ARE RULES VALID IN VIRTUE OF MEANING OR IS MEANING DETERMINED BY RULES?

PER MARTIN-LÖF

NOTE. This lecture was given by Per Martin-Löf on 21 June 2002 as an invited lecture at Logica '02, held at the Zahrádky Castle in Czechia. The transcription, made by Ansten Klev, is based on a video recording preserved on a CD-ROM that was discovered in the winter of 2025. The video recording was made by David Göttlich.

It is a traditional view, going back to Plato and Aristotle, that axioms and rules of inference are valid in virtue of the concepts they contain, or in a more modern way of expressing it, in terms of the meaning of the terms that we bring into their formulation. The view is roughly as follows. We have the capacity for grasping concepts, the nous. With our nous, we grasp concepts, and with the same capacity, also the axioms, particularly the axioms of mathematics, become evident, namely evident in virtue of how the concepts that they contain have been defined. From the axioms the discursive faculty takes over, the *dianoia*, and then it is the business of the mathematicians: that is precisely what the mathematicians are good at, deriving consequences from the axioms. But the justification of the axioms was not thought to be the business of the mathematicians. Rather, the justification of the axioms belonged to dialectics, so it was a properly philosophical task.

This traditional view is to be contrasted with the modern view, and I think it is no doubt that it is Wittgenstein who is to be associated, above all, with this modern view. We could call it the functionalist view or operational view, to use two of the terms that were very popular around 1930, when this modern view broke through. The best expression that I have found of this modern view is not from Wittgenstein himself, but from Moore's reports of Wittgenstein's lectures of 1930–33. He says that, according to Wittgenstein, the meaning of any single word in a language is defined, constituted, determined, or fixed—Wittgenstein used all four expressions in different places—by the grammatical rules with which it is used in that language, and the meaning of a word is its place in a grammatical system. Alberto Coffa, in his stimulating book *The Semantic Tradition from Kant to Carnap*, has referred to this transition, from the traditional view to the Wittgensteinian view, as a new Copernican turn, or in a second place, as the Copernican turn in semantics, alluding, of course, to Kant.

The contrast that I have presented here is, of course, strongly correlated to the contrast between two different conceptions of meaning, namely the conception of meaning as object versus the conception of meaning as use. The traditional view is correlated with the conception of meaning as objects. We have concepts, in the

traditional terminology, or meanings, in modern terminology, which are objects of a conceptual realm—world of ideas, if you want—and then we have words which express those concepts. The word gets its meaning by being tied to the concept that it expresses. This is the essentially Platonic view. A paradigmatic example of this view is Bolzano's, in the 19th century.

Wittgenstein's contrasting view is that meaning is use. Meaning as use is, no doubt, the most well-known formulation of the later Wittgenstein's view, but I would like to be more precise here. Use is always use for a certain purpose, so I would like to rephrase meaning as use in the following way. Each linguistic unit, linguistic atom, has a very particular purpose, and the meaning of that linguistic unit is the way in which it fulfils its purpose: meaning is mode of operation, or mode of functioning, for the particular purpose that it has. Explaining what the purpose of a linguistic unit is, is to explain what its syntactic category, or grammatical category, is. But there is not only the category: there is also the object, and the object—the meaning—is characterized by the way in which it fulfils the purpose.

These are two different conceptions of meaning—no doubt about that. But it would be very strange if we could not make good sense of both, because, after all, almost all the history has been tied to the old-fashioned view of meaning as object, whereas meaning as mode of operation is a relatively new insight. The way in which these two views can be reconciled is this, and this is something which underlies all of my own work: the meaning in the objective sense, the meaning as object, is simply identified with the full, that is to say, the meaningful expression, to be contrasted with the empty expression, the expression divested of sense. This distinction between the empty expression, its pure shape or form, as contrasted with the full, or meaningful, expression has been made, perhaps for the first time, by Husserl: Leerausdruck contra voller Ausdruck. So the way to reconcile these views is simply to identify the meaning as object with the full expression, which is to say in functionalist terms, the functioning expression, or in terms of use, the expression in its use, or the expression considered together with its mode of employment.

Now I would like throw some light on the original question: do we have meanings first, and then rules valid in virtue of these meanings, or is it the other way around, that it is the system of rules that is the starting point, and then the meanings somehow come out of the system of rules? To this end, I will have to sketch, outline in some way, the informal semantics, or meaning theory, that I have developed for constructive type theory. I am using the term informal semantics, or meaning theory, rather than just semantics, since the term semantics has been so much tied to Tarski semantics, or formal semantics, that we need some word that contrasts sufficiently well with it. Since formal semantics has been a popular term from Carnap and onward, why not informal semantics, or intuitive semantics, or if you want, meaning theory, which is the term generally preferred by Dummett.

What does this meaning theory look like? I will first have to specify the main ingredients of the system as such, and as with every deductive system, you have to start by displaying the forms of judgement on which it is based. In this case, the

forms of judgement are the following—there are four of them:

$$A : \text{type} \quad (x_1 : A_1, \ldots, x_n : A_n) \tag{1}$$

$$A = B : \text{type} \quad (x_1 : A_1, \ldots, x_n : A_n) \tag{2}$$

$$a : A \quad (x_1 : A_1, \ldots, x_n : A_n) \tag{3}$$

$$a = b : A \quad (x_1 : A_1, \ldots, x_n : A_n) \tag{4}$$

The first one says that A is a type in a context formed by a variable x_1 ranging over a previously introduced type A_1, et cetera, x_n ranging over a previously introduced type A_n. Then (2) is the corresponding equality judgement, which says that A and B are equal types in a context of that form. The third form of judgement says that a is an object of type A, again, in general, in such a context, and (4) that a and b are equal objects of type A in such a context.

These are the four forms of judgement, and they have certain presuppositions, which are almost obvious. If you read (1) as saying that A is a type provided that x_1 ranges over the type A_1, et cetera, of course this judgement presupposes that A_1 is a type, and then that A_2 is a type which may possibly depend on A_1, and so on, that A_n is a type, which may possibly depend on the types A_1 up to A_{n-1}. For the second form of judgement, it is clear that I cannot say that A and B are equal types unless they are types to begin with, so this second form of judgement presupposes the first form of judgement for both A and B. When we come to (3), if we say that a is an object of type A, of course it is presupposed that A is a type in the given context, which means that we are presupposing (1). Finally, if we say that a and b are equal objects of type A in this context, then, first of all, we are presupposing that A is a type in this context, and then that both a and b are objects of type A in that context.

Once the forms of judgement have been displayed, the next thing you have to do is to explain what they mean. This proceeds by induction on the length, n, of the context. When $n = 0$ we have a judgement of the form $A : \text{type}$. What must you know in order to have the right to make the judgement $A : \text{type}$? Well, you must know what an object of the type A is—that is the criterion of application—and also what it means for objects of the type A to be equal. That explains the notion of type.

Now we have to deal with the case $n > 0$, when the context is non-empty. Then a judgement of the form (1) means that if

$$a_1 : A_1, \ldots, a_n : A_n(a_1/x_1, \ldots, a_{n-1}/x_{n-1})$$

then

$$A(a_1/x_1, \ldots, a_n/x_n) : \text{type}$$

Since I have already explained what a type is, this explanation makes perfectly good sense. In addition it must also be the case that if

$$a_1 = b_1 : A_1, \ldots, a_n = b_n : A_n(a_1/x_1, \ldots, a_{n-1}/x_{n-1})$$

then

$$A(a_1/x_1, \ldots, a_n/x_n) = A(b_1/x_1, \ldots, b_n/x_n) : \text{type}$$

I will in general try to avoid stressing all these equality rules in the following.

It is clear from this explanation of what a family of types is that the following rule is valid:

$$\frac{A : \text{type } (x_1 : A_1, \ldots, x_n : A_n) \qquad a_1 : A_1, \ldots, a_n : A_n(a_1/x_1, \ldots, a_{n-1}/x_{n-1})}{A(a_1/x_1, \ldots, a_n/x_n) : \text{type}}$$

Why is the rule valid? Because the first premiss means precisely

$$A(a_1/x_1, \ldots, a_n/x_n) : \text{type}$$

provided

$$a_1 : A_1, \ldots, a_n : A_n(a_1/x_1, \ldots, a_{n-1}/x_{n-1})$$

You see that I started in the traditional way by first explaining what a family of types is and then justifying the rule in terms of that explanation. I could, however, equally well have written up the rule first and then said that you can read the meaning of (1) off the rule, namely that you charge (1) with the meaning that is necessary to ensure the validity of the rule. In that sense, you can say that the meaning of the form of judgement (1) is determined by this particular rule together with the corresponding equality rule: they are the meaning determining rules for the form of judgement (1).

When we come to the second form of judgement, we first take the case when $n = 0$, that is, we explain what it means for two types to be equal. What does it mean for two types to be equal? They are equal if an object of the one type is also an object of the other type, and equal objects of the one type are equal objects of the other type, and vice versa. It is then obvious that the following rule is valid:

$$\frac{a : A \qquad A = B : \text{type}}{a : B}$$

This is valid because I have charged the judgement $A = B : \text{type}$ with the meaning necessary to ensure the validity of the rule. Ryle has a nice way of expressing this. In his terminology, I could say that the judgement $A = B : \text{type}$ is an inference licence, or an inference ticket, which licenses you to infer $a : B$ from $a : A$.

Then we come to the third form of judgement. When $n = 0$ we already know what a judgement of the form (3) means, since such a judgement has the presupposition that A is a type, and once it is a type, we know what an object of that type is. It depends, of course, on the type what it is, but it must be fixed what an object of the type A is, and $a : A$ says precisely that a is an object of type A, and we know what that means because of the presupposition.

In the case $n > 0$, when we have variables, then a judgement of the form (3) means that if

$$a_1 : A_1, \ldots, a_n : A_n(a_1/x_1, \ldots, a_{n-1}/x_{n-1})$$

then

$$a(a_1/x_1, \ldots, a_n/x_n) : A(a_1/x_1, \ldots, a_n/x_n) : \text{type}$$

Again, either I can phrase it in this traditional way, explaining the meaning first, and then the rule becomes valid, or I can say that this rule is meaning determining for the form of judgement (3). The treatment of the fourth form of judgement is similar.

These four forms of judgement are forms of analytical judgement. This is reflected in the fact that it can be checked by a computer whether a judgement of any of these forms is correct or not. This is the type-checking algorithm that underlies all computer systems based on type theory that are running at the moment. They can be checked because all the information that we need in order to convince ourselves of such a judgement is there, in it. No information is missing that we might have to search for. That is why it is appropriate to call them analytic. They are valid entirely in virtue of the meanings of the constants that occur in them.

In a synthetic judgement, by contrast, you do not say that a is an object of type A, but just that A has existence,

$$A \text{ exists},$$

which means, of course, that there is an object of that type. (I am pleased by the fact that Bolzano was the first to consider this combination.) A judgement of this form can be made in a context that consists, not only of variable declarations, but also of assumptions,

$$\text{(Exists)} \qquad A \text{ exists } (x_1 : A_1, \ldots, x_n : A_n, A_{n+1} \text{ exists}, \ldots, A_m \text{ exists})$$

What does such a judgement mean? A judgement of this form is just an abbreviated way of saying that we have constructed a function

$$a : A \ \ (x_1 : A_1, \ldots, x_n : A_n, x_{n+1} : A_{n+1}, \ldots, x_m : A_m)$$

The object a may depend on all of these variables, $x_1, \ldots, x_m$, not only the first ones, $x_1, \ldots, x_n$. If nothing depends on the variables from x_{n+1} onwards, which means that none of the types $A_{n+1}, \ldots, A_m$, nor A, depends on the preceding variables, then you can abbreviate the latter judgement by (Exists). You are, so to say, fed up by writing out the construction a: you are just telling that there is such a construction. It is an incomplete judgement, or a judgement abstract, in a terminology that was used in the Hilbert school in the 1920s, and which Kleene changed to incomplete communcation. A judgement of the form (Exists) is an incomplete communication that the object a has been found. The judgement (Exists) is properly called synthetic because, of course, you cannot decide whether it is correct or not. The a has to be found, it has to be constructed, and that requires inventiveness.

This finishes the explanations of the basic forms of judgement, and now we come to at least some rules. Let us begin with the rules of type formation. The rules of type formation are a generalization of Church's rules of type formation in his "A formulation of the simple theory of types" from 1940. We have as the first axiom

$$\text{set} : \text{type}$$

Now I will immediately have to say something about sets. I am going to use the word set, not in the sense of Zermelo–Fraenkel set theory—that is rather an iterative set—but I am going to use set for quantificational domain, or individual domain. I am, moreover, going to take for granted—now that 33 years have passed—the Curry–Howard correspondence, or Curry–Howard isomorphism, between propositions and sets: propositions and sets are simply identified,

$$\text{prop} = \text{set} : \text{type}$$

This means that the Cartesian product operation on sets is identified with the conjunction operation on propositions, the disjoint union operation is identified with disjunction, and the Cartesian product of a family of sets is identified with the universal quantifier, et cetera. I assume that most you know about this very important idea, which is basic to all present versions of constructive type theory.

This is the first rule of type formation. The second rule is

$$\frac{A : \text{set}}{\text{element}(A) : \text{type}} \qquad \frac{A : \text{prop}}{\text{proof}(A) : \text{type}}$$

If we have a set, or proposition, A, then we may form the type, element(A), of its elements, or if we use the logical reading, it will be the type, proof(A), of proofs of the proposition A. It is too tedious to write out element(A) or proof(A), so in practice one writes A here: any set is also a type.

Finally we have the rule for forming function types.

$$\frac{A : \text{type} \qquad \begin{matrix} x : A \\ B : \text{type} \end{matrix}}{(x : A)B : \text{type}}$$

If A is a type, and if B is a type depending on a variable x ranging over A—I am using natural deduction style—then we may form the dependent function type $(x : A)B$.

In what way does this generalize Church's rules? Well, the type prop, of propositions, is Church's o. Church's other ground type was the type ι of individuals. The characteristic thing of constructive type theory is that we have many, many individual domains, and they are built up in the theory itself. To compare with Church, you could call element(A), if you want, $\iota(A)$, that is, for each set A, there is a type of individuals $\iota(A)$, namely the type of elements belonging to that set. Because of this rule, you will have types depending on variables, and since B is allowed to depend on a variable x ranging over A, the function type $(x : A)B$ is called the dependent function type. This contains Church's function type as a special case. In Schütte's notation, Church's rule of function type formation is

$$\frac{A : \text{type} \qquad B : \text{type}}{(A)B : \text{type}}$$

When B depends vacuously on A, you may define $(A)B$ as $(x : A)B$. So Church's type structure is embedded in this dependent type structure.

Now I am a bit ahead with the formalities—ahead of the semantics, and this is always the case: you have to write up something in order to have something to

explain. According to my official explanation of what a type is, corresponding to the axiom set : type there has to be an explanation of what a set is, as well as an explanation of what it is for two sets to be equal.

What is a set? A set, on this conception, is defined by laying down the rules for forming its primitive, or canonical, elements, as well as the rules for forming equal primitive, or canonical, elements—I will forget about the equality part here, as usual. I am using both expressions: primitive and canonical,

primitive	defined
= canonical	= non-canonical

This distinction is necessary, because, if you think of the natural numbers, for instance, not only 0, s(0), et cetera, are natural numbers, but also $2 + 2$ and 10^{10}, which are not directly generated by the first two Peano axioms. They are natural numbers because they evaluate either to 0 or to something of successor form.

So the natural numbers is taken to be defined by the first two Peano axioms,

$$0 : \mathbb{N} \qquad \frac{n : \mathbb{N}}{\mathrm{s}(n) : \mathbb{N}}$$

The Cartesian product of two sets, A and B, is taken to be defined by the rule

$$\frac{a : A \qquad b : B}{\langle a, b\rangle : A \times B}$$

If a is an element of A and b is an element of B, then $\langle a, b\rangle$ is a canonical element of $A \times B$—that defines the Cartesian product. If we call any such rule an introduction rule for the set-forming operation in question, or under the logical interpretation, the logical operation in question, then you see that Getnzen's idea, contained in a single sentence in his 1934 paper, that the logical constants are, so to speak, defined by their introduction rules, is contained in the meaning theory here and generalized from propositions to sets: a set is defined by its introduction rules.

What I have said here is correct, strictly speaking, only of primitive, or canonical sets. There may also be defined, or non-canonical sets, and there the explanation is easier: a defined, or non-canonical set, is a method, or program which yields as result a canonical set.

This explains what a set is. I also have to lay down the equality criterion: what does it mean for two sets to be equal? I could say, just as I said for types before, that two sets are equal just in case an element of the one set is also an element of the other, and vice versa, and similarly for equal elements, namely that equal elements of the one set are also equal elements of the other set, and vice versa. Then the following rule is valid:

$$\frac{A = B : \mathrm{set}}{\mathrm{element}(A) = \mathrm{element}(B) : \mathrm{type}}$$

But I could equally well say that this rule is meaning determining for the form of judgement $A = B$: set, or that a judgement of this form is a license to infer $\mathrm{element}(A) = \mathrm{element}(B)$: type.

Now I have explained the axiom set : type. Let me drop the explanation of what an element of a set is, because that is obvious from the explanation of what a set

is, but let us look at the rule for forming function types,

$$\frac{A : \text{type} \qquad \begin{array}{c} x : A \\ B : \text{type} \end{array}}{(x : A)B : \text{type}}$$

Suppose we know the premisses here. What do we have to know in order to have the right to make the judgement $(x : A)B : \text{type}$? We must know what an object of that type is as well as what it is for two objects of that type to be equal.

What is an object of type $(x : A)B$? Well, it is something, f, which when applied to an object, a, of type A, yields as result an object, $f(a)$, of type $B[a/x]$, and when applied to equal objects, a and b, of type A, yields as result equal objects, $f(a)$ and $f(b)$, of type $B[a/x]$. With that explanation, it is clear that these two rules are valid:

$$\frac{f : (x : A)B \qquad a : A}{f(a) : B[a/x]} \qquad\qquad \frac{f : (x : A)B \qquad a = b : A}{f(a) = f(b) : B[a/x]}$$

Again, I could equally well write up the rules first, and then say that these rules are meaning determining for the form of judgement $f : (x : A)B$.

I also have to give the criterion of identity—I have to explain when f is equal to g. That they are equal objects of type $(x : A)B$ means that, when applied to an object $a : A$, they yield equal results, $f(a)$ and $g(a)$, of type $B[a/x]$. The following rule is therefore valid:

$$\frac{f = g : (x : A)B \qquad a : A}{f(a) = g(a) : B[a/x]}$$

But I could also say that this rule is meaning determining for the form of judgement $f = g : (x : A)B$, and therefore the rule is valid by fiat.

Owing to time limitations, I will not be able to give more rules here. After these general rules, what you come to is the axiom that $\mathbb{N}$ is a set, namely the set of natural numbers, and you give the introduction rules for it, which are the first two Peano axioms, and then you give the corresponding elimination rule, which introduces the recursion operator, if you take the set-theoretic interpretation, or in the logical interpretation, the principle of mathematical induction. This recursion operator is defined by its recursion equations, which say what you get when you put in the argument 0 and what you get when you put in an argument of successor form. Now Gentzen's dictum that a proposition is defined by its introduction rule is complemented by the principle that an eliminatory operation, an eliminator, if you want, is defined by its associated equality rule, or computation rule. All the other meaning explanations that I have given previously here are also not contained in the original Gentzen explanation, which was concerned only with what a proposition is.

Instead of giving more rules now, let me return to the original question: are rules valid in virtue or meaning, or is meaning determined by rules? This is a question about the order of priority between meanings and rules: is it meanings first, and then rules valid in virtue of them, or is it the other way around? After what I have said here, I hope it is not difficult for you to see that there is no conflict between these two views that I started by contrasting. Are rules valid in virtue of meaning?

Of course, yes, all the time here you see that rules are valid because of the way I have explained the meaning of the terms involved. Is meaning determined by rules? Yes, all the time. I can rephrase the meaning explanation for a form of judgement as saying that its meaning is determined by a particular rule.

So there is no conflict between these two views. They are fully compatible with each other. Under more extreme interpretations, to the one side or the other side, they are incompatible. If you take the first question, which was meant to refer to the traditional view—there is an extreme form of the traditional view, namely that we have the realm of concepts, or meanings, which is there first, and then we come with language and give meaning to our words by associating each word with a pre-existing concept. That is the Platonic view. This view has a well-known problem, namely if we are to know the meanings first, then how do we get to know them? We are supposed to grasp meanings with our nous—that is fine enough, but how do we get to grasp them? This problem is not, to my mind, to be solved in a comprehensible way by referring to innate ideas or anything like that. That is not a comprehensible position.

So, how do we get into this conceptual realm, say in the case of mathematics? We are not born with these type-theoretical ideas, so how do we get into this world? Clearly, by training. We train ourselves in using this language with all its rules, and once we have mastered the language, we know these concepts.

It is the virtue of the second position that it makes it comprehensible how we get to know concepts. Does not this speak in favour of the second position, that we have the system of rules first, the language first, and then meaning is determined by rules? One would like to say that, but as a matter of fact, this very powerful idea—functionalist, or operationalist, if you want—about meanings, originating above all from Wittgenstein, unfortunately leads almost immediately to conventionalism. You might interpret the dictum that meaning is determined by rules in the conventionalist fashion that we just write up any system of rules—that fixes the game, and then the meanings of the symbols are just the way that they are moved in that game. This is outspokenly so in Carnap's *Logische Syntax der Sprache*, in the section on conventionalism. We are free to fix any system of rules we want, and the symbols get their meanings from the way they are moved in this system.

We must remember that conventionalism was inspired by mathematical formalism, hence when we apply conventionalism to mathematics, we obtain just formalism. But that is not a good philosophy of mathematics, and we can see now what is wrong. There is something right in the idea that meaning is determined by rules. It is, however, not determined in this facile way, that we just write down any system of rules that we want and automatically get meaning. Instead, for each constant, we have to point out precisely which rule is meaning determining for it. That is what I have done here for some of the constants of constructive type theory.

The analogy that was used at the time when conventionalism, or formalism, broke through, the comparison with chess, is good for explaining the formal character of the system: you give precise rules, and then you can move the pieces. It is, however, not a good analogy to explain the semantical side of language. A much better

comparison is a complicated machinery, or a complicated system. This machinery consists of many little parts, each of which has a very specific purpose, and fulfils that purpose in a particular way. Explaining the way in which it fulfils its purpose is, precisely, explaining its meaning.

In a more traditionally philosophical terminology, one could say that the conceptual realm, the world of ideas, the *kosmos noētos*, is a system of interrelated or interlocking parts, and what a meaning theory does is to explain in teleological terms—it is purpose that is involved here—how that system works. Hence you may say that syntax and semantics, done in this way, give a theory of constitution for the conceptual realm.