EXTENDED ABSTRACT:

Denotational semantics of general store and polymorphism*

Jonathan Sterling Daniel Gratzer Lars Birkedal

March 10, 2023

We contribute the first denotational semantics of polymorphic dependent type theory extended by an equational theory for general (higher-order) reference types and recursive types, based on a combination of guarded recursion and impredicative polymorphism; because our model is based on *recursively defined semantic worlds*, it is compatible with polymorphism and relational reasoning about stateful abstract datatypes. What is new in relation to prior typed denotational models of higher-order store is that our Kripke worlds need not be syntactically definable, and are thus compatible with relational reasoning in the heap. Our work combines recent advances in the operational semantics of state with the purely denotational viewpoint of *synthetic guarded domain theory*.

1 A new metalanguage for denotational semantics

The basis of our result is ***impredicative guarded dependent type theory*** (**iGDTT**) a new metalanguage for synthetic denotational semantics that extends ordinary guarded dependent type theory [2] by an impredicative universe of sets. Syntactically, an impredicative universe is a model of higher-order polymorphism with universal and existential types; on the semantic side, an impredicative universe corresponds a *complete* full internal subcategory, and can be found in any category of assemblies [8, 9]. Completeness of full internal subcategories is preserved by the Hofmann and Streicher [6] lifting of universes, and hence a particularly bountiful source of models for **iGDTT** is to take internal presheaves [10] on a well-founded order [1, 16] internal to any category of assemblies [15, 7].

Impredicativity and guarded recursion are therefore two orthogonal features that, when combined, lead to a very simple description of a possible worlds model of higher-order store supporting polymorphism, in contrast to several prior possible worlds models [13, 12]. To summarize:

1. **Section 2**: a suitable preorder $\mathcal{W}$ of semantic Kripke worlds can be defined in ***predicative*** guarded dependent type theory (**GDTT**); covariant presheaves on $\mathcal{W}$ are immediately closed under reference types.
2. **Section 3**: to define the function space and the state monad in our semantic domain, we require the additional impredicativity of **iGDTT**.

*Full preprint: *Denotational semantics of general store and polymorphism* [19].

3. **Section** 4: finally, we show how to extend the denotational semantics of Sections 2 and 3 from a stateful version of System F_ω to an extension of **iGDTT** itself with general reference types, the first model of polymorphic dependent type theory with higher-order store and recursive types.

2 Domain equations for higher-order store in GDTT

If $\mathcal{U}$ is *any* universe in the predicative version of **GDTT**, we may define a preorder of $\mathcal{U}$-valued Kripke worlds by solving a guarded domain equation:

$$\mathcal{W}_{\mathcal{U}} = \mathsf{Loc} \rightharpoonup_{\mathit{fin.}} \blacktriangleright \mathcal{T}_{\mathcal{U}} \qquad \mathcal{T}_{\mathcal{U}} = \mathbf{Functor}(\mathcal{W}_{\mathcal{U}}, \mathcal{U})$$

Given $A \in \mathcal{T}_{\mathcal{U}}$, we may define the type $\mathsf{ref}_{\mathcal{U}} A \in \mathcal{T}_{\mathcal{U}}$ of references to A already, taking the locations at the given world that will be labeled by A in one step:

$$(\mathsf{ref}_{\mathcal{U}} A)_w = \{ l \in |w| \mid wl = \mathsf{next}\, A \}$$

Unfortunately, the category $\mathcal{T}_{\mathcal{U}}$ is not well-behaved unless $\mathcal{U}$ is impredicative: it need not even be cartesian closed, much less support a state monad.

3 A monad for general store in iGDTT

Now let $\mathcal{S}$ be the *impredicative* universe of **iGDTT** and let $\mathcal{W}_{\mathcal{S}}$ and $\mathcal{T}_{\mathcal{S}}$ be the solution to the domain equation for Kripke worlds as in Section 2. Because $\mathcal{S}$ is impredicative, it happens that $\mathcal{T}_{\mathcal{S}}$ is locally cartesian closed: in fact, for any of the predicative universes $\mathcal{U} \ni \mathcal{S}$ of **iGDTT**, the family fibration $\mathbf{Fam}_{\mathcal{U}}(\mathcal{T}_{\mathcal{S}}) \longrightarrow\!\!\triangleright\, \mathcal{U}$ is strongly complete and so $\mathcal{T}_{\mathcal{S}}$ models the full System F_ω. It *also* happens that we may define a state monad on $\mathcal{T}_{\mathcal{S}}$ mirroring that of Levy [13], but in which the heap can store elements of *any* type classified by $\mathcal{T}_{\mathcal{S}}$ regardless of syntactical definability. We first define the type of heaps at a given world $w \in \mathcal{W}_{\mathcal{S}}$:

$$\mathcal{H}_w = \textstyle\prod_{l \in |w|} \blacktriangleright [X \leftarrow wl].Xw$$

Letting L be the ***guarded lift monad*** [18, 17, 14], we may define a state monad $\mathsf{T} : \mathcal{T}_{\mathcal{S}} \longrightarrow \mathcal{T}_{\mathcal{S}}$ using a combination of universals and existentials in $\mathcal{S}$:

$$(\mathsf{T}A)_w = \forall_{w' \geq w}\, \mathcal{H}_{w'} \to \mathsf{L} \exists_{w'' \geq w'}\, \mathcal{H}_{w''} \times A_{w''}$$

It is then routine to close the monad T under effects for reading, writing, and allocating reference cells. The only subtlety is that the heap stores its elements under the later modality $\blacktriangleright$; this is dealt with using the $\blacktriangleright$-algebra structure on $(\mathsf{T}A)_w$ induced by the inner occurrence of the guarded lift monad L. The resulting (monadic) model of effectful System F_ω also supports recursive types.

4 Adding general store to iGDTT itself

In Sections 2 and 3 we have described how to construct an extremely simple but versatile denotational semantics for System F_ω with monadic higher-order store, general reference types, and recursive types. The same construction, however, can be used to give a denotational semantics to a version of **iGDTT** itself such that $\mathcal{S}$ is closed under general reference types and a state monad. This is carried out by applying the Hofmann–Streicher universe lifting once more to obtain an impredicative universe in the internal functor category **Functor**$(\mathcal{W}_{\mathcal{S}}, \mathcal{U})$.

The resulting language, $\mathbf{iGDTT}^{\mathrm{ref}}$, is the first to soundly combine "full-spectrum" dependently typed programming and equational reasoning with both higher-order store and recursive types. The necessity of *semantics* for higher-order store with dependent types is not hypothetical: both Idris 2 and Lean 4 are dependently typed, higher-order functional programming languages [3, 5] that feature a Haskell-style `IO` monad with general `IORef` types [11], but until now no semantics have been proposed for these features. In fact, our investigations have revealed a subtle problem in the `IO` monads of both Idris and Lean: as higher-order store is inherently impredicative, it is not consistent with our semantics to close multiple nested universes under the `IO` monad [4].

References

[1] Lars Birkedal, Rasmus Ejlers Møgelberg, Jan Schwinghammer, and Kristian Støvring. "First Steps in Synthetic Guarded Domain Theory: Step-Indexing in the Topos of Trees". In: *Proceedings of the 2011 IEEE 26th Annual Symposium on Logic in Computer Science*. Washington, DC, USA: IEEE Computer Society, 2011, pp. 55–64. ISBN: 978-0-7695-4412-0. DOI: 10.1109/LICS.2011.16. arXiv: 1208.3596 [cs.LO].

[2] Aleš Bizjak, Hans Bugge Grathwohl, Ranald Clouston, Rasmus E. Møgelberg, and Lars Birkedal. "Guarded Dependent Type Theory with Coinductive Types". In: *Foundations of Software Science and Computation Structures: 19th International Conference, FOSSACS 2016, Held as Part of the European Joint Conferences on Theory and Practice of Software, ETAPS 2016, Eindhoven, The Netherlands, April 2–8, 2016, Proceedings*. Ed. by Bart Jacobs and Christof Löding. Berlin, Heidelberg: Springer Berlin Heidelberg, 2016, pp. 20–35. ISBN: 978-3-662-49630-5. DOI: 10.1007/978-3-662-49630-5_2. arXiv: 1601.01586 [cs.LO].

[3] Edwin Brady. "Idris 2: Quantitative Type Theory in Practice". In: *35th European Conference on Object-Oriented Programming (ECOOP 2021)*. Ed. by Anders Møller and Manu Sridharan. Vol. 194. Leibniz International Proceedings in Informatics (LIPIcs). Dagstuhl, Germany: Schloss Dagstuhl – Leibniz-Zentrum für Informatik, 2021, 9:1–9:26. ISBN: 978-3-95977-190-0. DOI: 10.4230/LIPIcs.ECOOP.2021.9. URL: https://drops.dagstuhl.de/opus/volltexte/2021/14052.

[4] Thierry Coquand. "An Analysis of Girard's Paradox". In: *Proceedings of the First Symposium on Logic in Computer Science*. IEEE Computer Society, 1986, pp. 227–236. ISBN: 0-8186-0720-3.

[5] Leonardo De Moura and Sebastian Ullrich. "The Lean 4 Theorem Prover and Programming Language (System Description)". To appear in the proceedings of the 28th International Conference on Automated Deduction. 2021.

[6] Martin Hofmann and Thomas Streicher. "Lifting Grothendieck Universes". Unpublished note. 1997. URL: https://www2.mathematik.tu-darmstadt.de/~streicher/NOTES/lift.pdf.

[7] J. M. E. Hyland. "The effective topos". In: *The L.E.J. Brouwer Centenary Symposium*. Ed. by A. S. Troelstra and D. Van Dalen. North Holland Publishing Company, 1982, pp. 165–216.

[8] J. M. E. Hyland. "A small complete category". In: *Annals of Pure and Applied Logic* 40.2 (1988), pp. 135–165. ISSN: 0168-0072. DOI: 10.1016/0168-0072(88)90018-8.

[9] J. M. E. Hyland, E. P. Robinson, and G. Rosolini. "The Discrete Objects in the Effective Topos". In: *Proceedings of the London Mathematical Society* s3-60.1 (Jan. 1990), pp. 1–36. ISSN: 0024-6115. DOI: 10.1112/plms/s3-60.1.1.

[10] Peter T. Johnstone. *Sketches of an Elephant: A Topos Theory Compendium: Volumes 1 and 2*. Oxford Logical Guides 43. Oxford Science Publications, 2002.

[11] Simon Peyton Jones. "Tackling the awkward squad: monadic input/output, concurrency, exceptions, and foreign-language calls in Haskell". In: *Engineering theories of software construction, Marktoberdorf Summer School 2000*. Ed. by C. A. R. Hoare, M. Broy, and R. Steinbrueggen. IOS Press, 2001, pp. 47–96.

[12] Ohad Kammar, Paul Blain Levy, Sean K. Moss, and Sam Staton. *A monad for full ground reference cells*. 2017. arXiv: 1702.04908 [cs.PL].

[13] Paul Blain Levy. *Call-by-Push-Value: A Functional/Imperative Synthesis*. Kluwer, Semantic Structures in Computation, 2, Jan. 1, 2003. ISBN: 1-4020-1730-8.

[14] Rasmus Ejlers Møgelberg and Marco Paviotti. "Denotational Semantics of Recursive Types in Synthetic Guarded Domain Theory". In: *Proceedings of the 31st Annual ACM/IEEE Symposium on Logic in Computer Science*. New York, NY, USA: Association for Computing Machinery, 2016, pp. 317–326. ISBN: 978-1-4503-4391-6. DOI: 10.1145/2933575.2934516.

[15] Jaap van Oosten. *Realizability: An Introduction to its Categorical Side*. Elsevier Science, San Diego, 2008.

[16] Daniele Palombi and Jonathan Sterling. "Classifying topoi in synthetic guarded domain theory". In: *Proceedings 38th Conference on Mathematical Foundations of Programming Semantics, MFPS 2022*. To appear. 2022. arXiv: 2210.04636 [math.CT].

[17] Marco Paviotti. "Denotational semantics in Synthetic Guarded Domain Theory". PhD thesis. Denmark: IT-Universitetet i København, 2016. ISBN: 978-87-7949-345-2.

[18] Marco Paviotti, Rasmus Ejlers Møgelberg, and Lars Birkedal. "A Model of PCF in Guarded Type Theory". In: *Electronic Notes in Theoretical Computer Science* 319.Supplement C (2015). The 31st Conference on the Mathematical Foundations of Programming Semantics (MFPS XXXI), pp. 333–349. ISSN: 1571-0661. DOI: 10.1016/j.entcs.2015.12.020.

[19] Jonathan Sterling, Daniel Gratzer, and Lars Birkedal. "Denotational semantics of general store and polymorphism". Unpublished manuscript. July 2022. DOI: 10.48550/arXiv.2210.02169.