Modeling Formal Languages in Grammatical Framework

On the Grammar of Proof

Warrick Macmillan

7^{th} August 2021

Table of Contents

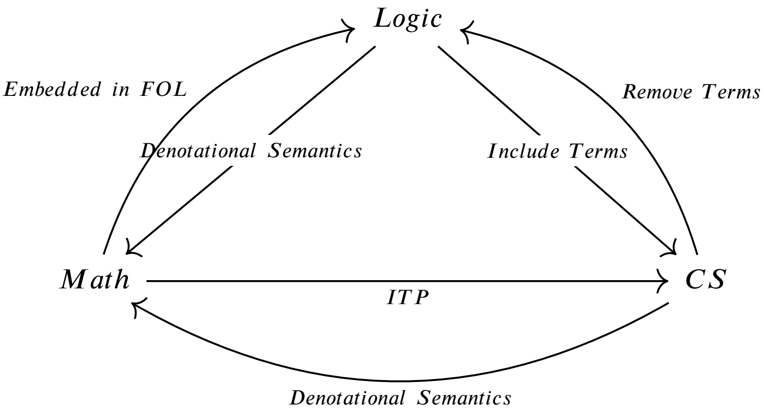
1. Explore abstract relationships between math, CS, Type Theory, and Linguistics
2. Practical and brief intro to MLTT and Agda
3. Grammars elaborating the abstractions above

Abstraction Ladders

$$
\begin{tikzcd}
Strings \arrow[d, "Lexical\ Analysis"] \arrow[dd, bend right=60, "Front\ End"'] \\
Lexemes \arrow[d, "Parsing"] \\
ASTs \arrow[d, "Type\ Checker"] \\
Typed\ ASTs \arrow[d, bend right, "Interpreter"'] \arrow[d, phantom, "\cdots"] \arrow[d, bend left, "Code\ Generator"] \\
Target\ Language
\end{tikzcd}
$$

$$
\begin{tikzcd}
Phonemes \arrow[d, dashed, "Morhphophonological"] \\
Morphemes \arrow[d, "Parse"] \\
\{\ Syntactic\ Representation\ \} \arrow[d, bend right, "Montague"'] \arrow[d, dashed, "\cdots"] \arrow[d, bend left, "Ranta"] \\
\{\ STLC,\ \ldots,\ DTT\ \} \arrow[d, dashed, "?"] \\
\{\ NearalEncoding,\ \ldots,\ I\ Language\ \} \arrow[d, dashed, "?"] \\
Phonemes
\end{tikzcd}
$$

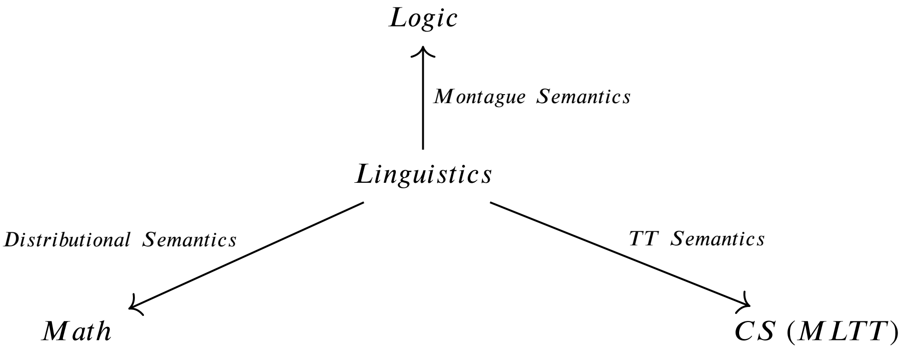
Computational Trinitarianism



Interpretation Language

Observation 1.1

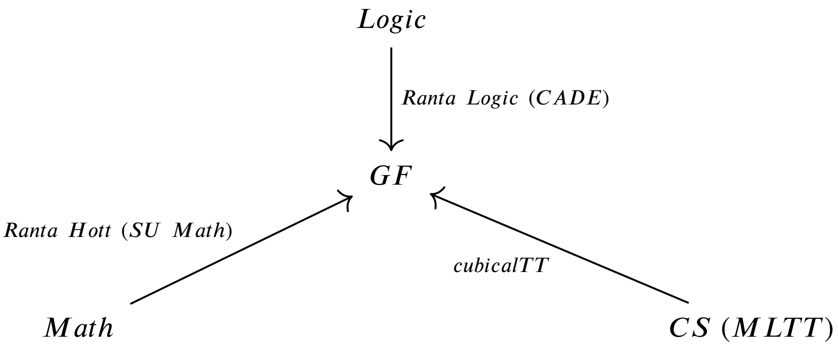
We acknowledge this is only semantic interpretations in these domains. One may decide on syntactic, pragmatic, or other ways in which to treat linguistics via these fields



Trinitarian Linguistics

$$
\begin{tikzcd}
 & Logic \arrow[d, "Embedding"] & \\
 & Linguistics & \\
Math \arrow[ur, "Language\ Of\ \ Mathematics"] & & CS\ (MLTT) \arrow[ul, "Meaning\ \ Explanations"']
\end{tikzcd}
$$

Trinitarian Grammars



Models of GF

$$
\begin{tikzcd}
 & Logic & \\
 & GF \arrow[u, "?"] \arrow[dl, "Theory\ of\ Operads"'] \arrow[dr, bend left, "Implementation\ of"] \arrow[dr, bend right, "Agda\ Embedding"'] & \\
Math & & CS\ (MLTT)
\end{tikzcd}
$$

Remarks

- Trinitarian doctrine is in the “formal” space
- Trinitarian + Linguistics is partially formal, and very underexplored
- Introduces many philosophical concerns, perhaps a rereading of Wittgenstein should take place in this context

- Frege : Formal Proof, Predicate Logic
- Russel : Type Theory to resolve his paradox
- Brouwer : Constructivism

Mathematical logic and the relation between logic and mathematics have been interpreted in at least three different ways:

i. mathematical logic as symbolic logic, or logic using mathematical symbolism;
ii. mathematical logic as foundations (or philosophy) of mathematics;
iii. mathematical logic as logic studied by mathematical methods, as a branch of mathematics.

We shall here mainly be interested in mathematical logic in the second sense. What we shall do is also mathematical logic in the first sense, but certainly not in the third.

(Per Martin-Löf, Padua Italy, June 1980)

Syntactic Comparisons

First Order Logic

- $\forall$
- $\exists$
- $\supset$
- $\wedge$
- $\vee$
- $\neg$
- $\top$
- $\bot$
- $=$

Dependent Type Theory

- Π
- Σ
- $\rightarrow$
- $\times$
- $+$
- $\neg$
- $\top$
- $\bot$
- $\equiv$

Sets

- $\mathbb{N}$
- $\mathbb{N} \times \mathbb{N}$
- $\mathbb{N} \to \mathbb{N}$
- $\{x|P(x)\}$
- $\emptyset$
- $?$
- $\cup$
- $?$

More Sets

- 1
- $(1, 0)$

Types

- Nat
- $Nat \times Nat$
- $Nat \to Nat$
- $\Sigma x : \_.P(x)$
- $\bot$
- $\top$
- $?$
- U_1

Programs

- $suc\ \ zero$
- $(suc\ \ zero, zero)$

Judgments

Type Theoretic Judgments

- T is a type
- T and T' are equal types
- t is a term of type T
- t and t' are equal terms of type T

Mathematical Judgments

- P is a proposition
- P is true

- Notice that judgmental equality is uniquely type theoretic
- Judgments in type theory are decidable
- Truth (inhabitation) is not decidable
- More exotic judgments are available in TT, i.e. P is possible.

Important Differences

- The rules of the types make explicit that they are not equivalent to those of classical FOL
- An existential assertion in type theory requires data
- Excluded middle and double negation are not admitted in MLTT
- To be *not unhappy* is clearly of a different meaning than to be *happy*.
- This makes our approach to general translation of non-constructive mathematics *impossible* (at least such that it type-checks)

- One doesn't define logics, type systems in mathematics (e.g. metamathemeatics)
- Encoding things like rational and real numbers in type theory are
- already, category theorists and set theorists are at odds, (small and large categories), higher categories, which skeletons of categories are canonical, etc. incredibly difficult
- Additionally, intensional type theory comes with two distinct notions of equality, judgmental/definitional/computational and propositional equality

Example Donkey Anaphora

Interpret the sentence “every man who owns a donkey beats it” in MLTT via the following judgment :

$$\Pi z : (\Sigma x : man.\ \Sigma y : donkey.\ owns(x, y)).\ beats(\pi_1 z, \pi_1(\pi_2 z))$$

We judge $\vdash man$:type and $\vdash donkey$:type. type really denotes a universe

What is Agda?

- Implementation of MLTT
- Logical Framework
- Interactive proof development environment
- Inductive Types, Modules, Pattern Matching, more

Twin Prime Conjecture in Agda

```
is-prime : ℕ → Set
is-prime n =
  (n ≥ 2) ×
  ((x y : ℕ) → x * y ≡ n → (x ≡ 1) ⊎ (x ≡ n))

twin-prime-conjecture : Set
twin-prime-conjecture = (n : ℕ) → Σ[ p ∈ ℕ ] (p ≥ n)
  × is-prime p
  × is-prime (p + 2)
```

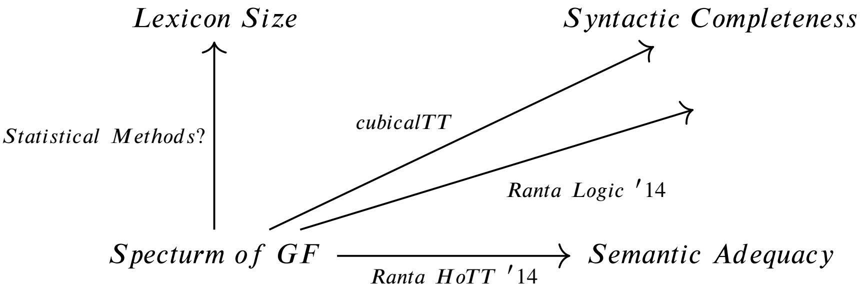
Mathematical Declarations

- Theorem
- Proof
- Lemma
- Axiom
- Definition
- Example

Boolean example
(type theoretic syntax)

What is a Proof?

Defintions
Syntctic Completeness Semantic Adequacy


Lexicon Size
Syntactic Completeness
Statistical Methods?
cubicalTT
Ranta Logic '14
Specturm of GF
Semantic Adequacy
Ranta HoTT '14