How to Believe a Twelf Proof

Robert Harper Karl Crary

May 9, 2005

1 Introduction

Logical systems are represented in LF by giving a full and faithful (adequate) embedding of the deductive apparatus of the logic as canonical forms of certain types and kinds in LF in specified contexts. The collection of contexts over which the representation is adequate is called a *world*, because it provides generators for the canonical forms in question. Transferring adequacy from one world to another relies on the concept of *subordination*, which expresses the irrelevance of any "extra" variables in the target world.

Given such a representation any metatheoretic property of the logic may be stated and proved in terms of its representation as certain canonical forms. The Twelf meta-theorem prover[1] for LF supports mechanical verification of Π_2 ($\forall\exists$) sentences that quantify over canonical forms of specified types. The proof of such a Π_2 proposition may be seen as a totality proof for an associated relation, with the universal variables designated as "inputs" and the existential variables designated as "outputs". Such totality proofs often take the form of a lexicographic induction over the structure of various canonical forms.

The representation of a logic typically involves higher types of LF; this is called *higher-order abstract syntax*. Exploiting the LF type structure typically streamlines the presentation, but does not in any way inhibit the expressive power of the Twelf theorem prover. This is because the theorem prover works at the meta-level of LF by induction over the structure of canonical forms of higher type.

It is frequently alleged that logical relations arguments cannot be formalized in LF. This is not so; LF is capable of encoding nearly any logic in which a logic relations argument might be expressed. What is true, however, is that the Twelf meta-logic, being limited to Π_2 sentences, cannot express *meta-theorems* whose proof proceeds by logical relations. The reason is that the core idea of logical relations is that the logical complexity of the theorem varies in proportion to

[1] By "Twelf meta-theorem prover" we refer to Twelf's system for checking the totality of meta-theorems. Often it is referred to as Twelf's "totality checker," but we refer to it as a theorem prover in recognition of the fact that it is actually proving an interesting meta-logical theorem (the totality of a higher-order logic program) in its own right. We also refer to it in this manner in recognition that, like all automated theorem provers, there are limits to its abilities. We are *not* referring to Schürmann's automated system for meta-proof construction.

the complexity of the type — the property at higher types is of correspondingly higher quantifier complexity. Since the Twelf meta-logic is restricted to Π_2 sentences, it cannot express such arguments. But there is no obstacle of any kind to using LF to formalize a logical relations proof that is constructed by hand and then to use Twelf to check it.

2 Adequate Representations

Any means of mechanizing the metatheory of a logical system $\mathcal{L}$ using a meta-logical system $\mathcal{M}$ is based on some notion of *representation* of $\mathcal{L}$ within $\mathcal{M}$. The deductive apparatus of $\mathcal{L}$ must be encoded within that of $\mathcal{M}$ in such a way that all reasoning about $\mathcal{L}$ can be conducted by reasoning about its encoding within $\mathcal{M}$. For this to be possible the representation must be *full and faithful*, or *adequate*, in the sense that it must be possible to isolate all and only those constructs in $\mathcal{M}$ that represent constructs in $\mathcal{L}$ — to borrow an old catch-phrase, the representation must permit "no junk" and "no confusion".

A logical system $\mathcal{L}$ is represented in the LF λ-calculus by defining a *compositional bijection* between the objects of each syntactic class of $\mathcal{L}$ and the *canonical forms* of a corresponding canonical type in $\mathcal{LF}$. The canonical forms are the "long" $\beta\eta$-normal forms of an LF type, which we regard as canonical representatives of $\alpha\beta\eta$-equivalence classes. A compositional bijection is one that commutes with substitution, whenever it is meaningful to ask that it do so — namely, whenever there is a notion of "substitutable variable" amongst the objects in a given syntactic class of $\mathcal{L}$. Put in other terms, an adequate representation must take account not just of theorems ("closed" objects), but also consequence and generality ("open" objects).

The canonical forms of a type are determined by the constants and variables that may occur within them. The constants are determined by a *signature* declaring their types and kinds; the variables are determined by a *context* declaring their types. The following judgements are inductively defined:

1. the canonical Kinds: $\Gamma \vdash_\Sigma K \downarrow$;
2. the canonical families of a canonical kind: $\Gamma \vdash_\Sigma A \downarrow K$;
3. the canonical elements of a canonical type: $\Gamma \vdash_\Sigma M \downarrow A$.

The definition of these judgements gives rise to the principal of structural induction over canonical forms. This induction principle is often extended lexicographically to tuples of canonical forms. Such reasoning amounts to transfinite induction up to the ordinal ω^ω.

We may now make precise the methodology of representation of logical systems in $\mathcal{LF}$. The representation of a logical system $\mathcal{L}$ in $\mathcal{LF}$ consists of a signature Σ together with an encoding of the classes of objects of $\mathcal{L}$ as canonical types and the elements of these classes as canonical terms of these types. But to make this precise, we must specify the *contexts* over which the correspondences

are specified so as to take account of open, as well as closed, objects. Thus, to encode a logical system $\mathcal{L}$, we specify

1. A signature, Σ, declaring primitive families of types and elements thereof;
2. For each class of objects of $\mathcal{L}$,
   (a) a *world*, $\mathcal{W}$, a class of contexts over Σ specifying the degree of generality of the representation of that class of objects;
   (b) for each context $\Gamma \in \mathcal{W}$,
      i. a canonical type $\Gamma \vdash_\Sigma A \downarrow \mathsf{type}$ representing the class of objects over the context Γ;
      ii. a compositional bijection between the canonical forms $\Gamma \vdash_\Sigma M \downarrow A$ and the objects of the class represented by the canonical type $\Gamma \vdash_\Sigma A \downarrow \mathsf{type}$.

As an example, let us consider the representation of the simply typed λ-calculus and its associated β-reduction relation. Rather than specify base types, we will instead consider types with free type variables. Here is the signature specifying this system:[2]

```
% syntax of types
tp        : type.
o         : tp.
arr       : tp -> tp -> tp.
% syntax of terms
tm        : type.
lam       : tp -> (tm -> tm) -> tm.
app       : tm -> tm -> tm.
% typing judgement
of        : tm -> tp -> type.
of_lam    : ({x:tm} of x T1 -> of (F x) T2) ->
            of (lam T1 F) (arr T1 T2).
of_app    : of E1 (arr T2 T) -> of E2 T2 -> of (app E1 E2) T.
% reduction judgement
red       : tm -> tm -> type.
red_beta  : red (app (lam T F) E) (F E).
red_lam   : ({x:tm} red (F x) (F' x)) ->
            red (lam T F) (lam T F').
red_app_1 : red E1 E1' -> red (app E1 E2) (app E1' E2).
red_app_2 : red E2 E2' -> red (app E1 E2) (app E1 E2').
```

Here are the adequacy conditions for the representation based on this signature:

[2] We'll use Twelf syntax for this.

Class	*Type*	*World*
types	$\mathtt{tp}$	*empty*
terms	$\mathtt{tm}$	$x_1{:}\mathtt{tm}, \ldots, x_n{:}\mathtt{tm}$
typ. deriv.'s	$\mathtt{of}\, E\, T$	$x_1{:}\mathtt{tm}, d_1{:}\mathtt{of}\, x_1\, T_1, \ldots, x_n{:}\mathtt{tm}, d_n{:}\mathtt{of}\, x_n\, T_n$
red. deriv.'s	$\mathtt{red}\, E_1\, E_2$	$x_1{:}\mathtt{tm}, \ldots, x_n{:}\mathtt{tm}$

Notice that the worlds consists of *blocks* of declarations that follow a pattern that may be easily specified using a regular expression. For example, in the case of typing derivations, variables of type `tm` are paired up with variables standing for typing derivations of these variables. We have suppressed the definition of the compositional bijection between objects of a class and canonical forms of the type representing that class.

Subordination Since the representation of different syntactic classes is adequate with respect to different worlds, this raises the question of whether the canonical forms of a type in one world, $\mathcal{W}_1$, can be transported to another world $\mathcal{W}_2$. For this question to even make sense we must first of all require that the relevant canonical type(s) in $\mathcal{W}_1$ remain types in $\mathcal{W}_2$. And then we ask whether the canonical forms of any of these types are the same when viewed in world $\mathcal{W}_2$ as they are when viewed in world $\mathcal{W}_1$. A minimum criterion is that every context $\mathcal{W}_2$ be an extension of some context in $\mathcal{W}_1$, so that no canonical forms are lost.

However, this is not enough, because we also need to know that, in the extensions, no canonical forms are gained. For example, consider the type family (judgement) `of` and the worlds:

$$\begin{array}{lcl} \mathcal{W}_1 & = & x_1{:}\mathtt{tm}, d_1{:}\mathtt{of}\, x_1\, T_1, \ldots x_n{:}\mathtt{tm}, d_n{:}\mathtt{of}\, x_n\, T_n \\ \mathcal{W}_2 & = & x_1{:}\mathtt{tm}, d_1{:}\mathtt{of}\, x_1\, T_1, d_1'{:}\mathtt{of}\, x_1\, T_1', \ldots x_n{:}\mathtt{tm}, d_n{:}\mathtt{of}\, x_n\, T_n, d_n'{:}\mathtt{of}\, x_n\, T_n' \end{array}$$

The two worlds give very different notions of the typing judgement `of`, because the latter provides for two typing assumptions for every term variable. Although every canonical form in $\mathcal{W}_1$ is also a canonical form of $\mathcal{W}_2$, the latter provides many additional and problematic canonical forms.

On the other hand, some extensions are not problematic. For example, consider the type family `tm` and the worlds:

$$\begin{array}{lcl} \mathcal{W}_1 & = & x_1{:}\mathtt{tm}, \ldots x_n{:}\mathtt{tm}, \\ \mathcal{W}_2 & = & x_1{:}\mathtt{tm}, d_1{:}\mathtt{of}\, x_1\, T_1, \ldots x_n{:}\mathtt{tm}, d_n{:}\mathtt{of}\, x_n\, T_n \end{array}$$

Although $\mathtt{W}_2$ adds a number of new assumptions, none of these can contribute to canonical forms of type `tm`. Thus, the two worlds provide exactly the same canonical forms of type `tm`.

In the former example, the extra assumptions could contribute to the type in question (`of`), and in the latter they could not. This notion is generalized as a relation, called *subordination*, between type families. We say that a type family a is subordinate to a type family b (written $a \preceq b$), if a canonical form belonging to family a can appear within a canonical form of family b. In the former example,

$\texttt{of} \preceq \texttt{of}$, so the additional assumptions are relevant (and problematic); but in the latter, $\texttt{of} \not\preceq \texttt{tm}$, so the additional assumptions are irrelevant.

In general, two worlds $\mathcal{W}_1$ and $\mathcal{W}_2$ are equivalent for the type family a when for every $\Gamma_1 \in \mathcal{W}_1$, there exists $\Gamma_2 \in \mathcal{W}_2$ and an intermediate context Γ, such that Γ is obtainable from both Γ_1 and Γ_2 by dropping irrelevant assumptions (that is, assumptions of types not subordinate to a), and vice versa. Note that world equivalence depends on the type family; worlds equivalent for one family may not be equivalent for another.

3 Informal Metatheory

Given an adequate encoding of a logical system, we may use it to do metatheory for that system by reasoning about canonical forms in certain worlds. An example is the well-known subject reduction theorem for simply typed λ-calculus.

Theorem 1 (Subject Reduction)
For every context $\Gamma = \Gamma_1, \ldots, \Gamma_n$ such that each Γ_i $(1 \leq i \leq n)$ has the form $x_i{:}\texttt{tm}, d_i{:}\texttt{of}\, x_i\, T_i$ for some $\bullet \vdash_\Sigma T_i \downarrow \texttt{tp}$,

if $\Gamma \vdash_\Sigma E_1 \downarrow \texttt{tm}$ and $\Gamma \vdash_\Sigma E_2 \downarrow \texttt{tm}$ and $\bullet \vdash_\Sigma T \downarrow \texttt{tp}$, then

for every $\Gamma \vdash_\Sigma U \downarrow \texttt{red}\, E_1\, E_2$ and $\Gamma \vdash_\Sigma V_1 \downarrow \texttt{of}\, E_1\, T$,

there exists $\Gamma \vdash_\Sigma V_2 \downarrow \texttt{of}\, E_2\, T$.

Proof: The proof proceeds by induction on the structure of the canonical form U. We consider one case here to illustrate what is involved. Suppose that $U = \texttt{red\_lam}\,([x{:}\texttt{tm}]\, U'[x])$. By inversion, $E_1 = \texttt{lam}\,([x{:}\texttt{tm}]\, F_1[x])$ and $E_2 = \texttt{lam}\,([x{:}\texttt{tm}]\, F_2[x])$ and

$$\Gamma, x{:}\texttt{exp} \vdash_\Sigma U'[x] \downarrow \texttt{red}\, F_1[x]\, F_2[x].$$

By inversion, $T = \texttt{arr}\, T_1\, T_2$, $V_1 = \texttt{of\_lam}\,([x{:}\texttt{tm}]\,[d{:}\texttt{of}\, x\, T_1]\, V_1'[x,d])$, and then

$$\Gamma, x{:}\texttt{tm}, d{:}\texttt{of}\, x\, T_1 \vdash_\Sigma V_1'[x,d] \downarrow \texttt{of}\, F_1[x]\, T_2.$$

By weakening, $\Gamma, x{:}\texttt{exp}, d{:}\texttt{of}\, x\, T_1 \vdash_\Sigma U'[x] \downarrow \texttt{red}\, F_1[x]\, F_2[x]$. Observe that $\bullet \vdash_\Sigma T_1 \downarrow \texttt{tp}$. Therefore $\Gamma, x{:}\texttt{tm}\, d{:}\texttt{of}\, x\, T_1$ is a suitable context for the inductive hypothesis. Therefore, by induction,for som

$$\Gamma, x{:}\texttt{tm}, d{:}\texttt{of}\, x\, T_1 \vdash_\Sigma V_2'[x,d] \downarrow \texttt{of}\, F_2[x]\, T_2$$

for some $V_2'[x,d]$. Therefore,

$$\Gamma \vdash_\Sigma [x{:}\texttt{tm}]\,[d{:}\texttt{of}\, x\, T_1].V_2'[x,d] \downarrow \{x{:}\texttt{tm}\}\, \texttt{of}\, x\, T_1 \rightarrow \texttt{of}\, F_2[x]\, T_2.$$

Hence,

$$\Gamma \vdash_\Sigma \texttt{of\_lam}\,([x{:}\texttt{tm}]\,[d{:}\texttt{of}\, x\, T_1].V_2'[x,d]) \downarrow \texttt{of}\,(\texttt{lam}\,([x{:}\texttt{tm}]F_2[x]))\,(\texttt{arr}\, T_1\, T_2).$$

Finally, note that $\texttt{lam}\,([x{:}\texttt{tm}]F_2[x]) = E_2$ and that $\texttt{arr}\,T_1\,T_2 = T$. So let V_2 be $\texttt{of\_lam}\,([x{:}\texttt{tm}]\,[d{:}\texttt{of}\,x\,T_1].V_2'[x,d])$. ■

It is important to note that the inductive hypothesis is conditional on a context of the appropriate world. In the proof we must ensure that any context used in the induction satisfies the world constraint appropriate to the theorem.

4 Formal Metatheory

The subject reduction theorem fits into the Π_2 fragment of the informal metalogic used in its proof, and is therefore amenable to formalization in Twelf. This is achieved by introducing a judgement form `sr` that relates canonical forms representing reductions and typings in the manner described by the theorem. We then designate the *mode* of each argument position in this relation, marking universally quantified positions as "input" (notated perversely, for historical reasons, by a "+") and the existentially quantified positions as "output" (notated by a "−"). Finally, we ask that Twelf check the well-modedness and totality of the so-designated relation `sr` using a specified pattern of induction. If it succeeds, as it does here, the theorem is proved.

Here is an excerpt of the declarations required to carry out this process in Twelf:

```
sr : red E1 E2 -> of E1 T -> of E2 T -> type.
%mode sr +D1 +D2 -D.

sr_lam : sr (red_lam U') (of_lam V1') (of_lam V2')
           <- ({x:tm} {d:of x T1}
                 sr (U' x) (V1' x d) (V2' x d)).

... other cases ...

%block bind : some {t:tp} block {x:tm} {d:of x t}.
%worlds (bind) (sr _ _ _).
%total D (sr D _ _).
```

It is important to note that the *statement* of the subject reduction theorem, when formalized in Twelf, consists of the relation `sr`, the specification of its mode, and the specification of the applicable world. The *proof* of the subject reduction theorem consists of the clauses defining the derivable instances of the `sr` relation, together with an indication of the termination ordering used by the totality checker. Thus, we are *not* treating meta-propositions as (LF) types, nor proofs of meta-propositions as (LF) terms![3]

[3]It should be possible, however, to regard the meta-logic underlying Twelf as a type system based on a suitably strong theory of inductive definitions to support reasoning by lexicographic induction over tuples of canonical forms of various LF types and kinds. It seems plausible that CiC would be sufficient, but so may also some weaker systems of inductive reasoning.

The `%block` directive indicates that the `sr` theorem should be checked to be total in the world consisting of contexts made up of `bind` blocks (let us call this the world $\mathcal{W}_{\texttt{bind}}$). However, this does not mean that `sr` may be used only in $\mathcal{W}_{\texttt{bind}}$. In principle, it may be used in any world in which it may be determined that `sr` is still total. If $\mathcal{W}$ is such a world, we say that $\mathcal{W}_{\texttt{bind}}$ subsumes $\mathcal{W}$ for `sr`.

World subsumption is related to world equivalence for adequacy (discussed earlier), but is not identical. In principle[4] we say that the world $\mathcal{W}'$ subsumes the world $\mathcal{W}$ for the meta-theorem a if for every context $\Gamma \in \mathcal{W}$, there exists $\Gamma' \in \mathcal{W}'$, such that Γ' is obtainable from Γ by dropping irrelevant assumptions (that is, assumptions of types not subordinate to a).

This notion of world subsumption puts an upper limit on the portability of `sr`. We may use `sr` in contexts outside of $\mathcal{W}_{\texttt{bind}}$, provided that the context changes cannot invalidate Twelf's argument for the totality of `sr`. This can happen when any assumptions expected by `sr` are dropped, and can also happen when any relevant assumptions unexpected by `sr` are added.

This example illustrates the general pattern of using the Twelf meta-theorem prover to check the proofs of Π_2 sentences over canonical forms. Many meta-theorems naturally fall into (or can be made to fall into) this class. Consequently, we may use Twelf to check formally many such proofs.

However, since the Twelf meta-prover is limited to Π_2 sentences, it is not capable of verifying meta-theoretic arguments that make use of the method of logical relations (also known as the "computability method", "Tait's method", and, in its most potent form, "Girard's method"). The fundamental reason for this is that the power of logical relations stems from associating a distinct proposition to each type in such a way that as the type increases so does the quantifier complexity of the proposition. This gives us appropriate additional leverage at higher types sufficient to push through an induction on types to establish the theorem. However, since the Twelf meta-prover is resolutely "stuck" at the Π_2 level of the quantifier alternation hierarchy, there is no chance of using it to verify such an argument.

This does not mean that logical relations arguments cannot be formalized in Twelf! We may, if we wish, use Twelf to formalize the ambient meta-theory of a logical relations argument (a fragment of set theory or higher-order logic, for example), and to use Twelf to check every detail of the proof. Such a verification must proceed entirely "by hand" in the sense that we must fully formalize the meta-logical system and fully formalize the logical relations argument within it in order to check the proof.

Finally, the limitation to transfinite induction up to ω^ω is occasionally an obstacle in practice. It would be useful to enhance the expressive power of the Twelf meta-logic to permit more complex forms of induction.

[4] At present, Twelf's implementation of world subsumption is somewhere more limited than this.