$\mathsf{LF}_{\mathcal{P}}$ - A Logical Framework with External Predicates[1]

Furio Honsell, Università di Udine, Italy
Marina Lenisa, Università di Udine, Italy
Petar Maksimović, INRIA Sophia Antipolis Méditerranée, France and Mathematical Institute of the Serbian Academy of Sciences and Arts, Serbia
Ivan Scagnetto, Università di Udine, Italy
Luigi Liquori, INRIA Sophia Antipolis Méditerranée, France

The Edinburgh Logical Framework LF, presented in [2], is a first-order constructive type theory featuring dependent types. It was first introduced as a *general meta-language for logics*, as well as a specification language for *generic proof-checking/proof-development environments*. In this abstract, we present an extension of LF with *predicates* and *oracle calls*, which is accomplished by defining a mechamism serving for the *locking* and *unlocking* of types and terms.

Following the standard specification paradigm in Constructive Type Theory, we define locked types using *introduction*, *elimination*, and *equality rules*. We introduce a lock *constructor* for building objects $\mathcal{L}^{\mathcal{P}}_{N,\sigma}[M]$ of type $\mathcal{L}^{\mathcal{P}}_{N,\sigma}[\rho]$, via the *introduction rule* ($O{\cdot}Lock$), presented below, and a corresponding unlock *destructor*, $\mathcal{U}^{\mathcal{P}}_{N,\sigma}[M]$, and an *elimination rule* ($O{\cdot}Unlock$) which allows elimination of the locked type constructor, under the condition that a specific predicate $\mathcal{P}$ is verified, possibly *externally*, on an appropriate *correct*, *i.e.* derivable, judgement.

$$\frac{\Gamma \vdash_\Sigma M : \rho \qquad \Gamma \vdash_\Sigma N : \sigma}{\Gamma \vdash_\Sigma \mathcal{L}^{\mathcal{P}}_{N,\sigma}[M] : \mathcal{L}^{\mathcal{P}}_{N,\sigma}[\rho]} \qquad (O{\cdot}Lock)$$

$$\frac{\Gamma \vdash_\Sigma M : \mathcal{L}^{\mathcal{P}}_{N,\sigma}[\rho] \qquad \Gamma \vdash_\Sigma N : \sigma \qquad \mathcal{P}(\Gamma \vdash_\Sigma N : \sigma)}{\Gamma \vdash_\Sigma \mathcal{U}^{\mathcal{P}}_{N,\sigma}[M] : \rho} \qquad (O{\cdot}Unlock)$$

The *equality rule* for locked types amounts to a new form of reduction we refer to as *lock-reduction* ($\mathcal{L}$-reduction), $\mathcal{U}^{\mathcal{P}}_{N,\sigma}[\mathcal{L}^{\mathcal{P}}_{N,\sigma}[M]] \to_{\mathcal{L}} M$, which allows elimination of a *lock*, in the presence of an *unlock*. The $\mathcal{L}$-reduction combines with standard β-reduction into $\beta\mathcal{L}$-reduction.

$\mathsf{LF}_{\mathcal{P}}$ is parametric over a potentially unlimited set of predicates $\mathcal{P}$, which are defined on derivable typing judgements of the form $\Gamma \vdash_\Sigma N : \sigma$. The syntax of $\mathsf{LF}_{\mathcal{P}}$ predicates is not specified, with the main idea being that their truth is to be verified via a *call* to an *external validation tool*; one can view this externalization as an *oracle call*. Thus, $\mathsf{LF}_{\mathcal{P}}$ allows for the invocation of external "modules" which, in principle, can be executed elsewhere, and whose successful verification

[1]This work was supported by the Serbian Ministry of Education, Science, and Technological Development (projects ON174026 and III044006).

can be acknowledged in the system via $\mathcal{L}$-reduction. Pragmatically, locked types allow for the factoring out of the complexity of derivations by delegating the {checking, verification, computation} of such predicates to an external proof engine or tool. The proof terms themselves do not contain explicit evidence for external predicates, but just record that a verification {has to be (lock), has been successfully (unlock)} carried out. In this manner, we combine the reliability of formal proof systems based on constructive type theory with the efficiency of other computer tools, in the style of the *Poincaré Principle* [5].

In this abstract, we only outline the main results on $\mathsf{LF}_{\mathcal{P}}$, which have been treated in detail in [3, 4]. As far as meta-theoretic properties are concerned, strong normalization and confluence of the system have been proven without imposing any additional assumptions on the predicates. However, for subject reduction, we require the predicates to be *well-behaved*, *i.e.closed under weakening and permutation of the signature and context*, *substitution*, and *$\beta\mathcal{L}$-reduction* in the arguments. $\mathsf{LF}_{\mathcal{P}}$ is decidable, if the external predicates are decidable. Furthermore, a *canonical* presentation of $\mathsf{LF}_{\mathcal{P}}$ is constructed, in the style of [1, 6], based on a suitable extension of the notion of $\beta\eta$-long normal form, allowing for simple proofs of the adequacy of the encodings.

When it comes to illustrating the main benefits of $\mathsf{LF}_{\mathcal{P}}$ in practice, we have provided a number of relevant encodings. We have encoded in $\mathsf{LF}_{\mathcal{P}}$ the call-by-value λ-calculus, as well as its extension supporting the *design-by-contract* paradigm. We also provide smooth encodings of side conditions in the rules of Modal Logics, both in Hilbert and Natural Deduction styles, and also show how to encode sub-structural logics, *i.e.*non-commutative Linear Logic. We also illustrate how $\mathsf{LF}_{\mathcal{P}}$ can naturally support *program correctness* systems and Hoare-like logics. We show that other related systems can be embedded into $\mathsf{LF}_{\mathcal{P}}$ via locked types, and provide pseudo-code for some of the used predicates.

As far as expressiveness is concerned, $\mathsf{LF}_{\mathcal{P}}$ is a stepping stone towards a general theory of *shallow vs. deep encodings*, with our encodings being shallow by definition. Clearly, by Church's thesis, all external decidable predicates in $\mathsf{LF}_{\mathcal{P}}$ can be encoded, possibly with very deep encodings, in standard LF. It would be interesting to state in a precise categorical setting the relationship between such deep internal encodings and the encodings in $\mathsf{LF}_{\mathcal{P}}$.

$\mathsf{LF}_{\mathcal{P}}$ can also be viewed as a neat methodology for separating the logical-deductive contents from, on one hand, the verification of structural and syntactical properties, which are often needlessly cumbersome but ultimately computable, or, on the other hand, from more general means of validation.

From a philosophical point of view, the mechanism of *locking and unlocking types* in the presence of *external oracles*, which we are introducing in $\mathsf{LF}_{\mathcal{P}}$, effectively opens up the Logical Framework to alternate means of providing evidence for judgements. In standard LF, there are only two ways of providing this evidence, namely discovering types to be inhabited or postulating that types are inhabited by introducing appropriate constants. The *locked/unlocked types* of $\mathsf{LF}_{\mathcal{P}}$ open the door to an intermediate level, one provided by external means, such as computation engines or automated theorem proving tools. However, among these, one could also think of graphical tools based on neural networks,

or even intuitive visual arguments, as were used in ancient times for giving the first *demonstrations* of the Pythagoras' theorem, for instance. In a sense, $\mathsf{LF}_{\mathcal{P}}$, by allowing formal accommodation of any alternative proof method to pure axiomatic deduction, vindicates all of the "proof cultures" which have been used pragmatically in the history of mathematics, not only in the Western tradition.

The traditional LF answer to the question "What is a Logic?" was: "A signature in LF". In $\mathsf{LF}_{\mathcal{P}}$, we can give the homologue answer, namely "A signature in $\mathsf{LF}_{\mathcal{P}}$", since external predicates can be read off the types occurring in the signatures themselves. But, we can also use this very definition to answer a far more intriguing question:

"What is a Proof Culture?".

References

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

[2] Robert Harper, Furio Honsell, and Gordon Plotkin. A framework for defining logics. *Journal of the ACM*, 40:143–184, January 1993.

[3] F. Honsell, M. Lenisa, L. Liquori, P. Maksimovic, and I. Scagnetto. LF_P – a logical framework with external predicates. In *Proceedings of LFMTP 2012*, pages 13–22. ACM Digital Library, 2013.

[4] Furio Honsell, Marina Lenisa, Luigi Liquori, Petar Maksimović, and Ivan Scagnetto. An open logical framework. *Journal of Logic and Computation*, 2013. DOI:10.1093/logcom/ext028.

[5] H. Poincaré. *La Science et l'Hypothèse*. Flammarion, Paris, 1902.

[6] K. Watkins, I. Cervesato, F. Pfenning, and D. Walker. A Concurrent Logical Framework I: Judgments and Properties. Tech. Rep. CMU-CS-02-101, 2002.