conditions under which it is coherent to deny that sentence. The dissertation develops a theory of quantification as marking coherent ways a language can be expanded and modality as the means by which we can reflect on the norms governing the assertion and denial conditions of our language. If the view of quantification that is argued for is correct, then there is no tension between second-order quantification and nominalism. In particular, the ontological commitments one can incur through the use of a quantifier depend wholly on the ontological commitments one can incur through the use of atomic sentences. The dissertation concludes by applying the developed theory of meaning to the metaphysical issue of necessitism and contingentism. Two objections to a logic of contingentism are raised and addressed. The resulting logic is shown to meet all the requirement that the dissertation lays out for a theory of meaning for quantifiers and modal operators.

Abstract prepared by Andrew Parisi
E-mail: andrew.p.parisi@gmail.com
URL: https://opencommons.uconn.edu/dissertations/1480/

Sewon Park, ***Continuous Abstract Data Types for Verified Computation***, KAIST, South Korea, 2021. Supervised by Martin Ziegler. MSC: 68N15, 03F60, 03B70. Keywords: real number computation, continuous abstract data type, computable analysis, imperative programming, formal verification.

Abstract

We devise imperative programming languages for verified real number computation where real numbers are provided as abstract data types such that the users of the languages can express real number computation by considering real numbers as abstract mathematical entities. Unlike other common approaches toward real number computation, based on an algebraic model that lacks implementability or transcendental computation, or finite-precision approximation such as using double precision computation that lacks a formal foundation, our languages are devised based on computable analysis, a foundation of rigorous computation over continuous data. Consequently, the users of the language can easily program real number computation and reason about the behaviours of their programs, relying on their mathematical knowledge of real numbers without worrying about artificial roundoff errors. As the languages are imperative, we adopt precondition–postcondition-style program specification and Hoare-style program verification methodologies. Consequently, the users of the language can easily program a computation over real numbers, specify the expected behaviour of the program, including termination, and prove the correctness of the specification. Furthermore, we suggest extending the languages with other interesting continuous data, such as matrices, continuous real functions, et cetera.

Abstract taken directly from the thesis.
E-mail: sewonpark17@gmail.com
URL: https://sewonpark.com/thesis

https://doi.org/10.1017/bsl.2021.51 Published online by Cambridge University Press