Dependent Type Theory

David Corfield

Evidence Seminar

20 November, 2020

As you may know, I've published a book on

- Modal homotopy type theory,

arguing that it would be a good idea for philosophy take it up as its basic formalism.

I'll spare you the 'modal' and 'homotopy' parts and focus today on

- Dependent type theory

I'll do this via a recent classification of logical systems that includes 'dependent typing' as one of its varieties.

This should allow for a useful contrast with first-order logic.

Motivation from mathematical reasoning

Bucket loads, but for now

Kevin Buzzard's talk – Is HoTT the way to do mathematics?

Kevin is a number theorist at Imperial College London who is looking to train his undergraduates to produce computer-checked proofs of mainstream theory (algebraic geometry, etc.) in the Lean theorem-prover.

Why Lean? Well, Lean is based on dependent type theory, and at (12:14) in the talk Kevin claims of ordinary mathematicians that

> *They use dependent types, even though they don't know they are using dependent types.*

My claim:

> *Mathematicians use dependent types because they are speakers of natural language. We all use dependent types.*

The case for the advantages of DTT over FOL can be made not only for advanced reasoning such as mathematics, but also for everyday reasoning.

In the Preface to his book, *Type-theoretic Grammar* (OUP, 1994), Aarne Ranta recounts how the idea of studying natural language in dependent type theory occurred to him in 1986:

> *In Stockholm, when I first discussed the project with Per Martin-Löf, he said that he had designed type theory for mathematics, and that natural language is something else. I said that similar work had been done within predicate calculus, which is just a part of type theory, to which he replied that he found it equally problematic. But his general attitude was far from discouraging: it was more that he was so serious about natural language and saw the problems of my enterprise more clearly than I, who had already assumed the point of view of logical semantics. His criticism was penetrating but patient, and he was generous in telling me about his own ideas. So we gradually developed a view that satisfied both of us, that formal grammar begins with what is well understood formally, and then tries to see how this formal structure is manifested in natural language, instead of starting with natural language in all it unlimitedness and trying to force it into some given formalism.*

A distant target

From the very beginning, from the first moment I may almost say, of my acquaintance with you, your manners impressing me with the fullest belief of your arrogance, your conceit, and your selfish disdain of the feelings of others, were such as to form the groundwork of disapprobation, on which succeeding events have built so immovable a dislike.

A proposition, so a type which ought to be formed by type-formation rules.

Something simpler for now

- Whenever a man's daughter speaks politely to him, he praises her for it.

- $Man(x)\&Person(y)\&Daughter(y,x)\&Event(z)\&Speaking(z)\&$ Agent(z) = y & Object(z) = x & Manner(z) = politely $\vdash$ $\exists p(Event(p)\&Praising(p)\&Agent(p) = x\&Object(p) = y\&$ Reason(p)= z)

- $x : Man, y : Daughter(x), z : PoliteSpeak(x, y) \vdash p : Praise(x, y, z)$

Then apply universal quantification/dependent product.

Of course, I've made the type-theoretic version look simpler by leaving some work to do:

- Whenever a man's daughter speaks politely to him, he praises her for it.
- $x : \mathit{Man}, y : \mathit{Daughter}(x), z : \mathit{PoliteSpeak}(x, y) \vdash p : \mathit{Praise}(x, y, z)$

I would have to have formed already the types $\mathit{Daughter}(x)$, $\mathit{PoliteSpeak}(x, y)$, etc.

Note that pronouns match types, and that p is a function.

Two limitations of (usual) first-order logic

- No typing used, so the user has to carve out kinds of entity from a single domain.
- Dependency is limited to predicates and relations.

The first may be rectified by *typed* first-order logic.

The second requires a major shift in logic.

Mike Shulman's classification

As detailed here, when specifying a theory in a logic there are three steps:

- Specify the judgment structure
- Specify the rules of the deductive structure
- Specify the generating types, terms, and axioms

Mike Shulman's classification

As detailed here, when specifying a theory in a logic there are three steps:

- Specify the judgment structure : 3-theory
- Specify the rules of the deductive structure : 2-theory
- Specify the generating types, terms, and axioms : 1-theory

Just to say 'theory' is enormously ambiguous.

(Numbering relates to the fact that models of an n-theory form an n-category.)

Judgment structure

The expressivity of our logic is determined by which judgments are allowed:

- $x : A \vdash b : B$
- $x : A, y : B, z : C \vdash d : D$
- $x : A, y : B, z : C, P(x), Q(y, z), R \vdash S(x, y, z)$
- $x : A, y : B(x), z : C(x, y) \vdash d : D(x, y, z)$
- variants of first three with multiple consequents

Propositional logic as a 2-theory

Sequent Calculus LJ

- $\dfrac{\Gamma \vdash G}{A,\Gamma \vdash G}(weak)$ $\dfrac{A,A,\Gamma \vdash G}{A,\Gamma \vdash G}(contr)$ $\dfrac{\Gamma \vdash A \quad A,\Delta \vdash G}{\Gamma,\Delta \vdash G}(cut)$
- $\dfrac{}{A \vdash A}(axiom)$ $\dfrac{}{\bot \vdash G}(exfalso)$
- $\dfrac{\Gamma \vdash A \quad B,\Gamma \vdash G}{A \to B,\Gamma \vdash G}(\to_L)$ $\dfrac{A,\Gamma \vdash B}{\Gamma \vdash A \to B}(\to_R)$
- $\dfrac{A,B,\Gamma \vdash G}{A \wedge B,\Gamma \vdash G}(\wedge_L)$ $\dfrac{\Gamma \vdash A \quad \Gamma \vdash B}{\Gamma \vdash A \wedge B}(\wedge_R)$
- $\dfrac{A,\Gamma \vdash G \quad B,\Gamma \vdash G}{A \vee B,\Gamma \vdash G}(\vee_L)$ $\dfrac{\Gamma \vdash A}{\Gamma \vdash A \vee B}(\vee_{R1})$ $\dfrac{\Gamma \vdash B}{\Gamma \vdash A \vee B}(\vee_{R2})$

(Masaki Hara, slides)

First-order logic as a 2-theory

$$\frac{}{\Gamma, A \vdash A}\ hyp$$

$$\frac{\Gamma \vdash A \quad \Gamma \vdash B}{\Gamma \vdash A \wedge B}\ \vdash\wedge \qquad \frac{\Gamma, A \vdash B}{\Gamma \vdash A \to B}\ \vdash\to \qquad \frac{\Gamma \vdash A}{\Gamma \vdash A \vee B}\ \vdash\vee_L \qquad \frac{\Gamma \vdash B}{\Gamma \vdash A \vee B}\ \vdash\vee_R$$

$$\frac{\Gamma \vdash A \to B \quad \Gamma \vdash B \to A}{\Gamma \vdash A \equiv B}\ \vdash\equiv \qquad \frac{\Gamma \vdash A \to \bot}{\Gamma \vdash \neg A}\ \vdash\neg \qquad \frac{\Gamma \vdash P(m)}{\Gamma \vdash \forall x.P(x)}\ (\text{FRESH } m)\ \vdash\forall \qquad \frac{\Gamma \vdash P(B)}{\Gamma \vdash \exists x.P(x)}\ \vdash\exists$$

$$\frac{\Gamma, A, B \vdash C}{\Gamma, A \wedge B \vdash C}\ \wedge\vdash \qquad \frac{\Gamma \vdash A \quad \Gamma, B \vdash C}{\Gamma, A \to B \vdash C}\ \to\vdash \qquad \frac{\Gamma, A \vdash C \quad \Gamma, B \vdash C}{\Gamma, A \vee B \vdash C}\ \vee\vdash \qquad \frac{\Gamma, A \to B, B \to A \vdash C}{\Gamma, A \equiv B \vdash C}\ \equiv\vdash$$

$$\frac{\Gamma, A \to \bot \vdash B}{\Gamma, \neg A \vdash B}\ \neg\vdash \qquad \frac{}{\Gamma, \bot \vdash A}\ \bot\vdash \qquad \frac{\Gamma, P(B) \vdash C}{\Gamma, \forall x.P(x) \vdash C}\ \forall\vdash \qquad \frac{\Gamma, P(m) \vdash C}{\Gamma, \exists x.P(x) \vdash C}\ (\text{FRESH } m)\ \exists\vdash$$

$$\frac{\Gamma \vdash B \quad \Gamma, B \vdash C}{\Gamma \vdash C}\ cut \qquad \frac{\Gamma \vdash C}{\Gamma, A \vdash C}\ weaken \qquad \frac{\Gamma, A, A \vdash C}{\Gamma, A \vdash C}\ contract$$

(Richard Bornat and Bernard Sufrin, paper)

‘Someone is loved by everyone’ implies ‘everyone loves someone’.

$$\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{}{p(x,y) \vdash p(x,y)}\,(I)}{\forall x\,(p(x,y)) \vdash p(x,y)}\,(\forall L)}{\forall x\,(p(x,y)) \vdash \exists y\,(p(x,y))}\,(\exists R)}{\exists y\,(\forall x\,(p(x,y))) \vdash \exists y\,(p(x,y))}\,(\exists L)}{\exists y\,(\forall x\,(p(x,y))) \vdash \forall x\,(\exists y\,(p(x,y)))}\,(\forall R)$$

Note that this first-order reasoning doesn't exploit the full resources of the third 3-theory:

- $x : A, y : B, z : C, P(x), Q(y, z), R \vdash S(x, y, z)$

Usually the first-order logic you see is untyped (or unityped), so no A, B, C:

- $\cancel{x : A, y : B, z : C}, P(x), Q(y, z), R \vdash S(x, y, z)$

The domain remains implicit, and all variables range over it.

Note also that we don't show elements of the premises and conclusion.

Philosophers have chosen so far to use a logic that's less powerful in two ways: unityping and only dependent propositions (predicates).

It recovers 'types' by carving out from the total domain using predicates: entities that are P.

Surely this is rather odd though.

- When we say 'All humans are mortal', do we really mean for any 'thing' in the universe (time, place, emotion, event, possible object,...), then if it is the case that the thing is a human, then that thing is mortal.
- Do we say of a binary relation that it makes sense to ask if it holds for any two entities? 'Democracy is the mother of the Orca.'

I mentioned that types *are* available in a typed first-order logic, which allows for propositions depending on types A, B, ..., or in other words, typed predicates and relations.

- For $x : A$, we might define a predicate $B(x)$.

The big departure now is dependent **types**:

- $x : A \vdash B(x) : \mathit{Type}$.

This allows for the natural treatment of the common situation of a function $f : B \to A$, where we consider the $b : B$ being sent to a given $a : A$ as a subtype of B.

Assigning players to their team, for $t : \mathit{Team}$, $\mathit{Player}(t)$ collects players of the same team.

Dependent type 2-theories

- Some common type formation rules look like first-order logic's conjunction, disjunction, implication, quantification, except now they apply to all types. Propositions are now just a kind of type.
- We may want identity types: $Id_A(a, b)$.
- We may also want a type of (small) types to represent dependent types.

type formation	$\dfrac{\vdash X : \mathrm{Type} \quad x : X \vdash A(x) : \mathrm{Type}}{\vdash (\prod_{x:X} A(x)) : \mathrm{Type}}$
term introduction	$\dfrac{x : X \vdash a(x) : A(x)}{\vdash (x \mapsto a(x)) : \prod_{x':X} A(x')}$
term elimination	$\dfrac{\vdash f : (\prod_{x:X} A(x)) \quad \vdash x : X}{x : X \vdash f(x) : A(x)}$
computation rule	$(y \mapsto a(y))(x) = a(x)$

This is now a powerful 2-theory in which we can compare types via the existence of kinds of function, e.g., to compare sizes. Any notion of a mathematics/logic distinction is made problematic.

Identity only applies within a type. 'Venus $=$ 3' is not well-formed.

(We can consider a map $\widetilde{Type} \to Type$, from the type of all entities, with elements (A, a) for $a : A$, to the type of types.

Identity in this type doesn't allow us to ask whether $a = b$. But we can ask, 'Does $(A, a) = (B, b)$?'.)

From the 2-theories mentioned (judgment structure + type rules) we then formulate 1-theories. For instance, to do mathematics we might use FOL + set theory.

The excitement of the 3-theory DTT is that we can specify a 2-theory (HoTT) that's powerful enough to do maths, and do it 'naturally'.

We can then devise an applied DTT 1-theory, specifying worldly types and axioms, to describe the world.

An applied FOL 1-theory is much weaker. If we need more power, we modify the pure FOL 1-theory, set theory, to a set theory with atoms/urelements.

Higher-order logic

Within the same 3-theory as FOL, we find HOL. This is a 2-theory (or perhaps a family of 2-theories) where rules specify a universe of propositions.

There are proposed translations between kinds of DTT 2-theory and HOL 2-theory, e.g., Bart Jacobs' Translating dependent type theory into higher order logic.

The issue is one of naturalness, like a good programming language over another, even machine code.

There are interesting things for philosophy of language/metaphysics to learn from DTT 2-theories, and how we might devise worldly 1-theories.

- Ofra Magidor: Category mistakes
- Jason Stanley: 'All truth-conditional effects of extra-linguistic context can be traced to logical form.' (Language in Context, OUP, 2007, p. 30)

Consider an example from Stanley:

- Every time John lit a cigarette, it rained.

Despite appearances this can't merely be quantification over times. If John always lights a cigarette in New York and whenever he does it rains in London, this doesn't support the proposition. It must rain at the same location as John.

We might say that the verb 'rain' has implicit parameters for time and place. An achievement, such as 'John lights a cigarette', has an associated time and place. The proposition is claiming that every achievement which is John lighting a cigarette occurs at a time and place such that it rains then and there.

- $X : \mathit{Achievement\ Type}, x : X \vdash t(x) : \mathit{Time}, l(x) : \mathit{Location}$
- $u : \mathit{Time}, v : \mathit{Location} \vdash \mathit{Rain}(u, v) : \mathit{Prop}$
- $\vdash \mathit{John\ lights\ a\ cigarette} : \mathit{Achievement\ Type}$

Hence,

- $z : \mathit{John\ lights\ a\ cigarette} \vdash t(z) : \mathit{Time}; l(z) : \mathit{Location}$
- $\vdash \prod_{z:\mathit{John\ lights\ a\ cigarette}} \mathit{Rain}(t(z), l(z)) : \mathit{Prop}.$

There are hidden parameters all over the place:

- When you say 'every', it's really 'every$_A$': e.g., on returning from the shop, 'Every bottle is green'.
- Dependency on an element or a proposition is important. When you say 'X knows P', I claim that it's formed by:

  $X : \mathit{Person}, P : \mathit{Proposition}, p : P \vdash X \ \mathit{knows}\ P(p) : \mathit{Prop}$

Stanley speaks of ‘context-dependency’, ‘domain indices’, ‘covert pronomial elements’, and ‘unpronounced syntactic structure’.

Right! So use a logic that can handle dependency properly.

Please never let me see any untyped first-order logic again in this department!

A distant target

From the very beginning, from the first moment I may almost say, of my acquaintance with you, your manners impressing me with the fullest belief of your arrogance, your conceit, and your selfish disdain of the feelings of others, were such as to form the groundwork of disapprobation, on which succeeding events have built so immovable a dislike.

A proposition, so a type which ought to be formed by type-formation rules.

Very prominent here is the temporal structure (Secs 2.6 and 4.3 of my book).

During an initial interval of their acquaintance, which is almost an instant, disapproval is generated. Subsequent to that a number of events have occurred in the remaining time interval which have brought about a powerful dislike.