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

Categories and Subject Descriptors

F.3.1 [**Logics and Meanings of Programs**]: Specifiying and Verifying and Reasoning about Programs—*mechanical verification,logics of programs*; F.4.1 [**Mathematical Logic and Formal Languages**]: Mathematical Logic—*lambda calculus and related systems, mechanical theorem proving*

General Terms

Theory

Keywords

Logical Frameworks, Hybrid Logics, Modal Logics

ABSTRACT

We discuss the extension of the 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 for this 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 logics and systems, and it greatly simplifies encodings by providing a representation language with an object-level 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 some systems that cannot be encoded adequately in LF without heavy manipulation of structures that must be dealt with manually both when defining their encoding and when reasoning about the system. One such example can be obtained by extending a standard logic, such as intuitionistic logic, by *hybrid* operations as suggested by [Pri67] and introduced later in standard proof theory — some 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* of the semantics. This yields elegant proof systems for logics with connectives performing 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 properties of such a modal logic then depend on the axioms on the 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 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 manipulating structures inside the syntax to, for example, automatically handle reflexivity, transitivity or other properties. Such an infrastructure has been developed for the specific purpose of encoding linearity in the HLF framework [Ree09] that extends LF with some support for hybrid operations. In HLF, the types and terms can use worlds that are not always variables but can also be *compounds* built from a binary $*$ operator with unit ε. This structure of worlds was used to encode linear implication in HLF at the level of the representation language: reasoning *linearly* is possible in HLF in the sense that $\multimap$ is available as a type — a macro using 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 the rules of the congruence:

```
o : type.
ω : type.
pf : o → ω → type.
rc : ω → ω → type.

⊃ : o → o → o.
□ : o → o.
```


LFMTP'14, July 17 2014, Vienna, Austria.

ACM 978-1-4503-2817-3/14/07 ...$15.00.
http://dx.doi.org/10.1145/2631172.2631178.

```
◇ : o → o.

refl : {α : ω} rc α α.
trans : {α, γ, σ : ω} rc α γ → rc γ σ → rc α
  σ.

□_I : {A : o}{α : ω} ({γ : ω} rc α γ → pf A γ)
  → pf (□ A) α.
□_E : {A : o} {α, γ : ω} pf (□ A) α → rc α γ
  → pf A γ.
(…)
```

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. As a consequence, encoding modal logics in LF or HLF is problematic from the viewpoint of adequacy, since there can be several proofs of reachability that are all identified in the proof on paper. In a similar way, the work we present stems from an attempt to represent and reason about a calculus based on hypersequents for some variant of linear logic [Mon13], which 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. This is essentially the same problem as encountered when encoding modal logics in HLF, which is 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 extensive use of the expressivity of hybrid operations. 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 hybrid framework, called HyLF, and discuss its 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 relying on canonical typing derivations [HL07]. In particular, we use the standard λ-calculus as our base, without *spines* [CP03], to keep the theory as simple as possible. The language of terms and types of our HyLF framework is an extension of canonical LF that supports various user-defined operators on worlds.

In the following, we denote by letters such as x, y and z *term variables*, by M, N and V *canonical terms*, by R and S *atomic terms*, by c *term constants*, by A and B *canonical types*, by F and G *atomic types*, by a *type constants* 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:

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

This system is similar to HLF [Ree09], with primitives at the level of terms reflecting the elimination rules of the world operators in the object language. For the sake of simplicity, no cartesian product is used in HyLF, and we collapse the dependent product and the 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, always of the shape:

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

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

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

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

Example 2.1. *The structure of worlds described in* HLF *is obtained in* HyLF *by an instantiation where the language of worlds is defined through the signature* $\{* : 2, \varepsilon : 0\}$*, which corresponds to the grammar:*

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

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

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

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

which are implementing AC-unification. ∎

In the rules from Figure 1, we use two kinds of judgements to indicate whether the type is *synthesized* from the term or is *checked* against the term, always at some world p, which will be denoted by $\Omega; \Gamma \vdash R \Rightarrow A[p]$ and by $\Omega; \Gamma \vdash M \Leftarrow A[p]$, respectively. In both cases, Ω is a set of world variables and Γ is a list of assumptions of the shape $x : A[p]$. This means that assumptions are assigned some world, as is often done

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

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

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

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

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

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

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

$$\boldsymbol{c}\ \frac{c:A\in\Sigma\quad p\in\mathcal{W}}{\Omega;\Gamma\vdash c\Rightarrow A[p]}\qquad \forall\boldsymbol{e}\ \frac{\Omega;\Gamma\vdash R\Rightarrow\forall\alpha.A[q]\quad p\in\mathcal{W}}{\Omega;\Gamma\vdash R\ \{p\}\Rightarrow A\{p/\alpha\}[q]}$$

Figure 1: Typing rules for HyLF terms

in sequent calculi for modal logics [GS11], but not in HLF. The two judgement forms are meant to enforce a separation between canonical and atomic terms, so that all terms typed are canonical. Moreover, in these rules:

- the condition $p\in\mathcal{W}$ ensures that p appears in the set $\mathcal{W}$ of worlds *well-formed* according to the signature $\mathcal{O}$,
- the list Σ is some *constants signature* implicitly associated to the judgement — we could write $\vdash_\Sigma$ but omit this for the sake of readability,
- we can 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,
- in the axioms $\boldsymbol{x}$ and $\boldsymbol{c}$, the context Γ and signature Σ should be checked for well-formation, following rules that we omit here but are straightforward,
- the notation $\{p/\alpha\}$ corresponds to the standard notion of capture-avoiding substitution in a term, of a world for a world variable,
- the notation $[M/x]_p$ corresponds to the notion of hereditary substitution, which we define below.

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

$$\boldsymbol{a}\ \frac{a:K\in\Sigma}{\Omega;\Gamma\vdash a\Rightarrow K}\qquad \forall\boldsymbol{f}\ \frac{\Omega;\Gamma\vdash F\Rightarrow\forall\alpha.K\quad p\in\mathcal{W}}{\Omega;\Gamma\vdash F\ \{p\}\Rightarrow K\{p/\alpha\}}$$

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

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

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

$$\downarrow\ \frac{\Omega,\alpha;\Gamma\vdash A::\mathtt{type}}{\Omega;\Gamma\vdash\downarrow\alpha.A::\mathtt{type}}\qquad @\ \frac{\Omega;\Gamma\vdash A::\mathtt{type}\quad p\in\mathcal{W}}{\Omega;\Gamma\vdash A@p::\mathtt{type}}$$

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

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

Figure 2: Kinding rules for HyLF

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

- the universal quantification over worlds $\forall\alpha.A$ yields a simple mechanism using abstraction and application, distinguished from the standard λ-calculus constraints by the $\{p\}$ syntax used in abstraction and application, so that $\lambda\{\alpha\}.M$ is related only to $R\ \{p\}$ and not standard application,
- the world localisation operation $A@p$ yields the two operations $M\,\mathtt{at}\,p$ and $R\,\mathtt{to}\,p$ which indicate that some term M 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 a term M will be evaluated using the operation $\mathtt{here}\,\alpha.M$, and associating this name to the world where some term R is currently evaluated, with the operation of *call-current-world* written $\mathtt{ccw}\,R$.

Canonical forms. 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 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 the notion of *reduction* to offer some way to represent the dynamics of systems we encode — for example, reductions for cut elimination in a given logic presented in a sequent calculus. This cannot be achieved inside a canonical system, as 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:

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

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

Reduction. Allowing non-canonical forms allows us to type more terms, but we want to reason under some equivalence relation such that any non-canonical term is associated to a canonical term. This relies on the notion of reduction over non-canonical terms from a grammar where M appears in the category R as shown in (1) — this can be 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 parametrized by some p where evaluation happens. The main reduction rules are:

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

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:

$$\begin{array}{rcll} \lambda x.M & \longrightarrow_p & \lambda x.N & \textit{if } M \longrightarrow_p N \\ M\ N & \longrightarrow_p & M'\ N' & \textit{if } M \longrightarrow_p M' \textit{ and } N \longrightarrow_p N' \\ \mathtt{ccw}\,M & \longrightarrow_p & \mathtt{ccw}\,N & \textit{if } M \longrightarrow_p N \\ \mathtt{here}\,\alpha.M & \longrightarrow_p & \mathtt{here}\,\alpha.N & \textit{if } M \longrightarrow_p N \end{array}$$

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

$$\begin{array}{rcll} M\ \mathtt{at}\ q & \longrightarrow_p & N\ \mathtt{at}\ q & \textit{if } M \longrightarrow_q N \\ M\ \mathtt{to}\ p & \longrightarrow_p & N\ \mathtt{to}\ p & \textit{if } M \longrightarrow_q N \quad \textit{for some } q \in \mathcal{W} \end{array}$$

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

Apart from this use of worlds in the evaluation of terms, the computational semantics of HyLF is based on standard notions. In particular, the key element during reduction is *substitution*. There are two kinds of substitution used in (2): the substitution $\{M/x\}$ of some term u for a term variable x, capture-avoiding and using 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 binding operations of world abstraction and of current world abstraction. Intuitively, this is a 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 parametrized to an *hereditary* form of substitution, where the redexes created by substitution are reduced immediately [WCPW04].

Definition 2.2. *For well-typed terms M, N, some variable x and a world p, the* hereditary substitution $M[N/x]_p$ *of N for x in M at world p is defined by the relation presented in Figure 3.*

$$\frac{}{x[N/x]_p = N} \quad \frac{}{y[N/x]_p = y} \quad \frac{}{c[N/x]_p = c}$$

$$\frac{R[N/x]_p = R' \quad M[N/x]_p = M'}{(R\ M)[N/x]_p = R'\ M'}$$

$$\frac{R[N/x]_p = \lambda y.V' \quad M[N/x]_p = M' \quad V'[M'/y]_p = V}{(R\ M)[N/x]_p = V}$$

$$\frac{M[N/x]_p = V}{(\lambda y.M)[N/x]_p = \lambda y.V} \qquad \frac{M[N/x]_p = V}{(\lambda\{\alpha\}.M)[N/x]_p = \lambda\{\alpha\}.V}$$

$$\frac{M[N/x]_q = V}{(M\ \mathtt{at}\ q)[N/x]_p = V\ \mathtt{at}\ q} \qquad \frac{M[N/x]_p = V}{(\mathtt{here}\,\alpha.M)[N/x]_p = \mathtt{here}\,\alpha.V}$$

$$\frac{R[N/x]_q = S}{(R\ \mathtt{to}\ p)[N/x]_p = S\ \mathtt{to}\ p} \qquad \frac{R[N/x]_q = M\ \mathtt{at}\ p}{(R\ \mathtt{to}\ p)[N/x]_p = M}$$

$$\frac{R[N/x]_p = S}{(\mathtt{ccw}\,R)[N/x]_p = \mathtt{ccw}\,S} \qquad \frac{R[N/x]_p = \mathtt{here}\,\alpha.M}{(\mathtt{ccw}\,R)[N/x]_p = M\{p/\alpha\}}$$

$$\frac{R[N/x]_p = S}{(R\ \{q\})[N/x]_p = S\ \{q\}} \qquad \frac{R[N/x]_p = \lambda\{\alpha\}.M}{(R\ \{q\})[N/x]_p = M\{q/\alpha\}}$$

Figure 3: Hereditary substitution

Notice 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 $(M\ \mathtt{at}\ p)\ \mathtt{to}\ q$, but it is well-typed only if $p = q$. We can now state that hereditary substitution is actually a particular implementation of the reductions shown in (2). Proof for some representative cases is provided in the Appendix.

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

Notice that the correspondence of hereditary substitution and reduction of non-canonical HYLF terms is established only for well-typed terms, which simplifies the situation as it prevents the creation of redexes through world substitutions making world substitution non-hereditary.

A critical property of the notion of hereditary substitution is that it preserves typeability in the canonical system, along with the fact that given two well-typed terms M and N, we can perform the substitution of N for a variable inside M. Proving this requires an induction, made more complex by the case where a β-redex is created, since it involves the type of the substituted N. This is however standard, and we only need to consider a simple approximation of the type of N.

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

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

We can now state the main theorem allowing us 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]. A generalized formulation of this theorem, together with representative cases of its proof, can be found in the Appendix.

THEOREM 2.2 (SUBSTITUTION).
For any terms M and N such that:
$\Omega; \Gamma, x : A[q], \Delta \vdash M \Leftarrow B[p]$ *and* $\Omega; \Gamma \vdash N \Leftarrow A[q]$, *it follows that*
$\Omega; \Gamma, \Delta[N/x] \vdash M[N/x] \Leftarrow B[N/x][p]$.

There is another theorem that allows us to perform the same operation on worlds, corresponding to the observation that if a world variable is used in a typing derivation, this derivation is parametric in that variable. Consistently replacing the variable with any given world always therefore yields a valid typing derivation. The proof of representative cases can again be found in the Appendix.

THEOREM 2.3 (WORLD SUBSTITUTION).
For a world q and a term M with $\Omega, \alpha; \Gamma \vdash M \Leftarrow B[p]$, there is a derivation of the judgement
$\Omega; \Gamma\{q/\alpha\} \vdash M\{q/\alpha\} \Leftarrow B\{q/\alpha\}[p\{q/\alpha\}]$.

We will not go into further details about the properties of terms and derivations forming the meta-theory 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 consider here two ways of using the hybrid operations to encode logics and systems. The first one is the standard encoding idea of LF, where the given system is defined with rules represented by typed constants added to a signature, so that adequacy can be proven between the system as seen *"on paper"* and its LF representation. The second approach follows the HLF example of encoding certain logical connectives into the type level of LF, and arguing 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, using standard encodings.

$$\boldsymbol{ax}\ \frac{A[x] \in \Gamma}{\Omega; \Sigma; \Gamma \vdash A[x]} \qquad \supset\boldsymbol{i}\ \frac{\Omega; \Sigma; \Gamma, A[x] \vdash B[x]}{\Omega; \Sigma; \Gamma \vdash A \supset B[x]}$$

$$\supset\boldsymbol{e}\ \frac{\Omega; \Sigma; \Gamma \vdash A[x] \qquad \Omega; \Sigma; \Gamma \vdash A \supset B[x]}{\Omega; \Sigma; \Gamma \vdash B[x]}$$

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

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

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

Figure 4: Inference rules for the basic logic IK

Notice that this first approach, in the HYLF framework, requires not only the definition of typed constants representing the rules of the system, but also the definition of operators and a congruence on worlds to instantiate the framework, as it is now parametric. However, there is a trade-off, where the added specification of the level of worlds subsequently makes the representation of the rules simpler.

Intuitionistic modal logics. The most natural system that can be encoded using the hybrid operations from HYLF is a natural deduction calculus for intuitionistic modal logics defined by Simpson [Sim94], using rules shown in Figure 4. In this system, the relation on worlds defining the particular flavour of modal logic used is mentioned explicitly, so that the same rules properly represent different modal logics, for example IK, IS4 or IS5. This system is well-suited for a presentation in HYLF, as we will be able to define inference 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 is made slightly more precise than the one given by Simpson, on the syntactic level, as 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:

$$A, B \ ::= \ a \mid A \supset B \mid \Box\, A \mid \Diamond\, A$$

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 using 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 is meant to represent reachability as described by the 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 presentation of basic modal logic* IK *as well as in its extensions, the assumptions of the shape pRq involve only world names, so that it should be xRy, but our grammar does not enforce such a restriction. Indeed, the current definition of* HYLF *only allows the operators to be specified with some arity, and through a complete grammar.*

This is however not a problem, since the encoding of rules preserves 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 through the congruence. There will be a part of this relation common to all the systems based on the rules given for IK, which will actually be the congruence for the logic IK itself. Then, extending $\equiv$ with other axioms concerning R yields other, richer logics. This basic part of the congruence $\equiv$ is defined by the equations:

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

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

```
o : type.
pf : o → ∀ α. type.
⊃ : o → o → o.
□ : o → o.
◇ : o → o.
```

Then, the purely implicational part of IK is described by rules for $\supset$ which do not use the reachability relation:

```
⊃I : (pf A {x} → pf B {x}) → pf (A ⊃ B) {x}
⊃E : pf (A ⊃ B){x} → pf A {x} → pf B {x}
```

where we omit all the outer bindings on A, B and x, that are necessary only to obtain a fully closed term and can be easily reconstructed.

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

```
□I : (∀ α . pf A {α} @ (s * x R α)) →
      pf (□ A){x} @ s.
□E : pf (□ A) {x} @ (s * x R y) →
      pf A {y} @ (s * x R y).
◇I : pf A {x} @ (s * x R y) →
      pf (◇ A) {y} @ (s * y R x).
◇E : pf (◇ A) {x} @ s →
       (∀ α . pf A {α} @ s →
        pf B {y} @ (s * x R α)) →
      pf B {y} @ s.
```

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

$$\Omega; \Sigma; \Gamma \vdash A[x] \quad \leftrightarrow \quad \texttt{pf A \{x\} @ s}$$

where `s` is some world representing Σ by turning recursively Σ', xRy into `s' * x R y`, where `s'` represents Σ', and ε is the empty set. Then, Ω and Γ are handled implicitly as usual in LF using the binders on world and term variables from the representation language. The same applies to logics such as for example IS4, where the Kripke semantics contains the reflexivity and transitivity axioms for R. Representing IS4 is achieved in our encoding by extending the congruence with the equations:

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

without changing the rules from Figure 4. 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 use of the congruence can control precisely the modal logic being represented, just as the axioms of some Kripke semantics. In order to obtain IS5, we just add the following axiom:

$$xRy * xRz \equiv yRz$$

to the ones used before to define IS4. Various other axioms from the standard proof theory of modal logics can be added 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 a type $A \multimap B$, and in particular this is 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:

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

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

$$\begin{array}{r} @\boldsymbol{i}\ \dfrac{\Omega, \alpha; \Gamma, A@\alpha[p] \vdash B[\alpha * p]}{\Omega, \alpha; \Gamma, A@\alpha[p] \vdash B@(\alpha * p)[p]} \\ \Pi\boldsymbol{i}\ \dfrac{\Omega, \alpha; \Gamma, A@\alpha[p] \vdash B@(\alpha * p)[p]}{\Omega, \alpha; \Gamma \vdash A@\alpha \to B@(\alpha * p)[p]} \\ {\downarrow}\boldsymbol{i}\ \dfrac{\Omega, \alpha; \Gamma \vdash A@\alpha \to B@(\alpha * p)[p]}{\Omega, \alpha; \Gamma \vdash {\downarrow}\gamma.(A@\alpha \to B@(\alpha * \gamma))[p]} \\ \forall\boldsymbol{i}\ \dfrac{\Omega, \alpha; \Gamma \vdash {\downarrow}\gamma.(A@\alpha \to B@(\alpha * \gamma))[p]}{\Omega; \Gamma \vdash \forall\alpha.{\downarrow}\gamma.(A@\alpha \to B@(\alpha * \gamma))[p]} \end{array}$$

and similarly a direct encoding of the elimination rule based on the elimination rules for each of the components used in the encoding of $\multimap$ in HYLF:

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

where we omit terms and simplify notations by omitting also the side conditions. The constraints on worlds induced by the use of @ restrict the set of valid proofs of $A \multimap B$ to those proofs of $A \to B$ that are actually linear ones. Details on this encoding and the use of $\multimap$ to represent other systems in a logical framework can be found in the literature [Ree09].

Towards modal reasoning. Just as linearity could be encoded at 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 ideas found in the encoding of linearity from HLF, one encoding of $\Box$ could be:

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

with $\triangleleft$ standing for an operator on worlds designed to allow the distinction, in the current world of 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. Notice that with an encoding as 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. Note that in the variant of the framework presented in this paper, enforcing that $\triangleleft$ take precisely a single world variable as its first argument is not possible. A modification allowing a compound world to contain a fixed set of zones, built using different operators, would therefore be required. We leave this for future work.

Furthermore, encoding the $\Diamond$ connective this way is more complicated, as it requires to use existential quantification over worlds. This is not inconceivable, but the current HyLF framework would need to be extended beyond the definition given here in terms of syntax and typing rules.

4. CONCLUSION AND FUTURE WORK

We have presented here an extension of the standard 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 need to fully develop the meta-theory 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 any non-decidable congruence, that would lead to problems in attempts at performing unification,
- we can now try to encode other logics in HyLF than the few modal systems we mentioned, for example any temporal, spatial or epistemic logic, but also just other presentations of logic, for example based on the notion of 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 attempt to identify other forms of reasoning that can be reified by some modality, and encoded into the world structure available in HyLF,
- on the side of implementation, the extended features of the worlds in HyLF yield the question of feasibility for any reasonable implementation of the algorithms 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 this hybrid framework might allow to encode 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 define some general means of combining the encodings, so that for example the two levels of structure on worlds needed for the linear treatment of context on one side, and the access to the worlds of modal logics on the other, could merge.

Acknowledgements. This work was partially funded by the *Demtech* grant number 10-092309 from the Danish Council for Strategic Research.

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] Robert Harper, Furio Honsell, and Gordon Plotkin. A framework for defining logics. *Journal of the ACM*, 40(1):143–184, January 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] Jason Reed. *A hybrid logical framework*. PhD thesis, Carnegie Mellon, 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.

APPENDIX

We present here partial proofs and generalized reformulations of theorems mentioned in Section 2 of this paper.

THEOREM .1 (HEREDITARY SUBSTITUTION).
For any terms M and N, if for a given world p there exists V such that $M[N/x]_p = V$, then we have the reduction $M\{N/x\} \longrightarrow\!\!\!\!\!\rightarrow_p V$.

PROOF. By structural induction on the derivation of $M[N/x]_p = V$. The following are representative cases:

Case: $$\frac{M[N/x]_p = V}{(\lambda y.M)[N/x]_p = \lambda y.V}$$

By the induction hypothesis, we obtain $M\{N/x\} \longrightarrow\!\!\!\!\!\rightarrow_p V$. Using the definition of capture-avoiding substitution and the properties of $\longrightarrow\!\!\!\!\!\rightarrow$, we conclude that indeed $(\lambda y.M)\{N/x\} = \lambda y.M\{N/x\} \longrightarrow\!\!\!\!\!\rightarrow_p \lambda y.V$

Case: $$\frac{}{x[N/x]_p = N}$$

By the definition of capture-avoiding substitution, $x\{N/x\} = N$. Therefore $x\{N/x\}$ reduces in zero steps to itself: $N \longrightarrow\!\!\!\!\!\rightarrow_p N$.

Case: $$\frac{R[N/x]_p = R' \quad M[N/x]_p = M'}{(R\ M)[N/x]_p = R'\ M'}$$

By the induction hypothesis, we obtain $R\{N/x\} \longrightarrow\!\!\!\!\!\rightarrow_p R'$ and $M\{N/x\} \longrightarrow\!\!\!\!\!\rightarrow_p M'$. Using the definition of capture-avoiding substitution and the standard distributivity properties of $\longrightarrow\!\!\!\!\!\rightarrow$, we conclude that indeed $(R\ M)\{N/x\} = (R\{N/x\})\ (M\{N/x\}) \longrightarrow\!\!\!\!\!\rightarrow_p R'\ M'$.

Case: $$\frac{R[N/x]_p = \lambda y.V' \quad M[N/x]_p = M' \quad V'[M'/y]_p = V}{(R\ M)[N/x]_p = V}$$

By the induction hypothesis, we obtain $R\{N/x\} \longrightarrow\!\!\!\!\!\rightarrow_p \lambda y.V'$, as well as $M\{N/x\} \longrightarrow\!\!\!\!\!\rightarrow_p M'$ and $V'\{M'/y\} \longrightarrow\!\!\!\!\!\rightarrow_p V$. Therefore, using the definitions of capture-avoiding substitution and relations $\longrightarrow\!\!\!\!\!\rightarrow$ and $\longrightarrow$, as well as the distributivity properties of $\longrightarrow\!\!\!\!\!\rightarrow$, we are able to conclude that $(R\ M)\{N/x\} = (R\{N/x\})\ (M\{N/x\}) \longrightarrow\!\!\!\!\!\rightarrow_p (\lambda y.V')\ M' \longrightarrow_p V'\{M'/y\} \longrightarrow\!\!\!\!\!\rightarrow_p V$.

Case: $$\frac{R[N/x]_q = M\ \texttt{at}\ p}{(R\ \texttt{to}\ p)[N/x]_p = M}$$

By the induction hypothesis, we obtain $R\{N/x\} \longrightarrow\!\!\!\!\!\rightarrow_p M\ \texttt{at}\ p$. Therefore, using the definitions of capture-avoiding substitution and relations $\longrightarrow\!\!\!\!\!\rightarrow$ and $\longrightarrow$, as well as the properties of $\longrightarrow\!\!\!\!\!\rightarrow$, we conclude that $(R\ \texttt{to}\ p)\{N/x\} = R\{N/x\}\ \texttt{to}\ p \longrightarrow\!\!\!\!\!\rightarrow_p (M\ \texttt{at}\ p)\ \texttt{to}\ p \longrightarrow_p M$.

□

THEOREM .2 (SUBSTITUTION).
Given any term N such that $\Omega; \Gamma \vdash N \Leftarrow A[q]$, and:

- *given any term M such that:*
  $\mathcal{D} :: \Omega; \Gamma, x : A[q], \Delta \vdash M \Leftarrow B[p]$,
  it follows that $\Omega; \Gamma, \Delta[N/x] \vdash M[N/x] \Leftarrow B[N/x][p]$;
- *given any term R such that:*
  $\mathcal{D} :: \Omega; \Gamma, x : A[q], \Delta \vdash R \Rightarrow B[p]$,
  it follows that either
  $\Omega; \Gamma, \Delta[N/x] \vdash R[N/x] \Rightarrow B[N/x][p]$ or
  $\Omega; \Gamma, \Delta[N/x] \vdash R[N/x] \Leftarrow B[N/x][p]$.

PROOF. We prove both parts by mutual lexicographic induction over first the simple type of N, $\tau(N)$, and second – the derivation of $\mathcal{D}$. The following are representative cases:

Case: $$\frac{\Omega; \Gamma, x : A[q], \Delta, y : B_0[p] \vdash R \Leftarrow B_1[p]}{\Omega; \Gamma, x : A[q], \Delta \vdash \lambda y.R \Leftarrow \Pi y{:}B_0.B_1[p]}$$

By the induction hypothesis and substitution properties, we obtain that
$\Omega; \Gamma, \Delta[N/x], y : B_0[N/x][p] \vdash R[N/x] \Leftarrow B_1[N/x][p]$.
Therefore, using the typing rule for lambda abstraction, we can also conclude
$\Omega; \Gamma, \Delta[N/x] \vdash \lambda y.R[N/x] \Leftarrow \Pi y{:}B_0[N/x].B_1[N/x][p]$
– which, using the definition of capture-avoiding substitution, can be equivalently written as
$\Omega; \Gamma, \Delta[N/x] \vdash (\lambda y.R)[N/x] \Leftarrow (\Pi y{:}B_0.B_1)[N/x][p]$

Case: $$\frac{\Omega; \Gamma, x : A[q], \Delta \vdash R \Rightarrow F[p'] \qquad p \equiv p' \qquad p \in \mathcal{W}}{\Omega; \Gamma, x : A[q], \Delta \vdash R \Leftarrow F[p]}$$

By the induction hypothesis, we may obtain one of two possible results:

- $\Omega; \Gamma, \Delta[N/x] \vdash R[N/x] \Rightarrow F[N/x][p']$ – in which case we may use the swap rule again, to move between synthesizing and checking, obtaining:
  $\Omega; \Gamma, \Delta[N/x] \vdash R[N/x] \Leftarrow F[N/x][p]$.
- $R[N/x] = M'$ and $\Omega; \Gamma, \Delta[N/x] \vdash M' \Leftarrow F[N/x][p']$ – in which case, we have to first invert the last typing rule used. Since M' is still of an atomic type $F[N/x]$, we can conclude that the last inference rule used was in fact the phase changing one above, therefore $M' = R[N/x]$ is in fact a maximally applied atomic term:
  $$\frac{\Omega; \Gamma, x : A[q], \Delta[N/x] \vdash R[N/x] \Rightarrow F[N/x][p''] \ p' \equiv p'' \ p' \in \mathcal{W}}{\Omega; \Gamma, \Delta[N/x] \vdash R[N/x] \Leftarrow F[N/x][p']}$$
  – but in this case, using the fact that the structure on worlds is a congruence, we can also rewrite the last step to be:
  $$\frac{\Omega; \Gamma, x : A[q], \Delta[N/x] \vdash R[N/x] \Rightarrow F[N/x][p''] \quad p \equiv p'' \quad p \in \mathcal{W}}{\Omega; \Gamma, \Delta[N/x] \vdash R[N/x] \Leftarrow F[N/x][p]}$$
  which is precisely, what we needed to show.

Case: $$\frac{}{\Omega; \Gamma, x : A[p], \Delta \vdash x \Rightarrow A[p]}$$

Since $\Omega; \Gamma \vdash N \Leftarrow A[p]$, using weakening we are able to obtain $\Omega; \Gamma, \Delta[N/x] \vdash N \Leftarrow A[p]$.

Case:

$$\frac{\Omega; \Gamma, x{:}A[p], \Delta \vdash R \Rightarrow \Pi y{:}B_0.B_1[p] \quad \Omega; \Gamma, x{:}A[p], \Delta \vdash M \Leftarrow B_1[p]}{\Omega; \Gamma, x{:}A[p], \Delta \vdash R\, M \Rightarrow B_1[M/y][p]}$$

By the induction hypothesis on the typing derivation for M, we obtain $\Omega; \Gamma, \Delta[N/x] \vdash M' \Leftarrow B_0[N/x][p]$ for $M' = M[N/x]$. Further, applying the induction hypothesis to the typing derivation of R yields two possible outcomes:

- $\Omega; \Gamma, \Delta[N/x] \vdash R[N/x] \Rightarrow \Pi y{:}B_0[N/x].B_1[N/x][p']$ – in which case we simply use the same typing rule again to obtain:
  $\Omega; \Gamma, \Delta[N/x] \vdash R[N/x]\, M[N/x] \Leftarrow B_1[N/x][M'/y][p]$.
  Since $B_1[N/x][M'/y] = B_1[M/y][N/x]$, this, in turn, is equivalent to
  $\Omega; \Gamma, \Delta[N/x] \vdash (R\, M)[N/x] \Leftarrow (B_1[t/y])[N/x][p]$
- $R[N/x] = M_0$ and
  $\Omega; \Gamma, \Delta[N/x] \vdash M_0 \Leftarrow \Pi y{:}B_0[N/x].B_1[N/x][p']$ – in which case, we have to first invert the last typing rule used, concluding that M_0 must in fact be $\lambda y.M''$ for some M'':

  $$\frac{\Omega; \Gamma, \Delta[N/x], y : B_0[N/x] \vdash M'' \Leftarrow B_1[N/x][p']}{\Omega; \Gamma, \Delta[N/x] \vdash \lambda y.M'' \Leftarrow \Pi y{:}B_0[N/x].B_1[N/x][p']}$$

  This is the case where hereditary substitution acts differently to an inductively defined one. Recall that the appropriate rule in this case requires us to continue substituting, as $(\lambda y.M'')\ M'$ contains a redex:

  $$\frac{R[N/x] = \lambda y.M'' \quad M[N/x] = M' \quad M''[M'/y] = M'''}{(R\, M)[N/x] = M'''}$$

  We can now use the inductive hypothesis again, this time on derivations $\Omega; \Gamma, \Delta[N/x] \vdash M' \Leftarrow B_0[N/x][p]$ as well as
  $\Omega; \Gamma, \Delta[N/x], y : B_0[N/x] \vdash M'' \Leftarrow B_1[N/x][p']$,
  to obtain
  $\Omega; \Gamma, \Delta[N/x] \vdash M''[M'/y] \Leftarrow B_1[N/x][M'/y][p]$,
  or equivalently
  $\Omega; \Gamma, \Delta[N/x] \vdash M''[M'/y] \Leftarrow (B_1[M/y])[N/x][p]$

□

Theorem .3 (World substitution).
Given a world $w \in \mathcal{W}$, and

- *given a term M such that $\mathcal{D} :: \Omega, \alpha; \Gamma \vdash M \Leftarrow B[p]$, it follows that $\Omega; \Gamma\{q/\alpha\} \vdash M\{q/\alpha\} \Leftarrow B\{q/\alpha\}[p\{q/\alpha\}]$;*
- *given a term R such that $\mathcal{D} :: \Omega, \alpha; \Gamma \vdash R \Rightarrow B[p]$, it follows that $\Omega; \Gamma\{q/\alpha\} \vdash R\{q/\alpha\} \Rightarrow B\{q/\alpha\}[p\{q/\alpha\}]$.*

Proof. By induction on the typing derivation $\mathcal{D}$. The following are representative cases:

Case:

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

Using the induction hypothesis we obtain:
$\Omega; \Gamma\{q/\alpha\}, x : A\{q/\alpha\}[p\{q/\alpha\}] \vdash M\{q/\alpha\} \Leftarrow B\{q/\alpha\}[p\{q/\alpha\}]$.
We continue by using the λ abstraction typing rule again, which yields
$\Omega; \Gamma\{q/\alpha\} \vdash \lambda x.M\{q/\alpha\} \Leftarrow \Pi x : A\{q/\alpha\}.B\{q/\alpha\}[p\{q/\alpha\}]$,
which is precisely
$\Omega; \Gamma\{q/\alpha\} \vdash (\lambda x.M)\{q/\alpha\} \Leftarrow (\Pi x : A.B)\{q/\alpha\}[p\{q/\alpha\}]$.

Case:

$$\frac{\Omega, \alpha; \Gamma \vdash R \Rightarrow F[r] \quad p \equiv r \quad p \in \mathcal{W}}{\Omega, \alpha; \Gamma \vdash R \Leftarrow F[p]}$$

By the induction hypothesis:
$\Omega; \Gamma\{q/\alpha\} \vdash R\{q/\alpha\} \Rightarrow F\{q/\alpha\}[r\{q/\alpha\}]$. Further, as $\equiv$ is a congruence rule, $p\{q/\alpha\} \equiv r\{q/\alpha\}$. Therefore, using the same typing rule yields:
$\Omega; \Gamma\{q/\alpha\} \vdash R\{q/\alpha\} \Leftarrow F\{q/\alpha\}[p\{q/\alpha\}]$.

Case:

$$\frac{x : A[p] \in \Gamma \quad p \in \mathcal{W}}{\Omega, \alpha; \Gamma \vdash x \Rightarrow A[p]}$$

If $x : A[p] \in \Gamma$, then also $x : A\{q/\alpha\}[p\{q/\alpha\}] \in \Gamma\{q/\alpha\}$ – and therefore $\Omega; \Gamma\{q/\alpha\} \vdash x \Rightarrow A\{q/\alpha\}[p\{q/\alpha\}]$.

Case:

$$\frac{\Omega, \alpha; \Gamma \vdash R \Rightarrow \Pi x : A.B[p] \qquad \Omega, \alpha; \Gamma \vdash M \Leftarrow A[p]}{\Omega, \alpha; \Gamma \vdash R\ M \Rightarrow B[M/x]_p[p]}$$

Using the induction hypothesis, we obtain:
$\Omega; \Gamma\{q/\alpha\} \vdash R\{q/\alpha\} \Rightarrow \Pi x : A\{q/\alpha\}.B\{q/\alpha\}[p\{q/\alpha\}]$ and
$\Omega; \Gamma\{q/\alpha\} \vdash M\{q/\alpha\} \Leftarrow A\{q/\alpha\}[p\{q/\alpha\}]$. Using the typing rule for application yields:
$\Omega; \Gamma\{q/\alpha\} \vdash (R\ M)\{q/\alpha\} \Rightarrow (B[M/x]_p)\{q/\alpha\}[p\{q/\alpha\}]$.

□