CHAPTER 1

Logical Theories

To start, we will consider the notion of a *logical theory*; in my mind, it begins with a species (or set) of judgements that can be proposed, asserted, and (if they are evident) known.

1. Judgements of a logical theory

The basic forms of judgement for a logical theory will be $\boxed{P\ \mathit{prop}}$ and $\boxed{P\ \mathit{true}}$; and what is P? It is a member of the species of terms, which are made meaningful in the course of making the judgement P *prop* ("P is a proposition") evident for a proposition P.

The forms of judgement may be construed as containing *inputs* and *outputs*; an *input* is something which is inspected in the course of knowing a judgement, whereas an *output* is something which is synthesized (or created) in the course of knowing a judgement. The positions of *inputs* and *outputs* in constitute for a judgement form what is called its *mode*, and we color-code it in this presentation for clarity.[1]

To each judgement is assigned a *meaning explanation*, which explicates the knowledge-theoretic content of the judgement. For a judgement $\mathcal{J}$, a meaning explanation should be in the form:

> To know $\mathcal{J}$ is to know...

The meaning of the judgement P *prop* is, then, as follows:

> To know P *prop* is to know that P is a proposition, which is to know what would count as a direct verification of P.

So if a symbol P is taken to denote a proposition, we must know *what sort of thing* is to be taken as a direct verification of P, and this is understood as part of the definition of P. A "direct verification" is understood in constrast with an "indirect verification", which is to be thought of as a means or plan for verifying the proposition; these distinctions will be explained in more detail later on. Now,

[1]We will not see any judgements with *outputs* at first, but it will become necessary as soon as we consider judgements about computation, where the reduction of a term is synthesized from the redex. Modes may be used to construe a judgement as having algorithmic content.

the judgement P *true* ("P is true") is only meaningful in case we know P *prop* (this is called a presupposition). Then the meaning of P *true* is as follows:

> To know P *true* is to have a verification of P.

From the (implicit) presupposition P *prop*, we already know what counts as a verification, so the meaning explanation is well-defined. Note that having a means or plan for verifying P is equivalent to having a (direct) verification; this follows from the fact that one may put into action a plan for verifying P and achieve such a verification, and likewise, it is possible to propound a plan of verification by appeal to an existing verification.

2. Higher-order judgements

The judgements we have described so far are "categorical" in the sense that they are made without assumption or generality.

2.1. Hypothetical judgement. We will need to define a further form of judgement, which is called "hypothetical", and this is the judgement under hypothesis $\boxed{\mathcal{J}\ (\mathcal{J}')}$, pronounced "$\mathcal{J}$ under the assumption $\mathcal{J}'$".Its meaning explanation is as follows:

> To know the judgement $\mathcal{J}\ (\mathcal{J}')$ is to know the categorical judgement $\mathcal{J}$ assuming you know the judgement $\mathcal{J}'$.

Hypothetical judgement may be iterated, and $\mathcal{J}\ (\mathcal{J}_1, \mathcal{J}_2)$ will be used as notation for for $\mathcal{J}\ (\mathcal{J}_2)\ (\mathcal{J}_1)$.

2.2. General judgement. Another kind of higher order judgement is "general judgement", which is judgement with respect to a variable, $\boxed{|_x\, \mathcal{J}}$, pronounced "for an arbitrary x, $\mathcal{J}$". The meaning explanation for this new judgement is as follows:

> To know the judgement $|_x\, \mathcal{J}$ is, to know $[E/x]\mathcal{J}$ (i.e. the substitution of E for x in the expression $\mathcal{J}$) for any arbitrary expression E, [2]

As far as notation is concerned, the bar symbol binds the least tightly of all the other notations we have considered. Likewise, general judgement may be iterated, and the notation $|_{x,y}\, \mathcal{J}$ will be used as notation for $|_x\, |_y\, \mathcal{J}$.

[2] Technically, E is qualified as being of the same valence as x, but because we have not developed a formal theory of expressions in this presentation, I choose to ignore this issue.

2.3. Hypothetico-general judgement. When hypothetical judgement is used inside general judgement, as in $|_x\, A(x)\ \mathit{true}\ (B(x)\ \mathit{true})$, we term the whole thing a "hypothetico-general" judgement. One thing bears clarifying, which is, Why do we write $P\ \mathit{true}\ (P\ \mathit{true})$ rather than $|_P\, P\ \mathit{true}\ (P\ \mathit{true})$?

The former is really not a single judgement, but rather a *scheme* for judgements, where P is intended to be replaced with a concrete expression by the person asserting the judgements. On the other hand, the latter is itself a single judgement which may be asserted all on its own.

3. Propositions and verifications

Now that we have propounded and explained the minimal system of judgements for a logical theory, let us populate it with propositions. First, we have falsity $\bot$, and we wish to make $\bot\ \mathit{prop}$ evident; to do this, we simply state what counts as a direct verification of $\bot$: there is no direct verification of $\bot$.

The next basic proposition is trivial truth $\top$, and to make $\top\ \mathit{prop}$ evident, we state that a direct verification of $\top$ is trivial. The definition of $\top$ thus validates the judgement $\top\ \mathit{true}$ (i.e. that we have a verification of $\top$; this is immediate).

Next, let us define conjunction; in doing so, we will make evident the hypothetical judgement $P \wedge Q\ \mathit{prop}\ (P\ \mathit{prop}, Q\ \mathit{prop})$; equivalently, we can display this as a rule of inference:[3]

$$\frac{P\ \mathit{prop} \quad Q\ \mathit{prop}}{P \wedge Q\ \mathit{prop}}$$

A direct verification of $P \wedge Q$ consists in a verification of P and a verification of Q; this validates the assertion of the judgement $P \wedge Q\ \mathit{true}\ (P\ \mathit{true}, Q\ \mathit{true})$. Because it is a valid inference, we can write it as an inference rule:

$$\frac{P\ \mathit{true} \quad Q\ \mathit{true}}{P \wedge Q\ \mathit{true}}$$

A direct verification of $P \vee Q$ may be got either from a verification of P or one of Q. From this definition we know $P \vee Q\ \mathit{prop}\ (P\ \mathit{prop}, Q\ \mathit{prop})$, or

$$\frac{P\ \mathit{prop} \quad Q\ \mathit{prop}}{P \vee Q\ \mathit{prop}}$$

[3]Evident hypothetical judgements are often written as rules, i.e.

$$\frac{\mathit{premise}}{\mathit{conclusion}}$$

rather than *conclusion* (*premise*). It must be stressed that only *evident*/*known* judgements may be written in this way.

The verification conditions of disjunction give rise to two evident judgements $P \vee Q$ *true* (P *true*) and $P \vee Q$ *true* (Q *true*), which we can write as inference rules:

$$\frac{P \ \textit{true}}{P \vee Q \ \textit{true}} \qquad \frac{Q \ \textit{true}}{P \vee Q \ \textit{true}}$$

Finally, we must define the circumstances under which $P \supset Q$ is a proposition (i.e. when $P \supset Q$ *prop* is evident). And we intend this to be under the circumstances that P is a proposition, and also that Q is a proposition assuming that P is true. In other words, $P \supset Q$ *prop* (P *prop*, Q *prop* (P *true*)), or

$$\frac{P \ \textit{prop} \quad Q \ \textit{prop} \ (P \ \textit{true})}{P \supset Q \ \textit{prop}}$$

Now, to validate this judgement will be a bit more complicated than the previous ones. But by unfolding the meaning explanations for hypothetical judgement, proposition-hood and truth of a proposition, we arrive at the following explanation:

> To know $P \supset Q$ *prop* (P *prop*, Q *prop* (P *true*)) is to know what counts as a direct verification of $P \supset Q$ when one knows what counts as a direct verification of P, and, when one has such a verification, what counts as a direct verification of Q.[4]

If the judgement $P \supset Q$ *prop* (P *prop*, Q *prop* (P *true*)) is going to be made evident, then we must come up with what should count as a direct verification of $P \supset Q$ under the assumptions described above.

And so to have a direct verification of $P \supset Q$ is to have a verification of Q assuming that one has one of P; this is the meaning of implication, and it validates the judgement $P \supset Q$ *true* (Q *true* (P *true*)), and may be written as an inference rule as follows:

$$\frac{Q \ \textit{true} \ (P \ \textit{true})}{P \supset Q \ \textit{true}}$$

4. Judgements for verifications

So far, we have given judgements which define what it means to be a proposition, namely P *prop*, and thence for each proposition, we have by definition a notion of what should count as a verification of that proposition. And we have a judgement P *true*, which in its assertion means that one has (a way to obtain) such a verification of P, but we have not considered any judgements which actually refer to the verifications themselves symbolically.

It is a hallmark of Martin-Löf's program to resolve the contradiction between syntax and semantics not by choosing symbols over meanings or meanings over

[4]Note that unless P *true*, it need not be evident that Q *prop*; in other words, Q only has to be a proposition if P is true. It would also be acceptable to give a stronger definition to implication, but this is the one accepted by Martin-Löf.

symbols, but by endowing symbols with meaning in the course of knowing the evident judgements. **As such, P is a symbol, but when we assert P *prop* we are saying that we know what proposition P denotes.**

A similar thing can be done with verifications themselves, by representing them with symbols in the same way we have done for the propositions. And then, we can consider a judgement such as "M is a verification of P", and in knowing that judgement, we know what verification M is meant to denote. In practice, this judgement has been written in several ways:

Notation	Pronunciation
$M \in P$	M is an element of P
$M \Vdash P$	M realizes P
$P \llcorner\mathsf{ext}\ M\lrcorner$	P is witnessed by M

But they all mean the same thing, so we will choose the notation $\boxed{M \in P}$ and pronounce it "M verifies P". Tentatively, the following defective meaning explanation could be given:

* To know $M \in P$ is to know that M is a verification of P.

But now that we have started to assign expressions to verifications, we must be more careful about differentiating *direct verifications* (which we will call "canonical") from *indirect verifications* (which we will call "non-canonical"). So the domain of expressions must itself be accorded with a notion of reduction to canonical form, and this corresponds with putting into action a plan of verification in order to get a direct (canonical) verification; reduction to canonical form will be represented by a judgement $\boxed{M \Rightarrow M'}$, pronounced "M evaluates to M'".

To know $M \Rightarrow M'$ is to know that M is an expression which reduces to a canonical form M'.

An example of an evident reduction judgement in elementary mathematics would be $3 + 4 \Rightarrow 7$; note that $3 + 4 \Rightarrow 1 + 6$ is, on the other hand, not evident, since this judgement describes reduction to *canonical* form, whereas $1 + 6$ is not a canonical number.

Now, we can correct the previous meaning explanation as follows:

To know $M \in P$ is to know an M' such that $M \Rightarrow M'$ and M' is a canonical (direct) verification of P.

If it is not yet clear why it would have been a mistake to fail to use the notion of reduction to canonical form in the above meaning explanation, consider that each time a proposition is defined, it should be possible to do so without knowing what other propositions exist in the theory. But if we consider non-canonical forms (as would be necessary if we omitted the $M \Rightarrow M'$ premise), then we would have to fix in advance all the possible non-canonical forms in the computation system in the course of defining each proposition. As such, the open-ended nature of the logic would be destroyed; in a later chapter, the seriousness of this problem will be made even more clear.

The meaning explanation for P *prop* must be accordingly modified to take into account the computational behavior of expressions:

> To know P *prop* is to know a P' such that $P \Rightarrow P'$ and P' is a canonical proposition, which is to say, that one knows what counts as a canonical verification for P'.

In practice, when it is clear that P is canonical, then we will simply say, "To know P *prop* is to know what counts as a canonical verification of P". As an example, then, we will update the evidence of the following assertion:

$$P \supset Q \; \mathit{prop} \; (P \; \mathit{prop}, Q \; \mathit{prop} \; (P \; \mathit{prop}))$$

The meaning of this, expanded into spoken language, is as follows:

> To know $P \supset Q \; \mathit{prop} \; (P \; \mathit{prop}, Q \; \mathit{prop} \; (P \; \mathit{prop}))$ is to know what counts as a canonical (direct) verification of $P \supset Q$ under the circumstances that $P \Rightarrow P'$, such that one knows what counts as a canonical verification P', and, if one has such a verification, $Q \Rightarrow Q'$ such that one knows what counts as a canonical verification of Q'.

And the above judgement is evident, since we will say that a canonical verification of $P \supset Q$ is an expression $\lambda x.E$ such that we know the hypothetico-general judgement $|_x \, E \in Q \; (x \in P)$. This validates the assertion $\lambda x.E \in P \supset Q \; (|_x \, E \in Q \; (x \in P))$, or, written as an inference rule:

$$\frac{|_x \, E \in Q \; (x \in P)}{\lambda x.E \in P \supset Q}$$

By the addition of this judgement, we have graduated from a logical theory to a type theory, in the sense of *Constructive Mathematics and Computer Programming* (Martin-Löf, 1979). In fact, we may dispense with the original P *true* form of judgement by *defining* it in terms of the new $M \in P$ judgement as follows:

$$\frac{M \in P}{P \; \mathit{true}}$$