Fernando Lucatelli Nunes
Utrecht University & CMUC

CHAD: elementary categorical aspects of automatic differentiation

Automatic differentiation (AD) is a technique for computing derivatives that has long been popular in the numerical methods community. There has recently been an increase in research in AD by the *programming languages (PL)* community.

The primary source of interest of the PL community comes from the fact that automatic differentiation is one of the fundamental cornerstones of machine learning and scientific computing since it is usually the method of choice for computing derivatives in these two contexts. In this direction, the main goals of the community have been the following.

- Giving concise, clear and easy-to-implement versions of AD algorithms;
- expanding the languages and programming techniques that AD can be applied to;
- relating AD to its mathematical foundations and proving that AD implementations correctly calculate derivatives;
- performing AD at compile-time, to avoid interpreter overhead and make the most of compiler optimizations;
- providing formal complexity guarantees for AD implementations.

Automatic differentiation comes in two modes, the reverse mode and the forward one. Because of its efficiency, the former is ultimately more useful in contexts where the dimension of the domain is way bigger than the dimension of the codomain. This is the case for many of the applications in machine learning. Not surprisingly, the reverse mode is the most difficult to understand and implement, due to its reversal of the execution order of the program.

Combinatory Homomorphic Automatic Differentiation (CHAD) is our approach to addressing all the points above. Among the nice features of CHAD, the reverse mode becomes the categorical dual of the forward mode. Moreover, and this is the main point of the talk, the definitions of the derivatives follow the universal properties, being uniquely defined. This leads to concise and clear *AD* for functional languages, also leading to neat (categorical) correctness proofs.

In this talk, we stress the elementary categorical aspects of this work. Under this elementary categorical perspective, the main problem is to extend the definition of a derivative to various (PL-relevant) categorical structures (starting with bicartesian closed categories). We define this in a unique way, following our nose, or, more precisely, the universal property of the syntax. In the semantic side of the story, this gives us a definition up to descent data. However, we can show that this, indeed, does not depend on the descent data when it matters. Moreover, it is correct, meaning that it is consistent with our usual *Fréchet* definition between cartesian spaces, provided that we started assigning correct derivatives to our primitive morphisms.

Most of the talk intends to be accessible to people with basic knowledge in category theory, such as the definition of graphs, cartesian closed categories, comma categories, etc. Naturally, no knowledge of programming languages or denotational semantics will be assumed, and we will not address any problem requiring some knowledge in these topics.

This work is part of a joint project with Matthijs Vákár and Tom Smeding. The talk will be mostly about [2] and [3]. But it will be heavily influenced by [5], where CHAD for higher-order functional languages is introduced, [4], [1] and ongoing work.

References:

[1] Implementation: https://github.com/VMatthijs/CHAD

[2] Fernando Lucatelli Nunes and Matthijs Vákár. CHAD for Expressive Total Languages. arXiv:2110.00446. Preprints CMUC/DMUC 21-36. 2021.

[3] Fernando Lucatelli Nunes and Matthijs Vákár. Foward-Mode Automatic Differentiation for Iteration and Recursion. In preparation. Hopefully, soon.

[4] Matthijs Vákár and Tom Smeding. CHAD: Combinatory Homomorphic Automatic Differentiation. 2021.

[5] Matthijs Vákár. Reverse AD at higher types: Pure, principled and denotationally correct. In ESOP, pp. 607–634. 2021.