Relational Reasoning for Verified Reiterative Implementations of Multivalued Real Computations

Hyunwoo Lee[1], Sewon Park[2]

[1]Deptartment of Computer Science and Engineering, Seoul National University – Republic of Korea

[2]Graduate School of Informatics, Kyoto University – Japan

We recall relational program logic [Fra83, MHRVM19] for ordinary programming languages over intervals with discrete, multi-precision endpoints, to verify the correctness of reiterative implementations of exact real number computations, such as iRRAM [Mül00]. The C++ implementation of iRRAM introduces a so-called multivalue cache, consisting of shared stores preserved and accessed throughout reiterations. It is used to achieve consistency in multivalued computation: when a computation is repeated with higher precision, the computation path, including IO values, must remain consistent. Such consistency is a crucial property of exact real number computation. It justifies hiding internal representations and reiterations, allowing users to reason about real numbers based on the familiar structure of abstract real numbers [BCZ22, PBC$^+$24, BPS24]. We show that reiteration consistency can be encoded as a relational property, and propose relational correctness, together with reasoning invariant over the multivalue cache, as a plausible approach to verifying whether implementations truly achieve this consistency.

We further show how our framework for reasoning about implementation correctness can be extended to arbitrary continuous data types (e.g., [LLPZ19, HP20]) and their representations in the sense of computable analysis [Wei00]. We claim that this is particularly useful when a complex data type requires native low-level support for performance reasons.

It is addressed in [PT23] that reiterative implementations of real computations, in general, fail to provide modularity when, for example, c_1 and c_2 interfere due to a shared multi-valued cache. In such cases, the limit behavior of $c_1; c_2$ may not be identical to the composition of the two. We overcome this problem by enforcing strong physical separation in the cache to allow program composition. Easing this condition into weaker logical separation remains a main direction for future work.

References

[BCZ22] Franz Brauße, Pieter Collins, and Martin Ziegler. Computer science for continuous data. In François Boulier, Matthew England, Timur M. Sadykov, and Evgenii V. Vorozhtsov, editors, *Computer Algebra in Scientific Computing*, pages 62–82, Cham, 2022. Springer International Publishing.

[BPS24] Andrej Bauer, Sewon Park, and Alex Simpson. An imperative language for verified exact real-number computation. *arXiv preprint arXiv:2409.11946*, 2024.

[Fra83] Nissim Francez. Product properties and their direct verification. *Acta Informatica*, 20(4):329–344, Dec 1983.

[HP20] Jiman Hwang and Sewon Park. Compact subsets in exact real computation. *KIISE Conference Proceedings*, pages 1104–1106, 2020.

[LLPZ19] Seokbin Lee, Donghyun Lim, Sewon Park, and Martin Ziegler. Grassmannian as continuous data type with computable semantics. *KIISE Conference Proceedings*, pages 1767–1769, 2019.

[MHRVM19] Kenji Maillard, Cătălin Hriţcu, Exequiel Rivas, and Antoine Van Muylder. The next 700 relational program logics. *Proc. ACM Program. Lang.*, 4(POPL), December 2019.

[Mül00] Norbert Th Müller. The iRRAM: Exact arithmetic in C++. In *International Workshop on Computability and Complexity in Analysis*, pages 222–252. Springer, 2000.

[PBC$^+$24] Sewon Park, Franz Brauße, Pieter Collins, SunYoung Kim, Michal Konečný, Gyesik Lee, Norbert Müller, Eike Neumann, Norbert Preining, and Martin Ziegler. Semantics, Specification Logic, and Hoare Logic of Exact Real Computation. *Logical Methods in Computer Science*, Volume 20, Issue 2, June 2024.

[PT23] Sewon Park and Holger Thies. Towards verified implementation of iterative and interactive realram. In *Proceedings of the Twentieth International Conference on Computability and Complexity in Analysis (CCA 2023)*, Dubrovnik, Croatia, September 2023. Conference talk, Sep 07–09, 2023.

[Wei00] K. Weihrauch. Computable analysis. Springer, Berlin, 2000.