Strong Types for Direct Logic

Carl Hewitt
http://plus.google.com/+CarlHewitt-StandardIoT

This article is dedicated to Alonzo Church, Richard Dedekind, Stanisław Jaśkowski, Bertrand Russell, Ludwig Wittgenstein. and Ernst Zermelo.

Abstract

This article follows on the introductory article "Direct Logic for Intelligent Applications" [Hewitt 2017a]. Strong Types enable new mathematical theorems to be proved including the Formal Consistency of Mathematics. Also, Strong Types are extremely important in Direct Logic because they block all known paradoxes[Cantini and Bruni 2017]. Blocking known paradoxes makes Direct Logic safer for use in Intelligent Applications by preventing security holes.

Inconsistency Robustness is performance of information systems with pervasively inconsistent information.[1] Inconsistency Robustness of the community of professional mathematicians is their performance repeatedly repairing contradictions over the centuries. In the Inconsistency Robustness paradigm, deriving contradictions has been a progressive development and not "game stoppers." Contradictions can be helpful instead of being something to be "swept under the rug" by denying their existence, which has been repeatedly attempted by authoritarian theoreticians (beginning with some Pythagoreans). Such denial has delayed mathematical development. This article reports how considerations of Inconsistency Robustness have recently influenced the foundations of mathematics for Computer Science continuing a tradition developing the sociological basis for foundations.[2]

Mathematics here means the common foundation of all classical mathematical theories from Euclid to the mathematics used to prove Fermat's Last [McLarty 2010]. Direct Logic provides categorical axiomatizations of the Natural Numbers, Real Numbers, Ordinal Numbers, Set Theory, and the Lambda Calculus meaning that up a unique isomorphism there is only one model that satisfies the respective axioms. Good evidence for the consistency Classical Direct Logic derives from how it blocks the known paradoxes of classical mathematics. Humans have spent millennia devising paradoxes for classical mathematics.

Having a powerful system like Direct Logic is important in computer science because computers must be able to formalize all logical inferences (including inferences about their own inference processes) without requiring recourse to human intervention. Any inconsistency in Classical Direct Logic would be a potential security hole because it could be used to cause computer systems to adopt invalid conclusions.

After [Church 1934], logicians faced the following dilemma:

- 1st order theories cannot be powerful lest they fall into inconsistency because of Church's Paradox.
- 2nd order theories contravene the philosophical doctrine that theorems must be computationally enumerable.

The above issues can be addressed by requiring Mathematics to be strongly typed using so that:

- Mathematics self proves that it is "open" in the sense that theorems are not computationally enumerable.[3]
- Mathematics self proves that it is *formally* consistent.[4]
- Strong mathematical theories for Natural Numbers, Ordinals, Set Theory, the Lambda Calculus, Actors, etc. are inferentially decidable, meaning that every true proposition is provable and every proposition is either provable or disprovable. Furthermore, theorems of these theories are not enumerable by a provably total procedure.

Mathematical Foundation for Computer Science

Computer Science brought different concerns and a new perspective to mathematical foundations including the following requirements:[5] [Arabic numeral superscripts refer to endnotes at the end of this article]

- provide powerful inference machinery so that arguments (proofs) can be short and understandable and all logical inferences can be formalized
- establish standard foundations so people can join forces and develop common techniques and technology
- incorporate axioms thought to be consistent by the overwhelming consensus of working professional mathematicians, e.g., natural numbers [Dedekind 1888], Actors, real numbers [Dedekind 1888], ordinals, sets, lambda calculus, *etc*.
- facilitate inferences about the mathematical foundations used by computer systems.

Sociology of Foundations

> "Faced with the choice between changing one's mind and proving that there is no need to do so, almost everyone gets busy on the proof." John Kenneth Galbraith [1971 pg. 50]

> "Max Planck, surveying his own career in his Scientific Autobiography [Planck 1949], sadly remarked that 'a new scientific truth does not triumph by convincing its opponents and making them see the light, but rather because its opponents eventually die, and a new generation grows up that is familiar with it.' " [Kuhn 1962]

The inherently social nature of the processes by which principles and propositions in logic are produced, disseminated, and established is illustrated by the following issues with examples:[6]

- **The formal presentation of a demonstration (proof) has not led automatically to consensus.** Formal presentation in print and at several different professional meetings of the extraordinarily simple proof in this paper have not lead automatically to consensus about the theorem that "Mathematics proves that it is formally consistent". New results can sound crazy to those steeped in conventional thinking. Paradigm shifts often happen because conventional thought is making assumptions taken as dogma. As computer science continues to advance, such assumptions can get in the way and have to be discarded.
- **There has been an absence of universally recognized central logical principles**. Disputes over the validity of the Principle of Excluded Middle led to the development of Intuitionistic Logic.
- **There are many ways of doing logic.** One view of logic is that it is about *truth*; another view is that it is about *argumentation* (i.e. proofs).[7]
- **Argumentation and propositions have be variously (re-)connected and both have been re-used.** Church's paradox [Church 1934] is that assuming theorems of mathematics are computationally enumerable leads to contradiction. In this article, Church's Paradox is transformed into the fundamental principle that "Mathematics is Open" (*i.e.* it is a theorem of mathematics that the proofs of mathematics are not computationally enumerable).[i]

[i] See discussion in this article.

- **New technological developments have cast doubts on traditional logical principles.** The pervasive inconsistency of modern large-scale information systems has cast doubt on some logical principles, *e.g*., Excluded Middle.[8] That there are proofs that cannot be expressed through text alone, overturns a long-held philosophical dogma about mathematical theories, i.e., that all theorems of a theory can be computationally generated by starting with axioms and mechanically applying rules of inference
- **Political actions have been taken against views differing from the establishment theoreticians**. According to [Kline 1990, p. 32], Hippasus was literally thrown overboard by his fellow Pythagoreans "*…for having produced an element in the universe which denied the…doctrine that all phenomena in the universe can be reduced to whole numbers and their ratios*." Fearing that he was dying and the influence that Brouwer might have after his death, Hilbert fired[9] Brouwer as an associate editor of *Mathematische Annalen* because of "*incompatibility of our views on fundamental matters*"[10] *e.g.*, Hilbert ridiculed Brouwer for challenging the validity of the Principle of Excluded Middle. [Gödel 1931] results were for Principia Mathematica as the foundation for the mathematics of its time including the categorical axiomatization of the natural numbers. In face of Wittgenstein's devastating criticism, Gödel insinuated[11] that he was crazy and retreated to relational 1st order theory in an attempt to salvage his results. Since theoreticians found it difficult to prove anything significant about practical mathematical theories, they cut them down to unrealistic relational 1st order theories where results could be proved (*e.g.* compactness) that did not hold for practical mathematical theories. In the famous words of Upton Sinclair:

  "It is difficult to get a man to understand something,
  when his salary depends on his not understanding it."

  Some theoreticians have ridiculed dissenting views and attempted to limit their distribution by political means.[12]

Foundations with strong parameterized types

> "Everyone is free to elaborate [their] own foundations. All that is required of [a] Foundation of Mathematics is that its discussion embody absolute rigor, transparency, philosophical coherence, and addresses fundamental methodological issues."[13]

> "The aims of logic should be the creation of "a unified conceptual apparatus which would supply a common basis for the whole of human knowledge." [Tarski 1940]

Note: types in Direct Logic are much stronger than constructive types with constructive logic because Classical Direct Logic has all of the power of Classical Mathematics.

Booleans are Propositions although Propositions are not reducible to Booleans:

- **True**:Boolean
- **False**:Boolean
- Boolean⊑Proposition◁1▷ //each Boolean is a Proposition
- Boolean≠Proposition◁1▷ //some Propositions are **not** Booleans
- (3=3) ≠ **True** //the proposition 3=3 is **not** equal to **True**
- (3=3) ≠ (4=4)
  //the proposition 3=3 is **not** equal to the proposition 4=4
- (3=4) ≠ **False** //the proposition 3=4 is **not** equal to **False**

In Direct Logic, unrestricted recursion is allowed in programs. For example, There are uncountably many Actors.[14] For example, Real.[ ] can output any real number[i] between 0 and 1 where

Real.[ ]:ℝ ≡ [(0 **either** 1), ∀**Postpone** Real.[ ]]

where

- (0 **either** 1) is the nondeterministic choice of 0 or 1,
- [*first*, ∀*rest*] is the list that begins with *first* and whose remainder is *rest*, and
- **Postpone** *expression* delays execution of *expression* until the value is needed.

---

[i] using binary representation.

Also, there are uncountably many propositions (because there is a different proposition for every real number). For example,

$$p[x{:}\mathbb{R}]{:}\mathbb{P}\text{roposition}\triangleleft 1\triangleright^{\mathbb{R}} \equiv \lambda[y{:}\mathbb{R}]\ (y{=}x)$$

defines a different predicate p[x] for each real number x, which holds for only one real number, namely x.[i]

Strings can be abstracted into sentences and sentences can be abstracted into propositions that can be asserted.

For example:

Propositions

e.g. $\forall[n{:}\mathbb{N}]\ \exists[m{:}\mathbb{N}]\ m>n$

i.e., ***proposition*** that for every $\mathbb{N}$ there is a larger

Sentences

e.g. $\llparenthesis\forall[n{:}\mathbb{N}]\ \llparenthesis\exists[m{:}\mathbb{N}]\ \llparenthesis m>n\rrparenthesis\rrparenthesis\rrparenthesis$

i.e., ***sentence*** for *proposition* that
for every $\mathbb{N}$ there is a larger $\mathbb{N}$

$\lfloor\llparenthesis\forall[n{:}\mathbb{N}]\ \llparenthesis\exists[m{:}\mathbb{N}]\ \llparenthesis m>n\rrparenthesis\rrparenthesis\rrparenthesis\rfloor = \forall[n{:}\mathbb{N}]\ \exists[m{:}\mathbb{N}]\ m>n$

Strings

e.g. "$\llparenthesis\forall[n{:}\mathbb{N}]\ \llparenthesis\exists[m{:}\mathbb{N}]\ \llparenthesis m>n\rrparenthesis\rrparenthesis\rrparenthesis$"

i.e., ***string*** for *sentence* for *proposition* that
for every $\mathbb{N}$ there is a larger $\mathbb{N}$

e.g. "$\forall[n{:}\mathbb{N}]\ \exists[m{:}\mathbb{N}]\ m>n$"

i.e., ***string*** for *proposition* that
for every $\mathbb{N}$ there is a larger $\mathbb{N}$

$\lfloor$"$\llparenthesis\forall[n{:}\mathbb{N}]\ \llparenthesis\exists[m{:}\mathbb{N}]\ \llparenthesis m>n\rrparenthesis\rrparenthesis\rrparenthesis$"$\rfloor = \llparenthesis\forall[n{:}\mathbb{N}]\ \llparenthesis\exists[m{:}\mathbb{N}]\ \llparenthesis m>n\rrparenthesis\rrparenthesis\rrparenthesis$

$\lfloor$"$\forall[n{:}\mathbb{N}]\ \exists[m{:}\mathbb{N}]\ m>n$"$\rfloor = \forall[n{:}\mathbb{N}]\ \exists[m{:}\mathbb{N}]\ m>n$

[i] For example (p[3])[y] holds if and only if y=3.

Classical Direct Logic is a foundation of mathematics for Computer Science, which has a foundational theory (for convenience called “Mathematics”) that can be used in any other theory. A bare turnstile is used for Mathematics so that $\vdash\Psi$ means that Ψ is a mathematical proposition that is a theorem of Mathematics and $\Phi\vdash\Psi$ means that Ψ can be inferred from Φ.

Direct Logic develops foundations for Mathematics by deriving sets from types *and* categorical axioms for the natural numbers and ordinals.

Mathematics here means the common foundation of all classical mathematical theories from Euclid to the mathematics used to prove Fermat's Last [McLarty 2010].

Proof by Contradiction in Mathematics

Proof by Contradiction is one of the most fundamental principles of Classical Mathematics (going back to before Euclid), which can be formalized

- axiomatically to say that if Ψ implies Φ and $\neg\Phi$ then $\neg\Psi$:

$$(\Psi\Rightarrow\Phi\wedge\neg\Phi)\Rightarrow\neg\Psi$$

- proof theoretically to say that proving $\neg\Psi\Rightarrow\Phi\wedge\neg\Phi$ means that Ψ is a theorem:

$$(\neg\Psi\Rightarrow\Phi\wedge\neg\Phi)\Rightarrow\vdash\Psi$$

- in [Jaśkowski 1934] natural deduction to say that (Ψ infers Φ and $\neg\Phi$) holds in a subproof [15] of a proof infers that $\neg\Psi$ holds in the proof:

$$(\Psi\vdash\Phi\wedge\neg\Phi)\vdash\neg\Psi$$

Mathematics self proves its own formal consistency (contra [Gödel 1931])

The following are fundamental to Mathematics[16]:

- Derivation by Contradiction, *i.e.* $\vdash(\neg\Phi\Rightarrow(\Theta\wedge\neg\Theta))\Rightarrow\Phi$, which says that a proposition can be proved showing that its negation implies a contradiction.
- A theorem can be used in a proof[17], *i.e.* $\vdash((\vdash\Phi)\Rightarrow\Phi)$

Theorem: Mathematics self proves its own formal consistency[18], i.e., ⊢Consistent

Formal Derivation. Suppose to obtain a contradiction, that mathematics is formally inconsistent, i.e., ¬Consistent. By definition of formal consistency, there is some proposition Ψ_0 such that ⊢ $(\Psi_0 \wedge\neg\Psi_0)$ which by the Theorem Use means $\Psi_0\wedge\neg\Psi_0$, which is a contradiction. Thus, ⊢Consistent by Derivation by Contradiction.

1) ¬Consistent	// hypothesis to derive a contradiction **just in this subargument**
2) ⊢$(\Psi_0\wedge\neg\Psi_0)$	// definition of inconsistency using **1)**
3) $\Psi_0\wedge\neg\Psi_0$	// axiom of Soundness using **2)**
⊢ Consistent	// axiom of Proof by Contradiction using 1) and 3)

Natural Deduction[i] Proof of Formal Consistency of Mathematics

Please note the following points:

- The above argument formally mathematically proves that Mathematics is formally consistent and that **it is not a premise of the theorem that Mathematics is formally consistent.**
- Mathematics was designed for consistent theories and consequently Mathematics can be used to prove its own formal consistency regardless of other axioms.[19]

The above derivation means that "Mathematics is formally consistent" is a theorem in Classical Direct Logic.

The above self-proof of formal consistency shows that the current common understanding that [Gödel 1931] proved "Mathematics cannot prove its own formal consistency, if it is formally consistent" is inaccurate.[20]

Mathematics Self Proves that it is Open.

Mathematics proves that it is open in the sense that it can prove that its theorems cannot be computationally enumerated by a provably total procedure:

Theorem $\vdash$Mathematics is Open, *i.e.*,

$$\vdash\neg\text{TheoremsEnumerableByProvedTotalProcedure}$$

Proof.[i]

Suppose to obtain a contradiction that

$$\text{TheoremsEnumerableByProvedTotalProcedure}$$

Then by the definition of TheoremsEnumerableByProvedTotalProcedure there is a deterministic total procedure TheoremsEnumerator:$[\mathbb{N}]\to\text{Proposition}$ such that the following hold where Total:$\text{Proposition}^{[\mathbb{N}]\to\mathbb{N}}$:[21]

- $\vdash$Total[TheoremsEnumerator]
- $\forall[i{:}\mathbb{N}]\ \vdash$TheoremsEnumerator.[i]

$$\forall[p{:}\text{Proposition}]\ (\vdash p) \Rightarrow \exists[i{:}\mathbb{N}]\ \text{TheoremsEnumerator}.[i]=p$$

A subset of the theorems enumerated by TheoremsEnumerator are those stating that certain deterministic procedures $[\mathbb{N}]\to\mathbb{N}$ are total. Consequently, there is a deterministic total procedure ProvedTotalsEnumerator:$([\mathbb{N}]\to([\mathbb{N}]\to\mathbb{N}))$[22], which enumerates proved total deterministic procedures:

- $\vdash$Total[ProvedTotalsEnumerator]
- $\forall[i{:}\mathbb{N}]\ \vdash$Total[ProvedTotalsEnumerator.[i]]
- $\forall[f{:}([\mathbb{N}]\to\mathbb{N})]\ (\vdash\text{Total}[f]) \Rightarrow \exists[i{:}\mathbb{N}]\ \text{ProvedTotalsEnumerator}.[i]=f$

ProvedTotalsEnumerator can be used to implement the deterministic total procedure Diagonal:$([\mathbb{N}]\to\mathbb{N})$ as follows:

$$\text{Diagonal}.[i{:}\mathbb{N}]{:}\mathbb{N} \equiv 1+(\text{ProvedTotalsEnumerator}.[i]).[i]$$

Consequently:

- $\vdash$Total[Diagonal] because it is the deterministic composition of proved total deterministic procedures.
- $\neg\vdash$Total[Diagonal] because Diagonal differs from every procedure enumerated by ProvedTotalsEnumerator.

The above contradiction completes the proof.

---

[i] This argument appeared in [Church 1934] expressing concern that the argument meant that there is "*no sound basis for supposing that there is such a thing as logic*."

[Franzén 2004] argued that Mathematics is inexhaustible because of inferential undecidability[i] of mathematical theories. The above theorem that Mathematics is open provides another independent argument for the inexhaustibility of Mathematics.

Higher Order Logic

> *"If the mathematical community at some stage in the development of mathematics has succeeded in becoming (informally) clear about a particular mathematical structure, this clarity can be made mathematically exact ... Why must there be such a characterization? Answer: if the clarity is genuine, there must be a way to articulate it precisely. If there is no such way, the seeming clarity must be illusory ... for every particular structure developed in the practice of mathematics, there is [a] categorical characterization of it."*[23]

Classical Direct Logic is much stronger than 1^{st} order axiomatizations of set theory in that it provides categoricity for natural numbers $\mathbb{N}$, reals $\mathbb{R}$, ordinals $\mathbb{O}$. set theory, the lambda calculus and Actors. Categoricity is very important in Computer Science so that there are no nonstandard elements in models of computational systems, e.g., infinite integers and infinitesimal reals. For example, nonstandard models cause problems in model checking if a model has specified properties.

Natural Number Induction

The mathematical theory[24] ***Nat*** categorically axiomatises the Natural Numbers using the following induction axiom:[25]

$$\forall[P{:}\mathrm{Proposition}{\triangleleft}1{\triangleright}^{\mathbb{N}}]\ \ (P[0] \wedge \forall[i{:}\mathbb{N}]\ P[i] \Rightarrow P[+_1[i]]) \Rightarrow \forall[i{:}\mathbb{N}]\ P[i]$$

---

[i] See section immediately below.

The other axioms of *Nat* are as follows:

- $0{:}\mathbb{N}$
- $\forall[i{:}\mathbb{N}]\ +_1[i]{:}\mathbb{N}$
- $\nexists[i{:}\mathbb{N}]\ +_1[i]=0$
- $\forall[i,j{:}\mathbb{N}]\ +_1[i]=+_1[j] \Rightarrow i=j$

Proof by Contradiction in *Nat*

Proof by Contradiction is one of the most fundamental principles of Classical Mathematics (going back to before Euclid), which can be formalized

- axiomatically to say that if **Ψ** implies **Φ** and ¬**Φ** then¬**Ψ**:

  $$(\Psi\Rightarrow\Phi\wedge\neg\Phi) \Rightarrow \neg\Psi$$

- proof theoretically to say that proving $\neg\Psi\Rightarrow\Phi\wedge\neg\Phi$ means that **Ψ** is a theorem:

  $$(\neg\Psi\Rightarrow\Phi\wedge\neg\Phi) \Rightarrow \vdash_{Nat}\Psi$$

- in [Jaśkowski 1934] natural deduction to say that (**Ψ** infers **Φ** and ¬**Φ**) holds in a subproof[26] of a proof infers that ¬**Ψ** holds in the proof:

  $$(\Psi\vdash\Phi\wedge\neg\Phi)\vdash_{Nat}\neg\Psi$$

Theorem *Nat* proves that it is formally consistent:[i] $\vdash_{Nat}$ Consistent[*Nat*]

Proof: Suppose to derive an inconsistency that ¬Consistent[*Nat*] . By the definition of formal inconsistency for *Nat*, there is some proposition Ψ_0:Proposition◁1▷ such that $\vdash_{Nat}(\Psi_0\wedge\neg\Psi_0)$ which can be used to infer in *Nat* that $\Psi_0\wedge\neg\Psi_0$. The above contradiction completes the proof.

Theorem (Indiscernibility for *Nat*):[27]

$$\forall[i,j{:}\mathbb{N}]\ i=j \Leftrightarrow \forall[P{:}\text{Proposition}\triangleleft 1\triangleright^{\mathbb{N}}]\ P[i]\Leftrightarrow P[j]$$

Theorem (Model Soundness of *Nat*): $(\vdash_{Nat}\Psi) \Rightarrow (\vDash_{\mathbb{N}}\Psi)$

Proof: Suppose $\vdash_{Nat}\Psi$. The theorem immediately follows because the axioms for the theory *Nat* hold in the type $\mathbb{N}$**.**

---

[i] Note that the results in [Gödel 1931] **do not apply** because propositions in Mathematics are strongly typed and consequently the fixed point used construct Gödel's proposition *I'mUnprovable* **does not exist** in Mathematics. See the critique of Gödel's results in this article.

Theorem (Categoricity of *Nat*):[28]

If $\mathbb{X}$ be a type satisfying the axioms for the natural numbers *Nat*, then there is a unique isomorphism I with $\mathbb{N}$ defined as follows:

- $I{:}\mathbb{X}^{\mathbb{N}}$
- $I[0_{\mathbb{N}}] \equiv 0_{\mathbb{X}}$
- $I[+_1[j]] \equiv +_1^{\mathbb{X}}[I[j]]$

because

- I is defined on $\mathbb{N}$
- I is 1-1
- I is onto $\mathbb{X}$
- I is a homomorphism
  - $I[0_{\mathbb{N}}] \equiv 0_{\mathbb{X}}$
  - $\forall[i{:}\mathbb{N}]\ I[+_1[j]] \equiv +_1^{\mathbb{X}}[I[j]]$
- I^{-1} is a homomorphism
  - $I^{-1}[0_{\mathbb{X}}] \equiv 0_{\mathbb{N}}$
  - $\forall[z{:}\mathbb{X}]\ I^{-1}[+_1^{\mathbb{X}}[z]] \equiv [+_1[I^{-1}[z]]$
- If g is an isomorphism with $\mathbb{X}$, then g=I

Richard Dedekind

Corollary There are no infinite numbers in models of the theory *Nat*, i.e.,

$\forall[\mathbb{X}{::}]\ Nat\triangleleft\mathbb{X}\triangleright \Rightarrow \nexists[i{:}\mathbb{X}]\ \forall[j{:}\mathbb{X}]\ j<i$

Definition: ClosedTerms◁*Nat*▷ is all terms of *Nat* with no free variables.

Corollary: *Nat*◁ClosedTerms◁*Nat*▷▷

Proof. ClosedTerms◁*Nat*▷ clearly satisfies the axioms of *Nat*.[29]
Categoricity provides the answer as to which closed terms are equal.

Theorem:[30] **Logical completeness of *Nat***

$\forall[\Psi{:}\text{Proposition}\triangleleft 1\triangleright]\ (\models_{\mathbb{N}} \Psi) \Rightarrow \vdash_{Nat} \Psi$

Proof.

Suppose in *Nat*, $\Psi{:}\text{Proposition}\triangleleft 1\triangleright$ and $\models_{\mathbb{N}} \Psi$. Further suppose to obtain a contradiction that $\neg\Psi$. Hence Ψ and $\neg\Psi$, which is a contradiction. Therefore $\vdash_{Nat}\Psi$ using proof by contradiction in *Nat*.[31]

Although proposition has finite length, there are uncountably many propositions. Consequently, even though every proof has finite length, there are uncountably many proofs because there are uncountably many propositions. Thus a proof may not be expressible as a character string because there are uncountable many proofs. Although by the above theorem *Nat* is inferentially complete, some proofs are not expressible as character strings. It is an open problem to characterize theorems of *Nat* whose proofs cannot be expressed as character strings.

Corollary. Equivalence of satisfiability and provability in *Nat*, i.e.,

$$\forall[\Psi:\text{Proposition}\lhd Nat\rhd]\ (\vDash_{\mathbb{N}} \Psi) \Leftrightarrow (\vdash_{Nat} \Psi)$$

Theorem. Inferential Decidability of *Nat*, i.e.,

$$\forall[\Psi:\text{Proposition}\lhd Nat\rhd]\ (\vdash_{Nat} \Psi) \vee (\vdash_{Nat} \neg\Psi)$$

Proof. Follows immediately from $(\vDash_{\mathbb{N}} \Psi) \Leftrightarrow (\vdash_{Nat} \Psi)$

Theorem (Instance Adequacy of *Nat*):[32]

$$\forall[P:\text{Proposition}\lhd 1\rhd^{\mathbb{N}}](\forall[i:\mathbb{N}]\ \vdash_{Nat} P[i])\ \Rightarrow\ \vdash_{Nat} \forall[i:\mathbb{N}]\ P[i]$$

Proof: Suppose $\forall[i:\mathbb{N}]\ \vdash_{Nat} P[i]$ which means by completeness $\forall[i:\mathbb{N}]\ \vDash_{\mathbb{N}} P[i]$.

Therefore $\forall[i:\mathbb{N}]\ \vDash_{\mathbb{N}} P[i]$ which means by completeness $\vdash_{Nat} \forall[i:\mathbb{N}]\ P[i]$

Definition $\text{Total}[f:([\mathbb{N}]\to\mathbb{N})]:\text{Proposition}\lhd Nat\rhd \equiv \forall[i:\mathbb{N}]\ \exists[j:\mathbb{N}]\ f_{\blacksquare}[i]=j$

Corollary (Instance Adequacy of *Nat*):[33]

$$\vdash_{Nat} \forall[i:\mathbb{N}]\ \text{Total}[\text{NatProvablyTotal}_{\blacksquare}[i]]$$

Proof: $\vdash_{Nat} \forall[i:\mathbb{N}]\ \vdash_{Nat} \text{Total}[\text{NatProvablyTotal}_{\blacksquare}[i]]$ The proof follows immediately from Instance Adequacy of *Nat*.

Lemma. NatProvablyComputableR is not computationally enumerable.[34]

Theorem ***Nat*** proves that its proofs cannot be expressed as character strings that are validity computationally decidable.

Proof: Suppose to obtain a contradiction that proofs can be expressed as character string that are validity computationally decidable. Since ProvablyComputableR is not computationally enumerable, proofs in ***Nat*** for ProvablyComputableR cannot be represented as character strings that are validity computationally decidable.

Theorem *Nat* proves that its theorems are not enumerable by a provably total procedure, i.e.

$\vdash_{Nat}$¬TheoremsEnumerableByProvedTotalProcedure[*Nat*]

Proof:[35]

Suppose to obtain a contradiction that

TheoremsEnumerableByProvedTotalProcedure[*Nat*]

Then there is a deterministic procedure TheoremsEnumerator:[ℕ]→Proposition◁*Nat*▷ such that the following hold:

- $\vdash_{Nat}$Total[TheoremsEnumerator]
- ∀[p:Theorem◁*Nat*▷] ∃[i:ℕ] TheoremsEnumerator▪[i]=p
- ∀[i:ℕ] $\vdash_{Nat}$TheoremsEnumerator▪[i]

A subset of the theorems enumerated by TheoremsEnumerator are those stating that certain deterministic procedures [ℕ]→ℕ are total. Consequently, there is a deterministic total procedure ProvedTotalsEnumerator:([ℕ]→([ℕ]→ℕ))[36] such that the following hold:

- $\vdash_{Nat}$Total[ProvedTotalsEnumerator]
- ∀[i:ℕ] $\vdash_{Nat}$Total[ProvedTotalsEnumerator▪[i]]
- ∀[f:([ℕ]→ℕ)] ($\vdash_{Nat}$Total[f])⇒∃[i:ℕ] ProvedTotalsEnumerator▪[i]=f because ∀[f:([ℕ]→ℕ)] (⊢Total[f])⇒∃[i:ℕ] TheoremsEnumerator▪[i]=Total[f]

ProvedTotalsEnumerator can be used to implement the deterministic total procedure Diagonal:([ℕ]→ℕ) as follows:

Diagonal▪[i:ℕ]:ℕ ≡ 1+(ProvedTotalsEnumerator▪[i])▪[i]

Consequently:

- $\vdash_{Nat}$ Total[Diagonal] because Diagonal is the deterministic composition of proved total procedures.
- ¬$\vdash_{Nat}$ Total[Diagonal] because Diagonal differs from every procedure enumerated by ProvedTotalsEnumerator.

The above contradiction completes the proof.

Corollary. There are theorems[37] in *Nat* that procedures are total whose proofs **cannot** be expressed as a character string.[38]

Proof. If all of the proofs of could be expressed using character strings, then Then there is a provably total deterministic procedure TheoremsEnumerator:[ℕ]→Proposition◁*Nat*▷ such that

- $\vdash_{Nat}$Total[TheoremsEnumerator]
  because the procedure for enumerating character string proofs is total
- ∀[p:Theorem◁*Nat*▷] ∃[i:ℕ] TheoremsEnumerator▪[i]=p
  because every character string of a proof is enumerated
- ∀[i:ℕ] $\vdash_{Nat}$TheoremsEnumerator▪[i]
  because only character strings of proofs are enumerated

Theorem: Proof verification in *Nat* is **computationally undecidable**

Proof: Proofs of totality in *Nat* of procedures are countable because [ℕ]→ℕ is countable. But proofs of totality in *Nat* are **not** computationally enumerable.

Weakest Preconditions

WeakestPrecondition[Φ:Proposition◁anOrder▷^{ℕ},
f:([ℕ]→ℕ)]:Proposition◁anOrder+1▷^{ℕ} ≡
λ[i:ℕ] Φ[f▪[i]]

Theorem Weakest Preconditions are monotonic in both arguments, i.e.,[i]

- ∀[$Φ_1$,$Φ_2$:Proposition◁anOrder▷^{ℕ}; f:([ℕ]→ℕ)]
  ($Φ_1$⇴$Φ_2$)
  ⇨ (WeakestPrecondition[$Φ_1$, f]⇴WeakestPrecondition[$Φ_2$, f])
- ∀[Φ:Proposition◁anOrder▷^{ℕ}; f_1,f_2:([ℕ]→ℕ)]
  f_1⊒f_2 ⇨ (WeakestPrecondition[Φ, f_1]⇴WeakestPrecondition[Φ, f_2])

---

[i] $Φ_1$⇴$Φ_2$ means ∀[i:ℕ] $Φ_1$[i]⇨$Φ_2$[i]
Weakest precondition because:
∀[Ψ:Proposition◁anOrder▷^{ℕ}]
(Ψ⇴λ[i:ℕ] Φ[f▪[i]]) ⇨ (Ψ⇴WeakestPrecondition[Φ, f])

Summary of *Nat*

Nat can be summarized as follows:

- ***Nat* is inferentially decidable**

  $\forall[\Psi\text{:}\mathbb{Proposition}\triangleleft Nat \triangleright]\ (\vdash_{Nat} \Psi) \vee (\vdash_{Nat} \neg\Psi)$

- **A proposition is true ⇔ provable in *Nat***

  $\forall[\Psi\text{:}\mathbb{Proposition}\triangleleft Nat \triangleright]\ (\vDash_{\mathbb{N}}\Psi) \Leftrightarrow (\vdash_{Nat} \Psi)$

- **Indiscernibility for *Nat***:

  $\forall[i,j\text{:}\mathbb{N}]\ \ i{=}j \Leftrightarrow \forall[P\text{:}\mathbb{Proposition}\triangleleft 1\triangleright^{\mathbb{N}}]\ \ P[i]\Leftrightarrow P[j]$

- **Instance Adequacy of *Nat***:

  $\forall[P\text{:}\mathbb{Proposition}\triangleleft 1\triangleright^{\mathbb{N}}](\forall[i\text{:}\mathbb{N}] \vdash_{Nat} P[i])\ \Rightarrow \vdash_{Nat} \forall[i\text{:}\mathbb{N}]\ P[i]$

- ***Nat* is categorical for ℕ**

  $\vdash_{Nat} \forall[\mathbb{X}\text{::}]\ Nat\triangleleft\mathbb{X}\triangleright \Leftrightarrow \text{Isomorphic}[\mathbb{X},\mathbb{N}]$

- ***Nat* proves its own consistency**

  $\vdash_{Nat} (\neg\exists[\Psi\text{:}\mathbb{Proposition}\triangleleft Nat\triangleright] \vdash_{Nat} \Psi\wedge\neg\Psi)$

Actors

For each Actor x, x[t] is the behavior of x at time t of type Time◁x▷, where Behavior[39], where Com is the type for a communication and an outcome for a communication received has a finite set of created Actors, a finite set of sent communications, and a behavior for the next communication received. The mathematical theory ***Act*** categorically axiomatises Actors using the following axioms where ↷ is transitive and irreflexive:

- Primitive Actors
  - ∀[i:ℕ] i:Actor // natural numbers are Actors
  - ∀[x_1,x_2:Actor] [x_1, x_2]:Actor // a tuple of Actors is an Actor
- An Actor's event ordering
  - ∀[x:Actor, c_1,c_2:Com] $c_1 \neq c_2$ ⇒ $\text{Received}_x[c_1]$↷$\text{Received}_x[c_2]$ ∨ $\text{Received}_x[c_2]$↷$\text{Received}_x[c_1]$
  - ∀[x:Actor, c_1:Com] ∄[c_2:Com] $\text{Received}_x[c_1]$↷$\text{Received}_x[c_2]$↷$\text{After}_x[c_1]$
  - ∀[x:Actor, c:Com] Initial_x↷$\text{Received}_x[c]$↷$\text{After}_x[c]$
  - ∀[x:Actor, c_1,c_2:Com] Finite[{c:Com | $\text{Received}_x[c_1]$↷$\text{Received}_x[c]$↷$\text{Received}_x[c_2]$}]
- An Actor's behavior change
  - ∀[x:Actor, c_1:Com] (∄[c_2:Com] $\text{Received}_x[c_2]$↷$\text{Received}_x[c_1]$) ⇒ x[$\text{Received}_x[c_1]$]=x[Initial_x]
  - ∀[x:Actor, c_1,c_2:Com] (∄[c_3:Com] $\text{After}_x[c_1]$↷$\text{Received}_x[c_3]$↷$\text{Received}_x[c_2]$) ⇒ x[$\text{Received}_x[c_2]$]=x[$\text{After}_x[c_1]$]
- Between Actors event ordering
  - ∀[c:Com] Sent[c]↷Received[c]
  - ∀[c_1,c_2:Com] Finite[{c:Com | ∃[x_1,x_2:Actor] Sent[c_1]↷$\text{Received}_{x_1}[c]$↷$\text{Received}_{x_2}[c_2]$}]

Theorem: Actor Induction

∀[x:Actor, P:Proposition◁1▷$^{\text{Behavior}}$]

(P[x[Initial_x]] ∧ ∀[c:Com] P[x[$\text{Received}_x[c]$]]⇨P[x[$\text{After}_x[c]$]])

⇨ ∀[c:Com] P[x[$\text{Received}_x[c]$]] ∧ P[x[$\text{After}_x[c]$]]

Provably Responds

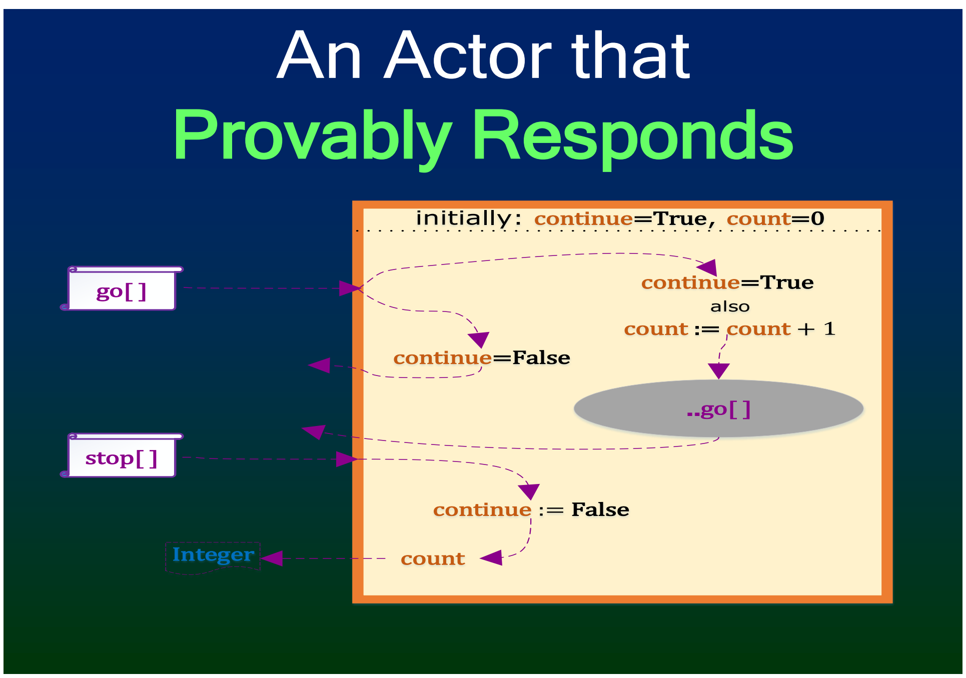
By contrast with the nondeterministic lambda calculus and pure Logic Programs, there is an always-halting Actor Unbounded that when sent a start[ ] message can compute an integer of unbounded size. This is accomplished by creating a counter with the following variables:

- count initially **0**
- continue initially **True**

and concurrently sending it both a stop[ ] message and a go[ ] message such that:

- When a go[ ] message is received:
    1. if continue is **True**, increment count by 1 and return the result of sending this counter a go[ ] message.
    2. if continue is **False**, return **Void**
- When a stop[ ] message is received, return count and set continue to **False** for the next message received.

By the Actor Model of Computation, the above Actor will eventually receive the stop[ ] message and return an unbounded number.



The following hold:

- $\forall[t{:}\mathbb{N}]\ \nvdash_{Act}$ ResponseBefore[t] // unbounded response time
- $\vdash_{Act} \exists[t{:}\mathbb{N}]$ ResponseBefore[t] // provably responds

Theorem. Unbounded Nondeterminacy of Actors
The Actor Unbounded described above cannot be implemented as a nondeterministic lambda calculus expression and cannot be implemented as a pure Logic Program.

Theorem. Computational Adequacy of Actors.
If for each i:ℕ, F_i is a nondeterministic λ expression such that
$\forall[i{:}\mathbb{N}]\ F_i \sqsubseteq F_{i+1}$, then $(\text{limit}_{i:\mathbb{N}}\ F_i)$:Actor

Theorem. Categoricity of *Act*
If X be a type satisfying the axioms for *Act*, then there is a unique isomorphism with Actor.

Theorem: Logical completeness of *Act*

$$\forall[\Psi{:}\text{Proposition}\triangleleft Act\triangleright]\ (\vDash_{\text{Actor}} \Psi) \Rightarrow (\vdash_{Act} \Psi)$$

Corollary. Equivalence of satisfiability and provability in *Act*, i.e.,

$$\forall[\Psi{:}\text{Proposition}\triangleleft Act\triangleright]\ (\vDash_{\text{Actor}} \Psi) \Leftrightarrow (\vdash_{Act} \Psi)$$

Theorem. Inferential Decidability of *Act*, i.e.,

$$\forall[\Psi{:}\text{Proposition}\triangleleft Act\triangleright]\ (\vdash_{Act} \Psi) \vee (\vdash_{Act} \neg\Psi)$$

Proof. Follows immediately from $(\vDash_{\text{Actor}} \Psi) \Leftrightarrow (\vdash_{Act} \Psi)$

Conclusion

Strong Types enable new mathematical theorems to be proved including the Formal Consistency of Mathematics. Also, Strong Types enable proofs of the Categoricity of axiomatizations of the ordinals and the cumulative hierarchy of sets of a type.

Furthermore, Strong Types are extremely important in Direct Logic because they block all know paradoxes[Cantini and Bruni 2017]. Blocking known paradoxes makes Direct Logic safer for use in Intelligent Applications by preventing security holes. For example, Strong Types block the following paradoxes: Berry [Russell 1906], Burali-Forti [Burali-Forti 1897], Church [Church 1934], Curry [Curry 1941], Girard [Coquand 1986], and Liar [Eubulides of Miletus], and Löb [Löb 1955].

Information Invariance is a fundamental technical goal of logic consisting of the following:

1. *Soundness of inference*: information is not increased by inference
2. *Completeness of inference*: all information that necessarily holds can be inferred.

Computer Science needs a rigorous foundation for all of mathematics that enables computers to carry out all reasoning without human intervention.[40] [Russell 1925] attempted basing foundations entirely on types, but foundered on the issue of being expressive enough to carry to some common mathematical reasoning. [Church 1932, 1933] attempted basing foundations entirely on untyped higher-order functions, but foundered because it was shown to be inconsistent [Kleene and Rosser 1935]. Presently, Isabelle [Paulson 1989] and Coq [Coquand and Huet 1986] are founded on types and do not allow theories to reason about themselves. Classical Direct Logic is a foundation for all of mathematical reasoning based on strong types (to provide grounding for concepts) that allows general inference about reasoning.

[Gödel 1931] claimed inferential undecidability[i] results for mathematics using the proposition *I'mUnprovable* In opposition to Wittgenstein's correct argument his proposition leads to contradictions in mathematics, Gödel claimed that the results of [Gödel 1931] were for a cut-down relational 1st order theory of natural numbers. However, relational 1st order theories are not a suitable foundation for Computer Science because of the requirement that computer systems be able to carry out all reasoning without requiring human intervention (including reasoning about their own inference systems).

Following [Russell 1925, and Church 1932-1933], Direct Logic was developed and then investigated propositions with results below.

Formalization of Wittgenstein's proof that Gödel's proposition *I'mUnprovable* leads to contradiction in mathematics. So the consistency of mathematics had to be rescued against Gödel's proposition constructed using what [Carnap 1934] later

[i] sometimes called logical "incompleteness"

called the "Diagonal Lemma" which is equivalent to the Y untyped fixed point operator on propositions. Use of the Y untyped fixed point operator on propositions in results of [Curry 1941] and [Löb 1955] also lead to inconsistency in mathematics. Consequently, mathematics had to be rescued against these uses of the Y untyped fixed point operator for propositions.

Self-proof of the formal consistency of mathematics. Consequently, mathematics had to be rescued against the claim [Gödel 1931] that mathematics cannot prove its own formal consistency. Also, it became an open problem whether mathematics proves its own formal consistency, which was resolved by the author discovering an amazing simple proof.[41] A solution is to require strongly typed mathematics to bar use of the Y untyped fixed point operator for propositions.[42] However, some theoreticians have very reluctant to accept the solution.
According to [Dawson 2006]:[43]

- *Gödel's results altered the mathematical landscape, but they did **not** "produce a debacle".*
- *There is **less** controversy today over mathematical foundations than there was **before** Gödel's work.*

However, [Gödel 1931] has produced a controversy of a very different kind from the one discussed by Dawson:

- The common understanding that mathematics cannot prove its own formal consistency[44] has been disproved.
- Consequently, [Gödel 1931] has now led to increased controversy over mathematical foundations.

Requirement to use higher order logic because moderately strong theories of 1st order logic are inconsistent. Categorical higher order theories of Natural Numbers, Reals, and Actors are inferentially complete and inferentially decidable. In general, theorems of theories in higher order logic are not computationally enumerable, proof correctness is computationally undecidable, and some proofs are inexpressible as character strings. Consequently, it will be forever necessary to invent new proof notations that were previously not expressed in a process called "Progressive Knowing".

The development of Direct Logic has strengthened the position of working mathematicians as follows:[i]

- Allowing freedom from the philosophical dogma of the 1st Order Thesis
- Providing usable strong types for all of Mathematics that provides theories that have categorical models
- Allowing theories to freely reason about theories
- Providing Inconsistency Robust Direct Logic for safely reasoning about theories of practice that are (of necessity) pervasively inconsistent.

Acknowledgements

Tom Costello, Eric Martin, Per Stenstrom, and Johan van Benthem made very helpful comments and suggestions. Interactions with John Woods were very helpful in developing a resolution to Church's Paradox. Conversations with Gordon Plotkin were helpful and his suggestions led to the introduction of the fixed point operator in the categorical theory of the lambda calculus. Michael Beeson suggested a clarification in the argument of Church's Paradox. Hendrik Boom provided an excellent critique, which resulted in a fundamental reorganization. Wilfried Sieg provided information about provability in 1st order logic.

Bibliography

Anthony Anderson and Michael Zelëny (editors). *Logic, Meaning and Computation: Essays in Memory of Alonzo Church* Springer. 2002.

Andrea Asperti, Herman Geuvers, Raja Natarajan. *Social processes, program verification and all that* "Mathematical Structures in Computer Science" Cambridge University Press. 2009.

Jeremy Avigad and John Harrison. *Formally Verified Mathematics*. CACM. April 2014.

Steve Awodey and Erich Reck. *Completeness and Categoricity. Parts I and II: Nineteenth-century Axiomatics to Twentieth-century Metalogic*. History and Philosophy of Logic. Vol. 23. 2002.

Steve Awodey, Álvaro Pelayo, and Michael A. Warren. *Voevodsky's Univalence Axiom in Homotopy Type Theory* Notices of AMS. October 2013.

Jon Barwise. *Model-Theoretic Logics: Background and Aims* in "Model Theoretic Logics" Springer-Verlag. 1985.

Francesco Berto. *The Gödel Paradox and Wittgenstein's Reasons* Philosophia Mathematica. February, 2009.

[i] Of course, Direct Logic must preserve as much previous learning as possible.

Andrey Bovykin. *Brief introduction to unprovability.* Logic Colloquium 2006. Lecture Notes in Logic 2009.

Georg Cantor. *Ueber eine elementare Frage der Mannigfaltigkeitslehre* Jahresbericht der Deutschen Mathematiker-Vereinigung. 1891

Carlo Cellucci "Gödel's Incompleteness Theorem and the Philosophy of Open Systems" *Kurt Gödel: Actes du Colloque, Neuchâtel 13-14 juin 1991*, Travaux de logique N. 7, Centre de Recherches Sémiologiques, University de Neuchâtel. http://w3.uniroma1.it/cellucci/documents/Goedel.pdf

Gregory Chaitin Interview in *Dangerous Knowledge* BBC4 documentary. 2007.

Alan Chalmers. "What is this thing called science?" Open University Press. 1999.

Haskell Curry. "Some Aspects of the Problem of Mathematical Rigor" *Bulletin of the American Mathematical Society* Vol. 4. 1941.

Alonzo Church. *The Richard Paradox.* Proceedings of American Mathematical Society. Vol. 41. No. 6. 1934.

Alonzo Church. *An unsolvable problem of elementary number theory* Bulletin of the American Mathematical Society 19, May, 1935. American Journal of Mathematics, 58. 1936,

Alonzo Church: *A Formulation of the Simple Theory of Types*, Journal of Symbolic Logic. vol. 5. 1940.

Will Clinger. *Foundations of Actor Semantics* MIT Mathematics Doctoral Dissertation. June 1981.

Thierry Coquand and Gérard Huet: *The calculus of constructions.* Technical Report 530, INRIA, Centre de Rocquencourt, 1986.

John Corcoran. *Gaps between logical theory and mathematical practice* in The methodological unity of science. 1973.

John Corcoran. *Categoricity.* History and Philosophy of Logic. Vol. 1. 1980

John Corcoran. *Second-order Logic.* Logic, Meaning and Computation. Kluwer. 2001.

John Dawson. *Shaken Foundations or Groundbreaking Realignment? A Centennial Assessment of Kurt Gödel's Impact on Logic, Mathematics, and Computer Science* FLOC'06.

Richard Dedekind (1888) "What are and what should the numbers be?" (Translation in *From Kant to Hilbert: A Source Book in the Foundations of Mathematics*. Oxford University Press. 1996) Braunschweig.

Freeman Dyson. *Heretical Thoughts about Science and Society* Boston University. November 1, 2005.

Heinz-Dieter Ebbinghaus. *Ernst Zermelo: An Approach to His Life and Work* Springer. 2007.

Patrik Eklund, M. Angeles Galan, Robert Helgesson, and Jari Kortelainenc. *Fuzzy Terms* Fuzzy Sets and Systems. 256. 2014.

Erwin Engeler. *Algebras and Combinators* Algebra Universalis. 1981.

Solomon Feferman. “Axioms for determinateness and truth” *Review of Symbolic Logic*. 2008.

Mike Genesereth and Eric Kao. *The Herbrand Manifesto Thinking Inside the Box*. Rule ML. August 2-5, 2015.

Kurt Gödel (1931) “On formally undecidable propositions of *Principia Mathematica*” in *A Source Book in Mathematical Logic, 1879-1931*. Translated by Jean van Heijenoort. Harvard Univ. Press. 1967.

Carl Hewitt. *Planner: A Language for Proving Theorems in Robots* IJCAI. 1969.

Carl Hewitt. “Procedural Embedding of Knowledge In Planner” *IJCAI* 1971.

Carl Hewitt and John Woods assisted by Jane Spurr, editors. *Inconsistency Robustness*. College Publications. 2015.

Carl Hewitt. 2015a. *Actor Model of Computation for Scalable Robust Information Systems* in “Inconsistency Robustness” College Publications. 2015.

Carl Hewitt. 2015b. *ActorScript™ extension of C#®, Java®, Objective C®, C++, JavaScript®, and SystemVerilog using iAdaptive™ concurrency* in Inconsistency Robustness. College Publications. 2015.

Carl Hewitt. 2015c. *Inconsistency Robustness in Logic Programs* in “Inconsistency Robustness” College Publications. 2015.

Carl Hewitt. 2015d. *Formalizing common sense reasoning for scalable inconsistency-robust information coordination using Direct Logic™ Reasoning and the Actor Model* in “Inconsistency Robustness” College Publications. 2015.

Carl Hewitt. 2015e. *Inconsistency Robustness in Logic Programs* in “Inconsistency Robustness” College Publications. 2015.

Carl Hewitt. 2015f. *Actors for CyberThings.* Erlang Keynote. YouTube. March 23, 2015.

Carl Hewitt. 2016b. *Future Cyberdefenses Will Defeat Cyberattacks* CACM. August 2016.

Carl Hewitt 2016c. *Islets™ Protect Sensitive IoT Information: Verifiably ending use of sensitive IoT information for mass surveillance fosters (international) commerce* SSRN WP 2836282. 2016.

Carl Hewitt 2017a. *Direct Logic for Intelligent Applications* Logic and Collaboration for Intelligent Applications. Stanford. March 30-31, 2017.

Carl Hewitt 2017b. *Axiomatics for Inconsistency Robust Direct Logic* Logic and Collaboration for Intelligent Applications. Stanford. March 30-31, 2017.

J. Roger Hindley and Jonathan Seldin. *λ-calculus and Combinators: An Introduction* Cambridge University Press. 2008

Stanisław Jaśkowski. “On the Rules of Suppositions in Formal Logic” *Studia Logica* 1, 1934. (reprinted in: *Polish logic 1920-1939*, Oxford University Press, 1967.

Morris Kline. *Mathematical thought from ancient to modern times* Oxford University Press. 1972.

Thomas Kuhn. The Structure of Scientific Revolutions University of Chicago Press. 1962.

Imre Lakatos. *Proofs and Refutations* Cambridge University Press. 1976

John Law. *After Method: mess in social science research* Routledge. 2004.

Martin Löb. "Solution of a problem of Leon Henkin." *Journal of Symbolic Logic*. Vol. 20. 1955.

David Malone. *Dangerous Knowledge* BBC4 documentary. 2007. http://www.dailymotion.com/playlist/x1cbyd_xSilverPhinx_bbc-dangerous-knowledge/1

Colin McLarty. *What Does it Take to Prove Fermat's Last Theorem? Grothendieck and the Logic of Number Theory* Journal of Symbolic Logic. September 2010.

John-Jules Meyer. *Review of Inconsistency Robustness* Amazon. January, 2016.

Ray Monk. *Bourgeois, Boshevist or anarchist? The Reception of Wittgenstein's Philosophy of Mathematics* in Wittgenstein and his interpreters. Blackwell. 2007.

Nick Nielsen Alternative Foundations/philosophical February 28, 2014. http://www.cs.nyu.edu/pipermail/fom/2014-February/017861.htmlBashar

Francis Pelletier *A Brief History of Natural Deduction* "History and Philosophy of Logic" Vol. 20, Issue. 1, 1999.

William Quine. *Philosophy of Logic* Prentice Hall. 1970.

Bertrand Russell. *Principles of Mathematics* Norton. 1903.

Bertrand Russell. *Les paradoxes de la logique* Revue de métaphysique et de morale. 1906.

Bertrand Russell and Alfred Whitehead, *Principia Mathematica* (3 volumes). Cambridge University Press. 1910-1913.

Natarajan Shankar. *Meta-mathematics, Machines, and Gödel's Proof*. Cambridge University Press. 1994.

Dana Scott. *Setoids/Modest Sets/PERs: Adding and Using Types with a Type-free λ-Calculus* Domains XII. August 2015.

Stephen Simpson *Nonprovability of certain combinatorial properties of finite trees* in Studies in Logic and the Foundations of Mathematics. North-Holland. 1985.

Alfred Tarski *Introduction to Logic* Oxford University Press. 1940 (and many subsequent editions).

R. Gregory Taylor. *Zermelo's Cantorian Theory of Systems of Infinitely Long Propositions Bulletin of Symbolic Logic* December, 2002.

Rineke Verbrugge *Provability Logic* The Stanford Encyclopedia of Philosophy. 2010.

Ludwig Wittgenstein. 1956. Bemerkungen ¨uber die Grundlagen der Mathematik/*Remarks on the Foundations of Mathematics*, *Revised Edition* Basil Blackwell. 1978

Hao Wang *A Logical Journey, From Gödel to Philosophy* MIT Press. 1974.

Andrew Wiles. *Modular elliptic curves and Fermat's Last Theorem* Annals of Mathematics. 141. 1995.

John Woods. *How robust can inconsistency get?* IfCoLoG Journal of Logics and Their Applications. 2014.

John Woods. *Inconsistency: Its present impacts and future prospects* Inconsistency Robustness 2015.

Noson Yanofsky. *The Outer Limits of Reason* MIT Press 2013.

Ernst Zermelo *Uber Grenzzahlen und Mengenbereiche: Neue Untersuchungen Äuber die Grundlagen der Mengenlehre* Fundamenta mathematicae. 1930; English translation by Michael Hallett, "On boundary numbers and domains of sets: new investigations in the foundations of set theory" *From Kant to Hilbert: a Source Book in the Foundations of Mathematics*, Oxford University Press, 1996.

Ernst Zermelo. *Uber matematische System und die Logic des Unendichen* Forschungen and Fortschritte. Vol. 8. 1932.

Ernst Zermelo. *Collected Works/Gesammelte Werke: Volume I/Band I - Set Theory, Miscellanea/Mengenlehre, Varia (Schriften der Mathematisch-naturwissenschaftlichen Klasse) (English and German Edition)* Springer. 2010.

Yoni Zohar. *Reasoning Inside The Box: Gentzen Calculi for Herbrand Logics* Stanford Logic Group. June 7, 2017.

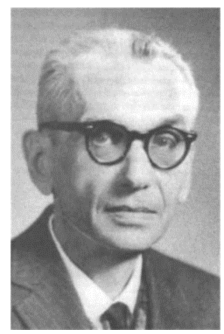
Appendix 1. Historical Background

> "The powerful (try to) insist that their statements are literal depictions of a single reality. 'It really is that way', they tell us. 'There is no alternative.' But those on the receiving end of such homilies learn to read them allegorically, these are techniques used by subordinates to read through the words of the powerful to the concealed realities that have produced them." [Law 2004]

Gödel was certain

> "'Certainty' is far from being a sign of success; it is only a symptom of lack of imagination and conceptual poverty. It produces smug satisfaction and prevents the growth of knowledge." [Lakatos 1976]

Paul Cohen [2006] wrote as follows of his interaction with Gödel:[45]



Kurt Gödel

> "His [Gödel's] main interest seemed to lie in discussing the 'truth' or 'falsity' of these [mathematical] questions, not merely in their undecidability. He struck me as having an almost unshakable belief in this "realist" position, which I found difficult to share. His ideas were grounded in a deep philosophical belief as to what the human mind could achieve. I greatly admired this faith in the power and beauty of Western Culture, as he put it, and would have liked to understand more deeply what were the sources of his strongly held beliefs. Through our discussions, I came closer to his point of view, although I never shared completely his 'realist' point of view, that all questions of Set Theory were in the final analysis, either true or false."

According to John von Neumann, Gödel was "the greatest logician since Aristotle."[46] However, [von Neumann 1961] expressed a very different mathematical philosophy than Gödel:

John von Neumann

> "It is **not** necessarily true that the mathematical method is something absolute, which was revealed from on high, or which somehow, after we got hold of it, was evidently right and has stayed evidently right ever since."

[Gödel 1931] based incompleteness results on the thesis that mathematics necessarily has the proposition *I'mUnprovable* in Principia Mathematica [Russell 1902].

Wittgenstein's Paradox

Wittgenstein correctly noted that Gödel's *I'mUnprovable* infers inconsistency in mathematics:[47]

> "Let us suppose [Gödel's writings are correct and therefore] I prove[48] the improvability (in Russell's system) of [Gödel's *I'mUnprovable*] P; [*i.e.*, $\vdash\nvdash P$ where $P\Leftrightarrow\nvdash P$] then by this proof I have proved P [*i.e.*, $\vdash P$]. Now if this proof were one in Russell's system [*i.e.*, $\vdash\vdash P$] — I should in this case have proved at once that it belonged [*i.e.*, $\vdash P$] and did not belong [*i.e.*, $\vdash\neg P$ because $\neg P\Leftrightarrow\vdash P$] to Russell's system.
>
> But there is a contradiction here! [*i.e.*, $\vdash P$ and $\vdash\neg P$]
>
> **[This] is what comes of making up such sentences."** [*emphasis added*]

Ludwig Wittgenstein

According to [Monk 2007]:

> "Wittgenstein hoped that his work on mathematics would have a cultural impact, that it would threaten the attitudes that prevail in logic, mathematics and the philosophies of them. On this measure it has been a spectacular failure."

Unfortunately, recognition of the worth of Wittgenstein's work on mathematics came long after his death. For decades, many theoreticians mistakenly believed that they had been completely victorious over Wittgenstein.

Gödel's maintained:

> "Wittgenstein did not understand it [Gödel's 1931 article on Principia Mathematica] (or pretended not to understand it). He interpreted it as a kind of logical paradox, while in fact it is just the opposite, namely a mathematical theorem within an absolutely uncontroversial part of mathematics (finitary number theory or combinatorics)."[49]

In the above, Gödel retreated from the [Gödel 1931] results on Principia Mathematic to claiming that the results were for the relational 1st order theory

Relational1stOrderNatualNumbers in order to defend his *I'mUnprovableInRelational1stOrderNatualNumbers*. However, the [Gödel 1931] incompleteness result is not very impressive because *Relational1stOrderNatualNumbers* is a very weak theory which cannot even prove that the Ackermann procedure is total.

Trying to retain *I'mUnprovable* forced Gödel into a very narrow and constricted place of reducing propositions to strings for sentences and then to Gödel numbers axiomatized in a 1^{st} order theory to avoid Wittgenstein's devastating criticism. This narrow constricted place is intolerable for computer science, which needs to reason about propositions in a more natural and flexible way using Strong Types.

Let *T* be a theory capable of representing all computable functions on Strings and Natural Numbers with GödelNumber[aWellFormedString] being the Gödel number of aWellFormedString, *where a well-formed string is here considered to be a proposition*. A Diagonal Lemma is:

If F is a well-formed string in the language with one free variable, then
there is a well-formed string S such that the following is provable in *T*:
S ⇔ F[GödelNumber[S]]

Letting GödelNumberToWellFormedString[n] be the well-formed string with Gödel number n, define Eubulides as follows (where "¬GödelNumberToWellFormedString[n]" is the string formed by prefixing the character ¬ to the well-formed string with Gödel number n):

Eubulides[n] ≡ "¬GödelNumberToWellFormedString[n]"

By the above Diagonal Lemma, there is a well-formed string *I'mFalse* such that the following is provable in *T* (where "¬GödelNumberToWellFormedString[GödelNumber[*I'mFalse*]]" is the result of prefixing the well-formed string GödelNumberToWellFormedString[GödelNumber[*I'mFalse*]] with ¬):[50]

I'mFalse ⇔ Eubulides[GödelNumber[*I'mFalse*]]
⇔ "¬GödelNumberToWellFormedString[GödelNumber[*I'mFalse*]]"
⇔ ¬*I'mFalse*

[Chaitin 2007] complained about basing something as important as incompleteness something so trivial as *I'mUnprovable*:

> "[Gödel's proof] was too superficial. It didn't get at the real heart of what was going on. It was more tantalizing than anything else. It was not a good reason for something so devastating and fundamental. It was too clever by half. It was too superficial. [It was based on the clever construction] *I'mUnprovable* So what? This doesn't give any insight how serious the problem is."

[Gödel 1931] results can be formalized as follows:

NotProvable◁n▷[Ψ:Proposition◁n▷]:Proposition◁n+1▷] ≡ ¬⊢Ψ

The construction of *I'mUnprovable* is blocked because the procedure NotProvable does ***not*** have a fixed point (by Gödel's Diagonal Lemma) *I'mUnprovable* such that *I'mUnprovable*⇔¬⊢*I'mUnprovable* because the procedure NotProvable maps a proposition Ψ of degree n into a proposition ¬⊢Ψ of degree n+1.

However, Gödel, Church, Turing, and many other logicians continued up to the present time to believe in the importance of Gödel's proof based on the proposition *I'mUnprovable*.[51]

Although Gödel's incompleteness results for *I'mUnprovable* have fundamental problems, the work was extremely significant in further the development of the history of metamathematics. For example, the following paradoxes were developed following along Gödel's work:

Curry's Paradox **[**Curry 1941**]** Suppose Ψ:Proposition◁anOrder▷.

Curry◁n▷[p:Proposition◁n▷]:Proposition◁Max[n+1,anOrder+1]▷ ≡ p⇒Ψ

Curry's Paradox is blocked because the procedure Curry does ***not*** have a fixed point.

Löb's Paradox **[**Löb 1955**]** Suppose Ψ:Proposition◁anOrder▷.

Löb◁n▷[p:Proposition◁n▷]:Proposition◁Max[n+1,anOrder+1]▷ ≡ (⊦p)⇒Ψ

Löb's Paradox is blocked because the procedure Löb does ***not*** have a fixed point.

A key difference is that Direct Logic works directly with propositions as opposed to the work of Gödel, Curry, and Löb, which was based on relational 1^{st} order theories with propositions from sentence strings coded as integers.

Nat$_1$

Nat$_1$ is a 1st order axiomatization of the Natural Numbers with the following computationally enumerable axioms:

- $\vdash_{Nat_1}$ 0:ℕ
- $\vdash_{Nat_1}$ ∀[i:ℕ] $+_1$[i]:ℕ
- $\vdash_{Nat_1}$ ∄[i:ℕ] $+_1$[i]=0
- $\vdash_{Nat_1}$ ∀[i,j:ℕ] $+_1$[i]=$+_1$[j] ⇨ i=j
- ∀[P:String◁Proposition◁1▷^{ℕ}▷] $\vdash_{Nat_1}$ Induction[P]
  where
    Induction[P:String◁Proposition◁1▷^{ℕ}▷] ≡
      (⌊P⌋[0] ∧ ∀[i:ℕ] ⌊P⌋[i]⇨ ⌊P⌋[i+1]) ⇨ ∀[i:ℕ] ⌊P⌋[i]

Nat proves *Nat*$_1$, i.e., $\vdash_{Nat}$ *Nat*$_1$

Theorem $\nvdash_{Nat_1}$ ∀[i:ℕ]Total[Nat$_1$ProvablyComputableℝ$_{[0,1]}$Enumerator.[i]]

Proof: Suppose to obtain a contradiction that

$\vdash_{Nat_1}$ ∀[i:ℕ]Total[Nat$_1$ProvablyComputableℝ$_{[0,1]}$Enumerator.[i]]

Diagonal.[i:ℕ] ≡ 1-(Nat$_1$ProvablyComputableℝ$_{[0,1]}$Enumerator.[i]).[i]

∴ Diagonal:Nat$_1$ProvablyComputableℝ$_{[0,1]}$ which is a contradiction

Theorem $\vDash_{ℕ}$ ∀[i:ℕ]Total[Nat$_1$ProvablyComputableℝ$_{[0,1]}$Enumerator.[i]]

Proof: By construction of Nat$_1$ProvablyComputableℝ$_{[0,1]}$Enumerator

As a consequence of the above two theorems,

∀[i:ℕ]Total[Nat$_1$ProvablyComputableℝ$_{[0,1]}$Enumerator. [i]]

is true but unprovable in *Nat*$_1$.

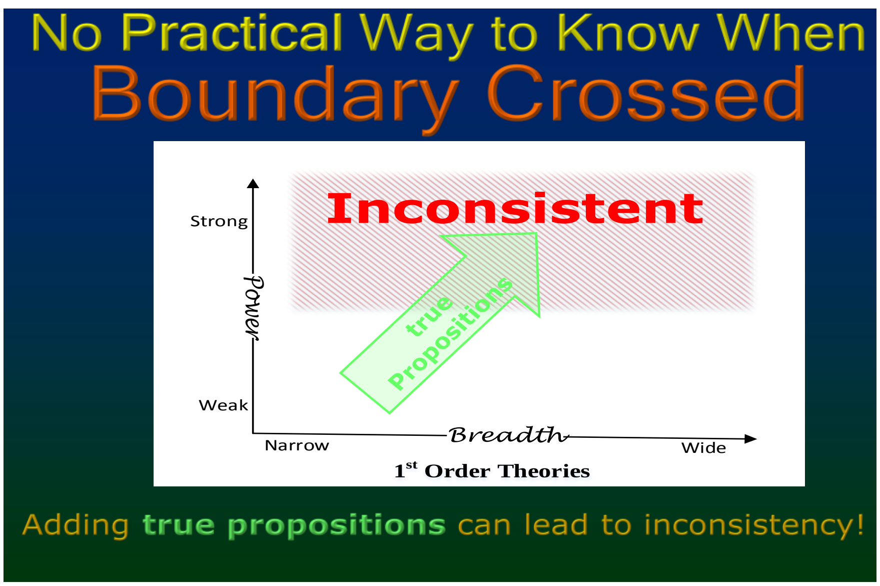
Also, can fail to prove responsiveness of Actor systems as illustrated by the following theorem for the Actor Unbounded discussed elsewhere in this article.

Theorem $\nvdash_{Nat_1} \exists[t{:}\mathbb{N}]\ \text{ResponseBefore}[t]$

Proof: Suppose to obtain a contradiction $\vdash_{Nat_1} \exists[t{:}\mathbb{N}]\ \text{ResponseBefore}[t]$,
i.e., $\vdash_{Nat_1} \neg\forall[t{:}\mathbb{N}]\ \neg\text{ResponseBefore}[t]$ ∴
$\neg\text{Consistent}[\{\neg\text{ResponseBefore}[t] | t{:}\mathbb{N}\}]$ which by compactness for Nat_1
$\exists[S\subseteq\{\neg\text{ResponseBefore}[t] \mid t{:}\mathbb{N}\}{:}\mathbb{Finite}\triangleleft\mathbb{Boolean}^{\mathbb{N}}\triangleright]\ \neg\text{Consistent}[S]$
∴ $\exists[t{:}\mathbb{N}]\ \neg\text{Consistent}[\{\neg\text{ResponseBefore}[j] \mid j<t\}]$ meaning
$\neg\forall[t{:}\mathbb{N}]\ \text{Consistent}[\{\neg\text{ResponseBefore}[j] \mid j<t\}]$ which is
a contradiction

Theorem $\vDash \exists[t{:}\mathbb{N}]\ \text{ResponseBefore}[t]$

As a consequence of the above two theorems, $\exists[t{:}\mathbb{N}]\ \text{ResponseBefore}[t]$ is true but unprovable in Nat_1.



After [Church 1934], logicians faced the following dilemma:

- 1st order theories cannot be powerful enough to be a foundation for Computer Science lest they fall into inconsistency because of Church's Paradox.
- 2nd order theories contravene the philosophical doctrine that theorems must be computationally enumerable.

Since *Nat* is more powerful than Nat_1, it must be able to formalize the argument in Church's Paradox. The following section shows why the argument in Church's Paradox fails against *Nat*.

Church's Paradox fails for Higher Order Logic

Mathematics can formalize axioms Instance_1 which are strong enough to prove Church's Paradox using Instance_1 provably computable reals, which can be defined as follows:

$$\text{Instance}_1\text{ProvablyComputable}\mathbb{R}_{[0,1]} \equiv \mathbb{R}_{[0,1]} \ni\lambda[r]\ \text{Instance}_1\vdash \text{Computable}[r]$$

where Instance_1 has axioms given just below:

- $(0{:}\mathbb{N}){:}\text{Instance}_1$
- $(\forall[i{:}\mathbb{N}]\ +_1[i]{:}\mathbb{N}){:}\text{Instance}_1$
- $(\nexists[i{:}\mathbb{N}]\ +_1[i]=0){:}\text{Instance}_1$
- $(\forall[i,j{:}\mathbb{N}]\ +_1[i]=+_1[j] \Rightarrow i=j){:}\text{Instance}_1$
- $\forall[P{:}\text{String}\triangleleft\text{Proposition}\triangleleft 1\triangleright^{\mathbb{N}}\triangleright]\ \text{Induction}[P]{:}\text{Instance}_1$

  where

  $\text{Induction}[P{:}\text{String}\triangleleft\text{Proposition}\triangleleft 1\triangleright^{\mathbb{N}}\triangleright] \equiv$

  $(\lfloor P\rfloor[0] \wedge \forall[i{:}\mathbb{N}]\ \lfloor P\rfloor[i]\Rightarrow \lfloor P\rfloor[i+1]) \Rightarrow \forall[i{:}\mathbb{N}]\ \lfloor P\rfloor[i]$
- $\forall[P{:}\text{String}\triangleleft\text{Proposition}\triangleleft 1\triangleright^{\mathbb{N}}\triangleright]\ \text{Extension}[P]{:}\text{Instance}_1$

  $\text{Extension}[P{:}\text{String}\triangleleft\text{Proposition}\triangleleft 1\triangleright^{\mathbb{N}} \equiv$

  $\forall[i{:}\mathbb{N}]\ i\in\text{Extension}[\lfloor P\rfloor] \Leftrightarrow \lfloor P\rfloor[i]$
- $\text{Instance}_1\text{ProvablyComputable}\mathbb{R}_{[0,1]}\text{Enumerator}$

  ${:}([\mathbb{N}] \to \text{Instance}_1\text{ProvablyComputable}\mathbb{R}_{[0,1]})$

  - $(\forall[r{:}\text{Instance}_1\text{ProvablyComputable}\mathbb{R}_{[0,1]}]$

    $\exists[i{:}\mathbb{N}]\ r=\text{Instance}_1\text{ProvablyComputable}\mathbb{R}_{[0,1]}\text{Enumerator}_\centerdot[i]){:}\text{Instance}_1$
  - $(\forall[i{:}\mathbb{N}]$

    $\text{Instance}_1\text{ProvablyComputable}\mathbb{R}_{[0,1]}\text{Enumerator}_\centerdot[i]$

    ${:}\text{Instance}_1\text{ProvablyComputable}\mathbb{R}_{[0,1]}){:}\text{Instance}_1$

Instance_1 is inconsistent

Define Diagonal:[ℕ]→ $\text{Instance}_1\text{ProvablyComputableR}_{[0,1]}$

Diagonal.[i:ℕ] ≡ 1-(Instance_1ComputableEnumerator.[i]).[i]

Diagonal is not in the range of Instance_1ComputableEnumerator, which is a contradiction because

Instance_1 ⊢ Diagonal:$\text{Instance}_1\text{ProvablyComputableR}_{[0,1]}$

[Church 1934] pointed out that there is no obvious way to remove the inconsistency meaning that if Instance_1 is taken to be an exact description of logic[52] then,

"*Indeed, if there is no formalization of logic as a whole, then there is no exact description of what logic is, for it in the very nature of an exact description that it implies a formalization. And if there no exact description of logic, then there is no sound basis for supposing that there is such a thing as logic*."

Alonzo Church

Instance_1 does **not** stand as legitimate Mathematics because the axioms are "self-referential." Therefore, it makes sense to use Inconsistency Robust logic for Instance_1 instead of classical logic.

Discussion

Church's Paradox and other paradoxes raise a number of issues that can be addressed by requiring mathematics to be strongly typed and using higher order logic as follows:

1. Requiring Mathematics to be strongly typed using so that
   - Mathematics self proves that it is "open" in the sense that theorems are not computationally enumerable.[53]
   - Mathematics self proves that it is *formally* consistent.[54]
   - Strong mathematical theories for Natural Numbers, Ordinals, Set Theory, the Lambda Calculus, Actors, etc. are inferentially decidable, meaning that every true proposition is provable and every proposition is either provable or disprovable. Furthermore, theorems of these theories are not enumerable by a provably total procedure.
2. It was initially thought that mathematics could be based just on character strings. Then diagonalization was discovered and things haven't been the same since. The string for the general 1st order Nat_1 non-categorical induction ***schema*** is as follows for each $P{:}\mathrm{String}\lhd\mathrm{Proposition}\lhd 1\rhd^{\mathbb{N}}\rhd$:

   $$(\lfloor P\rfloor[0] \wedge \forall[i{:}\mathbb{N}]\ \lfloor P\rfloor[i] \Rightarrow \lfloor P\rfloor[i+1]) \Rightarrow \forall[i{:}\mathbb{N}]\ \lfloor P\rfloor[i]$$

   which has countably many 1st order propositions as instances that are abstracted from the countably many character strings of type $\mathrm{String}\lhd\mathrm{Proposition}\lhd 1\rhd^{\mathbb{N}}\rhd$ and which differs fundamentally from the character string for the ***more general*** 2nd order categorical induction ***axiom***, which is as follows:[55]

   $$\text{"}\forall[P{:}\mathrm{Proposition}\lhd 1\rhd^{\mathbb{N}}]\ (P[0] \wedge \forall[i{:}\mathbb{N}]\ P[i] \Rightarrow P[i+1]) \Rightarrow \forall[i{:}\mathbb{N}]\ P[i]\text{"}$$

Although the theory *Nat* has only ***finitely*** many axioms, the above string abstracted as a proposition has ***uncountably*** many 1st order propositions as instances.[i] In this way, *Nat* differs fundamentally from the 1st order theory Nat_1 because, being uncountable, ***not all*** instances of the *Nat* induction axiom can be obtained by abstraction from character strings. Proofs abstracted from character strings for the axioms of Nat_1 can be computationally enumerated and are valid proofs in *Nat*, but this does not enumerate ***all*** of the proofs of


Ernst Zermelo

[i] with the consequence that the argument in Church's Paradox is blocked in the theory *Nat* because theorems are not enumerable by a provably total procedure

Nat! What is to be made of the ***uncountable*** number of theorems of ***Nat*** whose proofs cannot be written down in text?

Additional limitations of Relational 1st order theories

> "By this it appears how necessary it is for nay man that aspires to true knowledge to examine the definitions of former authors; and either to correct them, where they are negligently set down, or to make them himself. For the errors of definitions multiply themselves, according as the reckoning proceeds, and lead men into absurdities, which at last they see, but cannot avoid, without reckoning anew from the beginning; in which lies the foundation of their errors..."
> [Hobbes *Leviathan*, Chapter 4][56]

A relational 1st order theory is very weak. For example, a relational 1st order theory is incapable of characterizing even the natural numbers, *i.e*., there are infinite integers in models of every relational 1st order axiomatization of the natural numbers. Furthermore, there are infinitesimal real numbers in models of every relational 1st order axiomatization of the real numbers.[i] Of course, infinite integers and infinitesimal reals are monsters that must be banned from the mathematical foundations of Computer Science.

However, some theoreticians have found relational 1st order theory to be useful for their careers because it is weak enough that they can prove theorems about relational 1st order axiomatizations whereas they cannot prove such theorems about stronger practical systems, *e.g*., Classical Direct Logic.[57]

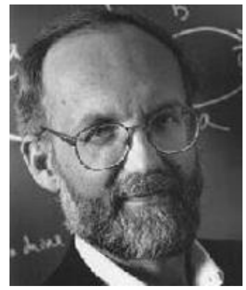
Zermelo considered the 1st Order Thesis to be a mathematical "hoax" because it necessarily allowed unintended models of axioms.[58]

[Barwise 1985] critiqued the 1st Order Thesis that mathematical foundations should be restricted to 1st order theories as follows:

---

[i] Likewise, relational 1st order set theory (*e.g.* ZFC) is very weak. See discussion in this article.

The reasons for the widespread, often uncritical acceptance of the first-order thesis are numerous. The first-order thesis ... confuses the subject matter of logic with one of its tools. First-order language is just an artificial language structured to help investigate logic, much as a telescope is a tool constructed to help study heavenly bodies. From the perspective of the mathematics in the street, the first-order thesis is like the claim that astronomy is the study of the telescope.[59]



Jon Barwise

Computer Science is making increasing use of Model Analysis[60] in the sense of analyzing relationships among the following:

- concurrent programs and their Actor Model denotations
- domain axiom systems and computations on these domains

In Computer Science, it is important that the natural numbers be axiomatized in a way that does not allow non-numbers (*e.g.* infinite ones) in models of the axioms.

Theorem: If $\mathbb{N}$ is a model of a 1^{st} order axiomatization $\mathcal{T}$, then $\mathcal{T}$ has a model $\mathbb{M}$ with an infinite integer.

Proof: The model $\mathbb{M}$ is constructed as an extension of $\mathbb{N}$ by adding a new element ∞ with the following atomic relationships:

$$\{\neg\infty<\infty\} \cup \{ m<\infty \mid m:\mathbb{N}\}$$

It can be shown that $\mathbb{M}$ is a model of $\mathcal{T}$ with an infinite integer ∞.

The infinite integer ∞ is a monster that must be banned from the mathematical foundations of Computer Science.

Theorem: If $\mathbb{R}$ is a model of a 1^{st} order axiomatization $\mathcal{T}$, then $\mathcal{T}$ has a model $\mathbb{M}$ with an infinitesimal.

Proof: The model $\mathbb{M}$ is constructed as an extension of $\mathbb{R}$ by adding a new element ∞ with the following atomic relationships:

$$\{\neg\infty<\infty\} \cup \{m<\infty \mid m:\mathbb{N}\}$$

Defining ε to be $\frac{1}{\infty}$, it follows that $\forall[r:\mathbb{R}]\ 0<\varepsilon<\frac{1}{r}$. It can be shown that $\mathbb{M}$ is a model of $\mathcal{T}$ with an infinitesimal ε, which is a monster that must be banned from the mathematical foundations of Computer Science.

On the other hand, since it is not limited to 1st order propositions, Classical Direct Logic characterizes structures such as natural numbers and real numbers up to isomorphism.[i]

There are many theorems that cannot be proved from 1st order axioms [Goodstein 1944, Simpson 1985, Wiles 1995, Bovykin 2009, McLarty 2010].

Unbounded Nondeterminism

Of greater practical import, 1st order theory is *not* a suitable foundation for the Internet of Things in which specifications require a device respond to a request.[ii] The specification that a computer responds can be formalized as follows: ∃[i:ℕ] ResponseBefore[i]. However, the specification cannot be proved in a 1st order theory.

Proof: In order to obtain a contradiction, suppose that it is possible to prove in a 1st order theory ∃**[**i:ℕ**]** ResponseBefore[i]. Therefore the infinite set of propositions {¬ResponseBefore[i] | i:ℕ} is inconsistent. By the compactness theorem of 1st order theory, it follows that there is finite subset of the set of propositions that is inconsistent. But this is a contradiction, because all the finite subsets are consistent since the amount of time before a server responds is unbounded, that is,

∄**[**i:ℕ**]** ⊢ResponseBefore[i].

However, the above specification axiom does *not* compute any actual output! Instead the above axiom simply asserts the *existence* of unbounded outputs for Unbounded.[ ].

---

i proving that software developers and computer systems are using the same structures

ii An implementation of such a system is given below in this article.

Theorem. The nondeterministic function defined by Unbounded (earlier in this article) cannot be implemented by a nondeterministic Logic Program[i] or a nondeterministic Turing Machine:

Proof.[61]

> *The task of a nondeterministic Logic Program* P *is to start with an initial set of axioms and prove* Output=n *for some numeral* n. *Now the set of proofs of* P *starting from initial axioms will form a tree. The branching points will correspond to the nondeterministic choice points in the program and the choices as to which rules of inference to apply. Since there are always only finitely many alternatives at each choice point, the branching factor of the tree is always finite. Now König's lemma says that if every branch of a finitary tree is finite, then so is the tree itself. In the present case this means that if every proof of* P *proves* Output=n *for some numeral* n*, then there are only finitely many proofs. So if* P *nondeterministically proves* Output=n *for every numeral* n*, it must contain a nonterminating computation in which it does not prove* Output=n *for some numeral* n.

The following arguments support unbounded nondeterminism in the Actor model [Hewitt 1985, 2006]:

- There is no bound that can be placed on how long it takes a computational circuit called an *arbiter* to settle. Arbiters are used in computers to deal with the circumstance that computer clocks operate asynchronously with input from outside, *e.g.*, keyboard input, disk access, network input, *etc.* So it could take an unbounded time for a message sent to a computer to be received and in the meantime the computer could traverse an unbounded number of states.
- Electronic mail enables unbounded nondeterminism since mail can be stored on servers indefinitely before being delivered.
- Communication links to servers on the Internet can be out of service indefinitely

1st order theory is **not** a suitable mathematical foundation for Intelligent Applications for the Internet of Things.

[i] the lambda calculus is a special case of Logic Programs

As a foundation of mathematics for Computer Science, Classical Direct Logic provides categorical[62] numbers (integer and real), sets, lists, trees, graphs, etc. which can be used in arbitrary mathematical theories including theories for categories, large cardinals, etc. These various theories might have "monsters" of various kinds. However, these monsters should not imported into models of computation used in Computer Science.

Computer Science needs *stronger* systems than provided by 1st order theory in order to weed out unwanted models. In this regard, Computer Science doesn't have a problem computing with "infinite" objects (*i.e.* Actors) such as π and uncountable sets such as the set of real numbers **Set**◁ℝ▷. However, the mathematical foundation of Computer Science is very different from the general philosophy of mathematics in which the infinite integers and infinitesimal reals allowed by models of 1st order theories may be of some interest. Of course, it is always possible to have special theories that are *not* part of the foundations with infinite integers, infinitesimal reals, unicorns, *etc*.[63]

Of course some problems are theoretically not computable. However, even in these cases, it is often possible to compute approximations and cases of practical interest.[i]

The mathematical foundation of Computer Science is very different from the general philosophy of mathematics in which infinite integers and infinitesimal reals may be of some interest. Of course, it is always possible to have special theories with infinite integers, infinitesimal reals, unicorns, *etc*.

[i] *e.g.* see Terminator [Knies 2006], which practically solves the halting problem for device drivers

Berry Paradox

The Berry Paradox [Russell 1906] can be formalized using the proposition Characterize[s, k] meaning that the string s characterizes the integer k as follows:

Characterize[s:String◁Proposition◁anOrder▷$^{\mathbb{N}}$▷, k:ℕ]:Proposition◁anOrder+1▷
≡ ∀[x:ℕ]⌊s⌋[x] ⇔ x=k

The Berry Paradox is to construct a string BString for the string for the proposition that holds for integer n if and only if every string with length less than 100 does not characterize n using the following definition:[64]

BString:String◁Proposition◁anOrder+1▷$^{\mathbb{N}}$▷ ≡
"⦅λ[n:ℕ] ∀[s:String◁Proposition◁anOrder▷$^{\mathbb{N}}$▷]
Length[s]<100 ⇨ ¬Characterize[s, n]⦆"

Note that

- Length[BString]<100.
- {s:String◁Proposition◁anOrder▷$^{\mathbb{N}}$▷ | Length[s]<100} is finite.
- Therefore, the following set is finite:
  {n:$\mathbb{N}_+$ | ∃[s:String◁Proposition◁anOrder▷$^{\mathbb{N}}$▷]
  Length[s]<100 ∧ Characterize[s, n]}

BSet:Set◁ℕ▷ ≡ {n:$\mathbb{N}_+$ | ⌊BString⌋[n]}

BSet≠{ } because is {n:ℕ | n≧1} is infinite.

1. BNumber:ℕ ≡ Least[BSet]
2. ⌊BString⌋[BNumber][65]

⌊⦅λ[n:ℕ] ⦅∀[s:String◁Proposition◁anOrder▷$^{\mathbb{N}}$▷]
Length[s]<100 ⇨ ¬Characterize[s, n]⦆⌋[BNumber][66]

3. ∀[s:String◁Proposition◁anOrder▷$^{\mathbb{N}}$▷]
   Length[s]<100 ⇨ ¬Characterize[s, BNumber][67]
4. Length[BString]<100 ⇨ ¬Characterize[BString, BNumber]
   // above is invalid because of attempted substitution of
   // BString:String◁Proposition◁anOrder+1▷$^{\mathbb{N}}$▷ for
   // s:String◁Proposition◁anOrder▷$^{\mathbb{N}}$▷

Appendix 2. Appendix 1. More Categorical Mathematical Theories

Theory of Nondeterministic Lambda Calculus ($\mathcal{Lam}_\tau$)

Definition: $\mathbb{Functional}\triangleleft\tau_1,\tau_2\triangleright \equiv [([\tau_1]\to\tau_2)]\to([\tau_1]\to\tau_2)$

Theory $\mathcal{Lam}_\tau$

In addition to Lambda Induction (above), the theory $\mathcal{Lam}_\tau$ has the following axioms:[i]

- $\textbf{Identity}\triangleleft\tau_1\triangleright:([\tau_1]\to\tau_1)$
  $\textbf{Identity}\triangleleft\tau_1\triangleright\blacksquare[f_1] = f_1$
- $\textbf{Const}\triangleleft\tau_1,\tau_2\triangleright:([\tau_1]\to([\tau_2]\to\tau_1))$
  $\textbf{Const}\triangleleft\tau_1, \tau_2\triangleright\blacksquare[f_1]\blacksquare[f_2] = f_1$
- $\textbf{Subst}\triangleleft\tau_1, \tau_2, \tau_3\triangleright:([[\tau_3]\to([\tau_2]\to\tau_4), [\tau_3]\to\tau_2], \tau_3] \to \tau_4)$[68]
  $\textbf{Subst}\triangleleft\tau_1, \tau_2, \tau_3\triangleright\blacksquare[f_1]\blacksquare[f_2]\blacksquare[f_3] = (f_1\blacksquare[f_3])\blacksquare[f_2\blacksquare[f_3]]$
- $\textbf{Fix}\triangleleft\tau_1,\tau_2\triangleright:([\mathbb{Functional}\triangleleft\tau_1,\tau_2\triangleright]\to\mathbb{Functional}\triangleleft\tau_1,\tau_2\triangleright)$[69]
  $\textbf{Fix}\triangleleft\tau_1,\tau_2\triangleright\blacksquare[F] = F\blacksquare[\textbf{Fix}\triangleleft\tau_1,\tau_2\triangleright\blacksquare[F]]$
- $\textbf{Either}\triangleleft\tau_1\triangleright:([\tau_1] \to ([\tau_1]\to\tau_1))$
  $\textbf{Either}\triangleleft\tau_1\triangleright\blacksquare[f_1]\blacksquare[f_2]=f_1 \lor \textbf{Either}\triangleleft\tau_1\triangleright\blacksquare[f_1]\blacksquare[f_2]=f_2$
- **Equality Axiom**
  $\forall[f_1,f_2:([\tau_1]\to\tau_2)]\ f_1=f_2 \Leftrightarrow \forall[f_3:\tau_1]\ f_1\blacksquare[f_3]=f_2\blacksquare[f_3]$
- **Lambda Equality**[ii]
  $\forall[f_1:([\tau]\to\tau)]\ f_1=\lambda[f_2:\tau]\ f_1\blacksquare[f_2]$
- **Basis:** For all f:$\Lambda\triangleleft\tau\triangleright$, f is equal to a composition of **Identity, Const, Subst, Fix,** and **Either**.

Theorem. Computational Inadequacy of Nondeterministic Lambda Calculus. The nondeterministic lambda calculus is inadequate to implement all computable procedures.

Proof. $F_i[j:\mathbb{N}] \equiv j>i$? **True**⦂ InfiniteLoop$\blacksquare$[ ] **False**⦂ i **either** $F_i[j+1]$
For each i:$\mathbb{N}$, F_i is a nondeterministic λ expression but (limit$_{i:\mathbb{N}}$ F_i) **cannot** be implemented as a nondeterministic λ expression. However (limit$_{i:\mathbb{N}}$ F_i):$\mathbb{Actor}$

---

[i] τ_1,τ_2,τ_3:$\mathbb{Type}\triangleleft\Lambda\triangleleft\tau\triangleright\triangleright$

[ii] Because of Lambda Equality, the domain of [Scott 2015] is *not* a valid model of $\mathcal{Lam}_\tau$.

Lambda Induction

The theorem of Lambda Induction is as follows:[i]

$$\forall[P{:}\mathrm{Proposition}\triangleleft 1\triangleright^{\Lambda\triangleleft\tau\triangleright}]$$

$$(P[\mathbf{Identity}\triangleleft\tau_1\triangleright] \wedge P[\mathbf{Const}\triangleleft\tau_1, \tau_2\triangleright] \wedge P[\mathbf{Subst}\triangleleft\tau_1,\tau_2,\tau_3\triangleright] \wedge P[\mathbf{Fix}\triangleleft\tau_1\triangleright]$$

$$\wedge\ P[\mathbf{Either}\triangleleft\tau_1\triangleright] \ \wedge \forall[f_1{:}\tau_1, f_2{:}\tau_2]\ P[f_1]\wedge P[f_2] \Rightarrow P[\mathbf{Const}\triangleleft\tau_1, \tau_2\triangleright\centerdot[f_1, f_2]]$$

$$\wedge \forall[f_1{:}\tau_1, f_2{:}\tau_2, f_3{:}\tau_3]\ P[f1]\wedge P[f2]\wedge P[f3]\Rightarrow P[\mathbf{Subst}\triangleleft\tau_1,\tau_2,\tau_3\triangleright\centerdot[f_1]\centerdot[f_2]\centerdot[f_3]]$$

$$\wedge \forall[f{:}([\tau_1]\rightarrow\tau_2)]\ P[f] \Rightarrow P[\mathbf{Fix}\triangleleft\tau_1, \tau_2\triangleright\centerdot[f]]$$

$$\wedge\ \forall[f_1{:}\tau_1, f_2{:}([\tau_1]\rightarrow\tau_2)]\ P[f_1]\wedge P[f_2] \ \Rightarrow P[f_2\centerdot[f_1]]) \ \Rightarrow \forall[f{:}\Lambda\triangleleft\tau\triangleright]\ P[f]$$

Convergence: $\forall[f_1{:}([\tau_1]\rightarrow\tau_2),f_2{:}\tau_1]\ f_1\centerdot[f_2]\downarrow \Leftrightarrow \exists[f_3{:}\tau_2]\ f_1\centerdot[f_2]=f_3$

Approximation: $\forall[f_1, f_2{:}([\tau_1]\rightarrow\tau_2)]\ f_1\leqq f_2 \Leftrightarrow \forall[f_3{:}\tau_1]\ f_1\centerdot[f_3]\downarrow \Rightarrow f_1\centerdot[f_3]=f_2\centerdot[f_3]$

Bottom: $\perp\triangleleft\tau_1\triangleright\centerdot[f{:}\tau_1] \equiv f$

Note that $\forall[f_2{:}\tau_1]\ \neg(\perp\triangleleft\tau_1\triangleright\centerdot[f_2])\downarrow$ and $\forall[f{:}([\tau_1]\rightarrow\tau_1)]\ \perp\triangleleft\tau_1\triangleright\leqq f$

Monotone:

$$F{:}\mathrm{Monotone}\triangleleft\tau_1,\tau_2\triangleright \Leftrightarrow F{:}\mathrm{Functional}\triangleleft\tau_1,\tau_2\triangleright \wedge \forall[g{:}([\tau_1]\rightarrow\tau_2)]\ g\leqq F\centerdot[g]$$

Limit Theorem: $\forall[F{:}\mathrm{Monotone}\triangleleft\tau_1,\tau_1\triangleright]\ F=\mathrm{limit}_{i:\mathbf{N}_+} F^i\centerdot[\perp\triangleleft\tau_1\triangleright]$[70]

Theorem: Deterministic procedures have bounded nondeterminism

$$\forall[f{:}([\tau_1]\rightarrow\tau_1))]\ \mathrm{Deterministic}[f] \Rightarrow f{:}\Lambda\triangleleft\tau\triangleright$$

Theorem: Some nondeterministic procedures have unbounded nondeterminism[ii]

$$\exists[f{:}([\tau_1]\rightarrow\tau_1))]\ \neg f{:}\Lambda\triangleleft\tau\triangleright$$

---

[i] $\tau_1,\tau_2,\tau_3{:}\mathbf{Type}\triangleleft\Lambda\triangleleft\tau\triangleright\triangleright$

[ii] e.g., ones using concurrent Actors. See discussion in this article.

Theorem.[i] Lam_τ is categorical with a unique isomorphism.

Proof: Suppose that $\mathbb{X}$ satisfies the axioms for Lam_τ.

By lambda induction, the isomorphism $I:\mathbb{X}^{\Lambda\triangleleft\tau\triangleright}$ is defined as follows:[ii]

- $I[\mathbf{Identity}\triangleleft\tau_1\triangleright] \equiv \mathbf{Identity}_{\mathbb{X}}\triangleleft\tau_1\triangleright$
- $I[\mathbf{Const}\triangleleft\tau_1, \tau_2\triangleright] \equiv \mathbf{Const}_{\mathbb{X}}\triangleleft\tau_1, \tau_2\triangleright$
- $I[\mathbf{Subst}\triangleleft\tau_1, \tau_2, \tau_3\triangleright] \equiv \mathbf{Subst}_{\mathbb{X}}\triangleleft\tau_1, \tau_2, \tau_3\triangleright$
- $I[\mathbf{Fix}\triangleleft\tau_1, \tau_2\triangleright] \equiv \mathbf{Fix}_{\mathbb{X}}\triangleleft\tau_1, \tau_2\triangleright$
- $I[\mathbf{Either}\triangleleft\tau_1\triangleright] \equiv \mathbf{Either}_{\mathbb{X}}\triangleleft\tau_1\triangleright$
- $\forall[f_1:\tau_1, f_2:([\tau_1]\rightarrow\tau_2)]\ I[f_2{\scriptstyle\blacksquare}[f_1]] \equiv I[f_2]{\scriptstyle\blacksquare}_{\mathbb{X}}[I[f_1]]$

I is the unique isomorphism:

- I is one to one
- The range of I is $\mathbb{X}$
- I is a homomorphism
- $I^{-1}:\Lambda\triangleleft\tau\triangleright^{\mathbb{X}}$ is a homomorphism
- I is the unique isomorphism: If $g:\mathbb{X}^{\Lambda\triangleleft\tau\triangleright}$ is an isomorphism, then g= I

Theorem (Model Soundness of Lam_τ): $(\vdash_{\mathit{Lam}_\tau}\Psi) \Rightarrow\ \vDash_{\Lambda\triangleleft\tau\triangleright}\Psi$

Proof: Suppose $\vdash_{\mathit{Lam}_\tau}\Psi$. The theorem immediately follows because the axioms for the theory Lam_τ hold in the type $\Lambda\triangleleft\tau\triangleright$.

Theorem (Indiscernibility for Lam_τ):[71]

$$\forall[f,g:\Lambda\triangleleft\tau\triangleright]\ \ f{=}g \Leftrightarrow \forall[P:\mathbb{Proposition}\triangleleft1\triangleright^{\mathbb{N}}]\ \ P[f]\Leftrightarrow P[g]$$

Theorem: Logical completeness of Lam_τ

$$\forall[\Psi:\mathbb{Proposition}\triangleleft\mathit{Lam}_\tau\triangleright]\ \ (\vDash_{\Lambda\triangleleft\tau\triangleright}\Psi) \Rightarrow\ \vdash_{\mathit{Lam}_\tau}\Psi$$

Corollary. Equivalence of satisfiability and provability in Lam_τ, i.e.,

$$\forall[\Psi:\mathbb{Proposition}\triangleleft\mathit{Lam}_\tau\triangleright]\ (\vDash_{\Lambda\triangleleft\tau\triangleright}\Psi) \Leftrightarrow\ \vdash_{\mathit{Lam}_\tau}\Psi$$

---

[i] cf. [Engeler 1981; Hindley, and Seldin 2008]

[ii] τ_1,τ_2,τ_3:$\mathbb{Type}\triangleleft\Lambda\triangleleft\tau\triangleright\triangleright$

Theorem. Inferential Decidability of Lam_τ, i.e.,

$$\forall[P:\mathbb{Proposition}\triangleleft \mathit{Lam}_\tau\triangleright^{\Lambda\triangleleft\tau\triangleright}]$$
$$(\vdash_{\mathit{Lam}_\tau}\forall[f:\Lambda\triangleleft\tau\triangleright]\ P[f]) \vee \vdash_{\mathit{Lam}_\tau}\exists[f:\Lambda\triangleleft\tau\triangleright]\ \neg P[f]$$

Theory of Reals (*Reals*)

Reals is strictly more powerful than the relational 1st order theory of ***RealClosedFields***.[72]

Theorem (Categoricity of *Reals*):[73]

If **X** is a type satisfying the axioms[74] for the real numbers ***Reals***, then there is a unique isomorphism with $\mathbb{R}$.

Theory of Ordinals (*Ord*)

A theory of the ordinals can be axiomatized[75] using a 2nd order ordinal induction axiom as follows: For each order:$\mathbb{N}_+$ and P:$\mathbb{Proposition}$◁order▷$^{\mathbb{O}}$,

$$(\forall[\alpha:\mathbb{O}]\ \forall[\beta<\alpha:\mathbb{O}]\ P[\beta]\Rightarrow P[\alpha]) \Rightarrow \forall[\alpha:\mathbb{O}]\ P[\alpha]$$

In order to fill out the ordinals, the following limit axioms are included in ***Ord***:

- $\forall[\alpha:\mathbb{O},\ f:\mathbb{O}^{\mathbb{O}}]\ \bigcup_\alpha f:\mathbb{O}$
- $\forall[\alpha,\beta:\mathbb{O};\ f:\mathbb{O}^{\mathbb{O}}]\ \beta<\bigcup_\alpha f \Leftrightarrow \exists[\delta<\alpha]\ \beta\leqq f[\delta]$
- $\forall[\alpha,\beta:\mathbb{O};\ f:\mathbb{O}^{\mathbb{O}}]\ (\forall[\delta<\alpha]\ f[\delta]\leqq\beta) \Rightarrow \bigcup_\alpha f\leqq\beta$

In order to guarantee that there are uncountable ordinals, the following axioms are also included in ***Ord***:

- $\omega_0 = \mathbb{N}$
- $\forall[\alpha:\mathbb{O}]\ \alpha>0_{\mathbb{O}} \Rightarrow \omega_\alpha \prec \mathbf{Boolean}^{\bigcup_{\beta<\alpha}\omega_\beta}$
- $\forall[\alpha,\beta:\mathbb{O}]\ \beta\doteq\omega_\alpha \Rightarrow \omega_\alpha\leqq\beta$

  where $\tau_1\doteq\tau_2 \Leftrightarrow \exists[f:{\tau_2}^{\tau_1}]\ \mathrm{1to1Onto}\triangleleft\tau_1,\tau_2\triangleright[f]$
  - $\mathrm{1to1}\triangleleft\tau_1,\tau_2\triangleright[f:{\tau_2}^{\tau_1}] \Leftrightarrow \forall[x_1,x_2:\tau_1]\ f[x1]=f[x2] \Rightarrow x1=x2$
  - $\mathrm{1to1Onto}\triangleleft\tau_1,\tau_2\triangleright[f:{\tau_2}^{\tau_1}]$
    $\Leftrightarrow \mathrm{1to1}\triangleleft\tau_1,\tau_2\triangleright[f:{\tau_2}^{\tau_1}] \wedge \forall[y:\tau_2]\ \exists[x:\tau_1]\ f[x]=y$

Theorem Ordinals have the following properties:

- Ordinals are well-ordered:

  Least:$\mathbf{O}^{\mathbf{Boolean}^{\mathbf{O}}}$

  Least[{ }] = $0_{\mathbb{O}}$

  $\forall$[S:$\mathbb{Boolean}^{\mathbb{O}}$] S≠{ } ⇨ Least[S]∈S

  $\forall$[S:$\mathbb{Boolean}^{\mathbb{O}}$] S≠{ } ⇨ $\forall$[α∈S] Least[S]≦α
- Reals can be well-ordered because $\omega_1 \doteq \mathbb{R}$
- $\forall$[α:$\mathbb{O}$] ∃[β:$\mathbb{O}$] $\alpha<\omega_\beta$
- The set of all ordinals Ω is $\mathbb{Boolean}^{\mathbb{O}}$ so that:

  $\forall$[α:$\mathbb{O}$] α∈Ω ⇔ α:$\mathbb{O}$

  Note that it is **not** the case that Ω is of type $\mathbb{O}$, thereby thwarting the Burali-Forti paradox

Theorem (Categoricity of *Ord*):
If **X** be a type satisfying the axioms the theory of the ordinals *Ord*, then there is a unique isomorphism with $\mathbb{O}$.[76]

Theorem (Model Soundness of *Ord*): $(\vdash_{\mathit{Ord}} \Psi) \Rightarrow\ \vDash_{\mathbb{O}} \Psi$

Proof: Suppose $\vdash_{\mathit{Ord}} \Psi$. The theorem immediately follows because the axioms for the theory *Ord* hold in the type $\mathbb{O}$.

Theorem (Indiscernibility for *Ord*):[77]

$\forall$[α,β:$\mathbb{O}$] α=β ⇔ $\forall$[P:$\mathbb{Proposition}$⊲1⊳$^{\mathbb{N}}$] P[α]⇔P[β]

Theorem: Logical completeness of *Ord*

$\forall$[Ψ:$\mathbb{Proposition}$⊲*Ord*⊳] $(\vDash_{\mathbb{O}} \Psi) \Rightarrow\ \vdash_{\mathit{Ord}} \Psi$

Corollary. Equivalence of satisfiability and provability in *Ord*, i.e.,

$\forall$[Ψ:$\mathbb{Proposition}$⊲*Ord*⊳] $(\vDash_{\mathbb{O}} \Psi) \Leftrightarrow\ \vdash_{\mathit{Ord}} \Psi$

Theorem. Inferential Decidability of *Ord*, i.e.,

$\forall[\Psi:\text{Proposition}\triangleleft \mathit{Ord}\triangleright^{\mathrm{O}}]\ (\vdash_{\mathit{Ord}} \Psi) \vee \vdash_{\mathit{Ord}} \Psi$

Proof. $\forall[\Psi:\text{Proposition}\triangleleft \mathit{Ord}\triangleright]\ (\vDash_{\mathrm{O}} \Psi) \vee \vDash_{\mathrm{O}} \Psi$

Theorem follows from Equivalence of satisfiability and provability in *Ord*.

Type Choice

$\forall[f:(\mathbf{Boolean}^{\sigma})^{\tau}]\ \exists[\text{choice}:\sigma^{\tau}]\ \forall[x:\tau]\ f[x]\neq\{\} \Rightarrow \text{choice}[x]\in f[x]$

Sets_{τ} defined using strong parameterized types

Set Theory

A theory of the ordinals can be axiomatized using a 2nd order set induction axiom as follows: For each order:$\mathbb{N}_{+}$ and P:Proposition◁order▷$^{\mathrm{O}}$:

$(\forall[S:\text{Set}\triangleleft\tau\triangleright, \alpha:\mathrm{O}]\ (S\doteq\alpha \Rightarrow \forall[X:\text{Set}\triangleleft\tau\triangleright, \beta<\alpha:\mathrm{O}]\ P[X]\wedge X\doteq\beta \Rightarrow P[X])$

$\Rightarrow \forall[S:\text{Set}\triangleleft\tau\triangleright]\ P[S]$

The type Set◁τ▷ can be characterized as follows:

$\text{Set}\triangleleft\tau\triangleright \equiv \text{Boolean}^{\tau}$

Of course set membership is defined as follows:

$\forall[x:\tau:, S:\text{Set}\triangleleft\tau\triangleright]\ x\in S \Leftrightarrow S[x]=\text{True}$

Inductive definition:

1. $\text{Sets}_{0}\triangleleft\tau\triangleright \equiv \text{Boolean}^{\tau}$
2. $\text{Sets}_{\alpha+1}\triangleleft\tau\triangleright \equiv \text{Set}\triangleleft\text{Sets}_{\alpha}\triangleleft\tau\triangleright\triangleright$
3. $\alpha:\text{Limit}\triangleleft\mathrm{O}\triangleright \Rightarrow (S:\text{Sets}_{\alpha}\triangleleft\tau\triangleright \Leftrightarrow \forall[X\in S]\ \exists[\beta<\alpha:\mathrm{O}, Y:\text{Sets}_{\beta}\triangleleft\tau\triangleright]\ X\in Y)$

$S:\text{Sets}\triangleleft\tau\triangleright \Leftrightarrow \exists[\alpha:\mathrm{O}]\ S:\text{Sets}_{\alpha}\triangleleft\tau\triangleright$

The properties below mean that Sets◁τ▷ is a "universe" of mathematical discourse.[78]

- Foundation: There are no downward infinite membership chains.[79]
- Transitivity of ∈[80]: ∀[S:Sets◁τ▷] ∀[X∈S] X:Sets◁τ▷
- Powerset:[81] ∀[S:Sets◁τ▷] Boolean^{S}:Sets◁τ▷
- Union:[82]
  ∀[S:Sets◁τ▷] ∪S:Sets◁τ▷
  ∀[S:Sets◁τ▷] ∀[X:Sets◁τ▷] X∈∪S ⇔ ∃[Y∈S] X∈Y
- Replacement:[83] The function image of any set is also a set, *i.e.*:
  $\text{Image}\triangleleft\tau\triangleright:\mathbf{Sets}\triangleleft\tau\triangleright^{[\mathbf{Sets}\triangleleft\tau\triangleright^{\mathbf{Sets}\triangleleft\tau\triangleright},\ \mathbf{Sets}\triangleleft\tau\triangleright]}$
  $\forall[f:\mathbf{Sets}\triangleleft\tau\triangleright^{\mathbf{Sets}\triangleleft\tau\triangleright},\ S:\text{Sets}\triangleleft\tau\triangleright]$
  ∀[y:Sets◁τ▷] y∈Image◁τ▷[f, S] ⇔ ∃[x∈S] f[x]=y

Sets◁τ▷ is *much stronger* than relational 1^{st} order ZFC.[84]

Theorem. *Sets*$_τ$ is categorical with a unique isomorphism.

Proof:[85] Suppose that **X** satisfies the axioms for ***Sets***$_τ$.

By ordinal induction, the isomorphism I:$\mathbb{X}^{\text{Sets}◁τ▷}$ as follows:

1. S:$\mathbb{S}\text{ets}_0$◁τ▷

   I[S] ≡ S

2. S:$\mathbb{S}\text{ets}_{α+1}$◁τ▷

   Z$\in_{\mathbb{X}}$I[S] ⇔ ∃[Y:$\mathbb{S}\text{ets}_α$◁τ▷] I[Y]$\in_{\mathbb{X}}$Z

3. S:$\mathbb{S}\text{ets}_α$◁τ▷and α:$\mathbb{L}$imit◁$\mathbb{O}$▷

   Z$\in_{\mathbb{X}}$I[S] ⇔ ∃[β<α:$\mathbb{O}$, Y:$\mathbb{S}\text{ets}_β$◁τ▷] I[Y]$\in_{\mathbb{X}}$Z

4. I is a unique isomorphism:
   - I is one to one
   - The range of I is $\mathbb{X}$
   - I is a homomorphism:
     - I[**{ }**$_{\text{Sets}◁τ▷}$] = **{ }**$_{\mathbb{X}}$
     - ∀[S1,S2:$\mathbb{S}$ets◁τ▷] I [S1 ∪ S2] = I[S1] $\cup_{\mathbb{X}}$ I [S2]
     - ∀[S1 S2:$\mathbb{S}$ets◁τ▷] I[S1 ∩ S2] = I[S1] $\cap_{\mathbb{X}}$ I[S2]
     - ∀[S1,S2:$\mathbb{S}$ets◁τ▷] I[S1 - S2] = I[S1] $-_{\mathbb{X}}$ I[S2]
     - ∀[S:$\mathbb{S}$ets◁τ▷] I[⋃S] = $\bigcup_{\mathbb{X}}$ {I[x] | x∈S}
   - I^{-1}:$\mathbb{S}\text{ets}◁τ▷^{\mathbf{X}}$ is a homomorphism
   - I is the unique isomorphism: If g:$\mathbb{X}^{\text{Sets}◁τ▷}$ is an isomorphism, then g= I

Theorem (Model Soundness of *Sets*$_τ$): ($\vdash_{\boldsymbol{Sets}_τ}$**Ψ**) ⇨ $\vDash_{\text{Sets}◁τ▷}$**Ψ**

Proof: Suppose $\vdash_{\boldsymbol{Sets}_τ}$**Ψ**. The theorem immediately follows because the axioms for the theory ***Sets***$_τ$ hold in the type **Sets**◁τ▷.

Theorem (Indiscernibility for *Sets*$_τ$):[86]

∀[s_1,s_2:$\mathbb{S}$ets◁τ▷] s_1=s_2 ⇔ ∀[P:$\mathbb{P}\text{roposition}◁1▷^{\mathbb{N}}$] P[$s_1$]⇔P[$s_2$]

Theorem: Logical completeness of Sets_τ

$\forall[\Psi{:}\mathrm{Proposition}\triangleleft \mathit{Sets}_\tau\triangleright^{\mathrm{Sets}\triangleleft\tau\triangleright}]\ (\vDash_{\mathrm{Sets}\triangleleft\tau\triangleright}\Psi) \Rightarrow\ \vdash_{\mathit{Sets}_\tau}\Psi$

Corollary. Equivalence of satisfiability and provability in Sets_τ, i.e.,

$\forall[\Psi{:}\mathrm{Proposition}\triangleleft \mathit{Sets}_\tau\triangleright]\ (\vDash_{\mathrm{Sets}\triangleleft\tau\triangleright}\Psi)\Leftrightarrow\vdash_{\mathit{Sets}_\tau}\Psi$

Theorem. Inferential Decidability of Sets_τ, i.e.,

$\forall[\Psi{:}\mathrm{Proposition}\triangleleft \mathit{Sets}_\tau\triangleright]\ \ (\vdash_{\mathit{Sets}_\tau}\Psi) \vee \vdash_{\mathit{Sets}_\tau}\neg\Psi$

Appendix 3: Notation of Direct Logic

Types *i.e.*, a type is a *discrimination*[87] of the following:[88]

- Boolean::[89], ℕ::, O::[90] and Act::[91]
- Term◁τ▷::[92], Expression◁τ▷::[93], Λ◁τ▷[94], String◁τ▷::[95], and Type◁τ▷::[96], where τ::[97]
- Proposition◁anOrder▷::[98] and Sentence◁anOrder▷▷:: where anOrder:$\mathbb{N}_+$
- $(\tau_1 ⦶ \tau_2)$::[99], $[\tau_1,\tau_2]$::[100], $([\tau_1]\rightarrow\tau_2)$::[101] and $\tau_2{}^{\tau_1}$::[102] where τ_1:: and τ_2::
- (τ∌P):: where τ:: and P:Proposition◁1▷$^{\tau}$ [103]

Propositions, *i.e.*, a Proposition is a *discrimination* of the following:

- (¬Φ):τ where Φ:τ and τ::
- Φ∧Ψ,Φ∨Ψ,Φ⇨Ψ,(Φ⇔Ψ):τ where Φ, Ψ:τ and τ::
- (p ⍰ True⸽ Φ_1, False⸽ Φ_2):τ[104] where p:Boolean, Φ,Ψ:τ and τ::
- $(x_1=x_2)$:Proposition◁1▷ where x_1,x_2:τ and τ::
- $(s_1 \subseteq s_2)$:Proposition◁1▷ where s_1,s_2:Set◁τ▷ and τ::
- (x∈s):Proposition◁1▷ where x:τ, s:Set◁τ▷ and τ::
- $(\tau_1 \sqsubseteq \tau_2)$:Proposition◁1▷[105] where τ_1:: and τ_2::
- (x::):Proposition◁1▷[106]
- (x:τ):Proposition◁1▷ where τ::
- p[x]:Proposition◁anOrder+1▷ where x:τ, p:Proposition◁anOrder▷$^{\tau}$
- $(\Phi_1 \vdash^{p}_{T} \Phi_2)$:τ[107] where p:Proof, T:Theory, Φ_1:τ, Φ_2:τ and τ⊑Proposition◁anOrder▷ and anOrder:$\mathbb{N}_+$
- $(\forall\tau_1\ p)$:τ_2[i] and $(\exists\tau_1\ p)$:τ_2[ii] where p:$\tau_2{}^{\tau_1}$, τ_1::, τ⊑Proposition◁anOrder▷ and anOrder:$\mathbb{N}_+$
- $(\models_{T}\Phi)$:τ where Φ:τ, *T*:Theory[108], τ⊑Proposition◁anOrder▷ and anOrder:$\mathbb{N}_+$
- ⌊s⌋:Proposition◁anOrder▷ where s:Sentence◁anOrder▷ with no free variables and anOrder:$\mathbb{N}_+$

---

[i] meaning $\forall[x:\tau_1]:\tau_2$ p[x]

[ii] meaning $\exists[x:\tau_1]:\tau_2$ p[x]

Grammar (syntax) trees (*i.e.* terms, expressions and sentences) are defined below.

Terms, *i.e.,* a Term◁τ▷ is a *discrimination* of the following:

- ⦅Boolean⦆:Constant◁Type◁Boolean▷▷, ⦅ℕ⦆:Constant◁Type◁ℕ▷▷, ⦅ℴ⦆:Constant◁Type◁ℴ▷▷ and ⦅Act⦆:Constant◁Type◁Act▷▷
- **x**:Term◁τ▷ where **x**:Constant◁τ▷ and **τ**::
- **x**:Term◁τ▷ where **x**:Variable◁τ▷ and **τ**::
- ⦅**f**[**x**:τ_1]:τ_2 ≡ **d in y**⦆:Term◁τ_3▷ where **f**:Variable◁$\tau_2{}^{\tau_1}$▷ in **d** and **y**, **x**:Variable◁τ_1▷ in **d**, **d**:Term◁τ_2▷, **y**:Term◁τ_3▷, and τ_1,τ_2,τ_3:: [109]
- ⦅**x**:τ_1 ≡ **d in y**⦆:Term◁τ_2▷ where **x**:Variable◁τ_1▷ in **d** and **y**, **d**:Term◁τ_1▷, **y**:Term◁τ_2▷, and τ_1,τ_2:: [110]
- ⦅t_1⦶t_2⦆:Term◁τ_1⦶τ_2▷, ⦅[t_1, t_2]⦆:Term◁[τ_1, τ_2]▷, ⦅[t_1]→t_2⦆:Term◁[τ_1]→τ_2▷ and ⦅$e_2{}^{e_1}$⦆:Term◁$\tau_2{}^{\tau_1}$▷ where t_1:Term◁τ_1▷, t_2:Term◁τ_2▷, τ_1:: and τ_2::
- ⦅t_1 ⍰ **True**⁝ t_2 **, False**⁝ t_3⦆:Term◁τ▷[i] where t_1:Term◁Boolean▷, t_2,t_3:Term◁τ▷ and **τ**::
- ⦅λ[**x**:τ_1]:τ_2 **t**⦆:Term◁$\tau_2{}^{\tau_1}$▷ where **t**:Term◁τ_2▷, **x**:Variable◁τ_1▷ in **t**, and τ_1,τ_2::
- ⦅**t**[**x**]⦆:Term◁τ_2▷ where **t**:Term◁$\tau_2{}^{\tau_1}$▷, **x**:Term◁τ_1▷, τ_1,τ_2::
- ⌊**t**⌋:**τ** where **t**:Term◁τ▷ with **no** free variables and **τ**::

[i] ⦅*if* e_1 *then* e_2 *else* e_3⦆

Expressions, *i.e.,* an Expression◁τ▷ is a *discrimination* of the following:

- **x**:Expression◁τ▷ where **x**:Constant◁τ▷ and **τ**::
- **x**:Expression◁τ▷ where **x**:Identifier◁τ▷ and **τ**::
- ⦅**f**▪[**x**:τ_1]:τ_2 ≡ **d** **in** **y**⦆:Expression◁τ_3▷ where **f**:Identifier◁$\tau_2^{\tau_1}$▷ in **d** and **y**, **x**:Identifier◁τ_1▷ in **d**, **d**:Expression◁τ_2▷, **y**:Expression◁τ_3▷, and τ_1,τ_2,τ_3:: [111]
- ⦅**x**:τ_1 ≡ **d** **in** **y**⦆:Expression◁τ_2▷ where **x**:Identifier◁τ_1▷ in **d** and **y**, **d**:Expression◁τ_1▷, **y**:Expression◁τ_2▷, and τ_1,τ_2:: [112]
- ⦅e_1⦶e_2⦆:Expression◁τ_1⦶τ_2▷, ⦅[e_1, e_2]⦆:Expression◁[τ_1, τ_2]▷, ⦅[e_1]→e_2⦆:Expression◁[τ_1]→τ_2▷ and ⦅$e_2^{e_1}$⦆:Expression◁$\tau_2^{\tau_1}$▷ where e_1:Expression◁τ_1▷, e_2:Expression◁τ_2▷, τ_1:: and τ_2::
- ⦅e_1 ⍰ **True**⦂ e_2, **False**⦂ e_3⦆:Expression◁τ▷[i] where e_1:Expression◁Boolean▷, e_2,e_3:Expression◁τ▷ and **τ**::
- ⦅λ[**x**:τ_1]τ_2 **e**⦆:Expression◁$\tau_2^{\tau_1}$▷ where **e**:Expression◁τ_2▷, **x**:Identifier◁τ_1▷ in **e**, and τ_1,τ_2::
- ⦅**e**▪[**x**]⦆:Expression◁τ_2▷ where **e**:Expression◁[τ_1]→τ_2▷, **x**:Expression◁τ_1▷, τ_1:: and τ_2::
- Sentence◁anOrder▷⊑Term◁Sentence◁anOrder▷▷ and Sentence◁anOrder▷⊑Expression◁Sentence◁anOrder▷▷ where anOrder:$\mathbb{N}_+$[ii]
- ⌊**e**⌋:**τ** where **e**:Expression◁τ▷ with **no** free identifiers and **τ**::

---

[i] ⦅*if* e_1 *then* e_2 *else* e_3⦆

[ii] Sentences are both Terms and Expressions in order to facilitate writing functions and procedures over Terms.

Sentences, *i.e.,* a Sentence is a *discrimination* of the following:

- $(\!|\mathbf{x}|\!)$:Sentence◁anOrder+1▷[i] where x:Variable◁Sentence◁anOrder▷▷ and anOrder:$\mathbb{N}_+$
- $(\!|\neg s|\!):\tau$ where $s:\tau$ and τ::
- $(\!|s_1 \wedge s_2|\!), (\!|s_1 \vee s_2|\!), (\!|s_1 \Rightarrow s_2|\!), (\!|s_1 \Leftrightarrow s_2|\!):\tau$ where $s_1, s_2:\tau$ and τ::
- $(\!|$e [?] True⁝ s_1, False⁝ $s_2|\!)$[ii]$:\tau$ where e:Expression◁Boolean▷, $s_1, s_2:\tau$ and τ::
- $(\!|e_1 = e_2|\!)$:Sentence◁1▷ where e_1, e_2:Expression◁τ▷ and τ::
- $(\!|e_1 \sqsubseteq e_2|\!)$:Sentence◁1▷ where e_1, e_2:Expression◁τ_1▷, $\tau_1:\tau_2$ and τ_2::
- $(\!|e_1 \subseteq e_2|\!)$:Sentence◁1▷ where e_1, e_2:Expression◁Set◁τ▷▷ and τ::
- $(\!|e_1 \in e_2|\!)$:Sentence◁1▷ where e_1:Expression◁Set◁τ▷▷, e_2:Expression◁Set◁τ▷▷ and τ::
- $(\!|e_1 : e_2|\!)$:Sentence◁1▷ where e_1:Expression◁τ_1▷, e_2:Expression◁τ_2▷ and τ_1, τ_2::
- $(\!|e::|\!)$:Sentence◁1▷ where e:Expression◁τ▷ and τ::
- $(\!|\forall[x:\tau_1]:\tau_2\ s|\!):\tau_2$ and $(\!|\exists[x:\tau_1]:\tau_2\ s|\!):\tau_2$ where x:Variable◁τ_1▷, $s:\tau_2$ and τ_1, τ_2::
- $(\!|p[x]|\!)$:Sentence◁anOrder+1▷[113] where x:Expression◁τ▷, p:Expression◁Sentence◁anOrder▷$^{\text{Expression}\triangleleft\tau\triangleright}$▷, τ:: and anOrder:$\mathbb{N}_+$ [114]
- $(\!|s_1 \vdash^{p}_{T} s_2|\!):\tau$ where T:Expression◁Theory▷, $s_1:\tau$, $s_2:\tau$, p:Expression◁Proof▷ and τ::
- $(\!|\vDash_T s|\!):\tau$ where $s:\tau$, T:Expression◁Theory▷ and τ::
- Sentence◁anOrder▷⊑Term◁Sentence◁anOrder▷▷ and Sentence◁anOrder▷⊑Expression◁Sentence◁anOrder▷▷ where anOrder:$\mathbb{N}_+$[115]
- ⌊s⌋:Proposition◁anOrder▷ where s:Sentence◁anOrder▷, anOrder:$\mathbb{N}_+$ and there are **no** free variables in s.[iii]

---

[i] The type of $(\!|\mathbf{x}|\!)$ means that the Y fixed point construction cannot be used to construct sentences for "self-referential" propositions in Direct Logic.

[ii] *if* **e** *then* $\mathbf{s}_1$ *else* $\mathbf{s}_1$

[iii] The type binding achieves much of what Russel sought to achieve in the ramified theory of types. [Russell and Whitehead 1910-1913]

Strings for sentences, *i.e.,* a string for a sentence is a *discrimination* of the following where anOrder:ℕ₊:

- "x":String◁Sentence◁anOrder+1▷▷[116] where x:Variable◁String◁Sentence◁anOrder▷▷▷ and
- "¬" s":τ where s:τ and τ⊑String◁Sentence◁anOrder▷▷
- "s_1 "∧" s_2","s_1 "∨" s_2","s_1 "⇨" s_2","s_1 "⇔" s_2":τ where s_1,s_2:τ and τ⊑String◁Sentence◁anOrder▷▷
- "e "?" "True⁸" s_1 "," "False⁸" s_2":τ where e:String◁Expression◁Boolean▷▷, s_1,s_2:τ and τ⊑String◁Sentence◁anOrder▷▷
- "e_1 "=" e_2":String◁Sentence◁1▷▷ where e_1,e_2:String◁Expression◁τ▷▷ and τ⊑String◁Sentence◁anOrder▷▷
- "e_1 "⊑" e_2":String◁Sentence◁1▷▷ where e_1,e_2:String◁Term◁τ_1▷▷, τ_1:τ_2 and τ_2::
- "e_1 "⊆" e_2":String◁Sentence◁1▷▷ where e_1,e_2:String◁Term◁Set◁τ▷▷▷ and τ::
- "e_1 "∈" e_2":String◁Sentence◁1▷▷ where e_1:String◁Term◁τ▷▷, e_2:String◁Term◁Set◁τ▷▷▷ and τ::
- "e_1 ":" e_2":String◁Sentence◁1▷▷ where e_1:String◁Expression◁τ_1▷▷, e_2:String◁Expression◁τ_2▷▷, τ_1:τ_3, τ_2:τ_4 and τ_3,τ_4::
- "e "::"":String◁Sentence◁anOrder▷▷ where e:String◁Expression◁τ▷▷ and τ::
- ""∀[" x ":" τ_1 "]:" τ_2 s":τ_2 and ""∃[" x ":" τ_1 "]:" τ_2 s":τ_2 where x:Variable◁τ_1▷, s:τ_2 and τ_1,τ_2⊑String◁Sentence◁anOrder▷▷
- "p "[" x "]"":String◁Sentence◁anOrder+1▷▷[i] where x:String◁Expression◁τ▷▷, τ:: , p:String◁Expression◁Sentence◁anOrder▷^String◁Expression◁τ▷ ▷▷,
- "s_1 "⊢" $\frac{p}{T}$ s_2":τ where T:String◁Expression◁Theory▷, s_1:τ, s_2:τ p:String◁Expression◁Proof▷▷ and τ⊑String◁Sentence◁anOrder▷▷
- String◁Sentence◁anOrder▷▷⊑String◁Term◁Sentence◁anOrder▷▷▷ and String◁Sentence◁anOrder▷▷⊑String◁Expression◁Sentence◁anOrder▷▷▷[117]
- ⌊s⌋:Sentence◁anOrder▷ where s:String◁Sentence◁anOrder▷▷

---

[i] The type of "p "[" x "]"" means that the Y fixed point construction cannot be used to construct strings for "self-referential" propositions in Direct Logic.

String for terms, *i.e.,* a string for a term is a *discrimination* of the following:

- **x**:String◁Term◁**τ**▷▷ where **x**:String◁Constant◁**τ**▷▷ and **τ**::
- **x**: String◁Term◁**τ**▷▷ where **x**:String◁Variable◁**τ**▷▷ and **τ**::
- “"⦅" **f** "[" **x** ":" τ_1 "]:" τ_2 "≡" **d** "in" **y** "⦆"”:String◁Term◁τ_3▷▷ where **f**:String◁Variable◁$\tau_2{}^{\tau_1}$▷▷ in **d** and **y**, **x**:String◁Variable◁τ_1▷▷ in **d**, **d**:String◁Term◁τ_2▷▷, **y**:String◁Term◁τ_3▷▷, and τ_1,τ_2,τ_3:: [118]
- “"⦅" **x** ":" τ_1 "≡" **d** "in" **y** "⦆"”:String◁Term◁τ_2▷ where **x**:String◁Variable◁τ_1▷▷ in **d** and **y**, **d**:String◁Term◁τ_1▷▷, **y**:String◁Term◁τ_2▷▷, and τ_1,τ_2:: [119]
- “"⦅" e_1 "⦶" e_2 "⦆"”:String◁Term◁τ_1⦶τ_2▷▷,
- “"⦅" "[" e_1 "," e_2"]" "⦆"”:String◁Term◁[τ_1,τ_2]▷▷,
  “"⦅" "[" e_1 "]→" e_2 "⦆"”:String◁Term◁[τ_1]→τ_2▷▷, and
  “"⦅" $e_2{}^{e_1}$ "⦆"”:String◁Term◁$\tau_2{}^{\tau_1}$▷▷ where e_1:String◁Term◁τ_1▷▷, e_2:String◁Term◁τ_2▷▷, and τ_1:: and τ_2::
- “"⦅" e_1 "⍰" "**True**⸰" e_2 "," "**False**⸰" e_3 "⦆"”:String◁Term◁**τ**▷▷[i] where e_1:String◁Term◁Boolean▷▷, e_2,e_3:String◁Term◁**τ**▷▷ and **τ**::
- “"⦅" "λ[" **x** ":" τ_1 "]:" τ_2 **e** "⦆"”:String◁Term◁$\tau_2{}^{\tau_1}$▷▷ where **e**:String◁Term◁τ_2▷▷, **x**:String◁Variable◁τ_1▷▷ in **e**, and τ_1,τ_2::
- “"⦅" **e** "[" **x** "]" "⦆"”:String◁Term◁τ_2▷▷ where **e**:String◁Term◁$\tau_2{}^{\tau_1}$▷▷, **x**:String◁Term◁τ_1▷▷, τ_1:: and τ_2::
- String◁Sentence◁anOrder▷▷⊑String◁Term◁Sentence◁anOrder▷▷▷ where anOrder:$\mathbb{N}_+$
- ⌊**e**⌋:Term◁**τ**▷, where **e**:String◁Term◁**τ**▷ and **τ**::

---

[i] “*if* e_1 *then* e_2 *else* e_3”

- ***String for expressions***, *i.e.,* a string for an expression is a *discrimination* of the following:
  - **x**:String◁Expression◁τ▷▷ where **x**:String◁Constant◁τ▷▷ and τ::
  - **x**: String◁Expression◁τ▷▷ where **x**:String◁Variable◁τ▷▷ and τ::
  - “"⦅" **f** "▪[" **x** ":" τ_1 "]:" τ_2 "≡" **d** "in" **y** "⦆"”:String◁Expression◁τ_3▷▷ where **f**:String◁Variable◁$\tau_2^{\tau_1}$▷▷ in **d** and **y**, **x**:String◁Variable◁τ_1▷▷ in **d**, **d**:String◁Expression◁τ_2▷▷, **y**:String◁Expression◁τ_3▷▷, and τ_1,τ_2,τ_3:: [120]
  - “"⦅" **x** ":" τ_1 "≡" **d** "in" **y** "⦆"”:String◁Expression◁τ_2▷▷ where **x**:String◁Variable◁τ_1▷▷ in **d** and **d**:String◁Expression◁τ_1▷▷, **y**:String◁Expression◁τ_2▷▷, and τ_1,τ_2:: [121]
  - “"⦅" e_1 "⦶" e_2 "⦆"”:String◁Expression◁τ_1⦶τ_2▷▷,
    “"⦅" "[" e_1 "," e_2"]" "⦆"”:String◁Expression◁[τ_1,τ_2]▷▷,
    “"⦅" "[" e_1 "]→" e_2 "⦆"”:String◁Expression◁[τ_1]→τ_2▷▷, and
    “"⦅" $e_2^{e_1}$ "⦆"”:String◁Expression◁$\tau_2^{\tau_1}$▷▷ where
    e_1:String◁Expression◁τ_1▷▷, e_2:String◁Expression◁τ_2▷▷, and τ_1,τ_2::
    “"⦅" e_1 "⍰" "True⸰" e_2 "," "**False**⸰" e_3 "⦆"”:String◁Expression◁τ▷▷[i] where
    e_1:String◁Expression◁Boolean▷▷, e_2,e_3:String◁Expression◁τ▷▷
    and τ::
    “"⦅" "λ[" **x** ":" τ_1 "]:" τ_2 **e** "⦆"”:String◁Expression◁[τ_1]→τ_2▷▷ where
    e:String◁Expression◁τ_2▷▷, **x**:String◁Variable◁τ_1▷▷ in **e**, and τ_1,τ_2::
    “"⦅" **e** "▪[" **x** "]" "⦆"”:Expression◁τ_2▷ where **e**:Expression◁[τ_1]→τ_2▷,
    x:Expression◁τ_1▷, τ_1:: and τ_2::
  - String◁Sentence◁anOrder▷▷⊑String◁Term◁Sentence◁anOrder▷▷▷ and
    String◁Sentence◁anOrder▷▷⊑String◁Expression◁Sentence◁anOrder▷▷▷
    where anOrder:ℕ₊
    ⌊**e**⌋:Expression◁τ▷, where **e**:String◁Expression◁τ▷▷ and τ::

---

[i] “*if* e_1 *then* e_2 *else* e_3”

Index

⦅, 55
⦆, 55
⌊, 42, 52, 53, 54, 55, 56, 57, 58
⦶, 52, 53, 54, 56
∶, 52, 55, 56
∷, 52, 55
∷, 𝕆, 52, 53
═, 52, 55
∀, 52, 55
∃, 52, 55
∈, 49
■, 54
≐, 46, 48
≡, 53, 54, 57, 58
⊑, 52, 54, 55, 56
⊢, 52, 55
⊨, 11, 47, 50, 51, 52, 55
⋃, 49
⍰, 52, 53, 54, 55, 56, 57, 58
→, 52, 53, 54
⇨, 52, 55
⇔, 52, 55
⸰, 52, 53, 54, 55, 56, 57, 58
Ackermann procedure, 30
Act, 52, 53
Actor
  Unbounded Nondeterminacy, 20
Actors
  Computational Adequacy, 20
Adequate$_1$, 34, 35
argumentation, 3
Barwise, J., 37
Beeson, M., 23
Berry Paradox, 42
Boolean, 5, 52, 53
Boom, H., 23
categorical, 10, 41
categoricity
  natural numbers, 12
  ordinals, 47
  reals, 46
Chaitin, G., 31
choice
  type, 48
Church, A., 1, 21, 35
Church's Paradox, 23, 34
Classical Direct Logic, 2
Cohen, P., 28
Computational Adequacy
  Actors, 20

computational inadequacy
  nondeterministic lambda
    calculus, 43
computational undecidability
  proof correctness, 22
Consistency of Mathematics, 7
Constant, 53
Coq, 21
Coquand, T., 21
Curry, H., 31
Dawson, J., 22
Dedekind, R., 1
Direct Logic
  sentence, 55, 56
Direct Logic, 1
  expression, 54
  proposition, 52
  term, 53
Direct Logic
  expression, 57
Direct Logic
  expression, 58
expression
  Direct Logic, 54, 57, 58
Expression, 54, 57, 58
False, 5
First-Order Thesis, 37
Foundation
  sets, 49
Galbraith, J. K., 3
Gödel
  validity of incompleteness
    arguments, 29
Gödel, K., 28
higher order logic, 10
Huet, G., 21
I'mUnprovable, 21, 29
Inconsistency Robustness, 1
indiscernibility, 11, 17, 45, 47, 50
inexpressibility, 22
inexpressible proofs, 22
Isabelle, 21
Kleene, S., 21
Knies, 41
Kuhn, T., 3
Lakatos, I., 28
Lam$_\tau$, 43
lambda calculus
  computational inadequacy, 43
Lambda Calculus
  Unbounded Nondeterminacy, 20
Lambda Induction, 44
Lambda τ, 44
Law, J., 28
Löb, M., 31
Logic Program
  nondeterministic, 40
  pure, 19, 20
Logic Programs
  Unbounded Nondeterminacy, 20
Monk, R., 29
ℕ, 38, 52, 53
Nat, 10, 11, 15, 36
  Proof verification, 16
Nat$_1$, 32
Natural Deduction, 8
nondeterministic lambda calculus,
  19, 20
Nondeterministic Lambda Calculus,
  43
nondeterministic Logic Program, 40
nondeterministic Turing Machine,
  40
Ord, 39, 47
Paulson, L., 21
Planck, M., 3
Plotkin, G., 23
Powerset

sets, 49
Principle of Excluded Middle, 3
proof correctness
computationally undecidable, 22
Proof verification
Nat, 16
proofs inexpressiblity, 22
proposition
Direct Logic, 52
Proposition, 5, 52
Pythagoreans, 1
RealClosedFields, 46
reals
categoricity, 46
Reals, 46
Replacement
sets, 49
Rosser, J. B., 21
Russell, B., 1, 21
sentence
Direct Logic, 55, 56
Sentence, 52, 55, 56
Set, 52
sets
Axiom of Foundation, 49
Axiom of Replacement, 49
axiom of Union, 49
powerset, 49
Transitivity of ∈, 49
ZFC, 49
Sets◁τ▷, 49
Sets$_{\kappa}$, 48
String, 52
Tarski, A., 5
term
Direct Logic, 53
Term, 53
Terminator, 41
Transitivity of ∈
sets, 49
True, 5
truth, 3
Turing Machine
nondeterministic, 40
Turing, A., 31
Type, 52
Type◁τ▷, 52
types
choice, 48
Unbounded Nondeterminacy, 20
Unbounded Nondeterminism, 39
Union
sets, 49
universe
sets, 49
von Neumann, J., 28
Weakest Preconditions, 16
Wittgenstein, L., 1, 29
Woods, J., 23
Zermelo, E., 1, 37
ZFC
sets, 49
λ, 53, 54

End Notes

[1] Performance of computer information systems is measured in consumption of processing cycles and storage space as well as latency for response. Pervasive inconsistency for information systems means that there are numerous inconsistencies that cannot be readily found and that many of the ones that are found cannot be easily removed.

[2] [White 1956, Wilder 1968, Rosental 2008]

[3] In other words, the paradox that concerned [Church 1934] (because it could mean the demise of formal mathematical logic) has been transformed into fundamental theorem of foundations!

[4] Which is not the same as proving the much ***stronger*** proposition that Mathematics is inferentially consistent, i.e., that there is no proof of contradiction from the axioms and inference rules of Direct Logic.

[5] Mathematical foundations of Computer Science must be general, rigorous, realistic, and as simple as possible. There are a large number of highly technical aspects with complicated interdependencies and trade-offs. Foundations will be used by humans and computer systems. Contradictions in the mathematical foundations of Computer Science cannot be allowed and if found must be repaired.

Classical mathematics is the subject of this article. In a more general context:

- Inconsistency Robust Direct Logic is for pervasively inconsistent theories of practice, e.g., theories for climate modeling and for modeling the human brain.
- Classical Direct Logic can be freely used in theories of Inconsistency Robust Direct Logic. See [Hewitt 2010] for discussion of Inconsistency Robust Direct Logic. Classical Direct Logic for mathematics used in inconsistency robust theories.

[6] cf. [Rosental 2008]

[7] According to [Concoran 2001]:

> "after first-order logic had been isolated and had been assimilated by the logic community, people emerged who could not accept the idea that first-order logic was not comprehensive. These logicians can be viewed not as conservatives who want to reinstate an outmoded tradition but rather as radicals who want to overthrow an established tradition [of Dedekind, etc.]."

[8] for discussion see [Hewitt 2010]

[9] in an unlawful way (Einstein, a member of the editorial board, refused to support Hilbert's action)

[10] Hilbert letter to Brouwer, October 1928

[11] Gödel said "Has Wittgenstein lost his mind?"

[12] For example:

From: Harvey Friedman
Sent: Wednesday, April 20, 2016 10:53
To: Carl Hewitt
Cc: Martin Davis @cs.nyu; Dana Scott @cmu; Eric Astor @uconn; Mario Carneiro @osu; Dave Mcallester @ttic; Joe Shipman
Subject: Re: Parameterized types in the foundations of mathematics

Not if I have anything to say about it!

Harvey

On Wed, Apr 20, 2016 at 11:25 AM, Carl Hewitt wrote:

> Hi Martin,
>
> Please post the message below to FOM [Foundations of Mathematics forum].
>
> Thanks!
>
> Carl
>
> According to Harvey Friedman on the FOM Wiki: "I have not yet seen any seriously alternative foundational setup that tries to be better than ZFC in this [categoricity of models] and other respects that isn't far far worse than ZFC in other even more important respects."
>
> Of course, ZFC is a trivial consequence of parameterized types with the following definition for set of type τ:
>
> $\text{Set}\triangleleft\tau\triangleright \equiv \text{Boolean}^{\tau}$

>> Also of course, classical mathematics can be naturally formalized using parameterized types. For example, see "Inconsistency Robustness in Foundations: Mathematics self proves its own Consistency and Other Matters" in HAL Archives.
>
> Regards,
> Carl

[13] [Nielsen 2014]

[14] By the *Computational Representation Theorem* [Hewitt 2006], which can define all the possible executions of a procedure.

[15] highlighted below

[16] Again, *Mathematics* here means the common foundation of all classical mathematical theories from Euclid to the mathematics used to prove Fermat's Last [McLarty 2010].

[17] Note that the results in [Löb 1955] **do not apply** because propositions in Mathematics are strongly typed and consequently the fixed point used to establish his result does not exist. See discussion of Löb's Paradox in this article.

[18] Note that the results in [Gödel 1931] **do not apply** because propositions in Mathematics are strongly typed and consequently the fixed point used construct Gödel's proposition *I'mUnprovable* **does not exist** in Mathematics. See the critique of Gödel's results in this article.

[19] As shown above, there is a simple proof in Classical Direct Logic that Mathematics ($\vdash$) is formally consistent. If Classical Direct Logic has a bug, then there might also be a proof that Mathematics is inconsistent. Of course, if a such a bug is found, then it must be repaired. The Classical Direct Logic proof that Mathematics ($\vdash$) is formally consistent is very robust. One explanation is that formal consistency is built in to the very architecture of Mathematics because it was designed to be consistent. Consequently, it is not absurd that there is a simple proof of the formal consistency of Mathematics ($\vdash$) that does not use all of the machinery of Classical Direct Logic.

The usefulness of Classical Direct Logic depends crucially on the much ***stronger*** proposition that Mathematics is inferentially consistent, i.e., that there is no proof of contradiction from the ***sentences*** for the axioms using the inference rules of Direct Logic. Good evidence for the inferential consistency of Mathematics comes from the way that Classical Direct Logic avoids the known paradoxes. Humans have spent millennia devising paradoxes.

In reaction to paradoxes, philosophers developed the dogma of the necessity of strict separation of "object theories" (theories about basic mathematical entities such as numbers) and "meta theories" (theories about theories). This linguistic separation can be very awkward in Computer Science. Consequently, Direct Logic does not have the separation in order that some propositions can be more "directly" expressed. For example, Direct Logic can use $\vdash\vdash\Psi$ to express that it is provable that Ψ is provable in Mathematics. It turns out in Classical Direct Logic that $\vdash\vdash\Psi$ holds if and only if $\vdash\Psi$ holds. By using such expressions, Direct Logic contravenes the philosophical dogma that the proposition $\vdash\vdash\Psi$ must be expressed using Gödel numbers.

[20] [Gödel 1931] based incompleteness results on the thesis that Mathematics necessarily has the proposition *I'mUnprovable* using what was later called the "Diagonal Lemma" [Carnap 1934], which is equivalent to the Y untyped fixed point operator on propositions. **Using strong parameterized types, it is impossible to construct *I'mUnprovable* because the Y untyped fixed point operator does not exist for strongly typed propositions.** In this way, formal consistency of Mathematics is preserved without giving up power because there do not seem to be any practical uses for *I'mUnprovable* in Computer Science.

A definition of NotProvable could be attempted as follows:

$$\mathit{NotProvable}[p] \equiv\ \nvdash p$$

With strong types, the attempted definition becomes:

$$\text{NotProvable}\lhd n{:}\mathbb{N}_+\rhd[p{:}\text{Proposition}\lhd n\rhd]{:}\text{Proposition}\lhd n+1\rhd \equiv\ \nvdash p$$

Consequently, there is no fixed point *I'mUnprovable* for the procedure NotProvable$\lhd n{:}\mathbb{N}_+\rhd$ such that the following holds:

$$\text{NotProvable}\lhd n{:}\mathbb{N}_+\rhd[\mathit{I'mUnprovable}] \Leftrightarrow \mathit{I'mUnprovable}$$

Thus Gödel's *I'mUnprovable* does not exist in Strongly Typed Mathematics.

In arguing against Wittgenstein's criticism, Gödel maintained that his results on *I'mUnprovable* followed from properties of $\mathbb{N}$ using Gödel numbers for strings that are well-formed. The procedure NotProvable could be attempted for strings as follows: *NotProvable[s] ≡ "⊬ s"* With strong types, the attempted definition becomes:

$$\text{NotProvable}\lhd n{:}\mathbb{N}_+\rhd[s{:}\text{String}\lhd\text{Proposition}\lhd n\rhd\rhd]{:}\text{String}\lhd\text{Proposition}\lhd n+1\rhd\rhd \equiv \text{``"}\nvdash\text{" s''}$$

Consequently, there is no fixed point *I'mUnprovableString* for the procedure NotProvable$\lhd n{:}\mathbb{N}_+\rhd$ such that the following holds (where $\lfloor s\rfloor$ is the proposition for well-formed string s):

$$\lfloor\text{NotProvable}\lhd n{:}\mathbb{N}_+\rhd\,[\mathit{I'mUnprovableString}]\rfloor \Leftrightarrow \lfloor\mathit{I'mUnprovableString}\rfloor$$

Thus Gödel's *I'mUnprovableString* does not exist in Strongly Typed Mathematics.

Furthermore, Strong Types thwart the known paradoxes while at the same time facilitating proof of new theorems, such as categoricity of the set theory.

[21] $\text{Total}[f] \Leftrightarrow \forall[i{:}\mathbb{N}]\ \exists[j{:}\mathbb{N}]\ f_{\blacksquare}[i] = j$

[22] ProvedTotalsEnumerator▪[i:ℕ]:([ℕ]→ℕ)) ≡ Next▪[i, 0, 0]
Next▪[i:ℕ, totalsIterator:ℕ, theoremsIterator:ℕ]:([ℕ]→ℕ)) ≡
  TheoremsEnumerator[theoremsIterator] ⍰
    Total[f] ⦂ // TheoremsEnumerator[theoremsIterator]=Total[f]
      totalsIterator=i ⍰
        True ⦂ f,
        False⦂ Next▪[i, totalsIterator+1, theoremsIterator+1]
    else ⦂ Next▪[i, totalsIterator, theoremsIterator+1]

Theorem ⊢Total[ProvedTotalsEnumerator]

Proof: ProvedTotalsEnumerator always converges because.

⊢∀[i:ℕ] ∃[j:ℕ, g:([ℕ]→ℕ)] j>i ∧ TheoremsEnumerator▪[j]=Total[g]

[23] [Isaacson 2007]

[24] A theory is defined by a set of propositions in Direct Logic that are taken to be axioms of the theory.

[25] The whole induction axiom is of type Proposition⊲2⊳. However, ∀[i:ℕ] P[i] within the induction axiom is of type Proposition⊲1⊳. Quine famously criticized 2^{nd} order theory as nothing more than "*set theory in sheep's clothing*" [Quine 1970, pg. 66]. However, the induction axiom is a more natural axiomatization of the Natural Numbers than the 1^{st} order induction schema which provides an infinitely large number of axioms.

Theorem. ∀[X:$\text{Boolean}^{ℕ}$] (0∈X ∧ ∀[i:ℕ] i∈X⇨i+1∈X) ⇨ ∀[i:ℕ] i∈X

Proof: Suppose X:$\text{Boolean}^{ℕ}$. P[i:ℕ]:Proposition⊲1⊳ ≡ i∈X. The theorem follows immediately.

Theorem. Set theory version of the Natural Number induction axiom implies propositional version.

Proof: Suppose ∀[X:$\text{Boolean⊲1⊳}^{ℕ}$] (0∈X∧∀[i:ℕ] i∈X⇨i+1∈X) ⇨ ∀[i:ℕ] i∈X

Further suppose P:$\text{Proposition⊲1⊳}^{ℕ}$. Define X:$\text{Boolean}^{ℕ}$ ≡ {i:ℕ | P[i]}.

It follows that (P[0] ∧ ∀[i:ℕ] P[i]⇨P[i+1]) ⇨ ∀[i:ℕ] P[i].

[26] highlighted below

[27] Prove by induction that $\forall[i:\mathbb{N}]\ \forall[j:\mathbb{N}]\ \ i=j \Leftrightarrow \forall[P:\mathbb{Proposition}\triangleleft 1\triangleright^{\mathbb{N}}]\ \ P[i]\Leftrightarrow P[j]$
Suppose the following: $(\forall[i,j<k:\mathbb{N}]\ \ i=j \Leftrightarrow \forall[P:\mathbb{Proposition}\triangleleft 1\triangleright^{\mathbb{N}}]\ \ P[i]\Leftrightarrow P[j])$
Show $\forall[i,j\leq k+1:\mathbb{N}]\ \ i=j \Leftrightarrow \forall[P:\mathbb{Proposition}\triangleleft 1\triangleright^{\mathbb{N}}]\ \ P[i]\Leftrightarrow P[j]$ Suppose $i,j\leq k+1:\mathbb{N}$ and $P:\mathbb{Proposition}\triangleleft 1\triangleright^{\mathbb{N}}$ such that $P[i]\Leftrightarrow P[j]$. To show i=j, consider the following cases:

- $i,j<k$ then $i=j$ by inductive hypothesis
- $i=k+1$ and $j<k$ then define $Q[m:\mathbb{N}] \equiv (m=k+1)$. Applying the inductive hypothesis for $Q[j]$, $i=k+1 \Leftrightarrow j=k+1\ \ \therefore i=j$
- $j=k+1$ and $i<k$ then define $Q[m:\mathbb{N}] \equiv (m=k+1)$. Applying the inductive hypothesis for $Q[i]$, $j=k+1 \Leftrightarrow i=k+1\ \ \therefore i=j$
- $i=j=k+1$ then $i=j$

[28] [Dedekind 1888] According to [Isaacson 2007]:

> "Second-order quantification is significant for philosophy of mathematics since it is the means by which mathematical structures may be characterized. But it is also significant for mathematics itself. It is the means by which the significant distinction can be made between the independence of Euclid's Fifth postulate from the other postulates of geometry and the independence of Cantor's Continuum hypothesis [conjecture] from the axioms of set theory. The independence of the Fifth postulate rejects the fact, which can be expressed and established using second-order logic, that there are different geometries, in one of which the Fifth postulate holds (is true), in others of which it is false."

[29] cf. [Genesereth and Kao 2015; Zohar 2017]

[30] cf. [Zermelo 1932] pp. 6-7.

[31] Examples:

- $\forall[P:\mathbb{Proposition}\triangleleft 1\triangleright^{\mathbb{N}}]\ \ (\vDash_{\mathbb{N}} \forall[i:\mathbb{N}]\ P[i]) \Rightarrow\ \vdash_{\mathit{Nat}} \forall[i:\mathbb{N}]\ P[i]$

  Suppose in *Nat*, $P:\mathbb{Proposition}\triangleleft 1\triangleright^{\mathbb{N}}$ and $\vDash_{\mathbb{N}} \forall[i:\mathbb{N}]\ P[i]$. Further suppose to obtain a contradiction that $\neg\forall[i:\mathbb{N}]\ P[i]$. Therefore $\exists[i:\mathbb{N}]\ \neg P[i]$ and by Existential Elimination $\neg P[i_0]$ where $i_0:\mathbb{N}$, which contradicts $\vDash_{\mathbb{N}} P[i_0]$, from the hypothesis of the theorem. Therefore $\vdash_{\mathit{Nat}} \forall[i:\mathbb{N}]\ P[i]$ using proof by contradiction in *Nat*.

- $\forall[P:\mathbb{Proposition}\triangleleft 1\triangleright^{\mathbb{N}}]\ (\vDash_{\mathbb{N}} \exists[i:\mathbb{N}]\ P[i]) \Rightarrow\ \vdash_{\mathit{Nat}} \exists[i:\mathbb{N}]\ P[i]$

  Suppose in *Nat*, $P:\mathbb{Proposition}\triangleleft 1\triangleright^{\mathbb{N}}$ and $\vDash_{\mathbb{N}} \exists[i:\mathbb{N}]\ P[i]$. Further suppose to obtain a contradiction that $\neg\exists[i:\mathbb{N}]\ P[i]$ and therefore $\forall[i:\mathbb{N}]\ \neg P[i]$. However, $\vDash_{\mathbb{N}} P[i_0]$ where $i_0:\mathbb{N}$, which contradicts $\neg P[i_0]$. Therefore $\vdash_{\mathit{Nat}} \exists[i:\mathbb{N}]\ P[i]$ using proof by contradiction in *Nat*.

[32] often misleading called ω-consistency [Gödel 1931]

[33] often misleading called ω-consistency [Gödel 1931]

[34] **Proof**: Suppose to obtain a contradiction that ProvablyComputableR is computationally enumerable by

ProvablyComputableEnumerator:([ℕ]→ProvablyComputableR)

Diagonal▪[i:ℕ]:ℕ ≡ 1+(ProvablyComputableEnumerator▪[i])▪[i]

Diagonal is not in the range of ProvablyComputableEnumerator, which is a contradiction. QED.

The above proof builds on the argument in the following theorem:

Theorem [Cantor 1891]. $\mathbb{N}^{\mathbb{N}}$ is not countable.

Proof. For proof by contradiction, suppose that is $\mathbb{N}^{\mathbb{N}}$ countable by a procedure FunctionEnumerator:$\mathbb{N}^{\mathbb{N}^{\mathbb{N}}}$.

Diagonal[n] ≡ 1+(FunctionEnumerator [n])[n]

Diagonal is not in the range of FunctionEnumerator, which is a contradiction. QED.

[35] This argument appeared in [Church 1934] expressing concern that the argument meant that there is "*no sound basis for supposing that there is such a thing as logic.*"

[36] ProvedTotalsEnumerator▪[i:ℕ]:([ℕ]→ℕ)) ≡ Next▪[i, 0, 0]

Next▪[i:ℕ, totalsIterator:ℕ, theoremsIterator:ℕ]:([ℕ]→ℕ)) ≡
  TheoremsEnumerator[theoremsIterator] ?
    Total[f] ⦂   // TheoremsEnumerator[theoremsIterator]=Total[f]
      totalsIterator=i ?
        True⦂ f,
        False⦂ Next▪[i, totalsIterator+1, theoremsIterator+1]
    else ⦂ Next▪[i, totalsIterator, theoremsIterator+1]

Theorem $\vdash_{\mathcal{Nat}}$Total[ProvedTotalsEnumerator]

Proof: ProvedTotalsEnumerator always converges because.

$\vdash_{\mathcal{Nat}}$∀[i:ℕ] ∃[j:ℕ,g:([ℕ]→ℕ)] j>i ∧ TheoremsEnumerator▪[j]=Total[g]

[37] The theorems themselves can be represented as character strings because totality can be expressed as the abstraction of a character string and each procedure can be represented as the abstraction of a character string.

[38] Theorem. There are uncountably many countable ordinals (order types). Totality proofs have countable ordinals of arbitrarily high degree.

[39] of type [Com]→

Outcome[created= FiniteSet◁Actor▷, // **new** Actors
sent= FiniteSet◁Com▷, // **new** Communications
next= Behavior]

[40] Consequently, there can cannot be any escape hatch into an unformalized "meta-theory."

[41] The claim also relied on Gödel's proposition *I'mUnprovable.*

[42] Fixed points exist for types other than propositions.

[43] emphasis in original

[44] [Gödel 1931] was accepted doctrine by mainstream logicians for over eight decades.

[45] According to Solomon Feferman, Gödel was "the most important logician of the 20th century" and according to John Von Neumann he was "the greatest logician since Aristotle." [Feferman 1986, pg. 1 and 8]

[46] [Feferman 1986, pg. 1 and 8]

[47] Wittgenstein in 1937 published in Wittgenstein 1956, p. 50e and p. 51e]

[48] Wittgenstein was granting the supposition that [Gödel 1931] had proved inferential undecidability (sometimes called "incompleteness") of Russell's system, that is., $\vdash\nvdash P$. However, inferential undecidability is easy to prove using the proposition P *where* $P\Leftrightarrow\nvdash P$:

Proof. Suppose to obtain a contradiction that $\vdash P$. Both of the following can be inferred:

1) $\vdash\nvdash P$ from the hypothesis because $P\Leftrightarrow\nvdash P$
2) $\vdash\vdash P$ from the hypothesis by Adequacy.

But 1) and 2) are a contradiction. Consequently, $\vdash\nvdash P$ follows from proof by contradiction.

[49] [Wang 1972]

[50] The Liar Paradox [Eubulides of Miletus] is an example of using untyped propositions to derive an inconsistency:

F◁n▷[p:Proposition◁n▷]:Proposition◁n+1▷ ≡ ¬p

// above definition has no fixed point because ¬p has
// order greater than p

The following argument derives a contradiction assuming the existence of a fixed point for F:

1) *I'mFalse* ⇔ ¬*I'mFalse* // *nonexistent* fixed point of F
2) ¬***I'mFalse*** // proof by contradiction from 1)
3) ***I'mFalse*** // from 1) and 2)

[51] [Church 1935] correctly proved computational undecidability without using Gödel's *I'mUnprovable*. The Church theorem and its proof are very robust.

[52] in accord with the opinion of a large fraction of contemporary philosophers of logic

[53] In other words, the paradox that concerned [Church 1934] (because it could mean the demise of formal mathematical logic) has been transformed into fundamental theorem of foundations!

[54] Which is not the same as proving the much ***stronger*** proposition that Mathematics is inferentially consistent, i.e., that there is no proof of contradiction from the axioms and inference rules of Direct Logic.

[55] **Theorem**: $\vdash_{Nat} \forall[P{:}\mathbb{String}\triangleleft\mathbb{Proposition}\triangleleft 1\triangleright^{\mathbb{N}}\triangleright]$

$$(\lfloor P\rfloor[0] \wedge \forall[i{:}\mathbb{N}]\ \lfloor P\rfloor[i] \Rightarrow \lfloor P\rfloor[i+1]) \Rightarrow \forall[i{:}\mathbb{N}]\ \lfloor P\rfloor[i]$$

[56] In 1666, England's House of Commons introduced a bill against atheism and blasphemy, singling out Hobbes' Leviathan. Oxford university condemned and burnt Leviathan four years after the death of Hobbes in 1679.

[57] ContinuumForReals is defined as follows:

$$\text{ContinuumForReals} \Leftrightarrow\ \nexists[S{:}\mathbb{Boolean}^{\mathbb{N}}]\ \mathbb{N} \prec S \prec \mathbb{Boolean}^{\mathbb{N}}$$

ContinuumForReals has been proved for well-behaved subsets of the reals, such as Borel sets as follows:

$$\text{ContinuumForBorelSets} \Leftrightarrow\ \nexists[S{:}\mathbb{BorelSet}]\ \mathbb{N} \prec S \prec \mathbb{Boolean}^{\mathbb{N}}$$

where a $\mathbb{BorelSet}$ is formed from the countable union, countable intersection, and relative complement of open sets

That ContinuumForReals is an open problem is not so important for Computer Science because for ContinuumForComputableReals is immediate because the computable real numbers are enumerable.

For less well behaved subset of $\mathbb{R}$, ContinuumForReals remains an open problem.

Note that it is important not to confuse ContinuumForReals with ContinuumForRelational1stOrderZFC. ***Relational1stOrderZFC*** has countably many 1st order propositions as axioms. [Cohen 1963] proved the following theorem which is much weaker than ContinuumForReals because sets in the models of ***Relational1stOrderZFC*** do **not** include all of $\mathbb{Proposition}\triangleleft 1\triangleright^{\mathbb{N}}$ and the theory ***Relational1stOrderZFC*** is much weaker than the theory $Sets_{\mathbb{N}}$:

- $\nvdash_{\mathit{Relational1stOrderZFC}}$ ContinuumForRelational1stOrderZFC
- $\nvdash_{\mathit{Relational1stOrderZFC}}$ ¬ContinuumForRelational1stOrderZFC

Cohen's result above is very far from being able to decide the following:

$\vdash_{\boldsymbol{Sets}_{\mathbb{N}}}$ ContinuumForReals

[58] [Zermelo 1930, van Dalen 1998, Ebbinghaus 2007]

[59] 1st order theories fall prey to paradoxes like the Löwenheim–Skolem theorems (*e.g.* any 1st order theory of the real numbers has a countable model). Theorists have used the weakness of 1st order theory to prove results that do not hold in stronger formalisms such as Direct Logic [Cohen 1963, Barwise 1985].

[60] a restricted form of Model Checking in which the properties checked are limited to those that can be expressed in Linear-time Temporal Logic has been studied [Clarke, Emerson, Sifakis, *etc*. ACM 2007 Turing Award].

[61] *cf.* Plotkin [1976]

[62] up to a unique isomorphism

[63] Rejection of the 1st Order Thesis resolves the seeming paradox between the formal proof in this article that Mathematics formally proves its own formal consistency and the proof that 'Every "strong enough" formal system that admits a proof of its own consistency is actually inconsistent.' [Paulson 2014]. Although Mathematics is "strong enough," the absence of "self-referential" propositions (constructed using the Y untyped fixed point operator on propositions) blocks the proof of formal inconsistency to which Paulson referred.

[64] Note that the Berry paradox is blocked using strong types because BString is a string for a term of a proposition of anOrder+1 thereby preventing it from being substituted for a string for a term of a proposition of anOrder.

[65] using definition of BSet

[66] using definition of BExpression

[67] substituting BNumber for n

[68] **Subst** is the substitution procedure, which substitutes its third argument into the application of its first two arguments

[69] **Fix** implements recursion. It can be defined in Direct Logic as follows;

Fix◁τ_1,τ_1▷▪[F:Functional◁τ_1,τ_1▷]:([τ_1]→τ_2)

≡ **λ**[x:τ_1] (F▪[**Fix**◁τ_1,τ_1▷▪[F]])▪[x]

For example, suppose

F[g:[ℕ]→ℕ]:([ℕ]→ℕ) ≡ **λ**[i:ℕ] i=1 ? **True**⦂ 1 , **False**⦂ i∗g▪[i-1]

Therefore by the **Fix** axiom, **Fix**◁ℕ, ℕ▷▪[F]=F▪[**Fix**◁ℕ, ℕ▷▪[F]] and **Fix**◁ℕ, ℕ▷▪[F] = F▪[Factorial] = Factorial where

Factorial ≡ **λ**[i:ℕ] i=1 ? **True**⦂ 1 , **False**⦂ i∗Factorial▪[i-1]

[70] where F^1▪[x] ≡ F▪[x]

F^{n+1}▪[x] ≡ F^n▪[F▪[x]]

[71] Prove by induction on f,g:**Λ**◁τ▷

[72] Robinson [1961]

[73] [Dedekind 1888]

[74] The following can be used to characterize the real numbers (ℝ) up to a unique isomorphism:

∀[S:Set◁ℝ▷] S≠{ } ∧ Bounded[S] ⇨ HasLeastUpperBound[S]

where

Bounded[S:Set◁ℝ▷] ⇔ ∃[b:ℝ] UpperBound[b, S]

UpperBound[b:ℝ, S:Set◁ℝ▷] ⇔ b∈S ∧ ∀[x∈S] x≦b

HasLeastUpperBound[S:Set◁ℝ▷]] ⇔ ∃[b:ℝ] LeastUpperBound[b, S]

LeastUpperBound[b:ℝ, S:Set◁ℝ▷]

⇔ UpperBound[b,S] ∧ ∀[x∈S] UpperBound[x, S] ⇨ x≦b

[75] The theory of the ordinals *Ord* is axiomatised as follows:

- $0_{\mathbb{O}}:\mathbb{O}$
- Successor ordinals
  - $\forall[\alpha:\mathbb{O}]\ +_1[\alpha]:\mathbb{O} \wedge +_1[\alpha]>\alpha$
  - $\forall[\alpha:\mathbb{O}]\ \nexists[\beta:\mathbb{O}]\ \alpha<\beta<+_1[\alpha]$
- Replacement for ordinals:
  - $\forall[\alpha:\mathbb{O},f:\mathbb{O}^{\mathbb{O}}]\ \dot{\cup}_{\alpha}f:\mathbb{O}$
  - $\forall[\alpha,\beta:\mathbb{O},f:\mathbb{O}^{\mathbb{O}}]\ \beta\in\dot{\cup}_{\alpha}f \Leftrightarrow \exists[\delta<\alpha]\ \beta\leqq f[\delta]$
  - $\forall[\alpha,\beta:\mathbb{O},f:\mathbb{O}^{\mathbb{O}}]\ (\forall[\delta<\alpha]\ f[\delta]\leqq\beta) \Rightarrow \dot{\cup}_{\alpha}f\leqq\beta$
- Cardinal ordinals

  $\omega_0 = \mathbb{N}$

  $\forall[\alpha:\mathbb{O}]\ \alpha>0_{\mathbb{O}} \Rightarrow \omega_{\alpha}\doteq\mathbf{Boolean}^{\dot{\cup}_{\beta<\alpha}\omega_{\beta}}$

  $\forall[\alpha,\beta:\mathbb{O}]\ \beta\doteq\omega_{\alpha} \Rightarrow \omega_{\alpha}=\beta \vee \omega_{\alpha}\in\beta$

  where $\tau_1\doteq\tau_2 \Leftrightarrow \exists[f:{\tau_2}^{\tau_1}]\ 1to1onto\lhd\tau_1,\tau_2\rhd[f]$

  $1to1\lhd\tau_1,\tau_2\rhd[f:{\tau_2}^{\tau_1}] \Leftrightarrow \forall[x_1,x_2:\tau_1]\ f[x1]=f[x2] \Rightarrow x1=x2$

  $1to1onto\lhd\tau_1,\tau_2\rhd[f:{\tau_2}^{\tau_1}]$

  $\Leftrightarrow 1to1\lhd\tau_1,\tau_2\rhd[f:{\tau_2}^{\tau_1}] \wedge \forall[y:\tau_2]\ \exists[x:\tau_1]\ f[x]=y$
- Tansitivity of <

  $\forall[\alpha,\beta<\alpha,\delta<\beta:\mathbb{O}]\ \alpha<\delta$
- $\forall[\alpha,\beta:\mathbb{O}]\ \alpha<\beta \vee \alpha=\beta \vee \beta<\alpha$
- $\forall[\alpha,\beta:\mathbb{O}]\ \alpha<\beta \Rightarrow \neg\beta<\alpha$
- The following ordinal induction axiom holds:

  $\forall[P:\mathrm{Proposition}\lhd\mathrm{order}\rhd^{\mathbb{O}}]$

  $(\forall[\alpha:\mathbb{O}]\ \forall[\beta<\alpha:\mathbb{O}]\ P[\beta]\Rightarrow P[\alpha]) \Rightarrow \forall[\alpha:\mathbb{O}]\ P[\alpha]$

[76] For each type $\mathbb{X}$ that satisfies the theory $\mathcal{Ord}$ there is a unique isomorphism $I:\mathbb{X}^{\mathbb{O}}$ inductively defined as follows:

$$I[0_{\mathbb{O}}] \equiv 0_{\mathbb{X}}$$

$$\forall[\alpha:\mathbb{O}]\ I[+_1[\alpha]] \equiv +_1^{\mathbf{X}}[I[\alpha]]$$

$$\forall[\alpha:\mathbb{Limit}\triangleleft\mathbb{O}\triangleright]\ I[\alpha] \equiv y$$

$$\text{where } y:\mathbb{X} \wedge \forall[\beta<\alpha]\ y \leqq_{\mathbb{X}} I[\beta]$$

$$\wedge \forall[z:X]\ (\forall[\beta<\alpha]\ z \leqq_{\mathbb{X}} I[\beta]) \Rightarrow y \leqq_{\mathbb{X}} z$$

Using proofs by ordinal induction on $\mathbb{O}$ and $\mathbb{X}$, the following follow:

1. I is defined for every $\mathbb{O}$
2. I is one-to-one: $\forall[\alpha,\beta:\mathbb{O}]\ I[\alpha] = I[\beta] \Rightarrow \alpha=\beta$
3. The range of I is all of $\mathbb{X}$: $\forall[y:\mathbb{X}]\ \exists[\alpha:\mathbb{O}]\ I[\alpha] = y$
4. I is a homomorphism:
   - $I[0_{\mathbb{O}}] = 0_{\mathbb{X}}$
   - $\forall[\alpha:\mathbb{O}]\ I[+_1[\alpha]] = +_1^{\mathbf{X}}[I[\alpha]]$
   - $\forall[\alpha:\mathbb{Limit}\triangleleft\mathbb{O}\triangleright, f:\mathbb{O}^{\mathbb{O}}]\ I[\cup_\alpha f] = \cup_{f[\alpha]}^{\mathbf{X}} I \circ f \circ I^{-1}$
5. $I^{-1}:\mathbb{O}^{\mathbb{X}}$ is a homomorphism
6. I is the unique isomorphism: If $g:\mathbb{X}^{\mathbf{O}}$ is an isomorphism then g=I

[77] Prove by ordinal induction on $\alpha,\beta:\mathbb{O}$

[78] [Bourbaki 1972; Fantechi, et. al. 2005]

[79] This implies, for example, that no set is an element of itself.

[80] Proof: Suppose S:Sets◁τ▷ and therefore ∃[α:O] S:Sets$_{\alpha}$◁τ▷.

Proof by ordinal induction on

P[β:O]:Proposition◁1▷ ≡ ∀[X∈S] X:Sets$_{\beta}$◁τ▷

Assume: (∀[β<α:O] ∀[X∈S] X:Set$^{\beta}$◁τ▷) ⇨ ∀[X∈S] X:Sets$_{\alpha}$◁τ▷

Show: ∀[X∈S] X:Sets$_{\alpha}$◁τ▷

Assume: X∈S

Show X:Sets$_{\alpha}$◁τ▷

Proof by cases on α

1. X:Sets$_{0}$◁τ▷

   X:Boolean$^{\tau}$

2. ∀[α:O] Sets$_{\alpha}$◁τ▷ = Set◁Sets$_{\alpha-1}$◁τ▷▷

   X:Sets$_{\alpha-1}$◁τ▷ QED by induction hypothesis

3. ∀[α:Limit◁O▷] ∃[β<α,Y:Sets$_{\beta}$◁τ▷] X∈Y

   QED by induction hypothesis

[81] Proof: Suppose S:Sets◁τ▷ and therefore ∃[α:O] S:Sets$_{\alpha}$◁τ▷

S:Sets$_{\alpha}$◁τ▷

Show: Boolean^{s}:Sets◁τ▷

Boolean^{s}:Sets$_{\alpha+1}$◁τ▷ QED

[82] Proof by ordinal induction on

P[α:O]:Proposition◁1▷ ≡ ∀[S:Sets$_{\alpha}$◁τ▷] US:Sets◁τ▷

Assume: ∀[β<α:O] ∀[S:Sets$_{\beta}$◁τ▷] US:Sets◁τ▷

Show: ∀[S:Sets$_{\alpha}$◁τ▷] US:Sets◁τ▷

Assume: S:Sets$_{\alpha}$◁τ▷

Show: US:Sets◁τ▷

∀[X:Sets◁τ▷] X∈US ⇔ ∃[Y∈S] X∈Y

∀[X:Sets◁τ▷] X∈US ⇔ ∃[β<α:O,Y:Sets$_{\beta}$◁τ▷] X∈Y

∀[X:Sets◁τ▷] X∈US ⇒ X:Sets◁τ▷

QED by definition of Sets◁τ▷

[83] Suppose f:Sets◁τ▷$^{\text{Sets}◁τ▷}$ and S:Sets◁τ▷

Show Image◁τ▷[f, S]:Sets◁τ▷

Proof by ordinal induction on

P[α:O] ⇔ S:Set$_{\alpha}$◁τ▷ ⇒ Image◁τ▷[f, S]:Sets◁τ▷

Suppose ∀[β<α:O] S:Set$_{\beta}$◁τ▷ ⇒ Image◁τ▷[f, S]:Sets◁τ▷

Show S:Set$_{\alpha}$◁τ▷ ⇒ Image◁τ▷[f, S]:Sets◁τ▷

Suppose S:Set$_{\alpha}$◁τ▷

Show Image◁τ▷[f, S]:Sets◁τ▷

∀[y:Sets◁τ▷] y:Image◁τ▷[f, S] ⇔ ∃[x∈S] f[x]=y

Show ∀[y:Sets◁τ▷] y∈Image◁τ▷[f, S] ⇒ y:Sets◁τ▷

Suppose y:Sets◁τ▷ ∧ y∈Image◁τ▷[f, S]

Show y:Sets◁τ▷

∃[x∈S] f[x]=y because y∈Image◁τ▷[f, S]

∃[β<α:O] x:Sets$_{\beta}$◁τ▷ because x∈S and S:Sets$_{\alpha}$◁τ▷

Image◁τ▷[f, x]:Sets◁τ▷ by induction hypothesis

Show f[x]:Sets◁τ▷

Suppose z∈f[x]

Show z:Sets◁τ▷

z∈Sets◁τ▷ because z∈f[x] and Image◁τ▷[f, x]:Sets◁τ▷

f[x]:Sets◁τ▷

y:Sets◁τ▷ because f[x]=y

[84] [Mizar; Matuszewski1 and Rudnicki: 2005; Naumowicz and Artur Korniłowicz 2009; Naumowicz 2009]

[85] Note that this proof is fundamentally different from the categoricity proof in [Martin 2015].

[86] Prove by ordinal rank on s_1,s_2:Sets◁τ▷

[87] For every type there is a larger type, i.e.., ∀[τ_1::] ∃[τ_2::] $\tau_1 \subsetneq \tau_2$

[88] There is no universal type. Instead, Type is parameterized, e.g., Boolean:Type◁Boolean▷ and N:Type◁N▷

[89] **True≠False**, **True:**Boolean, and **False:**Boolean

∀[x:Boolean] x=**True** ∨ x=**False**

[90] O is the type of ordinals

91 $\mathbb{A}\mathrm{ct}$ is the type of Actors

92 expression of type **τ.** The following axiom holds:
$\forall[\boldsymbol{\tau}{::},t{:}\mathbb{T}\mathrm{erm}\triangleleft\boldsymbol{\tau}\triangleright]\ \lfloor t\rfloor{::}\boldsymbol{\tau}$

93 expression of type **τ.** The following axiom holds:
$\forall[\boldsymbol{\tau}{::},\ e{:}\mathbb{E}\mathrm{xpression}\triangleleft\boldsymbol{\tau}\triangleright]\ \lfloor \mathbf{e}\rfloor{::}\boldsymbol{\tau}$

94 $\boldsymbol{\Lambda}\triangleleft\tau\triangleright$ is the type of lambda procedures over τ

95 string of type **τ.** The following axiom holds:
$\forall[\boldsymbol{\tau}{::},s{:}\mathbb{S}\mathrm{tring}\triangleleft\boldsymbol{\tau}\triangleright]\ \lfloor \mathbf{s}\rfloor{::}\boldsymbol{\tau}$

96 type of type **τ**

97 $\forall[\boldsymbol{\tau}{::}]\ \ \boldsymbol{\tau}{:}\mathbb{T}\mathrm{ype}\triangleleft\boldsymbol{\tau}\triangleright$

98 $\mathbb{P}\mathrm{roposition}\triangleleft\mathrm{anOrder}\triangleright$ is the parametrized type consisting of type $\mathbb{P}\mathrm{roposition}$ parametrized by anOrder.

99 Discrimination of $\boldsymbol{\tau}_1$ and $\boldsymbol{\tau}_2$
For i=1,2

- If $x{:}\boldsymbol{\tau}_i$, then $((\boldsymbol{\tau}_1 \oplus \boldsymbol{\tau}_2)[x]){:}(\boldsymbol{\tau}_1 \oplus \boldsymbol{\tau}_2)$ and $x{=}((\boldsymbol{\tau}_1 \oplus \boldsymbol{\tau}_2)[x])\downarrow\boldsymbol{\tau}_i$.
- $\forall[z{:}\tau]\ z{:}\boldsymbol{\tau}_1 \oplus \boldsymbol{\tau}_2 \Leftrightarrow \exists[x{:}\boldsymbol{\tau}_i]\ z{=}(\boldsymbol{\tau}_1 \oplus \boldsymbol{\tau}_2)[x]$

100 type of 2-element list with first element of type $\boldsymbol{\tau}_1$ and with second element of type $\boldsymbol{\tau}_2$

101 Type of computable *nondeterministic* procedures from $\boldsymbol{\tau}_1$ into $\boldsymbol{\tau}_2$.
If $f{:}([\boldsymbol{\tau}_1]\rightarrow\boldsymbol{\tau}_2)$ and $x{:}\boldsymbol{\tau}_1$, then $f.[x]{:}\boldsymbol{\tau}_2$. The following holds:
$\forall[f{:}([\mathbb{N}]\rightarrow\mathbb{N})]\ \exists[\mathrm{aString}{:}(\mathbb{S}\mathrm{tring}\triangleleft\mathbb{E}\mathrm{xpression}\triangleleft[\mathbb{N}]\rightarrow\mathbb{N}\triangleright\triangleright)]\ f = \lfloor\lfloor \mathrm{aString}\rfloor\rfloor$
Furthermore, if $e{:}\mathbb{E}\mathrm{xpression}\triangleleft[\boldsymbol{\tau}_1]\rightarrow\boldsymbol{\tau}_2\triangleright$ with no free variables, then $\lfloor \mathbf{e}\rfloor{:}[\boldsymbol{\tau}_1]\rightarrow\boldsymbol{\tau}_2$).

102 Type of functions from $\boldsymbol{\tau}_1$ into $\boldsymbol{\tau}_2$. If $f{:}\tau_2{}^{\boldsymbol{\tau}_1}$ and $\mathbf{x}{:}\boldsymbol{\tau}_1$, then $\mathbf{f}[\mathbf{x}]{:}\boldsymbol{\tau}_2$.

103 $\forall[x{:}\boldsymbol{\tau}]\ \ x{:}\boldsymbol{\tau}\ni\!\!\!/\,\mathbf{P} \Leftrightarrow \mathbf{P}[x]$
For example,
$$\forall[\boldsymbol{\tau}{::},\ X{:}\mathrm{Boolean}^{\mathrm{Boolean}^{\tau}}]\ \bigcup X \equiv \boldsymbol{\tau}\ni\!\!\!/\,\lambda[y{:}\boldsymbol{\tau}]\ \exists[Z{:}\mathrm{Boolean}^{\tau}]\ Z{\in}X{\wedge}y{\in}Z$$

104 *if* **p** *then* $\boldsymbol{\Phi}_1$ *else* $\boldsymbol{\Phi}_2$

105 $\mathbf{x}_1$ is a subtype of $\mathbf{x}_2$, i.e., $\forall[\mathbf{x}{:}\boldsymbol{\tau}_1]\ \mathbf{x}{:}\boldsymbol{\tau}_2$

106 The proposition that **x** is a type

107 $\boldsymbol{\Phi}_1$, … and $\boldsymbol{\Phi}_{n\text{-}1}$ infer $\boldsymbol{\Phi}_n$

[108] The following hold:

- $(\models_{\mathbb{N}} \Phi\wedge\Psi) \Leftrightarrow (\models_{\mathbb{N}} \Phi) \wedge (\models_{\mathbb{N}} \Psi)$
- $(\models_{\mathbb{N}} \Phi\vee\Psi) \Leftrightarrow (\models_{\mathbb{N}} \Phi) \vee (\models_{\mathbb{N}} \Psi)$
- $(\models_{\mathbb{N}} \Phi\Rightarrow\Psi) \Leftrightarrow (\models_{\mathbb{N}} \Phi) \Rightarrow (\models_{\mathbb{N}} \Psi)$
- $(\models_{\mathbb{N}} \neg\Phi) \Leftrightarrow \neg\models_{\mathbb{N}} \Phi$
- $(\models_{\mathbb{N}} \forall[x{:}\mathbb{N}]\ p[x]) \Leftrightarrow \forall[x{:}\tau] \models_{\mathbb{N}} p[x]$
- $(\models_{\mathbb{N}} \exists[x{:}\mathbb{N}]\ p[x]) \Leftrightarrow \exists[x{:}\tau] \models_{\mathbb{N}} p[x]$

[109] mutually recursive definitions of functions $\mathbf{f}_{1\ \mathrm{to}\ n}$

[110] mutually recursive definitions of variables $x_{1\ \mathrm{to}\ n}$

[111] mutually recursive definitions of functions $\mathbf{f}_{1\ \mathrm{to}\ n}$

[112] mutually recursive definitions of variables $x_{1\ \mathrm{to}\ n}$

[113] The type of ⦅**p[x]**⦆ means that the Y fixed point construction cannot be used to construct sentences for "self-referential" propositions in Direct Logic.

[114] Also, as a special case, ⦅**p**[x]⦆:Sentence◁anOrder▷[114] where

p:Constant◁Expression◁Sentence◁anOrder▷$^{\tau}$▷▷

[115] Sentences are both Terms and Expressions in order to facilitate writing functions and procedures, respectively, over terms.

[116] The type of "**x**" means that the Y fixed point construction cannot be used to construct strings for "self-referential" propositions in Direct Logic.

[117] A Sentences is both a Term and an Expression in order to facilitate writing functions and procedures, respectively, over terms.

[118] mutually recursive definitions of functions $\mathbf{f}_{1\ \mathrm{to}\ n}$

[119] mutually recursive definitions of variables $x_{1\ \mathrm{to}\ n}$

[120] mutually recursive definitions of functions $\mathbf{f}_{1\ \mathrm{to}\ n}$

[121] mutually recursive definitions of variables $x_{1\ \mathrm{to}\ n}$