Hybrid Extensions in a Logical Framework

Taus Brock-Nannestad Nicolas Guenot Agata Murawska Carsten Schürmann
IT University of Copenhagen, Denmark
{tbro,ngue,agmu,carsten}@itu.dk

ABSTRACT

We discuss the extension of the standard **LF** logical framework with operators for manipulating *worlds*, as found in hybrid logics or in the **HLF** framework. To overcome the restrictions of **HLF**, we present a more general approach to worlds in **LF**, where the structure of worlds can be described in an explicit way. We give a canonical presentation of the system and discuss the encoding of logical systems, beyond the limited scope of linear logic that formed the main goal of **HLF**.

1. HYBRID LOGICS AND LF

The **LF** logical framework [HHP93] has been successfully used to represent adequately many different logics and systems, and it greatly simplifies their encoding by providing a representation language where the object-level is based on the λ-calculus. This offers the possibility to use *higher-order abstract syntax* as well as hypothetical judgements, where the usual notions of abstraction and substitution are primitives.

There are however systems that cannot be encoded adequately in **LF** without a heavy manipulation of structures that must be dealt with manually both when defining the encoding and when reasoning about the system. One such example can be obtained by extending a standard logic, such as intuitionistic logic, with *hybrid* operations as suggested by Prior [Pri67] and introduced later in standard proof theoretical systems — some proof theory for hybrid logics can be found for example in [Tza99], [ABM01] and [GS11]. The idea of hybrid logics is simply to make explicit the Kripke semantics usually given to logics, in particular modal logics, by allowing inference rules to manipulate the *worlds* from this semantics. This yields elegant proof systems for logics with connectives that perform complex operations on these worlds. For example, one can define a natural deduction system for the intuitionistic form of modal logics [Sim94] where rules for $\Box$ are:

$$\Box_I \frac{\Gamma, xRy \vdash A[y]}{\Gamma \vdash \Box A[x]} \qquad \Box_E \frac{\Gamma \vdash \Box A[x] \quad (xRy)}{\Gamma \vdash A[y]}$$

where $A[x]$ indicates that A is provable at a particular world x, while an assumption of the shape xRy in the context is a witness of the condition that for this rule to hold, y must be reachable from x in the relation R of the associated Kripke semantics. The particular properties of such a modal logic then depend on the axioms concerning this relation used in the semantics, and for example a reflexive and transitive relation yields **IS4**.

The problem with the encoding of such a system in **LF** is that worlds and assumptions of the shape xRy must be encoded and manipulated manually, so that each time a property of R needs to be used, the same procedure is applied. What is lacking in **LF** is support for such structures and manipulations inside the syntax to, for example, automatically deal with reflexivity, transitivity or other properties of the relation. Such infrastructure has been developed, with the specific purpose of encoding linearity, in the **HLF** framework [Ree09] that extends **LF** with some support for hybrid operations. In this setting, types and terms can use worlds that are not always variables but can also be *compound* worlds built with the binary $*$ operator and its unit ε. This structure of worlds has been used to encode linear implication in **HLF** at the level of the representation language: one can reason *linearly* in **HLF** in the sense that $\multimap$ is available as a type, simply defined as a macro from primitive operations on worlds.

If we consider the naive encoding of a modal logic in **LF**, we need to explicitly manipulate the worlds and define the constants corresponding to rules of the congruence:

$$\begin{array}{lll}
\mathtt{o} : \mathtt{type} & \mathtt{pf} : \mathtt{o} \to \mathtt{w} \to \mathtt{type} & \supset : \mathtt{o} \to \mathtt{o} \to \mathtt{o} \\
\mathtt{w} : \mathtt{type} & \mathtt{rc} : \mathtt{w} \to \mathtt{w} \to \mathtt{type} & \Box : \mathtt{o} \to \mathtt{o} \\
 & & \Diamond : \mathtt{o} \to \mathtt{o}
\end{array}$$

$$\begin{array}{rl}
\mathtt{refl} : & \{\alpha : \mathtt{w}\}\, \mathtt{rc}\ \alpha\ \alpha \\
\mathtt{trans} : & \{\alpha, \gamma, \sigma : \mathtt{w}\}\, \mathtt{rc}\ \alpha\, \gamma \to \mathtt{rc}\ \gamma\ \sigma \to \mathtt{rc}\ \alpha\ \sigma
\end{array}$$

$$\begin{array}{rl}
\Box_I & : \{A : \mathtt{o}\}\{\alpha : \mathtt{w}\}(\{\gamma : \mathtt{w}\}\, \mathtt{rc}\, \alpha\, \gamma \to \mathtt{pf}\, A\, \gamma) \to \mathtt{pf}\ \Box A\ \alpha \\
\Box_E & : \{A : \mathtt{o}\}\{\alpha, \gamma : \mathtt{w}\}\, \mathtt{pf}\ \Box A\ \alpha \to \mathtt{rc}\ \alpha\, \gamma \to \mathtt{pf}\, A\, \gamma
\end{array}$$

but this encoding is not adequate in **LF** because of the many ways of making two worlds equivalent. The situation could not really be improved in **HLF**, since the relation on worlds that is needed here does not fit the syntax of $*$ and ε under AC-unification. The work we present here stems from an attempt to represent and reason about a hypersequent calculus for a variant of linear logic [Mon13], that fails in **HLF** because of the structure of worlds: it is used to ensure linearity and cannot be used to also represent the connections between sequents in a hypersequent. Similarly, encoding modal logics in **HLF** would be uneasy since the relation on worlds is incompatible with the notion of equality on worlds in this framework.

The goal of the work presented here is therefore to define a more general extension of **LF** that allows a more extensively use

LFMTP'14, Vienna, Austria

of the expressivity of hybrid operators. To do this, we follow the standard presentation of **LF** in its canonical form [HL07] and add ingredients from **HLF**, and more, to support the encoding of advanced hybrid systems. The key to do this is the generalisation of the structure of worlds, from a fixed set of operations $\{*, \varepsilon\}$ to an abstract notion combining any number of operators and an equivalence relation on worlds. The resulting framework is then parametric in the definition given for worlds.

We start in Section 2 by describing our new hybrid framework, called **HyLF**, and discuss reduction, normal forms and notions of substitution in this setting. Then, in Section 3, we illustrate the use of the system by considering an encoding of modal logics exploiting an advanced structure of worlds.

2. EXTENDING HYBRID LF

Instead of starting from **HLF** and enriching the system, we go back to the standard framework of **LF** [HHP93], in its form based on canonical typing derivations [HL07]. In particular, we use the standard λ-calculus as base, without *spines* [CP03], to keep the theory as simple as possible. The languages of terms and types of our **HyLF** framework extend canonical **LF** to support various user-defined operators on worlds.

In the following, we denote by letters such as x, y and z *term variables*, by t, u and v *canonical terms*, by r and s *atomic terms*, by A, B and C *canonical types*, by F and G *atomic types*, and by K or L kinds of **HyLF**. Moreover, we use Greek letters such as α or γ for *world variables* and p or q for *worlds* in general. Terms, types and kinds are defined by the following grammar:

$$\begin{array}{rcl} K,L & ::= & \mathtt{type} \mid \Pi x:A.K \mid \forall\alpha.K \\ A,B & ::= & F \mid \Pi x:A.B \mid \forall\alpha.A \mid A@p \mid \downarrow\alpha.A \\ F,G & ::= & a \mid F\ t \mid F\ \{p\} \\ t,u & ::= & r \mid \lambda x.t \mid \lambda\{\alpha\}.t \mid t\ \mathtt{at}\ p \mid \mathtt{here}\ \alpha.t \\ r,s & ::= & x \mid c \mid r\ t \mid r\ \{p\} \mid r\ \mathtt{to}\ p \mid \mathtt{ccw}\ r \end{array} \tag{1}$$

This system is similar to **HLF** [Ree09], with more primitives at the level of terms reflecting elimination rules for world operators into the object language. For the sake of simplicity, no cartesian product is used in **HyLF**, and we collapse dependent product and universal quantification when it comes to worlds.

The generalisation of **HyLF** with respect to **HLF** lies in the way worlds can be defined: instead of defining one fixed structure of worlds with the operator $*$ and its unit ε, we will make the whole framework parametric in the definition of worlds. The first step in the instantiation of the framework is to define the language of worlds, which is always of the shape:

$$p,q \ ::= \ \alpha \mid o(\vec{p}) \qquad \text{where } o:|\vec{p}| \in \mathscr{O}$$

where o is any operator defined in the *operators signature* $\mathscr{O}$ that contains entries of the form $o : k$ indicating that o is an operator of arity k, and $\vec{p}$ is a sequence of worlds of length $|\vec{p}|$. The second step of the instantiation is to define the *equivalence* relation $\equiv$ over worlds, which must be a congruence for the operators in $\mathscr{O}$, by specifying additional equations.

DEFINITION 2.1. *An instance **HyLF**$(\mathscr{O}, \equiv)$ of the parametric **HyLF** framework is defined by providing an operators signature $\mathscr{O}$ and a congruence $\equiv$ over worlds.*

In the following, we write **HyLF** when describing properties of any particular instance, and specify the exact operators signature and congruence used only when necessary. The typing rules for the canonical term level of **HyLF** is shown in Figure 1. Binding a world can be done in this system through a λ-abstraction, but also with $\mathtt{here}\ \alpha.t$, a construct binding the *current* world.

$$\Pi\boldsymbol{i}\ \frac{\Omega;\Gamma,x:A[p]\vdash t \Leftarrow B[p]}{\Omega;\Gamma\vdash \lambda x.t \Leftarrow \Pi x:A.B\,[p]}$$

$$\downarrow\boldsymbol{i}\ \frac{\Omega;\Gamma\vdash t\{p/\alpha\} \Leftarrow A\{p/\alpha\}\,[p]}{\Omega;\Gamma\vdash \mathtt{here}\ \alpha.t \Leftarrow \downarrow\alpha.A\,[p]}$$

$$@\boldsymbol{i}\ \frac{\Omega;\Gamma\vdash t \Leftarrow A[q] \quad p\in\mathscr{W}}{\Omega;\Gamma\vdash t\ \mathtt{at}\ q \Leftarrow A@q\,[p]} \qquad \forall\boldsymbol{i}\ \frac{\Omega,\alpha;\Gamma\vdash t \Leftarrow A[p] \quad \alpha\notin\Omega}{\Omega;\Gamma\vdash \lambda\{\alpha\}.t \Leftarrow \forall\alpha.A[p]}$$

$$\boldsymbol{s}\ \frac{\Omega;\Gamma\vdash t \Rightarrow F\,[q] \quad p\equiv q \quad p\in\mathscr{W}}{\Omega;\Gamma\vdash t \Leftarrow F\,[p]}$$

$$\Pi\boldsymbol{e}\ \frac{\Omega;\Gamma\vdash r \Rightarrow \Pi x:A.B\,[p] \qquad \Omega;\Gamma\vdash t \Leftarrow A[p]}{\Omega;\Gamma\vdash r\ t \Rightarrow B\{t/x\}\,[p]}$$

$$\downarrow\boldsymbol{e}\ \frac{\Omega;\Gamma\vdash r \Rightarrow \downarrow\alpha.A[p]}{\Omega;\Gamma\vdash \mathtt{ccw}\ r \Rightarrow A\{p/\alpha\}\,[p]}$$

$$\boldsymbol{x}\ \frac{x:A[p]\in\Gamma \quad p\in\mathscr{W}}{\Omega;\Gamma\vdash x \Rightarrow A[p]} \qquad @\boldsymbol{e}\ \frac{\Omega;\Gamma\vdash r \Rightarrow A@p\,[q] \quad p\in\mathscr{W}}{\Omega;\Gamma\vdash r\ \mathtt{to}\ p \Rightarrow A[p]}$$

$$\boldsymbol{c}\ \frac{c:A[p]\in sig \quad p\in\mathscr{W}}{\Omega;\Gamma\vdash c \Rightarrow A[p]} \qquad \forall\boldsymbol{e}\ \frac{\Omega;\Gamma\vdash r \Rightarrow \forall\alpha.A[q] \quad p\in\mathscr{W}}{\Omega;\Gamma\vdash r\ \{p\} \Rightarrow A\{p/\alpha\}\,[q]}$$

Figure 1: Typing rules for HyLF terms

EXAMPLE 2.1. *The structure of worlds used in **HLF** is obtained in **HyLF** by an instantiation where the language of worlds is defined by the signature $\{* : 2, \varepsilon : 0\}$, corresponding to the grammar:*

$$p,q \ ::= \ \alpha \mid p*q \mid \varepsilon$$

and where the congruence over worlds is defined by the rules:

$$\frac{}{(p*q)*p' \equiv p*(q*p')} \qquad \frac{}{p*q \equiv q*p} \qquad \frac{}{p*\varepsilon \equiv p}$$

$$\frac{}{p\equiv p} \qquad \frac{q\equiv p}{p\equiv q} \qquad \frac{p\equiv q \qquad q\equiv p'}{p\equiv p'} \qquad \frac{p\equiv p' \qquad q\equiv q'}{p*q\equiv p'*q'}$$

which are implementing AC-unification. ∎

In the rules shown in Figure 1, we use two kinds of judgements to indicate whether a type is *synthetised* from the term or *checked* against the term, always at some world p, which are denoted by $\Omega;\Gamma\vdash t \Rightarrow A[p]$ and $\Omega;\Gamma\vdash t \Leftarrow A[p]$ respectively. In both cases, Ω denotes a set of world variables and Γ is a list of assumptions of the shape $x : A[p]$. This means that assumptions are assigned a world, as often done in sequent calculi for modal logics [GS11], but not done in **HLF**. The two judgements are meant to enforce a separation between canonical and atomic terms, so that all terms typed are canonical. Moreover, in these rules:

- the condition $p\in\mathscr{W}$ ensures that p appears in the set $\mathscr{W}$ of worlds *well-formed* according to the signature $\mathscr{O}$,
- the list *sig* is a *constants signature*, implicitly associated to the judgement — we could write $\vdash_{sig}$ but omit this for the sake of readability,

$$\boldsymbol{a}\,\frac{a : K[p] \in sig \quad p \in \mathscr{W}}{\Omega; \Gamma \vdash a \Rightarrow K[p]} \qquad \forall\boldsymbol{f}\,\frac{\Omega; \Gamma \vdash F \Rightarrow \forall \alpha.K[q] \quad p \in \mathscr{W}}{\Omega; \Gamma \vdash F\ \{p\} \Rightarrow K\{p/\alpha\}[q]}$$

$$\Pi\boldsymbol{f}\,\frac{\Omega; \Gamma \vdash F \Rightarrow \Pi x : A.K[p] \quad \Omega; \Gamma \vdash t \Leftarrow A[p]}{\Omega; \Gamma \vdash F\ t \Rightarrow K\{t/x\}[p]}$$

$$\boldsymbol{f}\,\frac{\Omega; \Gamma \vdash F \Rightarrow \mathtt{type}[p]}{\Omega; \Gamma \vdash F :: \mathtt{type}[p]} \qquad \forall\,\frac{\Omega, \alpha; \Gamma \vdash A :: \mathtt{type}[p] \quad \alpha \notin \Omega}{\Omega; \Gamma \vdash \forall \alpha.A :: \mathtt{type}[p]}$$

$$\Pi\,\frac{\Omega; \Gamma \vdash A :: \mathtt{type}[p] \quad \Omega; \Gamma, x : A[p] \vdash B :: \mathtt{type}[p]}{\Omega; \Gamma \vdash \Pi x : A.B :: \mathtt{type}[p]}$$

$$\downarrow\,\frac{\Omega; \Gamma \vdash A\{p/\alpha\} :: \mathtt{type}[p]}{\Omega; \Gamma \vdash \mathtt{here}\ \alpha.A :: \mathtt{type}[p]} \qquad @\,\frac{\Omega; \Gamma \vdash A :: \mathtt{type}[q] \quad p \in \mathscr{W}}{\Omega; \Gamma \vdash A@q :: \mathtt{type}[p]}$$

$$\boldsymbol{t}\,\frac{p \in \mathscr{W}}{\Omega; \Gamma \vdash \mathtt{type} :: \mathtt{kind}[p]} \qquad \forall\boldsymbol{k}\,\frac{\Omega, \alpha; \Gamma \vdash K :: \mathtt{kind}[p] \quad \alpha \notin \Omega}{\Omega; \Gamma \vdash \forall \alpha.K :: \mathtt{kind}[p]}$$

$$\Pi\boldsymbol{k}\,\frac{\Omega; \Gamma \vdash A :: \mathtt{type}[p] \quad \Omega; \Gamma, x : A[p] \vdash K :: \mathtt{kind}[p]}{\Omega; \Gamma \vdash \Pi x : A.K :: \mathtt{kind}[p]}$$

Figure 2: Kinding rules for HyLF

- we go from one kind of judgement to the other only in the $\boldsymbol{s}$ rule, which *swaps* from synthesis to checking, and this is also the only rule relying on the congruence $\equiv$,
- in the axioms $\boldsymbol{x}$ and $\boldsymbol{c}$, the context Γ and the signature *sig* should be checked for well-formation, following rules that we omit here but are straightforward,
- the notations $\{t/x\}$ and $\{p/\alpha\}$ correspond to the standard notion of capture-avoiding substitution in a term, of a term for a variable and a world for a variable, respectively.

Finally, we show in Figure 2 the rules for kinding type families in **HyLF**, which are again an extension of the standard rules for **LF**, where abstraction can be performed on worlds and atomic types can also be applied to the worlds. Note that the same kind of conditions are used in these rules as in Figure 1, and contexts and signatures should be checked at axiom rules $\boldsymbol{a}$ and $\boldsymbol{t}$. There are three new judgements in these rules, $\Omega; \Gamma \vdash F \Rightarrow K[p]$ and $\Omega; \Gamma \vdash A :: \mathtt{type}[p]$, and $\Omega; \Gamma \vdash K :: \mathtt{kind}[p]$ which represent the validity of a type A at some world p, having a certain kind K or just type, and the validity of a kind K at p, respectively.

The term level of this system reflects the extension of the type level by offering primitives to manipulate worlds. The meaning of these constructs can be intuitively understood as follows:

- the universal quantification on worlds $\forall \alpha.A$ yields a simple mechanism of abstraction and application, distinguished from the standard λ-calculus constraints by the $\{p\}$ syntax used in both abstraction and application, so that $\lambda\{\alpha\}.t$ is related only to $r\ \{p\}$ and not standard application,
- the world localisation operation $A@p$ yields the operations $t\ \mathtt{at}\ p$ and $r\ \mathtt{to}\ p$ which indicate that some term t must be evaluated at a world p, and that r has been moved to the world p, respectively: this plays a role in the semantics of computation in this setting, where reduction happens at a certain world to reflect the constraints imposed by typing judgements,
- the current world (binding) operation $\downarrow \alpha.A$ is similar to the world quantification but it yields a mechanism for binding the world where the term t will be evaluated through the operation $\mathtt{here}\ \alpha.t$, and associating this name to the world where the term r is currently evaluated, with the operation of *call-current-world* denoted by $\mathtt{ccw}\ r$.

Canonicity. In a logical framework such as **LF**, it is important to be able to isolate *canonical forms*, so that adequacy can later be proven, to correctly relate the structures being encoded and their actual **LF** encodings. This is why the typing rules for **HyLF** are *bidirectional* and restrict the formation of terms to the grammar given in (1). However, we need to have a notion of *reduction* to offer to represent the dynamics of the systems we encode — for example, reductions for cut elimination in logics presented in a sequent calculus. This cannot be done in the canonical system, since reductions correspond to elimination of *detours*, where an introduction rule appears immediately above the corresponding elimination rule in a typing derivation.

In order to recover a system where reductions are possible, we need to bypass the restrictions imposed by the use of $\Rightarrow$ and $\Leftarrow$ annotations. Moving from one kind of judgement to the other is already possible using the swap rule $\boldsymbol{s}$. All we need is therefore a rule $\boldsymbol{s}^{-1}$ opposite to this rule:

$$\boldsymbol{s}^{-1}\,\frac{\Omega; \Gamma \vdash t \Leftarrow A[q] \quad p \equiv q \quad p \in \mathscr{W}}{\Omega; \Gamma \vdash t \Rightarrow A[p]}$$

to be able to type non-canonical forms. Note that this rule applies at any type and not just on atomic types, yielding the possibility to type sequences of introduction and elimination rules. Here, we use a canonical presentation where this rule does not appear, and keep reduction as an *"external"* device. In the following, we call *well-typed* a term t such that for some type A there exists a typing derivation for $\Omega; \cdot \vdash t \Leftarrow A[p]$ in **HyLF** — this implies that t is canonical, since the $\boldsymbol{s}^{-1}$ rule is not used.

Reduction. Allowing non-canonical forms allows us to accept more terms, but we want to reason under an equivalence relation such that any non-canonical term is associated to some canonical term. This relies on a notion of reduction on the non-canonical terms of a grammar where t appears in the category r as shown in (1) — this is obtained with the $\boldsymbol{s}^{-1}$ rule shown above. Since in the hybrid setting all terms are reduced *at a certain world*, the reduction relation $\longrightarrow_p$ must be parameterised by some p where evaluation happens. The main reduction rules are:

$$\begin{array}{rcl} (\lambda x.t)\ u & \longrightarrow_p & t\{u/x\} \\ \mathtt{ccw}\,(\mathtt{here}\ \alpha.t) & \longrightarrow_p & t\{p/\alpha\} \\ (t\ \mathtt{at}\ p)\ \mathtt{to}\ p & \longrightarrow_p & t \\ (\lambda\{\alpha\}.t)\ \{q\} & \longrightarrow_p & t\{q/\alpha\} \end{array} \qquad (2)$$

where the first is simply β-reduction and the others represent the elimination of other detours in **HyLF** typing derivations. But there are more rules needed here, to allow reduction under any construct. These rules are standard in most cases, and we have for example:

$$\begin{array}{rcll} \lambda x.t & \longrightarrow_p & \lambda x.u & \textit{if } t \longrightarrow_p u \\ t\ u & \longrightarrow_p & v\ w & \textit{if } t \longrightarrow_p u \textit{ and } v \longrightarrow_p w \\ \mathtt{ccw}\ t & \longrightarrow_p & \mathtt{ccw}\ u & \textit{if } t \longrightarrow_p u \\ \mathtt{here}\ \alpha.t & \longrightarrow_p & \mathtt{here}\ \alpha.u & \textit{if } t \longrightarrow_p u \end{array}$$

but the reduction rules involving the at and to operators have a specific effect on the world where evaluation happens:

$$\begin{array}{llll} t \,\mathtt{at}\, q & \longrightarrow_p & u \,\mathtt{at}\, q & \text{if } t \longrightarrow_q u \\ t \,\mathtt{to}\, p & \longrightarrow_p & u \,\mathtt{to}\, p & \text{if } t \longrightarrow_q u \quad \text{for some } q \in \mathscr{W} \end{array}$$

corresponding to the meaning of these operations. Indeed, even when $t\,\mathtt{at}\,q$ is evaluated at p, the evaluation of t is performed at world q, and if t is evaluated into u at q, then $t\,\mathtt{to}\,u$ transfers the result of this evaluation to world p — this can be related to the fetch/get operations affecting the current world of evaluation in the modal λ-calculus presented in [MCHP04].

Apart from this use of world in the evaluation of terms, the computational semantics of **HyLF** relies on standard notions. In particular, the key element in reduction is *substitution*. There are two kinds of substitution applied in (2): the usual substitution $\{u/x\}$ of some term u for a term variable x, capture-avoiding and relying on α-conversion for λ-abstractions, and the substitution $\{p/\alpha\}$ of a complex world p for a world variable α. This second form of substitution is defined in a standard way, relying on the α-conversion of world names in the binding operations of world abstraction and current world abstraction. Intuitively, this is the simultaneous replacement of all the free occurrences of α by the world p, in any term. We will not discuss here the properties of the $\longrightarrow_p$ reduction or of its reflexive, transitive closure $\longrightarrow\!\!\!\!\!\rightarrow_p$.

Substitution. The dynamics of non-canonical terms is based on the notion of substitution. In the canonical **HyLF** system, we cannot define the usual notion of substitution because it does not necessarily yield a canonical form. Such a notion can be defined here, and thus preserve canonical forms, only if it is parameterised to an *hereditary* form of substitution, where redexes created by substitution are reduced immediately [WCPW04].

DEFINITION 2.2. *For any well-typed terms t, u, any variable x and a world p, the* hereditary substitution $t[u/x]_p$ *of u for x in t at world p is defined recursively by:*

$$\frac{}{x[u/x]_p = u} \qquad \frac{}{y[u/x]_p = y} \qquad \frac{r[u/x]_p = s \quad t[u/x]_p = v}{(r\ t)[u/x]_p = s\ v}$$

$$\frac{}{c[u/x]_p = c} \qquad \frac{r[u/x]_p = \lambda y.v' \quad t[u/x]_p = t' \quad v'[t'/y]_p = v}{(r\ t)[u/x]_p = v}$$

$$\frac{t[u/x]_p = v}{(\lambda y.t)[u/x]_p = \lambda y.v} \qquad \frac{t[u/x]_p = v}{(\lambda\{\alpha\}.t)[u/x]_p = \lambda\{\alpha\}.v}$$

$$\frac{t[u/x]_q = v}{(t\,\mathtt{at}\,q)[u/x]_p = v\,\mathtt{at}\,q} \qquad \frac{t[u/x]_p = v}{(\mathtt{here}\,\alpha.t)[u/x]_p = \mathtt{here}\,\alpha.v}$$

$$\frac{r[u/x]_q = s}{(r\,\mathtt{to}\,p)[u/x]_p = s\,\mathtt{to}\,p} \qquad \frac{r[u/x]_q = t\,\mathtt{at}\,p}{(r\,\mathtt{to}\,p)[u/x]_p = t}$$

$$\frac{r[u/x]_p = s}{(\mathtt{ccw}\,r)[u/x]_p = \mathtt{ccw}\,s} \qquad \frac{r[u/x]_p = \mathtt{here}\,\alpha.t}{(\mathtt{ccw}\,r)[u/x]_p = t\{p/\alpha\}}$$

$$\frac{r[u/x]_p = s}{(r\ \{q\})[u/x]_p = s\ \{q\}} \qquad \frac{r[u/x]_p = \lambda\{\alpha\}.t}{(r\ \{q\})[u/x]_p = t\{q/\alpha\}}$$

Note that only the case of crossing an application can create new redexes in **LF**, but here there are three more non-trivial cases corresponding to other reduction rules in **HyLF**. However, none of these new cases trigger a term substitution, and substitution of worlds never creates new redexes, so that it does not need to be defined hereditarily to stay in the canonical fragment. Indeed, all redexes in (2) rely on the shape of terms rather than on worlds, except of $(t\,\mathtt{at}\,p)\,\mathtt{to}\,q$, but it can only be well-typed if $p = q$. We can now prove that hereditary substitution is actually a particular implementation of the reductions shown in (2).

THEOREM 2.1 (HEREDITARY SUBSTITUTION). *For any terms t and u, if there exists $v = t[u/x]_p$ for a given world p then we have the reduction $t\{u/x\} \longrightarrow\!\!\!\!\!\rightarrow_p v$.*

PROOF. By structural induction on the tree used to justify the statement $t[u/x]_p = v$, with base cases in the three axioms given above. Most cases are direct calls to the induction hypothesis, as they reflect the propagation of substitution in the term. The main cases are the ones involving a newly created redex:

- if substitution creates an abstraction inside an application, then we have $r[u/x]_p \longrightarrow\!\!\!\!\!\rightarrow_p \lambda y.v'$ and $t'[u/x]_p \longrightarrow\!\!\!\!\!\rightarrow_p t''$ by induction hypothesis, but also $v'[t'/x]_p \longrightarrow\!\!\!\!\!\rightarrow_p v$, so that we can conclude $(r\ t')[u/x]_p \longrightarrow\!\!\!\!\!\rightarrow_p v$,
- if substitution creates an at statement inside a to, we have $r[u/x]_p \longrightarrow\!\!\!\!\!\rightarrow_p v\,\mathtt{at}\,p$ by induction hypothesis and therefore we have $(r\,\mathtt{to}\,p)[u/x]_p \longrightarrow\!\!\!\!\!\rightarrow_p v$,
- if substitution creates a current world binding in a current world call, we have $r[u/x]_p \longrightarrow\!\!\!\!\!\rightarrow_p \mathtt{here}\,\alpha.v$ by induction hypothesis and thus $(\mathtt{ccw}\,r)[u/x]_p \longrightarrow\!\!\!\!\!\rightarrow_p v\{p/\alpha\}$,
- if substitution creates a world abstraction in an application to some world, we have $r[u/x]_p \longrightarrow\!\!\!\!\!\rightarrow_p \lambda\{\alpha\}.v$ by induction hypothesis and thus $(r\ \{q\})[u/x]_p \longrightarrow\!\!\!\!\!\rightarrow_p v\{q/\alpha\}$,

so that in any case reduction produces the resulting term v. □

Note that the correspondence between hereditary substitution and the reduction of non-canonical **HyLF** terms is established only for well-typed terms, which simplifies greatly the situation as it prevents the creation of redexes through world substitution, making world substitution non-hereditary.

The critical property of the notion of hereditary substitution is that it preserves typeability in the canonical **HyLF** system, along with the fact that given well-typed terms t and u, we can perform the substitution of one for a variable in this other. The proof of this uses an induction that is made more complicated by the the case where a new β-redex is created: it must involve the type of the u being substituted. This is however standard, and it only requires to consider a simple approximation of the type of u.

DEFINITION 2.3 (TYPE ERASURE). *For a term u of type A, the* simple type $\tau(u)$ *of u is defined by induction on A as $[\![A]\!]_\tau$, where:*

$$[\![a]\!]_\tau = a \qquad \begin{array}{c} [\![\Pi x : A.B]\!]_\tau = A \to B \\ [\![F\ t]\!]_\tau = [\![F]\!]_\tau\ t \\ [\![F\ \{t\}]\!]_\tau = [\![F]\!]_\tau\ \{p\} \end{array} \qquad \begin{array}{c} [\![\forall\alpha.A]\!]_\tau = A \\ [\![A@p]\!]_\tau = A \\ [\![\downarrow\alpha.A]\!]_\tau = A \end{array}$$

We can now state and prove the main theorem allowing to use the notion of substitution in the canonical presentation of **HyLF**. More details on this result in standard **LF** but also in **HLF** can be found in the literature [HL07, Ree09].

THEOREM 2.2 (SUBSTITUTION). *Given any two terms t and u such that $\Omega;\Gamma,x : A[q],\Delta \vdash t \Leftarrow B[p]$ and $\Omega;\Gamma \vdash u \Leftarrow A[q]$ there exists a term $t[u/x]$ such that $\Omega;\Gamma,\Delta \vdash t[u/x] \Leftarrow B\{u/x\}[p]$.*

PROOF. We proceed by induction on the pair $(\tau(u), |t|)$, under lexicographic ordering, with base cases when t is a term variable or a constant: if it is x then we can simply use the given typing derivation for u, and the two other cases are trivial. Then, in the general case, most of the configurations require only to apply the induction hypothesis and conclude using the right typing rule. In the cases involving the creation of a redex of world constructs, a world substitution might be needed in the derivation obtained by induction hypothesis.

Finally, the complicated case is the one involving the creation of a β-redex, when t is an application: the induction hypothesis can be used for substitution in the function and in its argument since they are both structurally smaller than t, but the induction hypothesis can be used on the hereditarily spawned substitution only because the term being substituted in a term v potentially larger than t has a simple type smaller than $\tau(u)$. □

There is another theorem that allows to perform the same kind of operation when a world is concerned. This corresponds to the observation that if a world variable is used in a typing derivation, this derivation is parametric in this variable, so that consistently replacing this variable with any given world always yields a valid typing derivation.

THEOREM 2.3 (WORLD SUBSTITUTION). *Given a term t such that $\Omega, \alpha; \Gamma \vdash t \Leftarrow B[p]$ and some world q, there is a derivation of the judgement $\Omega; \Gamma\{q/\alpha\} \vdash t\{q/\alpha\} \Leftarrow B\{q/\alpha\}[p\{q/\alpha\}]$.*

PROOF. By induction on the given typing derivation, with base cases when an axiom rule ***x*** or ***c*** is used, where the substitution can be performed provided q is a valid world — the substitution was also applied to the signature implicitly present in $\vdash$. In the general case, we can always use the induction hypothesis and apply the right typing rule. Note that rules for world constructs can still be applied after substitution because the substitution is performed on every part of the judgement. □

We will not go into further details about the properties of terms and derivations forming the metatheory of the **HyLF** framework, but rather present examples of how extending **LF** with hybrid constructs allows us to elegantly represent logics that are defined by a hybrid system

3. ENCODING LOGICS IN HyLF

We will consider here two ways of using the hybrid operations to encode logics and systems. The first approach is the standard encoding idea of **LF**, where the given system is defined with rules represented by typed constants added to the signature, such that adequacy can be proven between the system as seen *"on paper"* and its **LF** representation. The second approach is to encode any given logical connective into the type level of **LF**, and prove that the typing rules from **LF** correspond to the rules intended for this connective, so that the form of reasoning embodied by this connective is made available to encode further systems following the standard encoding approach.

Note that the first approach, in the **HyLF** framework, requires not only to define typed constants representing the rules of the system, but also the definition of operators and a congruence on worlds to instantiate the parametric framework. However, this is a trade-off, where the added specification of the level of worlds makes the representation of the rules simpler.

Intuitionistic modal logics. The most natural system that we can encode using the hybrid operations of **HyLF** is the natural deduction calculus for various intuitionistic modal logics defined by Simpson [Sim94], using the rules shown in Figure 3. In this system, the relation R on worlds defining the particular flavour

$$\textbf{\textit{ax}}\,\frac{A[x] \in \Gamma}{\Omega; \Sigma; \Gamma \vdash A[x]} \qquad \to\!\textbf{\textit{i}}\,\frac{\Omega; \Sigma; \Gamma, A[x] \vdash B[x]}{\Omega; \Sigma; \Gamma \vdash A \to B[x]}$$

$$\to\!\textbf{\textit{e}}\,\frac{\Omega; \Sigma; \Gamma \vdash A[x] \qquad \Omega; \Sigma; \Gamma \vdash A \to B[x]}{\Omega; \Sigma; \Gamma \vdash B[x]}$$

$$\Box\textbf{\textit{i}}\,\frac{\Omega, y; \Sigma, xRy; \Gamma \vdash A[y]}{\Omega; \Sigma; \Gamma \vdash \Box A[x]} \qquad \Box\textbf{\textit{e}}\,\frac{\Omega; \Sigma; \Gamma \vdash \Box A[x]}{\Omega; \Sigma, xRy; \Gamma \vdash A[y]}$$

$$\Diamond\textbf{\textit{i}}\,\frac{\Omega, y; \Sigma; \Gamma \vdash A[y]}{\Omega; \Sigma, xRy; \Gamma \vdash \Diamond A[x]}$$

$$\Diamond\textbf{\textit{e}}\,\frac{\Omega; \Sigma; \Gamma \vdash \Diamond A[x] \qquad \Omega; \Sigma, xRy; \Gamma, A[y] \vdash B[z]}{\Omega; \Sigma; \Gamma \vdash B[z]}$$

Figure 3: Inference rules for the basic logic IK

of modal logic used is mentioned explicitly, so that the same rules properly represent many different modal logics, such as **IK**, **IS4** or **IS5**. This system is well-suited for a presentation in **HyLF**, as we will be able to define the rules as constants and simply change the definition of the congruence on worlds to switch between different logics — by specifying exactly the axioms defining the Kripke semantics of these logics.

The presentation of the **IK** system here is made slightly more precise than the one given by Simpson, on the syntactic level: we use the sequent notation and distinguish between three parts of a context, denoted by Ω, Σ and Γ, to hold available world names, assumptions on R and logical assumptions, respectively. In this system, worlds are always just names such as x, y or z. A sequent is written $\Omega; \Sigma; \Gamma \vdash A[x]$ for provability of A at a world x under these three contexts. The formulas are defined by the standard grammar:

$$A, B \;::=\; a \mid A \to B \mid \Box A \mid \Diamond A$$

where one can observe that the **IK** system is *modal* but not hybrid in the sense that worlds are used in sequents but not mentioned in formulas. The presentation we have given is equivalent to the original one [Sim94], and the distinctions made inside contexts are meant to make adequacy as obvious as possible for the given encoding of the system in **HyLF**.

The first step of the encoding is to define the structure of the worlds, and the congruence relation $\equiv$. In all modal logics that can be represented by the rules of **IK** shown above, the grammar of worlds is:

$$p, q, o \;::=\; \alpha \mid pRq \mid p * q \mid \varepsilon$$

where R is of arity 2 and represents the reachability relation of the Kripke semantics, and $*$ and ε of arities 2 and 0 respectively are used to encode sets of worlds.

REMARK 3.1. *In the natural deduction **IK** and its variants, the assumptions of the shape pRq involve only world names so that it should be xRy, but our grammar does not enforce such restriction. Indeed, the current definition of **HyLF** only allows the operators to be specified with an arity, not a complete grammar.*

This is however not a problem, as the encoding of rules preserve the invariant that in any world of the shape pRq, both p and q are variables: the worlds inside the assumptions are never accessed and decomposed by the rules, but simply compared, so that replacing a variable by a compound world does not break the encoding.

The precise meaning of operators on worlds is partly given by the congruence. There will be a part of this relation common to all systems based on the rules given for **IK**, which will actually be the congruence for the logic **IK** itself. Then, extending $\equiv$ with axioms concerning R will yield other, richer logics. The basic part of the congruence is defined by:

$$\begin{array}{rclcrcl} p * q & \equiv & q * p & \qquad & p * \varepsilon & \equiv & p \\ p * (q * o) & \equiv & (p * q) * o & & p * p & \equiv & p \end{array}$$

We can now define constants that will represent the inference rules of the system. These terms are given types representing the structure of formulas and sequents in **IK**, following the usual **LF** approach, where $\rightarrow$ stands for a non-dependent product:

$$\begin{array}{lll} \mathtt{o} : \mathtt{type} & \mathtt{pf} : \mathtt{o} \rightarrow \forall\sigma.\forall\alpha.\mathtt{type} & \Box : \mathtt{o} \rightarrow \mathtt{o} \\ & \supset : \mathtt{o} \rightarrow \mathtt{o} \rightarrow \mathtt{o} & \Diamond : \mathtt{o} \rightarrow \mathtt{o} \end{array}$$

Then, the purely implicational part of **IK** is described by rules for $\supset$ which have no effect on the current world, and they simply preserve and propagate it:

$$\begin{array}{l} \supset_I : (\mathtt{pf}\, A\, s\, x \rightarrow \mathtt{pf}\, B\, s\, x) \rightarrow \mathtt{pf}\, (A \supset B)\, s\, x \\ \supset_E : \mathtt{pf}\, (A \supset B)\, s\, x \rightarrow \mathtt{pf}\, A\, s\, x \rightarrow \mathtt{pf}\, B\, s\, x \end{array}$$

where we omit the outer bindings on A, B, s and x.

We can now consider the modal part of the system, encoding the rules for $\Box$ and $\Diamond$, which actually affect the worlds:

$$\begin{array}{rl} \Box_I & : (\forall\alpha.\, \mathtt{pf}\ A\, (s * xR\alpha)\, \alpha) \rightarrow \mathtt{pf}\, \Box A\, s\, x \\ \Box_E & : \mathtt{pf}\, \Box A\, s\, x \rightarrow \mathtt{pf}\, A\, (s * xRy)\, y \\ \Diamond_I & : \mathtt{pf}\, A\, s\, x \rightarrow \mathtt{pf}\, \Diamond A\, (s * yRx)\, y \\ \Diamond_E & : \mathtt{pf}\, \Diamond A\, s\, x \rightarrow (\forall\alpha.\, \mathtt{pf}\ A\, s\, \alpha \rightarrow \mathtt{pf}\, B\, (s * xR\alpha)\, y) \\ & \quad \rightarrow \mathtt{pf}\ B\, s\, y \end{array}$$

where we omit outer bindings on A, B, s, x and y. This encoding of the **IK** rules can be proven adequate in a straightforward way, since the types used here rely on the following correspondence between a sequent and its encoding:

$$\Omega; \Sigma; \Gamma \vdash A[x] \quad \longleftrightarrow \quad \mathtt{pf}\, A\, s\, x$$

where s is a world representing Σ by turning recursively Σ', xRy into $s' * xRy$, where s' represents Σ', and ε represents the empty set. The parts Ω and Γ are handled implicitly, as usual in **LF**, by the binders on world and term variables from the representation language. The same applies for other logics, such as **IS4**, where the Kripke semantics contains axioms for the reflexivity and the transitivity of R. Representing **IS4** is achived in our encoding by extending the congruence $\equiv$ with the equations:

$$xRx \equiv \varepsilon \qquad\qquad xRy * yRz \equiv xRz$$

without changing the rules given in Figure 3. The effect of this extension is to modify the set of formulas validated by the logic, so that, in particular we can prove the axioms $\Box A \supset A$ and $A \supset \Diamond A$ by using reflexivity, as well as $\Box A \supset \Box\Box A$ and $\Diamond\Diamond A \supset \Diamond A$ by using transitivity. These axioms illustrate how the the use of the congruence controls precisely the modal logic being represented, just as the axioms of a Kripke semantics. In order to obtain **IS5**, we can add the following axiom:

$$xRy * xRz \equiv yRz$$

to the ones introduced before to define **IS4**. Various other axioms from the standard proof theory of modal logics can be add in a similar way.

Linear reasoning. Another use of hybrid operations in **HyLF** consists in extending the representation language of types with an encoding of linear implication $\multimap$. This allows to subsequently represent other systems using the type $A \multimap B$, and in particular this the way a sequent calculus for linear logic can be adequately represented, as done in **HLF** [Ree09] where it was the goal of the introduction of hybrid operators. We can use in **HyLF** the exact same encoding as in **HLF**, provided that we use:

$$A \multimap B \;\triangleq\; \forall\alpha.{\downarrow}\gamma.\,(A@\alpha \rightarrow B@(\alpha * \gamma))$$

because the two operations $\forall\alpha$ and $@p$ behave the same in both frameworks. This encoding yields a direct encoding of the rules of introduction and elimination for $\multimap$, in **HyLF**:

$$\cfrac{\cfrac{\cfrac{\cfrac{\Omega, \alpha; \Gamma, A@\alpha[p] \vdash B[\alpha * p]}{\Omega, \alpha; \Gamma, A@\alpha[p] \vdash B@(\alpha * p)[p]}\,@\boldsymbol{i}}{\Omega, \alpha; \Gamma \vdash A@\alpha \rightarrow B@(\alpha * p)[p]}\,\Pi\boldsymbol{i}}{\Omega, \alpha; \Gamma \vdash {\downarrow}\gamma.(A@\alpha \rightarrow B@(\alpha * \gamma))[p]}\,{\downarrow}\boldsymbol{i}}{\Omega; \Gamma \vdash \forall\alpha.{\downarrow}\gamma.(A@\alpha \rightarrow B@(\alpha * \gamma))[p]}\,\forall\boldsymbol{i}$$

$$\cfrac{\cfrac{\cfrac{\cfrac{\Omega; \Gamma \vdash \forall\alpha.{\downarrow}\gamma.(A@\alpha \rightarrow B@(\alpha * \gamma))[p]}{\Omega; \Gamma \vdash {\downarrow}\gamma.(A@q \rightarrow B@(q * \gamma))[p]}\,\forall\boldsymbol{e}}{\Omega; \Gamma \vdash A@q \rightarrow B@(q * p)[p]}\,{\downarrow}\boldsymbol{e} \qquad \Omega; \Gamma \vdash A@q[p]}{\Omega; \Gamma \vdash B@(q * p)[p]}\,\Pi\boldsymbol{e}}{\Omega; \Gamma \vdash B[p]}\,@\boldsymbol{e}$$

where we omit the terms and simplify the notations by omitting also the side conditions, and the constraints on worlds induced by the use of @ restricts the set of valid proofs of $A \multimap B$ to those proofs of $A \rightarrow B$ that are actually the linear ones. Recall that the structure defined for worlds to make this encoding work relies on the following grammar:

$$p, q \;::=\; \alpha \;\mid\; p * q \;\mid\; \varepsilon$$

with the congruence interpreting $*$ as a constructor for building multisets, and ε as the empty multiset:

$$\frac{}{(p * q) * p' \equiv p * (q * p')} \qquad \frac{}{p * q \equiv q * p} \qquad \frac{}{p * \varepsilon \equiv p}$$

$$\frac{}{p \equiv p} \qquad \frac{q \equiv p}{p \equiv q} \qquad \frac{p \equiv q \qquad q \equiv p'}{p \equiv p'} \qquad \frac{p \equiv p' \qquad q \equiv q'}{p * q \equiv p' * q'}$$

In this definition, all the rules of the second line simply make $\equiv$ a congruence over the operators signature, which is a requirement in **HyLF**, while the three axioms given in the first line are actually assigning a meaning to these operators. More details on such an encoding and the use of $\multimap$ to represent other systems in a logical framework can be found in the literature [Ree09].

Modal reasoning. Just as linearity can be encoded inside the representation level of **HLF** and **HyLF**, it is conceivable to extend this language further, by defining modalities such as the $\Box$ and $\Diamond$ of modal logics within the syntax of types in **HyLF**. This would allow us to represent systems for which an adequate encoding relies on the ability to control separate worlds within a relation as in Kripke semantics. Using the same ideas as in the encoding of linearity in **HLF**, we can propose an encoding of $\Box$ in **HyLF**:

$$\Box A \;\triangleq\; \forall\alpha.{\downarrow}\gamma.A@(\alpha \triangleleft \alpha R\gamma)$$

where $\triangleleft$ would be an operator on worlds allowing to distinguish within the current world, in a sequent, between an actual world and a constraint on the relation R that must be validated. In the structure of worlds, we would then require enough operators to keep a structured form of information about the relation.

The worlds language needed to implement such an encoding in **HyLF** is similar to the one used for the object-level encoding of modal logics:

$$p, q \ ::= \ \alpha \mid pRq \mid p \triangleleft q \mid \varepsilon$$

but the congruence is made rather complicated by the distinction to be kept between the actual worlds and the parts of the worlds representing information on the relation R:

$$\frac{}{pR(p' \triangleleft q) \equiv pRp' \triangleleft q} \qquad \frac{}{(p \triangleleft q)Rp' \equiv (p \triangleleft q) \triangleleft pRp'}$$

$$\frac{}{p \triangleleft \varepsilon \equiv p} \qquad \frac{}{(p \triangleleft q) \triangleleft p' \equiv (p \triangleleft p') \triangleleft q}$$

$$\frac{}{p \triangleleft p \equiv p} \qquad \frac{}{p \triangleleft (q \triangleleft p') \equiv (p \triangleleft q) \triangleleft p'}$$

$$\frac{}{p \equiv p} \qquad \frac{q \equiv p}{p \equiv q} \qquad \frac{p \equiv q \qquad q \equiv p'}{p \equiv p'} \qquad \frac{p \equiv p' \qquad q \equiv q'}{p \triangleleft q \equiv p' \triangleleft q'}$$

especially since worlds will be composed in a way that will not preserve the clear distinction seen in the encoding of $\Box$, which justifies the use of rules to reorganise between actual worlds and constraints of the shape pRq.

Notice that with the encoding shown above, the distributivity axiom **K** is provable without any requirement on the structure of worlds, and the axioms **T** and **4** yield the standard conditions on worlds, reflexivity and transitivity. For example:

$$\dfrac{\dfrac{\dfrac{\dfrac{}{\Omega; \Gamma, \forall\alpha.{\downarrow}\gamma.A@(\alpha \triangleleft \alpha R\gamma)[p] \vdash \forall\alpha.{\downarrow}\gamma.(A@\alpha \triangleleft \alpha R\gamma)[p]}\,\boldsymbol{x}}{\Omega; \Gamma, \forall\alpha.{\downarrow}\gamma.A@(\alpha \triangleleft \alpha R\gamma)[p] \vdash {\downarrow}\gamma.(A@q \triangleleft qR\gamma)[p]}\,\forall\boldsymbol{e}}{\Omega; \Gamma, \forall\alpha.{\downarrow}\gamma.A@(\alpha \triangleleft \alpha R\gamma)[p] \vdash A@(q \triangleleft qRp)[p]}\,{\downarrow}\boldsymbol{e}}{\dfrac{\Omega; \Gamma, \forall\alpha.{\downarrow}\gamma.A@(\alpha \triangleleft \alpha R\gamma)[p] \vdash A[p]}{\Omega; \Gamma \vdash (\forall\alpha.{\downarrow}\gamma.A@(\alpha \triangleleft \alpha R\gamma)) \to A[p]}\,\Pi\boldsymbol{i}}\,@\boldsymbol{e}$$

which represents the axiom $\Box A \to A$, requires the use the rule for reflexivity in the congruence, for the proof to be valid in **HyLF**.

Furthermore, encoding the $\Diamond$ connective in such a way is more complicated, since it requires to use an existential quantification over worlds. This is not inconceivable, but the **HyLF** framework would then be extended beyond the definition given here.

4. CONCLUSION AND FUTURE WORK

We have presented here an extension of the **LF** framework that allows hybrid operators to be explicitly used for encodings, and discussed its properties, as well as the representation of systems for modal logics in this setting. This opens a number of questions for future work:

- we still to to fully develop the metatheory of **HyLF**, and in particular the underlying notion of reduction as well as the expressive power of the framework — from a practical viewpoint, we would need to impose some restrictions on the structure given to worlds, since for example we could think of defining a non-decidable congruence, that would lead to problems during unification,
- we can now try to encode other logics in **HyLF** than the few modal systems we mentioned, such as temporal, spatial or epistemic logics, but also other presentations of logic, for example based on hypersequents, that could be encoded using worlds,
- we need to investigate further the question of encoding the operators necessary for modal reasoning, starting with $\Box$ and $\Diamond$, but more generally we could try to identify other forms of reasoning that can be reified by a modality, and encoded into the world structure of **HyLF**,
- on the implementation side, the extended features of the worlds in **HyLF** yield the question of the feasability of any reasonable implementation of unification or coverage, but we hope that restrictions imposed on the congruence can reduce the complexity of the problem,
- we could also consider further extensions to the syntax of types and terms in **HyLF**, in particular to allow existential quantification over worlds, and over terms, or introduce a cartesian product as done in **HLF**,
- the expressive power of the hybrid framework might allow for the encoding of complex systems that combine several aspects requiring a particular world structure, such as the hybrid linear logic presented in [CD14], and the freedom offered in the definition of operators on worlds could even be enough to have general means of combining encodings, so that the two levels of structure on worlds needed for a linear treatment of context on one side, and access to the worlds on the other, could be merged.

5. REFERENCES

[ABM01] C. Areces, P. Blackburn, and M. Marx. Hybrid logics: Characterization, interpolation and complexity. *Journal of Symbolic Logic*, 66(3):977–1010, 2001.

[CD14] K. Chaudhuri and J. Despeyroux. A hybrid linear logic for constrained transition systems. In R. Matthes and A. Schubert, editors, *TYPES'13 Post-proceedings*, to appear in LIPIcs, 2014.

[CP03] I. Cervesato and F. Pfenning. A linear spine calculus. *Journal of Logic and Computation*, 13(5):639–688, 2003.

[GS11] D. Galmiche and Y. Salhi. Sequent calculi and decidability for intuitionistic hybrid logic. *Journal of Information and Computation*, 209(12):1447–1463, 2011.

[HHP93] R. Harper, F. Honsell, and G. Plotkin. A framework for defining logics. *Journal of the ACM*, 40(1):143–184, 1993.

[HL07] R. Harper and D. Licata. Mechanizing metatheory in a logical framework. *Journal of Functional Programming*, 17(4–5):613–673, 2007.

[MCHP04] T. Murphy VII, K. Crary, R. Harper, and F. Pfenning. A symmetric modal lambda calculus for distributed computing. In *LICS'04*, pages 286–295, 2004.

[Mon13] F. Montesi. *Choreographic Programming*. PhD thesis, 2013.

[Pri67] A. Prior. *Past, Present and Future*. Oxford University Press, 1967.

[Ree09] J. Reed. *A Hybrid Logical Framework*. PhD thesis, 2009.

[Sim94] A. Simpson. *The Proof Theory and Semantics of Intuitionistic Modal Logic*. PhD thesis, 1994.

[Tza99] M. Tzakova. Tableau calculi for hybrid logics. In N. Murray, editor, *TABLEAUX'99*, volume 1617 of *LNCS*, pages 278–292, 1999.

[WCPW04] K. Watkins, I. Cervesato, F. Pfenning, and D. Walker. A concurrent logical framework I: The propositional fragment. In S. Berardi, M. Coppo, and F. Damiani, editors, *TYPES'03 Post-proceedings*, volume 3085 of *LNCS*, pages 355–377, 2004.