Synchronous Programming in Intermediate Constructive Logic

Michael Mendler[1]

Otto-Friedrich Universität, Bamberg:

Abstract. Cyclic and concurrent logical programs, such as those arising in asynchronous circuits or in the declarative semantics of synchronous programming languages, do not admit unique canonical execution semantics. Instead, different approaches impose different restrictions on stabilization behavior to define different classes of synchronous step reactions. Typically, these are defined by operational semantics based on scheduling policies. This is good for implementations. However, it flies in the face of the beauty of declarative programming, which is to give purely logical meaning to computations. Can we regain logical sense in synchronous reactions? We argue this is possible by replacing operational stability in the classical Boolean semantics by logical stability in intermediate constructive logics. This tentative talk will report some results in this direction, featuring intermediate constructive logics related to Gödel, Dummett and Maximova.