Typed Operational Semantics for Dependent Record Types*

Yangyue Feng and Zhaohui Luo

Department of Computer Science
Royal Holloway, University of London
Egham, Surrey TW20 0EX, UK

{yangyue,zhaohui}@cs.rhul.ac.uk

Typed operational semantics is a method developed by H. Goguen to prove meta-theoretic properties of type systems. This paper studies the metatheory of a type system with dependent record types, using the approach of typed operational semantics. In particular, the metatheoretical properties we have proved include strong normalisation, Church-Rosser and subject reduction.

1 Introduction

H. Goguen [Gog94, Gog99] has developed a method called *typed operational semantics* (TOS for short) to prove meta-theoretic properties of type theories, including strong normalisation, Church-Rosser and subject reduction. In this paper, using the TOS approach, we study the meta-theoretic properties of a type system with dependent record types.

A record type is a type of labelled tuples called records. A dependent record type (DRT) is a type of records whose fields may have types that depend on the values of earlier fields. Dependent records have been studied previously for various different type systems [HL94, BT98, Pol02, CPT05], with applications to the study of module mechanisms for both programming and proof languages. Recently, in the context of studying manifest fields of module types, the second author has proposed a formulation of dependent record types [Luo09b], for type theories with canonical objects such as Martin-Löf's type theory, and shown in [Luo09a] that, in some applications, dependent record types are more useful than Σ-types (dependent types of tuples without labels).

Studying the meta-theory of dependent record types, the contributions of the current paper are two-fold. First of all, the meta-theory of dependent record types has not been well-studied. This work makes a positive contribution, showing that our formulation of dependent record types has the good meta-theoretic properties such as strong normalisation. Secondly, the type theory we study has record *types* as studied in [Pol02, Luo09b], rather than record *kinds* as in [BT98, CPT05]. Since types have a much more sophisticated structure than kinds, the meta-theory for dependent record types is expected to be much more difficult than that for dependent record kinds as found in, e.g., [CPT05]. We shall study the meta-theory by taking the TOS approach, which is shown to be robust enough to deal with dependent record types. In particular, we study the *intensional* DRTs, that is, the dependent record types without the so-called weakly extensional rules (these rules are considered in [Luo09b]). The typed operational semantics for intensional DRTs is developed and shown to be sound and complete and, based on this, it is proved that the intensional DRTs have good meta-theoretic properties, including strong normalisation, Church-Rosser and subject reduction.

The paper is arranged as follows. The type system IDRT for intensional DRTs is described in Section 2. In Section 3, after introducing the basic idea of TOS, we define the TOS for dependent record types. The properties of the TOS are studied in Section 4 and the meta-theoretic properties of IDRT in Section 5. Discussions of related work and future work are given in the conclusion.

2   Dependent Record Types

A dependent record type is a type of labelled tuples whose fields may have types that depend on the values of earlier fields. For instance, if *Nat* and *Vect*(n) are the types of natural numbers and vectors of length n, respectively, $\langle n : Nat, \ v : Vect(n) \rangle$ is the dependent record type with objects (called *records*) such as $\langle n = 2, \ v = [5,6] \rangle$, where dependency is respected: the vector $[5,6]$ must be of type *Vect*(2).

Formally, in our study, dependent record types are formulated as an extension of the logical framework that we describe briefly first.

Logical Framework.   LF [Luo94] is the typed version of Martin-Löf's logical framework [NPS90]. It is itself a type system that serves as a meta-language to specify type theories such as Martin-Löf's intensional type theory [NPS90] and the Unifying Theory of dependent Types (UTT) [Luo94]. Here, we give only a brief introduction, fixing the notations to be used in the paper. (For details of, for example, how inductive types like *Nat*, Π-types and Σ-types can be specified in the logical framework, see Part III of [NPS90] or Chapter 9 of [Luo94].)

In LF, the syntactical entities *contexts*, *kinds* and *terms* are of the following forms:

$$
\begin{array}{rcl}
Contexts & \Gamma \ ::= & () \mid \Gamma, x : A \\
LF\ Kinds & K \ ::= & Type \mid El(A) \mid (x{:}K)K' \\
LF\ Terms & M \ ::= & x \mid [x{:}K]M \mid M(M')
\end{array}
$$

The types in LF are called *kinds*, including:

- *Type* – the kind representing the collection of all types (A is a type if $A : Type$);

- *El*(A) – the kind of objects of type A (we often omit *El*); and

- $(x{:}K)K'$ (or simply $(K)K'$ when $x \notin FV(K')$) – the kind of dependent functional operations.

The judgement forms in LF include, for example,

- $\Gamma \vdash k : K$, which asserts that k is an object of kind K; and

- $\Gamma \vdash k = k' : K$, which asserts that k and k' are (computationally) equal objects of kind K.

The inference rules of LF to define the typing relation and the computational equality are given in Appendix A. In particular, $\beta\eta$-equal objects are computationally equal. For instance, an abstraction $[x{:}K]M$ can be applied to form $([x{:}K]M)(a)$ that is computationally equal to $[a/x]M$.

Notation   We shall use $\equiv$ to denote the syntactical identity (up to α-conversion).

Dependent Record Types.   We now give a formal presentation of the system IDRT of intensional dependent record types, which is an extension of LF. The syntax of this type system is given as follows, where $\mathcal{L}$ is an (infinite) set of labels, $l \in \mathcal{L}$ and $L \subset \mathcal{L}$ is finite:

$$
\begin{array}{rcl}
Kinds\ of\ Record\ Types & K_R & ::=\ \ RType \mid RType[L] \\
Record\ Types & R & ::=\ \ \langle\rangle \mid \langle R, l:A \rangle \\
Records & r & ::=\ \ \langle\rangle \mid \langle r, l=a:A \rangle
\end{array}
$$

The inference rules of IDRT consist of the rules for LF (Appendix A) and the additional rules in Figure 1. Here are some informal explanations.

- We add new kinds *RType* and *RType*[L] of record types. Intuitively, *RType*[L] is the kind of the record types whose (top-level) labels are all in *L*, a finite set of labels. Naturally, if $L \subseteq L'$, every record type in *RType*[L] is also in *RType*[L']. The kind *RType* is the kind of all record types and could conceptually be understood as '*RType*[$\mathcal{L}$]'. Finally, every record type is also a type. These are formally reflected in the rules for the kinds of record types in Figure 1.

- Record types are types of the form $\langle\rangle$ or $\langle R, l:A \rangle$. Intuitively, a record type is of the form $\langle l_1:A_1, ..., l_n:A_n \rangle$,[1] where each $l_i:A_i$ is a *field* labelled by *l*. An object of this record type is a labelled tuple $\langle l_1=a_1:A_1, ..., l_n=a_n:A_n \rangle$, where a_i is of the type of the corresponding field. Note that, formally, each A_i in the record type is not a type, but a family of types; this is how dependency is incorporated – we have dependent record types.

  Notation-wise, we shall adopt the following notational conventions: for record types, we write $\langle l_1:A_1, ..., l_n:A_n \rangle$ for $\langle\langle\langle\rangle, l_1:A_1 \rangle, ..., l_n:A_n \rangle$ and often use label occurrences/non-occurrences to show dependency/non-dependency respectively. For instance, we write $\langle n:Nat, v:Vect(n) \rangle$ for $\langle\langle\langle\rangle, n:NAT \rangle, v:[x:\langle n:NAT \rangle]Vect(x.n) \rangle$ where $NAT \equiv [\_:\langle\rangle]Nat$, and $\langle R, l:Vect(2) \rangle$ for $\langle R, l:[\_:R]Vect(2) \rangle$.

- There are two operations on records: *restriction* (or first projection) [r] that removes the last component of record *r* and *field selection r.l* that selects the value of the field labelled by *l*. For instance, intuitively, for the record $r \equiv \langle l_1=a_1:A_1, l_2=a_2:A_2, l_3=a_3:A_3 \rangle$ of type $\langle l_1:A_1, l_2:A_2, l_3:A_3 \rangle$, we have $[r] = \langle l_1:A_1, l_2:A_2 \rangle$ and $r.l_2 = [r].l_2 = a_2$. These are formally reflected in the introduction, elimination and computation rules in Figure 1.

- The congruence rules for record types and records in Figure 1 propagate the computational equality through the term structure. Also, we do not include the weakly extensional equality rules as considered in [Luo09b]. Therefore, we call the system the type system for intensional DRTs.

We shall adopt the following terminology: the terms of the form $\langle r, l=a:A \rangle$ will be called *pair-records*. (For example, we shall use this terminology in specifying the TOS-rules for record types in Figure 3 in Section 3.2.)

Record types v.s. record kinds.   It is worth pointing out that our type system contains dependent record *types* (as studied by Pollack [Pol02], Luo [Luo09b, Luo09a] and the current paper), rather than dependent record *kinds* (as studied by Betarte and Tasistro [BT98] and Coquand, Pollack and Takeyama [CPT05][2]). We would like to distinguish these two notions clearly: in a type theory with inductive types,

---

[1]We overload the $\langle ... \rangle$ notation for records and their types. It is always possible to distinguish between the two.

[2]*Types* in the terminology of Martin-Löf's type theory are what we call *kinds* in this paper. Therefore, the so-called record types in [BT98] and [CPT05] are really record kinds.

Kinds of record types

$$\frac{\Gamma\,valid}{\Gamma \vdash RType\,kind} \quad \frac{\Gamma\,valid}{\Gamma \vdash RType[L]\,kind}$$

$$\frac{\Gamma \vdash R : RType[L] \quad L \subseteq L'}{\Gamma \vdash R : RType[L']} \quad \frac{\Gamma \vdash R : RType[L]}{\Gamma \vdash R : RType} \quad \frac{\Gamma \vdash R : RType}{\Gamma \vdash R : Type}$$

Formation rules

$$\frac{\Gamma\,valid}{\Gamma \vdash \langle \rangle : RType[\emptyset]} \quad \frac{\Gamma \vdash R : RType[L] \quad \Gamma \vdash A : (R)Type \quad l \notin L}{\Gamma \vdash \langle R,\, l : A \rangle : RType[L \cup \{l\}]}$$

Introduction rules

$$\frac{\Gamma\,valid}{\Gamma \vdash \langle \rangle : \langle \rangle} \quad \frac{\Gamma \vdash \langle R,\, l : A \rangle : RType \quad \Gamma \vdash r : R \quad \Gamma \vdash a : A(r)}{\Gamma \vdash \langle r,\, l = a : A \rangle : \langle R,\, l : A \rangle}$$

Elimination rules

$$\frac{\Gamma \vdash r : \langle R,\, l : A \rangle}{\Gamma \vdash [r] : R} \quad \frac{\Gamma \vdash r : \langle R,\, l : A \rangle}{\Gamma \vdash r.l : A([r])}$$

$$\frac{\Gamma \vdash r : \langle R,\, l : A \rangle \quad \Gamma \vdash [r].l' : B \quad l \neq l'}{\Gamma \vdash r.l' : B}$$

Computation rules

$$\frac{\Gamma \vdash \langle r,\, l = a : A \rangle : \langle R,\, l : A \rangle}{\Gamma \vdash [\langle r,\, l = a : A \rangle] = r : R} \quad \frac{\Gamma \vdash \langle r,\, l = a : A \rangle : \langle R,\, l : A \rangle}{\Gamma \vdash \langle r,\, l = a : A \rangle.l = a : A(r)}$$

$$\frac{\Gamma \vdash r : \langle R,\, l : A \rangle \quad \Gamma \vdash [r].l' : B \quad l \neq l'}{\Gamma \vdash r.l' = [r].l' : B}$$

Congruence rules for record types

$$\frac{\Gamma\,valid}{\Gamma \vdash \langle \rangle = \langle \rangle : RType[\emptyset]} \quad \frac{\Gamma \vdash R = R' : RType[L] \quad \Gamma \vdash A = A' : (R)Type \quad l \notin L}{\Gamma \vdash \langle R,\, l : A \rangle = \langle R',\, l : A' \rangle : RType[L \cup \{l\}]}$$

Congruence rules for records

$$\frac{\Gamma\,valid}{\Gamma \vdash \langle \rangle = \langle \rangle : \langle \rangle} \quad \frac{\Gamma \vdash R : RType[L] \quad l \notin L \quad \Gamma \vdash r = r' : R \quad \Gamma \vdash a = a' : A(r) \quad \Gamma \vdash A = A' : (R)Type}{\Gamma \vdash \langle r,\, l = a : A \rangle = \langle r',\, l = a' : A' \rangle : \langle R,\, l : A \rangle}$$

$$\frac{\Gamma \vdash r = r' : \langle R, l : A \rangle}{\Gamma \vdash [r] = [r'] : R} \quad \frac{\Gamma \vdash r = r' : \langle R, l : A \rangle}{\Gamma \vdash r.l = r'.l : A([r])}$$

Figure 1: Inference Rules of IDRT

types include those such as *Nat* of natural numbers and Σ-types of dependent pairs, while the examples of kinds include, for example, the kind *Type* of all types. They exist at two completely different levels and have rather different structures and properties.

In general, types have a much more sophisticated and richer structure than kinds. For instance, it is easy to show that a kind is of the form either *Type* or $(x{:}K)K'$, but types are not (e.g., a type may be of the form $f(a)$). To appreciate the difference, let us consider the issue of ensuring label distinctness. If one considers only record kinds, it is easy to guarantee that the labels in the same record kind are distinct because of the limited syntactic forms of kinds (see, for example, [CPT05]). However, this is not easy at all for record types (think, for example, how one ensures that a label does not occur in a type of the form $f(a)$). In our case, we have to introduce the kinds $RType[L]$ to ensure that it is the case that the (top-level) labels in the same record type are distinct. In other words, intuitively, $l \neq l'$ for any record type $\langle ..., \, l : A, \, ..., \, l' : A', \, ... \rangle$. This is guaranteed by means of the side condition $l \notin L$ of the second formation rule in Figure 1.

That a type system with record types is more powerful than one with only record kinds can be understood from another angle when one wants to introduce universes of record types. It is possible to introduce type universes for dependent record types, as shown in [Luo09a]; this, however, cannot be done for record kinds. Therefore, record types are more useful than record kinds (for example, in representing module types in data refinement [Luo09a]).

Since types have a more sophisticated structure than kinds, it is more difficult to study the meta-theoretic properties of a system with record types, as compared with a meta-theoretic study of record kinds. As we show in this paper, the approach of using typed operational semantics can be used in this endeavour.

3   Typed Operational Semantics for Dependent Record Types

The typed operational semantics (TOS for short) is a proof-theoretic method to prove the meta-theoretic properties of type theories. It was developed by H. Goguen in his PhD thesis [Gog94], where he studied the meta-theory of UTT and proved that UTT has the nice properties such as Church-Rosser, Subject Reduction and Strong Normalisation.

In this paper, the TOS approach is applied to study the meta-theory of dependent record types. After a brief informal introduction of the approach, we develop the typed operational semantics for the system IDRT of intensional DRTs and show that it has the soundness and completeness properties. The meta-theoretic properties of dependent record types are studied in the next section.

3.1   The TOS Approach

For a type theory, its typed operational semantics captures its computational behaviour, usually given by its (untyped) reduction relation. For example, in TOS, the following judgement

$$\Gamma \models M \rightarrow N \rightarrow P : A$$

informally asserts that, among other things, N and P are the weak-head normal form and the normal form of the term M, respectively.[3] For the logical framework LF, for example, its corresponding TOS has been studied [Gog99] and its inference rules are given in Appendix B. Since many meta-theoretic properties

---

[3]Formally, the reduction relation and the TOS are related to each other by means of the 'adequacy theorems' such as Lemmas 4.4 and 4.5 for IDRT in Section 4.2.

$$
\begin{array}{cc}
\textit{Basic forms:} & \textit{Abbreviated forms:} \\
\models \Gamma \rightarrow \Delta & \Gamma \models ok \\
\Gamma \models A \rightarrow B & \Gamma \models M \rightarrow_w N : A \\
\Gamma \models M \rightarrow N \rightarrow P : A & \Gamma \models M \rightarrow_n P : A \\
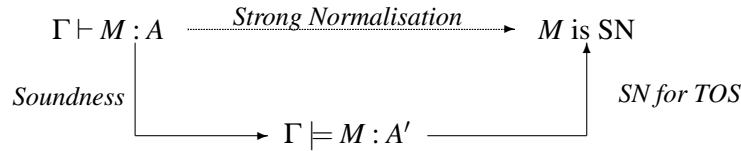& \Gamma \models M : A
\end{array}
$$

Figure 2: Judgement Forms in Typed Operational Semantics

of a type theory are concerned with its computational behaviour, it is not a surprise that TOS provides an effective approach to the meta-theory of type theories.[4]

The TOS and its corresponding type theory are related to each other by means of the soundness and completeness theorems. Using the judgement $\Gamma \models M : A$ to abbreviate '$\Gamma \models M \rightarrow N \rightarrow P : A$ for some N and P', we can state the soundness and completeness properties as follows:

- Soundness: $\Gamma \vdash M : A$ implies $\Gamma \models M : A'$ (for A' that is the 'normal form' of A).

- Completeness: $\Gamma \models M : A$ implies $\Gamma \vdash M : A$.

Based on soundness and completeness, we can prove many meta-theoretic properties of the type theory. For example, it can be shown that, if $\Gamma \models M : A'$, then M is strongly normalisable. Therefore, strong normalisation, the property that every well-typed term is strongly normalisable, can be proved by means of such a fact together with the soundness property, as pictured as follows:

$$
\begin{array}{ccc}
\Gamma \vdash M : A & \xrightarrow{\textit{Strong Normalisation}} & M \text{ is SN} \\
\text{\small \textit{Soundness}} \downarrow & & \uparrow \text{\small \textit{SN for TOS}} \\
& \Gamma \models M : A' &
\end{array}
$$

As shown in this paper, for dependent record types, the SN property for the corresponding TOS is proven in Theorem 4.13. Then, by the Soundness Theorem (Theorem 4.8), we can show that strong normalisation for IDRT (Corollary 5.2).

Note that, to implement such ideas is not a simple matter: it requires one to prove:

- that the TOS is 'adequate' w.r.t. the (untyped) reduction relation,

- that the TOS is sound and complete w.r.t. the original type theory, and

- that the TOS satisfies some specific meta-theoretic properties (e.g., strong normalisation).

Then, one can transfer the results to the original type theory to show that it has nice meta-theoretic properties. This is what we shall do for IDRT, the type theory with dependent record types.

3.2 TOS for Dependent Record Types

The typed operational semantics for dependent record types is described in this section. The judgement forms in a TOS are given in Figure 2, three of which are the basic forms of judgements whose informal meanings are:

---

[4]It is worth noting that, although it is useful to study the meta-theory for many type theories, the TOS approach would not be suitable for non-normalising type theories. See [Gog94] for discussions.

- $\models \Gamma \to \Delta$: the context Γ has context Δ as its normal form;

- $\Gamma \models A \to B$: the kind A is well-formed in context Γ and has normal form B; and

- $\Gamma \models M \to N \to P : A$: the terms M, N, P are well-formed in context Γ of kind A and M has weak-head normal form N and normal form P.

From these basic judgements, one can define other forms of judgements, including the following:

- $\Gamma \models ok$ stands for '$\models \Gamma \to \Delta$ for some Δ';

- $\Gamma \models M \to_w N : A$ stands for '$\Gamma \models M \to N \to P : A$ for some P';

- $\Gamma \models M \to_n P : A$ stands for '$\Gamma \models M \to N \to P : A$ for some N'; and

- $\Gamma \models M : A$ stands for '$\Gamma \models M \to N \to P : A$ for some N and P'.

The typed operational semantics for the type system IDRT of intensional DRTs is the extension of that for LF (Appendix B) with the inference rules given in Figure 3. Most of the rules are self-explanatory. We only mention that, besides using the abbreviated forms of judgement (see above) in the rules, we also use the terminology of 'pair-record' as introduced in Section 2. For example, in ($BASE_{RESTR}$), we require that p or q be not a pair-record, for otherwise, for instance, $[p]$ could be a redex and would not be in normal form.

4   Properties of TOS for Dependent Record Types

We shall study the properties of the TOS for IDRT, as presented above in Section 3.2. These include those properties w.r.t. the relationship with IDRT (soundness and completeness) and those w.r.t. the reduction relation.

4.1   Basic Structural Properties

The typed operational semantics satisfy some basic properties as stated in the following lemma, which can all be proved by induction on the TOS-derivations.[5]

Lemma 4.1

1. *(Context Validity) Any derivation of $\Gamma_0, \Gamma_1 \models J$ has a sub-derivation of $\Gamma_0 \models ok$.*

2. *(Variables) Let $dom\{\Gamma\}$ be the set of variables declared in context Γ and $FV(M)$ the set of free variables occurring in term M.*

   (a) If $\models \Gamma \to \Delta$, then $dom\{\Delta\} = dom\{\Gamma\}$.
   (b) If $\Gamma \models A \to B$, then $FV(A) \cup FV(B) \subseteq dom\{\Gamma\}$.
   (c) If $\Gamma \models M \to N \to P : A$, then $FV(M) \cup FV(N) \cup FV(P) \cup FV(A) \subseteq dom\{\Gamma\}$.

3. *(Weakening) If $\Gamma \models J$ and $\Gamma, \Delta \models ok$, then $\Gamma, \Delta \models J$.*

4. *(Strengthening) If $\Gamma_0, z{:}C, \Gamma_1 \models J$ and $z \notin FV(\Gamma_1) \cup FV(J)$, then $\Gamma_0, \Gamma_1 \models J$.*

5. *(Determinacy)*
   - *If $\models \Gamma \to \Delta$ and $\models \Gamma \to \Phi$, then $\Delta \equiv \Phi$.*

---

[5]Some of the lemmas (e.g., the strengthening lemma) can only be proved by proving a stronger statement by induction on derivations. We omit the details here.

Record Kinds

$$\frac{\Gamma \models ok}{\Gamma \models RType \to RType} \; RTYPE \qquad \frac{\Gamma \models ok}{\Gamma \models RType[L] \to RType[L]} \; RTYPE[L]$$

Record Types

$$\frac{\Gamma \models ok}{\Gamma \models \langle\rangle \to \langle\rangle \to \langle\rangle : RType[\emptyset]} \; EMP_{RCDT}$$

$$\frac{\Gamma \models R \to_n P : RType[L] \quad \Gamma \models A \to_n B : (P)Type \quad l \notin L}{\Gamma \models \langle R, l : A\rangle \to \langle R, l : A\rangle \to \langle P, l : B\rangle : RType[L \cup \{l\}]} \; RCDT$$

Pair-records

$$\frac{\Gamma \models ok}{\Gamma \models \langle\rangle \to \langle\rangle \to \langle\rangle : \langle\rangle} \; EMP_{RCD}$$

$$\frac{\Gamma \models \langle R, l : A\rangle \to_n \langle P, l : B\rangle : RType \quad \Gamma \models r \to_n p : P}{\Gamma \models A(r) \to_n C : Type \quad \Gamma \models a \to_n b : C} \; RCD$$
$$\frac{}{\Gamma \models \langle r, l = a : A\rangle \to \langle r, l = a : A\rangle \to \langle p, l = b : B\rangle : \langle P, l : B\rangle}$$

Restrictions

$$\frac{\Gamma \models r \to q \to p : \langle P, l : B\rangle \quad p, q \; not \; pair\text{-}records}{\Gamma \models [r] \to [q] \to [p] : P} \; BASE_{RESTR}$$

$$\frac{\Gamma \models r \to_w \langle p, l = b : A\rangle : \langle P, l : B\rangle \quad \Gamma \models p \to s \to t : P}{\Gamma \models [r] \to s \to t : P} \; RESTR$$

Selections

$$\frac{\Gamma \models r \to q \to p : \langle P, l : B\rangle \quad p, q \; not \; pair\text{-}records}{\Gamma \models B([r]) \to_n C : Type}$$
$$\frac{}{\Gamma \models r.l \to q.l \to p.l : C} \; BASE_{FLDSEL}$$

$$\frac{\Gamma \models r \to_w \langle p, l = b : A\rangle : \langle P, l : B\rangle \quad \Gamma \models b \to c \to d : C \quad \Gamma \models A(p) \to_n C : Type}{\Gamma \models r.l \to c \to d : C} \; FLDSEL$$

$$\frac{\Gamma \models r \to_n s : \langle P, l : B\rangle \quad \Gamma \models [r].l' \to c \to d : C \quad l \neq l'}{\Gamma \models r.l' \to c \to d : C} \; FLDSL'$$

Figure 3: Inference Rules of Typed Operational Semantics for IDRT

- *If* $\Gamma \models A \rightarrow B$ *and* $\Gamma \models A \rightarrow C$, *then* $B \equiv C$.
- *If* $\Gamma \models M \rightarrow N \rightarrow P : B$ *and* $\Gamma \models M \rightarrow Q \rightarrow R : C$, *then* $N \equiv Q$, $P \equiv R$ *and* $B \equiv C$.

Remark The above Lemma 4.1(5) of 'Determinacy' says that the TOS-normal forms are unique. Of course, in order to show that the normal form of a well-typed term (under the usual reduction relation) is unique, one has to prove that the TOS-reductions are adequate. This is what we do in the following subsection.

4.2 Adequacy w.r.t. the Untyped Reduction

We shall show in this section that the notions of computation captured in TOS are adequate w.r.t. the usual (untyped) reduction relation, which is defined in the following definition.

Definition 4.2 (Untyped Reduction for IDRT) *The (untyped) one-step reduction over terms, notation $\rightarrow$, is the compatible closure*[6] *of the relation given by the following rules:*

$$
\begin{array}{llll}
(\beta) & ([x{:}A]M)N & \rightarrow & [N/x]M \\
(\eta) & [x{:}A]M(x) & \rightarrow & M & (x \notin FV(M)) \\
(\pi_1) & [\langle r, l = a : A \rangle] & \rightarrow & r \\
(\pi_2) & \langle r, l = a : A \rangle.l & \rightarrow & a \\
(\pi_2') & \langle r, l = a : A \rangle.l' & \rightarrow & r.l' & (l \neq l')
\end{array}
$$

We write $\rightarrow^+$ and $\rightarrow^*$ for the corresponding transitive closure and reflexive and transitive closure, respectively.

   A term of the form on the left of an arrow is called a redex. *For example, a π_2-redex is a term of the form $\langle r, l = a : A \rangle.l$.*

Definition 4.3 (Weak-Head Normal Forms and Normal Forms)

- *A term M is* weak-head normal *if*
    - *$M \equiv x$ is a variable;*
    - *$M \equiv [x{:}K]k$;*
    - *$M \equiv f(a)$, where f is weak-head normal and not an abstraction;*
    - *$M \equiv \langle \rangle$;*
    - *$M \equiv \langle r, l = a : A \rangle$; or*
    - *$M \equiv [r]$ or $M \equiv r.l$, where r is weak-head normal and not a pair-record.*
- *A term M is* normal *if*
    - *$M \equiv x$ is a variable;*
    - *$M \equiv [x{:}K]k$, which is not an η-redex, and K and k are normal;*
    - *$M \equiv f(a)$ and f and a are normal and f not an abstraction;*
    - *$M \equiv \langle \rangle$;*
    - *$M \equiv \langle r, l = a : A \rangle$ and r, a and A are normal.*

---

[6]The compatible closure of a relation R over terms propagates R to all of the terms. We omit its formal definition here; see [Gog94, Fen10] for formal details.

    – *$M \equiv [r]$ or $M \equiv r.l$, where r is normal and not a pair-record.*

The notions of weak-head normal forms and normal forms are lifted to record types, kinds and contexts in the usual way.

This case was in our original proof of a DRT system with the WER rules, it was of interest because the weakly extensional rules are η-like rules that cause problems, such as strong normalization fails for untyped raw terms. For reason of discussion we keep this case still here. The following lemmas show that the notion of computation captured in TOS is adequate w.r.t. the untyped reduction and the associated notions of normal forms.

Lemma 4.4 (Adequacy of TOS w.r.t. Untyped Reduction)

- *If $\Gamma \models A \to C$ then there exists B such that $A \to^*_{\beta R} B \to^*_\eta C$.*
- *If $\Gamma \models M \to N \to P : A$, then there exists N' such that $M \to^*_{\beta R} N \to^*_{\beta R} N' \to^*_\eta P$.*

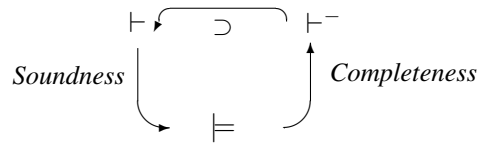**Proof.** By induction on derivations.

Lemma 4.5 (Adequacy of TOS w.r.t. Normal Forms and WHNFs)

- *If $\models \Gamma \to \Delta$, then Δ is normal.*
- *If $\Gamma \models A \to B$, then B is normal.*
- *If $\Gamma \models M \to N \to P : A$, then N is weak-head normal and P and A are normal.*

Proof. By induction on derivations.

4.2.1 Soundness and Completeness

The TOS we have studied is sound and complete w.r.t. the type system IDRT of dependent record types. In the informal introduction to TOS in Section 3.1, we have over-simplified the situation. In fact, what we shall do is to show that completeness holds for a simpler system $IDRT^-$ (with judgements of the form $\Gamma \vdash^- J$), which is obtained from IDRT by removing the seven substitution rules in Appendix A. Therefore, the soundness and completeness may be pictured as follows:

$$\vdash \underbrace{\qquad \supset \qquad}_{} \vdash^-$$

Soundness $\Big\downarrow \qquad\qquad \Big\uparrow$ *Completeness*

$$\models$$

Theorem 4.6 (Completeness of TOS w.r.t. $IDRT^-$)

- *If $\Gamma \models ok$ then $\vdash^- \Gamma$ valid.*
- *If $\Gamma \models A \to B$ then $\Gamma \vdash^- A$ kind and $\Gamma \vdash^- A = B$.*
- *If $\Gamma \models M \to N \to P : A$ then $\Gamma \vdash^- M : A$, $\Gamma \vdash^- M = N : A$, $\Gamma \vdash^- M = P : A$ and $\Gamma \vdash^- A = A$.*

Proof. By simultaneous induction on derivations and examining each case of the *TOS* inference rules.

Corollary 4.7 (Completeness of TOS w.r.t. IDRT)

- *If $\Gamma \models ok$ then $\vdash \Gamma$ valid.*
- *If $\Gamma \models A \to B$ then $\Gamma \vdash A$ kind and $\Gamma \vdash A = B$.*

- *If $\Gamma \models M \to N \to P : A$ then $\Gamma \vdash M : A$, $\Gamma \vdash M = N : A$, $\Gamma \vdash M = P : A$ and $\Gamma \vdash A = A$.*

Proof. By Theorem 4.6 and the inclusion of $IDRT^-$ in IDRT.

The Soundness Theorem is harder to prove. We have to consider all the inference rules of IDRT including the structural rules. In the following, we only consider some selected cases. The detailed proof can be found in [Fen10].

Theorem 4.8 (Soundness of TOS w.r.t. IDRT)

- *If $\Gamma \vdash ok$, then there exists Δ such that $\models \Gamma \to \Delta$.*
- *If $\Gamma \vdash A$ kind, then there exists B such that $\Gamma \models A \to B$.*
- *If $\Gamma \vdash A = B$ then there exists C such that $\Gamma \models A \to C$ and $\Gamma \models B \to C$.*
- *If $\Gamma \vdash M : A$ then there exist P, B such that $\Gamma \models A \to B$ and $\Gamma \models M \to_n P : B$.*
- *If $\Gamma \vdash M = N : A$, then there exist P, B such that $\Gamma \models A \to B$, and $\Gamma \models M \to_n P : B$, $\Gamma \models N \to_n P : B$.*

Proof. By induction on derivations. For the cases of LF-rules, see [Gog99]. We consider the following two cases about record types.

- The second introduction rule in Figure 1:

$$\frac{\Gamma \vdash \langle R, \, l : A \rangle : RType \quad \Gamma \vdash r : R \quad \Gamma \vdash a : A(r)}{\Gamma \vdash \langle r, \, l = a : A \rangle : \langle R, \, l : A \rangle}$$

  By induction hypothesis, the following hold:

  1. $\Gamma \models \langle R, l : A \rangle \to_n \langle P, l : B \rangle : RType$ for some P and B,
  2. $\Gamma \models r \to_n p : P'$ and $\Gamma \models R \to_n P' : RType[L]$ for some p, P' and L, and
  3. $\Gamma \models a \to_n b : C$ and $\Gamma \models A(r) \to C$ for some b and C.

  By Lemma 4.1(5) (Determinacy) and inversion of the rule $(RCDT)$ in Figure 3, $P \equiv P'$. Therefore, by rule (RCD) in Figure 3, $\Gamma \models \langle r, l = a : A \rangle \to_n \langle p, l = b : B \rangle : \langle P, l : B \rangle$.

- The third elimination rule in Figure 1:

$$\frac{\Gamma \vdash r : \langle R, \, l : A \rangle \quad \Gamma \vdash [r].l' : B \quad l \neq l'}{\Gamma \vdash r.l' : B}$$

  By induction hypothesis, the following hold:

  1. $\Gamma \models r \to_n s : \langle P, l : B \rangle$ and $\Gamma \models \langle R, l : A \rangle \to \langle P, l : B \rangle$ for some s, P and B, and
  2. $\Gamma \models [r].l' \to_n c : C$ and $\Gamma \models B \to C$ for some c and C.

  Since $l \neq l'$, by $(FLDSL')$ in Figure 3, we have $\Gamma \models r.l' \to_n c : C$.

4.3 Strong Normalisation in TOS

The strong normalisation property of the TOS says that, if a term M is well-typed in the TOS (i.e., $\Gamma \models M : A$), then M is strongly normalisable. This result, together with the soundness theorem, will then be enough to show that the original type theory has the property of strong normalisation.

The strong normalisation property of the TOS by introducing a notion of *parallel reduction* and showing that it has the so-called *Parallel Subject Reduction* property.

$$VAR \ \frac{}{x \Rightarrow x} \qquad \lambda \ \frac{A \Rightarrow A' \quad M \Rightarrow M'}{[x{:}A]M \Rightarrow [x{:}A']M'} \qquad APP \ \frac{M \Rightarrow M' \quad N \Rightarrow N'}{M(N) \Rightarrow M'(N')}$$

$$\beta \ \frac{M \Rightarrow M' \quad N \Rightarrow N'}{([x{:}A]M)(N) \Rightarrow [N'/x]M'} \qquad \eta \ \frac{M_0 \Rightarrow M'(x) \quad x \notin FV(M')}{[x{:}A_1]M_0 \Rightarrow M'}$$

$$RCDT_{EMP} \ \frac{}{\langle\rangle \Rightarrow \langle\rangle} \qquad RCDT \ \frac{R \Rightarrow R' \quad A \Rightarrow A'}{\langle R, l : A \rangle \Rightarrow \langle R', l : A' \rangle}$$

$$RCD_{EMP} \ \frac{}{\langle\rangle \Rightarrow \langle\rangle} \qquad RCD \ \frac{r \Rightarrow r' \quad a \Rightarrow a' \quad A \Rightarrow A'}{\langle r, l = a : A \rangle \Rightarrow \langle r', l = a' : A' \rangle}$$

$$BASE_{RESTR} \ \frac{r \Rightarrow r'}{[r] \Rightarrow [r']} \qquad BASE_{FLDSEL} \ \frac{r \Rightarrow r'}{r.l \Rightarrow r'.l}$$

$$RESTR \ \frac{r \Rightarrow r'}{[\langle r, l = a : A \rangle] \Rightarrow r'} \qquad FLDSEL \ \frac{a \Rightarrow a'}{\langle r, l = a : A \rangle.l \Rightarrow a'}$$

$$FLDSL' \ \frac{r \Rightarrow r' \quad l \neq l'}{\langle r, l = a : A \rangle.l' \Rightarrow r'.l'}$$

Figure 4: Parallel Reduction for IDRT

Definition 4.9 (Parallel Reduction) *Parallel reduction $\Rightarrow$ is defined as the least relation closed under the rules in Figure 4, and is extended in an obvious way to kinds and contexts.*

Remark Parallel reduction has some simple properties. First, $M \Rightarrow M$ for all M. Furthermore, if $M \to N$ then $M \Rightarrow N$, and if $M \Rightarrow N$ then $M \to^* N$. Finally, if $M \Rightarrow M'$ and $N \Rightarrow N'$ then $[N/x]M \Rightarrow [N'/x]M'$.

Lemma 4.10 (Parallel Subject Reduction)

1. *If $\models \Gamma \to \Delta$ and $\Gamma \Rightarrow \Gamma'$, then $\models \Gamma' \to \Delta$.*

2. *If $\Gamma \models A \to B$, $\Gamma \Rightarrow \Gamma'$ and $A \Rightarrow A'$, then $\Gamma' \models A' \to B$.*

3. *If $\Gamma \models M \to N \to P : A$, $\Gamma \Rightarrow \Gamma'$ and $M \Rightarrow M'$, then there exist N' and N'' such that $N \Rightarrow N'$, $\Gamma' \models M' \to N'' \to P : A$ and $\Gamma' \models N' \to N'' \to P : A$.*

Proof. By simultaneous induction on derivations. The detailed proof can be found in [Fen10].

Lemma 4.11 (Subject Reduction) *If $\Gamma \models M \to N \to P : A$, $\Gamma \Rightarrow \Gamma'$ and $M \Rightarrow M'$, then there exists N' such that $\Gamma' \models M' \to N' \to P : A$ and $N \to^* N'$.*

Proof. By simultaneous induction on derivations, using Lemma 4.10.

The proof of strong normalisation of the TOS (Theorem 4.13) uses the following lemma.

Lemma 4.12

• *If M is weak-head normal and not an abstraction and $M \to^* N$, then N is weak-head normal and not an abstraction.*

- *If M is weak-head normal and not a pair record and M →* N, then N is weak-head normal and not a pair record.*

Proof. By induction on length of reduction for untyped terms.

Theorem 4.13 (Strong Normalisation of TOS)

1. *If $\Gamma \models A \to B$ then A is strongly normalisable.*

2. *If $\Gamma \models M \to N \to P : A$ then M is strongly normalisable.*

Proof. By simultaneous induction on derivations. The full proof can be found in [Fen10]. We shall consider one of the most difficult cases – when the last rule used is ($BASE_{RESTR}$) in Figure 3:

$$\frac{\Gamma \models r \to q \to p : \langle P,\, l : B \rangle \quad p,\, q \text{ not pair-records}}{\Gamma \models [r] \to [q] \to [p] : P}$$

By induction hypothesis, r is strongly normalisable. It suffices for us to show that $[r]$ is strongly normalisable if $\Gamma \models r \to_w q : \langle P, l : B \rangle$ such that q is not a pair-record. We do this by induction on the maximal length of reductions starting from r.

Assume that $r \to r_1$. We then have $r \Rightarrow r_1$, which implies by Lemma 4.10 (Parallel Subject Reduction) that there exist r' and r'' such that

$$\Gamma \models r_1 \to_w r'' : \langle P, l : B \rangle, \quad \Gamma \models r' \to_w r'' : \langle P, l : B \rangle, \quad \text{and} \quad q \Rightarrow r'.$$

We therefore have that q is weak-head normal (by Lemma 4.5) and that $q \to^* r'$ (see the remark above). From these and Lemma 4.12, we have

(*) r' is weak-head normal and not a pair-record.

Furthermore, by Lemma 4.4, we have

(**) $r' \to^* r''$.

From (*) and (**), r'' is not a pair-record by Lemma 4.12. Therefore, since $\Gamma \models r_1 \to_w r'' : \langle P, l : B \rangle$, we conclude by induction hypothesis that $[r_1]$ is strongly normalisable.

5   Meta-theoretic Properties of IDRT

From the properties of the TOS that have been proved in the last section, the meta-theoretic properties of IDRT, the type theory for dependent record types, can be proved. Here, we give the theorems for Subject Reduction, Church-Rosser and Strong Normalisation. (For further details and other properties, see [Fen10].)

Theorem 5.1 (Subject Reduction for IDRT) *If $\Gamma \vdash M : A$ and $M \to N$, then $\Gamma \vdash N : A$.*

Proof. First of all, we have $\Gamma \models M : A$ (by the Soundness Theorem 4.8) and $M \Rightarrow N$ (since $M \to N$). Therefore, by Lemma 4.11, we have $\Gamma \models N : A$. Now, by Completeness (Theorem 4.7), $\Gamma \vdash N : A$.

Theorem 5.2 (Strong Normalisation for IDRT)

1. *If Γ valid, then Γ is strongly normalisable.*

2. *If $\Gamma \vdash A$ kind, then A is strongly normalisable.*

3. *If $\Gamma \vdash A = B$, then both A and B are strongly normalisable to some C.*

4. *If $\Gamma \vdash M : A$, then M and A are strongly normalisable and $\Gamma \vdash P : B$, where P and B are the normal forms of M and A, respectively.*

5. *If $\Gamma \vdash M = N : A$, then both M and N are strongly normalisable to some P, A is strongly normalisable to some B such that $\Gamma \vdash P : B$.*

Proof. By the Soundness Theorem 4.8 and Theorem 4.13.

Theorem 5.3 (Church-Rosser for IDRT) *If $\Gamma \vdash M = N : A$, then $M \to^* P$ and $N \to^* P$ for some P.*

Proof. By Soundness (Theorem 4.8), there exist P and B such that $\Gamma \models M \to_n P : B$ and $\Gamma \models N \to_n P : B$. Then, by Adequacy (Lemma 4.4), we have $M \to^* P$ and $N \to^* P$.

6 Conclusions

We have studied the meta-theory of a type theory with dependent record types, by studying its typed operational semantics. As we have mentioned in Section 2, dependent record *types* are rather different from dependent record *kinds*, with the former having a much richer structure and being more difficult to study. The meta-theory of dependent record kinds has been studied by Coquand *et. al* [CPT05], where they have given a proof of termination of type-checking. As far as we know, ours is the first attempt to study the meta-theory of dependent record types formulated in a logical framework [Pol02, Luo09b, Luo09a]. In the light that the TOS-approach has been successfully applied to the meta-theoretic study of type theories with η-equality [Gog99] and with inductive types [Gog94], the current work can be used to justify the incorporation of dependent record types in a full-scale type theory as implemented in the proof assistants such as Agda and Coq.

The dependent record types studied in this paper are *intensional* in the sense that we do not have the following extensional equality rules [BT98, Luo09b]:

$$\frac{\Gamma \vdash r : \langle\rangle}{\Gamma \vdash r = \langle\rangle : \langle\rangle} \qquad \frac{\Gamma \vdash r : \langle R, l : A\rangle \quad \Gamma \vdash r' : \langle R, l : A\rangle \quad \Gamma \vdash [r] = [r'] : R \quad \Gamma \vdash r.l = r'.l : A([r])}{\Gamma \vdash r = r' : \langle R, l : A\rangle}$$

They basically say that two records are computationally equal if their components are. For instance, from the second rule above, we would have $\langle r, l = r.l\rangle = r$ for any r of type $\langle R, l : A\rangle$. It is unclear whether the TOS-approach as adopted in this paper can be applied to such (weakly) extensional record types. It would be obviously problematic if one considered the reduction relation for the records as follows:

$$\langle r, l = r.l\rangle \to r$$

for, together with the η-reduction for λ-terms, the Church-Rosser property would fail to hold. A natural question arises here: would it possible if one takes the TOS-approach by considering a reduction relation that treats η-long normal forms (e.g., by taking the above reduction in the other direction)? This involves the development of the TOS-approach to incorporate η-long normal forms and research is needed to see whether it is possible.

References

[BT98]   G. Betarte and A. Tasistro. Extension of Martin-Löf's type theory with record types and subtyping. In G. Sambin and J. Smith, editors, *Twenty-five Years of Constructive Type Theory*. Oxford University Press, 1998.

[CPT05]   T. Coquand, R. Pollack, and M. Takeyama. A logical framework with dependently typed records. *Fundamenta Informaticae*, 65(1-2), 2005. doi:10.1007/3-540-44904-3_8.

[Fen10]   Y. Feng. *A theory of dependent record types with structural subtyping*. PhD thesis, Royal Holloway, Univ. of London, 2010.

[Gog94]   H. Goguen. *A Typed Operational Semantics for Type Theory*. PhD thesis, University of Edinburgh, 1994.

[Gog99]   H. Goguen. Soundness of typed operational semantics for the logical framework. *Typed Lambda Calculi and Applications (TLCA'99), LNCS'1581*, 1999. doi:10.1007/3-540-48959-2_14.

[HL94]   R. Harper and M. Lillibridge. A type-theoretic approach to higher-order modules with sharing. *POPL'94*, 1994. doi:10.1145/174675.176927.

[Luo94]   Z. Luo. *Computation and Reasoning: A Type Theory for Computer Science*. Oxford University Press, 1994.

[Luo09a] Z. Luo. Dependent record types revisited. In *Proc. of the 1st Inter. Workshop on Modules and Libraries for Proof Assistants (MLPA'09), Montreal*, volume 429 of *ACM Inter. Conf. Proceeding Series*, 2009. doi:10.1145/1735813.1735819.

[Luo09b] Z. Luo. Manifest fields and module mechanisms in intensional type theory. In *Types for Proofs and Programs, Proc. of Inter. Conf. of TYPES'08. LNCS 5497*, 2009. doi:10.1007/978-3-642-02444-3_15.

[NPS90]   B. Nordström, K. Petersson, and J. Smith. *Programming in Martin-Löf's Type Theory: An Introduction*. Oxford University Press, 1990.

[Pol02]   R. Pollack. Dependently typed records in type theory. *Formal Aspects of Computing*, 13:386–402, 2002. doi:10.1007/s001650200018.

A   Inference Rules of LF

The inference rules of the logical framework LF are given below. (See Chapter 9 of [Luo94] for further details.)

Contexts and assumptions

$$\frac{}{()\ \textit{valid}} \qquad \frac{\Gamma \vdash K\ \textit{kind}\quad x \notin FV(\Gamma)}{\Gamma, x{:}K\ \textit{valid}} \qquad \frac{\Gamma, x{:}K, \Gamma'\ \textit{valid}}{\Gamma, x{:}K, \Gamma' \vdash x : K}$$

General equality rules

$$\frac{\Gamma \vdash K\ \textit{kind}}{\Gamma \vdash K = K} \qquad \frac{\Gamma \vdash K = K'}{\Gamma \vdash K' = K} \qquad \frac{\Gamma \vdash K = K'\quad \Gamma \vdash K' = K''}{\Gamma \vdash K = K''}$$

$$\frac{\Gamma \vdash k : K}{\Gamma \vdash k = k : K} \qquad \frac{\Gamma \vdash k = k' : K}{\Gamma \vdash k' = k : K} \qquad \frac{\Gamma \vdash k = k' : K\quad \Gamma \vdash k' = k'' : K}{\Gamma \vdash k = k'' : K}$$

Equality typing rules

$$\frac{\Gamma \vdash k : K\quad \Gamma \vdash K = K'}{\Gamma \vdash k : K'} \qquad \frac{\Gamma \vdash k = k' : K\quad \Gamma \vdash K = K'}{\Gamma \vdash k = k' : K'}$$

Substitution rules

$$\frac{\Gamma,x{:}K,\Gamma' \ valid \ \ \Gamma \vdash k : K}{\Gamma,[k/x]\Gamma' \ valid}$$

$$\frac{\Gamma,x{:}K,\Gamma' \vdash K' \ kind \ \ \Gamma \vdash k : K}{\Gamma,[k/x]\Gamma' \vdash [k/x]K' \ kind} \quad \frac{\Gamma,x{:}K,\Gamma' \vdash K' \ kind \ \ \Gamma \vdash k = k' : K}{\Gamma,[k/x]\Gamma' \vdash [k/x]K' = [k'/x]K'}$$

$$\frac{\Gamma,x{:}K,\Gamma' \vdash k' : K' \ \ \Gamma \vdash k : K}{\Gamma,[k/x]\Gamma' \vdash [k/x]k' : [k/x]K'} \quad \frac{\Gamma,x{:}K,\Gamma' \vdash k' : K' \ \ \Gamma \vdash k_1 = k_2 : K}{\Gamma,[k_1/x]\Gamma' \vdash [k_1/x]k' = [k_2/x]k' : [k_1/x]K'}$$

$$\frac{\Gamma,x{:}K,\Gamma' \vdash K' = K'' \ \ \Gamma \vdash k : K}{\Gamma,[k/x]\Gamma' \vdash [k/x]K' = [k/x]K''} \quad \frac{\Gamma,x{:}K,\Gamma' \vdash k' = k'' : K' \ \ \Gamma \vdash k : K}{\Gamma,[k/x]\Gamma' \vdash [k/x]k' = [k/x]k'' : [k/x]K'}$$

The kind Type

$$\frac{\Gamma \ valid}{\Gamma \vdash Type \ kind} \quad \frac{\Gamma \vdash A : Type}{\Gamma \vdash El(A) \ kind} \quad \frac{\Gamma \vdash A = B : Type}{\Gamma \vdash El(A) = El(B)}$$

Dependent product kinds

$$\frac{\Gamma \vdash K \ kind \ \ \Gamma,x{:}K \vdash K' \ kind}{\Gamma \vdash (x{:}K)K' \ kind} \quad \frac{\Gamma \vdash K_1 = K_2 \ \ \Gamma,x{:}K_1 \vdash K_1' = K_2'}{\Gamma \vdash (x{:}K_1)K_1' = (x{:}K_2)K_2'}$$

$$\frac{\Gamma,x{:}K \vdash k : K'}{\Gamma \vdash [x{:}K]k : (x{:}K)K'} \quad \frac{\Gamma \vdash K_1 = K_2 \ \ \Gamma,x{:}K_1 \vdash k_1 = k_2 : K}{\Gamma \vdash [x{:}K_1]k_1 = [x{:}K_2]k_2 : (x{:}K_1)K}$$

$$\frac{\Gamma \vdash f : (x{:}K)K' \ \ \Gamma \vdash k : K}{\Gamma \vdash f(k) : [k/x]K'} \quad \frac{\Gamma \vdash f = f' : (x{:}K)K' \ \ \Gamma \vdash k_1 = k_2 : K}{\Gamma \vdash f(k_1) = f'(k_2) : [k_1/x]K'}$$

$$\frac{\Gamma,x{:}K \vdash k' : K' \ \ \Gamma \vdash k : K}{\Gamma \vdash ([x{:}K]k')(k) = [k/x]k' : [k/x]K'} \quad \frac{\Gamma \vdash f : (x{:}K)K' \ \ x \notin FV(f)}{\Gamma \vdash [x{:}K]f(x) = f : (x{:}K)K'}$$

B Inference Rules of Typed Operational Semantics for LF

The inference rules of the TOS for LF are given below. (See [Gog99, Gog94] for further details.)

Contexts

$$\frac{}{\models () \to ()} \ EMP \quad \frac{\models \Gamma \to \Delta \ \ \Gamma \models A \to B \ \ x \notin dom\{\Gamma\}}{\models \Gamma,x{:}A \to \Delta,x{:}B} \ WEAK$$

Kinds

$$\frac{\Gamma \models ok}{\Gamma \models Type \to Type} \ TYPE \quad \frac{\Gamma \models M \to N \to P : Type}{\Gamma \models El(M) \to El(P)} \ EL$$

$$\frac{\Gamma \models A_1 \to B_1 \ \ \Gamma,x{:}A_1 \models A_2 \to B_2}{\Gamma \models (x{:}A_1)A_2 \to (x{:}B_1).B_2} \ PI$$

Terms

$$\frac{\Gamma_0,x{:}A,\Gamma_1 \models A \to B}{\Gamma_0,x{:}A,\Gamma_1 \models x \to x \to x : B} \ VAR$$

$$\frac{\Gamma \models A_1 \to B_1 \quad \Gamma, x{:}A_1 \models M_0 \to_n P_0 : B_2 \quad [x{:}B_1]P_0 \; not \; \eta - redex}{\Gamma \models [x{:}A_1]M_0 \to [x{:}A_1]M_0 \to [x{:}B_1]P_0 : (x{:}B_1)B_2} \; LAM$$

$$\frac{\Gamma \models A_1 \to B_1 \quad \Gamma, x{:}A_1 \models M_0 \to_n P(x) : B_2 \quad \Gamma \models P \to P \to P : (x{:}B_1)B_2}{\Gamma \models [x{:}A_1]M_0 \to [x{:}A_1]M_0 \to P : (x{:}B_1)B_2} \; ETA$$

$$\frac{\begin{array}{c} \Gamma \models M_1 \to N_1 \to P_1 : (x{:}B_1)B_2 \quad \Gamma \models M_2 \to N_2 \to P_2 : B_1 \\ \Gamma \models [M_2/x]B_2 \to C \quad N_1 \; not \; abstraction \end{array}}{\Gamma \models M_1(M_2) \to N_1(M_2) \to P_1(P_2) : C} \; BASE$$

$$\frac{\begin{array}{c} \Gamma \models M_1 \to_w [x{:}A_1]N_0 : (x{:}B_1)B_2 \quad \Gamma \models M_2 : B_1 \\ \Gamma \models [M_2/x]N_0 \to P \to Q : C \quad \Gamma \models [M_2/x]B_2 \to C \end{array}}{\Gamma \models M_1(M_2) \to P \to Q : C} \; BETA$$