Constructive Analysis in Nuprl

Robert Constable

November 5, 2019

Abstract

These notes discuss the implementation of Errett Bishop and Douglas Bridges' account of the real numbers from their book *Constructive Analysis* [4]. Many of the concepts are formalized in Nuprl by Dr. Mark Bickford and displayed on the PRL group home page with permission of the publisher. See this url:
http://www.nuprl.org/MathLibrary/ConstructiveAnalysis/

In the display of results from *Constructive Analysis* Dr. Bickford has created links to his formalization of the definitions and results in Nuprl. In addition he created an index of the formal definitions and theorems in Nuprl and linked those to the corresponding account in the Bishop and Bridges book.

These notes also discuss elements of Brouwer's *Intuitionistic Analysis.* Bishop and Bridges use constructive logic which includes some elements of Brouwer's logic, but they do not adopt Brouwer's deeper insights about space, time, and the real numbers. These include the notion of *free choice sequences*, the *Continuity Principle*, and *Bar Induction* among others.

1 Bishop's Constructivist Manifesto and Brouwer's Intuitionism

In his book *Foundations of Constructive Analysis* [3] Bishop provides his philosophical perspective on constructive mathematics. He cites Brouwer in part to agree with him and in part to present variants on Brouwer's approach. Oddly Bishop's critique of Brouwer led the Cornell PRL group to better appreciate Brouwer and in due course augment Bishop and Bridges constructive approach with some of Brouwer's deepest and most useful concepts from his intuitionism. In a series of five articles over a period of five years we have implemented Brouwer's *intuitionistic analysis* as an extension of the Bishop and Bridges approach. We will discuss some of the differences here.

2 Calculus and the Real Numbers in Nuprl

Bishop and Bridges use the word "set" for what we call a type. Bishop says on page 5:"a set exists only when it has been defined. To define a set we prescribe what we must do to construct an element of a set, and what we must do in order to show that two members of a set are equal." This is almost verbatim the constructive account of a type. So these notes will use the word *type*.

In Chapter 2 page 15 Bishop and Bridges define the notion of an *operation*. They say that an operation is a *finite mechanical procedure*. They write $f : (A \to B)$ to indicate that f maps elements from the set A into set B. If the operation f respects the equality on the set A, i.e. if $a = b$ in A implies $f(a) = f(b)$ then f is called a *function*. This wording is essentially the same as used by logicians. They define a *sequence* as a function whose domain is the set of positive integers.

It is noteworthy that Brouwer proposed a wider notion of real numbers. He allows them to be given by *free choice sequences* [6, 7, 8] in addition to operations. The choices used to generate the sequence need not be given by a rule or an *algorithm*. The values can be freely chosen.

Definition 2.1: A sequence of rational numbers q_n is *regular* if $|q_m - q_n| \leq 1/m + 1/n$ for all m, n positive natural numbers. A *real number* is a regular sequence of rationals. We call q_n the nth *approximation* to the real number.

Two real numbers x_n, y_n are *equal* iff $|x_n - y_n| \leq 2/n$. The type of real numbers just defined is denoted by $\Re$. It is easy to see that equality on real numbers is an equivalence relation.

Two real numbers are *separated*, $x \neq y$ if and only if $x < y$ or $y < x$.

The arithmetic operations on reals are easy to define, and they make intuitive sense. Here are the definitions. We call x_n the n-th approximation to the real number.

1. $x + y = (x_{2n}) + (y_{2n})$ for each natural number n.
2. $xy = (x_{2kn}) \times (y_{2kn})$ for all n.
3. $max(x, y) = max(x_n, y_n)$ for all n.
4. $-x = (-x_n)$ for all n.
5. $\alpha^\star = (\alpha, \alpha, \alpha, ...)$.

It might be more clear to have an operation symbol for the product of two real numbers. When that is helpful we will write $x \star y$.

Proposition Each of the above five sequences of rational numbers defines a real number. We also associate with each real number x an integer K_x called the *canonical bound* for x. We define it as the least integer K_x such that the absolute value of the $n_t h$ approximation is less than K_x.

Definition A real number (x_n) is *positive*, writing $\Re^+$, if $x_n \geq 1/n$ for some positive n. A real is *nonnegative*, say $x_n \epsilon \Re^{0+}$ if and only if $x_n \geq -n^{-1}$ for n a positive integer.

Here is an important theorem about real numbers that is similar to Cantor's theorem. It shows that if we have an *enumeration* of reals, say (a_n) and two reals x_0 and y_0 such that $x_0 < y_0$, then we can find a real number x such that $x_0 \leq x \leq y_0$ moreover, $x \neq a_n$ for all positive integers n.

Theorem 2.19. Let (a_n) be a sequence of real numbers, and let x_0 and y_0 be real numbers such that $x_0 < y_0$. Then we can find a real number x such that $x_0 \leq x \leq y_0$ and $x \neq a_n$ for all natural numbers n.

The proof of **Theorem** 2.19 on page 27 is formalized in Nuprl by Mark Bickford. The verbatim account from Bishop and Bridges is given in the part of their book which is posted on-line at the Nuprl web site. The url for the book is given above in the abstract. Here is an English language version of the theorem.

Theorem 2.19 in English: Given a sequence of real numbers $a_1, a_2, a_3, ...$ and given a proper non-empty closed interval of the reals, we can effectively find another real number x in this interval, different from all of the a_i.

This is a constructive version of *Cantor's Theorem* that the real numbers are uncountable in the sense that we can find a real number a not on this list.

3 Proof Assistants

The thirty five year steady increase in the effectiveness of *proof assistants* has brought them into classrooms, university research groups, and industrial labs around the globe. At Cornell we designed, built, maintain, and continue to extend the *Nuprl* proof assistant [1, 9, 15]. We were inspired by de Bruijn's *Automath* [10] and by the *Edinburgh LCF system* [12], and Sir Tony

Hoare's work on data structures [14]. Moreover proof assistants are a harbinger of something *broader and more impactfull* that we discuss in this article. Continuing advances in proof assistant design and implementation will become "game changing." In this effort, the US and EU will continue their highly productive cooperation, dating back to de Bruijn's *Automath*, to the creation and deployment of of modern proof assistants, such as *Agda* [5], *Coq* [2], *HOL* [11, 13], and *Nuprl* among others. [1]

[1]There is no complete list of proof assistants, but a Wikipedia page on proof assistants lists 14 of them as of 2018. More are under construction.

References

[1] J. L. Bates and Robert L. Constable. Proofs as programs. *ACM Transactions of Programming Language Systems*, 7(1):53–71, 1985.

[2] Yves Bertot and Pierre Castéran. *Interactive Theorem Proving and Program Development; Coq'Art: The Calculus of Inductive Constructions*. Texts in Theoretical Computer Science. Springer-Verlag, 2004.

[3] E. Bishop. *Foundations of Constructive Analysis*. McGraw Hill, NY, 1967.

[4] E. Bishop and D. Bridges. *Constructive Analysis*. Springer, New York, 1985.

[5] Ana Bove, Peter Dybjer, and Ulf Norell. A Brief Overview of Agda – a functional language with dependent types. In Stefan Berghofer, Tobias Nipkow, Christian Urban, and Makarius Wenzel, editors, *LNCS 5674, Theorem Proving in Higher Order Logics*, pages 73–78. Springer, 2009.

[6] L.E.J. Brouwer. Intuitionism and formalism. *Bull Amer. Math. Soc.*, 20(2):81–96, 1913.

[7] L.E.J. Brouwer. Über definitionsbereiche von funktionen. *Mathematische Annalen*, 97:60–75, 1927.

[8] L.E.J. Brouwer. Intuitionism and formalism. In P. Benacerraf and H. Putnam, editors, *Philosophy of mathematics: selected writings*. Cambridge University Press, 1983.

[9] Robert L. Constable, Stuart F. Allen, H. M. Bromley, W. R. Cleaveland, J. F. Cremer, R. W. Harper, Douglas J. Howe, T. B. Knoblock, N. P. Mendler, P. Panangaden, James T. Sasaki, and Scott F. Smith. *Implementing Mathematics with the Nuprl Proof Development System*. Prentice-Hall, NJ, 1986.

[10] N. G. de Bruijn. The mathematical language Automath: its usage and some of its extensions. In J. P. Seldin and J. R. Hindley, editors, *Symposium on Automatic Demonstration*, volume 125 of *Lecture Notes in Mathematics*, pages 29–61. Springer-Verlag, 1970.

[11] Michael Gordon and Tom Melham. *Introduction to HOL: A Theorem Proving Environment for Higher-Order Logic*. Cambridge University Press, Cambridge, 1993.

[12] Michael Gordon, Robin Milner, and Christopher Wadsworth. *Edinburgh LCF: a mechanized logic of computation*, volume 78 of *Lecture Notes in Computer Science*. Springer-Verlag, NY, 1979.

[13] John Harrison. HOLLight: A tutorial introduction. In *Formal Methods in Computer-Aided Design (FMCAD'96)*, volume 1166 of *Lecture Notes in Computer Science*, pages 265–269. Springer, 1996.

[14] C. A. R. Hoare. Notes on data structuring. In *Structured Programming*. Academic Press, New York, 1972.

[15] Robert Constable Mark Bickford, Vincent Rahli. The good, the bad and the ugly. Technical report, Cornell University, 2017.