Notes on Dependent Type Theory

Alyson Mei

January 20, 2026

These notes are based on video recordings of R. Harper's lectures.

Conventions.

- Bullet lists are used for strongly logically connected items (primarily in definitions), while dashed lists are used for (somewhat) distinct items;
- Remarks marked with * are additions by the present author.

1 Introduction

Boolean algebra is associated with classical logic, Heyting algebra – with intuitionistic logic.

Definition 1.1 (Boolean algebra)**.** A *Boolean algebra* can be defined as a complemented distributive lattice:

- Pre-order:
$$x \le x, \qquad x \le y \,\&\, y \le z \to x \le z;$$
- Has finite meets and joins:
$$x \le 1, \qquad z \le x \,\&\, z \le y \to z \le (x \wedge y), \qquad x \wedge y \le x, \qquad x \wedge y \le y;$$
$$0 \le x, \qquad x \le z \,\&\, y \le z \to x \vee y \le z, \qquad x \le x \vee y, \qquad y \le x \vee y;$$
- Has complements:
$$1 \le \bar{x} \vee x, \qquad \bar{x} \wedge x \le 0;$$
- Distributive:
$$x \wedge (y \vee z) \equiv (x \wedge y) \vee (x \wedge z), \qquad x \vee (y \wedge z) \equiv (x \vee y) \wedge (x \vee z).$$

Additionally, the exponential is defined as
$$y^x := \bar{x} \vee y.$$

Definition 1.2 (Heyting algebra)**.** A *Heyting algebra* is defined as a lattice with exponentials:

- Pre-order;
- Has finite meets and joins;

- Has exponentials:
$$y^x \wedge x \leq y, \qquad z \wedge x \leq y \to z \leq y^x.$$

Exercise 1.3.

1. "Yoneda lemma": $x \leq y$ iff $\forall z\ (z \leq x \to z \leq y)$;
2. Every Heyting algebra is distributive.

Quotes:

- "Boolean algebra (closed world) = Heyting algebra (open world) with complements";
- "Classical logic is a logic with complete information".

The definitions provide us with standard rules:

- Weakening:
$$x \leq x \vee y \qquad (x \wedge y \leq x);$$
- Contraction:
$$x \leq x \wedge x;$$
- Exchange:
$$x \wedge y \equiv y \wedge x.$$

Relation to classical logic:

- Sequent $\Gamma \vdash A$, where $\Gamma = A_1, \ldots, A_n$, corresponds to:
$$A_1 \wedge \cdots \wedge A_n \leq A;$$
- Rules for $\wedge$:
$$\frac{\Gamma \vdash A \quad \Gamma \vdash B}{\Gamma \vdash A \wedge B} \qquad \frac{}{\Gamma, A \wedge B \vdash A} \qquad \frac{}{\Gamma, A \wedge B \vdash B}\ ;$$
- ...and so on.

Definition 1.4 (Lindenbaum algebra)**.** A *Lindenbaum algebra* is defined as the algebra of equivalence classes of a given theory:

- $[A] = \{B \mid B \equiv A\}$;
- $[A] \wedge [B] := [A \wedge B]$;
- ... and so on.

Theorem 1.5 (Soundness and Completeness for Intuitionistic Propositional Logic)**.** Let Γ be a context and A a formula. Then
$$\Gamma \vdash A \quad \text{iff} \quad \forall H \left([\![\Gamma]\!]_H \leq [\![A]\!]_H \right),$$
where H ranges over all Heyting algebras, and $[\![-]\!]_H$ denotes the interpretation of formulas as elements of H under a valuation of propositional variables.

2 Simple Type Theory

Definition 2.1 (Simple type theory)**.** We define the *simple type theory* as follows:

- Unit 1:

$$\frac{}{\Gamma \vdash 1 \ \text{type}}\ 1\text{-}F \qquad \frac{}{\Gamma \vdash \langle\rangle : 1}\ 1\text{-}I \qquad \text{(no 1-E)}\ ;$$

- Product $\times$:

$$\frac{\Gamma \vdash A \ \text{type} \quad \Gamma \vdash B \ \text{type}}{\Gamma \vdash A \times B \ \text{type}}\ \times\text{-}F \qquad \frac{\Gamma \vdash M : A \quad \Gamma \vdash N : B}{\Gamma \vdash \langle M, N\rangle : A \times B}\ \times\text{-}I \qquad \frac{\Gamma \vdash M : A \times B}{\begin{array}{c}\Gamma \vdash \text{fst}(M) : A \\ \Gamma \vdash \text{snd}(M) : B\end{array}}\ \times\text{-}E \ ;$$

- Exponential $\to$:

$$\frac{\Gamma \vdash A \ \text{type} \quad \Gamma \vdash B \ \text{type}}{\Gamma \vdash A \to B \ \text{type}} \to\text{-}F \qquad \frac{\Gamma, x : A \vdash M : B}{\Gamma \vdash \lambda x.M : A \to B} \to\text{-}I$$

$$\frac{\Gamma \vdash M : A \to B \quad \Gamma \vdash N : A}{\Gamma \vdash M(N) : B} \to\text{-}E \ ;$$

- Void 0:

$$\frac{}{\Gamma \vdash 0 \ \text{type}}\ 0\text{-}F \qquad \text{(no 0-I)} \quad \frac{\Gamma \vdash M : 0}{\Gamma \vdash \text{abort}(M) : A}\ 0\text{-}E \ ;$$

- Coproduct $+$:

$$\frac{\Gamma \vdash A \ \text{type} \quad \Gamma \vdash B \ \text{type}}{\Gamma \vdash A + B \ \text{type}}\ +\text{-}F \qquad \frac{\Gamma \vdash M : A}{\Gamma \vdash \text{inl}(M) : A + B}\ +\text{-}I_1 \qquad \frac{\Gamma \vdash N : B}{\Gamma \vdash \text{inr}(N) : A + B}\ +\text{-}I_2 \ ,$$

$$\frac{\Gamma, x : A \vdash N : C \quad \Gamma, y : B \vdash P : C}{\Gamma, z : A + B \vdash \text{case}(x.N, y.P)(z) : C}\ +\text{-}E \ .$$

Remark 2.2 (Categorical interpretation*)**.** The type theory given above corresponds to a category $\mathcal{C}$ that is both cartesian closed and cocartesian. We also assume that $\mathcal{C}$ has all morphisms from the terminal object 1 to the objects corresponding to the types of context variables (this is somewhat experimental).

- Types are the objects of $\mathcal{C}$;
- A context $\Gamma = x_1 : A_1, \ldots, x_n : A_n$ corresponds to the product of its types, together with a morphism from the terminal object naming the variables:

$$\Gamma := A_1 \times \cdots \times A_n, \qquad \langle x_1, \ldots, x_n\rangle : 1 \to \Gamma;$$

- Terms are morphisms in $\mathcal{C}$:

$$\Gamma \vdash M : A \quad \mapsto \quad M : \Gamma \to A;$$

- Unit 1 is a terminal object; the 1-I rule corresponds to arrows *to* 1. Void 0 is an initial object; the 0-E rule corresponds to arrows *from* 0:

$$\begin{tikzcd}
\Gamma \arrow[d, dashed, "\langle\rangle"] \\
1
\end{tikzcd}
\qquad
\begin{tikzcd}
\Gamma \arrow[r, "M"] \arrow[dr, "\mathrm{abort}(M)"'] & 0 \arrow[d, dashed, "\mathrm{abort}"] \\
 & A
\end{tikzcd} ;$$

- Product $\times$ and coproduct $+$ are the usual categorical limits and colimits (finite in our case):

$$\begin{tikzcd}
 & \Gamma \arrow[dl, "M"'] \arrow[dr, "N"] \arrow[d, dashed, "\langle M,N\rangle"] & \\
A & A\times B \arrow[l, "\mathrm{fst}"] \arrow[r, "\mathrm{snd}"'] & B
\end{tikzcd}
\qquad
\begin{tikzcd}
 & \Gamma \arrow[dl, "\mathrm{fst}(M)"'] \arrow[dr, "\mathrm{snd}(M)"] \arrow[d, dashed, "M"] & \\
A & A\times B \arrow[l, "\mathrm{fst}"] \arrow[r, "\mathrm{snd}"'] & B
\end{tikzcd} ,$$

$$\begin{tikzcd}
 & A+B \\
A \arrow[ur, "\mathrm{inl}"] & \Gamma \arrow[l, "M"] \arrow[u, "\mathrm{inl}(M)"']
\end{tikzcd}
\qquad
\begin{tikzcd}
A+B & \\
B \arrow[u, "\mathrm{inr}(N)"] & \Gamma \arrow[l, "N"] \arrow[ul, "\mathrm{inr}"']
\end{tikzcd} ,$$

$$\begin{tikzcd}
 & \Gamma\times 1 \arrow[dl, "\Gamma\times x"'] \arrow[d, dashed, "\Gamma\times z"] \arrow[dr, "\Gamma\times y"] & \\
\Gamma\times A \arrow[dr, "N"'] & \Gamma\times(A+B) \arrow[d, dashed, "\mathrm{case}(x.N,y.P)"] & \Gamma\times B \arrow[dl, "P"] \\
 & C &
\end{tikzcd} ;$$

- Exponential $\to$ is the right adjoint to product $\times$:

$$(A\times -) \dashv (A\to -)$$

$$\begin{tikzcd}
\Gamma \arrow[d, "\lambda x.M"'] \\
A\to B
\end{tikzcd}
\qquad
\begin{tikzcd}
\Gamma\times A \arrow[d, "M"'] & \Gamma\times 1 \arrow[l, "\mathrm{id}_\Gamma\times x"'] \arrow[dl, "M\circ\mathrm{id}_\Gamma\times x"] \\
B &
\end{tikzcd} ;$$

$$\begin{tikzcd}
\Gamma \arrow[d, "M"'] \\
A\to B
\end{tikzcd}
\quad
\begin{tikzcd}
\Gamma \arrow[d, "N"'] \\
A
\end{tikzcd}
\qquad
\begin{tikzcd}
\Gamma \arrow[d, "\langle M,N\rangle"'] \arrow[dr, "\varepsilon_B\circ\langle M,N\rangle"] & \\
A\times(A\to B) \arrow[r, "\varepsilon_B"'] & B
\end{tikzcd} .$$

Remark 2.3 (Categorical notation*)**.** Formally, we should've been writing $1\to\Gamma$ instead of just Γ for each diagram. But for brevity, we explicitly defined only those variables that are mentioned in the type definitions. Maybe I will refine this later.

Definition 2.4 (β and η equivalences)**.**

– Unit 1:

$$(\beta)\text{ none} \qquad (\eta)\ \Gamma \vdash \langle\rangle \equiv M : 1;$$

– Product $\times$:

$$(\beta)\begin{array}{l}\Gamma \vdash \mathrm{fst}(\langle M, N\rangle) \equiv M : A \\ \Gamma \vdash \mathrm{snd}(\langle M, N\rangle) \equiv N : B\end{array} \qquad (\eta)\ \Gamma \vdash \langle \mathrm{fst}\, M, \mathrm{snd}\, M\rangle \equiv M : A \times B;$$

– Exponential $\to$:

$$(\beta)\ \Gamma \vdash (\lambda x.M)(N) \equiv [N/x]M : B \qquad (\eta)\ \Gamma \vdash (\lambda x.M(x)) \equiv M : A \to B;$$

– Zero 0:

$$(\beta)\text{ none} \qquad (\eta)\ \Gamma, z : 0 \vdash R \equiv \mathrm{abort}(M) : C;$$

– Coproduct $+$:

$$(\beta)\begin{array}{l}\mathrm{case}(x.M; y.N)(\mathrm{inl}(P)) \equiv [P/x]M : C \\ \mathrm{case}(x.M; y.N)(\mathrm{inr}(Q)) \equiv [Q/y]N : C\end{array},$$

$$\frac{\Gamma \vdash [\mathrm{inl}(P)/z]R \equiv [P/x]M \quad \Gamma \vdash [\mathrm{inr}(Q)/z]R \equiv [Q/y]N}{\Gamma, z : A + B \vdash R \equiv \mathrm{case}(x.M; y.N)(z)}\ (\eta).$$

Next, we're going to augument this STT with "data" types.

Definition 2.5 (Natural numbers type)**.** The *natural numbers type* is defined as follows:

- Introduction: zero is a Nat, succ(x : Nat) is a Nat:

$$\frac{}{\Gamma \vdash \mathrm{zero} : \mathrm{Nat}} \quad \frac{}{\Gamma, x : \mathrm{Nat} \vdash \mathrm{succ}(x) : \mathrm{Nat}};$$

- Elimination:

$$\frac{\Gamma \vdash M : C \quad \Gamma, x : C \vdash N : C}{\Gamma, z : \mathrm{Nat} \vdash \mathrm{iter}(M, x.N)(z) : C},$$

  given by the recursion:

$$\mathrm{iter}(M, x.N)(\mathrm{zero}) = M, \qquad \mathrm{iter}(M, x.N)(\mathrm{succ}(n)) = [\mathrm{iter}(M, x.N)(n)/x]N.$$

Note: (η) here is not easy to formulate, because it would require ω-rule.

(next there were brief comments by R.Harper about inductive types and categorical notation)

Remark 2.6 (Explaining the iter*)**.** Given $M \in C$, $f \in \mathrm{Hom}(C, C)$ and $z \in \mathbb{N}$, iter essentially applies function f to M z times:

$$\mathrm{iter} : C \times \mathrm{Hom}(C, C) \times \mathbb{N} \to C, \qquad \mathrm{iter} : (M, f, z) \mapsto f_{(z)}(...(f_{(1)}(M))).$$

The recursion takes the following form:

$$\mathrm{iter}(M, f, 0) = M, \qquad \mathrm{iter}(M, f, \mathrm{succ}(n)) = f(\mathrm{iter}(M, f, n)).$$

Remark 2.7 (Natural numbers object*)**.** In a category $\mathcal{C}$ with a terminal object 1, a *natural number object (NNO)*

$$(N, z : 1 \to N, s : N \to N)$$

is an initial object in the category induced by morphisms $a : (x, g) \to (y, h)$:

$$\begin{tikzcd}
1 \arrow[r, "x"] \arrow[dr, "y"'] & X \arrow[r, "g"] \arrow[d, "a"] & X \arrow[d, "a"] \\
& Y \arrow[r, "h"'] & Y
\end{tikzcd}
\qquad
\begin{tikzcd}
1 \arrow[r, "z"] \arrow[dr, "q"'] & N \arrow[r, "s"] \arrow[d, dashed, "u"] & N \arrow[d, dashed, "u"] \\
& A \arrow[r, "f"'] & A
\end{tikzcd} .$$

Example 2.8 (Addition for Nat)**.** We define plus : Nat → Nat → Nat as:

$$\lambda x \lambda y.\, \mathrm{iter}(x, x'.\mathrm{succ}(x'))(y)$$

3 Families of types

Motivation: Propositions as Types:

- Proposition $\sim$ Type,
- Predicate ("propositional function") $\sim$ Family of types ("typical function")