Chapter 2

CLASSICAL LOGIC—SEMANTICS

1. Classical Models

Defining the semantics of any higher-order logic is relatively complicated. Since modalities add special complexities, it is fortunate I can discuss underlying classical issues before bringing them into the picture. In this Chapter the "real" notion of higher-order model is defined first, and truth in them is characterized. Then Henkin's modification of these models is considered—sometimes these are called general models—as well as a non-extensional version of them.

I don't want just syntactic objects, terms, to have types. I want sets and relations to have them too. After all, we think of terms as designating sets and relations, and we want type information to move back and forth between syntactic object and its designation.

DEFINITION 2.1 (RELATION TYPES) *Let S be a non-empty set. For each type t the collection $[\![t, S]\!]$ is defined as follows.*

1 $[\![0, S]\!] = S$.

2 $[\![\langle t_1, \ldots, t_n\rangle, S]\!]$ *is the collection of all subsets of* $[\![t_1, S]\!] \times \cdots \times [\![t_n, S]\!]$.

O is an object of type t over S *if* $O \in [\![t, S]\!]$. *O is systematically used, with or without subscripts, to stand for objects in this sense.*

For example, a member of $[\![\langle 0, 0\rangle, S]\!]$ is a subset of $S \times S$, and in standard first-order logic it would simply be called a two-place relation on S. But now relations of relations are allowed, and even more complex things as well, so terminology gets more complicated.

A classical model consists of an underlying domain, thought of as the "ground level objects," and an interpretation, assigning some denota-

tion in the model to each constant symbol of the language. But that denotation must be consistent with type information.

DEFINITION 2.2 (CLASSICAL MODEL) *A higher-order* classical model *for $L(C)$ is a structure $\mathcal{M} = \langle \mathcal{D}, \mathcal{I} \rangle$, where $\mathcal{D}$ is a non-empty set called the* domain *of the model, and $\mathcal{I}$ is a mapping, the* interpretation, *meeting the following conditions.*

1 *If A is a constant symbol of $L(C)$ of type t, $\mathcal{I}(A) \in [\![t, \mathcal{D}]\!]$.*

2 *If $=$ is the equality constant symbol of type $\langle t, t \rangle$ then $\mathcal{I}(=)$ is the equality relation on $[\![t, \mathcal{D}]\!]$.*

2. Truth in a Model

Assume $\mathcal{M} = \langle \mathcal{D}, \mathcal{I} \rangle$ is a classical model for a language $L(C)$. It is time to say which sentences of the language, or more generally, which formulas with free variables, are true in $\mathcal{M}$. This is symbolized by $\mathcal{M} \Vdash_v \Phi$. Informally it can be read: the formula Φ is true in the model $\mathcal{M}$, with respect to the valuation v which assigns meanings to free variables. But as will be seen, to properly define this one must also assign denotations to all terms. The denotation of a term of type t will be an object of type t over $\mathcal{D}$. And this can not be done first, independently. The assignment of denotations to terms, and the determination of formula truth constitutes a mutually recursive pair of definitions, just as was the case for the syntactic notions of term and formula in Section 1. Still, it is all rather straightforward.

DEFINITION 2.3 (VALUATION) *The mapping v is a* valuation *in the classical model $\mathcal{M} = \langle \mathcal{D}, \mathcal{I} \rangle$ if v assigns to each variable α^t of type t some object of type t, that is, $v(\alpha^t) \in [\![t, \mathcal{D}]\!]$.*

DEFINITION 2.4 (VARIANT) *A valuation w is an α-*variant *of a valuation v if v and w agree on all variables except possibly α. More generally, w is an $\alpha_1, \ldots, \alpha_n$-*variant *if v and w agree on all variables except possibly $\alpha_1, \ldots, \alpha_n$.*

Now, the following two definitions constitute a single recursive characterization.

DEFINITION 2.5 (DENOTATION OF A TERM) *Let $\mathcal{M} = \langle \mathcal{D}, \mathcal{I} \rangle$ be a classical model, and let v be a valuation in it. A mapping is defined, $(v * \mathcal{I})$, assigning to each term of $L(C)$ a denotation for that term.*

1 *If A is a constant symbol of $L(C)$ then $(v * \mathcal{I})(A) = \mathcal{I}(A)$.*

2 *If* α *is a variable then* $(v * \mathcal{I})(\alpha) = v(\alpha)$.

3 *If* $\langle \lambda\alpha_1, \ldots, \alpha_n.\Phi \rangle$ *is a predicate abstract of* $L(C)$ *of type* t, *then* $(v * \mathcal{I})(\langle \lambda\alpha_1, \ldots, \alpha_n.\Phi \rangle)$ *is the following member of* $[\![t, \mathcal{D}]\!]$:

$$\{\langle w(\alpha_1), \ldots, w(\alpha_n)\rangle \mid w \text{ is an } \alpha_1, \ldots, \alpha_n \text{ variant of } v \text{ and } \mathcal{M} \Vdash_w \Phi\}$$

DEFINITION 2.6 (TRUTH OF A FORMULA) *Again let* $\mathcal{M} = \langle \mathcal{D}, \mathcal{I} \rangle$ *be a classical model, and let* v *be a valuation in it. The notion of formula* Φ *of* $L(C)$ *being true in model* $\mathcal{M}$ *with respect to* v, *denoted* $\mathcal{M} \Vdash_v \Phi$, *is characterized as follows.*

1 *For terms* $\tau, \tau_1, \ldots, \tau_n$, $\mathcal{M} \Vdash_v \tau(\tau_1, \ldots, \tau_n)$ *provided* $\langle (v * \mathcal{I})(\tau_1), \ldots, (v * \mathcal{I})(\tau_n)\rangle \in (v * \mathcal{I})(\tau)$.

2 $\mathcal{M} \Vdash_v \neg\Phi$ *if it is not the case that* $\mathcal{M} \Vdash_v \Phi$.

3 $\mathcal{M} \Vdash_v \Phi \wedge \Psi$ *if* $\mathcal{M} \Vdash_v \Phi$ *and* $\mathcal{M} \Vdash_v \Psi$.

4 $\mathcal{M} \Vdash_v (\forall\alpha)\Phi$ *if* $\mathcal{M} \Vdash_{v'} \Phi$ *for every* α*-variant* v' *of* v.

There is an alternative notation that makes evaluating the truth of formulas in models somewhat easier.

DEFINITION 2.7 (SPECIAL NOTATION) *Suppose* v *is a valuation, and* w *is the* $\alpha_1, \ldots, \alpha_n$ *variant of* v *such that* $w(\alpha_1) = O_1, \ldots, w(\alpha_n) = O_n$. *Then, if* $\mathcal{M} \Vdash_w \Phi$ *this may be symbolized by*

$$\mathcal{M} \Vdash_v \Phi[\alpha_1/O_1, \ldots, \alpha_n/O_n].$$

Now part 3 of Definition 2.5 can be restated as follows.

3 $(v * \mathcal{I})(\langle \lambda\alpha_1, \ldots, \alpha_n.\Phi \rangle) =$
$\{\langle O_1, \ldots, O_n\rangle \mid \mathcal{M} \Vdash_v \Phi[\alpha_1/O_1, \ldots, \alpha_n/O_n]\}$

Likewise part 4 of Definition 2.6 becomes

4 $\mathcal{M} \Vdash_v (\forall\alpha)\Phi$ if $\mathcal{M} \Vdash_v \Phi[\alpha/O]$ for every object O of the same type as α.

Defined symbols like $\supset$ and $\exists$ have their expected behavior, which are explicitly stated below. Alternately, this can be considered an extension of the definition above.

5 $\mathcal{M} \Vdash_v \Phi \vee \Psi$ if $\mathcal{M} \Vdash_v \Phi$ or $\mathcal{M} \Vdash_v \Psi$.

6 $\mathcal{M} \Vdash_v \Phi \supset \Psi$ if $\mathcal{M} \Vdash_v \Phi$ implies $\mathcal{M} \Vdash_v \Psi$.

7 $\mathcal{M} \Vdash_v \Phi \equiv \Psi$ if $\mathcal{M} \Vdash_v \Phi$ iff $\mathcal{M} \Vdash_v \Psi$.

8 $\mathcal{M} \Vdash_v (\exists\alpha)\Phi$ if $\mathcal{M} \Vdash_{v'} \Phi$ for some α-variant v' of v; equivalently if $\mathcal{M} \Vdash_v \Phi[\alpha/O]$ for some object O of the same type as α.

As in first-order logic, if Φ has no free variables, $\mathcal{M} \Vdash_v \Phi$ holds for some v if and only if it holds for every v. Thus for sentences (closed formulas), truth in a model does not depend on a choice of valuation.

DEFINITION 2.8 (VALIDITY, SATISFIABILITY, CONSEQUENCE) *Let Φ be a formula and S be a set of formulas.*

1 Φ *is* valid *if $\mathcal{M} \Vdash_v \Phi$ for every classical model $\mathcal{M}$ and valuation v.*

2 S *is* satisfiable *if there is some model $\mathcal{M}$ and some valuation v such that $\mathcal{M} \Vdash_v \varphi$ for every $\varphi \in S$.*

3 Φ *is a* consequence *of S provided, for every model $\mathcal{M}$ and every valuation v, if $\mathcal{M} \Vdash_v \varphi$ for all $\varphi \in S$, then $\mathcal{M} \Vdash_v \Phi$.*

The definitions above are of some complexity. Here is an example to help clarify their workings.

EXAMPLE 2.9 This example shows a formula that is valid and involves equality. In it, c is a constant symbol of type 0.

The expression $\langle\lambda X.(\exists x)X(x)\rangle$ is a predicate abstract of type $\langle\langle 0\rangle\rangle$, where X is of type $\langle 0\rangle$ and x is of type 0. Intuitively it is the "being instantiated" predicate. Likewise the expression $\langle\lambda x.x = c\rangle$ is a predicate abstract of type $\langle 0\rangle$, where x and c are of type 0. Intuitively this is the "being c" predicate. Since this predicate is, in fact, instantiated (by whatever c designates), the first predicate abstract correctly applies to it. That is, one should have the validity of the following.

$$\langle\lambda X.(\exists x)X(x)\rangle(\langle\lambda x.x = c\rangle) \tag{2.1}$$

I now verify this validity. Suppose there is a model $\mathcal{M} = \langle\mathcal{D}, \mathcal{I}\rangle$. I show the formula is true in $\mathcal{M}$ with respect to an arbitrary valuation v. To do this, I investigate the behavior, in $\mathcal{M}$, of parts of the formula, building up to the whole thing.

First, recalling that the interpretation of an equality symbol is by the equality relation of the appropriate type, we have the following.

$$\begin{aligned}(v * \mathcal{I})(\langle\lambda x.x = c\rangle) &= \{o \mid \mathcal{M} \Vdash_v (x = c)[x/o]\} \\ &= \{o \mid o = \mathcal{I}(c)\} \\ &= \{\mathcal{I}(c)\}\end{aligned}$$

We also have the following.

$$\begin{aligned}(v * \mathcal{I})(\langle \lambda X.(\exists x)X(x)\rangle) &= \{O \mid \mathcal{M} \Vdash_v (\exists x)X(x)[X/O]\} \\ &= \{O \mid \mathcal{M} \Vdash_v X(x)[X/O, x/o] \text{ for some } o\} \\ &= \{O \mid o \in O \text{ for some } o\} \\ &= \{O \mid O \neq \emptyset\}\end{aligned}$$

Now we have (2.1) because

$$\mathcal{M} \Vdash_v \langle \lambda X.(\exists x)X(x)\rangle(\langle \lambda x.x = c\rangle) \Leftrightarrow$$
$$(v * \mathcal{I})(\langle \lambda x.x = c\rangle) \in (v * \mathcal{I})(\langle \lambda X.(\exists x)X(x)\rangle) \Leftrightarrow \{\mathcal{I}(c)\} \in \{O \mid O \neq \emptyset\}.$$

You might try verifying, in a similar way, the validity of the following.

$$\neg\langle \lambda X.(\exists x)X(x)\rangle(\langle \lambda x.\neg(x = x)\rangle)$$

3. Problems

First-order classical logic has many nice features that do not carry over to higher-order versions. This is well-known, and partly accounts for the general emphasis on first-order. I sketch a few of the higher-order problems here.

3.1 Compactness

The compactness theorem for first-order logic says a set of formulas is satisfiable if every finite subset is. This is a fundamental tool for the construction of models of various kinds—non-standard models of analysis, for instance. The higher-order analog does not hold, and counter-examples are easy to come by. Here is one.

The Dedekind characterization of infinity is: a set is infinite if it can be put into a 1-1 correspondence with a proper subset. Consequently, a set is finite if any 1-1 mapping from it to itself can not be to a proper subset, i.e. must be onto. This can be said easily, as a second-order formula. Since function symbols are not available, I make do with relation symbols in the usual way—the following formula is true in a model if and only if the domain of the model is finite.

$$(\forall X)[(\mathsf{function}(X) \wedge \mathsf{one\text{-}one}(X)) \supset \mathsf{onto}(X)] \tag{2.2}$$

In (2.2) the following abbreviations are used.

$$\begin{array}{rcc} \mathsf{function}(X) & \text{for} & (\forall x)(\exists y)(\forall z)[X(x,z) \equiv (z = y)] \\ \mathsf{one\text{-}one}(X) & \text{for} & (\forall x)(\forall y)(\forall z)\{[X(x,z) \wedge X(y,z)] \supset (x = y)\} \\ \mathsf{onto}(X) & \text{for} & (\forall y)(\exists x)X(x,y) \end{array}$$

http://www.springer.com/978-1-4020-0604-3

Types, Tableaus, and Gödel's God
Fitting, M.
2002, XV, 181 p., Hardcover
ISBN: 978-1-4020-0604-3