syn.1 Structures for First-order Languages

First-order languages are, by themselves, uninterpreted: the constant symbols, function symbols, and predicate symbols have no specific meaning attached to them. Meanings are given by specifying a structure. It specifies the domain, i.e., the objects which the constant symbols pick out, the function symbols operate on, and the quantifiers range over. In addition, it specifies which constant symbols pick out which objects, how a function symbol maps objects to objects, and which objects the predicate symbols apply to. Structures are the basis for semantic notions in logic, e.g., the notion of consequence, validity, satisfiability. They are variously called “structures,” “interpretations,” or “models” in the literature.

Definition syn.1 (Structures). A structure \( \mathcal{M} \), for a language \( \mathcal{L} \) of first-order logic consists of the following elements:

1. **Domain:** a non-empty set, \( |\mathcal{M}| \)
2. **Interpretation of constant symbols:** for each constant symbol \( c \) of \( \mathcal{L} \), an element \( c^{\mathcal{M}} \in |\mathcal{M}| \)
3. **Interpretation of predicate symbols:** for each \( n \)-place predicate symbol \( R \) of \( \mathcal{L} \) (other than \( = \)), an \( n \)-place relation \( R^{\mathcal{M}} \subseteq |\mathcal{M}|^n \)
4. **Interpretation of function symbols:** for each \( n \)-place function symbol \( f \) of \( \mathcal{L} \), an \( n \)-place function \( f^{\mathcal{M}} : |\mathcal{M}|^n \to |\mathcal{M}| \)

Example syn.2. A structure \( \mathcal{M} \) for the language of arithmetic consists of a set, an element of \( |\mathcal{M}| \), \( 0^{\mathcal{M}} \), as interpretation of the constant symbol \( 0 \), a one-place function \( s^{\mathcal{M}} : |\mathcal{M}| \to |\mathcal{M}| \), two two-place functions \( +^{\mathcal{M}} \) and \( \times^{\mathcal{M}} \), both \( |\mathcal{M}|^2 \to |\mathcal{M}| \), and a two-place relation \( <^{\mathcal{M}} \subseteq |\mathcal{M}|^2 \).

An obvious example of such a structure is the following:

1. \( |\mathcal{M}| = \mathbb{N} \)
2. \( 0^{\mathcal{M}} = 0 \)
3. \( s^{\mathcal{M}}(n) = n + 1 \) for all \( n \in \mathbb{N} \)
4. \( +^{\mathcal{M}}(n, m) = n + m \) for all \( n, m \in \mathbb{N} \)
5. \( \times^{\mathcal{M}}(n, m) = n \cdot m \) for all \( n, m \in \mathbb{N} \)
6. \( <^{\mathcal{M}} = \{ (n, m) : n \in \mathbb{N}, m \in \mathbb{N}, n < m \} \)

The structure \( \mathcal{M} \) for \( \mathcal{L}_A \) so defined is called the standard model of arithmetic, because it interprets the non-logical constants of \( \mathcal{L}_A \) exactly how you would expect.

However, there are many other possible structures for \( \mathcal{L}_A \). For instance, we might take as the domain the set \( \mathbb{Z} \) of integers instead of \( \mathbb{N} \), and define the interpretations of \( 0, 1, +, \times, < \) accordingly. But we can also define structures for \( \mathcal{L}_A \) which have nothing even remotely to do with numbers.
Example syn.3. A structure \( \mathfrak{M} \) for the language \( \mathcal{L}_Z \) of set theory requires just a set and a single-two place relation. So technically, e.g., the set of people plus the relation “x is older than y” could be used as a structure for \( \mathcal{L}_Z \), as well as \( \mathbb{N} \) together with \( n \geq m \) for \( n, m \in \mathbb{N} \).

A particularly interesting structure for \( \mathcal{L}_Z \) in which the elements of the domain are actually sets, and the interpretation of \( \in \) actually is the relation “x is an element of y” is the structure \( \mathfrak{HF} \) of hereditarily finite sets:

1. \( |\mathfrak{HF}| = \emptyset \cup \wp(\emptyset) \cup \wp(\wp(\emptyset)) \cup \wp(\wp(\wp(\emptyset))) \cup \ldots \);  
2. \( \in_{\mathfrak{HF}} = \{(x, y) : x, y \in |\mathfrak{HF}|, x \in y\} \).

digression

The stipulations we make as to what counts as a structure impact our logic. For example, the choice to prevent empty domains ensures, given the usual account of satisfaction (or truth) for quantified sentences, that \( \exists x (\varphi(x) \lor \neg \varphi(x)) \) is valid—that is, a logical truth. And the stipulation that all constant symbols must refer to an object in the domain ensures that the existential generalization is a sound pattern of inference: \( \varphi(a) \), therefore \( \exists x \varphi(x) \). If we allowed names to refer outside the domain, or to not refer, then we would be on our way to a free logic, in which existential generalization requires an additional premise: \( \varphi(a) \) and \( \exists x x = a \), therefore \( \exists x \varphi(x) \).

Photo Credits

Bibliography