Type Refinements for the Working Class

Jon Sterling and Darin Morrison

There are two conflicting views which bedevil any discussion of the nature of type theory. First, there is the notion of type theory as an extension or generalization of universal algebra to support interdependency of sorts and operations, possibly subject to an arbitrary equational theory [1, 3]; we will call this *formal type theory*. Typing, in such a setting, is a mere matter of grammar and is nearly always decidable. In hindsight, we may observe that this is the sort of type theory which Martin-Löf first proposed in 1972 [11], even if we will admit that this was not the intention at the time. A model for such a type theory is usually given by interpreting the types or sorts as presheaves or sheaves over contexts of hypotheses, and as such, a proof theoretic interpretation of the hypothetical judgment is inevitable.

Secondly, there is the view of type theory as semi-formal theory of constructions for the Brouwer-Heyting-Kolmogorov interpretation of intuitionistic mathematical language, which we will call *behavioral* or *semantic type theory*. The most widely known development of this program is Martin-Löf's 1979 "extensional" type theory [12, 14], but we must give priority to Dana Scott for inventing this line of research in 1970 with his prophetic report, *Constructive Validity* [15]. Since the 1980s, behavioral type theory has been developed much further in the Nuprl family [2] of proof assistants, including MetaPRL [7] and JonPRL [16].

Martin-Löf's key innovation was the commitment to pervasive functionality (extensionality) as part of the *definitions* of the judgments and the types, in contrast to the state of affairs in formal type theory where functionality is a metatheorem which must be shown to obtain, based on the (somewhat arbitrary) equational theory which has been imposed. Furthermore, models for behavioral type theory interpret the types as partial equivalence relations on only closed terms, and the meaning of the hypothetical judgment is defined separately and uniformly in the logical relations style.

Our position is that these views of type theory are not in conflict, but rather merely describe two distinct layers in a single, harmonious system. From this perspective, formal type theories can do little more than negotiate matters of grammar, and therefore may serve as a syntactic (linguistic) framework for mathematical language, being responsible for the management of variable binding and substitution. On the other hand, the meaning of mathematical statements shall be specified *behaviorally* in the semantic type theory.

The types of the semantic theory can then be said to *refine* the types of the syntactic theory [5, 6, 4], both by placing restrictions on membership and by coarsening equivalence.

Thus far, all developments of behavioral type theory have been built on a *unityped* syntactic framework, and so the relation to type refinements has been difficult to see. In this paper, we contribute a full theory of behavioral refinements over multi-sorted abstract binding trees, a simple formal type theory [4, 17]; this hybrid system allows the deployment of a Nuprl-style type theory over any signature of sorts and operators.

1 Abstract Binding Trees and Symbols

See [17] for the development of abstract binding trees with symbols.

give a brief description of the framework, and present its rules.

2 The Ambient Logical Framework

In this paper, we hint at the *modes* of judgments and assertions [4] using colors, marking inputs with `blue` and outputs with `red`. As a rule of thumb, inputs are things which are supplied when checking the correctness of a judgment, and outputs are things which are synthesized in the process.

For any abt signature $\Sigma \equiv \langle \mathcal{S}, \mathcal{O} \rangle$, we can deploy a generic logical framework equipped with higher-order judgments. A judgment is always defined using a *meaning explanation*, which is a specification of the conditions under which it may be asserted (sc. its evidence) [13].

Renaming convention Because the evidence of a judgment will often contain symbols $u, v, \ldots$ we informally adopt the convention that the evidence of a judgment shall always be subject to fresh renamings of its free symbols (i.e. the validity of a judgment shall not depend on choice of names). This apparatus is developed formally in Appendix A, where the logical framework is cast as a topos of covariant presheaves on symbol contexts **SCtx**.

Definition 2.1 (Hypothetical Judgment)**.** For two judgments $\mathcal{J}_1$ and $\mathcal{J}_2$, we can form the hypothetical judgment $\mathcal{J}_2 \quad (\mathcal{J}_1)$, whose evidence shall be an effective transformation of any evidence for $\mathcal{J}_1$ into evidence for $\mathcal{J}_2$. It is important to keep in mind that this is a *semantic* consequence and therefore expresses admissibilities, in contrast to the *logical* consequence which expresses derivabilities [4]. We will not have any need for the latter.

Notation For a hypothetical judgment with multiple hypotheses, we will write $\mathcal{J}_3 \quad (\mathcal{J}_1, \mathcal{J}_2)$ rather than $\mathcal{J}_3 \quad (\mathcal{J}_1 \times \mathcal{J}_2)$ or $(\mathcal{J}_3 \quad (\mathcal{J}_2)) \quad (\mathcal{J}_1)$.

Definition 2.2 (General Judgment)**.** For a metavariable context Θ *mctx*, we can define the set of its substitutions as follows:

$$\Box\Theta \triangleq \prod_{(\mathfrak{m}:v)\in\Theta} \{E \mid \cdot \rhd \cdot \parallel \cdot \vdash E : v\}$$

For any $\Box\Theta$-indexed family of judgments $\mathcal{J}$, we can express the general judgment $|_{\Theta}\ \mathcal{J}$ as being evident when $\mathcal{J}(\vec{E})$ is evident for each $\vec{E} : \Box\Theta$; in other words, the evidence of the general judgment is an effective transformation of Θ-substitutions to their fibres in $\mathcal{J}$. Recall that metavariables have *valences* rather than sorts; as such, the general judgment allows us to quantify over terms of higher type.

Notation We will write $|_{\Theta}^{\Upsilon\parallel\Gamma}\ \mathcal{J}$ for $|_{\Theta'}\ \mathcal{J}$, where $\Theta' \ni \mathfrak{m} : \{\Upsilon, \vec{\sigma}\}[\Gamma, \vec{\tau}].\,\tau$ if $\Theta \ni \mathfrak{m} : \{\vec{\sigma}\}[\vec{\tau}].\,\tau$. Additionally, for clarity we will often write $|_{\mathfrak{m}:\sigma}\ \mathcal{J}$ rather than $|_{\mathfrak{m}:\{\cdot\}[\cdot].\,\sigma}\ \mathcal{J}$; likewise, in the body of $\mathcal{J}$, when there are contextually salient $\vec{u}$, $\vec{x}$, we will simply write $\mathfrak{m}$ rather than the more proper $\mathfrak{m}\{\vec{u}\}[\vec{x}]$.

After this section, we will define a judgment either by rules, or by an informal semantical explanation, in both cases leaving its intensional character implicit. A judgment is said to be *correct* or *valid* just in case it is globally inhabited.

3 Behavioral Refinements

Fixing a signature $\Sigma \equiv \langle \mathcal{S}, \mathcal{O} \rangle$ in the abt framework, we will define the notion of behavioral refinement by propounding several judgments and their semantical explanations.

3.1 Parametric Refinement

We will require two judgments, defined mutually: $\Phi \sqsubset^{\star} \Upsilon$, which means that Φ is a symbol context which refines the symbol context Υ (presupposing Υ *sctx*), and parametric refinement $\Phi \parallel \phi \sqsubset \tau$ (which shall presuppose $\Phi \sqsubset^{\star} \Upsilon$ and τ *sort*).

The first judgment we define inductively:

$$\frac{}{\cdot \sqsubset^{\star} \cdot}\ \sqsubset^{\star}_{\mathtt{nil}} \qquad \frac{\Phi \sqsubset^{\star} \Upsilon \quad \Phi \parallel \phi \sqsubset \tau}{\Phi, u : \phi \sqsubset^{\star} \Upsilon, u : \tau}\ \sqsubset^{\star}_{\mathtt{snoc}}$$

The second judgment is defined coinductively via the following meaning explanation:

Definition 3.1 (Parametric Refinement)**.** To know $\Phi \parallel \phi \sqsubset \tau$ (presupposing $\Phi \sqsubset^{\star} \Upsilon$ and τ *sort*) is to know, for any renaming $\rho : \Upsilon \hookrightarrow \Upsilon'$, and for any M, N, what it means for M and N to be identified under ϕ, supposing $\cdot \rhd \Upsilon' \parallel \cdot \vdash M : \tau$ and $\cdot \rhd \Upsilon' \parallel \cdot \vdash N : \tau$, requiring that if $M \sim_{\phi}^{\rho} N$ then $N \sim_{\phi}^{\rho} M$, and if $M \sim_{\phi}^{\rho} N$ and $N \sim_{\phi}^{\rho} O$, then $M \sim_{\phi}^{\rho} O$.

In other words, to know $\Phi \parallel \phi \sqsubset \tau$ (presupposing $\Phi \sqsubset^\star \Upsilon$) is to know, for any renaming of Υ to Υ', which partial equivalence relation on the closed τ-sorted Υ'-terms ϕ denotes. At this point, it should be remarked that this is very similar to the semantical explanation of typehood given in [12], except that we have generalized it to a multi-sorted setting, and that we have fibred the entire apparatus over collections Υ of symbols.

define maps of refined contexts, then change weakening lemma to monotonicity, which is stronger and more useful.

Theorem 3.2 (Weakening). *If* $\Phi \parallel \phi \sqsubset \tau$, *then* $\Phi, u : \psi \parallel \phi \sqsubset \tau$ *(supposing* $u \notin |\Phi|$ *and* $\Phi \parallel \psi \sqsubset \sigma$*).*

Proof. This follows trivially from the rule $\sqsubset^\star_{\mathsf{snoc}}$ and the canonical renaming $\Upsilon \hookrightarrow \Upsilon, u : \sigma$. □

When we have $\cdot \parallel \phi \sqsubset \tau$, we say that ϕ *globally* refines τ and may simply write $\phi \sqsubset \tau$. By weakening, if $\phi \sqsubset \tau$ then for all Φ, we have $\Phi \parallel \phi \sqsubset \tau$ supposing $\Phi \sqsubset^\star \Upsilon$.

3.1.1 Ordering and Equality of Parametric Refinement

We will write $\Phi \parallel \phi \subseteq \psi \sqsubset \tau$ to mean that ϕ is a *subrefinement* of ψ, presupposing $\Phi \sqsubset^\star \Upsilon$, $\Upsilon \parallel \phi \sqsubset \tau$ and $\Upsilon \parallel \psi \sqsubset \tau$:

Definition 3.3. To know $\Phi \parallel \phi \subseteq \psi \sqsubset \tau$ is to know, for any substitution $\rho : \Upsilon \hookrightarrow \Upsilon'$, the following judgments:

$$|_{m,n:\tau}\ M \sim^{\rho}_{\psi} N \quad \left(M \sim^{\rho}_{\phi} N\right)$$

Two refinements are equal when they denote the same PER. That is, we have $\Phi \parallel \phi = \psi \sqsubset \tau$ just when both $\Phi \parallel \phi \subseteq \psi \sqsubset \tau$ and $\Phi \parallel \psi \subseteq \phi \sqsubset \tau$.

3.2 Parametric Equality

The primary judgment concerning refinements is the parametric equality, $\Phi \parallel M = N \in \phi$, which presupposes $\Phi \sqsubset^\star \Upsilon$, $\Phi \parallel \phi \sqsubset \tau$, $\cdot \rhd \Upsilon \parallel \cdot \vdash M : \tau$ and $\cdot \rhd \Upsilon \parallel \cdot \vdash N : \tau$. The meaning of this judgment is that M and N are identified by ϕ at the identity renaming $1 : \Upsilon \hookrightarrow \Upsilon$:

$$\frac{M \sim^{1}_{\phi} N}{\Phi \parallel M = N \in \phi}$$

3.3 Functional Refinement

Next, we define functional refinement $\Phi \parallel \Psi \vDash \phi = \psi \sqsubset \tau$ in terms of parametric refinement, simultaneously with parametric context refinement $\Phi \parallel \Psi \sqsubset^\star \Gamma$ and functional equality $\Phi \parallel \Psi \vDash M = N \in \phi$. We will write $\Phi \parallel \Psi \vDash \phi \sqsubset \tau$ as a shorthand for $\Phi \parallel \Psi \vDash \phi = \phi \sqsubset \tau$.

Parametric context refinement $\Phi \parallel \Psi \sqsubset^\star \Gamma$ shall be defined as follows, presupposing $\Phi \sqsubset^\star \Upsilon$ and Γ *vctx*:

$$\frac{}{\Phi \parallel \cdot \sqsubset^\star \cdot}\ {\parallel}{\sqsubset}^\star_{\mathtt{nil}} \qquad \frac{\Phi \parallel \Psi \sqsubset^\star \Gamma \quad \Phi \parallel \Psi \vDash \phi \sqsubset \tau}{\Phi \parallel \Psi, x : \phi \sqsubset^\star \Gamma, x : \tau}\ {\parallel}{\sqsubset}^\star_{\mathtt{snoc}}$$

Functional refinement $\Phi \parallel \Psi \vDash \phi = \psi \sqsubset \tau$ presupposes $\Phi \parallel \Psi \sqsubset^\star \Gamma$ and τ *sort*, and is explained by induction on the evidence for its first presupposition.

Case $\frac{}{\Phi \parallel \cdot \sqsubset^\star \cdot}\ {\parallel}{\sqsubset}^\star_{\mathtt{nil}}$.

$$\frac{\Phi \parallel \phi = \psi \sqsubset \tau}{\Phi \parallel \cdot \vDash \phi = \psi \sqsubset \tau}$$

Case $\frac{\Phi \parallel \Psi \sqsubset^\star \Gamma \quad \Phi \parallel \Psi \vDash \chi \sqsubset \sigma}{\Phi \parallel \Psi, x : \chi \sqsubset^\star \Gamma, x : \sigma}\ {\parallel}{\sqsubset}^\star_{\mathtt{snoc}}$, where $\Phi \sqsubset^\star \Upsilon$.

$$\frac{\left|^{\Upsilon \parallel \Gamma}_{m,n:\sigma}\right. \Phi \parallel \Psi \vDash [m / x]\,\phi = [n / x]\,\psi \sqsubset \tau \quad (\Phi \parallel \Psi \vDash m = n \in \chi)}{\Phi \parallel \Psi, x : \chi \vDash \phi = \psi \sqsubset \tau}$$

3.4 Functional Equality

Functional equality $\Phi \parallel \Psi \vDash M = N \in \phi$ presupposes $\Phi \sqsubset^\star \Upsilon$, $\Phi \parallel \Psi \sqsubset^\star \Gamma$, $\Phi \parallel \Psi \vDash \phi \sqsubset \tau$, $\cdot \rhd \Upsilon \parallel \Gamma \vdash M : \tau$ and $\cdot \rhd \Upsilon \parallel \Gamma \vdash N : \tau$; the judgment is defined by induction on the evidence for the second presupposition $\Phi \parallel \Psi \sqsubset^\star \Gamma$:

Case $\frac{}{\Phi \parallel \cdot \sqsubset^\star \cdot}\ {\parallel}{\sqsubset}^\star_{\mathtt{nil}}$.

$$\frac{\Phi \parallel M = N \in \phi}{\Phi \parallel \cdot \vDash M = N \in \phi}$$

Case $\frac{\Phi \parallel \Psi \sqsubset^\star \Gamma \quad \Phi \parallel \Psi \vDash \psi \sqsubset \sigma}{\Phi \parallel \Psi, x : \psi \sqsubset^\star \Gamma, x : \sigma}\ {\parallel}{\sqsubset}^\star_{\mathtt{snoc}}$, where $\Phi \sqsubset^\star \Upsilon$.

$$\frac{\left|^{\Upsilon \parallel \Gamma}_{m,n:\sigma}\right. \Phi \parallel \Psi \vDash [m / x]\,\phi = [n / x]\,\psi \in \phi \quad (\Phi \parallel \Psi \vDash m = n \in \psi)}{\Phi \parallel \Psi, x : \psi \vDash M = N \in \phi}$$

4 Constructions on Refinements

Refinements can be arranged in a partial order via the $\Phi \parallel \phi \subseteq \psi \sqsubset \tau$ judgment, and admit a lattice structure at any sort τ.

We have the bottom refinement $\bot_\tau \sqsubset \tau$, which simply contains no elements. Next, the top refinement $\top_\tau \sqsubset \tau$ identifies all terms M, N such that $\cdot \rhd \Upsilon \parallel \cdot \vdash M : \tau$ and $\cdot \rhd \Upsilon \parallel \cdot \vdash N : \tau$. The join $\Phi \parallel \phi \cup_\tau \psi \sqsubset \tau$ of two refinements $\Phi \parallel \phi \sqsubset \tau$ and $\Phi \parallel \psi \sqsubset \tau$ is defined as follows:

$$\frac{\Phi \parallel M = N \in \phi}{\Phi \parallel M = N \in \phi \cup_\tau \psi} \qquad \frac{\Phi \parallel M = N \in \psi}{\Phi \parallel M = N \in \phi \cup_\tau \psi}$$

Finally the meet $\Phi \parallel \phi \cap_\tau \psi \sqsubset \tau$ of two refinements $\Phi \parallel \phi \sqsubset \tau$ and $\Phi \parallel \psi \sqsubset \tau$ is defined dually:

$$\frac{\Phi \parallel M = N \in \phi \quad \Phi \parallel M = N \in \psi}{\Phi \parallel M = N \in \phi \cap_\tau \psi}$$

It is trivial to verify that these constructions on refinements give rise to a lattice.

5 Multi-Sorted Computational Type Theory

Martin-Löf's type theory is a theory of *value types*: to define a type, you specify how to form its equal canonical inhabitants, or values. Then, the inhabitants of the types are explained uniformly by extending this relation over an operational semantics such that two terms are equal members of a type just when they compute to equal values of that type.

More generally, one may wish to consider types which include computations in addition to values; an example of such a type would be $\bar{\psi}$ when ψ is a type, which would include all the members of ψ as well as a divergent computation. Another example is the type $\top_\tau$ which identifies all closed terms of sort τ.

In order to ensure that such types are sensible, we must formulate a notion of computational equivalence, and then say that a refinement is a *type* just when it respects computational equivalence; then, we are able to define types as PERs on *programs* in addition to values.

5.1 Lazy Computation Systems

In this section we generalize Howe's notion of lazy computation system [8] to the multi-sorted, symbol-parameterized setting. An *lazy computation language* is an abt signature $\Sigma \equiv \langle \mathcal{S}, \mathcal{O} \rangle$ along with a distinguished copresheaf $\mathcal{K} : \mathbf{Set}^{\mathbf{SCtx} \times \mathcal{A}_\equiv}$ of *canonical* operators such that $\mathcal{K} \subseteq \mathcal{O}$. For a lazy computation language $L \equiv \langle \mathcal{S}, \mathcal{O}, \mathcal{K} \rangle$, let us define the covariant presheaves on $\mathbf{SCtx} \times \mathbf{Ctx}$ of open expressions, open values, and open bound terms as follows:

$$\begin{aligned}
\mathbf{E}_\tau(\Upsilon \parallel \Gamma) &\triangleq \{M \mid \cdot \rhd \Upsilon \parallel \Gamma \vdash M : \tau\} \\
\mathbf{V}_\tau(\Upsilon \parallel \Gamma) &\triangleq \left\{M \equiv \vartheta(\vec{E}) \mid M \in \mathbf{E}_\tau(\Upsilon \parallel \Gamma) \land \exists a.\mathcal{K}\langle \Upsilon, a\rangle \ni \vartheta\right\} \\
\mathbf{B}_\nu(\Upsilon \parallel \Gamma) &\triangleq \{E \mid \cdot \rhd \Upsilon \parallel \Gamma \vdash E : \nu\}
\end{aligned}$$

We'll write $\mathbf{E}_\tau(\Upsilon)$, $\mathbf{V}_\tau(\Upsilon)$ and $\mathbf{B}_\nu(\Upsilon)$ for $\mathbf{E}_\tau(\Upsilon \parallel \cdot)$, $\mathbf{V}_\tau(\Upsilon \parallel \cdot)$ and $\mathbf{B}_\nu(\Upsilon \parallel \cdot)$ respectively, viewed as covariant presheaves on **SCtx**. Then, a *lazy computation system* (lcs) is a lazy computation language $L \equiv \langle \Sigma, \mathcal{K}\rangle$ along with an **SCtx**-indexed $\equiv_\alpha$-functional evaluation relation $\Upsilon \parallel M \Downarrow_\tau^n N$ presupposing $n \in \mathbb{N}$, $M \in \mathbf{E}_\tau(\Upsilon)$ and $N \in \mathbf{V}_\tau(\Upsilon)$, expressing that M evaluates to N in n steps. We will write $\Upsilon \parallel M \Downarrow_\tau N$ to mean that there exists an n such that $\Upsilon \parallel M \Downarrow_\tau^n N$.

Remark 5.1. In any topos $\mathcal{E}$, for an object X we can define the object of relations on X as the exponential $\wp(X) \triangleq \Omega^X$. Then, the data of such a relation is contained in a natural transformation $R : \mathcal{E}[\mathbb{1}, \wp(X)]$.

Fix a sort-indexed family of binary relations on closed expressions $R_\tau : \mathbf{Set}^{\mathbf{SCtx}}\left[\mathbb{1}, \wp\left(\mathbf{E}_\tau^2\right)\right]$; we will also write the relation as a judgment scheme $\Upsilon \parallel M\ R_\tau\ N$. Now, we can always extend R_τ to a new family of relations $R_\nu : \mathbf{Set}^{\mathbf{SCtx}}\left[\mathbb{1}, \wp\left(\mathbf{B}_\nu^2\right)\right]$ for any valence $\nu \equiv \{\vec{\sigma}\}[\vec{\tau}].\tau$, defined pointwise as follows:

$$\frac{\begin{array}{l}\forall \vec{w} \mathrel{\#} |\Upsilon| \cup \vec{u} \cup \vec{v}. \quad \forall \vec{X} : \mathbf{E}_{\vec{\tau}}^{[\vec{\tau}]}(\Upsilon, \vec{w} : \vec{\sigma}). \\ \Upsilon, \vec{w} : \vec{\sigma} \parallel \left[\vec{X} / \vec{x}\right]\{\vec{w} / \vec{u}\} M\ R_\tau\ \left[\vec{X} / \vec{y}\right]\{\vec{w} / \vec{v}\} N\end{array}}{\Upsilon \parallel \lambda\{\vec{u}\}[\vec{x}].\,M\ R_{\{\vec{\sigma}\}[\vec{\tau}].\,\tau}\ \lambda\{\vec{v}\}[\vec{y}].\,N}$$

As a matter of convenience, we'll also define this judgment over vectors of bound terms and valences:

$$\frac{\forall (E, F, \nu) \in (\vec{E}, \vec{F}, \vec{\nu}).\ \Upsilon \parallel E\ R_\nu\ F}{\Upsilon \parallel \vec{E}\ R_{\vec{\nu}}\ \vec{F}}$$

Now, we can extend R_τ to a new relation $[R_\tau]$ on closed expressions which respects a single "layer" of computation, defined pointwise in the following way:

$$\frac{\left(\Upsilon \parallel N \Downarrow_\tau \vartheta(\vec{F}) \quad \Upsilon \parallel \vec{E}\ R_{\vec{\nu}}\ \vec{F}\right) \quad \text{supposing} \quad \left(\Upsilon \parallel M \Downarrow_\tau \vartheta(\vec{E}), \mathcal{K} \ni \vartheta : (\vec{\nu})\,\tau\right)}{\Upsilon \parallel M\ [R_\tau]\ N}$$

Definition 5.2 (Computational Approximation)**.** Because $[-_\tau]$ is monotonic, it has a greatest fixed point, which we will call *computational approximation*, $\preccurlyeq_\tau : \mathbf{Set}^{\mathbf{SCtx}}\left[\mathbb{1}, \wp\left(\mathbf{E}_\tau^2\right)\right]$, written pointwise as $\Upsilon \parallel M \preccurlyeq_\tau N$.

Definition 5.3 (Computational Equivalence)**.** Two terms are said to be *computationally equivalent* when they approximate each other:

$$\frac{\Upsilon \parallel M \preccurlyeq_{\tau} N \quad \Upsilon \parallel N \preccurlyeq_{\tau} M}{\Upsilon \parallel M \sim_{\tau} N}$$

Appendix A Formal Definition of the Logical Framework

As noted earlier, we must be cautious in the definitions of both atomic and higher-order judgments that their evidence shall always respect the *renaming convention*. We do so explicitly in this appendix by developing our logical framework as a presheaf topos over symbol contexts.

A judgment is, then, an object in the presheaf category $\mathbb{J} \triangleq \mathbf{Set}^{\mathbf{SCtx}}$; in other words, judgments are identified with the intensional set of its renameable evidence. $\mathbb{J}$, like all presheaf categories, is a topos. The task at hand, then, is to show how to replace the informal meaning explanations that we gave in section 2 with precise definitions in the topos $\mathbb{J}$.

A.1 Hypothetical Judgment

Hypothetical judgment $\mathcal{J}_2\ \ (\mathcal{J}_1)$ is defined using the exponential construction [10, p. 46], which expresses the semantic consequence of $\mathcal{J}_2$ from $\mathcal{J}_1$:

$$\begin{aligned}(\mathcal{J}_2\ \ (\mathcal{J}_1))\,(\Upsilon) &\triangleq \left(\mathcal{J}_2{}^{\mathcal{J}_1}\right)(\Upsilon)\\ &\cong \mathbb{J}[\mathbf{y}(\Upsilon) \times \mathcal{J}_1, \mathcal{J}_2]\\ &\cong \int_{\Upsilon'} \Upsilon \hookrightarrow \Upsilon' \Rightarrow \mathcal{J}_2(\Upsilon')^{\mathcal{J}_1(\Upsilon')}\end{aligned}$$

In other words, the hypothetical judgment $\mathcal{J}_2\ \ (\mathcal{J}_1)$ is evident when there shall "forevermore" (i.e. for any applicable renaming of symbols) be an effective transformation of evidence for the antecedent into evidence for the consequent.

A.2 General Judgment

For any diagram $\mathcal{J} : \mathbb{J}^{\Box\Theta}$, we can define the general judgment $|_{\Theta}\ \mathcal{J}$ as follows:

$$\begin{aligned}(|_{\Theta}\ \mathcal{J})\,(\Upsilon) &\triangleq \mathbb{J}\left[\mathbf{y}(\Upsilon), \varprojlim \mathcal{J}\right]\\ &\cong \int_{(\Upsilon', \vec{E})} \Upsilon \hookrightarrow \Upsilon' \Rightarrow \mathcal{J}(\vec{E})(\Upsilon')\end{aligned}$$

Intuitively, the general judgment $|_{\Theta}\ \mathcal{J}$ is evident when the family of judgments $\mathcal{J}$ shall forevermore have a global section (i.e. for any substitution $\vec{E} \in \Box\Theta$, $\mathcal{J}(\vec{E})$ shall be evident).

A.3 Internal Judgments

It often happens that the evidence for a judgment is the knowledge of the meaning of a new form of judgment; this is the case when we say "To know $\mathcal{J}$ is to know *the meaning of* another judgment $\mathcal{J}'$." This can be neatly expressed in our topos using the subobject classifier Ω, where the evidence of Ω at Υ gives rise to an *internal judgment* which is defined at all Υ' for which we have $\rho : \Upsilon \hookrightarrow \Upsilon'$.

A *global internal (family of) judgments* is a section $J : \int_{\mathbf{SCtx}} \Omega^{\mathcal{J}} \cong \int_{\Upsilon} \mathbb{J}[\mathbf{y}(\Upsilon) \times \mathcal{J}, \Omega]$, which can be externalized as a "total" judgment $\lfloor J \rfloor : \mathbb{J}$ for whose derivations at Υ are the arrows $\phi : \mathbb{J}[\mathbf{y}(\Upsilon), \mathbb{1} \times \mathcal{J}]$ such that the following diagram commutes:

$$\begin{tikzcd}
\mathbf{y}(\Upsilon) \arrow[r, dashed, "\phi"] \arrow[dr, "J_{\Upsilon}\langle 1, \pi_2 \circ \phi \rangle"'] & \mathbb{1} \times \mathcal{J} \arrow[d, "\mathbf{true} \circ \pi_1"] \\
& \Omega
\end{tikzcd}$$

Then, to say $\Upsilon \parallel \lfloor J \rfloor (E)$ where $E \in \mathcal{J}(\Upsilon)$ is to assert that we have evidence $F \in \lfloor J \rfloor (\Upsilon)$ such that $\pi_2(F) = E$.

References

[1] J. Cartmell. Generalised algebraic theories and contextual categories. *Annals of Pure and Applied Logic*, 32:209 – 243, 1986.

[2] R. L. Constable, S. F. Allen, H. M. Bromley, W. R. Cleaveland, J. F. Cremer, R. W. Harper, D. J. Howe, T. B. Knoblock, N. P. Mendler, P. Panangaden, J. T. Sasaki, and S. F. Smith. *Implementing Mathematics with the Nuprl Proof Development System*. Prentice-Hall, Inc., Upper Saddle River, NJ, USA, 1986.

[3] P. Dybjer. Internal type theory. In S. Berardi and M. Coppo, editors, *Types for Proofs and Programs*, volume 1158 of *Lecture Notes in Computer Science*, pages 120–134. Springer Berlin Heidelberg, 1996.

[4] R. Harper. *Practical Foundations for Programming Languages*. Cambridge University Press, New York, NY, USA, 2016.

[5] R. Harper and R. Davies. Refining Objects (Preliminary Summary). `https://www.cs.cmu.edu/~rwh/papers/lc60/lc60.pdf`, 2014.

[6] R. Harper and W. Duff. Type Refinements in an Open World. `https://www.cs.cmu.edu/~rwh/papers/trow/summary.pdf`, 2015.

[7] J. Hickey, A. Nogin, R. L. Constable, B. E. Aydemir, E. Barzilay, Y. Bryukhov, R. Eaton, A. Granicz, A. Kopylov, C. Kreitz, V. N. Krupski, L. Lorigo, S. Schmitt, C. Witty, and X. Yu. MetaPRL – a modular logical environment. In D. Basin and B. Wolff, editors, *Theorem Proving in Higher Order Logics*, volume 2758 of *Lecture Notes in Computer Science*, pages 287–303. Springer Berlin Heidelberg, 2003.

[8] D. J. Howe. Equality in lazy computation systems. In *Proceedings of Fourth IEEE Symposium on Logic in Computer Science*, pages 198–203.

[9] S. Mac Lane. *Categories for the Working Mathematician*. Number 5 in Graduate Texts in Mathematics. Springer-Verlag, 1971.

[10] S. Mac Lane and I. Moerdijk. *Sheaves in geometry and logic : a first introduction to topos theory*. Universitext. Springer, New York, 1992.

[11] P. Martin-Löf. An intuitionistic theory of types, 1972.

[12] P. Martin-Löf. Constructive mathematics and computer programming. In L. J. Cohen, J. Łoś, H. Pfeiffer, and K.-P. Podewski, editors, *Logic, Methodology and Philosophy of Science VI, Proceedings of the Sixth International Congress of Logic, Methodology and Philosophy of Science, Hannover 1979*, volume 104 of *Studies in Logic and the Foundations of Mathematics*, pages 153–175. North-Holland, 1982.

[13] P. Martin-Löf. On the meanings of the logical constants and the justifications of the logical laws. *Nordic Journal of Philosophical Logic*, 1(1):11–60, 1996.

[14] P. Martin-Löf and G. Sambin. *Intuitionistic type theory*. Studies in proof theory. Bibliopolis, Napoli, 1984.

[15] D. Scott. Constructive validity. In M. Laudet, D. Lacombe, L. Nolin, and M. Schtzenberger, editors, *Symposium on Automatic Demonstration*, volume 125 of *Lecture Notes in Mathematics*, pages 237–275. Springer Berlin Heidelberg, 1970.

[16] J. Sterling, D. Gratzer, and V. Rahli. JonPRL. `http://www.jonprl.org/`, 2015.

[17] J. Sterling and D. Morrison. Syntax and Semantics of Abstract Binding Trees. 2015.