A Generalized Logical Framework

András Kovács[1] and Christian Sattler[2]

Chalmers University of Technology & University of Gothenburg, Sweden
[1] `andrask@chalmers.se` [2] `sattler@chalmers.se`

Logical frameworks (LFs [3]) and the closely related two-level type theories (2LTTs [1]) let us work in a mixed syntax of a metatheory and a chosen object theory. Here, we have a second-order view on the object theory, where contexts, variables and substitutions are implicit, and binders are represented as meta-level functions. There are some well-known limitations to LFs. First, we have to pick a model of the object theory externally. Second, since we only have a second-order view on that model, many constructions cannot be expressed; for example, the induction principle for the syntax of an object theory requires a notion of first-order model, where contexts and substitutions are explicit. Various ways have been described to make logical frameworks more expressive by extending them with modalities (e.g. [8, 4, 7, 6]). In the current work we describe an LF with the following features:

- We can work with multiple models of multiple object theories at the same time. By "theory" we mean a second-order generalized algebraic theory (SOGAT [9, 5]); this includes all type theories and programming languages that only use structural binders.
- We have both an "external" first-order view and an "internal" second-order view on each model, and we can freely switch between perspectives. All models of object theories are defined internally in the LF.
- The LF is fully structural as a type theory; no substructural modalities are used.

The Logical Framework. The basic structure is as follows.

- We have a universe MetaTy[1] closed under the type formers of extensional type theory.
- We have $\mathsf{Base} : \mathsf{MetaTy}$, $1 : \mathsf{Base}$, $\mathsf{PSh} : \mathsf{Base} \to \mathsf{MetaTy}$ and $\mathsf{El} : \{i : \mathsf{Base}\} \to \mathsf{PSh}\,i \to \mathsf{MetaTy}$ such that each $\mathsf{PSh}\,i$ and El constitutes a Tarski-style universe closed under ETT type formers.
- Let us define $\mathsf{Cat}\,i : \mathsf{PSh}\,i$ as the type of categories internally to $\mathsf{PSh}\,i$. Then, we have $\mathsf{In} : \{i : \mathsf{Base}\} \to \mathsf{El}\,(\mathsf{Cat}\,i) \to \mathsf{MetaTy}$ and $\mathsf{base} : \mathsf{In}\,C \to \mathsf{Base}$.

We give some semantic intuition in the following. Each $\mathsf{PSh}\,i$ is a universe of presheaves over some base category. In the empty context, only $\mathsf{PSh}\,1$ is available, which is the universe of sets. Internally to $\mathsf{PSh}\,1$, we can define some $C : \mathsf{El}\,(\mathsf{Cat}\,1)$. Now, if we have $i : \mathsf{In}\,C$, we can form $\mathsf{PSh}\,(\mathsf{base}\,i)$ as the universe of presheaves over C.

1. We can define $\mathsf{PShExt}\,\mathsf{C} : \mathsf{PSh}\,1$ as the *external* type of presheaves over C.
2. Our semantics supports the isomorphism $\mathsf{El}\,(\mathsf{PShExt}\,C) \simeq ((i : \mathsf{In}\,C) \to \mathsf{PSh}\,(\mathsf{base}\,i))$. In other words, external and internal notions of presheaves coincide. More generally, we have this isomorphism for any $C : \mathsf{El}\,(\mathsf{Cat}\,j)$, i.e. starting from a category that's internal to any previously defined presheaf universe.

[1] More precisely, a $\mathbb{N}$-indexed universe hierarchy, but we shall omit "sizing" levels in this abstract.

Yoneda embeddings. Our semantics actually supports a more general notion of internalization than the above one, which we don't describe here. We have not yet finalized which operations to enshrine in the LF's syntax, but the special case of *Yoneda embeddings* seems to be especially useful. This works in the generality of SOGATs but we shall focus on the example of pure lambda calculus. A second-order model of pure LC in some universe U is simply $\mathsf{Tm} : \mathsf{U}$ together with an isomorphism $\mathsf{Tm} \simeq (\mathsf{Tm} \to \mathsf{Tm})$. A first-order model is a unityped category with families [2], where we write $\mathsf{Con} : \mathsf{U}$ for the type of contexts, $\mathsf{Tm} : \mathsf{Con} \to \mathsf{U}$ for the type of terms, $\Gamma{+} : \mathsf{Con}$ for the extension of $\Gamma : \mathsf{Con}$ with a binding, and we have a natural isomorphism $\mathsf{Tm}\,\Gamma \simeq \mathsf{Tm}\,(\Gamma{+})$.

- For each M a first-order model in $\mathsf{PSh}\,i$ and $j : \mathsf{In}\,M$[2], we have S_j as a second-order model in $\mathsf{PSh}\,j$. In other words, internally to presheaves over a model of lambda calculus, we have a second-order model of lambda calculus. In fact, this is the standard semantics of traditional LFs/2LTTs, and we get all such LFs/2LTTs as syntactic fragments of our generalized LF, by working under an assumption of $j : \mathsf{In}\,M$.
- Yoneda embedding has action on contexts, substitutions and terms:

$$\begin{aligned}
&\mathsf{Y} : \mathsf{El}\,\mathsf{Con}_M \to (\{j : \mathsf{In}\,M\} \to \mathsf{PSh}\,j)\\
&\mathsf{Y} : \mathsf{El}\,(\mathsf{Sub}_M\,\Gamma\,\Delta) \simeq (\{j : \mathsf{In}\,M\} \to \mathsf{El}\,(\mathsf{Y}\,\Gamma\,\{j\}) \to \mathsf{El}\,(\mathsf{Y}\,\Delta\,\{j\}))\\
&\mathsf{Y} : \mathsf{El}\,(\mathsf{Tm}_M\,\Gamma) \simeq (\{j : \mathsf{In}\,M\} \to \mathsf{El}\,(\mathsf{Y}\,\Gamma) \to \mathsf{El}\,\mathsf{Tm}_{\mathsf{S}_j})
\end{aligned}$$

Additionally, Y preserves empty contexts and extended contexts up to isomorphism and preserves all other structure strictly. Y allows ad-hoc switching between first-order and second-order syntax. For example, the identity substitution $\mathsf{id} : \mathsf{El}\,(\mathsf{Sub}_M\,\Gamma\,\Gamma)$ can be alternatively defined as $\mathsf{Y}^{-1}(\lambda\,\gamma.\,\gamma)$, where we use Y^{-1} to externalize $(\lambda\,\gamma.\,\gamma) : (\{j : \mathsf{In}\,M\} \to \mathsf{El}\,(\mathsf{Y}\,\Gamma) \to \mathsf{El}\,(\mathsf{Y}\,\Gamma))$. More generally, by using a modest amount of syntactic sugar and elaboration, we can develop Y and Y^{-1} into a "second-order notation" for any SOGAT, which constitutes a rigorous and nicely readable alternative to De Bruijn indices and explicit substitution operations.

Sketch of the semantics. The model of LF is constructed in two steps. First, we give a model for the theory that has PSh, Base and In as sorts but does not support MetaTy, and then take presheaves over that model to obtain a model of a 2LTT where MetaTy represents the outer layer. In the inner model, we start with an inductive definition of certain "trees in categories":

$$\begin{aligned}
&\textbf{data}\,\mathsf{Tree}\,(B : \mathsf{Cat}) : \mathsf{Set}\,\textbf{where}\\
&\quad \mathsf{node} : (\Gamma : \mathsf{PSh}\,B)(n : \mathbb{N})(C : \mathsf{Fin}\,n \to \mathsf{Fib}\,(B \triangleright \mathsf{Disc}\,\Gamma))\\
&\qquad\qquad ((i : \mathsf{Fin}\,n) \to \mathsf{Tree}\,(B \triangleright \mathsf{Disc}\,\Gamma \triangleright C\,i)) \to \mathsf{Tree}\,B
\end{aligned}$$

Here, PSh means presheaves in sets, Fib is cartesian fibrations, Disc creates a discrete fibration from a presheaf and $-\triangleright-$ takes the total category of a fibration. Now, the objects of the semantic base category are elements of $\mathsf{Tree}\,1$, and morphisms between trees are level-wise natural transformations between the Γ components together with $\mathsf{Fin}\,n \to \mathsf{Fin}\,m$ renamings of subtree indices. The non-discrete Fib components are preserved by morphisms. A semantic Base points to a subtree of a context, an In is a $\mathsf{Fin}\,n$ index pointing to a child of a given node, and a PSh is a dependent presheaf over a Γ inside a given node. Extending a context with an In binding adds a new empty subtree to a given node. Extending with an El binding extends the Γ presheaf in a node with a dependent presheaf.

[2] We implicitly take the underlying category of M here.

References

[1] Danil Annenkov, Paolo Capriotti, Nicolai Kraus, and Christian Sattler. Two-level type theory and applications. *ArXiv e-prints*, may 2019.

[2] Simon Castellan, Pierre Clairambault, and Peter Dybjer. Categories with families: Unityped, simply typed, and dependently typed. *CoRR*, abs/1904.00827, 2019.

[3] Robert Harper, Furio Honsell, and Gordon Plotkin. A framework for defining logics. *J. ACM*, 40(1):143–184, January 1993.

[4] Martin Hofmann. Semantical analysis of higher-order abstract syntax. In *14th Annual IEEE Symposium on Logic in Computer Science, Trento, Italy, July 2-5, 1999*, pages 204–213. IEEE Computer Society, 1999.

[5] Ambrus Kaposi and Szumi Xie. Second-order generalised algebraic theories: Signatures and first-order semantics. In Jakob Rehof, editor, *9th International Conference on Formal Structures for Computation and Deduction, FSCD 2024, July 10-13, 2024, Tallinn, Estonia*, volume 299 of *LIPIcs*, pages 10:1–10:24. Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 2024.

[6] Ian Orton and Andrew M. Pitts. Axioms for Modelling Cubical Type Theory in a Topos. In Jean-Marc Talbot and Laurent Regnier, editors, *25th EACSL Annual Conference on Computer Science Logic (CSL 2016)*, volume 62 of *Leibniz International Proceedings in Informatics (LIPIcs)*, pages 24:1–24:19, Dagstuhl, Germany, 2016. Schloss Dagstuhl–Leibniz-Zentrum fuer Informatik.

[7] Brigitte Pientka, Andreas Abel, Francisco Ferreira, David Thibodeau, and Rébecca Zucchini. Cocon: Computation in contextual type theory. *CoRR*, abs/1901.03378, 2019.

[8] Jonathan Sterling. *First Steps in Synthetic Tait Computability.* PhD thesis, Carnegie Mellon University Pittsburgh, PA, 2021.

[9] Taichi Uemura. A general framework for the semantics of type theory. *CoRR*, abs/1904.04097, 2019.