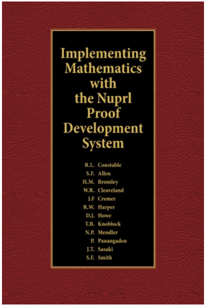
Formalized Brouwerian Real Analysis using the Nuprl proof assistant

Mark Bickford

Cornell University, Computer Science

November 11, 2016

Introduction



Implementing Mathematics with The Nuprl Proof Development System
By the PRL Group:
R. L. Constable, S. F. Allen,
H. M. Bromley, W. R. Cleaveland,
J. F Cremer, R. W. Harper, D. J. Howe,
T. B. Knoblock, N. P. Mendler,
P. Panangaden, J. T Sasaki, S. F. Smith

- Inspired by the work of Errett Bishop and with NSF funding to implement *proofs-as-programs*, Bob Constable and his group at Cornell developed Nuprl, a proof assistant now based on extensional, constructive type theory.
- We have formalized all of Chapter 2 of Bishop and Bridges, *Constructive Analysis* in Nuprl.

Brouwerian Nuprl

- From studying Kleene we became convinced that both bar induction and the continuity principle should be true in Nuprl and that they would be very useful for formalizing constructive analysis.
- In the last few years we have formalized the semantics of Nuprl in Coq. This allowed us to extend Nuprl in new ways and formally confirm our conviction.
  - We added named exceptions and a "fresh" name binding operation, and proved rules for reasoning about them. These allow us to prove a *strong continuity principle* for Nuprl.
  - We added a version of *free choice sequences* to the Nuprl semantics. This allows us to justify a strong *bar induction* rule, and use that rule to prove a general form of *bar recursion*.
- Using these new features we prove Brouwer's theorem that all real functions on a proper, compact interval are continuous. This allows us to simplify several aspects of our formalization of *Constructive Analysis*.

Nuprl in a nutshell

Nuprl built its own programming logic and later evolved towards the work of Per Martin-Löf. The basic concept is $x = y \in T$ where x and y are *terms* and T is a *type*.

Computation comes first: the terms are definitional extensions of a primitive, untyped programming language that includes

- numbers and tokens: $\ldots, -1, 0, 1, 2 \ldots,$ 'abc', ...
- $\lambda x.t$, $\langle t_1, t_2 \rangle$, $\mathit{inl}\ t$, $\mathit{inr}\ t$, $+, *, -, \mathit{div}, \mathit{rem}$
- $t_1(t_2)$, $\mathit{spread}(t_1; a, b.t_2)$, $\mathit{decide}(t_1; a.t_2; b.t_3)$, $\mathbf{fix}\ F$
- exceptions, try?catch, $\nu x.t$, and several more

We then define $t_1 \mapsto t_2$ (*computes to*) and $t_1 \sim t_2$ (computational bi-simulation).

A *type* T "is" a partial equivalence relation R_T on terms that respects the bi-simulation relation. Then $x \in T$ if $(x\ R_T\ x)$ and $x = y \in T$ if $(x\ R_T\ y)$.

Every type is a member of a *universe* $\mathbb{U}_i$ where $i \in 0, 1, 2, \ldots$

Nuprl is an extensional type theory

- Nuprl is *extensional*: if $x = y \in A$ and $A = B \in \mathbb{U}_i$ then $x = y \in B$.
- Nuprl has many types: quotients $x, y.T//E(x, y)$, intersection $\bigcap_{x:A} B(x)$, "partial" types $\overline{T}$ where **fix** $(\lambda x.x) = \perp \in \overline{T}$ ...
- The same term can be proved to have many different types.
- Type membership is undecidable, so there is no type-checking algorithm.
- Function extensionality:
  $f = g \in x : A \to B(x) \Leftrightarrow \forall x : A.\ f(x) = g(x) \in B(x)$
- Nuprl is expressive enough to formalize category theory and (the semantics of) homotopy type theory – viz. Cubical type theory.

Squashing

- The "set" type $\{x\colon T \mid P(x)\}$ is the subtype of $x \in T$ for which $P(x)$ is true.
  - A member x of $\{x\colon T \mid P(x)\}$ does not "come with" a proof of $P(x)$.
  - This type is useful mainly when $P(x)$ is "squash stable" – we can construct a witness for $P(x)$ from x when we know only that $P(x)$ is true (i.e inhabited).
  - This allows us to omit unneeded evidence.
- $\downarrow T$, the "squash" of type T is $\{x : \mathit{Unit} \mid T\}$
  - $\downarrow T$ is inhabited if and only if T is inhabited, but it contains no other information (about what terms inhabit T).

Truncation

- $\downharpoonright T$, the “truncation” of T is the quotient type $T//\textbf{true}$
  - The inhabitants of $\downharpoonright T$ are the same as the inhabitants of T, but they are all equal.
  - $\downharpoonright T$ can express the existence of something of type T that is not extensional w.r.t. its parameters.
- The *choice principle* for type T is
  $\forall P : T \to \mathbb{P}.\ (\forall x : T.\ \downharpoonright P[x]) \Leftrightarrow \downharpoonright (\forall x : T.\ P[x])$
- In Nuprl we can prove
  - The choice principle is true for $\mathbb{N}$ and for $\mathbb{N} \to \mathbb{N}$
  - The choice principle is false for $(\mathbb{N} \to \mathbb{N}) \to \mathbb{N}$

How does Nuprl prove Brouwer's theorem?

- It satisfies both weak and strong forms of the *continuity principle* for numbers.
  - Let $\mathcal{B} = \mathbb{N} \to \mathbb{N}$ and $\mathbb{N}_n = \{0, 1, \ldots n-1\}$
  - If $F \in \mathcal{B} \to \mathbb{N}$ then

weak $\forall f : \mathcal{B}. \downarrow\exists n : \mathbb{N}.\ \forall g : \mathcal{B}.\ F(g) = F(f)$ if $(f = g \in \mathbb{N}_n \to \mathbb{N})$

strong $\downarrow\exists M : n{:}\mathbb{N} \to (\mathbb{N}_n \to \mathbb{N}) \to (\mathbb{N} \bigcup Unit).\ \forall f{:}\mathcal{B}.\ (\downarrow\exists n : \mathbb{N}.\ M(n, f) = F(f)) \wedge (\forall n{:}\mathbb{N}.\ M(n, f) = F(f)\ \textit{if}\ M(n, f) \in \mathbb{N})$

- Nuprl has two (and only two) induction principles
  - Induction on $\mathbb{N}$
  - Bar Induction
- Using Bar Induction we prove FAN, and from FAN and the strong continuity principle we prove Brouwer's theorem that all functions from $[0, 1] \to \mathbb{R}$ are uniformly continuous.

The constructive real numbers

$$
\begin{array}{rcl}
\mathbb{R} & = & \{r:\mathbb{N}^+ \to \mathbb{Z} \mid \\
 & & \quad \forall n, m:\mathbb{N}.\ |n * r(m) - m * r(n)| \leq 2(n+m)\ \} \\
\text{then} & & |\frac{r(n)}{2n} - \frac{r(m)}{2m}| \leq \frac{1}{n} + \frac{1}{m} \\
\text{so} & & \lambda n.\frac{r(n)}{2n} \text{ is Bishop's regular} \\
(r = s) & = & \forall n.\ |r(n) - s(n)| \leq 4 \\
(r < s) & = & \exists n.\ r(n) + 4 < s(n) \\
(r \neq s) & = & (r < s) \vee (s < r) \\
Fun(f, I) & = & \forall x, y:\{r:\mathbb{R} \mid r \in I\}.(x = y) \Rightarrow (f(x) = f(y)) \\
SFun(f, I) & = & \forall x, y:\{r:\mathbb{R} \mid r \in I\}.(f(x) \neq f(y)) \Rightarrow (x \neq y)
\end{array}
$$

Brouwer's theorem

$$\begin{aligned} Cont(f, I) \quad = \quad & \forall \epsilon > 0.\ \exists \delta > 0.\ \forall x, y{:}\{r{:}\mathbb{R} \mid r \in I\}. \\ & |x - y| \le \delta \Rightarrow |f(x) - f(y)| \le \epsilon \end{aligned}$$

Theorem 1: $\forall a, b{:}\mathbb{R}.\ (a < b) \Rightarrow (Cont(f, [a, b]) \Leftrightarrow Fun(f, [a, b]))$
Theorem 2: $\forall a, b{:}\mathbb{R}.\ (a \le b) \Rightarrow (Cont(f, [a, b]) \Leftrightarrow SFun(f, [a, b]))$

Bar Induction Rule

$$\begin{array}{rcll}
H & \vdash & T \in \mathit{Type} & (1) \\
H,\ n{:}\mathbb{N},\ s:(\mathbb{N}_n \to T) & \vdash & B(n,s) \vee \neg B(n,s) & (2) \\
H,\ a{:}(\mathbb{N} \to T) & \vdash & \downarrow\exists n{:}\mathbb{N}.\ B(n,a) & (3) \\
H,\ n{:}\mathbb{N}, s{:}(\mathbb{N}_n \to T), x{:}B(n,s) & \vdash & f(n,s) \in X(n,s) & (4) \\
H,\ n{:}\mathbb{N}, s{:}(\mathbb{N}_n \to T), x{:}P & \vdash & f(n,s) \in X(n,s) & (5) \\
P = \forall t{:}T.\ f(n+1, s.t) & \in & X(n+1, s.t) & \\
s.t = \lambda m.\ \mathit{if}\ \ m = n & & \mathit{then}\ t\ \mathit{else}\ s(m) & (6) \\
\hline
H & \vdash & f(0,c) \in X(0,c) & (7)
\end{array}$$

The continuity principle is constructive

- For $F \in \mathcal{B} \to \mathbb{N}$, $\bullet \in \mathit{Unit}$ the constructive content of strong continuity is an $M_F \in n : \mathbb{N} \to (\mathbb{N}_n \to \mathbb{N}) \to (\mathbb{N} \bigcup \mathit{Unit})$
- $M_F = \lambda n, f.\ \nu e.(F(\overline{f})?e : \bullet)$ where $\overline{f}(x) =$ if $x < 0$ then $\perp$ else if $x < n$ then $f(x)$ else $\mathit{exception}(e; \bullet)$
- Kreisel and others (e.g. Escardo and Xu) have shown that no such M can be extensional. This means that extensionally equal $F = G \in \mathcal{B} \to \mathbb{N}$ can have $M_F(n, f) \neq M_G(n, f)$.
- We can (and must) truncate the strong continuity proposition.
- The truncated version is still strong enough for many purposes, in particular, Brouwer's theorem.

How do we know that Nuprl is consistent?

- Doug Howe defined the $t_1 \sim t_2$ (computational bi-simulation) relation and proved its crucial properties.
- Stuart Allen gave an "inductive-recursive" definition of the partial equivalence relation semantics for the types in a Nuprl universe. He then converted this into a recursive definition. Using this semantics he defined the truth of a Nuprl sequent $H \models C \; ext \; t$.
- Vincent Rahli and Abhishek Anand have formalized all of this in Coq and proved (most of) the rules of Nuprl (work still ongoing).
- Bar Induction is powerful enough to prove that the predicative part of Coq is consistent. Therefore its soundness proof needs some more powerful principle. We used $P \vee \neg P$ in the impredicative *Prop* universe. *We believe bar induction for other reasons due to Brouwer and Kleene.*