A Note on Typed Behavioural Differential Equations

Henning Basold[1,2], Helle H. Hansen[1,2], Jan Rutten[2,1]

[1] Radboud University Nijmegen
[2] CWI Amsterdam

"One man's observation is another man's closed book or flight of fancy."
— Willard Van Orman Quine

The aim of this short note is two-fold. Our main motivation is to widen the class of maps being definable by Behavioural Differential Equations (BDE) [8]. One way of giving semantics to BDE's is by using Structural Operational Semantics [7], but this prevents mixing induction and coinduction. Moreover, the ensuing syntactic formats have limited expressiveness, and exclude useful functions from being defined. These issues can be dealt with if we add type systems.

The second aim of this note is to advertise an observational approach to infinite structures in Type Theory. For example in Coq, coinductive types are introduced by "constructors, albeit with the power of being infinitely iterated" [3]. There is work on extending type systems with (weakly) final coalgebras (e.g. [6]), but only recently efforts have been made to bring this into theorem provers [2,1]. The use of *copatterns* in loc. cit. is a way of extending λ-calculus by BDE's. We want to study which maps one can define in this calculus and under which circumstances it leads to final coalgebras.

First of all, we search for the ingredients necessary to obtain final coalgebras. To this end, we organise the types and terms defined in [1] into a category $\mathbb{T}$. Using a fixed set TyVar of type variables, the syntax of our basic types is:

$$A, B ::= 1 \mid X \in \mathrm{TyVar} \mid A \times B \mid A + B \mid A \to B \mid \mu X.A \mid \nu X.A$$

For example, natural numbers and streams are defined by $\mathrm{N} := \mu X.(1 + X)$ and $\mathrm{Str}\, A := \nu X.(A \times X)$, respectively. More interesting may be the streams that have infinitely many entries of type A: $\mathrm{Inf}\, A\, B := \nu X.\mu Y.(A \times X + B \times Y)$.

For space reasons we do not define the terms here, but rather give a prototypical example of a definition using copatterns in Fig. 1. One should see $\mathrm{Str}\, A$ as final coalgebra for the functor $F = A \times \mathrm{Id}$ with the observation .out being the transition map. Thus for $\sigma : \mathrm{Str}\, A$ we have $\sigma.\mathrm{out} : F(\mathrm{Str}\, A)$. The head and tail of σ are obtained as $\sigma.\mathrm{out}.\pi_1 : A$ and $\sigma.\mathrm{out}.\pi_2 : \mathrm{Str}\, A$. The copatterns on the left-hand of $\mapsto$ in Fig. 1 denote the head and tail of $\mathrm{even}(\sigma)$, respectively.

We construct a category $\mathbb{T}$ with *closed* types as objects and *closed* terms of type $A \to B$ as arrows. The composition and identity arrows are defined in the obvious way, but the axioms of a category are a little trickier. We would like to use as equality the convertibility relation $\equiv$ induced by the reduction relation defined in [2]. Unfortunately, this reduction relation only contracts function application, and taking $\equiv$ as equality on terms yields neither a category, nor final coalgebras.

We fix these problems in one go, by introducing a notion of observational equivalence, which is inspired by the likewise named notion from λ-calculus. An *observation* for a type A is a closed term $O : A \to B$ for any type B, whose structure depends on A. For example, in the case of streams, $O : \mathrm{Str}\, A \to B$ is either constant or observes head and tail. Two terms $t_1, t_2 : A$ are *observationally equivalent*, $t_1 \equiv_{obs} t_2$, if $Ot_1 \equiv Ot_2$ for all observations $O : A \to B$. If we rule out non-normalising terms, for example by ensuring termination/productivity using the sized types in [1], we arrive at the desired result: sized types as objects and closed terms modulo $\equiv_{\mathrm{obs}}$ as arrows form a category $\mathbb{T}$. It has finite (co)products, and for every type A with a free variable X, there is a functor $F_A \colon \mathbb{T} \to \mathbb{T}$ given by substitution: $F_A(B) = A[B/X]$. Whether all these functors have initial algebras and final coalgebras is work in progress (it is true in the case of streams though).

$$
\begin{array}{l}
\mathrm{even} : \mathrm{Str}\, A \to \mathrm{Str}\, A = \{ \\
\quad \mathrm{even}\ \sigma\ .\mathrm{out}.\pi_1 \mapsto \sigma.\mathrm{out}.\pi_1; \\
\quad \mathrm{even}\ \sigma\ .\mathrm{out}.\pi_2 \mapsto \mathrm{even}\,(\sigma.\mathrm{out}.\pi_2.\mathrm{out}.\pi_2) \\
\}
\end{array}
$$

Fig. 1. Even positions of a stream

Next we study *which* maps can actually be defined. In the spirit of Brouwer's school of intuitionism (e.g. [4,5]), we consider here continuity as basic property. First we define a functor $\mathcal{T} \colon \mathbb{T} \to \mathbf{Set}$ by sending a type A to the set $\mathcal{T}(A)$ of its inhabiting terms modulo observational equivalence. A term $f : A \to B$ induces a map $\mathcal{T}(f) : \mathcal{T}(A) \to \mathcal{T}(B)$ by application. The sets $\mathcal{T}(A)$ carry a natural topology generated by the subbase consisting of all $U_O(t) = \{t' : A \mid Ot \equiv Ot'\}$ where $t : A$ is a term and $O : A \to B$ an observation. One can show that all $\mathcal{T}(f)$ are continuous with respect to the above topology. Finally, for example for streams, we find that the topology is the expected one: the space $\mathcal{T}(\mathrm{Str}\, A)$ is homeomorphic to a subspace of $\mathcal{T}(A)^\omega$ (product topology), provided that $\mathcal{T}(A)$ is discrete (e.g. if $A = \mathrm{N}$).

References

1. A. Abel and B. Pientka. Wellfounded recursion with copatterns: a unified approach to termination and productivity. In *ICFP*, pages 185–196, 2013.
2. A. Abel, B. Pientka, D. Thibodeau, and A. Setzer. Copatterns: Programming infinite structures by observations. In *Proceedings POPL '13*, page 27–38. ACM, 2013.
3. V. Capretta. Coalgebras in functional programming and type theory. *TCS*, 412(38):5043–5069, 2011.
4. M. P. Fourman. Notions of choice sequence. In D. v. Dalen and A. Troelstra, editors, *L.E.J. Brouwer Centenary Symposium*, page 91–105. North-Holland, 1982.
5. N. Ghani, P. Hancock, and D. Pattinson. Continuous functions on final coalgebras. *ENTCS*, 249:3–18, 2009.
6. P. Hancock and A. Setzer. *Interactive Programs and Weakly Final Coalgebras in Dependent Type Theory*, pages 115–134. Number 48 in Oxford Logic Guides. Oxford University Press, 2005.
7. B. Klin. Bialgebras for structural operational semantics: An introduction. *TCS*, 412(38):5043–5069, 2011.
8. J. J. M. M. Rutten. Behavioural differential equations: a coinductive calculus of streams, automata, and power series. *TCS*, 308(1-3):1–53, 2003.