On self-interpreters for Gödel's System T

Andrej Bauer

University of Ljubljana, Slovenia
`Andrej.Bauer@andrej.com`

In defiance of the received wisdom that a total programming language cannot have a self-interpreter, Brown and Palsberg [3] implemented a self-interpreter for System F_ω, which is a strongly normalizing typed λ-calculus and thus certainly a total language. In order to avoid the trivial self-interpreter they imposed certain constraints. I show that under the same constraints already Gödel's System T has a self-interpreter (Theorem 5). The construction is trivial, which makes one think that something is at fault with the notion of self-interpreter used by Brown and Palsberg. However, I show that there cannot be a significant improvement (Corollary 6) in the sense that the type of source codes must be at least as complex as the type of the programs they encode. I conclude by suggesting a definition of self-interpreter which is satisfied by Brown and Palsberg's interpreter but not by the one constructed in Theorem 5.

We work with the simply typed λ-calculus [2, §A.1], and in particular with Gödel's T, which is an extension of the simply typed λ-calculus with a ground type of natural numbers nat and *primitive* recursion at each type, see [2, §A.2] and [1]. It is strongly normalizing [1, §4.3] and expressive enough to manipulate syntax through Gödel encodings. We write $\mathsf{Prg}(\tau)$ for the set of all closed expressions (programs) of type τ.

Definition 1. A *typed self-interpreter* is given by a type ν of *(source) codes*, and for each type τ a *quoting function* $\ulcorner\text{-}\urcorner_\tau : \mathsf{Prg}(\tau) \to \mathsf{Prg}(\nu)$ and an *interpreter* $\mathsf{u}_\tau \in \mathsf{Prg}(\nu \to \tau)$ such that $\mathsf{u}_\tau\, \ulcorner e \urcorner_\tau \equiv_\beta e$ for all $e \in \mathsf{Prg}(\tau)$.

Note that the quoting functions need not be λ-definable, i.e., there may be no programs q_τ such that $\ulcorner e \urcorner_\tau \equiv_\beta \mathsf{q}_\tau\, e$. The following theorem is the justification for the popular opinion that total languages do not have self-interpreters.

Theorem 2. *If a λ-calculus has a self-interpreter then it has fixed-point operators at all types.*

The theorem can be inverted: a simply typed λ-calculus with natural numbers and fixed-point operators has a self-interpreter, as was affirmed by Longley and Plotkin [4, Prop. 6].

Corollary 3. *System T does not have a self-interpreter.*

Proof. In System T successor $\mathsf{succ} : \mathsf{nat} \to \mathsf{nat}$ has no fixed points. $\square$

The corollary holds for other kinds of calculi, as long as they posses endomaps without fixed points, which is typical of strongly normalizing calculi. To obtain a self-interpreter for System T we thus need to relax the definition of self-interpreters. The following one is fashioned after Brown and Palsberg [3]. We write $\mathfrak{n}(e)$ for the normal form of an expression e, and $\mathfrak{g}(e)$ for its Gödel code, which is a suitable encoding of e by a number. We write $\underline{n}$ for the numeral that represents $n \in \mathbb{N}$.

Definition 4. A *weak self-interpreter* is given by, for each type τ, a type of *(source) codes* $\Box\tau$, a *quoting function* $\ulcorner\text{-}\urcorner_\tau : \mathsf{Prg}(\tau) \to \mathsf{Prg}(\Box\tau)$, and an *interpreter* $\mathsf{u}_\tau : \mathsf{Prg}(\Box\tau \to \tau)$ such that $\mathsf{u}_\tau\, \ulcorner e \urcorner_\tau \equiv_\beta e$ for all $e \in \mathsf{Prg}(\tau)$. Such an interpreter is *strong* when for every type τ, the quoting function $\ulcorner\text{-}\urcorner_\tau$ is (1) *normal*: $\ulcorner e \urcorner_\tau$ is β-normal for all $e \in \mathsf{Prg}(\tau)$, and (2) *acceptable*: there is $\mathsf{g}_\tau : \Box\tau \to \mathsf{nat}$ such that $\mathsf{g}_\tau\, \ulcorner e \urcorner_\tau = \underline{\mathfrak{g}(e)}$ for all $e \in \mathsf{Prg}(\tau)$.

Normality expresses the idea that codes should be values and acceptability that the syntax of an expression is discernible from its code. Brown and Palsberg also require injectivity of the quoting function, which follows from our definition because $\mathfrak{g}$ is injective. They do not explicitly postulate acceptability, although they provide programs that extract the syntax of an expression from its code.

A strong self-interpreter cannot have a trivial quoting function $\ulcorner e \urcorner_\tau = e$ because codes must be β-normal, while injectivity of $\ulcorner\text{-}\urcorner_\tau$ prevents coding by β-normal forms $\ulcorner e \urcorner_\tau = \mathfrak{n}(e)$.

Theorem 5. *System T has a strong Brown-Palsberg self-interpreter.*

Proof. Define $\Box\tau = \mathsf{nat} \times \tau$, $\ulcorner e \urcorner_\tau = \langle \underline{\mathfrak{g}(e)}, \mathfrak{n}(e)\rangle$, $\mathsf{u}_\tau = \mathsf{snd}$, and $\mathsf{g}_\tau = \mathsf{fst}$. $\Box$

It is clear that the same proof applies to any calculus that has binary products, natural numbers, and any notion of normal form, such as System F_ω.

The proof of Theorem 5 abuses the fact that Brown-Palsberg interpreters allow codes to be as complex as the programs they encode. Theorem 2 prevents us from using a fixed type of codes, but perhaps $\Box\tau$ can at least be less complex than τ? We show that this is not possible for the standard notion of *level* defined inductively as $\mathsf{lev}(\mathsf{nat}) = 0$, $\mathsf{lev}(\sigma \times \tau) = \max(\mathsf{lev}(\sigma), \mathsf{lev}(\tau))$, and $\mathsf{lev}(\sigma \to \tau) = \max(1 + \mathsf{lev}(\sigma), \mathsf{lev}(\tau))$. That is, $\mathsf{lev}(\tau)$ gives the deepest nesting of $\to$ to the left in τ.

Theorem 6. *A weak self-interpreter for System T satisfies* $\mathsf{lev}(\Box\tau) \geq \mathsf{lev}(\tau)$ *for every type τ.*

The self-interpreter for F_ω given by Brown and Palsberg has important structural properties that Definition 4 fails to capture. For instance, their encoding of types commutes with substitution [3, Thm. 5.2] and is a congruence with respect to type equality [3, Thm. 5.3]. In the original work [5] on meta-circularity Pfenning and Lee called such phenomena *reflexivity*. Unfortunately they spoke of it at an informal level and did not provide a definition. A promising possibility, which thwarts the proof of Theorem 5 but allows Brown and Palsberg's construction, is to amend the definition of weak interpreters by requiring a term $\mathsf{app}_{\sigma,\tau} : \Box(\sigma \to \tau) \to \Box\sigma \to \Box\tau$ such that $\mathsf{app}_{\sigma,\tau}\ \ulcorner e_1 \urcorner_{\sigma\to\tau}\ \ulcorner e_2 \urcorner_\sigma \equiv_\beta \ulcorner e_1\ e_2 \urcorner_\tau$. This way we get conditions that correspond to the modal laws of necessity. Unfortunately, at present I do not know whether System T has a self-interpreter satisfying the above conditions *and* the acceptability condition from Definition 4.

Acknowledgment I thank Alex Simpson for helpful discussions and pointers to the literature.

References

[1] Jeremy Avigad and Solomon Feferman. Chapter V: Gödel's Functional ("Dialectica") Interpretation. In Samuel R. Buss, editor, *Handbook of Proof Theory*, volume 137 of *Studies in Logic and the Foundations of Mathematics*, pages 337–405. Elsevier, 1998.

[2] H. P. Barendregt. *The Lambda Calculus: Its Syntax and Semantics*, volume 103 of *Studies in Logic and the Foundations of Mathematics*. College Publications, 1984.

[3] Matt Brown and Jens Palsberg. Breaking through the normalization barrier: A self-interpreter for F-omega. In *Principles of Programming Languages (POPL)*, January 2016.

[4] John Longley and Gordon Plotkin. Logical full abstraction and pcf. In *Tbilisi Symposium on Language, Logic and Computation. SiLLI/CSLI*, pages 333–352. SiLLI/CSLI, 1996.

[5] Frank Pfenning and Peter Lee. Metacircularity in the polymorphic λ-calculus. *Theoretical Computer Science*, 89(1):137–159, 1991.