Propositions as [Types]

Steve Awodey*   Andrej Bauer†

Institut Mittag-Leffler
The Royal Swedish Academy of Sciences

June 2001

Abstract

Image factorizations in regular categories are stable under pullbacks, so they model a natural modal operator in dependent type theory. This unary type constructor $[A]$ has turned up previously in a syntactic form as a way of erasing computational content, and formalizing a notion of proof irrelevance. Indeed, semantically, the notion of a *support* is sometimes used as surrogate proposition asserting inhabitation of an indexed family.

We give rules for bracket types in dependent type theory and provide complete semantics using regular categories. We show that dependent type theory with the unit type, strong extensional equality types, strong dependent sums, and bracket types is the internal type theory of regular categories, in the same way that the usual dependent type theory with dependent sums and products is the internal type theory of locally cartesian closed categories.

We also show how to interpret first-order logic in type theory with brackets, and we make use of the translation to compare type theory with logic. Specifically, we show that the propositions-as-types interpretation is complete with respect to a certain fragment of intuitionistic first-order logic. As a consequence, a modified double-negation translation into type theory (without bracket types) is complete for all of classical first-order logic.

MSC2000 Classification: `03G30`, `03B15`, `18C50`

Keywords: categorical logic, type theory, regular categories, bracket types

*Department of Philosophy, Carnegie Mellon University, Pittsburgh, USA, e-mail: `awodey@cmu.edu`

†IMFM & Department of Mathematics, University in Ljubljana, Slovenia, e-mail: `Andrej.Bauer@andrej.com`

Acknowledgement. We gratefully acknowledge the support of the Institut Mittag-Leffler, the Royal Swedish Academy of Sciences, where this research was conducted. We also thank Peter Aczel, Lars Birkedal, Thierry Coquand, Nicola Gambino, Milli Maietti, Per Martin-Löf, Grigori Mints, Erik Palmgren, Frank Pfenning, Dana Scott, and Anton Setzer for their contributions. This work is part of the Logic of Types and Computation project at Carnegie Mellon University under the direction of Dana Scott.

1 Introduction

According to one conception of the theory of types, propositions and types are identified:

$$\text{Propositions} = \text{Types}\ .$$

This idea is well-known under the slogan *"Propositions as types"*, and has been developed by Martin-Löf [ML84] and others [How80, Tai]. In this report we distinguish propositions and types, but stay within a type-theoretic framework. We regard *some* types to be propositions, but not necessarily all of them. Additionally, each type A has an *associated proposition* $[A]$. This gives us a correspondence

$$\text{Propositions} \underset{[-]}{\overset{\hookrightarrow}{\leftrightarrows}} \text{Types}$$

which is in fact an *adjunction*. Since it will turn out that $[P] = P$ for any proposition P, the propositions are exactly the types in the image of the bracket constructor $[-]$. We call these types $[A]$ the *bracket types*. The picture is then simply

$$\text{Propositions} = [\text{Types}]\ .$$

These ideas are not new. Our work originated with Frank Pfenning's bracket types for erasing computational content [Pfe01]. Speaking somewhat vaguely, the idea is to use a bracket type for hiding computational content of a type. As a simple example, consider the computational content of a term p of type

$$\prod_{n:\mathbb{N}}\left(\sum_{m:\mathbb{N}}\mathsf{Eq}(2m, n) + \sum_{m:\mathbb{N}}\mathsf{Eq}(2m+1, n)\right)\ .$$

Given a natural number n, then $pn = \langle i, m \rangle$, where $i \in \{0,1\}$ and $m \in \mathbb{N}$, such that $n = 2m + i$. By bracketing the two dependent sums, we obtain the type

$$\prod_{n:\mathbb{N}} \left(\left[\sum_{m:\mathbb{N}} \mathsf{Eq}(2m, n) \right] + \left[\sum_{m:\mathbb{N}} \mathsf{Eq}(2m+1, n) \right] \right) .$$

A term q of this type hides the information that is provided by the dependent sums so that qn is either 0 or 1, depending on whether n is even or odd. In the extreme case, a term r of type

$$\left[\prod_{n:\mathbb{N}} \left(\sum_{m:\mathbb{N}} \mathsf{Eq}(2m, n) + \sum_{m:\mathbb{N}} \mathsf{Eq}(2m+1, n) \right) \right]$$

does not carry any computational content at all—it just witnesses the fact that every number is even or odd.

The bracket types which we consider are essentially the same as the *mono types* of Maietti [Mai98], in a suitable setting. Palmgren [Pal01] formulated a BHK interpretation of intuitionistic logic and used *image factorizations*, which are used in the semantics of our bracket types, to relate the BHK interpretation to the standard category-theoretic interpretation of propositions as subobjects. Aczel and Gambino [AG01] have promoted what they call *logic-enriched type theory* in which they separate the logic from type theory. The bracket types can be used to translate the primitive logic back into type theory (the usual translation of "propositions as types" works as well). Already in his *Dialectica* article, Lawvere [Law69] proposed a categorical treatment of proof theory that is closely related to bracket types.

The report is organized as follows. In Section 2 we introduce the bracket types. In Section 3 we give the semantics of bracket types in regular categories, and prove its soundness and completeness. In Section 4 we study some properties of bracket types. In Section 5 we show how bracket types are used in conjunction with other dependent types to define the logical connectives and quantifiers within type theory. In Section 6 we use bracket types to compare two interpretations of logic: the standard "propositions as types" interpretation, and the usual first-order one.

2 Bracket Types

We consider a Martin-Löf style dependent type theory [ML84, ML98]. For the formulation of bracket types we do not need dependent sums or products, but we sometimes assume that they are present in the type theory. We work in a type theory with *strong* and *extensional* equality and *strong* dependent sums, cf. [Jac99]. For reference, we list the rules in Appendix A.

Among the types, there are some that satisfy the following condition of *"proof irrelevance"*:

$$\frac{\Gamma \vdash P \ \mathsf{type} \qquad \Gamma \vdash q : P \qquad \Gamma \vdash p : P}{\Gamma \vdash p = q : P} \tag{1}$$

In words, this means that any two terms p and q of such a type P are (extensionally) equal. We call the types satisfying proof irrelevance *propositions.* They were called *mono types* by Maietti [Mai98], and there are other equivalent formulations.

If P and Q are propositions in this sense, then clearly so are

$$\mathsf{1}, \quad P \times Q, \quad P \to Q, \quad \textstyle\prod_{x:A} P$$

where in the last expression P may depend on an arbitrary type A. In logical terms, this means that propositions are already closed under the following logical operations:

$$\top, \quad P \wedge Q, \quad P \Longrightarrow Q, \quad \forall x{:}A.\, P\ .$$

In Section 5 we will see how to define the remaining first-order logical operations.

Because of proof irrelevance, if a proposition P is inhabited, then it is so by precisely one term (up to extensional equality). Thus, a typing judgment

$$\Gamma \vdash p : P$$

is like a statement of P's truth,

$$\Gamma \vdash P \ \mathsf{true}$$

as p does not play any role other than uniquely witnessing the fact that P holds.

We introduce a new type constructor $[-]$ which associates to each type A a proposition $[A]$, called the *associated proposition of* A. The axioms given in Figure 1 were designed with the following adjunction in mind, for any type A and proposition P:

$$\frac{x : A \vdash p : P}{\overline{\overline{x' : [A] \vdash p' : P}}} \tag{2}$$

The equivalence states that the bracket operation is left adjoint to the inclusion of propositions into types. We will derive this correspondence in the semantics of bracket types in Section 4. Using the rules provided in Figure 1, we can take $p' = (p \ \mathsf{where}\ [x] = x')$, since the equality condition on $p : P$ for elimination is satisfied by proof irrelevance (1). See remark 5 in Section 7 for consideration of alternate formulations of bracket types.

As an example, let us show that the term forming operation $[-]$ is 'epic' in the following sense:

$$\frac{\Gamma, x{:}A \vdash s\{[x]/u\} = t\{[x]/u\} : B}{\Gamma, u{:}[A] \vdash s = t : B} \tag{3}$$

If we think of a term $\Gamma, x{:}A \vdash r : B$ as an arrow $A \to B$ in the slice category over Γ, as we will in Section 3, then we have the following situation over Γ:

$$\begin{tikzcd} A \arrow[r, "{[-]}"] & {[A]} \arrow[r, shift left, "s"] \arrow[r, shift right, "t"'] & B \end{tikzcd}$$

Now (3) says that $s \circ [-] = t \circ [-]$ implies $s = t$ for arbitrary $s, t : A \to B$, which means that $[-]$ is epic. To prove (3), observe first that by the equality rule we have

$$\Gamma, x{:}A, y{:}A \vdash [x] = [y] : [A]$$

therefore

$$\Gamma, x{:}A, y{:}A \vdash s\{[x]/u\} = s\{[y]/u\} : [A]$$

which means that we can form the term $s\{[x]/u\}\ \mathsf{where}\ [x] = u$. Similarly, we can form the term $t\{[x]/u\}\ \mathsf{where}\ [x] = u$. Now we get

$$s \ =_\eta\ (s\{[x]/u\}\ \mathsf{where}\ [x] = u) \ = \ (t\{[x]/u\}\ \mathsf{where}\ [x] = u) \ =_\eta\ t\ .$$

The second equality follows from the assumption $s\{[x]/u\} = t\{[x]/u\}$ and the compatibility rule for where terms.

A consequence of (3) is the following conversion, called *exchange*:

$$b\ \mathsf{where}\ [x] = (p\ \mathsf{where}\ [y] = q) \quad = \quad (b\ \mathsf{where}\ [x] = p)\ \mathsf{where}\ [y] = q\ .$$

The rule is valid when $y \neq x$ and $y \notin \mathsf{FV}(b)$. By (3) it suffices to verify the exchange rule for the case $q = [z]$ where $z{:}A$ is a fresh variable. We then get

$$\begin{aligned} (b\ \mathsf{where}\ [x] = p)\ \mathsf{where}\ [y] = [z] \quad &=_\beta \quad b\{z/y\}\ \mathsf{where}\ [x] = (p\{z/y\}) \\ &= \quad b\ \mathsf{where}\ [x] = (p\{z/y\}) \\ &=_\beta \quad b\ \mathsf{where}\ [x] = (p\ \mathsf{where}\ [y] = [z]) \end{aligned}$$

Bracket types

$$\frac{\Gamma \vdash A \ \mathsf{type}}{\Gamma \vdash [A] \ \mathsf{type}} \text{ formation} \qquad \frac{\Gamma \vdash a : A}{\Gamma \vdash [a] : [A]} \text{ intro}$$

$$\frac{\Gamma \vdash q : [A] \quad \Gamma \vdash B \ \mathsf{type} \quad \Gamma, x{:}A \vdash b : B \quad \Gamma, x{:}A, y{:}A \vdash b = b\{y/x\} : B}{\Gamma \vdash b \ \mathsf{where}\ [x] = q : B} \text{ elim}$$

$$\frac{\Gamma \vdash p : [A] \qquad \Gamma \vdash q : [A]}{\Gamma \vdash p = q : [A]} \text{ equality}$$

Conversions

$$\begin{array}{rcl} b \ \mathsf{where}\ [x] = [a] & =_\beta & b\{a/x\} \\ b\{[x]/u\} \ \mathsf{where}\ [x] = q & =_\eta & b\{q/u\} \end{array}$$

Free variables

$$\begin{array}{rcl} \mathsf{FV}([A]) & = & \mathsf{FV}(A) \\ \mathsf{FV}([a]) & = & \mathsf{FV}(a) \\ \mathsf{FV}(b \ \mathsf{where}\ [x] = q) & = & (\mathsf{FV}(b) \setminus \{x\}) \cup \mathsf{FV}(q) \end{array}$$

Substitution

$$\begin{array}{rcl} [A]\{t/x\} & = & [A\{t/x\}] \\ [a]\{t/x\} & = & [a\{t/x\}] \\ (b \ \mathsf{where}\ [x] = q)\{t/y\} & = & b\{t/y\} \ \mathsf{where}\ [x] = (q\{t/y\}) \end{array}$$

(provided $x \neq y$ and capture of x in t is avoided)

Compatibility rules

$$\begin{array}{rcl} A = A' & \Longrightarrow & [A] = [A'] \\ a = a' & \Longrightarrow & [a] = [a'] \\ b = b' \wedge q = q' & \Longrightarrow & (b \ \mathsf{where}\ [x] = q) = (b' \ \mathsf{where}\ [x] = q') \end{array}$$

Figure 1: Bracket types

In the second equality we took into account the fact that y does not occur freely in b, and in the third equality we applied the η rule to the subterm $p\{z/y\}$, which we can do because of the compatibility rules.

3 Categorical Semantics of Bracket Types

In this section we present a semantics for bracket types in regular categories, see e.g. [Bor94] for the latter. The rules in Figure 1 are sound and complete for such semantics.

Definition 3.1 A *regular category* $\mathcal{C}$ is a category with finite limits in which

1. every kernel pair has a coequalizer, and
2. the pullback of a regular epimorphism is a regular epimorphism.

The first condition states that in a regular category we can form quotients by equationally defined equivalence relations, and the second condition requires such quotients to behave well with respect to finite limits.

Let us first recall how to interpret dependent type theory with dependent sums $\sum$ and strong extensional equality Eq in a category with finite limits. We use the semantic bracket $[\![X]\!]$ to denote the interpretation of X, where X could be a type, a term, a context, or a judgment. When no confusion can arise, we omit the semantic brackets, especially in diagrams, in order to improve readability. We usually denote the interpretation of a context $x_1{:}A_1, \ldots, x_n{:}A_n$ as $(A_1, \ldots, A_n)$ instead of $[\![x_1{:}A_1, \ldots, x_n{:}A_n]\!]$.

The empty context is interpreted as the terminal object 1. The interpretation of a type in a context

$$\Gamma \vdash A \ \mathsf{type}$$

is given in the slice category $\mathcal{C}/[\![\Gamma]\!]$ by an arrow, called a *display map*,

$$\begin{tikzcd} {[\![\Gamma, x{:}A]\!]} \arrow[d, "{[\![\Gamma \vdash A]\!]}"'] \\ {[\![\Gamma]\!]} \end{tikzcd}$$

where we here abbreviated the name of the arrow. Its domain is the interpretation of the context $\Gamma, x{:}A$.

A term in a context

$$\Gamma \vdash t : A$$

is interpreted by a point of (Γ, A) in the slice $\mathcal{C}/[\![\Gamma]\!]$

$$\begin{tikzcd}
(\Gamma) \arrow[rr, "{[\![\Gamma\vdash t:A]\!]}"] \arrow[dr, "="'] & & (\Gamma, A) \arrow[dl, "\Gamma\vdash A"] \\
& (\Gamma) &
\end{tikzcd}$$

In other words, a term $\Gamma \vdash t : A$ is interpreted as a section of the interpretation of $\Gamma \vdash A$ type. Normally, we write just $[\![t]\!]$ or t instead of $[\![\Gamma \vdash t : A]\!]$.

We interpret substitutions of a term a for a variable x,

$$\frac{\Gamma \vdash a : A \qquad \Gamma, x{:}A \vdash B \ \mathsf{type}}{\Gamma \vdash B\{a/x\} \ \mathsf{type}} \qquad\qquad \frac{\Gamma \vdash a : A \qquad \Gamma, x{:}A \vdash t : B}{\Gamma \vdash t\{a/x\} : B\{a/x\}}$$

as indicated in the following pullback diagram:

$$\begin{tikzcd}
(\Gamma) \arrow[r, "a"] \arrow[dr, "{[\![t\{a/x\}]\!]}"] \arrow[ddr, bend right, "="'] & (\Gamma, x{:}A) \arrow[dr, "t"] & \\
& {[\![\Gamma, B\{a/x\}]\!]} \arrow[r] \arrow[d, "{[\![\Gamma\vdash B\{a/x\}]\!]}"] \arrow[dr, phantom, "\lrcorner", very near start] & (\Gamma, x{:}A, B) \arrow[d, "{\Gamma, x:A\vdash B}"] \\
& (\Gamma) \arrow[r, "a"'] & (\Gamma, A)
\end{tikzcd}$$

The interpretation of a dependent sum formed as

$$\frac{\Gamma, x{:}A \vdash B \ \mathsf{type}}{\Gamma \vdash \sum_{x:A} B \ \mathsf{type}}$$

is the composition of the arrows

$$\begin{tikzcd}
(\Gamma, A, B) \arrow[d, "{\Gamma, A\vdash B}"'] \arrow[dd, bend left, "\Gamma\vdash\sum_A B"] \\
(\Gamma, A) \arrow[d, "\Gamma\vdash A"'] \\
(\Gamma)
\end{tikzcd}$$

This gives us a connection between the interpretation of contexts and dependent sums, because it must be the case that $[\![\Gamma, A, B]\!] = [\![\Gamma, \sum_{x:A} B]\!]$.

The interpretation of an equality type formed as

$$\frac{\Gamma \vdash s : A \qquad \Gamma \vdash t : A}{\Gamma \vdash \mathsf{Eq}_A(s,t)\ \mathsf{type}}$$

is the equalizer of s and t, as in the following diagram:

$$\begin{tikzcd} (\Gamma, \mathsf{Eq}_A(s,t)) \arrow[r, "{[\![\Gamma \vdash \mathsf{Eq}_A(s,t)]\!]}"] & (\Gamma) \arrow[r, shift left, "s"] \arrow[r, shift right, "t"'] & (\Gamma, A) \end{tikzcd}$$

When s and t are the same term, the equalizer is trivial and we have

$$[\![\Gamma, \mathsf{Eq}_A(t,t)]\!] = [\![\Gamma]\!]$$

From this we obtain the interpretation of a 'reflexivity' term

$$\frac{\Gamma \vdash t : A}{\Gamma \vdash \mathsf{r}(t) : \mathsf{Eq}_A(t,t)}$$

simply as the identity arrow

$$(\Gamma) \xrightarrow{[\![\mathsf{r}(t)]\!] \,=\, 1_{[\![\Gamma]\!]}} (\Gamma) = (\Gamma, \mathsf{Eq}_A(t,t))$$

Next, we give the interpretation of the first and the second projection from a dependent sum. Consider the terms

$$\frac{\Gamma \vdash p : \sum_{x:A} B}{\Gamma \vdash \pi_1(p) : A} \qquad \frac{\Gamma \vdash p : \sum_{x:A} B}{\Gamma \vdash \pi_2(p) : B\{\pi_1(p)/x\}}$$

We interpret $\pi_1(p)$ as the composition of arrows

$$(\Gamma) \xrightarrow{p} (\Gamma, A, B) \xrightarrow{\Gamma, A \vdash B} (\Gamma, A)$$

and $\pi_2(p)$ as in the following diagram:

$$\begin{tikzcd} (\Gamma) \arrow[dr, "{[\![\pi_2(p)]\!]}"] \arrow[drr, bend left, "p"] \arrow[ddr, bend right, "="'] & & \\ & (\Gamma, B\{\pi_1(p)/x\}) \arrow[r] \arrow[d] \arrow[dr, phantom, "\lrcorner", very near start] & (\Gamma, x{:}A, B) \arrow[d, "\Gamma, A \vdash B"] \\ & (\Gamma) \arrow[r, "\pi_1(p)"'] & (\Gamma, A) \end{tikzcd}$$

The arrow $[\![\pi_2(p)]\!]$ is the unique arrow obtained from the universal property of the displayed pullback diagram. A dependent pair formed as

$$\frac{\Gamma \vdash a : A \qquad \Gamma, x{:}A \vdash B \ \mathsf{type} \qquad \Gamma \vdash b : B\{a/x\}}{\Gamma \vdash \langle a, b\rangle : \sum_{x:A} B}$$

is interpreted as the composition of b with the top arrow in the diagram

$$\begin{tikzcd}
(\Gamma, B\{a/x\}) \arrow[r] \arrow[d, "\Gamma \vdash B\{a/x\}"] \arrow[dr, phantom, "\lrcorner", very near start] & (\Gamma, A, B) \arrow[d, "\Gamma, A \vdash B"] \\
(\Gamma) \arrow[r, "a"'] \arrow[u, bend left, "b"] & (\Gamma, A)
\end{tikzcd}$$

This completes the outline of the interpretation of dependent type theory with $\sum$ and Eq types in a finitely complete category.

Remark 3.2 It is well known that certain coherence problems arise when we interpret dependent type theory as above. The problems are caused by the fact that in general the result of successive pullbacks along arrows $g : B \to C$ and $f : A \to B$ is only *isomorphic* to the pullback along the composition $g \circ f$, whereas for a completely water-tight interpretation *equality* is required.

There are several standard ways of resolving this problem, most notably by interpreting the type theory in a suitable fibered category [Jac99], and then applying technical results pertaining to these [Hof95]. We do not wish to obscure matters by employing such technical devices. The interested reader may either translate our presentation into a suitable fibered setting, or assume some other remedy, such as making a coherent choice of pullbacks. (For the syntactic category in Section 3, such pullbacks can be chosen simply as substitutions.)

We now proceed with the interpretation of bracket types. A regular category $\mathcal{C}$ has stable *regular epi–mono image factorizations.* Every arrow $f : A \to B$ can be factored uniquely up to isomorphism as a regular epi followed by a mono

$$\begin{tikzcd}
A \arrow[rr, "f"] \arrow[dr, two heads] & & B \\
& \mathsf{Im}(f) \arrow[ur, tail] &
\end{tikzcd}$$

The factorization is obtained by taking the coequalizer q of the kernel pair (π_1, π_2) of f, as in the following diagram:

$$\begin{tikzcd} A \times_B A \arrow[r, shift left, "\pi_1"] \arrow[r, shift right, "\pi_2"'] & A \arrow[rr, "f"] \arrow[dr, twoheadrightarrow, "q"'] & & B \\ & & \mathsf{Im}(f) \arrow[ur, rightarrowtail, "i"'] & \end{tikzcd}$$

The arrow $i : \mathsf{Im}(f) \to B$ exists and is unique with $i \circ q = f$ because f coequalizes its own kernel pair. It can moreover be shown that i is always monic.

A bracket type

$$\frac{\Gamma \vdash A \ \mathsf{type}}{\Gamma \vdash [A] \ \mathsf{type}}$$

is interpreted as the image of $[\![\Gamma \vdash A]\!]$:

$$\begin{tikzcd} (\Gamma, A) \arrow[r, twoheadrightarrow, "{[-]}"] \arrow[dr, "\Gamma\vdash A"'] & {[\![\Gamma, [A]]\!]} = \mathsf{Im}(\Gamma \vdash A) \arrow[d, rightarrowtail, "{[\![\Gamma\vdash[A]]\!]}"] \\ & (\Gamma) \end{tikzcd}$$

The regular epi part of the factorization is used in the interpretation of term bracketing

$$\frac{\Gamma \vdash a : A}{\Gamma \vdash [a] : [A]}$$

The interpretation of $[a]$ is the composition

$$(\Gamma) \xrightarrow{a} (\Gamma, A) \xrightarrow{[-]} (\Gamma, [A])$$

It remains to interpret the where terms. Consider

$$\frac{\Gamma \vdash q : [A] \qquad \Gamma, x{:}A \vdash b : B \qquad \Gamma, x{:}A, y{:}A \vdash b = b\{y/x\} : B}{\Gamma \vdash b \ \mathsf{where}\ [x] = q : B}$$

The various terms and types occurring above are interpreted in the slice over $[\![\Gamma]\!]$, as shown in the following diagram:

$$\begin{tikzcd}
(\Gamma, x{:}A, y{:}A) \arrow[r, shift left, "x"] \arrow[r, shift right, "y"'] & (\Gamma, A) \arrow[r, two heads, "{[-]}"] \arrow[dr, "b"'] & (\Gamma, [A]) \arrow[d, "\overline{b}"] & (\Gamma) \arrow[l, "q"'] \arrow[dl, "(b \text{ where } [x]=q)"] \\
& & (\Gamma, A, B) &
\end{tikzcd}$$

By assumption, the arrow labelled b coequalizes the two projections. The regular epi $[-]$ is the coequalizer of those two projections, therefore $\Gamma \vdash b$ factors uniquely through $[-]$ via $\overline{b}$. The interpretation of $(b \text{ where } [x] = q)$ is the composition $\overline{b} \circ q$.

Theorem 3.3 *The interpretation of bracket types in regular categories is sound.*

Proof. We omit the routine proof of soundness of the interpretation of dependent sums and equality types, and concentrate on the interpretation of bracket types. We need to verify the equality rule, two conversion rules, the substitution rules and the compatibility rules.

When the equality rule is translated into the semantics, it states that the arrows p and q in the following diagram are equal:

$$\begin{tikzcd}
(\Gamma) \arrow[r, shift left, "p"] \arrow[r, shift right, "q"'] \arrow[dr, "="'] & (\Gamma, [A]) \arrow[d, tail, "\Gamma\vdash[A]"] \\
& (\Gamma)
\end{tikzcd}$$

Since p and q are sections of the mono $[\![\Gamma \vdash [A]]\!]$ they must be equal. Next, consider the β-rule

$$b \text{ where } [x] = [a] \quad =_\beta \quad b\{a/x\} \ .$$

The relevant diagram is

$$\begin{tikzcd}
(\Gamma, A) \arrow[d, two heads, "{[-]}"'] \arrow[dr, "b"] & (\Gamma) \arrow[l, "a"'] \arrow[d, "(b \text{ where } [x]=[a])"] \\
(\Gamma, [A]) \arrow[r, "\overline{b}"'] & (\Gamma, A, B)
\end{tikzcd}$$

The arrow $\overline{b}$ is the unique factorization of b through $[-]$. By construction, the lower-left triangle commutes, and the right-hand arrow is defined to be the composition $\overline{b} \circ [-] \circ a$, which implies that the upper-right triangle commutes. This is precisely what the β-rule states.

To verify the η-rule

$$b\{[x]/u\} \text{ where } [x] = q \quad =_\eta \quad b\{q/u\}$$

we consider the following diagram:

$$\begin{tikzcd}
(\Gamma, x{:}A) \arrow[r, two heads, "{[-]}"] \arrow[dr, "{b\{[x]/u\}}"'] & (\Gamma, u{:}[A]) \arrow[d, "b"] & (\Gamma) \arrow[l, "q"'] \arrow[dl, "{(b\{[x]/u\} \text{ where } [x]=q)}"] \\
& (\Gamma, A, B) &
\end{tikzcd}$$

The arrow $b\{[x]/u\}$ is the composition of $[-]$ and b. There is a unique factorization $\overline{b\{[x]/u\}}$ of $b\{[x]/u\}$ through $[-]$, and the interpretation of $b\{[x]/u\}$ where $[x] = q$ is the composition $\overline{b\{[x]/u\}} \circ q$. But $b\{[x]/u\}$ also factors through $[-]$ via b, so it must be that $\overline{b\{[x]/u\}} = b$. Now the η-rule follows, because the arrow $b \circ q$ is the interpretation of $b\{q/u\}$.

The substitution rules are valid because the regular epi–mono factorizations are stable under pullbacks. The compatibility rules are valid simply because we interpreted the bracket types and terms by well defined categorical operations (which therefore preserve equality). ■

Theorem 3.4 *The semantics of bracket types in regular categories is complete.*

Proof. In order to prove the theorem we build a syntactic category $\mathcal{S}$ from the dependent type theory $\mathbb{D}$ with $\mathsf{1}$, $\sum$, Eq, and $[-]$ types. We then show that $\mathcal{S}$ is a regular category, and that the interpretation of $\mathbb{D}$ in $\mathcal{S}$ validates precisely all the provable equations between terms.

The objects of $\mathcal{S}$ are the closed types of $\mathbb{D}$. The arrows from a closed type A to a closed type B are represented by terms

$$x{:}A \vdash t : B$$

where two such terms s and t represent the same arrow if, and only if, $\mathbb{D}$ proves that they are equal $x{:}A \vdash s = t : B$ (since we are working with extensional equality it does not matter which sense of 'equal' we take). Furthermore, two terms in a context will be considered equal if we can obtain one from the other by renaming bound and free variables in a capture-avoiding way.

The composition of arrows $x : A \vdash t : B$ and $y : B \vdash s : C$ is the arrow $x : A \vdash s\{t/y\}$. That composition is well-defined and associative follows from the properties of substitution. The identity arrow on A is represented by $x : A \vdash x : A$.

Since terms representing the same arrow must be provably equal, it suffices to show that $\mathcal{S}$ is regular. Let us prove first that it has finite limits. The terminal object is 1. Indeed, for any type A we have an arrow $x{:}A \vdash \star : 1$, and for any other arrow $x{:}A \vdash t : 1$ we have $x{:}A \vdash t = \star : 1$ by the equality axiom for the unit type. It is fairly easy to verify that $\mathcal{S}$ has binary products: the product of A and B is the type $A \times B = \sum_{x:A} B$.

The equalizer of arrows $x : A \vdash s : B$ and $x : A \vdash t : B$ is constructed as follows:

$$\textstyle\sum_{x:A} \mathsf{Eq}_B(s,t) \xrightarrow{\quad \pi_1 \quad} A \overset{s}{\underset{t}{\rightrightarrows}} B$$

Proving that the arrow π_1 equalizes s and t amounts to proving that

$$v : \textstyle\sum_{x:A} \mathsf{Eq}_B(s,t) \vdash s\{\pi_1(v)/x\} = t\{\pi_1(v)/x\} \ .$$

This follows from the extensionality of equality since $\pi_2(v)$ is a term of type

$$\mathsf{Eq}_B(s\{\pi_1(v)/x\}, t\{\pi_1(v)/x\}) \ .$$

Suppose $z : C \vdash h : A$ equalizes s and t:

$$z : C \vdash s\{h/x\} = t\{h/x\} : B$$

Then we can form the term witnessing the equality

$$z : C \vdash \mathsf{r}(s\{h/x\}) : \mathsf{Eq}_B(s\{h/x\}, t\{h/x\})$$

and from that the factorization of h through the equalizer:

$$z : C \vdash \langle h, \mathsf{r}(s\{h/x\}) \rangle : \textstyle\sum_{x:A} \mathsf{Eq}_B(s,t)$$

This arrow is unique because any two terms of a (strong extensional) Eq type are equal. Therefore, $\mathcal{S}$ has all finite limits.

It remains to show that $\mathcal{S}$ has stable coequalizers of kernel pairs. Before proceeding with the proof, let us spell out the interpretation of dependent types with $\mathsf{1}$, $\sum$ and Eq in $\mathcal{S}$.

Dependent contexts are interpreted by nested dependent sums

$$\begin{aligned} [\![\mathsf{1}]\!] &= \mathsf{1} \\ [\![x_1{:}A_1, \ldots, x_n{:}A_n]\!] &= \textstyle\sum_{x_1:A_1}\sum_{x_2:A_2}\cdots\sum_{x_{n-1}:A_{n-1}} A_n \end{aligned}$$

In order to keep the notation simple we denote such a nested sum by $(A_1, \ldots, A_n)$. A type in a context, $\Gamma \vdash A$ type, is interpreted by a suitable display map

$$\begin{tikzcd} (\Gamma, A) \arrow[d, "\Gamma\vdash A"'] \\ (\Gamma) \end{tikzcd}$$

More precisely, if Γ is $y_1{:}B_1, \ldots, y_n{:}B_n$, then the display map $\Gamma \vdash A$ is the term

$$p : (B_1, \ldots, B_n, A) \vdash \langle \pi_1(p), \pi_1(\pi_2(p)), \ldots, \pi_1(\pi_2^{n-1}(p))\rangle : (B_1, \ldots, B_n)\ .$$

With this notation, we get a good match between the syntax of dependent types and their interpretation in $\mathcal{S}$. For example, a dependent sum in a dependent context

$$\frac{\Gamma, x{:}A \vdash B \ \mathsf{type}}{\Gamma \vdash \sum_{x:A} B}$$

is interpreted essentially "by itself" as the arrow

$$(\Gamma, \textstyle\sum_{x:A} B) \xrightarrow{\Gamma\vdash\sum_{x:A} B} (\Gamma)$$

Similarly, an equality type in a dependent context is interpreted essentially by itself, except that we must form the nested dependent sums in order to interpret the dependent context in which the type is placed.

Consider an arrow $x : A \vdash t : B$ in $\mathcal{S}$. We can form the dependent type

$$y : B \vdash \textstyle\sum_{x:A} \mathsf{Eq}_B(t, y) \ \mathsf{type}$$

and re-interpret it in $\mathcal{S}$. Its interpretation is the display map

$$p : \textstyle\sum_{y:B}\sum_{x:A} \mathsf{Eq}_B(t, y) \vdash \pi_1(p) : B$$

This display map is isomorphic in $\mathcal{S}/B$ to the arrow $x : A \vdash t : B$ that we started with. Indeed, A is isomorphic to $\sum_{y:B}\sum_{x:A}\mathsf{Eq}_B(t, y)$ via the isomorphisms

$$x : A \vdash \langle t, \langle x, \mathsf{r}(t)\rangle\rangle : \textstyle\sum_{y:B}\sum_{x:A}\mathsf{Eq}_B(t, y)$$

and

$$p : \textstyle\sum_{y:B}\sum_{x:A}\mathsf{Eq}_B(t, y) \vdash \pi_1(\pi_2(p)) : A$$

It is easy to check that these two isomorphisms commute with the arrows t and π_1. This shows that every arrow in $\mathcal{S}$ is isomorphic to the interpretation of a dependent type in a context of the form $z : C \vdash D$ type. Thus, in order to show that $\mathcal{S}$ has coequalizers of kernel pairs, we only need to find the coequalizers of the kernel pairs of arrows that are interpretations of such types, namely those of the form

$$p : \textstyle\sum_{z:C} D \vdash \pi_1(p) : C \tag{4}$$

Because any $p : \sum_{z:C} D$ can be written uniquely as a pair $\langle z, w\rangle$ with $z : C$ and $w : D$, we shall write (4) more readably as

$$z : C, w : D \vdash z : C \tag{5}$$

This way we avoid the use of nested projections later on, when we have to deal with nested dependent sums. The kernel pair of (5) is easily seen to be

$$(z{:}C, v{:}D, w{:}D) \underset{\langle z,w\rangle}{\overset{\langle z,v\rangle}{\rightrightarrows}} (C, D) \tag{6}$$

We can get its coequalizer by applying bracket types:

$$(z{:}C, v{:}D, w{:}D) \underset{\langle z,w\rangle}{\overset{\langle z,v\rangle}{\rightrightarrows}} (z{:}C, u{:}D) \xrightarrow{\langle z,[u]\rangle} (C, [D]) \tag{7}$$

Indeed, $\langle z, [u]\rangle$ coequalizes the kernel pair by the equality rule for bracket types:

$$\frac{z{:}C, v{:}D, w{:}D \vdash [v] : [D] \qquad z{:}C, v{:}D, w{:}D \vdash [w] : [D]}{z{:}C, v{:}D, w{:}D \vdash [v] = [w] : [D]}$$

To see that it is the coequalizer, suppose we have an arrow

$$z{:}C, u{:}D \vdash t : B$$

that coequalizes the kernel pair, which means that

$$z{:}C, v{:}D, w{:}D \vdash t\{v/u\} = t\{w/u\} : B$$

Then we can form the factorization of t through $[u]$ as the where term

$$z{:}C, y{:}[D] \vdash t \;\mathsf{where}\; [u] = y : B$$

The situation is depicted in the following diagram:

$$\begin{tikzcd}
(z{:}C, v{:}D, w{:}D) \arrow[r, shift left, "\langle z,v\rangle"] \arrow[r, shift right, "\langle z,w\rangle"'] & (z{:}C, u{:}D) \arrow[rr, "t"] \arrow[dr, "\langle z,[u]\rangle"'] & & (B) \\
& & (C, y{:}[D]) \arrow[ur, "(t \;\mathsf{where}\; [u]=y)"'] &
\end{tikzcd}$$

The triangle commutes because

$$t \;\mathsf{where}\; [u] = [u] \quad = \quad t$$

by the β-rule for bracket types. The factorization is unique, because $[-]$ is epic, as we showed in Section 2.

Lastly, let us show that in $\mathcal{S}$ regular epis are stable under pullbacks. Since every arrow in $\mathcal{S}$ is isomorphic to one of the form (5), every kernel pair is isomorphic to one of the form (6) and so every regular epi is isomorphic to one of the form (7). Let us then compute, without loss of generality, a pullback of such a coequalizer along an arrow of the form (5):

$$\begin{tikzcd}
(y{:}B, z{:}C, u{:}D) \arrow[r, "\langle z,u\rangle"] \arrow[d, "\langle y,z,[u]\rangle"'] \arrow[dr, phantom, "\lrcorner", very near start] & (z{:}C, u{:}D) \arrow[d, "\langle z,[u]\rangle"] \\
(y{:}B, z{:}C, v{:}[D]) \arrow[r, "\langle z,v\rangle"'] & (C, [D])
\end{tikzcd}$$

It is evident that the left-hand arrow is a regular epi, because it is of the form (7). It is also clear that the diagram commutes. To see that it is a pullback, observe that it appears as the upper half of the following commutative diagram:

$$
\begin{tikzcd}
(y{:}B, z{:}C, u{:}D) \arrow[r, "\langle z,u\rangle"] \arrow[d, "\langle y,z,[u]\rangle"'] \arrow[dr, phantom, "\lrcorner", very near start] & (z{:}C, u{:}D) \arrow[d, "\langle z,[u]\rangle"] \\
(y{:}B, z{:}C, v{:}[D]) \arrow[r, "\langle z,v\rangle"'] \arrow[d, "\langle y,z\rangle"'] & (z{:}C, [D]) \arrow[d, "z"] \\
(B, z{:}C) \arrow[r, "z"'] & (C)
\end{tikzcd}
$$

The outer rectangle and the lower square are obviously pullbacks, hence the upper square is as well. ■

We can express the regular epi–mono factorization of an arrow $x : A \vdash t : B$ as

$$
\begin{tikzcd}
A \arrow[rr, "t"] \arrow[dr, two heads, "q"'] & & B \\
& \mathsf{Im}(t) \arrow[ur, tail, "m"'] &
\end{tikzcd}
$$

with

$$
\begin{array}{rcl}
\mathsf{Im}(t) & = & \sum_{y:B}\left[\sum_{x:A}\mathsf{Eq}_A(t,y)\right] \\
m & = & z : \mathsf{Im}(t) \vdash \pi_1(z) : B \\
e & = & x : A \vdash \langle t, [\langle x, \mathsf{r}(t)\rangle]\rangle : \mathsf{Im}(t)\ .
\end{array}
$$

4 Properties of Bracket Types

As a consequence of the complete semantics of the previous section we can prove facts about bracket types by arguing in a general regular category. Thus we identify types with objects and terms in contexts with arrows. We use this technique in the present section to establish some of the basic properties of bracket types.

First observe that, in any context Γ, the types satisfying proof irrelevance (1), are exactly the cartesian idempotents:

$$P \text{ prop} \iff P = P \times_\Gamma P \tag{8}$$

where here and in what follows, $=$ between types means that they are canonically isomorphic. For example, in (8) the canonical isomorphisms are the diagonal and the projection. If P and Q are propositions in the context Γ then the corresponding display maps $(\Gamma, P) \to (\Gamma)$ and $(\Gamma, Q) \to (\Gamma)$ are monos, so that we can think of P and Q as subobjects of Γ. In particular, we can write $P \leq Q$ when there exists a (necessarily unique) arrow $P \to Q$ in the slice over Γ.

Proposition 4.1 *For any types A, B in a context Γ:*

1. *$[-]$ is functorial*
2. *There is a canonical arrow $A \to [A]$, natural in A*
3. *$[[A]] = [A]$*
4. *$A = [A] \iff A = A \times_\Gamma A$*
5. *$\mathsf{1} = [\mathsf{1}]$*
6. *$[A \times_\Gamma B] = [A] \times_\Gamma [B]$*

Moreover, (1)–(4) characterize $[-]$ uniquely among stable functors on regular categories.

Proof. The action of $[-]$ on an arrow $x : A \vdash t : B$ is the bottom arrow in the following diagram

$$\begin{tikzcd} x{:}A \arrow[r, "t"] \arrow[d, "{[-]}"'] & B \arrow[d, "{[-]}"] \\ u{:}[A] \arrow[r, "{[t]\ \text{where}\ [x]=u}"'] & {[B]} \end{tikzcd}$$

The action of the *functor* $[-]$ on the arrow t is not to be confused with the arrow $t \circ [-] = [t] : A \to [B]$, which does not even have the correct domain.

The rest of the proposition is proved easily. By way of example, we prove that $[[A]] = [A]$. In the diagram

$$\begin{tikzcd} A \arrow[r] \arrow[d, two heads] & \Gamma \\ {[A]} \arrow[r, two heads] \arrow[ur, tail] & {[[A]]} \arrow[u, tail] \end{tikzcd}$$

the arrow $[A] \to [[A]]$ is the regular epi part of an image factorization of the mono $[A] \to \Gamma$, therefore it is an isomorphism. ∎

Let us see how the adjunction (2) is validated, for any type A and a proposition P, by which we mean that P satisfies (8): given any arrow (term) $A \to P$, we apply the functor $[-]$ to get a unique one $[A] \to [P]$, but $[P] = P$ since P is a proposition. Conversely, given $[A] \to P$, we precompose with $A \to [A]$ to get $A \to P$.

To see how $[-]$ and $\sum$ interact, consider the types $[\sum_A B]$ and $[\sum_A [B]]$ in a context Γ, obtained by applying the $\sum$ and $[-]$ formation rules in two different orders, as indicated in the following diagram.

$$\begin{tikzcd}
(\Gamma, A, B) \arrow[r, two heads] \arrow[d, two heads] \arrow[ddr, "\sum_A B"] & (\Gamma, [\sum_A B]) \arrow[dd, tail, "{[\sum_A B]}"] \\
(\Gamma, A, [B]) \arrow[d, two heads] \arrow[dr, "\sum_A [B]"'] & \\
(\Gamma, [\sum_A [B]]) \arrow[r, tail, "{[\sum_A [B]]}"'] & (\Gamma)
\end{tikzcd}$$

Since the two ways around the diagram from (Γ, A, B) to (Γ) are both regular epi–mono factorizations of the same arrow, by uniqueness of images we have:

$$[\textstyle\sum_A [B]] = [\sum_A B]$$

For equality types Eq_A, the elimination rule,

$$\frac{\Gamma \vdash e : \mathsf{Eq}_A(a, b)}{\Gamma \vdash e = \mathsf{r}(a) : \mathsf{Eq}_A(a, b)}$$

implies that $(\mathsf{Eq}_A(a, b))^2 = \mathsf{Eq}_A(a, b)$, whence:

$$[\mathsf{Eq}_A(a, b)] = \mathsf{Eq}_A(a, b)\ .$$

Together with $[1] = 1$, that summarizes the properties of $[-]$ on its own. Things become more interesting in the presence of other type-forming operations,

$$0,\ A + B,\ \textstyle\prod_A B,\ A \to B,\ \neg A\ ,$$

where $\neg A$ stands for $A \to \mathsf{0}$.

For finite sums we get

$$[\mathsf{0}] = \mathsf{0}\ , \qquad\qquad [A + B] = [[A] + [B]]\ .$$

by an argument similar to that for $\sum$.

For $\prod$ we have $(\prod_A [B])^2 = \prod_A [B]^2 = \prod_A [B]$, so that

$$[\textstyle\prod_A [B]] = \prod_A [B]\ ,$$

and one sees easily that

$$[\textstyle\prod_A B] \leq \prod_A [B]\ . \tag{9}$$

When we specialize this to a function type $A \to B$ we get

$$[A \to [B]] = A \to [B]\ , \qquad\qquad [A \to B] \leq A \to [B]\ . \tag{10}$$

For brackets on the left, it is easy to see that

$$A \to [B] = [A] \to [B] = [[A] \to [B]]\ . \tag{11}$$

Taking $B = \mathsf{0}$ therefore yields the noteworthy

$$\neg A = \neg [A] = [\neg A]\ . \tag{12}$$

Since $\neg A$ is thus always a proposition it is natural to ask whether perhaps:

$$[A] = \neg\neg A \quad ?$$

The answer is in general negative, since there are many simple models in regular lccc's in which double negation closure is not trivial on monos.

5 First Order Logic via Bracket Types

In dependent type theory with the type-forming operations,

$$\mathsf{0},\ \mathsf{1},\ [A],\ A + B,\ \mathsf{Eq}_A,\ \textstyle\sum_{x:A} B,\ \prod_{x:A} B\ ,$$

the propositions in every context model first-order logic, under the following definitions:

$$\begin{array}{rcl} \top & = & \mathsf{1} \\ \bot & = & \mathsf{0} \\ \varphi \wedge \psi & = & \varphi \times \psi \\ \varphi \vee \psi & = & [\varphi + \psi] \\ \varphi \Longrightarrow \psi & = & \varphi \rightarrow \psi \\ \neg\varphi & = & \varphi \rightarrow \mathsf{0} \\ x =_A y & = & \mathsf{Eq}_A(x, y) \\ \forall x{:}A.\, \varphi & = & \prod_{x:A} \varphi \\ \exists x{:}A.\, \varphi & = & [\sum_{x:A} \varphi] \end{array} \tag{13}$$

The bracket is thus used to rectify the operations $+$ and $\sum$ because they lead out of propositions.

Operations defined in (13) satisfy the usual rules for intuitionistic first-order logic, and the resulting system is a dependent type theory with first-order logic over each type. It can be described categorically as the internal logic of a regular lccc with finite sums. The chief difference between this formulation and more customary ones using both type theory and predicate logic is that the first-order logical operations on the propositions are here *defined* in terms of the operations on types, rather than taken as primitive.

In addition to first-order logic, one can use brackets to define *subset types.* For any type $\Gamma, x{:}A \vdash B$ type, the associated subset type

$$\Gamma \vdash \{x : A \mid B\}\ \mathsf{type}$$

is defined by

$$\{x : A \mid B\} = \textstyle\sum_{x:A}[B(x)].$$

These can be compared to the *toolbox for subset types* by Sambin [SV98].

By way of example, we remark that the category Equ of equilogical spaces [Sco96, BBS] is a regular lccc, and so supports all of the logical operations considered above. For $X \in \mathsf{Equ}$, the bracket of an X-indexed family of equilogical spaces

$$(E_x)_{x \in X}$$

is of course the regular epi–mono image factorization of the corresponding display map

$$p : E \rightarrow X\ .$$

The domain of the image $[E] \rightarrowtail X$ is constructed from the same underlying space $|E|$ as $E = (|E|, \sim_E)$, but with the new equivalence relation, given by

$$e \sim_{[E]} e' \iff pe = pe' \; .$$

The fact that Equ has this much internal logic without being a topos, or even a pretopos, was the original observation from which the present work grew [BBS].

6 First Order Logic vs. Propositions-as-Types

As an application of bracket types, we can compare the conventional interpretation of first-order logic with the propositions-as-types interpretation, and relate first-order provability to provability in dependent type theory (without brackets).

Suppose we have a single-sorted first-order theory $\mathbb{T}$, consisting of constants, function and relation letters, and axioms given as closed formulas. The standard propositions-as-types interpretation $*$ of $\mathbb{T}$ into type theory,

$$\mathbb{T} \xrightarrow{\;\;*\;\;} \mathrm{DTT} \tag{14}$$

is determined by fixing the interpretations of the basic sort, the constants, function and relation symbols. The rest of the interpretation is determined inductively in the evident way, using the type-forming operations in place of the corresponding logical ones, cf. [ML98]. For example,

$$(\forall x \exists y . R(x, y) \vee P(x))^* \quad = \quad \prod_{x:I^*} \sum_{y:I^*} (R^*(x, y) + P^*(x)) \; ,$$

where I^* is a new basic type interpreting the domain of individuals I, and the dependent types $x : I^* \vdash P^*(x)$ and $x : I^*, y : I^* \vdash R^*(x, y)$ interpret the relation symbols P and R.

If we add a constant $a : \alpha^*$ for each axiom α, the translation φ^* of a provable closed formula φ becomes inhabited by a term that is obtained from a straightforward translation of the proof of φ into type theory. Thus,

$$\mathrm{IFOL}(\mathbb{T}) \vdash \varphi \quad \text{implies} \quad \mathrm{DTT}(\mathbb{T}) \vdash \varphi^* \; , \tag{15}$$

where by $\mathrm{DTT}(\mathbb{T}) \vdash \varphi^*$ we mean that the type φ^* is inhabited in the dependent type theory enriched with the basic types and constants needed for the translation $*$, and with constants inhabiting the translations of axioms of $\mathbb{T}$.

The question we want to consider is the converse implication: if φ^* is inhabited in DTT($\mathbb{T}$), must φ be provable in the intuitionistic first-order theory $\mathbb{T}$? Note that functions of higher types may be used in a term inhabiting φ^*, so this is not merely a matter of tracing out proofs in first-order logic.

Proofs of partial converses of (15) for different fragments of first-order logic have been given by Martin-Löf ($\forall$, $\Rightarrow$ in [ML98]) and, recently, Tait ($\exists$, $\wedge$, $\forall$, $\Rightarrow$, $\neg$ in [Tai]). These results are for type theory with either no equality types, or intensional equality types, and proceed from normalization. We give a result below that applies to type theory with extensional equality for a large fragment of first-order logic.

Definition 6.1 A first-order formula ϑ is *stable* when it does not contain $\forall$ and $\Rightarrow$, but negation $\neg$ is allowed as a special case of $\Rightarrow$. A first-order formula φ is *left-stable* when in every subformula of the form $\vartheta \Rightarrow \psi$, the formula ϑ is stable.

Theorem 6.2 *If φ is left-stable then*

$$\mathrm{DTT}(\mathbb{T}) \vdash \varphi^* \quad \textit{implies} \quad \mathrm{IFOL}(\mathbb{T}) \vdash \varphi \;.$$

Proof. There is a regular lccc $\mathcal{E}$ with finite coproducts and a conservative, first-order interpretation y of $\mathbb{T}$ into $\mathcal{E}$, schematically:

$$\mathbb{T} \xhookrightarrow{\;y\;} \mathcal{E} \tag{16}$$

A formula φ with free variables $x_1, \ldots, x_n$ is translated into a subobject $y(\varphi) \in \mathsf{Sub}(y(I)^n)$, where $y(I)$ is the interpretation of the sort of individuals. A sentence φ is translated into a subobject $y(\varphi) \in \mathsf{Sub}(1)$ of the terminal object, and the conservativity of y means that

$$y(\varphi) = 1 \quad \text{implies} \quad \mathrm{IFOL}(\mathbb{T}) \vdash \varphi \;. \tag{17}$$

For $\mathcal{E}$ we can take the *first-order classifying topos* for the theory $\mathbb{T}$, in the sense of [BJ98] (sheaves on the syntactic logos generated by $\mathbb{T}$).

Since $\mathcal{E}$ is locally cartesian closed and has finite coproducts, we can interpret dependent type theory with disjoint sums in it. If we compose this interpretation with the translation (14), where we set $I^* = y(I)$, and $R^* = y(R)$, $f^* = y(f)$ for the basic relation and function symbols, then we obtain another interpretation $\star$ of $\mathbb{T}$ into $\mathcal{E}$, schematically:

$$\mathbb{T} \xrightarrow{\;\star\;} \mathcal{E}$$

Clearly if $\mathrm{DTT}(\mathbb{T}) \vdash \varphi^*$, then in $\mathcal{E}$ there exists a point

$$1 \longrightarrow \varphi^\star \tag{18}$$

namely the interpretation of the term inhabiting φ^*.

Because $\mathcal{E}$ is regular, it has bracket types. The composition $[-] \circ \star$ gives an interpretation of $\mathbb{T}$ into the "propositional" logic of $\mathcal{E}$ in each slice $\mathcal{E}/(I^\star)^n$. More precisely, every formula φ with free variables $x_1, \ldots, x_n$ is first interpreted as a type $\varphi^\star$ in the slice $\mathcal{E}/(I^\star)^n$, and then the bracket of that type gives us a subobject of $(I^\star)^n$,

$$[\varphi^\star] \rightarrowtail (I^\star)^n$$

We will prove the theorem by comparing the subobjects $[\varphi^\star]$ and $y(\varphi)$, for which we need the following two lemmas.

Lemma 6.3 *If ϑ is stable, then $[\vartheta^\star] = y(\vartheta)$.*

Proof. This follows from the equations (13) in the previous section, together with the fact that the indicated definitions agree with the first-order operations in any topos, as is easily seen. Specifically, since the two interpretations plainly agree on the atomic formulas and y preserves the first-order operations, we can proceed by straightforward induction. For instance, the case of disjunctions goes as follows:

$$\begin{aligned} [(\varphi \vee \psi)^\star] &= [\varphi^\star + \psi^\star] \\ &= [\varphi^\star] \vee [\psi^\star] \\ &= y(\varphi) \vee y(\psi) \\ &= y(\varphi \vee \psi) \end{aligned}$$

This completes the proof of the lemma.

If we tried to prove the previous lemma for formulas that contain universal quantifiers and implications, we would get stuck because (9) and (10) are only inequalities. Negation works, however, thanks to (12).

Lemma 6.4 *If φ is left-stable, then $[\varphi^\star] \leq y(\varphi)$.*

Proof. As in the previous lemma, we proceed by induction and use equations (13). The stable cases follow from Lemma 6.3. For universal quantifiers we have:

$$\begin{array}{rcll} [(\forall x.\varphi)^\star] & = & [\prod_{x:I^\star} \varphi^\star] & \\ & \leq & \prod_{x:I^\star}[\varphi^\star] & \text{(by (9))} \\ & = & \forall x{:}I^\star.\ [\varphi^\star] & \\ & \leq & \forall x{:}I^\star.\ y(\varphi) & (\varphi \text{ left-stable}) \\ & = & \forall x : y(I).\ y(\varphi) & \\ & = & y(\forall x.\ \varphi) & \end{array}$$

For implication we have:

$$\begin{array}{rcll} [(\vartheta \Rightarrow \psi)^\star] & = & [\vartheta^\star \to \psi^\star] & \\ & \leq & \vartheta^\star \to [\psi^\star] & \text{(by (10))} \\ & = & [\vartheta^\star] \to [\psi^\star] & \text{(by (11))} \\ & = & [\vartheta^\star] \Rightarrow [\psi^\star] & \\ & = & y(\vartheta) \Rightarrow [\psi^\star] & (\vartheta \text{ stable}) \\ & \leq & y(\vartheta) \Rightarrow y(\psi) & (\psi \text{ left-stable}) \\ & = & y(\vartheta \Rightarrow \psi) & \end{array}$$

This concludes the proof of the lemma.

To finish the proof of Theorem 6.2, let φ be a left-stable sentence such that $\mathrm{DTT}(\mathbb{T}) \vdash \varphi^*$. Then, continuing from (18) above, in $\mathcal{E}$ we have maps:

$$1 \to \varphi^\star \to [\varphi^\star] \leq y(\varphi) \leq 1\ .$$

So $y(\varphi) = 1$, and therefore $\mathrm{IFOL}(\mathbb{T}) \vdash \varphi$ by (17). ∎

Observe that every first-order formula φ is *classically* equivalent to one φ^s that is stable. The formula φ^s, which we call the *stabilized* translation of φ, is obtained by replacing in φ every $\forall x.\,\vartheta$ and $\vartheta \Longrightarrow \psi$ by $\neg\exists x.\,\neg\vartheta$ and $\neg(\vartheta \wedge \neg\psi)$, respectively. The equivalence $\varphi \iff \varphi^s$ holds intuitionistically if $\varphi = \psi^{\neg\neg}$ is the double-negation translation of a formula ψ. Therefore, the *stabilized double-negation translation*

$$(\varphi^{\neg\neg})^s$$

takes a formula φ of classical first-order logic (CFOL) to a stable one in IFOL, with the property

$$\mathrm{CFOL} \vdash \varphi \quad \text{if, and only if,} \quad \mathrm{IFOL} \vdash (\varphi^{\neg\neg})^s$$

If we compose the $\neg\neg s$ translation with the propositions-as-types translation $*$, we obtain a translation

$$\varphi^{+} = ((\varphi^{\neg\neg})^{s})^{*}$$

which takes formulas of classical first-order logic into dependent type theory.

Corollary 6.5 *The translation $\varphi \mapsto \varphi^{+}$ of classical first-order logic into dependent type theory has the following property:*

$$\mathrm{CFOL} \vdash \varphi \quad \textit{if, and only if,} \quad \mathrm{DTT} \vdash \varphi^{+}$$

Here $\mathrm{DTT} \vdash \varphi^{+}$ means that the type φ^{+} is inhabited.

Remark 6.6 The following formula was suggested to us by Thierry Coquand:

$$(\forall x\, \exists y.\, R(x,y)) \Longrightarrow$$
$$\forall x, x'\, \exists y, y'.\, (R(x,y) \wedge R(x',y') \wedge (x = x' \Longrightarrow y = y'))\ .$$

It is *not* provable in intuitionistic first-order logic [Min], but its $*$-translation is inhabited in dependent type theory. Theorem 6.2 therefore cannot be extended to full intuitionistic first-order logic.

Remark 6.7 For the special case of *intuitionistic propositional logic* (IPC, with connectives $\top$, $\wedge$, $\Longrightarrow$, $\bot$, $\vee$) completeness with respect to dependent type theory (DTT) it is easily seen to hold for *all* formulas. Briefly, first we use the Curry-Howard correspondence between proofs in IPC and terms in simply-typed λ-calculus with disjoint sums and the empty type (STT) to conclude that

$$\mathrm{IPC} \vdash \varphi \quad \text{if, and only if} \quad \mathrm{STT} \vdash \varphi^{*}\ .$$

Then we use the well-known correspondence between STT and bicartesian closed categories (BiCCC), to conclude that the completeness of IPC with respect to DTT follows from the fact that every BiCCC has a full and faithful BiCCC embedding into a locally cartesian closed category with finite coproducts.

7 Further Topics

There are several additional topics that one might consider in relation to bracket types, some of which we intend to purse in future work:

1. *Intensionality.* We have used extensional equality to determine bracket types, but one could as well consider modified brackets in type theory with intensional identity. Brackets in their current form might also be used in such intensional systems to distinguish those types for which identity is extensional.

2. *Modal operators.* The bracket is a diamond operation, in the sense of modal logic, in a system with dependent types. As such, it is an example of quantified modal logic. One can consider extending the work of Moggi [Mog91] and others on modal type theory to the dependent case. See also [DP00] in this connection.

3. *Interdefinability.* In certain systems of logic, the bracket operation is definable. For instance, in the presence of first-order existential quantifiers, we have

$$[A] \;=\; \exists x{:}A.\,(x = x)\ .$$

   In a topos, we also have the option:

$$[A] \;=\; \textstyle\prod_{p:\Omega}(A \to \{u{:}1 \mid p\}) \to \{u{:}1 \mid p\}$$

   where $\{u{:}1 \mid p\} \rightarrowtail 1$ is the extension of p. A similar trick works in systems of type theory with universes. See [Acz00] for a study of related operations.

4. *Classical type theory.* Consider the rules

$$(a)\quad [A] = \neg\neg A \qquad \text{and} \qquad (b)\quad [\textstyle\prod_A B] = \prod_A [B]\ .$$

   In toposes (a) is Excluded Middle and (b) is the Axiom of Choice, which is strictly stronger. In type theory, therefore, (b) cannot be proved from (a) (consider a permutation model), while the converse inference is plausible, but unverified. See [Awo95, Pal01] for related results.

5. *Alternate formulations.* We can also consider a formulation of bracket types in which we have a new judgment "$\Gamma \vdash P\ \mathsf{prop}$", expressing the

fact that P is a proposition. The rules would then be as follows:

$$\frac{\Gamma \vdash A \ \mathsf{type}}{\Gamma \vdash [A] \ \mathsf{prop}} \qquad \frac{\Gamma \vdash P \ \mathsf{prop}}{\Gamma \vdash P \ \mathsf{type}}$$

$$\frac{\Gamma \vdash a : A}{\Gamma \vdash [a] : [A]} \qquad \frac{\Gamma \vdash q : [A] \qquad \Gamma, x{:}A \vdash p : P \qquad \Gamma \vdash P \ \mathsf{prop}}{\Gamma \vdash p \ \mathsf{where} \ [x] = q : P}$$

$$\frac{\Gamma \vdash p : P \qquad \Gamma \vdash q : P \qquad \Gamma \vdash P \ \mathsf{prop}}{\Gamma \vdash p = q : P}$$

This formulation is better from the type-theoretic point of view because it does not involve an equality judgment as a premise for the elimination rule. As stated so far, however, the rules permit many different interpretations, including the trivial interpretation in which the only proposition is the unit type 1, and $[A] = 1$ for all A. One could additionally assert that certain types are propositions:

$$\frac{}{\Gamma \vdash 0 \ \mathsf{prop}} \qquad \frac{}{\Gamma \vdash 1 \ \mathsf{prop}} \qquad \frac{\Gamma \vdash P \ \mathsf{prop} \quad \Gamma \vdash Q \ \mathsf{prop}}{\Gamma \vdash P \times Q \ \mathsf{prop}}$$

$$\frac{\Gamma \vdash P \ \mathsf{prop} \quad \Gamma \vdash Q \ \mathsf{prop}}{\Gamma \vdash P \to Q \ \mathsf{prop}} \qquad \frac{\Gamma, x{:}A \vdash P \ \mathsf{prop}}{\Gamma \vdash \prod_{x:A} P \ \mathsf{prop}} \qquad \frac{\Gamma \vdash s : A \quad \Gamma \vdash t : A}{\Gamma \vdash \mathsf{Eq}_A(s, t) \ \mathsf{prop}}$$

This still leaves room for alternative interpretations. For example, nothing prevents interpreting propositions as *regular monos* and the bracket types as the epi–regular mono factorizations.

6. *Normalization.* It seems likely that the techniques of Maietti [Mai98] will succeed to prove normalization for terms in type theory with brackets. Both cases of intensional and extensional equality need to be investigated.

7. *Type models.* Type-theoretic models of bracket types could be built from setoids in type theory. A simpler construction is suggested by the regular completion of a left-exact category; namely, take only the *definable setoids*, which are the setoids whose equivalence relations are of the form $\mathsf{Eq}_A(a, b)$ for some type A and terms $a, b : A$.

A Dependent Sums and Equality Types

For completeness, we list the rules for dependent type theory with the unit type, strong dependent sums and strong extensional equality, cf. [Jac99].

Formation rules:

$$\frac{}{\Gamma \vdash \mathsf{1}\ \mathsf{type}} \qquad \frac{\Gamma, x{:}A \vdash B\ \mathsf{type}}{\Gamma \vdash \sum_{x:A} B\ \mathsf{type}} \qquad \frac{\Gamma \vdash A\ \mathsf{type}}{\Gamma, x{:}A, y{:}A \vdash \mathsf{Eq}_A(x,y)\ \mathsf{type}}$$

Introduction and elimination rules:

$$\frac{}{\Gamma \vdash \star : \mathsf{1}}$$

$$\frac{\Gamma \vdash a : A \qquad \Gamma, x{:}A \vdash B\ \mathsf{type} \qquad \Gamma \vdash b : B\{a/x\}}{\Gamma \vdash \langle a, b \rangle : \sum_{x:A} B}$$

$$\frac{\Gamma \vdash p : \sum_{x:A} B}{\Gamma \vdash \pi_1(p) : A} \qquad \frac{\Gamma \vdash p : \sum_{x:A} B}{\Gamma \vdash \pi_2(p) : B\{\pi_1(p)/x\}}$$

$$\frac{\Gamma \vdash t : A}{\Gamma \vdash \mathsf{r}(t) : \mathsf{Eq}_A(t,t)}$$

In $\sum_{x:A} B$, variable x is bound in A. Equality rules:

$$\frac{\Gamma \vdash t : \mathsf{1}}{\Gamma \vdash t = \star : \mathsf{1}} \qquad \frac{\Gamma \vdash e : \mathsf{Eq}_A(s,t))}{\Gamma \vdash s = t : A} \qquad \frac{\Gamma \vdash e : \mathsf{Eq}_A(s,t)}{\Gamma \vdash e = \mathsf{r}(s) : \mathsf{Eq}_A(s,t)}$$

Conversions:

$$\begin{aligned} \pi_1(\langle a, b \rangle) &= a \\ \pi_2(\langle a, b \rangle) &= b \\ \langle \pi_1(p), \pi_2(p) \rangle &= p \end{aligned}$$

References

[Acz00] P. Aczel. The Russell-Prawitz modality. Technical Report UMCS-00-12-1, Department of Computer Science, University of Manchester, 2000.

[AG01] P. Aczel and N. Gambino. Collection principles in dependent type theory. Draft manuscript, 2001.

[Awo95] S. Awodey. Axiom of choice and excluded middle in categorical logic. *Bulletin of Symbolic Logic*, 1:344, 1995. Abstract of unpublished manuscript.

[BBS] A. Bauer, L. Birkedal, and D.S. Scott. Equilogical spaces. *Theoretical Computer Science*. To appear.

[BJ98] C. Butz and P.T. Johnstone. Classifying toposes for first-order theories. *Annals of Pure and Applied Logic*, 91, 1998.

[Bor94] F. Borceux. *Handbook of Categorical Algebra*, volume 2. Cambridge University Press, 1994.

[DP00] R. Davies and F. Pfenning. A modal analysis of staged computation. *Journal of the ACM*, 2000. Significantly extended and revised version of a technical summary from POPL'96.

[Hof95] M. Hofmann. On the interpretation of type theory in locally cartesian closed categories. In L. Pacholski and J. Tiuryn, editors, *Computer Science Logic 1994*, volume 806 of *Lecture Notes in Computer Science*. Springer, 1995.

[How80] W. A. Howard. The formulae-as-types notion of construction. In J. R. Seldin and J. P. Hindley, editors, *To H. B. Curry: Essays on Combinaory Logic, Lambda Calculus, and Formalism.* Academic Press, 1980.

[Jac99] B. Jacobs. *Categorical Logic and Type Theory.* Elsevier Science, 1999.

[Law69] F. W. Lawvere. Adjointness in foundations. *Dialectica*, 1969.

[Mai98] M.E. Maietti. *The Type Theory of Categorical Universes.* PhD thesis, Università Delgi Studi di Padova, 1998.

[Min] G. Mints. Axiomatization of a skolem function in intuitionistic logic. Unpublished note.

[ML84] P. Martin-Löf. *Intuitionistic Type Theory. Notes by Giovanni Sambin of a series of lectures given in Padua, June 1980.* Bibliopolis, Napoli, 1984.

[ML98] P. Martin-Löf. An intuitionistic theory of types. In G. Sambin and J. Smith, editors, *Twenty-Five Years of Constructive Type Theory.* Oxford University Press, August 1998. Proceedings of a Congress held in Venice, Italy, October 1995.

[Mog91] E. Moggi. Notions of computation and monads. *Information and Computation*, 93(1), 1991.

[Pal01] E. Palmgren. A categorical version of the BHK-interpretation. Technical report, Institut Mittag-Leffler, The Royal Swedish Academy of Sciences, 2001.

[Pfe01] F. Pfenning. Intensionality, extensionality, and proof irrelevance in modal type theory. In *Proceedings of the 16th Annual Symposium on Logic in Computer Science (LICS'01)*, June 2001. To appear.

[Sco96] D.S. Scott. A new category? Unpublished Manuscript. Available at `http://www.cs.cmu.edu/Groups/LTC/`, December 1996.

[SV98] G. Sambin and S. Valentini. Building up a toolbox for Martin-Löf's type theory: Subset theory. In G. Sambin and J. Smith, editors, *Twenty-Five Years of Constructive Type Theory.* Oxford University Press, August 1998. Proceedings of a Congress held in Venice, Italy, October 1995.

[Tai] W. W. Tait. The completeness of intuitionistic first-order logic. Unpublished manuscript.