Formalization of Mathematics in Higher Order Set Theory

Chad E. Brown

November 16, 2022

Outline

Introduction

- Example: $x + y = y + x$
- Higher-Order Logic
  - Types, Terms, Proofs
  - Interactive and Automated Theorem Provers
- Set Theory
- Surreal Numbers: $x + y = y + x$ revisited

Outline

Example: Commutativity of Addition

- $\forall xy.x + y = y + x$
- where x and y range over natural numbers
  - i.e.: $\omega = \{0, 1, 2, \ldots\}$
- Assume we know:
  - $\forall x.x + 0 = x$
  - $\forall xy.x + (S\ y) = S(x + y)$
- Here S is the successor function: $S\ x$ is $x + 1$.
- Commutativity proven by induction, with two subclaims proven by induction:
  - $\forall y.0 + y = y$
  - $\forall xy.(S\ x) + \ y = S(x + y)$

Stating Induction

- Commutativity: $\forall xy.x + y = y + x$
- How to state induction as a formula?

$$\forall p.p\ 0 \ \rightarrow\ (\forall y.p\ y\ \rightarrow\ p\ (S\ y))\ \rightarrow\ \forall y.p\ y$$

- Here "p" ranges over predicates on natural numbers.
- p has a different type than x and y.
- First-order logic allows $\forall x$ and $\forall y$.
- We need to go beyond first-order logic to a logic that allows $\forall p$.

Applying Induction

$$\forall p.p\ 0 \ \rightarrow\ (\forall y.p\ y\ \rightarrow\ p\ (S\ y))\ \rightarrow\ \forall y.p\ y$$

- For a fixed x let $p\ y$ mean $x + y = y + x$.
- For this p induction specializes to

$$\begin{gathered} x + 0 = 0 + x \\ \rightarrow (\forall y.x + y = y + x \rightarrow x + (S\ y) = (S\ y) + x) \\ \rightarrow (\forall y.x + y = y + x) \end{gathered}$$

- Soon we will write the p as

$$\lambda y.x + y = y + x$$

- The other two subclaims will apply induction with different, but similar, values for p.

Outline

Higher-Order Logic


Alonzo Church


Peter B. Andrews

- Church created the simply typed λ-calculus version of higher-order logic in 1940.
- Andrews pioneered research in automated theorem proving in higher-order logic for many decades. (TPS)

Simple Types

- ι (individuals)
- o (propositions/booleans/truth values)
- $\alpha \to \beta$ (function types)

Frames: Interpreting Simple Types

Intended interpretation of simple types:

- $\mathcal{D}_\iota$: some nonempty set (e.g., the natural numbers or a universe of sets)
- $\mathcal{D}_o$: two truth values $\{0, 1\}$
- $\mathcal{D}_{\alpha\rightarrow\beta}$: some set of functions $f : \mathcal{D}_\alpha \rightarrow \mathcal{D}_\beta$
- Plus some closure conditions.

Frames: Interpreting Simple Types

Example:

- $\mathcal{D}_\iota = \{0, 1, 2, \ldots, \}$
- $\mathcal{D}_o = \{0, 1\}$
- $\mathcal{D}_{\alpha\to\beta} = (\mathcal{D}_\beta)^{\mathcal{D}_\alpha}$
- The successor function is in $\mathcal{D}_{\iota\to\iota}$.
- Curried binary $+$ function is in $\mathcal{D}_{\iota\to\iota\to\iota}$
  $(+\ 1)$ is in $\mathcal{D}_{\iota\to\iota}$ and $(+\ 1)\ 2 = 3 \in \mathcal{D}_\iota$
- The characteristic function ξ of the set of even numbers is in $\mathcal{D}_{\iota\to o}$.

$$\xi(n) = 1 \quad \text{iff} \quad n \quad \text{is even}$$

- Essentially, members of $\mathcal{D}_{\iota\to o}$ are sets of natural numbers.

Simply Typed λ-Terms

- Typed Variables x
- Typed Constants c
- Applications $s\ t$
- Abstractions $\lambda x.s$ (or $\lambda x : \alpha.s$)
- Implications $s \to t$
- Universal quantifiers $\forall x.s$ (or $\forall x : \alpha.s$)
- Obvious typing conditions:
  - $s\ t$ has type β if s has type $\alpha \to \beta$ and t has type α.
  - $\lambda x.s$ has type $\alpha \to \beta$ if x has type α and s has type β.
  - $s \to t$ and $\forall x.s$ have type o if s and t have type o.
- Propositions are terms of type o.
- Closed terms are those with no free variables.

Interpreting Terms

- φ : assignment mapping variables x (of type α) into $\mathcal{D}_\alpha$
- $\mathcal{I}$ is defined so that for every assignment φ and every term s of type α,

$$\mathcal{I}_\varphi(s) \in \mathcal{D}_\alpha.$$

Frame $\mathcal{D}$ + interpretation function $\mathcal{I}$
= "Henkin interpretation"

Interpreting Terms

- φ : assignment mapping variables x (of type α) into $\mathcal{D}_\alpha$
- $\mathcal{I}$ is defined so that for every assignment φ and every term s of type α,

$$\mathcal{I}_\varphi(s) \in \mathcal{D}_\alpha.$$

- $\mathcal{I}_\varphi(x) = \varphi(x)$
- $\mathcal{I}(c) \in \mathcal{D}_\alpha$ (user's choice for constants c of type α)

Frame $\mathcal{D}$ + interpretation function $\mathcal{I}$
= "Henkin interpretation"

Interpreting Terms

- φ : assignment mapping variables x (of type α) into $\mathcal{D}_\alpha$
- $\mathcal{I}$ is defined so that for every assignment φ and every term s of type α,

$$\mathcal{I}_\varphi(s) \in \mathcal{D}_\alpha.$$

- $\mathcal{I}_\varphi(s\ t)$ is $\mathcal{I}_\varphi(s)$ applied to $\mathcal{I}_\varphi(t)$

Frame $\mathcal{D}$ + interpretation function $\mathcal{I}$
= "Henkin interpretation"

Interpreting Terms

- φ : assignment mapping variables x (of type α) into $\mathcal{D}_\alpha$
- $\mathcal{I}$ is defined so that for every assignment φ and every term s of type α,

$$\mathcal{I}_\varphi(s) \in \mathcal{D}_\alpha.$$

- $\mathcal{I}_\varphi(\lambda x.s)$ is the function $f \in \mathcal{D}_{\alpha\to\beta}$ such that $f(a) = \mathcal{I}_{\varphi^x_a}(s)$.

Frame $\mathcal{D}$ + interpretation function $\mathcal{I}$
= "Henkin interpretation"

Interpreting Terms

- φ : assignment mapping variables x (of type α) into $\mathcal{D}_\alpha$
- $\mathcal{I}$ is defined so that for every assignment φ and every term s of type α,

$$\mathcal{I}_\varphi(s) \in \mathcal{D}_\alpha.$$

- $\mathcal{I}_\varphi(s \to t) = 1$ iff $\mathcal{I}_\varphi(s) = 0$ or $\mathcal{I}_\varphi(t) = 1$.
- $\mathcal{I}_\varphi(\forall x.s) = 1$ iff $\mathcal{I}_{\varphi_a^x}(s) = 1$ for all $a \in \mathcal{D}_\alpha$.

Frame $\mathcal{D}$ + interpretation function $\mathcal{I}$
= "Henkin interpretation"

Validity

A proposition s (term of type o) is valid if
$$\mathcal{I}_\varphi(s) = 1$$
for every Henkin interpretation $(\mathcal{D}, \mathcal{I})$ and assignment φ.

Examples of valid propositions:

- $\forall p : o.p \to p$
- $\forall q : o.(\forall p : o.p) \to q$.

Notation:

- $\bot := \forall p : o.p$
- $\top := \forall p : o.p \to p$

The valid propositions above: $\top$ and $\forall q : o.\bot \to q$.

Induction

- Assume ι corresponds to natural numbers ($\mathcal{D}_\iota = \{0, 1, 2, \ldots\}$).
- Let O be a constant of type ι, with $\mathcal{I}\ O = 0$.
- Let S be a constant of type $\iota \to \iota$, with $\mathcal{I}\ S$ being the function mapping x to $x + 1$, so $\mathcal{I}\ S\ x = x + 1$.
- We can now write induction:

$$\forall p : \iota \to o. p\ O\ \to\ (\forall y : \iota. p\ y\ \to\ p\,(S\ y))\ \to\ \forall y : \iota. p\ y$$

Equivalence of Terms

conversion:

- α: s and t are the same up to renamings of bound variables without causing collisions.
- β: $(\lambda x.s)t$ reduces to s^x_t.
- η: $\lambda x.sx$ reduces to s if x is not free in s.
- $s \approx t$ means s and t are $\alpha\beta\eta$-convertible.
- Note: Given a Henkin interpretation $(\mathcal{D}, \mathcal{I})$,

  $s \approx t$ implies $\mathcal{I}_\varphi(s) = \mathcal{I}_\varphi(t)$.

Equality

- Let s and t be terms of type α.
- There are various ways to define a proposition $s = t$ so that the proposition really means s and t are equal.
- One is attributed to Leibniz:

$$\lambda xy.\forall q : \alpha \to o.q\ x \to q\ y$$

- Here is a variant:

$$\lambda xy.\forall q : \alpha \to \alpha \to o.q\ x\ y \to q\ y\ x$$

- They are semantically equivalent.

Commutativity of Addition

- Assume ι corresponds to natural numbers ($\mathcal{D}_\iota = \{0, 1, 2, \ldots\}$).
- Let O be a constant of type ι, with $\mathcal{I}\ O = 0$.
- Let S be a constant of type $\iota \to \iota$, with $\mathcal{I}\ S\ x = x + 1$.
- Let A be a constant of type $\iota \to \iota \to \iota$, with $\mathcal{I}\ A\ x\ y = x + y$.
- "$x + 0 = x$:" $\forall x.A\ x\ O = x$
- "$x + Sy = S(x + y)$:" $\forall xy.A\ x\ (S\ y) = S\ (A\ x\ y)$
- "$x + y = y + x$:" $\forall xy.A\ x\ y = A\ y\ x$

Other Logical Operators

- We can define $\neg s$ to mean $s \to \bot$. It is easy to check that in any Henkin interpretation $\mathcal{I}_\varphi(\neg s) = 1$ iff $\mathcal{I}_\varphi(s) = 0$.
- There are also ways to define $\wedge$, $\vee$, $\leftrightarrow$ and $\exists$ (Russell-Prawitz style definitions).
- $s \vee t$ is $\forall p : o.(s \to p) \to (t \to p) \to p$
- Note the similarity of $s \vee t$ to induction:

$$\forall p : \iota \to o.p\, O \to (\forall y : \iota.p\, y \to p\,(S\, y)) \to \forall y : \iota.p\, y$$

- A special case of induction gives every natural is O or $S\ y$ (for some y).

Higher Order Proofs

Let $\mathcal{A}$ be a set of closed propositions (axioms).

Natural Deduction Rules (Γ finite set of propositions):

$$\frac{}{\Gamma \vdash s}\ s \in \Gamma \qquad \frac{}{\Gamma \vdash s}\ s \in \mathcal{A} \qquad \frac{\Gamma \vdash s}{\Gamma \vdash t}\ s \approx t$$

$$\frac{\Gamma, s \vdash_{\mathcal{A}} t}{\Gamma \vdash s \to t} \qquad \frac{\Gamma \vdash_{\mathcal{A}} s \to t \qquad \Gamma \vdash s}{\Gamma \vdash t}$$

$$\frac{\Gamma \vdash s}{\Gamma \vdash \forall x.s}\ x \text{ FRESH} \qquad \frac{\Gamma \vdash \forall x : \alpha.s}{\Gamma \vdash s^x_t}\ t : \alpha \qquad \frac{\Gamma, s \vdash_{\mathcal{A}} \neg\neg s}{\Gamma \vdash s}$$

$$\frac{\Gamma, s \vdash_{\mathcal{A}} t \qquad \Gamma, t \vdash_{\mathcal{A}} s}{\Gamma \vdash s = t}\ s, t : o$$

$$\frac{\Gamma \vdash s\ x \ = \ t\ x}{\Gamma \vdash s = t}\ s, t : \alpha \to \beta,\ x : \alpha \text{ FRESH}$$

Commutativity of Addition

- Assume $\mathcal{A}$ includes these:
  - Induction:

$$\forall p : \iota \to o . p\, O \to (\forall y : \iota . p\, y \to p\,(S\, y)) \to \forall y : \iota . p\, y$$

  - "$x + 0 = x$:" $\forall x . A\ x\ O = x$
  - "$x + Sy = S(x + y)$:" $\forall xy . A\ x\ (S\ y) = S\ (A\ x\ y)$
- Using the $\forall$-elimination rule with induction and $\lambda y . A\ O\ y = y$ gives:

$$\begin{array}{ll} \vdash & (\lambda y . A\ O\ y = y)\ O \\ & \to (\forall x : \iota . (\lambda y . A\ O\ y = y)\ x \to (\lambda y . A\ O\ y = y)\ (S\ x)) \\ & \to \forall x : \iota . (\lambda y . A\ O\ y = y)\ x . \end{array}$$

- By β conversion we obtain

$$\begin{array}{ll} \vdash & A\ O\ O = O \\ & \to (\forall x : \iota . A\ O\ x = x \to A\ O\ (S\ x) = (S\ x)) \\ & \to \forall x : \iota . A\ O\ x = x . \end{array}$$

Soundness and Completeness

- Let $\mathcal{A}$ be a set of closed propositions (axioms).
- A Henkin model of $\mathcal{A}$ is a Henkin interpretation $(\mathcal{D}, \mathcal{I})$ such that $\mathcal{I}(s) = 1$ for every $s \in \mathcal{A}$.
- A proposition s is $\mathcal{A}$-valid if $\mathcal{I}_\varphi(s) = 1$ in every Henkin model of $\mathcal{A}$.
- This proof system is "sound": if $\vdash s$, then s is $\mathcal{A}$-valid.
- This proof system is "complete": if s is $\mathcal{A}$-valid, then $\vdash s$.

Alternative Proof Systems

- Natural deduction is good for interactive theorem proving.
- Other calculi are good for automated theorem proving.
  - Sequent Calculi
  - Tableau
  - Resolution
  - Superposition

Automated Theorem Proving

- There are many ATPs based on Church's type theory:
  - Leo-III
  - Zipperposition
  - TPS
  - Satallax
  - Lash
- Two top FO provers have been extended to search in Church's type theory:
  - E
  - Vampire

Interactive Theorem Proving

- There are many ITPs based on extensions of Church's type theory:
  - HOL-light
  - HOL4
  - Isabelle-HOL
  - etc.
- The extensions include type variables and type definitions.
- These extensions make automated theorem proving much harder.
- My own ITP, Megalodon, does not include these extensions.

Satallax

About 10 years ago I worked on a higher-order theorem prover Satallax. It won the TH0 division of CASC for most years of the 2010s.

- Complete tableau calculus (in the Hintikka, Beth, Smullyan, Fitting sense) for higher-order logic with a choice operator.
- Instantiation based – used *no* unification in the basic calculus.
- Had interesting restriction on quantifiers at base types: only instantiate with *discriminating* terms.
- Able to reason with equations without rewriting deeply inside terms.
- People still think I work on this, though I haven't in years.
- The developer who took over from me about 5 years ago also is no longer working on it.

Satallax

$$\forall x.f\ x = x$$

$$p\ (f\ (f\ a))$$

$$\neg p\ a$$

Satallax

$$\forall x. f\ x = x$$

$$p\ (f\ (f\ a))$$

$$\neg p\ a$$

$$f\ (f\ a) \neq a$$

Satallax

$$\forall x. f\ x = x$$

$$p\ (f\ (f\ a))$$

$$\neg p\ a$$

$$f\ (f\ a) \neq a$$

$$f\ a = a$$

Satallax

$$\forall x. f\ x = x$$

$$p\ (f\ (f\ a))$$

$$\neg p\ a$$

$$f\ (f\ a) \neq a$$

$$f\ a = a$$

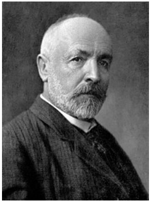
$$\begin{array}{c|c} f\ (f\ a) \neq f\ a & a \neq a \\ f\ a \neq a & \end{array}$$

Lash

- Lash is a new implementation of Satallax's calculus.
- Cezary Kaliszyk reimplemented terms/$\beta\eta$-normalization in C
- ...with perfect sharing.
- He also reimplemented important data structures like priority queues in C.
- "Better" than Satallax already (but not in CASC 2022), but it's still early days.

Outline

Set Theory



Georg Cantor

Ernst Zermelo

No one shall expel us from the paradise that Cantor has created. - David Hilbert

Set Theory

- Popular foundation for mathematics
- Natural choice for formalizers of mathematics
- The Mizar people knew this in the 1970s already.
- ZFC (and TG) are not finitely axiomatizable in first-order
- ...but higher-order versions are!

Constructors for HO Set Theory

- ι is now used as the type of sets.
- $\emptyset$ is a constant of type ι.
- $\bigcup$ is a constant of type $\iota \to \iota$ ($\bigcup X$ is the union of X).
- $\wp$ is a constant of type $\iota \to \iota$ ($\wp X$ is power set of X).
- R is a constant of type $\iota \to (\iota \to \iota) \to \iota$.
  - $R\ X\ (\lambda x.t)$ corresponds to the set $\{t | x \in X\}$.
  - Fraenkel "replacement" operator.
- plus two other constants for choice and universes.

Axioms for HO Set Theory

- Set extensionality: $X \subseteq Y \to Y \subseteq X \to X = Y$
- Foundation (an $\in$-induction principle)
- An axiom for each constant, e.g.:
  - $y \in \{t | x \in X\} \leftrightarrow \exists x \in X . y = t$

Axioms for HO Set Theory

- Set extensionality: $X \subseteq Y \to Y \subseteq X \to X = Y$
- Foundation (an $\in$-induction principle)
- An axiom for each constant, e.g.:
  - $y \in \{t|x \in X\} \leftrightarrow \exists x \in X.y = t$
  - That is:

$$\forall X : \iota.\forall F : \iota \to \iota.\forall y : \iota.y \ \in \ R\,X\,F \leftrightarrow \exists x.x \in X \wedge y = F\,x.$$

Natural Numbers in HO Set Theory

- $\emptyset$ as 0.
- ordsucc $X = X \cup \{X\}$ as successor.
- natp : $\iota \to o$ as the least predicate with 0 and closed under successor:

$$\lambda n.\forall p : \iota \to o.p\,\emptyset \;\to\; (\forall x.p\,x \;\to\; p\,(\text{ordsucc}\,x)) \;\to\; p\;n.$$

- Induction principle is now provable:

$$\begin{aligned}\forall p : \iota \to o. \quad & p\ \emptyset \\ & \to (\forall x.\text{natp}\ x \;\to\; p\ x \;\to\; p\ (\text{ordsucc}\ x)) \\ & \to \forall x.\text{natp}\ x \;\to\; p\ x.\end{aligned}$$

- Also, addition is definable, relevant identities are provable, and commutativity is provable again as before.
- Using a universe U, we can define a set ω as $\{x \in U | \text{natp}\ x\}$.

Ordinals in HO Set Theory

- Natural numbers are the finite ordinals.
- ω is the first infinite ordinal.
- Let TransSet be a constant of type $\iota \to o$.
- Defining equation:

$$\forall x : \iota.\text{TransSet } x \leftrightarrow \forall y \in x.y \subseteq x$$

- Let ordinal be a constant of type $\iota \to o$.
- Defining equation:

$$\forall x : \iota.\text{ordinal } x \leftrightarrow \text{TransSet } x \wedge \forall y \in x.\text{TransSet } y$$

Ordinals in HO Set Theory

- The following ordinal induction principle is provable:

$$\forall p : \iota \to o.$$
$$(\forall x.\text{ordinal } x \ \to \ (\forall y \in x.p\ y) \ \to \ p\ x)$$
$$\to (\forall x.\text{ordinal } x \ \to \ p\ x)$$

Outline

Surreal Numbers

- Example formalization: Conway's Surreal Numbers.
- John Conway. On Numbers and Games. 1976.
- If L is a set of surreal numbers and R is a set of surreal numbers and $x < y$ for every $x \in L$ and $y \in R$, then there is a "first" surreal z such that $L < z < R$ (pointwise).

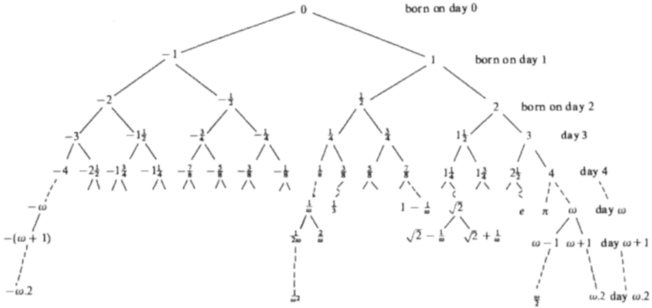


FIG. 0. When the first few numbers were born.

Surreal Numbers

- I formalized Surreals in HO set theory in Megalodon.
- 850 theorems starting from axioms of set theory up through the complex number field.
- The first 315 (37%) are before starting surreals (set theory infrastructure).
- Commutativity of addition on the surreals is the 556'th theorem.

Surreal Addition

- Let x and y be surreal numbers.
- Let L and R be such that x is the first number between L and R.
- Let L' and R' be such that y is the first number between L' and R'.
- $x + y$ is the first surreal number between

$$\{w + y | w \in L\} \cup \{x + w | w \in L'\}$$

  and

$$\{z + y | z \in R\} \cup \{x + z | z \in R'\}$$

- $\forall xy$ surreal.$x + y = y + x$
- Proof by a double induction principle on surreals.

Surreal Addition

- Let $\mathsf{SNo} : \iota \to o$ be a predicate true for surreals.
- Let $||x||$ be the ordinal at which x is born.
- Let S_α be the set of surreals born before ordinal α.
- The double induction principle:

$$
\begin{array}{l}
\qquad\qquad \forall p : \iota \to \iota \to o. \\
(\forall xy. \quad \mathsf{SNo}\ x \ \to\ \mathsf{SNo}\ y \\
\qquad\qquad \to (\forall w \in S_{||x||}. p\ w\ y) \\
\qquad\qquad \to (\forall z \in S_{||y||}. p\ x\ z) \\
\qquad\qquad \to (\forall w \in S_{||x||}. \forall w \in S_{||y||}. p\ w\ z) \\
\qquad\qquad \to p\ x\ y) \\
\ \to \forall xy. \mathsf{SNo}\ x \ \to\ \mathsf{SNo}\ y \ \to\ p\ x\ y.
\end{array}
$$

Outline

Conclusion

- Mathematics can be formalized in Church's type theory with axioms for set theory.
- This approach does not require additions to Church's type theory (e.g., type variables and type definitions).
- The approach keeps the interactive theorem proving formulation close to the automated theorem proving formulation.