Reynolds's Parametricity Theorem, Directly*

Robert Harper

Spring, 2021

1 Introduction

In a landmark paper Reynolds (1983) develops a mathematical account of Strachey's informal concept of *parametricity* of polymorphic functions. Parametricity characterizes the "uniform" behavior of polymorphic functions using *logical relations*, a concept introduced by Tait (1967) in the study of functionals of finite type.[1] Reynolds's work, which was motivated by the study of data abstraction in programming languages, was done around the same time as, and independently of, Girard's extension of Tait's method to second-order quantification, which was motivated by the analysis of proofs in second-order logic.[2] Whereas Girard made use of unary predicates, Reynolds used binary relations, a technically small, yet practically large, difference that gave rise to new methods for proving properties of programs knowing only their types. Reynolds observed that the type discipline of a language determines the abstraction properties enjoyed by its programs; in particular, clients of abstract types are polymorphic, and hence enjoy stability properties across changes of representation determined entirely by their types.

The formulation given here makes use of the aforementioned ideas from Tait, Girard, and Reynolds, but cast in an operational framework inspired by Martin-Löf (1982) and Constable et al. (1986) and used throughout Harper (2016). In contrast to the presentation in **PFPL** the formulation given here is independent of a prior notion of equality. In compensation candidates are required to enjoy a property called *zig-zag completeness* (Krishnaswami and Dreyer, 2012), which ensures that equality is transitive.

Thanks to Carlo Angiuli, Karl Crary and Yiyang Guo for helpful discussions.

2 The Language

Please see Harper (2020a) for the definition of the language F, including its syntax and dynamics. The dynamics is to be understood via a tacit *erasure* of type information from terms given as follows:

1. Type labels are removed from λ-abstractions, $\lambda_A(x\,.\,M)$ becomes $\lambda x.M$.

2. Λ-abstractions $\Lambda X.M$ are replaced by anonymous λ-abstractions $\lambda\_.M$, abbreviated $\Lambda.M$.

[1] See Harper (2020b) for an introduction to Tait's method.

[2] See Harper (2020a) for an introduction to Girard's method.

3. Type applications $\mathsf{Ap}(M, A)$ are replaced by ordinary applications $\mathsf{ap}(M, M_0)$ to a fixed trivial argument M_0, abbreviated $\mathsf{Ap}(M)$.

With this understanding it is never necessary to substitute types for type variables when defining the dynamics of terms.

3 Exact Equality Via Parametricity

A binary relation R over closed terms is a *type candidate*, written $R\ \mathsf{Cand}$, iff it satisfies the following two closure conditions:

1. *Head expansion*: if $M\ R\ M'$, then
   (a) if $N \longmapsto M$, then $N\ R\ M'$, and
   (b) if $N' \longmapsto M'$, then $M\ R\ N'$.
2. *Zig-Zag Completeness (ZZC)*: if $M\ R\ M'$, $N\ R\ N'$, and $N\ R\ M'$, then $M\ R\ N'$.

Head expansion is a natural requirement when thinking of types as specifications of program behavior. Zig-zag completeness is depicted in Figure 1. It may be re-stated in terms of the converse and composition of relations as the containment $R \circ R^{\mathsf{op}} \circ R \subseteq R$. The opposite containment is always valid, so zig-zag completeness may be re-stated as the equation $R \circ R^{\mathsf{op}} \circ R = R$.

Lemma 1. *Suppose that R is a zig-zag complete binary relation.*

1. *R^{op} is zig-zag complete.*
2. *$R^{\mathsf{op}} \circ R$ and $R \circ R^{\mathsf{op}}$ are symmetric and transitive, and hence zig-zag complete.*
3. *$R \circ R^{\mathsf{op}} \circ R$ is zig-zag complete.*

Proof. 1. If $R = R \circ R^{\mathsf{op}} \circ R$, then $R^{\mathsf{op}} = R^{\mathsf{op}} \circ R \circ R^{\mathsf{op}}$.

2. Exercise.

3. Immediate from the definition of R being zig-zag complete.

□

Exercise 1. *Complete the proof of Lemma 1.*

A *candidate assignment*, η, for Δ is a function such that $\eta(X)$ is a type candidate for each type variable X such that $\Delta \vdash X\ \mathsf{type}$.

Definition 2 (Logical Similarity)**.** Logical similarity *of two closed terms relative to an open type A such that $\Delta \vdash A\ \mathsf{type}$ and a candidate assignment for Δ, written $M \sim M' \in A\,[\eta]$, is defined by*

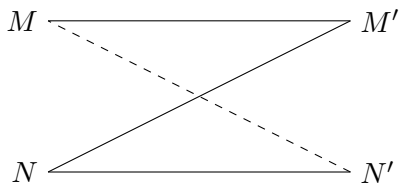

Figure 1: Zig-Zag Completeness

induction on the structure of A as follows:

$$M \sim M' \in \mathbf{2}\ [\,\eta\,] \Leftrightarrow \textit{either } M, M' \longmapsto^* \mathsf{yes} \textit{ or } M, M' \longmapsto^* \mathsf{no}$$

$$M \sim M' \in X\ [\,\eta\,] \Leftrightarrow M\ \eta(X)\ M'$$

$$M \sim M' \in A_1 \to A_2\ [\,\eta\,] \Leftrightarrow \begin{cases} M \longmapsto^* \lambda x.M_2,\ M' \longmapsto^* \lambda x.M_2', & \textit{and} \\ [M_1/x]M_2 \sim [M_1'/x]M_2' \in A_2\ [\,\eta\,] & \textit{if } M_1 \sim M_1' \in A_1\ [\,\eta\,] \end{cases}$$

$$M \sim M' \in \forall X.A_2\ [\,\eta\,] \Leftrightarrow \begin{cases} M \longmapsto^* \Lambda.M_2,\ M' \longmapsto^* \Lambda.M_2', & \textit{and} \\ M_2 \sim M_2' \in A_2\ [\,\eta[X \mapsto R]\,] & \textit{if } R\ \mathsf{Cand} \end{cases}$$

It is immediate from the form of the definition that logical similarity is closed under head expansion, given that candidates are required to be.

Lemma 3 (Zig-Zag Completeness)**.** *For each $\Delta \vdash A\ \mathsf{type}$, and each candidate assignment η for Δ, the similarity relation $\_ \sim \_ \in A\ [\,\eta\,]$ is zig-zag complete.*

Proof. By induction on the structure of A:

1. If A is a type variable X, note that $\eta(X)$ is required to be a type candidate, and hence is zig-zag complete.

2. If $A = A_1 \to A_2$. Assume that

   (a) $M \sim M' \in A\ [\,\eta\,]$,
   (b) $N \sim N' \in A\ [\,\eta\,]$, and
   (c) $N \sim M' \in A\ [\,\eta\,]$.

   It follows that

   (a) $M \longmapsto^* \lambda x.M_2$,
   (b) $M' \longmapsto^* \lambda x.M_2'$,
   (c) $N \longmapsto^* \lambda x.N_2$, and
   (d) $N' \longmapsto^* \lambda x.N_2'$.

   To show that $M \sim N' \in A\ [\,\eta\,]$, it suffices to assume that $M_1 \sim M_1' \in A_1\ [\,\eta\,]$, and show that $[M_1/x]M_2 \sim [M_1'/x]N_2' \in A_2\ [\,\eta\,]$.

   From the assumptions we have

(a) $[M_1/x]M_2 \sim [M_1'/x]M_2' \in A_2\ [\,\eta\,]$,
(b) $[M_1/x]N_2 \sim [M_1'/x]N_2' \in A_2\ [\,\eta\,]$,
(c) $[M_1/x]N_2 \sim [M_1'/x]M_2' \in A_2\ [\,\eta\,]$.

But then the result follows by the induction hypothesis that similarity at A_2 relative to η is zig-zag complete.

3. If $A = \forall X.A_2$. Assume that

(a) $M \sim M' \in A\ [\,\eta\,]$,
(b) $N \sim N' \in A\ [\,\eta\,]$, and
(c) $N \sim M' \in A\ [\,\eta\,]$.

It follows that

(a) $M \longmapsto^* \Lambda.M_2$,
(b) $M' \longmapsto^* \Lambda.M_2'$,
(c) $N \longmapsto^* \Lambda.N_2$, and
(d) $N' \longmapsto^* \Lambda.N_2'$.

To show that $M \sim N' \in A\ [\,\eta\,]$, it suffices to assume that R is a type candidate, and show that $M_2 \sim N_2' \in A_2\ [\,\eta[X \mapsto R]\,]$.

From the assumptions we have

(a) $M_2 \sim M_2' \in A_2\ [\,\eta[X \mapsto R]\,]$,
(b) $N_2 \sim N_2' \in A_2\ [\,\eta[X \mapsto R]\,]$, and
(c) $N_2 \sim M_2' \in A_2\ [\,\eta[X \mapsto R]\,]$.

But then the result follows by the inductive hypothesis that similarity at A_2 relative to $\eta[X \mapsto R]$ is zig-zag complete.

□

Exact equality of two terms in a type, $\Gamma \gg_\Delta M \doteq M' \in A$, is defined to mean that for all candidate assignments η for Δ, if $\gamma \sim \gamma' \in \Gamma\ [\,\eta\,]$, then $\widehat{\gamma}(M) \sim \widehat{\gamma'}(M) \in A\ [\,\eta\,]$. *Semantic membership*, written $\Gamma \gg_\Delta M \in A$, is defined to mean $\Gamma \gg_\Delta M \doteq M \in A$.

Lemma 4 (Compositionality)**.** *Suppose that* $\Delta \vdash B\ \mathsf{type}$ *and* $\Delta, X \vdash A\ \mathsf{type}$*. Let* η *be a relation assignment for* Δ*, and define* $E(N, N')$ *iff* $N \sim N' \in B\ [\,\eta\,]$*. Then,* $M \sim M' \in [B/X]A\ [\,\eta\,]$ *iff* $M \sim M' \in A\ [\,\eta[X \mapsto E]\,]$.

Exercise 2. *Prove Lemma 4.*

Theorem 5 (Parametricity)**.** *If* $\Gamma \vdash_\Delta M : A$*, then* $\Gamma \gg_\Delta M \in A$.

Proof. By induction on typing derivations, using Lemma 3 and 4. □

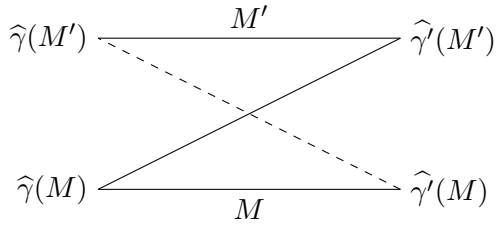

Figure 2: Symmetry via Zig-Zag Completeness

Exercise 3. *Prove the parametricity theorem.*

Theorem 6 (Soundness)**.** *If* $\Gamma \vdash_\Delta M \equiv M' : A$*, then* $\Gamma \gg_\Delta M \doteq M' \in A$

Sketch. Closure under head expansion suffices for the β principles:

1. If $\Gamma, x : A_1 \gg_\Delta M_2 \in A_2$ and $\Gamma \gg_\Delta M_1 \in A_1$, then
$$\Gamma \gg_\Delta \mathsf{ap}((\lambda x.M_2), M_1) \doteq [M_1/x]M_2 \in A_2.$$
2. If $\Gamma \gg_{\Delta, X\,\mathsf{type}} M_2 \in A_2$ and $\Delta \vdash A_1\,\mathsf{type}$, then
$$\Gamma \gg_\Delta \mathsf{Ap}(\Lambda.M_2) \doteq M_2 \in [A_1/X]A_2.$$

The definition of similarity ensures that the η principles are valid:

1. If $\Gamma \gg_\Delta M \in A_1 \to A_2$, then $\Gamma \gg_\Delta M \doteq \lambda x.\mathsf{ap}(M, x) \in A_1 \to A_2$.
2. If $\Gamma \gg_\Delta M \in \forall X.A_2$, then $\Gamma \gg_\Delta M \doteq \Lambda.\mathsf{Ap}(M) \in \forall X.A_2$.

Compatibility of exact equality with both forms of abstraction and application are easily obtained. Parametricity states that equality is reflexive. The proofs of symmetry and transitivity are given in Figures 2 and 3. □

Exercise 4. *Complete the proof of Theorem 6.*

The proof of symmetry is depicted in Figure 2. The assumptions give rise to the Z; its completion establishes the desired simulation. The proof of transitivity is depicted in Figure 3. The assumptions give rise to the emboldened Z, whose completion establishes the desired simulation.

Exercise 5. *Re-formulate the foregoing using typed terms, rather than erasure. Specifically, define the notion of a candidate* $R : A \to A'$ *being a binary relation between closed terms of closed types* A *and* A' *satisfying the closure conditions given above. Then, for* δ, δ' *being substitutions of closed types for type variables in* Δ*, define* $\eta : \delta \to \delta'$ *to be an assignment to each* $X \in \Delta$ *a candidate* $\eta(X) : \delta(X) \to \delta'(X)$*. Using these notions, reformulate the development of parametricity using typed terms.*

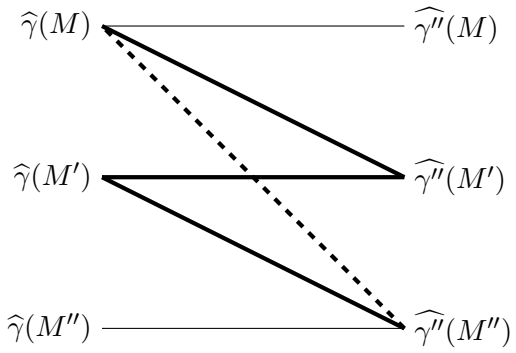

Figure 3: Transitivity via Zig-Zag Completeness

References

Robert L. Constable, Stuart F. Allen, Mark Bromley, Rance Cleaveland, J. F. Cremer, R. W. Harper, Douglas J. Howe, Todd B. Knoblock, N. P. Mendler, Prakash Panangaden, James T. Sasaki, and Scott F. Smith. *Implementing mathematics with the Nuprl proof development system.* Prentice Hall, 1986. ISBN 978-0-13-451832-9. URL `http://dl.acm.org/citation.cfm?id=10510`.

Robert Harper. *Practical Foundations for Programming Languages.* Cambridge University Press, Cambridge, England, Second edition, 2016.

Robert Harper. How to (re)invent Girard's method. Unpublished lecture note, Spring 2020a. URL `https://www.cs.cmu.edu/~rwh/courses/chtt/pdfs/girard.pdf`.

Robert Harper. How to (re)invent Tait's method. Unpublished lecture note, Spring 2020b. URL `https://www.cs.cmu.edu/~rwh/courses/chtt/pdfs/tait.pdf`.

Neelakantan R Krishnaswami and Derek Dreyer. A relationally parametric model of the calculus of constructions. *Auxilliary materials at http://www. mpi-sws. org/˜ neelk/paradep-techreport. pdf*, 2012.

Per Martin-Löf. Constructive mathematics and computer programming. In L. Jonathan Cohen, Jerzy Łoś, Helmut Pfeiffer, and Klaus-Peter 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. doi: 10.1016/S0049-237X(09)70189-2. URL `http://dx.doi.org/10.1016/S0049-237X(09)70189-2`.

John C. Reynolds. Types, abstraction and parametric polymorphism. In R. E. A. Mason, editor, *Information Processing 83, Proceedings of the IFIP 9th World Computer Congress, Paris, France, September 19-23, 1983*, pages 513–523. North-Holland/IFIP, 1983.

W. Tait. Intentional interpretations of functionals of finite type i. *Journal of Symbolic Logic*, 32: 198–212, 1967.