COMBINATORY LOGIC, A BRIDGE TO VERIFIED PROGRAMS

ALBERT HOOGEWIJS AND PIETER AUDENAERT

To facilitate program development, researchers and practitioners enhanced and invented new programming languages, all with the same goal: striving to balance five objective criteria: user acceptance rate, program correctness, development speed, execution speed, and program compactness. In [1] the authors introduced Alfred ("Another Language for Functional REDuction"), a new programming language based on untyped combinatory logic (CL). The main emphasis lies on program correctness, development speed and program compactness. The whole language is built from the two basic combinators S and K that satisfy the axioms $Sxyz = xz(yz)$ and $Kxy = x$ for all x, y and z in an equational theory as defined in [2]. This lazy functional language is shown to be Turing-complete. An Alfred program can be compiled to binary code and executed, and to a PVS specification and proved to be partial correct using PPHOL (see [3]).

REFERENCES

[1] P. Audenaert and A. Hoogewijs, *Alfred: Formal Lazy Functional Reduction*

[2] H. P. Barendregt, *The Lambda Calculus: Its Syntax and Semantics.* North-Holland, 1984.

[3] A. Hoogewijs and P. Audenaert, *Implementing Undefinedness in a Two-valued Prooftool through a Four-valued Kleene Logic*, The Bulletin of Symbolic Logic Volume 9, 2003 pp 92-93

GHENT UNIVERSITY, GALGLAAN 2, B-9000 GENT, BELGIUM

E-mail: bh@cage.ugent.be, paudenae@cage.ugent.be (Supported by research grant of UGent, BOF 011D0900)