Available online at www.sciencedirect.com

ScienceDirect

Electronic Notes in Theoretical Computer Science 317 (2015) 101–107

Electronic Notes in Theoretical Computer Science

www.elsevier.com/locate/entcs

A Parameterized Floating-Point Formalizaton in HOL Light

Charles Jacobsen[a,1,2] Alexey Solovyev[a,3,5]
Ganesh Gopalakrishnan[a,4,5]

[a] *School of Computing*
University of Utah
Salt Lake City, USA

Abstract

We present a new, open-source formalization of fixed and floating-point numbers for arbitrary radix and precision that is now part of the HOL Light distribution [10]. We prove correctness and error bounds for the four different rounding modes, and formalize a subset of the IEEE 754 [1] standard by gluing together a set of fixed-point and floating-point numbers to represent the subnormals and normals. In our floating-point proofs, we treat phases of floating-point numbers as copies of fixed-point numbers of varying precision so that we can reuse fixed-point rounding theorems.

Keywords: IEEE-754-2008, floating point, fixed point, formalization

1 Introduction

Programmers need tools that calculate error bounds and automate the tedious and error-prone reasoning involved in proofs of correctness for their floating-point algorithms. They follow the IEEE 754 standard because most hardware and software libraries support it and provide fast implementations with multiple levels of precision. As examples of adoption of this standard, four levels of precision are available on Intel CPUs [13] and three on Nvidia GPUs [2], and decimal floating-point numbers are in a package developed by Cornea, et al. [7].

But it is hard to reason about floating-point algorithms because floating-point numbers and operations are an approximate model of the real numbers and do not

[1] Supported in part by NSF Research Experience for Undergraduates (REU), Award CCF 1430497.
[2] Email: `charlesj@cs.utah.edu`
[3] Email: `monad@cs.utah.edu`
[4] Email: `ganesh@cs.utah.edu`
[5] Support in part by NSF Awards CCF 1421726 and 7298529.

http://dx.doi.org/10.1016/j.entcs.2015.10.010
1571-0661/© 2015 The Authors. Published by Elsevier B.V.

share desirable properties like associativity that their real-valued counterparts have. Programmers may miss edge cases that produce large errors, miss ways to improve accuracy by re-ordering operations, or use a precision that is too conservative. For example, Goualard [9] shows that Intlab V5.5 [14] reports μ as the midpoint of the interval $[-\mu, \mu]$ instead of 0, where μ is the smallest subnormal number. While the designers of Intlab V5.5 may have been aware of this edge case, it shows how subtle the errors in floating-point algorithms can be.

We present a new formalization of fixed- and floating-point numbers that can be used to reason about programs that use IEEE floating-point numbers. [6] Our work is motivated by John Harrison's formalization in HOL Light for binary floating-point [11]. To our knowledge, this formalization is not publicly available and doesn't use a fixed- point theory as a basis for floating-point theory, as ours does. Daumas, Rideau, and Théry [8] and Boldo and Melquiond [4] have developed advanced formalizations of fixed and floating point for arbitrary radix and precision in Coq. We chose not to use the Coq formalizations nor translate them to HOL Light since these formalizations are large and complicated and our immediate need was to formalize only a subset of fixed- and floating-point theory. HOL Light also provides the non-constructive Hilbert choice operator, which makes the formalization easier to write since we aren't trying to compute floating-point numbers in our formalization. Other formalizations for binary floating point have been done in Z, PVS, HOL4, Isabelle/HOL, and ACL2 [3,6,5,12,16,15]. To our knowledge, the Z formalization is not publicly available. Second, the theorems in the Isabelle/HOL formalization are for single precision only, and would not generalize easily. Finally, our formalization does not rely on higher level operations and corresponding theories, like real-valued `floor` as in ACL2.

2 Formalization Overview

We model IEEE floating-point numbers as a subset of $\mathbb{R}$. Subnormal numbers and zero are represented using a set of real numbers that are a fixed distance apart (fixed-point numbers). Normal numbers are represented using a subset of an infinite set of real numbers that are varying distances apart (floating-point numbers). Finally, we take any real number whose magnitude is above a carefully chosen threshold to be floating-point infinity. We do not model signed zero or NaNs. Each set is explained in more detail in the following sections.

We have defined rounding for fixed- and floating-point numbers in the four rounding modes (round to nearest with ties to even, round to zero, round to positive infinity, and round to negative infinity), and proved fundamental properties of rounding, like the $(1 + \delta)$ error rule.