investigating a translation between the contextual Kripke-style system (referred to as implicit) and the contextual dual-zone modal lambda calculus (referred to as explicit) was not pursued due to a lack of time. Instead, the notion of raising and lowering - which was meant to be a part of the construction of the desired translation - was further investigated and took more time than expected to fully understand.

In the end, I was able to gain a deep understanding of the explicit and implicit, introduced in sections 2 and 3, as well as the translations between the two systems, introduced in sections 4 and 5. Moreover, I was also able to gain an understanding of a contextual formulation of the explicit system, introduced in section 6, as well as further develop the idea of raising and lowering proof terms to translate terms from the contextual and non-contextual setting.

2 The Non-Contextual Explicit Modal λ-Calculus

In this section, a theoretical foundation for the study of meta-programming is introduced. This foundation is the **explicit modal λ-calculus**, first introduced by Pfenning and Davies[DP01] and denoted by $\lambda_e^{\to\Box}$. In their paper, Pfenning and Davies give a detailed description and motivation of their development of the calculus, which for the sake of brevity have been shortened here. Although this report focuses primarily on the type theory of the calculus, the operational semantics of the λ-fragment of a mini-ML language based on $\lambda_e^{\to\Box}$ have also been included for the sake of completeness.

2.1 Syntax of the Non-Contextual Explicit Modal λ-Calculus

Below is a summary of the syntax of $\lambda_e^{\to\Box}$. Here, lower case types refer to base types.

As can be seen in the syntax, there are two distinct contexts, the ordinary context and the modal context. The ordinary context contains propositions which are assumed to be **true**, labeled with ordinary variables, whereas the modal context contains propositions which are **valid** and labeled with modal variables. While the concept of a true proposition is familiar, validity is a new concept meant to model propositions which are true under all possible interpretations - meaning that the truth of a valid proposition cannot depend on any hypotheses about the truth of other propositions. As will be seen in the next section in which the typing judgments are given, validity is used to express code itself, making the concept useful for expressing meta-programs, since it adds the expressive power of being able to write programs which themselves manipulate code.

Types	A, B	$:= a \mid A \to B \mid \Box A$
Terms	E	$:= x \mid \lambda x{:}A.E \mid E_1 E_2$
		$u \mid \textbf{box}\ E \mid \textbf{let box}\ u = E_1\ \text{in}\ E_2$
Ordinary Contexts	Γ	$:= \cdot \mid \Gamma, x{:}A$
Modal Contexts	Δ	$:= \cdot \mid \Delta, u{:}A$

From this point onward, standard syntactic conventions are assumed. These are listed below.

- Using $x, y, z...$ for ordinary variables
- Using $u, v, w, ...$ for modal variables
- Variables can only be declared once in a context
- Bound variables can be renamed tacitly
- Leading empty contexts can be omitted
- $[E_1/x]E_2$ denotes replacing x for E_1 in E_2

2.2 Typing Rules of the Non-Contextual Explicit λ-Calculus

Below are the typing rules for $\lambda_e^{\to\Box}$. Notice that the turnstile of the typing judgement is superscripted with e in order to differentiate from the implicit calculus presented in the following section.

$$\frac{x{:}A \in \Gamma}{\Delta;\Gamma \vdash^e x{:}A}\ \text{ovar} \qquad \frac{u{:}A \in \Delta}{\Delta;\Gamma \vdash^e u{:}A}\ \text{mvar}$$

$$\frac{\Delta;(\Gamma, x{:}A) \vdash^e E{:}B}{\Delta;\Gamma \vdash^e \lambda x{:}A.E{:}(A \to B)} \to I \qquad \frac{\Delta;\cdot \vdash^e E{:}A}{\Delta;\Gamma \vdash^e \mathbf{box}\ E{:}\Box A}\ \Box I$$

$$\frac{\Delta;\Gamma \vdash^e E_1{:}(B \to A) \quad \Delta;\Gamma \vdash^e E_2{:}B}{\Delta;\Gamma \vdash^e E_1\ E_2{:}A} \to E \qquad \frac{\Delta;\Gamma \vdash^e E_1{:}\Box A \quad (\Delta, u{:}A);\Gamma \vdash^e E_2{:}B}{\Delta;\Gamma \vdash^e \mathbf{let\ box}\ u = E_1\ \mathbf{in}\ E_2{:}B}\ \Box E$$

The typing judgement is similar to the **ordinary simply typed λ-calculus** with the main difference that there are two contexts - the modal context and the ordinary context. The main aim of $\lambda_e^{\to\Box}$ is to extend the ordinary simply typed λ-calculus so that it can also model **meta-programs**. This is achieved through expressing **validity** through the Curry-Howard Isomorphism by basing the typing rules in a **modal logic with frame conditions S4**, allowing for "boxed terms" which can be thought of as closed programs. The ordinary context behaves the same as in the ordinary simply typed setting, except for the introduction of a boxed (valid) term in which it is required that the ordinary context is empty. As discussed in the previous subsection, this is because valid terms cannot depend on the assumption of the truth of any other propositions - which is the case for any proposition in the ordinary context. The modal context is not required to be empty in this case, since the modal context only contain valid assumptions - meaning that they are true at any time. Any such modal variable can be used in place of an actual program representing a program of a specific type and then be substituted by an actual closed program using the $\Box E$ rule.

2.3 Operational Semantics of a $\lambda_e^{\to\Box}$-Fragment

To give an intuition of how $\lambda_e^{\to\Box}$ could be used in practice, a shortened version of the operational semantics of a mini-ML language based on $\lambda_e^{\to\Box}$,introduced by Pfenning and Davies in their paper, have been included in this section. The semantics are **call-by-value**, meaning that value first have to be defined, which is done below.

$$\text{Values} \quad V ::= \lambda x{:}A.e \mid \mathbf{box}\ E$$

The operational semantics is defined using the following judgement.

$$E \hookrightarrow V \quad \text{Expression } E \text{ evaluates to value } V$$

The operational semantics are now given below. Functions behave as expected and since boxed terms are values, they only evaluate to themselves. The most insightful part of the operational semantics is how to evaluate a **let box**-term. Assuming that the term **let box** $u = E_1$ **in** E_2 is well-typed, it is known that E_1 has type $\Box A$. Thus, to evaluate the term, the code generated by evaluating E_1 is substituted for u in E_2, all of which is then evaluated to produce the value from the original **let box**-term. This evaluation strategy is in line with the goal of modelling meta-programs using modality, since it models using a real piece of code (E_1) in place of some variable (u) representing code of some known type in another term (E_2).

$$\frac{}{\lambda x{:}A.E \hookrightarrow \lambda x{:}A.E}\ \text{ev\_lam} \qquad \frac{E_1 \hookrightarrow \lambda x{:}E_1' \quad E_2 \hookrightarrow V_2 \quad [V_2/x]E_1' \hookrightarrow V}{E_1 E_2 \hookrightarrow V}\ \text{ev\_app}$$

$$\frac{}{\mathbf{box}\ E \hookrightarrow \mathbf{box}\ E}\ \text{ev\_box} \qquad \frac{E_1 \hookrightarrow \mathbf{box}\ E_1' \quad [E_1'/u]E_2 \hookrightarrow V_2}{\mathbf{let\ box}\ u = E_1\ \mathbf{in}\ E_2 \hookrightarrow V_2}\ \text{ev\_let\_box}$$

2.4 An Example Derivation

Below is an example of a derivation proving the proposition $\Box(A \to B) \to (\Box A \to \Box B)$ which corresponds to the distributive axiom in modal logic. In terms of a meta-programming view, the derivation shows that a program of a function type can be used to produce a function which takes a program of the type of the input of the original program and returns a program of the original result.

$$\dfrac{\dfrac{\dfrac{\dfrac{}{\cdot;(x{:}\Box(A \to B),\ y{:}\Box A) \vdash x{:}\Box(A \to B)}\ \text{ovar} \qquad \mathcal{D}_1}{\cdot;(x{:}\Box(A \to B),\ y{:}\Box A) \vdash \mathbf{let\ box}\ u = x\ \mathbf{in\ let\ box}\ v = y\ \mathbf{in\ box}\ uv{:}\Box B}\ \Box E}{\cdot;(x{:}\Box(A \to B)) \vdash \lambda y{:}\Box A.(\mathbf{let\ box}\ u = x\ \mathbf{in\ let\ box}\ v = y\ \mathbf{in\ box}\ uv){:}\Box A \to \Box B}\ \to I}{\cdot;\cdot \vdash \lambda x{:}\Box(A \to B).\lambda y{:}\Box A.(\mathbf{let\ box}\ u = x\ \mathbf{in\ let\ box}\ v = y\ \mathbf{in\ box}\ uv){:}\Box(A \to B) \to (\Box A \to \Box B)}\ \to I$$

Here, the subtree $\mathcal{D}_1$ has been omitted for formatting reasons and is instead shown below.

$$\dfrac{\dfrac{\dfrac{(u{:}A \to B,\ v{:}A);\cdot \vdash u{:}A \to B \qquad (u{:}A \to B,\ v{:}A);\cdot \vdash v{:}A}{(u{:}A \to B,\ v{:}A);\cdot \vdash u\ v{:}B}\ \to E}{(u{:}A \to B,\ v{:}A);(x{:}\Box(A \to B),\ y{:}\Box A) \vdash \mathbf{box}\ uv{:}\Box B}\ \Box I \qquad \dfrac{}{(u{:}A \to B);(x{:}\Box(A \to B),\ y{:}\Box A) \vdash y{:}\Box A}\ \text{ovar}}{(u{:}A \to B);(x{:}\Box(A \to B),\ y{:}\Box B) \vdash \mathbf{let\ box}\ v = y\ \mathbf{in\ box}\ uv{:}\Box B}\ \Box E$$

3 The Non-Contextual Implicit Modal λ-Calculus

While the development of $\lambda_e^{\to\Box}$ was based in the theoretic and logical motivation of capturing **valid** assumptions, this section is concerned with presenting Pfenning and Davies' λ-calculus corresponding to Kripke's multiple-world interpretation of modal logic[DP01]. In this report, this calculus will be referred to as $\lambda_i^{\to\Box}$.

The implicit calculus does not separate modal and ordinary variables and contexts and instead expresses validity through **worlds** - which correspond to a stage of computation. In a Kripke interpretation of modal logic, the truth of a proposition is relative to a world which is represented by a context of hypotheses containing assumptions known to be true in the current world. The modal operator $\Box$ then allows for reasoning about the truth of a proposition in all **accessible worlds** from the current world. What worlds are accessible from the current world depends on the frame condition of the modal logic - which is assumed to be S4 in order to model meta-programming.

Accessibility of worlds is modelled through context stacks, where a succeeding context represents any world accessible from the preceding world.

3.1 Syntax of the Non-Contextual Implicit λ-Calculus

Below is a summary of the syntax of $\lambda_i^{\to\Box}$. Going forward, $\odot$ is used for an empty context stack to differentiate it from an empty context.

Types	A, B	$::= a \mid A \to B \mid \Box A$
Terms	E	$::= x \mid \lambda x{:}A.M \mid M_1 M_2 \mid \mathbf{box}\ M \mid \mathbf{unbox}_n M$
Ordinary Contexts	Γ	$::= \cdot \mid \Gamma, x{:}A$
Context Stacks	Ψ	$::= \odot \mid \Psi; \Gamma$

3.2 Typing Rules of the Non-Contextual Implicit λ-Calculus

Below are the typing rules for $\lambda_i^{\to\Box}$. Once again, notice that the turnstile of the typing judgment is superscripted with i in order to differentiate it from the typing judgment of the explicit calculus.

$$\frac{x{:}A \in \Gamma}{\Psi; \Gamma \vdash^i x{:}A}\ \text{var}$$

$$\frac{\Psi; (\Gamma, x{:}A) \vdash^i M{:}B}{\Psi; \Gamma \vdash^i \lambda x{:}A.M{:}(A \to B)} \to I \qquad \frac{\Psi; \Gamma; \cdot \vdash^i M{:}A}{\Psi; \Gamma \vdash^i \mathbf{box}\ M{:}\Box A}\ \Box I$$

$$\frac{\Psi; \Gamma \vdash^i M{:}(B \to A) \quad \Psi; \Gamma \vdash^i N{:}B}{\Delta; \Gamma \vdash^e M\ \ N{:}A} \to E \qquad \frac{\Psi; \Gamma \vdash^i M{:}\Box A}{\Psi; \Gamma; \Gamma_1; ...; \Gamma_n \vdash^i \mathbf{unbox}_n M{:}A}\ \Box E$$

Since the top-most context of the context stack in the typing judgment represents the current world, the first three rules concerning variables and functions behave as expected. The $\Box I$-rule is motivated by the fact that a term of typ $\Box A$ should be true in the current world if A is true in all worlds accessible from the current world. Since no information of what worlds are reachable

from the current world is known, no information about the top-most context in the context-stack in the antecedent can be known. In the case of the $\Box E$-rule, the boxed term is known to be true in all accessible worlds. Recall that S4 frame condition consists of transativity and reflexivity. In the elimination rules, Γ_1 would be accessible in any frame condition by definition. In the current setting, $\mathbf{unbox}_0$ is possible because of reflexivity since this means that Γ is accessed from itself and $\Gamma_2, ..., \Gamma_n$ are all accessible from Γ because of transativity. from the current world, giving the variable amount of additional worlds in the context stack of the consequent of the judgment.

3.3 A Sample Derivation

The same proposition proven in section 2.4 is shown below. Notice the similarity of the proof terms. It will be shown later that the translations introduced in the following sections indeed map these two terms to each other.

$$\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{}{(x{:}\Box(A \to B),\ y{:}\Box A) \vdash x{:}\Box(A \to B)}\ \text{var}}{(x{:}\Box(A \to B),\ y{:}\Box A); \cdot \vdash (\mathbf{unbox}_1 x){:}A \to B}\ \Box E \quad \dfrac{\dfrac{}{(x{:}\Box(A \to B),\ y{:}\Box A) \vdash y{:}\Box A}\ \text{var}}{(x{:}\Box(A \to B),\ y{:}\Box A); \cdot \vdash (\mathbf{unbox}_1 y){:}A}\ \Box E}{(x{:}\Box(A \to B),\ y{:}\Box A); \cdot \vdash (\mathbf{unbox}_1 x)(\mathbf{unbox}_1 y){:}B}\ \to E}{(x{:}\Box(A \to B),\ y{:}\Box A) \vdash \mathbf{box}\ ((\mathbf{unbox}_1 x)(\mathbf{unbox}_1 y)){:}\Box B}\ \Box I}{(x{:}\Box(A \to B)) \vdash \lambda y{:}\Box A.\mathbf{box}\ ((\mathbf{unbox}_1 x)(\mathbf{unbox}_1 y)){:}\Box A \to \Box B}\ \Box I}{\cdot \vdash \lambda x{:}\Box(A \to B).\lambda y{:}\Box A.\mathbf{box}\ ((\mathbf{unbox}_1 x)(\mathbf{unbox}_1 y)){:}\Box(A \to B) \to (\Box A \to \Box B)}\ \Box I$$

4 Translation From the Explicit to the Implicit System

In this section a translation from $\lambda_e^{\to\Box}$ to $\lambda_i^{\to\Box}$ is given and in the following section a translation in the opposite direction is given. Together with the theorems of both sections, it is shown that that $\lambda_e^{\to\Box}$ and $\lambda_i^{\to\Box}$ are equally expressive and have a one-to-one correspondence in proof terms.

4.1 Definition of Environments

In order to define the translation of proof terms from $\lambda_e^{\to\Box}$ to $\lambda_i^{\to\Box}$, **environments** and **environment stacks** have to be defined. The syntax is as follows.

$$\begin{aligned} \text{Environments} \quad & \rho := \cdot \mid \rho, \mathbf{box}\ u = E \\ \text{Environment Stacks} \quad & R := \odot \mid R; \rho \end{aligned}$$

The main purpose of environments are to relate context pairs $\Delta;\Gamma$ in the explicit system to context stacks Ψ in the implicit system. The judgments used to do this are shown below. Here, Θ is used as a placeholder for any modal context, possibly empty.

$$\begin{aligned} \Delta;\Gamma \vdash^e \rho{:}\Theta \quad & \text{Environments } \rho \text{ matches } \Theta \text{ in context pair } \Delta;\Gamma \\ \Psi \models^e R{:}\Delta \quad & \text{Environment stack } R \text{ matches } \Delta \text{ in context stack } \Psi \end{aligned}$$

The rules for the judgments are shown below.

$$\frac{}{\Delta;\Gamma \vdash^e \cdot{:}\cdot}\ \text{tpv\_empty} \qquad \frac{\Delta;\Gamma \vdash^e \rho{:}\Theta \quad (\Delta,\Theta);\Gamma \vdash^e E{:}\Box A}{\Delta;\Gamma \vdash^e (\rho, \mathbf{box}\ u = E){:}(\Theta, u{:}A)}\ \text{tpv\_bind}$$

$$\frac{}{\Psi \models^e \odot{:}\cdot}\ \text{tpr\_empty} \qquad \frac{\Psi \models^e R{:}\Delta \quad \Delta;\Gamma \vdash^e \rho{:}\Theta}{\Psi;\Gamma \models^e (R;\rho){:}(\Delta,\Theta)}\ \text{tpr\_env}$$

Related to environments is the following lemma, which the correctness proof of the translation will depend on.

Lemma 4.1.1 (Environment Extension): If $\Psi;\Gamma \models^e (R;\rho){:}\Delta$ and $\Delta;\Gamma \vdash^e E{:}\Box A$, then $\Psi;\Gamma \models^e (R;\ \rho, \mathbf{box}\ u = E){:}(\Delta, u{:}A)$.

Proof: By inversion on the derivation of the environment, call it $\mathcal{D}$.

$$\mathcal{D} = \frac{\Psi \models^e R{:}\Delta' \quad \Delta';\Gamma \vdash^e \rho{:}\Theta}{\Psi;\Gamma \models^e (R;\rho){:}(\Delta',\Theta)}\ \text{tpr\_env}$$

(1) $\Delta = \Delta', \Theta$	By case
(2) $(\Delta',\Theta);\Gamma \vdash^e E{:}\Box A$	By ass. and (1)
(3) $\Delta';\Gamma \vdash^e \rho{:}\Theta$	By inversion on $\mathcal{D}$
(4) $\Delta';\Gamma \vdash^e (\rho,\ \mathbf{box}\ u = E){:}(\Theta, u{:}A)$	By tpv_bind using (2),(3)
(5) $\Psi \models^e R{:}\Delta'$	By inversion on $\mathcal{D}$
(6) $\Psi;\Gamma \models^e (R;\ \rho, \mathbf{box}\ u = E){:}(\Delta', \Theta, u{:}A)$	By tpr_env using (4),(5)

■

4.2 Translation Judgements

The translation of proof terms from $\lambda_e^{\to\Box}$ to $\lambda_i^{\to\Box}$ is based on the judgment below.

$$R;\rho \rhd E \mapsto M \quad \text{Expression } E \text{ translates to } M \text{ in environment stack } R;\rho$$

The rules of the translation are as shown below.

$$\frac{}{R;\rho \rhd x \mapsto x}\ \text{tx\_ovar} \qquad \frac{R;\rho'_n \rhd E \mapsto M}{R;(\rho'_n, \mathbf{box}\ u = E, \rho''_n);\rho_{(n-1)};\cdots;\rho_0 \rhd u \mapsto \mathbf{unbox}_n M}\ \text{tx\_mvar}$$

$$\frac{R;\rho \rhd E \mapsto M}{R;\rho \rhd \lambda x{:}A.E \mapsto \lambda x{:}A.M}\ \text{tx\_lam} \qquad \frac{R;\rho \rhd E_1 \mapsto M_1 \quad R;\rho \rhd E_2 \mapsto M_2}{R;\rho \rhd E_1\ E_2 \mapsto M_1\ M_2}\ \text{tx\_app}$$

$$\frac{R;\rho\,;\cdot \rhd E \mapsto M}{R;\rho \rhd \mathbf{box}\ E \mapsto \mathbf{box}\ M}\ \text{tx\_box} \qquad \frac{R;(\rho, \mathbf{box}\ u = E_1) \rhd E_2 \mapsto M}{R;\rho \rhd \mathbf{let\ box}\ u = E_1\ \mathbf{in}\ E_2 \mapsto M}\ \text{tx\_letbox}$$

4.3 Translation Theorem

Now the theorem of the translation from the explicit system to the implicit system can be given along with its proof. In the proof below, $\exists!$ is shorthand for "there exists a unique".

Theorem 4.3.1: Given $\Psi;\Gamma \models^e (R;\rho){:}\Delta$ and $\Delta;\Gamma \vdash^e E{:}A$, then:
(1) There is a unique M such that $(R;\rho) \rhd E \mapsto M$
(2) Whenever $(R;\rho) \rhd E \mapsto M$, then $\Psi;\Gamma \vdash^i M{:}A$.

Proof of Proposition 1: By simultaneous induction on the derivations of $\Psi;\Gamma \models^e (R;\rho){:}\Delta$ and $\Delta;\Gamma \vdash^e E{:}A$, call them $\mathcal{D}_1$ and $\mathcal{D}_2$ respectively.

Case 1: $\mathcal{D}_2 = \dfrac{(x{:}A) \in \Gamma}{\Delta;\Gamma \vdash^e x{:}A}\ \text{ovar}$

The result follows directly from tx_ovar.

Case 2: $\mathcal{D}_2 = \dfrac{\Delta;(\Gamma, x{:}A) \vdash^e E{:}B}{\Delta;\Gamma \vdash^e \lambda x{:}A.E{:}(A \to B)} \to I$

(1) $\Psi;\Gamma \models^e (R;\rho){:}\Delta$ — By ass.
(2) $\Psi;(\Gamma, x{:}A) \models^e (R;\rho){:}\Delta$ — By weakening of (1)
(3) $\exists! M$ s.t. $(R;\rho) \rhd E \mapsto M$ — By IH on subtree of $\mathcal{D}_2$ using (2)
(4) $\exists! M' = \lambda x{:}A.M$ s.t. $(R;\rho \rhd \lambda x{:}A.E \mapsto M')$ — By tx_lam and (3)

Case 3: $\mathcal{D}_2 = \dfrac{\Delta;\Gamma \vdash^e E_1{:}(B \to A) \qquad \Delta;\Gamma \vdash^e E_2{:}B}{\Delta;\Gamma \vdash^e E_1\ \ E_2{:}A} \to E$

(1) $\Psi;\Gamma \models^e (R;\rho):\Delta$ By ass.
(2) $\exists! M_1$ s.t. $(R;\rho) \rhd E_1 \mapsto M_1$ By IH on left subtree of $\mathcal{D}_2$ using (1)
(3) $\exists! M_2$ s.t. $(R;\rho) \rhd E_2 \mapsto M_2$ By IH on left subtree of $\mathcal{D}_2$ using (1)
(4) $\exists! M = M_1\ M_2$ s.t. $(R;\rho) \rhd M$ By tx_app and (2),(3)

Case 4: $\mathcal{D}_2 = \dfrac{\Delta;\cdot \vdash^e E{:}A}{\Delta;\Gamma \vdash^e \mathbf{box}\ E{:}\Box A}\ \Box I$

(1) $\Psi;\Gamma \models^e (R;\rho):\Delta$ By ass.
(2) $\Delta;\cdot \vdash^e \cdot{:}\cdot$ By tpv_empty
(3) $\Psi;\Gamma;\cdot \models^e (R;\rho;\cdot):\Delta$ By tpv_env using (1),(2)
(4) $\exists! M$ s.t. $(R;\rho;\cdot) \rhd E \mapsto M$ By IH on subtree of $\mathcal{D}_2$ using (3)
(5) $\exists! M' = (\mathbf{box}\ M)$ s.t. $(R;\rho) \rhd \mathbf{box}\ E \mapsto M'$ By tx_box and (4)

Case 5: $\mathcal{D}_2 = \dfrac{\Delta;\Gamma \vdash^e E_1{:}\Box A \qquad (\Delta, u{:}A);\Gamma \vdash^e E_2{:}B}{\Delta;\Gamma \vdash^e \mathbf{let\ box}\ u = E_1\ \mathbf{in}\ E_2{:}B}\ \Box E$

(1) $\Psi;\Gamma \models^e (R;\rho):\Delta$ By ass.
(2) $(\Delta, u{:}A);\Gamma \vdash^e E_2{:}B$ By inversion on $\mathcal{D}_2$
(3) $\Psi;\Gamma \models^e R;(\rho, \mathbf{box}\ u = E_1):(\Delta, u{:}A)$ By lemma 4.1.1 using (1),(2)
(4) $\exists! M$ s.t. $R;(\rho, \mathbf{box}\ u = E_1) \rhd E_2 \mapsto M$ By IH on right subtree of $\mathcal{D}_2$ using (3)
(5) $\exists! M =$ s.t. $(R;\rho) \rhd (\mathbf{let\ box}\ u = E_1\ \mathbf{in}\ E_2) \mapsto M$ By tx_letbox and (4)

Case 6: $\mathcal{D}_2 = \dfrac{u{:}A \in \Delta}{\Delta;\Gamma \vdash^e u{:}A}\ \text{mvar}$

(1) $\Psi_0;(\Gamma_n;\cdots;\Gamma_0) \models^e (R';(\rho'_n, \mathbf{box}\ u = E, \rho_n);\cdots\rho_0):(\Theta,(\Theta'_n, u{:}A, \Theta''_n), ..., \Theta_0)$ By ass.
(2) $\Psi_0 \models^e R'{:}\Theta$ By inv. of tpv_env $(n+1)$ times on (1)
(3) $\Theta;\Gamma_n \vdash^e (\rho'_n, \mathbf{box}\ u = E, \rho_n):(\Theta'_n, u{:}A, \Theta''_n)$ By inv. of tpv_env $(n+1)$ times on (1)
(4) $\Theta;\Gamma_n \vdash^e (\rho'_n, \mathbf{box}\ u = E):(\Theta'_n, u{:}A)$ By continuous inv. of tpv_bind on (3)
(5) $(\Theta, \Theta'_n);\Gamma_n \vdash^e E{:}\Box A$ By inv. of tpv_bind on (4)
(6) $\Theta;\Gamma_n \vdash^e \rho'_n{:}\Theta'_n$ By inv. of tpv_bind on (4)
(7) $\Psi_0;\Gamma_n \models^e (R';\rho'_n):(\Theta, \Theta'_n)$ By tpv_env using (2),(6)
(8) $\exists! M'$ s.t. $(R', \rho'_n) \rhd E \mapsto M$ By IH on (7) using (5)
(9) $\exists! M' = \mathbf{unbox}_n M$ s.t. $(R';\ (\rho'_n,\ \mathbf{box}\ u = E,\ \rho_n);\cdots;\rho_0) \rhd u \mapsto M'$ By tx_mvar and (8)

■

The proof of proposition 2 has been left out since it is given in great detail in Pfenning and Davies' paper.

4.4 A Sample Translation

Below is a translation from

$$\lambda x{:}\Box(A \to B).\lambda y{:}\Box A.(\textbf{let box } u = x \textbf{ in let box } v = y \textbf{ in box } uv)$$

as derived in $\lambda_e^{\to\Box}$ in section 2.4 to

$$\lambda x{:}\Box(A \to B).\lambda y{:}\Box A.\textbf{box } ((\textbf{unbox}_1 x)(\textbf{unbox}_1 y))$$

as derived in $\lambda_i^{\to\Box}$ in section 3.3. Remember that both of these terms have the type $\Box(A \to B) \to (\Box A \to \Box B)$ which showcases proposition 2 of theorem 4.3.1 which states that types are preserved under the translation.

$$\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{}{\cdot \rhd x \mapsto x}\,\text{tx\_ovar}}{(\textbf{box } u = x, \textbf{box } v = y\ ); \cdot \rhd u \mapsto \textbf{unbox}_1 x}\,\text{tx\_mvar} \quad \dfrac{}{(\textbf{box } u = x) \rhd y \mapsto y}\,\text{tx\_ovar}}{(\textbf{box } u = x, \textbf{box } v = y\ ); \cdot \rhd v \mapsto \textbf{unbox}_1 y}\,\text{tx\_mvar}}{(\textbf{box } u = x, \textbf{box } v = y\ ); \cdot \rhd u\, v \mapsto (\textbf{unbox}_1 x)(\textbf{unbox}_1 y)}\,\text{tx\_app}}{(\textbf{box } u = x, \textbf{box } v = y\ ); \cdot \rhd \textbf{box } u\, v \mapsto \textbf{box } (\textbf{unbox}_1 x)(\textbf{unbox}_1 y)}\,\text{tx\_box}}{(\textbf{box } u = x\ ); \cdot \rhd \textbf{let box } v = y \textbf{ in box } u\, v \ \mapsto \textbf{box } (\textbf{unbox}_1 x)(\textbf{unbox}_1 y)}\,\text{tx\_letbox}}{\cdot \rhd \textbf{let box } u = x \textbf{ in let box } v = y \textbf{ in box } u\, v \ \mapsto \textbf{box } (\textbf{unbox}_1 x)(\textbf{unbox}_1 y)}\,\text{tx\_letbox}}{\cdot \rhd \lambda y{:}\Box A.(\textbf{let box } u = x \textbf{ in let box } v = y \textbf{ in box } u\, v\ ) \mapsto \lambda y{:}\Box A.(\textbf{box } (\textbf{unbox}_1 x)(\textbf{unbox}_1 y))}\,\text{tx\_lam}$$

$$\cdot \rhd \lambda x{:}\Box(A \to B).\lambda y{:}\Box A.(\textbf{let box } u = x \textbf{ in let box } v = y \textbf{ in box } u\, v\ ) \mapsto \lambda x{:}\Box(A \to B).\lambda y{:}\Box A.(\textbf{box } (\textbf{unbox}_1 x)(\textbf{unbox}_1 y))$$

5 Translation From the Implicit to the Explicit System

In this section, the translation from $\lambda_i^{\to\Box}$ to $\lambda_e^{\to\Box}$ and all related constructs are presented. While not hugely different, one could consider this translation slightly more involved since the unbox level in the implicit system has to be translated with care.

5.1 Nested Let Box Expressions and Merging

In order to give the translation judgement and its rules, **nested let box expressions** have to be defined as is shown below.

$$\begin{aligned} \mathrm{Let}(\cdot)(E) &= E \\ \mathrm{Let}(\rho,\ \mathbf{box}\ u = E')(E) &= \mathrm{Let}(\rho)(\mathbf{let\ box}\ u = E'\ \mathbf{in}\ E) \end{aligned}$$

From this construct, the following derived typing rule for the nested let expressions and environment abstractions can be shown to be true.

$$\frac{\Delta;\Gamma \vdash^e \rho{:}\Theta \qquad (\Delta,\Theta);\Gamma \vdash^e E{:}B}{\Delta;\Gamma \vdash^e \mathrm{Let}(\rho)(E){:}B}\ \text{tpi\_env}$$

A **merge operation** of two environment stacks has to be defined in order to be able to translate function applications. This binary infix operation is denoted by '|' and is shown below.

$$\begin{aligned} \odot|R_2 &= R_2 \\ R_1|\odot &= R_1 \\ (R_1;\rho_1)|(R_2;\rho_2) &= (R_1|R_2);(\rho_1,\rho_2) \end{aligned}$$

5.2 Some Useful Lemmas

While necessary to be able to translate function applications, the merge operation is cumbersome to work with. In order to prove the main translation theorem, the following two lemmas have to be proven about the merge operation.

Lemma 5.2.1: If $\Delta_1;\Gamma \vdash^e \rho_1{:}\Theta_1$ and $\Delta_2;\Gamma \vdash^e \rho_2{:}\Theta_2$ then $(\Delta_1,\Delta_2);\Gamma \vdash^e (\rho_1,\rho_2){:}(\Theta_1,\Theta_2)$

Proof: By simultaneous induction on the derivation of $\Delta_1;\Gamma \vdash^e \rho_1{:}\Theta_1$ and $\Delta_2;\Gamma \vdash^e \rho_2{:}\Theta_2$, call them $\mathcal{D}_1$ and $\mathcal{D}_2$ respectively.

The cases where $\rho_1 = \cdot$ and/or $\rho_2 = \cdot$ are trivial since the desired result follows directly from weakening of the non-empty derivation. So suppose not. Then it must be the case that:

$$\mathcal{D}_1 = \frac{\Delta_1;\Gamma \vdash^e \rho_1'{:}\Theta_1' \qquad (\Delta_1,\Theta_1');\Gamma \vdash^e E_1{:}\Box A_1}{\Delta_1;\Gamma \vdash^e (\rho_1',\mathbf{box}\ u = E_1){:}(\Theta_1',u{:}A_1)} \qquad \mathcal{D}_2 = \frac{\Delta_2;\Gamma \vdash^e \rho_2'{:}\Theta_2' \qquad (\Delta_2,\Theta_2');\Gamma \vdash^e E_2{:}\Box A_2}{\Delta_2;\Gamma \vdash^e (\rho_2',\mathbf{box}\ v = E_2){:}(\Theta_2',u{:}A_2 e)}$$

(1) $\Delta_1;\Gamma \vdash^e (\rho_1',\mathbf{box}\ u = E_1){:}(\Theta_1',u{:}A_1)$ — From $\mathcal{D}_1$
(2) $\Delta_2;\Gamma \vdash^e \rho_2'{:}\Theta_2'$ — By inversion on $\mathcal{D}_2$
(3) $(\Delta_1,\Delta_2);\Gamma \vdash^e (\rho_1',\mathbf{box}\ u = E_1,\rho_2'){:}(\Theta_1',u{:}A_1,\Theta_2')$ — By IH on (2) using (1)

(4) $(\Delta_2, \Theta_2'); \Gamma \vdash^e E_2{:}\Box A_2$ — By inversion on $\mathcal{D}_2$
(5) $(\Delta_1, \Delta_2); \Gamma \vdash^e (\rho_1', \textbf{box}\ u = E_1, \rho_2', \textbf{box}\ v = E_2){:}(\Theta_1', u{:}A_1, \Theta_2', v{:}A_2)$ — By tpv_bind,(3),(4)
(6) $\rho_1 = (\rho_1', \textbf{box}\ u = E_1)$ and $\Theta_1 = (\Theta_1', u{:}A_1)$ — By case
(7) $\rho_2 = (\rho_2', \textbf{box}\ v = E_2)$ and $\Theta_2 = (\Theta_2', u{:}A_2)$ — By case
(8) $(\Delta_1, \Delta_2); \Gamma \vdash^e (\rho_1, \rho_2){:}(\Theta_1, \Theta_2)$ — Rewriting (5) using (6),(7)

■

Lemma 5.2.2: If $\Psi \models^e R_1{:}\Delta_1$ and $\Psi \models^e R_2{:}\Delta_2$, then $\Psi \models^e (R_1|R_2){:}(\Delta_1, \Delta_2)$

Proof: By simultaneous induction on the derivations of $\Psi \models^e R_1{:}\Delta_1$ and $\Psi \models^e R_2{:}\Delta_2$, call them $\mathcal{D}_1$ and $\mathcal{D}_2$ respectively.

Case 1: $\mathcal{D}_1 = \dfrac{}{\Psi \models^e \odot{:}\cdot}$ tpr_empty

(1) $\Psi \models^e R_2{:}\Delta_2$ — By assumption
(2) $(R_1|R_2) = (\odot|R_2)$ — By case
(3) $(\odot|R_2) = R_2$ — By def. of environment merge
(4) $(R_1|R_2) = R_2$ — Combining (2),(3)
(5) $\Delta_1 = \cdot$ — By case
(6) $\Delta_1, \Delta_2 = \Delta_2$ — By (5)
(7) $\Psi \models^e (R_1|R_2){:}(\Delta_1, \Delta_2)$ — Rewriting (1) using (4),(6)

Case 2: $\mathcal{D}_2 = \dfrac{}{\Psi \models^e \odot{:}\cdot}$ tpr_empty

Entirely analogous to case 1.

Case 3: $\mathcal{D}_1 = \dfrac{\Psi' \models^e R_1'{:}\Delta_1' \qquad \Delta_1'; \Gamma \vdash^e \rho_1{:}\Theta_1}{\Psi'; \Gamma \models^e (R_1'; \rho_1){:}(\Delta_1', \Theta_1)}$ $\mathcal{D}_2 = \dfrac{\Psi' \models^e R_2'{:}\Delta_2' \qquad \Delta_2'; \Gamma \vdash^e \rho_2{:}\Theta_2}{\Psi'; \Gamma \models^e (R_2'; \rho_2){:}(\Delta_2', \Theta_2)}$

(1) $\Psi = \Psi'; \Gamma$ — By case
(2) $R_1 = (R_1'; \rho_1)$ and $\Delta_1 = (\Delta_1', \Theta_1)$ — By case
(3) $R_2 = (R_2'; \rho_2)$ and $\Delta_2 = (\Delta_2', \Theta_2)$ — By case
(4) $\Psi' \models^e (R_1'|R_2'){:}(\Delta_1', \Delta_2')$ — By IH on left subderivations of $\mathcal{D}_1, \mathcal{D}_2$

Subcase 3.1: $\rho_1 = \cdot$
(5) $\Theta_1 = \cdot$ — By subcase
(6) $\Delta_2'; \Gamma \vdash^e \rho_2{:}\Theta_2$ — By inversion on $\mathcal{D}_2$
(7) $\Psi \models^e (R_1'|R_2'); \rho_2{:}(\Delta_1', \Delta_2', \Theta_2)$ — By tpr_env using (4),(6)
(8) $\Psi \models^e (R_1'|R_2'); (\rho_1, \rho_2){:}(\Delta_1', \Delta_2', \Theta_2)$ — By (7) and subcase
(9) $\Psi \models^e (R_1'; \rho_1|R_2'; \rho_2){:}(\Delta_1', \Delta_2', \Theta_2)$ — By def. of environment merge and (7)
(10) $\Psi \models^e (R_1|R_2){:}(\Delta_1, \Delta_2)$ — Rewriting (9) using (2),(3),(5)

Subcase 3.2: $\rho_2 = \cdot$
Analogous to subcase 3.1.

Subcase 3.3:

$$\mathcal{D}_1'' = \frac{\Delta_1';\Gamma \vdash^e \rho_1':\Theta_1' \quad (\Delta_1',\Theta_1');\Gamma \vdash^e E_1:\Box A_1}{\Delta_1';\Gamma \vdash^e (\rho_1',\ \mathbf{box}\ u = E_1):(\Theta_1', u{:}A_1)}\ \mathcal{D}_2'' = \frac{\Delta_2';\Gamma \vdash^e \rho_2':\Theta_2' \quad (\Delta_2',\Theta_2');\Gamma \vdash^e E_2:\Box A_2}{\Delta_2';\Gamma \vdash^e (\rho_2',\ \mathbf{box}\ v = E_2):(\Theta_2', v{:}A_2)}$$

(11) $\Delta_1';\Gamma \vdash^e \rho_1:\Theta_1$	From right subder. of $\mathcal{D}_1$
(12) $\Delta_2';\Gamma \vdash^e \rho_2:\Theta_2$	From right subder. of $\mathcal{D}_2$
(13) $(\Delta_1',\Delta_2');\Gamma \vdash^e (\rho_1,\rho_2):(\Theta_1,\Theta_2)$	By Lemma 5.2.1 using (11),(12)
(14) $\Psi';\Gamma \models^e (R_1'\vert R_2');(\rho_1,\rho_2):(\Delta_1',\Delta_2',\Theta_1',\Theta_2')$	By tpr_env using (4),(13]
(15) $\Psi';\Gamma \models^e (R_1'\vert R_2');(\rho_1,\rho_2):(\Delta_1',\Theta_1',\Delta_2',\Theta_2')$	By exchange of modal context
(16) $\Psi';\Gamma \models^e (R_1';\rho_1\vert R_2';\rho_2):(\Delta_1',\Theta_1',\Delta_2',\Theta_2')$	By def. of environment merge
(17) $\Psi \models^e (R_1\vert R_2):(\Delta_1,\Delta_2)$	Rewriting (16) using (2),(3)

■

5.3 Translation Judgments

The translation of proof terms from $\lambda_e^{\to\Box}$ to $\lambda_i^{\to\Box}$ is based on the judgment below.

$$M \mapsto R \rhd E \quad M \text{ compiles to term } E \text{ under stack } R$$

The rules of the translation are as shown below.

$$\frac{}{x \mapsto \odot \rhd x}\ \text{tr\_var}$$

$$\frac{M \mapsto R \rhd E}{\lambda x{:}A.M \mapsto R \rhd \lambda x{:}A.E}\ \text{tr\_lam} \qquad \frac{M_1 \mapsto R_1 \rhd E_1 \quad M_2 \mapsto R_2 \rhd E_2}{M_1\ M_2 \mapsto (R_1\vert R_2) \rhd E_1\ E_2}\ \text{tr\_app}$$

$$\frac{M \mapsto \odot \rhd E}{\mathbf{box}\ M \mapsto \odot \rhd \mathbf{box}\ E}\ \text{tr\_box0} \qquad \frac{M \mapsto (R;\rho) \rhd E}{\mathbf{box}\ M \mapsto R \rhd \mathrm{Let}(\rho)(\mathbf{box}\ E)}\ \text{tr\_box1}$$

$$\frac{M \mapsto R \rhd E}{\mathbf{unbox}_0 M \mapsto R \rhd \mathbf{let\ box}\ u = E\ \mathbf{in}\ u}\ \text{tr\_unb0}$$

$$\frac{M \mapsto R \rhd E}{\mathbf{unbox}_{n+1} M \mapsto R;(\mathbf{box}\ u = E);\underbrace{\cdot;\cdots;\cdot}_{n} \rhd u}\ \text{tr\_unb1}$$

Note that the tr_app rule is only applicable to context stacks R_1 and R_2 when they are disjoint. If the situation arises where tr_app is to be applied to two context stacks which are not disjoint, the variables in the derivations of the two antecedents can be renamed to produce an equivalent translation modulo variable names.

5.4 Translation Theorem

Now the theorem of the translation from the implicit system to the explicit system can be given along with its proof.

Theorem 5.4.1:

(1) For any M there exists unique R and E such that $M \mapsto R \rhd E$

(2) If $M \mapsto R \rhd E$ and $\Psi; \Gamma \vdash^i M{:}A$ then for some Δ we have $\Psi \models^e R{:}\Delta$ and $\Delta; \Gamma \vdash^e E{:}A$

Proof of proposition 1: By induction on the subterms of M since the translation is defined on the subterms with unique translations, except for the case of tr_box0 and tr_box1, in which case exactly one of the two rules apply depending on the given environment.

■

Proof of proposition 2: By induction on the structure of the derivation of $M \mapsto R \rhd E$, call it $\mathcal{D}$.

Case 1: $\mathcal{D} = \dfrac{}{x \mapsto \odot \rhd x}$ tr_var

(1) $\Psi; \Gamma \vdash^i x{:}A$	By ass.
(2) $\Gamma = \Gamma_1, x{:}A, \Gamma_2$	By inversion of (1)
(3) $\cdot; \Gamma \vdash^e x{:}A$	By ovar using (2)
(4) $\Psi \models^e \odot{:}\cdot$	By tpr_empty

Where (3),(4) are the desired conclusions with $\Delta = \cdot$.

Case 2: $\mathcal{D} = \dfrac{M \mapsto R \rhd E}{\lambda x{:}A.M \mapsto R \rhd \lambda x{:}A.E}$ tr_lam

(1) $\Psi; \Gamma \vdash^i \lambda x{:}A.M{:}A \to B$	By ass.
(2) $\Psi; (\Gamma, x{:}A) \vdash^i M{:}B$	By inversion on (1)
(3) $\Psi \models^e R{:}\Delta$	By IH on subderivation of $\mathcal{D}$ using (2)
(4) $\Delta; (\Gamma, x{:}A) \vdash^e E{:}B$	By IH on subderivation of $\mathcal{D}$ using (2)
(5) $\Delta; \Gamma \vdash^e \lambda x{:}A.E{:}A \to B$	By $\to I$ using (4)

Where (3),(5) are the desired conclusions.

Case 3: $\mathcal{D} = \dfrac{M_1 \mapsto R_1 \rhd E_1 \qquad M_2 \mapsto R_2 \rhd E_2}{M_1\ M_2 \mapsto (R_1 | R_2) \rhd E_1\ E_2}$ tr_app

(1) $\Psi; \Gamma \vdash^i M_1\ M_2{:}A$	By ass.
(2) $\Psi; \Gamma \vdash^i M_1{:}B \to A$	By inversion on (1)
(3) $\Psi; \Gamma \vdash^i M_2{:}B$	By inversion on (1)
(4) $M_1 \mapsto R_1 \rhd E_1$	By inversion on $\mathcal{D}$
(5) $M_2 \mapsto R_2 \rhd E_2$	By inversion on $\mathcal{D}$
(6) $\Psi \models^e R_1{:}\Delta_1$ and $\Delta_1; \Gamma \vdash^e E_1{:}(B \to A)$	By IH on (4) using (2)
(7) $\Psi \models^e R_2{:}\Delta_2$ and $\Delta_2; \Gamma \vdash^e E_2{:}B$	By IH on (5) using (3)
(8) $(\Delta_1, \Delta_2); \Gamma \vdash E_1{:}(B \to A)$	By weakening of (6)
(9) $(\Delta_1, \Delta_2); \Gamma \vdash E_2{:}B$	By weakening of (7)

(10) $(\Delta_1, \Delta_2); \Gamma \vdash^e E_1\ E_2 : A$ — By $\to E$
(11) $\Psi \models^e (R_1 | R_2) : (\Delta_1, \Delta_2)$ — By Lemma 5.4.2 using (6),(7)

Where (10),(11) are the desired conclusions with $\Delta = (\Delta_1, \Delta_2)$

Case 4: $\mathcal{D} = \dfrac{M \mapsto \odot \rhd E}{\mathbf{box}\ M \mapsto \odot \rhd \mathbf{box}\ E}$ tr_box0

(1) $\Psi; \Gamma \vdash^i \mathbf{box}\ M : \Box A$ — By ass.
(2) $\Psi; \Gamma; \cdot \vdash^i M : A$ — By inversion on (1)
(3) $M \mapsto \odot \rhd E$ — By inversion on $\mathcal{D}$
(4) $\Psi; \Gamma \models^e \odot \rhd \Delta$ — By IH on (3) using (2)
(5) $\Delta; \cdot \vdash^e E : A$ — By IH on (4) using (2)
(6) $\Delta; \Gamma \vdash^e \mathbf{box}\ E : \Box A$ — By $\Box I$ using (5)
(7) $\Delta = \cdot$ — By inversion on (4)
(8) $\cdot; \Gamma \vdash^e \mathbf{box}\ E_1 : \Box A$ — Rewriting (6) using (7)
(9) $\Psi \models^e \odot : \cdot$ — By tpr_empty

Where (8),(9) are the desired conclusions with $\Delta = \cdot$

Case 5: $\mathcal{D} = \dfrac{M \mapsto (R; \rho) \rhd E}{\mathbf{box}\ M \mapsto R \rhd \mathbf{Let}(\rho)(\mathbf{box}\ E)}$ tr_box1

(1) $\Psi; \Gamma \vdash^i \mathbf{box}\ M : \Box A$ — By ass.
(2) $\Psi\Gamma; \cdot \vdash^i M : A$ — By inversion on (1)
(3) $M \mapsto (R; \rho) \rhd E$ — By inversion on $\mathcal{D}$
(4) $\Psi; \Gamma \models^e (R; \rho) : \Delta_1$ — By IH on (3) using (2)
(5) $\Delta_1; \cdot \vdash^e E : A$ — By IH on (3) using (2)
(6) $\Psi \models^e R : \Delta_1'$ — By inversion on (4)
(7) $\Delta_1'; \Gamma \vdash^e \rho : \Theta$ — By inversion on (4)
(8) $\Delta_1 = \Delta_1', \Theta$ — By inversion on (4)
(9) $(\Delta_1', \Theta); \cdot \vdash^e E : A$ — Rewriting (5) using (8)
(10) $(\Delta_1', \Theta); \Gamma \vdash^e \mathbf{box}\ E : \Box A$ — By $\Box A$ using (9)
(11) $\Delta_1'; \Gamma \vdash^e \mathbf{let}\ (\rho)(\mathbf{box}\ E) : \Box A$ — By tpi_env using (7),(10)

Where (7),(11) are the desired conclusions with $\Delta = \Delta_1'$

Case 6: $\mathcal{D} = \dfrac{M \mapsto R \rhd E}{\mathbf{unbox}_0 M \mapsto R \rhd \mathbf{Let\ box}\ u = E\ \mathbf{in}\ u}$ tr_unbox0

(1) $\Psi; \Gamma \vdash^i \mathbf{unbox}\ {}_0 M : A$ — By ass.
(2) $\Psi; \Gamma \vdash^i M : \Box A$ — By inversion on (1)
(3) $M \mapsto R \rhd E$ — By inversion on $\mathcal{D}$
(4) $\Psi \models^e R : \Delta$ — By IH on (3) using (2)
(5) $\Delta; \Gamma \vdash^e E : \Box A$ — By IH on (3) using (2)
(6) $(\Delta, u{:}A); \Gamma \vdash^e u : A$ — By tpe_mvar
(7) $\Delta; \Gamma \vdash^e \mathbf{let\ box}\ u = E\ \mathbf{in}\ u : A$ — By $\Box E$ using (5),(6)

Where (4),(7) are the desired conclusions.

Case 7: $\mathcal{D} = \dfrac{M \mapsto R \rhd E}{\mathbf{unbox}_{n+1} M \mapsto R; (\mathbf{box}\ u = E); \underbrace{\cdot; \cdots; \cdot}_{n} \rhd u}\ \text{tr\_unb1}$

(1) $\Psi; \Gamma \vdash^i \mathbf{unbox}_{(n+1)} M{:}A$ By ass.
(2) $\Psi'; \Gamma' \vdash^i M{:}\Box A$ and $\Psi = \Psi'; \Gamma'; \Gamma_1; \cdots; \Gamma_n$ By inversion on (1)
(3) $M \mapsto R \rhd E$ By inversion on $\mathcal{D}$
(4) $\Psi' \models^e R{:}\Delta'$ By IH on (3) using (2)
(5) $\Delta'; \Gamma' \vdash^e E{:}\Box A$ By IH on (3) using (2)
(6) $\Delta'; \Gamma' \vdash^e \vdot$ By tpv_empty
(7) $\Delta'; \Gamma' \vdash (\mathbf{box}\ u = E){:}(u{:}A)$ By tpv_bind using (5),(6)
(8) $\Psi'; \Gamma' \models^e (R; \mathbf{box}\ u = E){:}(\Delta', u{:}A)$ By tpv_bind using (4),(7)
(9) $(\Delta', u{:}A); \Gamma_i \vdash^e \cdot{:}\cdot \quad \forall i. 1 \leq i \leq n$ By tpv_empty
(10) $\Psi'; \Gamma'; \Gamma_1; \cdots; \Gamma_n \models^e (R; \mathbf{box}\ u = E; \cdot; \cdots; \cdot){:}(\Delta', u{:}A)$ By tpr_env n times using (8),(9)
(11) $\Psi \models^e (R; \mathbf{box}\ u = E; \cdot; \cdots; \cdot){:}(\Delta', u{:}A)$ Rewriting (10) using (2)
(12) $(\Delta', u{:}A); \Gamma \vdash^e u{:}A$ By tpe_mvar

Where (11),(12) are the desired conclusions with $\Delta = \Delta', u{:}A$

■

5.5 A Sample Translation

Just like in section 4.4, a sample translation of the two terms of the derivations in sections 2.4 and 3.3 is shown below.

$$\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{}{x \mapsto \odot \rhd x}\text{tr\_var}}{\mathbf{unbox}_1 x \mapsto (\mathbf{box}\ u = x) \rhd u}\text{tr\_unb1} \quad \dfrac{\dfrac{}{y \mapsto \odot \rhd y}\text{tr\_var}}{\mathbf{unbox}_1 y \mapsto (\mathbf{box}\ v = y) \rhd b}\text{tr\_unb1}}{(\mathbf{unbox}_1 x)(\mathbf{unbox}_1 y) \mapsto (\mathbf{box}\ u = x, \mathbf{box}\ v = y) \rhd u\, v}\text{tr\_app}}{\mathbf{box}\ (\mathbf{unbox}_1 x)(\mathbf{unbox}_1 y) \mapsto \odot \rhd \mathbf{let\ box}\ v = y\ \mathbf{in\ let\ box}\ u = x\ \mathbf{in\ box}\ u\, v}\text{tr\_box1}}{\lambda y.\mathbf{box}\ (\mathbf{unbox}_1 x)(\mathbf{unbox}_1 y) \mapsto \odot \rhd \lambda y.\mathbf{let\ box}\ v = y\ \mathbf{in\ let\ box}\ u = x\ \mathbf{in\ box}\ u\, v}\text{tr\_lam}}{\lambda x.\lambda y.\mathbf{box}\ (\mathbf{unbox}_1 x)(\mathbf{unbox}_1 y) \mapsto \odot \rhd \lambda x.\lambda y.\mathbf{let\ box}\ v = y\ \mathbf{in\ let\ box}\ u = x\ \mathbf{in\ box}\ u\, v}\text{tr\_lam}$$

6 The Contextual Explicit Modal λ-Calculus

In this section, the **contextual explicit modal λ-calculus**, referred to as $\lambda_{ec}^{\rightarrow\Box}$, is introduced. This new calculus is a variation of $\lambda_{e}^{\rightarrow\Box}$ which was introduced in section 2 with the addition of **contextual validity** and was originally introduced by Nanevski, Pfenning and Pientka[NPP08]. Recall that validity is defined as the truth of a proposition which does not depend on any hypotheses about the truth of other propositions Note however that it may depend on other valid assumptions, since their own validity cannot by definition depend on truth assumptions. In previous sections, validity has been expressed using the modal operator $\Box$. In this section,the judgment of $\Box A$ is relativized to a list of assumptions $\Xi = (x_1{:}B_1, ..., x_n{:}B_n)$ of true propositions and $[\Xi \vdash A]$ is defined to be true in every world in which the propositions in Ξ are true.

As will be seen when the typing judgments are presented, taking $\Xi = \cdot$, $\lambda_{ec}^{\rightarrow\Box}$ is isomorphic to $\lambda_{e}^{\rightarrow\Box}$. In fact, the expressive power of $\lambda_{ec}^{\rightarrow\Box}$ is equal that of $\lambda_{e}^{\rightarrow\Box}$ when considering the proofs it can produce, as will be seen in the raising/lowering subsection. The question then arises why $\lambda_{ec}^{\rightarrow\Box}$ is interesting to study when $\lambda_{e}^{\rightarrow\Box}$ is a less convoluted and equivalent system. The reason for this is based in "providing further insights for language design"[NPP08] from a theoretical perspective, and some code optimization as will be seen in an example shown in this section.

6.1 Typing Rules of the Contextual Explicit Modal λ-Calculus

Before the contextual typing rules for the explicit system can be stated, the notion of a substitution has to be defined.

$$\text{Substitution} \quad \sigma \;=\; \cdot \mid \sigma, (E/x)$$

Well-typed substitutions are constructed through the following judgements. Here, Ξ is used for a, possible empty, typing context of ordinary variables. Because of the implicit assumption that all variables in a typing context are unique, all variables are also assumed to be unique in a substitution.

$$\frac{}{\Delta; \Gamma \vdash \cdot{:}\cdot}\ \text{sub\_exp\_empty} \qquad \frac{\Delta; \Gamma \vdash \sigma{:}\Xi \quad \Delta; \Gamma \vdash E{:}A}{\Delta; \Gamma \vdash \sigma, (E/x){:}(\Xi, x{:}A)}\ \text{sub\_exp\_bind}$$

Now the typing judgement for contextual types in the explicit system can be presented. The non-modal rules are unchanged. $\hat{\Xi}$ is used to strip the types of a typing context, resulting in only the variables in Ξ.

$$\frac{x{:}A \in \Gamma}{\Delta; \Gamma \vdash^e x{:}A}\ \text{ovar} \qquad \frac{u{:}(\Xi \vdash A) \in \Delta \quad \Delta; \Gamma \vdash \sigma{:}\Xi}{\Delta; \Gamma \vdash^e u[\sigma]{:}A}\ \text{mvar}$$

$$\frac{\Delta; (\Gamma, x{:}A) \vdash^e E{:}B}{\Delta; \Gamma \vdash^e \lambda x{:}A.E{:}(A \rightarrow B)} \rightarrow I \qquad \frac{\Delta; \Xi \vdash^e E{:}A}{\Delta; \Gamma \vdash^e \mathbf{box}(\hat{\Xi}.E){:}[\Xi \vdash A]}\ \Box I$$

$$\frac{\Delta; \Gamma \vdash^e E_1{:}(B \rightarrow A) \quad \Delta; \Gamma \vdash^e E_2{:}B}{\Delta; \Gamma \vdash^e E_1\ \ E_2{:}A} \rightarrow E \qquad \frac{\Delta; \Gamma \vdash^e E_1{:}[\Xi \vdash A] \quad (\Delta, u{:}(\Xi \vdash A)); \Gamma \vdash^e E_2{:}B}{\Delta; \Gamma \vdash^e \mathbf{let\ box}\ u = E_1\ \mathbf{in}\ E_2{:}B}\ \Box E$$

Notice that only $[\Xi \vdash A]$ is a valid type and $(\Xi \vdash A)$ is not. From a logic-theoretic perspective, what is meant by $u{:}(\Xi \vdash A)$ is that $(\Xi \vdash A)$ is valid, meaning that A is true in every world where

Ξ is true. This can be thought of intuitively as u being an already unboxed version of a proof-term with type $[\Xi \vdash A]$. $u[\sigma]$ is notation for a suspended substitution which will be executed eagerly once u is substituted and the variables in σ are present.

The substitution principles of this new system are shown below. Here $[\![\ ]\!]$ is notation for a substitution of modal variables. The proofs of the substitutions principles are standard and are left out for brevity.

Modal Substitution Principle with Contextual Types in Explicit System:
If $\Delta; \Xi \vdash E_1{:}A$ and $(\Delta, u{:}(\Xi \vdash A), \Delta'); \Gamma \vdash E_2{:}C$ then $(\Delta, \Delta'); \Gamma \vdash [\![\hat{\Xi}.E_1/u]\!]E_2{:}C$

Ordinary Substitution Principe with Contextual Types in Explicit System
If $\Delta; \Gamma \vdash E_1{:}A$ and $\Delta; (\Gamma, x{:}A, \Gamma') \vdash E_2{:}C$ then $\Delta; \Gamma, \Gamma' \vdash [E_1/x]E_2{:}C$

6.2 Lowering and Raising - The Variable-by-Variable Approach

In this section, the **variable-by-variable** approach for lowering and raising is introduced. While this approach builds on earlier work from Pientka[Pie03], this approach quickly becomes undesirable when applied to larger proof terms since it produces proof terms of a significantly larger size when applied. This has led to the development of the **derivational** approach, which is presented in a later chapter. While the derivational approach could have been presented on its own without the variable-by-variable approach, this section has been included for the sake of contrasting the two approaches and showcasing the value of the derivational approach in contrast to the variable-by-variable approach.

The main aim of lowering and raising of variables is to translate proof terms between the non-contextual and contextual calculi while preserving the type of the proof term. Lowering is conducted by replacing a variable of a function type in a proof term with a variable where the antecedent of the function type has been converted to a contextualized variable. Raising is done in the exact opposite way.

In the last subsection of this section, lowering and raising of proof terms themselves is introduced. While this naturally changes the type of the proof term, it does so in a predictable and desirable way similar to the type change of variables.

6.2.1 Lowering and Raising of Modal Variables in the Explicit System

In this subsection, the concept of **lowering and raising modal variables** is introduced. Lowering and raising of modal variables was first introduced by Pientka in their PhD thesis[Pie03] in a setting with dependent types. Since this treatment only deal with non-dependent types, a simplified version for non-dependent types is presented.

Before lowering can be proven, a trivial lemma about identity substitutions is stated and proven. For a context of ordinary variables $\Gamma = (x_1{:}A_1, \cdots, x_n{:}A_n)$, the identity Substitution id_Γ is defined as $\mathsf{id}_\Gamma = (x_1/x_1, \cdots x_n/x_n)$.

Lemma 6.2.1: For any ordinary context Γ, it is true that: $\Delta; \Gamma \vdash \mathsf{id}_\Gamma{:}\Gamma$

Proof: By induction on the size of Γ.

Case 1: $\Gamma = \cdot$

(1) $\mathsf{id}_\Gamma = \cdot$ By case
(2) $\Delta; \cdot \vdash \cdot{:}\cdot$ By sub_exp_bind
(3) $\Delta; \Gamma \vdash \mathsf{id}_\Gamma{:}\Gamma$ Rewriting (2) using case and (1)

Case 2: $\Gamma = \Gamma', x{:}A$

(1) $\mathsf{id}_\Gamma = \mathsf{id}_{\Gamma'}, x/x$ By case
(2) $\Delta; \Gamma' \vdash \mathsf{id}_{\Gamma'}{:}\Gamma'$ By IH
(3) $\Delta; (\Gamma', x{:}A) \vdash x{:}A$ By ovar
(4) $\Delta; (\Gamma', x{:}A) \vdash \mathsf{id}_{\Gamma'}, x/x{:}(\Gamma', x{:}A)$ By sub_exp_bind using (2),(3)
(5) $\Delta; \Gamma \vdash \mathsf{id}_\Gamma{:}\Gamma$ Rewriting (4) using case

■

Now the property of *lowering* can be introduced and proven.

Theorem 6.2.2 (Lowering of Modal Variables): If $(\Delta, u{:}(\Xi \vdash A_1 \to A_2), \Delta'); \Gamma \vdash E{:}A$ then $(\Delta, u'{:}(\Xi, x{:}A_1 \vdash A_2), \Delta'); \Gamma \vdash [\![\hat{\Xi}.\lambda x{:}A.u'[\mathsf{id}_\Xi, x/x]/u]\!]E{:}A$

Proof: Direct through the modal substitution principle and modal weakening.

(1) $(\Delta, u{:}(\Xi \vdash A_1 \to A_2), \Delta'); \Gamma \vdash E{:}A$ By ass.
(2) $(\Delta, u'{:}(\Xi, x{:}A_1 \vdash A_2), u{:}(\Xi \vdash A_1 \to A_2), \Delta'); \Gamma \vdash E{:}A$ By modal weakening of (1)
(3) $\Delta, u'{:}(\Xi, x{:}A_1 \vdash A_2); (\Xi; x{:}A_1) \vdash (\mathsf{id}_\Xi, x/x){:}(\Xi, x{:}A_1)$ By lemma 6.2.1 with $\Gamma = \Xi, x{:}A$
(4) $\Delta, u'{:}(\Xi, x{:}A_1 \vdash A_2); (\Xi, x{:}A_1) \vdash u'[\mathsf{id}_\Xi, x/x]{:}A_2$ By mvar using (3)
(5) $\Delta, u'{:}(\Xi, x{:}A_1 \vdash A_2); \Xi \vdash \lambda x{:}A_1.u'[\mathsf{id}_\Xi, x/x]{:}A_1 \to A_2$ By $\to I$ using (4)
(6) $(\Delta, u'{:}(\Xi, x{:}A_1 \vdash A_2), \Delta'); \Gamma \vdash [\![\hat{\Xi}.\lambda x{:}A.u'[\mathsf{id}_\Xi, x/x]/u]\!]E{:}A$ By m.s.p. using (2),(5)

■

The inverse operation of lowering is *raising* and is stated and proven below.

Theorem 6.2.3 (Raising of Modal Variables): If $(\Delta, u'{:}(\Xi, x{:}A_1 \vdash A_2), \Delta'); \Gamma \vdash E{:}A$ then $\Delta, u{:}(\Xi \vdash A_1 \to A_2), \Delta'; \Gamma \vdash [\![(\hat{\Xi}, x).(u[\mathsf{id}_\Xi]\ x)/u']\!]E{:}A$

Proof: Direct through the modal substitution principle and modal weakening.

(1) $(\Delta, u'{:}(\Xi, x{:}A_1 \vdash A_2), \Delta'); \Gamma \vdash E{:}A$ By ass.
(2) $(\Delta, u{:}(\Xi \vdash A_1 \to A_2), u'{:}(\Xi, x{:}A_1 \vdash A_2), \Delta'); \Gamma \vdash E{:}A$ By modal weakening of (1)
(3) $\Delta, u{:}(\Xi \vdash A_1 \to A_2); \Xi \vdash \mathsf{id}_\Xi{:}\Xi$ By lemma 6.2.1 with $\Gamma = \Xi$
(4) $\Delta, u{:}(\Xi \vdash A_1 \to A_2); \Xi \vdash u[\mathsf{id}_\Xi]{:}(A_1 \to A_2)$ By mvar using (3)
(5) $\Delta, u{:}(\Xi \vdash A_1 \to A_2); (\Xi, x{:}A_1) \vdash u[\mathsf{id}_\Xi]{:}(A_1 \to A_2)$ By weakening of (4)
(6) $\Delta, u{:}(\Xi \vdash A_1 \to A_2); (\Xi, x{:}A_1) \vdash x{:}A_1$ By ovar
(7) $\Delta, u{:}(\Xi \vdash A_1 \to A_2); (\Xi, x{:}A_1) \vdash (u[\mathsf{id}_\Xi]\ x){:}A_2$ By $\to E$ using (5),(6)
(8) $\Delta, u{:}(\Xi \vdash A_1 \to A_2), \Delta'; \Gamma \vdash [\![(\hat{\Xi}, x).(u[\mathsf{id}_\Xi]\ x)/u']\!]E{:}A$ By m.s.p. using (2),(7)

■

6.2.2 Lowering and Raising of Ordinary Variables in the Explicit System

Lowering and raising of ordinary variables are done almost entirely analogously to modal variables. Below are the theorems and their respective proofs, which are also similar in style to that of raising and lowering of modal variables.

Theorem 6.2.4 (Lowering of Ordinary Variables): If $\Delta; (\Gamma, x{:}[\Xi \vdash A \to B]) \vdash E{:}C$ then $\Delta; (\Gamma, x'{:}[\Xi, a{:}A \vdash B]) \vdash [(\textbf{let box}\ \ u = x'\ \ \textbf{in box}(\hat{\Xi}.\lambda a{:}A.u[\mathsf{id}_{(\Xi,a:A)}]))/x]E{:}C$

Proof: Direct through the ordinary substitution principle and weakening.

(1) $\Delta; (\Gamma, x{:}[\Xi \vdash A \to B]) \vdash E{:}C$ By ass.
(2) $\Delta; (\Gamma, x{:}[\Xi \vdash A \to B], x'{:}[\Xi, a{:}A \vdash B]) \vdash E{:}C$ By weakening of (1)
(3) $(\Delta, u{:}(\Xi, a{:}A \vdash B)); (\Xi, a{:}A) \vdash \mathsf{id}_{(\Xi,a:A)}{:}(\Xi, a{:}A)$ By lemma 6.2.1
(4) $(\Delta, u{:}(\Xi, a{:}A \vdash B)); (\Xi, a{:}A) \vdash u[\mathsf{id}_{(\Xi,a:A)}]{:}B$ By mvar using (3)
(5) $(\Delta, u{:}(\Xi, a{:}A \vdash B)); \Xi \vdash \lambda a{:}A.u[\mathsf{id}_{(\Xi,a:A)}]{:}(A \to B)$ By $\to I$ using (4)
(6) $(\Delta, u{:}(\Xi, a{:}A \vdash B)); (\Gamma, x'{:}[\Xi, a{:}A \vdash B]) \vdash \textbf{box}\ (\hat{\Xi}.\lambda a{:}A.u[\mathsf{id}_{(\Xi,a:A)}]){:}[\Xi \vdash A \to B]$ By $\Box I$, (5)
(7) $\Delta; (\Gamma, x'{:}[\Xi, a{:}A \vdash B]) \vdash x'{:}[\Xi, a{:}A \vdash B]$ By ovar
(8) $\Delta; (\Gamma, x'{:}[\Xi, a{:}A \vdash B]) \vdash \textbf{let box}\ u = x'\ \textbf{in box}\ (\hat{\Xi}.\lambda a{:}A.u[\mathsf{id}_{(\Xi,a:A)}]){:}[\Xi \vdash A \to B]$ $\Box E$,(6),(7)
(9) $\Delta; (\Gamma, x'{:}[\Xi, a{:}A \vdash B]) \vdash [(\textbf{let box}\ u = x'\ \textbf{in box}\ (\hat{\Xi}.\lambda a{:}A.u[\mathsf{id}_{(\Xi,a:A)}]))/x]E{:}C$ o.s.p., (2),(8)

■

Theorem 6.2.5 (Raising of Ordinary Variables): If $\Delta; (\Gamma, x{:}[\Xi, a{:}A \vdash B]) \vdash E{:}C$ then $\Delta; (\Gamma; x'{:}[\Xi \vdash A \to B]) \vdash [\textbf{let box}\ \ u = x'\ \ \textbf{in box}((\hat{\Xi}, a).(u[\mathsf{id}_{\Xi}]\ a))/x]E{:}C$

Proof: Direct through the ordinary substitution principle and weakening.

(1) $\Delta; (\Gamma, x{:}[\Xi, a{:}A \vdash B]) \vdash E{:}C$ By ass.
(2) $\Delta; (\Gamma, x'{:}[\Xi \vdash A \to B], x{:}[\Xi, a{:}A \vdash B]) \vdash E{:}C$ By weakening of (1)
(3) $(\Delta, u{:}(\Xi \vdash A \to B)); \Xi \vdash \mathsf{id}_{\Xi}{:}\Xi$ By lemma 20
(4) $(\Delta, u{:}(\Xi \vdash A \to B)); \Xi \vdash u[\mathsf{id}_{\Xi}]{:}(A \to B)$ By mvar using (3)
(5) $(\Delta, u{:}(\Xi \vdash A \to B)); (\Xi, a{:}A) \vdash u[\mathsf{id}_{\Xi}]{:}(A \to B)$ By weakening of (4)
(6) $(\Delta, u{:}(\Xi \vdash A \to B)); (\Xi, a{:}A) \vdash a{:}A$ By ovar
(7) $(\Delta, u{:}(\Xi \vdash A \to B)); (\Xi, a{:}A) \vdash u[\mathsf{id}_{\Xi}]\ a{:}B$ By $\to E$ using (5),(6)
(8) $(\Delta, u{:}(\Xi \vdash A \to B)); (\Gamma, x'{:}[\Xi \vdash A \to B]) \vdash \textbf{box}\ ((\hat{\Xi}, a).(u[\mathsf{id}_{\Xi}]\ a)){:}[\Xi, a{:}A \vdash B]$ By $\Box I$,(7)
(9) $\Delta; (\Gamma, x'{:}[\Xi \vdash A \to B]) \vdash x'{:}[\Xi \vdash A \to B]$ By ovar
(10) $\Delta; (\Gamma, x'{:}[\Xi \vdash A \to B]) \vdash \textbf{let box}\ u = x'\ \textbf{in box}\ ((\hat{\Xi}, a).(u[\mathsf{id}_{\Xi}]\ a)){:}[\Xi, a{:}A \vdash B]$ By $\Box E$, (8), (9)
(11) $\Delta; (\Gamma, x'{:}[\Xi \vdash A \to B]) \vdash [(\textbf{let box}\ u = x'\ \textbf{in box}\ ((\hat{\Xi}, a).(u[\mathsf{id}_{\Xi}]\ a)))/x]E{:}C$ By o.s.p., (2),(10)

■

6.2.3 Lowering and Raising of Proof Terms in the Explicit System

Lowering and raising of proof terms has a very syntactically similar flavour to that of lowering ordinary variables. This stems from the fact that only boxed terms and ordinary variables can be lowered, which is carried out by eliminating the box through a let box-expression and then re-using the modal variable representing the "unboxed" term or variable in an appropriate way. In the case of ordinary variables, this newly constructed term then has to be substituted in place of the old variable - whereas in the case of proof terms, no such substitution is needed. This leads to the following to theorems being very similar to, but a bit more compact than, the previous two theorems.

Theorem 6.2.6 (Lowering of Proof Terms): If $\Delta; \Gamma \vdash E{:}[\Xi \vdash A \to B]$ then
$\Delta; \Gamma \vdash \textbf{let box } u = E \textbf{ in box}((\hat{\Xi}, a).u[\mathsf{id}_{\Xi}]\ a){:}[\Xi, a{:}A \vdash B]$

Proof: Direct.

(1) $\Delta; \Gamma \vdash E{:}[\Xi \vdash A \to B]$	By assumption
(2) $(\Delta, u{:}(\Xi \vdash A \to B)); \Xi \vdash \mathsf{id}_{\Xi}{:}\Xi$	By lemma 6.2.1
(3) $(\Delta, u{:}(\Xi \vdash A \to B)); (\Xi, a{:}A) \vdash \mathsf{id}_{\Xi}{:}\Xi$	By weakening of (2)
(4) $(\Delta, u{:}(\Xi \vdash A \to B)); (\Xi, a{:}A) \vdash u[\mathsf{id}_{\Xi}]{:}(A \to B)$	By mvar using (3)
(5) $(\Delta, u{:}(\Xi \vdash A \to B)); (\Xi, a{:}A) \vdash a{:}A$	By ovar
(6) $(\Delta, u{:}(\Xi \vdash A \to B)); (\Xi, a{:}A) \vdash u[\mathsf{id}_{\Xi}]\ a{:}B$	By $\to E$ using (4),(5)
(7) $(\Delta, u{:}(\Xi \vdash A \to B)); \Gamma \vdash \textbf{box}((\hat{\Xi}, a).u[\mathsf{id}_{\Xi}]\ a){:}[\Xi, a{:}A \vdash B]$	By $\Box I$ using (6)
(8) $\Delta, \Gamma \vdash \textbf{let box } u = E \textbf{ in box}((\hat{\Xi}, a).u[\mathsf{id}_{\Xi}]\ a){:}[\Xi, a{:}A \vdash B]$	By $\Box E$ using (1),(7)

■

Theorem 6.2.7 (Raising of Proof Terms): If $\Delta; \Gamma \vdash E{:}[\Xi, a{:}A \vdash B]$ then
$\Delta; \Gamma \vdash (\textbf{let box } u = E \textbf{ in box}(\hat{\Xi}.\lambda a{:}A.u[\mathsf{id}_{\Xi, a:A}])){:}[\Xi \vdash A \to B]$

Proof: Direct.

(1) $\Delta; \Gamma \vdash E{:}[\Xi, a{:}A \vdash B]$	By ass.
(2) $(\Delta, u{:}(\Xi, a{:}A \vdash B)); (\Xi, a{:}A) \vdash \mathsf{id}_{\Xi, a:A}{:}(\Xi, a{:}A)$	By lemma 6.2.1
(3) $(\Delta, u{:}(\Xi, a{:}A \vdash B)); (\Xi, a{:}A) \vdash u[\mathsf{id}_{\Xi, a:A}]{:}B$	By mvar using (2)
(4) $(\Delta, u{:}(\Xi, a{:}A \vdash B)); \Xi \vdash \lambda a{:}A.u[\mathsf{id}_{\Xi, a:A}]{:}(A \to B)$	By $\to I$
(5) $(\Delta, u{:}(\Xi, a{:}A \vdash B)); \Gamma \vdash \textbf{box}(\hat{\Xi}.\lambda a{:}A.u[\mathsf{id}_{\Xi, a:A}]){:}[\Xi \vdash A \to B]$	By $\Box I$ using (4)
(6) $\Delta; \Gamma \vdash$ **let box** $u = E$ **in box**$(\hat{\Xi}.\lambda a{:}A.u[\mathsf{id}_{\Xi, a:A}]){:}[\Xi \vdash A \to B]$	By $\Box E$ using (1),(5)

■

6.3 Lowering and Raising - The Derivational Approach

In this section, the **derivational approach** of lowering and raising is introduced. The main aim of this approach is to produce intuitive translations from contextual to non-contextual terms and the other way around while still preserving the typing properties of the **variable-by-variable approach**.

As can be seen with the variable-by-variable approach, if a variable of type $[\cdot \vdash A \to B \to C]$ is to be lowered, it has to be lowered twice to produce a variable of the type $[x{:}A, y{:}B \vdash C]$, since theorem 6.2.4 only lowers one input of the function type at once. In contrast, the **derivational approach** when applied to the same variable would directly produce a variable of the desired type. Likewise, the derivational approach also spares a lot of "wrapper terms" which the variable-by-variable approach produces and instead produces a more intuitive raised/lowered proof term.

For both translations the rules with number attached to their names are applied with the lowest possible number for the given term. For terms to be invariant if raised and then lowered or the other way around, the order of contexts and substitutions are preserved.

The translation presented below is a judgement relating two lines of two derivations and is of the form $\Delta;\Gamma \vdash E{:}A \searrow \Delta^{\downarrow};\Gamma^{\downarrow} \vdash e{:}A^{\downarrow}$. Here, the superscripted $\downarrow$ on a type means a completely lowered version of the original type. An example is if $A = [\cdot \vdash B \to C]$ then $A^{\downarrow} = [x{:}B \vdash C]$. A superscripted $\downarrow$ of a context represents a corresponding context where every variable is completely lowered.

$$\frac{}{\Delta;(\Gamma, x{:}A) \vdash x{:}A \searrow \Delta^{\downarrow};(\Gamma^{\downarrow}, x{:}A^{\downarrow}) \vdash x{:}A^{\downarrow}}\ \text{ovar\_lw}$$

$$\frac{\Delta;(\Gamma; x{:}A) \vdash E{:}B \searrow \Delta^{\downarrow};(\Gamma^{\downarrow}, x{:}A^{\downarrow}) \vdash e{:}B^{\downarrow}}{\Delta;\Gamma\lambda x{:}A.E{:}(A \to B) \searrow \Delta^{\downarrow};\Gamma^{\downarrow} \vdash \lambda x{:}A^{\downarrow}.e{:}(A^{\downarrow} \to B^{\downarrow})}\ \text{lam\_lw}$$

$$\frac{\Delta;(x_1{:}A_1, \cdots, x_n{:}A_n) \vdash E{:}B \searrow \Delta^{\downarrow};(x_1{:}A_1^{\downarrow}, \cdots, x_n{:}A_n^{\downarrow}) \vdash e{:}B^{\downarrow}}{\Delta;\Gamma \vdash \mathbf{box}(\lambda x_1{:}A_1.\cdots\lambda x_n{:}A_n.E){:}[\cdot \vdash A_1 \to \cdots \to A_n \to B] \searrow \Delta^{\downarrow};\Gamma^{\downarrow} \vdash \mathbf{box}(x_1, ..., x_n.e){:}[x_1{:}A_1^{\downarrow}, \cdots, x_n{:}A_n^{\downarrow} \vdash B^{\downarrow}]}\ \text{box1\_lw}$$

$$\frac{\Delta;\cdot \vdash E{:}B \searrow \Delta^{\downarrow};\cdot \vdash e{:}B^{\downarrow}}{\Delta;\Gamma \vdash \mathbf{box}(E){:}[\cdot \vdash B] \searrow \Delta^{\downarrow};\Gamma^{\downarrow} \vdash \mathbf{box}(e){:}[\cdot \vdash B^{\downarrow}]}\ \text{box2\_lw}$$

$$\frac{\Delta;\Gamma \vdash E_1{:}[\cdot \vdash A] \searrow \Delta^{\downarrow};\Gamma^{\downarrow} \vdash e_1{:}[\cdot \vdash A]^{\downarrow} \qquad (\Delta, u{:}(\cdot \vdash A));\Gamma \vdash E_2{:}B \searrow (\Delta^{\downarrow}, u{:}(\cdot \vdash A)^{\downarrow});\Gamma^{\downarrow} \vdash e_2{:}B^{\downarrow}}{\Delta;\Gamma \vdash \mathbf{let\ box}\ u = E_1\ \mathbf{in}\ E_2{:}B \searrow \Delta^{\downarrow};\Gamma^{\downarrow} \vdash \mathbf{let\ box}\ u = e_1\ \mathbf{in}\ e_2{:}B^{\downarrow}}\ \text{letbox\_lw}$$

$$\frac{\Delta;\Gamma \vdash E_i{:}A_i \searrow \Delta^{\downarrow};\Gamma^{\downarrow} \vdash e_i{:}A_i^{\downarrow} \quad \forall i.0 \leq i \leq n}{(\Delta, u{:}(\cdot \vdash A_1 \to \cdots \to A_n \to B));\Gamma \vdash u[\cdot]E_1...E_n{:}B \searrow (\Delta^{\downarrow}, u{:}(x_1{:}A_1^{\downarrow}, ..., x_n{:}A_n^{\downarrow}));\Gamma^{\downarrow} \vdash u[e_1/x_1, ..., e_n/x_n]{:}B^{\downarrow}}\ \text{app1\_lw}$$

$$\frac{\Delta;\Gamma \vdash E_1{:}A \to B \searrow \Delta^{\downarrow};\Gamma^{\downarrow} \vdash e_1{:}A^{\downarrow} \to B^{\downarrow} \qquad \Delta;\Gamma \vdash E_2{:}A \searrow \Delta^{\downarrow};\Gamma^{\downarrow} \vdash e_2{:}A^{\downarrow}}{\Delta;\Gamma \vdash E_1E_2{:}B \searrow \Delta^{\downarrow};\Gamma^{\downarrow} \vdash e_1e_2{:}B^{\downarrow}}\ \text{app2\_lw}$$

As an example of how a derivation might be lowered the lowering of a simple program has been included below. For ease of notation, it is assumed that all simple types are primitive.

$$\dfrac{\dfrac{}{\cdot; x{:}A \vdash x{:}A \searrow \cdot; x{:}A \vdash x{:}A}}{\cdot \vdash a{:}A \vdash \mathbf{box}(\lambda x{:}A.x){:}[\cdot \vdash A \to A] \searrow \cdot; a{:}A \vdash \mathbf{box}(x.x){:}[x{:}A \vdash A]} \quad \dfrac{\dfrac{}{u{:}(\cdot \vdash A \to A); a{:}A \vdash a{:}A \searrow u{:}(x{:}A \vdash A); a{:}A \vdash a{:}A}}{u{:}(\cdot \vdash A \to A); a{:}A \vdash u[\cdot]a{:}A \searrow u{:}(x{:}A \vdash A); a{:}A \vdash u[a/x]{:}A}$$

$$\cdot; a{:}A \vdash \mathbf{let\ box}\ u = \mathbf{box}(\lambda x{:}A.x)\ \mathbf{in}\ u[\cdot]a{:}A \searrow \cdot; a{:}A \vdash \mathbf{let\ box}\ u = \mathbf{box}(x.x)\ \mathbf{in}\ u[a/x]{:}A$$

The raising translation is completely analogous and is only included separately for notational purposes. Once again, a superscripted $\uparrow$ of a type means a completely raised version of the original type.

$$\dfrac{}{\Delta; (\Gamma, x{:}A) \vdash x{:}A \nearrow \Delta^{\uparrow}; (\Gamma^{\uparrow}, x{:}A^{\uparrow}) \vdash x{:}A^{\uparrow}}\ \text{ovar\_rs}$$

$$\dfrac{\Delta; (\Gamma; x{:}A) \vdash e{:}B \nearrow \Delta^{\uparrow}; (\Gamma^{\uparrow}, x{:}A^{\uparrow}) \vdash E{:}B^{\uparrow}}{\Delta; \Gamma\lambda x{:}A.e{:}(A \to B) \nearrow \Delta^{\uparrow}; \Gamma^{\uparrow} \vdash \lambda x{:}A^{\uparrow}.E{:}(A^{\uparrow} \to B^{\uparrow})}\ \text{lam\_rs}$$

$$\dfrac{\Delta; (x_1{:}A_1, \cdots, x_n{:}A_n) \vdash E{:}B \nearrow \Delta^{\uparrow}; (x_1{:}A_1^{\uparrow}, \cdots, x_n{:}A_n^{\uparrow}) \vdash e{:}B^{\uparrow}}{\Delta; \Gamma \vdash \mathbf{box}(x_1, ..., x_n.e){:}[x_1{:}A_1, \cdots, x_n{:}A_n \vdash B] \nearrow \Delta^{\uparrow}; \Gamma^{\uparrow} \vdash \mathbf{box}(\lambda x_1{:}A_1^{\uparrow}. \cdots \lambda x_n{:}A_n^{\uparrow}.E){:}[\cdot \vdash A_1^{\uparrow} \to \cdots \to A_n^{\uparrow} \to B^{\uparrow}]}\ \text{box1\_rs}$$

$$\dfrac{\Delta; \cdot \vdash e{:}B \nearrow \Delta^{\uparrow}; \cdot \vdash E{:}B^{\uparrow}}{\Delta; \Gamma \vdash \mathbf{box}(e){:}[\cdot \vdash B] \nearrow \Delta^{\uparrow}; \Gamma^{\uparrow} \vdash \mathbf{box}(E){:}[\cdot \vdash B^{\uparrow}]}\ \text{box2\_rs}$$

$$\dfrac{\Delta; \Gamma \vdash e_1{:}[\cdot \vdash A] \nearrow \Delta^{\uparrow}; \Gamma^{\uparrow} \vdash E_1{:}[\cdot \vdash A]^{\uparrow} \qquad (\Delta, u{:}(\cdot \vdash A)); \Gamma \vdash e_2{:}B \nearrow (\Delta^{\uparrow}, u{:}(\cdot \vdash A)^{\uparrow}); \Gamma^{\uparrow} \vdash E_2{:}B^{\uparrow}}{\Delta; \Gamma \vdash \mathbf{let\ box}\ u = e_1\ \mathbf{in}\ e_2{:}B \nearrow \Delta^{\uparrow}; \Gamma^{\uparrow} \vdash \mathbf{let\ box}\ u = E_1\ \mathbf{in}\ E_2{:}B^{\uparrow}}\ \text{letbox\_rs}$$

$$\dfrac{\Delta; \Gamma \vdash e_i{:}A_i \nearrow \Delta^{\uparrow}; \Gamma^{\uparrow} \vdash E_i{:}A_i^{\uparrow} \quad \forall i.0 \leq i \leq n}{(\Delta, u{:}(x_1{:}A_1, ..., x_n{:}A_n)); \Gamma \vdash u[e_1/x_1, ..., e_n/x_n]{:}B \nearrow (\Delta^{\uparrow}, u{:}(\cdot \vdash A_1^{\uparrow} \to \cdots \to A_n^{\uparrow} \to B^{\uparrow})); \Gamma^{\uparrow} \vdash u[\cdot]E_1...E_n{:}B^{\uparrow}}\ \text{app1\_rs}$$

$$\dfrac{\Delta; \Gamma \vdash e_1{:}A \to B \nearrow \Delta^{\uparrow}; \Gamma^{\uparrow} \vdash E_1{:}A^{\uparrow} \to B^{\uparrow} \qquad \Delta; \Gamma \vdash e_2{:}A \nearrow \Delta^{\uparrow}; \Gamma^{\uparrow} \vdash E_2{:}A^{\uparrow}}{\Delta; \Gamma \vdash e_1e_2{:}B \nearrow \Delta^{\uparrow}; \Gamma^{\uparrow} \vdash E_1E_2{:}B^{\uparrow}}\ \text{app2\_rs}$$

As can be seen in the similarity of the two translation judgements, a term which is first lowered, then raised or the other way around will be translated back into the original term.

7 Conclusion and Future Work

At this point, the knowledge I have accumulated throughout the summer studying the various systems have been exposed, as well as the additional insights I had in regards to lowering and raising. The reason Pientka and I chose to pursue studying lowering and raising towards the end of the summer was twofold.

The first reason was based in the interested of studying the property of raising and lowering in itself and trying to understand the bridge between contextual types and non-contextual types. While progress was made in the understanding of the lowering and raising in the explicit system, some work is still required to formalize it fully. This includes more formal definitions and lemmas about what a raised/lowered type is, as well as some correctness-of-translation theorems.

The second reason was based in trying to construct a translation from the explicit contextual system to a implicit contextual system, referred to as $\lambda_{ic}^{\rightarrow\Box}$ here, first described by Hu and Pientka[HP22]. Recall that this was the main goal of the project originally. The main idea was to be able to lower and raise terms in both of these systems, and then rely on the translation given by Pfenning and Davies, summarized in sections 4 and 5, to translate contextual terms between the two systems. This idea is shown in the commutative diagram below in figure 1.

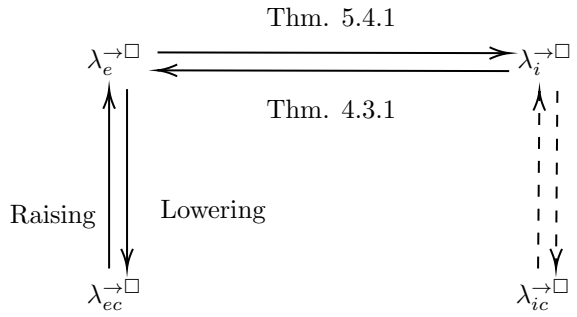

Figure 1: Commutative diagram of given translations (full lines) and potentially future translations (dashed lines) between the calculi studied in this report, as well as $\lambda_{ic}^{\rightarrow\Box}$ given by Hu and Pientka.

As time ran out for this project however, I did not get the time to study $\lambda_{ic}^{\rightarrow\Box}$ in depth and therefore also not investigate raising and lowering in $\lambda_{ic}^{\rightarrow\Box}$. For future possible work, constructing such a lowering and raising translation in $\lambda_{ic}^{\rightarrow\Box}$ is a worthwhile complement to the already know translations presented here in order to translate terms between $\lambda_{ic}^{\rightarrow\Box}$ and $\lambda_{ec}^{\rightarrow\Box}\lambda_{ic}^{\rightarrow\Box}$. Moreover, it is also interesting to look into the possibility of constructing a direct translation between these two contextual systems, either independent of the idea of raising/lowering or by taking inspiration from it but cutting some intermediate steps.

References

[DP01] Rowan Davies and Frank Pfenning. "A Modal Analysis of Staged Computation". In: *J. ACM* 48.3 (May 2001), pp. 555–604. ISSN: 0004-5411. DOI: 10.1145/382780.382785. URL: https://doi.org/10.1145/382780.382785.

[Pie03] Brigitte Pientka. "Tabled Higher-Order Logic Programming". PhD thesis. Carnegie Mellon University, Dec. 2003.

[NPP08] Aleksandar Nanevski, Frank Pfenning, and Brigitte Pientka. "Contextual Modal Type Theory". In: *ACM Trans. Comput. Logic* 9.3 (June 2008). ISSN: 1529-3785. DOI: 10.1145/1352582.1352591. URL: https://doi.org/10.1145/1352582.1352591.

[HP22] Jason Hu and Brigitte Pientka. "A Categorical Normalization Proof forthe Modal Lambda-Calculus". Submitted to MFPS 22. 2022.