# A semantical analysis of structural recursion

Andreas Abel, Thorsten Altenkirch

February 1st, 1999

We are developing a system (MuTTI - Munich Type Theory Implementation) with dependent types which can be used for the development of provably correct programs in Type Theory. Inspired by Coquand's pattern matching for dependent types [Coq92] as implemented in the ALF system [Alf94] and its successors, we define a total language as a subset of a partial one. Hence, we are faced with the problem of verifying termination.

We restrict ourselves to structural recursion, where by structural recursion we mean that the only termination orderings we consider are lexical products of the natural structural ordering on a strictly positive datatype. We also allow mutual recursion. A further restriction is that smaller terms are only generated by primitive operators like case-analysis, projection and application.

In the type-theoretic context this is sufficient, since general terminating recursion can be represented by adding additional (computationally irrelevant) parameters. We are not sure whether structural recursive without the afore mentioned restriction is actually decidable.

Abel implemented a termination checker for a simply typed sublanguage of MuTTI (called foetus), this system allows mutual recursive definitions on general strict positive datatypes and returns a lexical ordering on the arguments of the function if one exists.

In the work we want to present here, we show that from foetus output we can actually conclude that the function terminates. We define a semantic interpretation of each type and formally define the structural ordering on semantic values of possibly different types. A central result is that the structural ordering is wellfounded at each type. A general soundness theorem allows us to conclude that each term accepted by the foetus system actually terminates.

Our approach is related to the work by Telford & Turner, who are interested in a total functional programming language (ESFP). In a recent (unpublished) article [TTu98b] they present also a termination analysis based on abstract interpretation. It seems that they accept a larger set of functions but that our analysis of the output of the termination checker is more detailed.

Our next goal is to extend our termination checker to coinductively defined types [Coq93, TTu97b] and dependent types, including the definition of universes. We also hope to be able to answer the question whether the restriction of structural recursive function is actually necessary.

# References


[Alf94] Thorsten Altenkirch, Veronica Gaspes, Bengt Nordström, and Björn von Sydow. *A user's guide to ALF*. Department of Computing Science, University of Göteborg/Chalmers, http://www.cs.chalmers.se/Cs/Research/Logic/alf/guide.html, May 1994.

[Coq92] Thierry Coquand. Pattern matching with dependent types. *(to be updated)*, 1992.

[Coq93] Thierry Coquand. Infinite objects in type theory. In Henry Barendregt and Tobias Nipkow, editors, *Types for Proofs and Programs (TYPES '93)*, volume 806 of *Lecture Notes in Computer Science*, pages 62–78. Springer-Verlag, 1993.

[TTu97b] Alastair Telford and David Turner. Ensuring Streams Flow. In Michael Johnson, editor, *Algebraic Methodology and Software Technology, 6th International Conference, AMAST '97, Sydney Australia, December 1997*, volume 1349 of *Lecture Notes in Computer Science*, pages 509–523. AMAST, Springer-Verlag, December 1997.

[TTu98b] Alastair Telford and David Turner. Guarded Recursion in ESFP. *Submitted to TYPES '98 proceedings*, 1998.