Program extraction in computable analysis

Ulrich Berger

Swansea University, Swansea, SA2 8PP, Wales, UK
u.berger@swansea.ac.uk

Abstract. We describe a new approach to computable analysis based on inductive and coinductive definitions. Applying program extraction to proofs in this setting leads to representations of real numbers and continuous real functions by non-wellfounded trees, and to implementations of new algorithms in exact real arithmetic.

In computable analysis it is common to represent real numbers by fast Cauchy sequences of rational numbers, and continuous real functions on a compact domain by their restrictions to rational arguments together with a modulus of uniform continuity [BB85]. Hence, both, real numbers and continuous functions, are represented as type 1 objects which can, in principle, be encoded by functions on the natural numbers. Consequently, say, integration of continuous functions is a type 2 object, i.e. a function taking type 1 objects as inputs. From these considerations it seems inevitable for a formalization of computable analysis to include the formalization of functions and higher type functionals as well as a notion of computability in higher types [Sch08].

In this talk we present an alternative formalization of computable analysis that is based on a first-order theory of real numbers augmented by the possibility of defining predicates inductively and coinductively, i.e. by forming least and greatest fixed points of strictly positive set operators [Ber09,BS10b]. Algorithms for, say, exact real number computation or integration don't have to be formalized because they can be extracted automatically from constructive proofs via a realizability interpretation [BS10a].

As an example, consider a first-order theory of real numbers in the compact intervall $\mathbb{I} := [-1, 1]$. We define a coinductive predicate $\mathrm{C}_0 \subseteq \mathbb{R}$ such that $\mathrm{C}_0(x)$ expresses that x has a signed digit representation, i.e. x can be written in the form

$$x = \sum_{i=0}^{\infty} 2^{-(i+1)} * a_i$$

where $a_i \in \mathrm{SD} := \{0, 1, -1\}$. Note that within our given first-order theory, and using constructive reasoning only, it is not possible to prove that every $x \in \mathbb{I}$ has a signed digit representation. The predicate C_0 is defined as the largest subset of $\mathbb{I}$ such that for all $x \in \mathbb{I}$

$$\mathrm{C}_0(x) \to \exists i \in \mathrm{SD}\, \exists y \in \mathbb{I}\, (x = (i + y)/2 \wedge \mathrm{C}_0(y))$$

In a suitable realizability interpretation a realizer of $\mathrm{C}_0(x)$ will be an infinite stream of signed digits providing a signed digit representation of x. One can now, for example, prove (constructively, using coinduction) that the predicate C_0 is closed under the average function. Program extraction yields an implementation of the average function of real numbers w.r.t. to the signed representation. To reiterate the point made above regarding formalization, note that, although the extracted program operates on infinite streams, in our formal system no infinite streams or computation principles for such streams are needed.

Similarly to real numbers, on can describe real continuous functions on $\mathbb{I}$ by a combination of an inductive and a coinductive predicate C_1. This provide, via realizability, a representation of continuous functions by non-wellfounded trees. We will sketch a proof that the definite integral of a function in C_1 lies in C_0, and discuss the extracted integration algorithm.

We will conclude the talk by discussing ongoing work on an extension of the coinductive approach above to a new coinductive model for functionals in all finite types.

References

[BB85] E. Bishop and D. Bridges. *Constructive Analysis.* Grundlehren der mathematischen Wissenschaften 279. Springer, Berlin, Heidelberg, NewYork, Tokyo, 1985.

[Ber09] U. Berger. From coinductive proofs to exact real arithmetic. In E. Grädel and R. Kahle, editors, *Computer Science Logic*, volume 5771 of *Lecture Notes in Comput. Sci.*, pages 132–146. Springer, 2009.

[BS10a] U. Berger and M. Seisenberger. Program extraction via typed realisability for induction and coinduction. In R. Schindler, editor, *Ways of Proof Theory*, Ontos Series in Mathematical Logic, pages 157–181. Ontos Verlag, Frankfurt, 2010.

[BS10b] U. Berger and M. Seisenberger. Proofs, programs, processes. In F. Ferreira, B. Löwe, E. Mayordomo, and L. M. Gomes, editors, *Programs, Proofs, Processes, 6th Conference on Computability in Europe, CiE 2010, Ponta Delgada, Azores, Portugal, June 30 - July 4, 2010. Proceedings*, volume 6158 of *Lecture Notes in Comput. Sci.*, pages 39–48, 2010.

[Sch08] H. Schwichtenberg. Realizability interpretation of proofs in constructive analysis. *Theory Comput. Sys.*, 43(3-4):583–602, 2008.