# Draft

Dimitrios Koutsoulis
11838639

July 1, 2019

# 1 Type Theory

## 1.1 Introduction

Type theory is a formal language and deductive system, that is self sufficient in the sense that it need not be formulated as a collection of axioms on top of some other formal system like First Order Logic, instead its deductive system can be built on top of its own formal language. longer sentence to check formating

Central to Type Theory is the notion of *Type*. Every term $a$ in Type Theory we come across, must lie in some type $A$, which we denote as $a : A$. Note that the relation $:$ is transitive, so $a : A$ and $A : B$ imply $a : B$.

For the deductive part of Type Theory, we interpret propositions as types. Proving proposition $P$ amounts to providing some inhabitant $p : P$.

## 1.2 Type Construction Operations

Let's have a look at some important type constructions.

- Given types $A, B : \mathcal{U}$ we can define the type $A \to B$ of functions from $A$ to $B$. We can use $\lambda$-abstraction to construct elements of this type. $\lambda x.\Phi$ lies in $A \to B$ iff for $a : A$ we have $\Phi[a/x] : B$. For $f : A \to B$

and $a : A$ we have that the application of $f$ on $a$, denoted as $f(a)$ or $f\ a$, lies in $B$.

- Given some type $A : \mathcal{U}$ and a family of types $B$ over $A$, $B : A \to \mathcal{U}$, we have the type of dependent functions

$$\prod_{a:A} B(a)$$

where for $f : \prod_{a:A} B(a)$ and $x : A$ we have $f(x) : B(x)$. As in the case of non-dependent functions, we can use lambda abstraction to construct elements of a dependently-typed function type.

- Given $A, B : \mathcal{U}$ we can define the product type $A \times B : \mathcal{U}$. For $a : A$ and $b : B$ we have the pair $(a, b) : A \times B$. We also have the projection functions

$$\mathtt{pr}_1 : A \times B \to A : (a, b) \mapsto a$$

$$\mathtt{pr}_2 : A \times B \to B : (a, b) \mapsto b$$

- Given $A : \mathcal{U}$ and family of types $B$ over $A$, $B : A \to \mathcal{U}$, we can define the dependent pair type

$$\sum_{a:A} B(a)$$

Given $x : A$ and $b : B(x)$ we can construct the pair $(x, b) : \sum_{a:A} B(a)$. We have two projection functions, similar to the case of the product type.

- Given $A, B : \mathcal{U}$ we can construct the coproduct type $A + B$. We can construct elements of $A + B$ using the functions

$$\mathtt{inl} : A \to A + B$$

$$\mathtt{inr} : B \to A + B$$

This induces the induction principle

$$\mathtt{ind}_{A+B} : \prod_{C:A+B\to U} \left(\prod_{a:A} C(\mathtt{inl}\ a)\right) \to \left(\prod_{b:A} C(\mathtt{inr}\ b)\right) \to \prod_{x:A+B} C(x)$$

- Given $x, y : A$ we have the **identity type** $x =_A y$. An element of this type amounts to a proof that $x$ and $y$ are equal. Say $x$ and $y$ are judgmentally equal. This is captured by the element $\mathtt{idp}_x : x =_A y$. The relevant induction principle describes how we can use elements of an identity type

$$\mathtt{ind}_{=_A} : \prod_{C:\prod_{x,y:A}(x=_A y)\to\mathcal{U}} \Big(\prod_{x:A} C(x,x,\mathtt{idp}_x)\Big) \to \prod_{x,y:A}\prod_{(p:x=_A y)} C(x,y,p)$$

  The relevant computation gives us the judgmental equality

$$\mathtt{ind}_{=_A}(C,c,x,x,\mathtt{idp}_\mathtt{x}) \equiv c\ x$$

  We can concatenate those paths whose domains and codomains allow for it. Paths are equivalences. That is if $p : x = y$ is such a path, we can provide its inverse $p^{-1}$ for which we have in turn a path between $p \cdot p^{-1}$ and $\mathtt{idp}_y$ and another one between $p^{-1} \cdot p$ and $\mathtt{idp}_x$.

- For every type $A$ there is its **propositional truncation** $\|A\|$. For every element $a : A$ there exists $|a| : \|A\|$. For every $x, y : \|A\|$ we have $x = y$. Given mere proposition $B$ and $f : A \to B$, the recursion principle gives us $g : \|A\| \to B$ such that $g(|a|) \equiv f(a)$ for all $a : A$.

## 1.3 Important types

**Definition 1.1.** *We call a type $A$ a **mere proposition** if for every $a, b : A$ we have $a = b$.*
**Definition 1.2.** *We call a type $A$ a **set** if for every $a, b : A$ we have that $a =_A b$ is a mere proposition.*
**Definition 1.3.** *We call a type $A$ **contractible** if there exists $a : A$ such that for all $x : A$ it holds that $x = a$.*

## 1.4 Logic

Our informal deductions in Type Theory will be reminiscent of First Order Logic ones. To be able to use a similar verbiage, we will set down a handful of types, corresponding to the connectives that let us form well-formed

formulas in FOL. These types need to be mere propositions, so that we can form non-constructive deductions. This approach is called 'propositions as h-propositions' in the HoTT book. In the following, $A$ and $B$ are mere propositions.

- When we talk of conjunction $A \wedge B$, we mean the product $A \times B$.
- We interpret $A \vee B$, as the truncation $\|A + B\|$.
- We interpret $\forall a \in A,\ P(a)$, where $P(a)$ is a mere proposition for all $a \in A$, as $\prod_{a:A} P(a)$.
- We interpret $\exists a \in A,\ P(a)$, where $P(a)$ is a mere proposition for all $a \in A$, as $\|\Sigma_{a:A} P(a)\|$

The use of the above notation is interchangeable in the sections to follow.

# 2 Modalities

**Definition 2.1.** *A **modality** is a function $\bigcirc : \mathcal{U} \to \mathcal{U}$ with the following properties.*

1. *For every type $A$ we have a function $\eta^{\bigcirc}_A : A \to \bigcirc A$*
2. *for every $A : \mathcal{U}$ and every type family $B : \bigcirc A \to \mathcal{U}$ we have a function*

$$\mathtt{ind}_{\bigcirc} : \Big(\prod_{a:A} \bigcirc(B(\eta^{\bigcirc}_A\ a))\Big) \to \prod_{z:\bigcirc A} \bigcirc(B\ z)$$

3. *For every $f : \prod_{a:A} \bigcirc(B(\eta^{\bigcirc}_A\ a))$ there is a path $\mathtt{ind}_{\bigcirc}(f)(\eta^{\bigcirc}_A\ a) = f\ a$*
4. *For all $z, z' : \bigcirc A$, the function $\eta^{\bigcirc}_{z=z'} : (z = z') \to \bigcirc(z = z')$ is an equivalence.*

A modality $\bigcirc$ induces a $\Sigma$-closed reflective subuniverse.

**Definition 2.2.** *Given modality $\bigcirc : \mathcal{U} \to \mathcal{U}$, a **reflective subuniverse** is a 'subset' of $\mathcal{U}$ encoded by a family of h-propositions $P : \mathcal{U} \to \mathtt{Prop}$ such that the following conditions hold.*

- *For $A : \mathcal{U}$, we have $P(\bigcirc A)$.*

- *For $A : \mathcal{U}$ and $B$ such that $P(B)$, the function*

$$\lambda f.f \circ \eta^{\bigcirc}_{A} : (\bigcirc A \to B) \to (A \to B)$$

  *is an equivalence.*

*The subuniverse is* $\Sigma$***-closed*** *if for $X$ such that $P(X)$ and $Q : X \to \mathcal{U}$ such that $\prod_{x:X} P(Q(x))$, we have $P(\Sigma_{x:X} Q(x))$.*

**Theorem 2.3.** *Reflective subuniverses are closed under products. That is for subuniverse $P$ and $B : A \to \mathcal{U}$ such that $\prod_{a:A} P(B(a))$, we have that $P(\prod_{a:A} B(a))$.*

*Proof.* For $a : A$, consider $\mathtt{ev}_a : (\prod_{a:A} B(a)) \to B(a)$ defined by $\mathtt{ev}_a(f) :\equiv f(x)$. Since $P(B(a))$, we have

$$(\lambda f.f \circ \eta^{\bigcirc}_{\prod_{a:A} B(a)})^{-1}(\mathtt{ev}_a) : \bigcirc\big(\prod_{a:A} B(a)\big) \to B(a)$$

We can now define the retraction of $\eta^{\bigcirc}_{\prod_{a:A} B(a)}$ by pattern matching as such: For $z : (\prod_{a:A} B(a))$ and $a : A$ we have

$$(\lambda f.f \circ \eta^{\bigcirc}_{\prod_{a:A} B(a)})^{-1}(\mathtt{ev}_a)(z) : B(a)$$

□

**Definition 2.4.** *For $B : A \to \mathcal{U}$, we call $X$ $B$****-null*** *if the map*

$$\lambda x.\lambda b.x : X \to (B(a) \to X)$$

*is an equivalence for all $a : A$.*

# 3 LLPO

**Definition 3.1.** *The Lesser Limited Principle of Omniscience, states that given binary sequence $s : \mathbb{N} \to \mathbf{2}$ and the fact that there is at most one occurence of 1 in the sequence, formally*

$$\mathtt{atMost1one} : \prod_{n_1:\mathbb{N}} \prod_{n_2:\mathbb{N}} s(n_1) = 1 \to s(n_2) = 1 \to n_1 = n_2$$

*we can then have by the LLPO a witness for* $\mathtt{p_{odd}} \vee \mathtt{p_{even}}$*, where* $\mathtt{p_{odd}}$ *(with* $s$ *as an implicit argument) is the statement that for all odd positions* $n$*,* $s(n) = 0$*, formally* $\mathtt{p_{odd}} \equiv \prod_{n:\mathbb{N}} \mathtt{odd}(n) \to s(n) = 0$*. Similarly for* $\mathtt{p_{even}}$*.*

LLPO can be viewed as a weaker form of the Law of Excluded Middle.

**Lemma 3.2.** *The Law of Excluded Middle implies the Limited Principle of Omniscience.*

*Proof.* By LEM we have a witness $l_1 : \mathtt{p_{odd}} \vee \neg \mathtt{p_{odd}}$. Since this is a coproduct, by the relevant principle of induction, it's enough to prove LLPO from the disjuncts.

- $\mathtt{p_{odd}} \Rightarrow$ LLPO, trivially.
- $\neg \mathtt{p_{odd}}$, alongside LEM, implies that there exists odd $n_e : \mathbb{N}$ such that $s(n_e) = 1$. We can now provide $\prod_{n:\mathbb{N}} \mathtt{even}(n) \to s(n) = 0$. Let $n : \mathbb{N}$. By the definition of $s$, $s(n) = 0 \vee s(n) = 1$. We invoke the principle of induction of coproducts and prove LLPO from the disjuncts.
  - $s(n) = 0$, in this case we are done.
  - $s(n) = 1$. By `atMost1one` we have $n = n_e \Rightarrow n$ even and odd which is a contradiction $c : \bot$. Ex Falso, $\mathtt{efq} : \bot \to s(n) = 0$. Then $\mathtt{efq}(c) : s(n) = 0$.

□

**Lemma 3.3.** *If we replace the consequent of LLPO with its double negation, let's call it* ¬¬*LLPO, then we can prove it in Type Theory.*

*Proof.* From LEM ⇒ LLPO we have LEM ⇒ ¬¬LLPO by effectively the same proof. Since we invoke LEM only finitely many times, we can use their finite conjunction to prove ¬¬LLPO. The double negations of those instances are provable in Type Theory (include the proof of this?). We can leverage the fact that double negation is a modality, to construct a function that assumes the double negations of the LEM instances and concludes ¬¬LLPO. □

# 4 Extensional Type Theory

There exists a model of extensional type theory that validates the following 'Independence Principle'

$$\Big(\prod_{s:\mathbf{2}^{\mathbb{N}}} P\ s \to \big(\prod_{n:\mathbb{N}} s\ n = 0\big) \to \Big\|\sum_{z:\mathbb{N}} Q\ s\ z\Big\|\Big)$$

$$\to \Big(\prod_{s:\mathbf{2}^{\mathbb{N}}} P\ s \to \Big\|\sum_{z:\mathbf{2}} \big(\prod_{n:\mathbb{N}} s\ n = 0\big) \to Q\ s\ z\Big\|\Big)$$

where $P : \mathbf{2}^{\mathbb{N}} \to \mathtt{Prop}$ and $Q : \mathbf{2}^{\mathbb{N}} \to \mathbb{N} \to \mathtt{Prop}$ are families of propositions. Putting it plainly, if the left hand side of the above is true, then $z : \mathbb{N}$ does not depend on the proof of the constantness of $s$ to 0.

The same model also validates Markov's Principle

$$\Big(\prod_{a:A}(P\ a \vee \neg P\ a)\Big) \to \Big(\neg\prod_{a:A} \neg P\ a\Big) \to \Big\|\sum_{a:A} P\ a\Big\|$$

where $P : A \to \mathtt{Prop}$ is a family of propositions over $A : \mathcal{U}$. Informally, Markov's Principle states that if we have a collection of decidable propositions and the fact that not all of them are false, then one of them must be true.

In this section we will be working in the context of this model.

Let $A :\equiv \sum_{a:\mathbf{2}^{\mathbb{N}}} \mathtt{atMost1one}\ a$. When referring to elements of $A$ we implicitly mean the first part of the pair. Let

$$B : A \to \mathtt{Prop}$$

$$B :\equiv \lambda\, a.\ \mathtt{p_{odd}}\ a \vee \mathtt{p_{even}}\ a$$

**Lemma 4.1.** $\mathbb{N}$ *is* $B$*-null.*

*Proof.* Given $a : A$ and $f : B\ a \to \mathbb{N}$ we need to prove that there exists unique $f' : \mathbf{1} \to \mathbb{N}$ through which $f$ factors, in the sense that $f = f' \circ g$, where $g : B\ a \to \mathbf{1}$ is the sole inhabitant of its function space. Functions with $\mathbf{1}$ as their domain are constant. We therefore need to find some $z : \mathbb{N}$

so that $\lambda\ \_.\ z$ is $f'$. To that end, we will use the following instance of the Independence Principle

$$\Big(\prod_{s:\mathbf{2}^{\mathbb{N}}}\Big((\prod_{n:\mathbb{N}} s(n) = a(2{\cdot}n))\vee(\prod_{n:\mathbb{N}} s(n) = a(2{\cdot}n{+}1))\Big) \to \Big(\prod_{n:\mathbb{N}} s\ n = 0\Big) \to \Big\|\sum_{z:\mathbb{N}}\prod_{p:B\ a} f\ p = z\Big\|\Big)$$

$$\to \Big(\prod_{s:\mathbf{2}^{\mathbb{N}}}\Big((\prod_{n:\mathbb{N}} s(n) = a(2{\cdot}n))\vee(\prod_{n:\mathbb{N}} s(n) = a(2{\cdot}n{+}1))\Big) \to \Big\|\sum_{z:\mathbb{N}}\Big(\prod_{n:\mathbb{N}} s\ n = 0\Big) \to \prod_{p:B\ a} f\ p = z\Big\|\Big)$$

Note that the left hand side is provable. By providing the odd $s_1$ and even $s_2$ subsequences of $a$ to the right hand side, we get hold of $z_1 : \mathbb{N}$ and $z_2 : \mathbb{N}$ respectively. Equality on $\mathbb{N}$ is decidable, therefore $z_1 = z_2 \vee z_1 \neq z_2$ is provable. Since our goal is a proposition, we can drop the truncation of the disjunction. This leaves us with a coproduct.

We will first prove that the existence of a candidate for $f'$ follows from both constituents of the coproduct.

- First the case where $z_1 = z_2$. We arbitrarily pick $z_1$ and propose $f' :\equiv \lambda\ \_.\ z_1$. By axiom K we have function extensionality, so we reduce proving $f = f' \circ g$ to proving $f\ b = f' \circ g\ b$ for arbitrary $b : B\ a$. $b$ is a truncated coproduct witness. As before we can ignore the truncation and take cases on the coproduct.
  - We first consider the case $\mathtt{p_{odd}}\ a$. This tells us that the odd subsequence is constantly 0, which we can provide to the rhs of the IP alongside with $b$ itself (and $s_1$ as the first argument) to get $f\ \ = z_1$. We concatenate this with $f' \circ g\ b = z_1$ to conclude this case.
  - The $\mathtt{p_{even}}$ case is very similar. The only difference is that we provide $s_2$ as the first argument to the rhs of the IP and then have $z_1 = z_2$ be part of the concatenation of paths when we prove homotopy between $f$ and $f' \circ g$.
- Now consider the case $z_1 \neq z_2$. For any $n : \mathbb{N}$, $a\ n = 1$ is decidable. Suppose that $\prod_{n:\mathbb{N}} a\ n \neq 1$. Then $a$ and by extension $s_1$ and $s_2$ are constantly 0. By the rhs of the IP we then have $z_1 = f\ b = z_2$ where $b : B\ a$ is clearly available to us at this point. This contradicts $z_1 \neq z_2$. We have proven $\neg \prod_{n:\mathbb{N}} a\ n \neq 1$. By Markov's Principle there exists $n_1 : \mathbb{N}$ such that $a\ n_1 = 1$. We pick the subsequence $s_0$ with parity

opposite to $n_1$ and let $f'$ be always equal to the corresponding $z_0$. We prove $\prod_{n:\mathbb{N}} s_0\ n = 0$ by induction on $\mathbb{N}$ and cases on $s_0\ n$. In the case where $s_0\ n = 1$ we can reach falsum because $n$ will have to be both even and odd (in $a$) by `atMost1one` $a$. Ex falso trivializes this case. Now that we have established this too, we have enough arguments for the rhs of IP to output the desired equality $f\ b = z_0$ which proves homotopy between $f$ and $(\lambda\ \_.\ z_0) \circ g$.

We still need to prove the uniqueness of $f'$. Since $\mathbf{1}$ is finite and equality on $\mathbb{N}$ is decidable, we can decide equality on the function space

$$p : \prod_{f_1, f_2 : \mathbf{1} \to \mathbb{N}} f_1 = f_2 + f_1 \neq f_2$$

We will use this to prove that $\sum_{h:\mathbf{1}\to\mathbb{N}} f = h \circ g$ is a mere proposition, which entails the uniqueness of $f'$. Consider $h_1, h_2 : \mathbf{1} \to \mathbb{N}$ such that $f = h_i \circ g,\ i \in \{1, 2\}$. By $p$ we have $(h_1 = h_2) + (h_1 \neq h_2)$. We use induction on the coproduct. The left case is trivial. For the right case we have $q : h_1 \neq h_2$ and we will first try to prove $B\ a \to \bot$ as a stepping stone. Consider $b : B\ a$. Since $h_1 \circ g = f = h_2 \circ g$, by homotopy $h_1(g\ b) = h_2(g\ b)$. We can then prove $\prod_{*:\mathbf{1}} h_1(*) = h_2(*)$. By function extensionality $h_1 = h_2$. But this contradicts $q$, so we reach $\bot$. We've just proven $r : B\ a \to \bot$. We still have to prove $h_1 = h_2$, the main goal. By Lemma 3.3 and $r$ we get $\bot$. We conclude the proof by Ex Falso. □