Philosophical Aspects of Intuitionistic Type Theory

Per Martin-Löf

Lectures given at the Faculteit der Wijsbegeerte, Rijksuniversiteit Leiden,
23 September – 16 December 1993

Editorial note

This book consists of an almost complete transcription of twelve lectures given by Per Martin-Löf at Leiden University in the autumn of 1993. Martin-Löf had been invited to Leiden by Göran Sundholm, who also arranged for the lectures to be recorded and transcribed. The original transcription, including a table of contents, was prepared by Martien Wijers. The current version has been edited and transferred to LaTeX by Ansten Klev.

Contents

Lecture 1, 23.09.93

First of all I would like to thank you for this kind invitation to come here and talk about type theory. As you have already noticed, type theory is a subject that has turned out to be of interest both to philosophers, mathematicians, computer scientists and linguists (the overwhelming interest and competence in type theory at present is no doubt to be found among computer scientists). To people from these four different areas, type theory has to be presented in very different ways, because you have to attach to the common knowledge, or the background in the particular area that the people in question are coming from. In these lectures, I will try to give an explanation of type theory intended for philosophers.

There is immediately a problem here, because what I will do is to give a detailed explanation of the syntax and semantics of this particular interpreted formal language. A language that you do not know you have to learn by training, by ordinary language training, so if you do not know type theory, what you need above all is a simple training course, so that you get used to the symbolism and can use it properly, just as when you study a natural language, you need a training course. But a training course is something very different from a theoretical course about the syntax and semantics of the language in question. I would be perfectly willing to give such a training course, but it is an altogether different matter, and it is not what I have been contracted for, so to say, by Göran Sundholm. It is to give a course of a theoretical nature on the syntax and semantics of the language of type theory that I am here.

Before beginning I would like to clarify, as an introduction, the relation between logic and mathematics. No doubt, you know that type theory is logic: it is an interpreted logical system. But I think it would be good to have, as a background, a certain understanding of the relation between philosophy, on the one hand, and logic and mathematics, on the other.

The reason I want to try to clarify the relation between these three areas is that we all know that logic has a peculiar position: it does not have a department of its own, but sits between two departments, so to say, namely those of mathematics and philosophy. I want to explain, from a conceptual point of view, why that is natural.

If you are faced with the question, What is mathematics?, I think your first inclination is to try to answer it by saying that mathematics is geometry, arithmetic, algebra, analysis (that is, differential and integral calculus), combinatorics,

and so on. That is, you try to answer the question by simply listing the various areas, the various branches, of mathematics. But when you think of it, it is clear that that cannot be a satisfactory determination of what mathematics is, because mathematics does not change when a new area of mathematics is developed. When category theory or homotopy theory—to take branches of mathematics that have been developed in this century—were developed, everybody clearly recognized that they were new branches of mathematics: there was absolutely no doubt that these were new branches of mathematics.

It therefore seems clear that mathematics cannot be properly defined in this way. Instead, you must start afresh, so to say, in your attempt to answer this question. You do not need to reflect upon it very much in order to realize that mathematics is not defined by its content, but by its method, which is to say the deductive method, the method of proving theorems: proving theorems is what mathematicians do. The deductive method, in its widest understanding, is simply the method of acquiring knowledge through proof, the method that was consciously conceived and described already in ancient times, in particular in Aristotle's *Posterior Analytics*, and that was revived at the beginning of the modern era by the rationalists, by Descartes and Leibniz in particular.

This is then the answer to the question of what mathematics is that seems most reasonable to me. You see that it is a very wide determination: it widens mathematics to encompass all knowledge acquisition through deductive reasoning, or through proof. Mathematics is then conceived as generally as it was in the beginning of the modern era, as mathesis universalis. Whenever deductive reasoning—this typical stepwise acquisition of knowledge—is involved, there is mathematics, according to this general determination.

Let us now turn from this question to the question, What is logic? You may know the traditional definition of logic as the art of reasoning. The Greek for logic was ἡ λογικήˊ, which is a substantivized feminine adjective, and since it is feminine, you can think of it as followed either by τέχνη or by ἐπιστήμη. Presumably, τέχνη was the common understanding, because the Latin translation of it was ars logica, which is to say, the art of reasoning. Logic was thus the art of reasoning.

It was natural that this determination of logic could coexist with mathematics as separate branches, because at the time, that is, from antiquity essentially up to Leibniz, logic and mathematics were totally different areas. In the medieval university training, for instance, logic was part of the trivium, and mathematics was part of the quadrivium. However, once we have determined mathematics in the wide sense that I just explained, then it becomes questionable whether there is any difference, really, between logic and mathematics. Art means simply skilled activity. A logician is therefore someone skilled in reasoning, but that is also what a mathematician is. Mathematicians are trained in in making proofs, that is, in reaching their conclusions through a chain of reasoning, or through proof. Hence, if logic is determined as the art of reasoning, then it does not seem as if there is really any difference between logic and mathematics: is not logic and mathematics then simply the same? But, of course, we all know that logic is entirely different from

mathematics, which means that the determination of logic as the art of reasoning seems doubtful.

How should it be changed? I think it has to be changed by saying, not that logic is the art of reasoning, but rather that it is the theory of reasoning, or the study of reasoning, as one usually says—theory of reasoning would really be the best.

The difference between mathematics and logic comes out very clearly if you think of what you are directed towards, what is, so to say, the product of the mathematicians's activity as opposed to what is the product of the logicians's activity. What mathematicians strive after is to prove theorems, so their products, the goods they sell, are the theorems that they prove, whereas the products of the logicians's activity is not at all theorems: it is a logical system that the logician attempts at reaching, a system of forms of judgement and inference rules, by the use of which you may arrive at mathematical theorems. What the logician strives after is to give an explicit formulation of the rules that the mathematician is following in his reasoning activity.

Logic in this sense is, as I said, the theory of reasoning, or we might say theory of science here: if mathematics is widened to encompass all of demonstrative science, then logic is the theory of demonstrative science, and this is the reason why Bolzano chose to call his four-volume logic, not Logic, as one would have expected, but he actually called it *Theory of Science*, or *Wissenschaftslehre*.

If we have made this change in the determination of logic, from the art of reasoning to the theory of reasoning, then there is no real difference between logic and philosophy, or foundations, of mathematics: mathematics is demonstrative science, and logic is concerned with the conceptual foundations, the philosophical foundations, of demonstrative science. On the other hand, it is clear that logic in this sense is part of philosophy. Philosophy deals with conceptual problems arising in all sorts of areas of human activity, not only in mathematics, that is, not only in demonstrative reasoning, but logic deals precisely with the philosophical problems arising in that area. Logic in this sense is thus a small part of the general area of philosophy.

Admittedly, this is a very small and specialised part of philosophy—and abstruse, perhaps, because of its heavy use of technical symbols. If you are not trained as a mathematician, it is quite difficult, usually, to handle such complicated formalisms, so I admit that this is a very small part, a small area to subject to philosophical reflection. On the other hand, it is precisely by restricting ourselves to a very small area that we shall be able to dig deep. You see, if you try to dig in a very big area, you cannot get very deep. Conceptual problems arising in psychology, for instance, are certainly no less deep than those arising in mathematics, but that is another area, and it has to be subjected to philosophical reflection by those who have their primary experience from that area. The only area where I could profitably do some philosophical reflection is the area where I have my primary experience, and that happens to be in mathematics and related fields.

I would therefore like to defend the restriction to a very limited area, because that is the only way in which we can dig deep. Then we can see, when other people dig into deep conceptual problems in other areas, whether the general philosophical conclusions they reach are similar to the ones that we reach, digging our hole from this direction, where they are digging from a completely different direction. In particular, it is very interesting to see whether there is some similarity between the philosophical conclusions reached in the foundations of mathematics and those reached in the foundations of physics, quantum physics in particular. Physics is just next door to mathematics, and it would be even more interesting to see whether similar philosophical conclusions are reached from more different starting points.

Conversely, the results we arrive at, or the area that we are restricting ourselves to, may be used as a testing ground for more general philosophical views. Whenever someone expresses certain very general philosophical views, we have the possibility of immediately thinking, Well, does this fit? Does this agree with the experience that we have from this limited area that we have thought a lot about?

Our experience from a restricted area may therefore be used in two ways. Firstly, we may try to see if the conclusions we reach generalize to a more general philosophical viewpoint. Secondly, any more general philosophical viewpoint can be tested against what we know from being well acquainted with this very restricted area.

I do not think that we should be timid, in the sense of thinking that this is only of very limited interest because of the limitation of the area. After all, it is philosophical problems originating in the foundations of mathematics that have shaped a considerable portion of, not only contemporary philosophy, but also, philosophy in the past. If we think of contemporary philosophy, say the philosophy of the last hundred years, it is clear that this area has had a tremendous importance, beginning with Frege and Russell and the whole of analytical philosophy, where logic has had a central position. Many people would say it has taken up much too much space in philosophy, and I have sympathy with that view. On the other hand, it is just a fact that analytical philosophy is, so to say, the branch of philosophy that has grown out of problems in the foundations of mathematics and that logic has had this central position. But there is also Husserl and the whole phenomenological movement. We know that the problems that Husserl dealt with, and that gave rise, eventually, to phenomenology were also problems in the philosophy of mathematics and logic. Logic was not at all treated with the technical sophistication that it has been treated with in the analytical tradition, but, on the other hand, philosophical problems concerning the foundations of logic were what gave rise to phenomenology. Husserl's main early work was after all called *Logische Untersuchungen*.

So, as I said, I do not think we should be, so to say, too timid about our interest in this limited area. In particular, I think that digging in this area, in the way that I am going to do, may be one of the more promising ways in which the split between analytical and continental philosophy may be made less sharp or begun to be overcome. There are many signs at present time of a softening of that division, some such signs being the interest in phenomenology among analytical philosophers,

which is at the moment quite widespread, and, on the other hand, but probably to a smaller extent, interest among continental philosophers in analytical philosophy (I think of, say, Tugendhat—there are a few, at least). But somehow, it seems to me that that kind of division is not going to be overcome just by, so to say, learning what the other side is doing. What we need are some sufficiently hard problems that allow us to arrive at a viewpoint that perhaps transcends this sharp division that we have been living with for such a long time.

It is actually my view of the present work that, although it is very analytical—it deals with the foundations of mathematics, and logic has a very central position, since all the time I am going to discuss a logical system—you will see that the viewpoint that I have been led to has perhaps greater affinities with philosophy of the Kant–Husserl type, that is, the philosophy that is at the kernel of the continental tradition, and certainly also it is related to Wittgenstein. Wittgenstein is, of course, one of the philosophers who, perhaps more than anyone else, has been bridging the gap between these two traditions, having his continental background and nevertheless being active during all of his later career on analytical ground.

This much about how I view the relation between logic and mathematics. There is also another way of looking at it. Roughly in the 1920s, logic developed in a completely new direction, so that we got logic, so to say, of a new brand, or a new kind. It is very clear what the origin of that was, namely Hilbert's programme of securing classical mathematics by its formalization and by giving a mathematical consistency proof for that formal system. That was an entirely new viewpoint which led to the idea of studying logical systems mathematically, using mathematical means, and Hilbert created this new term, metamathematics, or proof theory, for this new area.

Logic in this sense has a very different position with respect to mathematics than logic in the original sense of the word. Logic in the sense of metamathematics, that is, logical calculi studied by mathematical means, is simply a branch of mathematics. It has with time become a branch of mathematics that is as technically developed as any other branch of mathematics, and naturally, it was developed by mathematicians, because who else could develop a certain portion of mathematics than sufficiently skilled mathematicians?

So we have this quite radical change that took place in logic, essentially with the First World War. We had logic before the First World War, the period from Frege over Peano to the *Principia*, which is certainly logic in the original sense of the word that I have explained here. Then after the First World War, it is the development of the work on the Hilbert programme that is the strongest impetus to the development of logic, and that leads in the 1930s to the well-known and important results of Gödel and Tarski and so on.

It is not until much later, I think, that we have realized what a profound change, really, in logic this was: logic was changed from being conceived as the philosophical foundations of mathematics to becoming a branch of mathematics. It was difficult to see that that change was so drastic, and those who were on the borderline—it is clear when you look at their work afterwards that there was

some uncertainty really what they were doing. If you think of the oldest logicians that are alive today, which is to say Church and Quine—Church was 90 in June and Quine is five years younger—they are old enough to have started rather in the *Principia* tradition. They started with the ambition of improving *Principia*, or constructing elaborate formal systems which were meant to improve *Principia*: Church with his set of postulates for the foundations of logic in the early 1930s and Quine with his new foundations also in the 1930s. That kind of work was, however, already, so to say, becoming obsolete at the time, and I think they were very much alone in these ambitions. Similarly, one must say that Wittgenstein, in his logical interests, was very much alone and apart. His interest in logic was not in designing new logical symbolisms. His interest was rather a semantical clarification of *Principia*—one may look upon the *Tractatus* as one big attempt to understand the language of *Principia* satisfactorily from a semantic point of view—and when logic took this new line of development, that is, developed into metamathematics, it became something entirely alien to him.

Now, this new kind of logic, logic as metamathematics, or as a branch of mathematics, has dominated completely during the last sixty years or so, beginning in the 1920s. On the other hand, now we can see some signs of change again. The work that I am going to talk about here is rather a continuation of this line in logic, the Frege–*Principia* line, and there are other signs at the present time. Surely, the greatest impetus at the moment for developments in logic come from computer science, and what computer science needs is not metamathematical theorems about various logical systems. Computer scientists need new effective symbolism, and to design a symbolism of that kind is something which is much more in line with logic in this original sense than with metamathematics. It is also fair to say, I think, that there is at the present time, say visible in the late 1980s or something like that, some sign of tiredness with the traditional mathematical logic, metamathematics. Metamathematics was a certain viewpoint that was very fruitful to exploit to begin with, when the idea was new. But like all good ideas, they get exhausted if you work on them sufficiently long, and I think it is fair to say that we have some such signs of exhaustion at the moment with the purely mathematical development of logic. Maybe this would be a good place for a break.

[Displaying[1] a logical system is displaying forms of judgement and forms of inference. What a logical system is has remained stable.

Aristotelian logic

Forms of jugement:

[1]The first part of the lecture after the break was not recorded. What follows inside these square brackets is taken from the notes of Martien Wijers.

All S are P
No S are P
Some S are P
Some S are not P

Forms of inference: Aristotle's syllogisms.
For this to become a formal calculus one would also need formation rules for the terms.

Stoic logic

Forms of judgement (at least):

If A then B
A or B
A and B
Not B

Inference rules
+ formation rules for the propositions.

Frege

Form of judgement:

$$\vdash A$$

Inference rules:

$$\frac{\vdash B \quad \vdash A}{\vdash B}$$

$$\frac{\vdash \Phi(a) \quad A}{\vdash \Phi(\mathfrak{a}) \quad A}$$

Gentzen

Form of judgement:

$$A_1, \ldots, A_n \vdash A$$

+ formation rules for propositions

Inference rules: introduction/elimination rules

Type Calculus

New forms of judgement

$$x_1 : \alpha_1, \ldots , x_n : \alpha_n \vdash a : \alpha$$
$$x_1 : \alpha_1, \ldots , x_n : \alpha_n \vdash a = b : \alpha$$

+ new forms of inference + formation rules for the types (one of the innovations: formation of dependent types)

Notation is something you can only partly decide about. It has become custom to use a colon, as in $x : \alpha$, but it was Peano's ε (from ἐστι(ν)).]

The first type structure in the modern sense was introduced by Frege. Frege tried to work with just one all-encompassing universe of objects, Gegenstände, very much like the Aristotelian category of substance, really, something that encompassed all substances in the old terminology, that is, all things that you use substantives to denote. Once he had introduced that all-embracing universe of objects, he needed functions over that universe. Hence, he had not only objects, or Gegenstände, but he also had functions taking objects into objects of arbitrary many arguments, and functions of such functions, etc., as we know, arbitrarily high up. There immediately arises a terminological problem here, since we are now talking about functions from objects to objects, functions taking such functions into objects, etc.: are they not objects then? In a sense they are objects, at least in a more informal or general sense of object, since we are talking about them in the usual substance type of way. So there is a tension here, a difficulty in Frege's use of the word object.

Forgetting about that terminological problem, what was his type structure? He did not talk about types, of course, but using modern notation and terminology, we would say that he had a type structure, namely the type structure which is generated by the following clause:

$$\text{If } \alpha_1, \ldots, \alpha_n \text{ are types, then } (\alpha_1, \ldots, \alpha_n) \text{ is a type.}$$

In this clause, n is allowed to be 0, 1, etc. In particular, for $n = 0$, we get the type $(\,)$, which is Frege's type of objects. Then putting $(\,)$ for each of $\alpha_1, \ldots, \alpha_n$, we get a type of the form $((\,), \ldots, (\,))$, which is the type of n-ary functions taking n objects into an object again. This can be repeated, of course, arbitrarily high up, so we get, for instance, the important type $(((\,)))$. A function of this type takes a unary function from objects to objects into an object, so it is the type of Frege's course-of-values operator, the Wertverlauf operator. Let this suffice as examples of Frege's types.

As I said, Frege did not use the word type, but he used the word level, Stufe, and his levels may be defined in the following way:

$$L((\alpha_1, \ldots, \alpha_n)) = \max_{1 \le i \le n} (L(\alpha_i) + 1) \qquad L((\,)) = 0$$

The level of a type of the form $(\alpha_1, \ldots, \alpha_n)$ is defined as the maximum of the levels of the argument types plus one, and if $n = 0$ this is the maximum of no numbers, which is to be interpreted as 0, so the level of the type of objects is equal to 0. So these are Frege's Stufen.

Now, this is the type structure of the *Grundgesetze*. In the *Begriffsschrift*, Frege did not treat what he called judgeable contents, beurteilbare Inhalte, as being of the same category as objects, so at that time he had, as we would say, a type also

of beurteilbare Inhalte. It was precisely the idea of treating propositions as objects, the true and the false, in this all-embracing universe of objects that turned out to be not such a good idea and that was eliminated by Russell by introducing—reintroducing, so to say—the base type of propositions.

We then get to Russell's type structure. I am not going to say anything about the ramified type structure, but only about the simple type structure. The type structure of the simple type theory is obtained by adding one new clause to the Fregean rule, namely the clause,

$$\text{If } \alpha_1, \ldots, \alpha_n \text{ are types, then } [\alpha_1, \ldots, \alpha_n] \text{ is a type.}$$

I am using notation here from Dummett's Frege book. The interpretation of $[\alpha_1, \ldots, \alpha_n]$ is the type of n-ary relations, that is, n-ary propositional functions whose arguments are of type $\alpha_1, \ldots, \alpha_n$. That means that, for $n = 0$, we get in particular $[\,]$ as the type of propositions. The notion of level extends, so that we define the level of a type in this type structure by adding the corresponding clause for the relational types:

$$L([\alpha_1, \ldots, \alpha_n]) = \max_{1 \leq i \leq n} (L(\alpha_i) + 1) \qquad L([\,]) = 0$$

This notation for types that I have used here takes the notion of a function of many arguments as a primitive notion. In each case we have a type of n-place functions with arguments of certain types, taking either an object or a proposition as value. Now I want to show the relation between this type notation and a very convenient type notation that was introduced in the 1930s and which is the one that is presently used. It builds on an idea of Schönfinkel's from 1924, namely the idea of representing a function of n arguments as a unary function, that is, as a function of one argument that as value has a function of $n - 1$ arguments, which in turn is a unary function that has as value a function of $n - 2$ arguments, etc., until we come down to a function that has values in one of the two ground types. Using that representation of functions of many arguments as iterated functions of a single argument, it becomes possible to give an equivalent formulation of the type structure that I have just given here, in the following way:

(1) ι is a type
(2) o is a type
(3) If α and β are types, then $(\beta\alpha)$ is a type.

We thus have three clauses for generating types. The first one says that ι is a type, which is Frege's type of objects, Gegenstände, but it is written ι because they are called individuals in *Principia*—by calling them individuals we eliminate the problem with the use of the word object that I mentioned a few minutes ago. Secondly, we have the ground type o, which is the type of propositions. Finally, we have the clause that, given any two types α and β, then we may form the type of functions from α to β, and Church used the notation $(\beta\alpha)$ for this type. These are the rules generating Church's type structure, from Church 1940.

There are alternative notations here, which I will write up:

Schütte	ι	o	$(\alpha)\beta$
Church (1940)	ι	o	$(\beta\alpha)$
Curry	J	H	$\mathrm{F}\alpha\beta$
Ajdukiewicz (1935)	n	s	β/α

Church used, as we have seen, ι and o, and he used $(\beta\alpha)$ for the function type. Curry, probably from the 1930s, but it might have been slightly earlier, but I am not quite sure, used J and H, and for the function types he used the notation $\mathrm{F}\alpha\beta$, so it was a Polish notation, parenthesis free notation, that he used. Then Ajdukiewicz, in 1935, it is from the paper "Syntaktische Konnexität", I think, used n for noun, I believe, s for sentence, presumably, and then the slash-notation, β/α, for the function type. The notation that I am going to use is neither of these. There is an even better notation to my mind, which I believe is due to Schütte, but that should be checked, so it is a later notation. It uses ι and o as before, but for the function type it uses $(\alpha)\beta$. ([Göran Sundholm:] "So for absolute clarity, the argument type is α, and the value type is β. Yes, you see, at least for a mathematician, it is quite inconvenient to have the value type before the argument type as you have in the Ajdukiewicz and Church notation for a function from α to β.)

Whatever notation is used here, the level corresponding to the level that I defined for the Frege and Russell types is instead defined as follows:

$$L(\iota) = L(o) = 0, \qquad L((\alpha)\beta) = \max(L(\alpha) + 1, L(\beta))$$

Now I have to show how you translate between these two notation systems for types, and one should also convince oneself that the level definitions agree. Let us first do it in the direction from the Frege–Russell types to this new notation.

$$\begin{aligned} (\alpha_1, \ldots, \alpha_n)^* &\implies (\alpha_1{}^*) \ldots (\alpha_n{}^*)\iota \\ [\alpha_1, \ldots, \alpha_n]^* &\implies (\alpha_1{}^*) \ldots (\alpha_n{}^*)o \end{aligned}$$

Suppose that we have already translated α_i as α_i^*, for i between 0 and n. Then $(\alpha_1, \ldots, \alpha_n)$ is translated as $(\alpha_1{}^*) \ldots (\alpha_n{}^*)\iota$. According to the intuitive explanation I gave, we treat a n-ary function with individuals as values as a unary function whose first argument is of type α_1 and whose value is again a unary function whose argument is of type α_2, etc., and then the value gets ι. And in the case $[\alpha_1, \ldots, \alpha_n]$ we do the same, except we put o here for proposition.

In the other direction, you just reverse the arrows:

$$\begin{aligned} (\alpha_1) \ldots (\alpha_n)\iota^* &\implies (\alpha_1{}^*, \ldots, \alpha_n{}^*) \\ (\alpha_1) \ldots (\alpha_n)o^* &\implies [\alpha_1{}^*, \ldots, \alpha_n{}^*] \end{aligned}$$

Ground types are included here, namely with $n = 0$. So for $n = 0$, we get ι corresponding to ( ), and o corresponding to [ ]. If it is not a ground type, then it has to be of the form $(\alpha)\beta$, where β has the form, say, $(\alpha_1)\beta'$, where β' again is of this form, say, $(\alpha_2)\beta''$, where β'' is again of this form, etc., until we come down to a ground type. So any type has to be either of the form $(\alpha_1) \ldots (\alpha_n)\iota$ or of the form $(\alpha_1) \ldots (\alpha_n)o$. ([B.G.S.:] "You have used implicitly association to the right?" That is the beauty of Schütte's notation, you see, that... "Oh, that's forced, yes." Yes, that is the beauty, you see, when you have two types α and β, you form the

function type by putting them side by side, and putting the parenthesis there, and not in the ultimate position.)

So we have a one-one correspondence between these type notations. It is therefore fundamentally the same type structure in both cases except that in one case we use the Schönfinkel device of representing functions of many arguments as iterated functions of one argument.

The introduction of the higher type structure, Church's type structure, made it possible to type all the logical operators, that is, treat each logical operator simply as a constant of a higher type, rather than as an operation, as it had been treated all the time up to this elegant formulation of simple type theory given by Church in 1940. Let me take the constants that we have in first-order predicate logic: how are they typed? This is just an exercise in using this type notation:

$$\begin{array}{ll} \bot, \top & [\,] = o \\ \sim & [\,[\,]\,] = (o)o \\ \&, \vee, \supset & [\,[\,], [\,]\,] = (o)(o)o \\ \forall, \exists & [\,[\,(\,)\,]\,] = ((\iota)o)o \\ f & ((\,), \ldots, (\,)) = (\iota)\ldots(\iota)\iota \\ R & [(\,), \ldots, (\,)] = (\iota)\ldots(\iota)o \end{array}$$

We have propositional constants like $\bot$, and perhaps $\top$, and their type is of course the type of propositions. Then we have the negation operation, what does it do? It takes propositions into propositions, so its type is clearly $[[\,]]$, which in the Schütte notation is $(o)o$. And then we have the binary connectives. They are binary functions taking two arguments, both of which are propositions, into a proposition, so that is $[[\,], [\,]]$ or, in the other notation, $(o)(o)o$. Then we have the quantifiers, $\forall$ and $\exists$. What does a quantifier do? It takes a propositional function over the individuals into a proposition, so the value is a proposition, and the argument—it should have one argument, and that should be a propositional function over individuals, so we get $[\,[(\,)]\,]$, and in the other notation that means $((\iota)o)o$. If we have a function constant, f say, an n-place function constant, what is its type? It has n arguments, all of which are individuals—I am thinking of first-order logic here—so it is clearly $((\,), \ldots, (\,))$. (If $n = 0$, that is what we call an individual constant.) What we get in the other notation is clearly $(\iota)\ldots(\iota)\iota$. Finally, if we have a relation constant, R say, for a n-ary relation, then the arguments are the same, but the value is now a proposition instead, $[(\,), \ldots, (\,)]$, or in the Church–Schütte notation, $(\iota)\ldots(\iota)o$. The simple type structure is thus able to type all the constants that you have in first-order logic in this very compact way.

The final thing I want to say is what cannot be typed within this simple type structure and which leads to the dependent types of intuitionistic type theory. What cannot be typed are the quantifiers, provided you vary the domain of the quantification. Suppose we write the quantifier making the domain, D say, explicit:

$$(\forall x \in D)P(x)$$

This is what we do when we interpret first-order logic: we always take an arbitrary set D as the individual domain, and this P would be a propositional function over the individual domain. If we now want to type this quantifier, it is no longer possible in the simple type structure, because the type of P here depends on D. We wish to analyse this as $\forall(D, P)$, the universal quantifier operating on a set D as its first argument, and then the second argument is P, which is a propositional function over D. Here D would have the type set, but P would have the type $(D)o$, that is, a propositional function over D, and then the type of the whole would be a proposition again, so type o. This is, however, not possible to type in the simple type structure, because the type of the second argument, P, depends now on the first argument, D. This is the simplest example of what cannot be done in the simple type structure and which forces us to generalize the simple type structure into a dependent type structure, but that will be for next time.

Lecture 2, 30.09.93

Göran asked me to tell you something about the structure of these lectures, and that is easy. You see, there is a formal structure, namely the formal structure of type theory, that will be, so to say, the skeleton of these lectures, and on that skeleton I will hang various philosophical remarks that are pertinent to making sense of this formal structure. These philosophical remarks will be of quite a varying nature: it will be about categories, about being, certainly a great lot about meaning and meaning theory, and comparison with Frege's "Über Sinn und Bedeutung", and it will be about essences, about definition and many other things. Essentially, my plan is that, when things threaten to become too formal, I will go over to some pertinent philosophical remarks, and vice versa.

Last time, in order to relate what I am doing to what I hope that you possibly know, I introduced the well-known simple type structure, in two formulations. One was the notation used by Dummett in his book, where the rules generating the type structure are these:

$$\frac{\alpha_1 : \text{type} \ \ldots \ \alpha_n : \text{type}}{(\alpha_1, \ldots, \alpha_n) : \text{type}} \qquad \frac{\alpha_1 : \text{type} \ \ldots \ \alpha_n : \text{type}}{[\alpha_1, \ldots, \alpha_n] : \text{type}}$$

Remember that the colon, $: \alpha$, is read 'is α'. I related this to Church's type structure, which is generated by the following rules:

$$\iota : \text{type} \qquad o : \text{type} \qquad \frac{\alpha : \text{type} \qquad \beta : \text{type}}{(\alpha)\beta : \text{type}}$$

In these rules, o is the type of propositions, ι is the type of individuals, and $(\alpha)\beta$ is the type of functions from α to β.

I moreoever introduced the notion of level of these types, which is precisely Frege's notion of Stufe. (We must remember that Frege himself did not have the notion of type: it was introduced only later, in the *Principia*.)

Now I want to show you how this compares with the type structure of my type theory. Let me display, first of all, the forms of judgement used in type theory. I have already shown you, during the first lecture, two of them, namely the forms

$$x_1 : \alpha_1, \ldots \ , x_n : \alpha_n \vdash a : \alpha$$

$$x_1 : \alpha_1, \ldots \ , x_n : \alpha_n \vdash a = b : \alpha$$

When displaying these two forms of judgement, I also said that there had to be rules for forming the types, $\alpha_1, \ldots, \alpha_n$ and α here, so that the whole judgement makes

good sense. Since the formation rules for the types have to be given formally, I therefore have to introduce two more forms of judgement, which are prior to these. These are therefore now the full forms of judgement used in type theory:

$$x_1 : \alpha_1, \dots , x_n : \alpha_n \vdash \alpha : \text{type}$$
$$x_1 : \alpha_1, \dots , x_n : \alpha_n \vdash \alpha = \beta : \text{type}$$
$$x_1 : \alpha_1, \dots , x_n : \alpha_n \vdash a : \alpha$$
$$x_1 : \alpha_1, \dots , x_n : \alpha_n \vdash a = b : \alpha$$

What stands to the left of the Frege sign here, the sequence $x_1 : \alpha_1, \dots , x_n : \alpha_n$, is what I call, following De Bruijn, a context and denote, following Gentzen, by capital gamma, Γ. The context is simply the sequence of assumptions under which a judgement of one of these four forms is made. Judgements in general are thus hypothetical, but of course, categorical judgements are included as the special case when $n = 0$.

Now I want to display the rules for forming types, that is, the rules whose conclusion is of the form $\alpha : \text{type}$, so that you see how my type structure differs from the type structure of simple type theory. In displaying these rules, I have two choices: either to use a natural-deduction style presentation, or to use a sequent-style presentation. The difference is that, in the natural-deduction style, you do not make any assumptions explicit except the ones that are involved in the inference that you are actually considering at the moment, whereas in the sequent calculus, you always make all assumptions explicit all the time, as I have done above: there I have used the sequent notation with all assumptions or hypotheses explicit, namely $x_1 : \alpha_1, \dots , x_n : \alpha_n$. I think I will vary here, sometimes using the natural-deduction formulation, when it is not necessary to be too formal, and in other cases, when one has to be careful, I will revert to the full sequent-calculus formulation.

Using the natural-deduction formulation, the rules of type formation, which by definition are the rules that have a judgement of the form '$\alpha : \text{type}$' as conclusion, are as follows:

$$\text{set} : \text{type} \qquad \frac{A : \text{set}}{\text{elem}(A) : \text{type}} \qquad \frac{\begin{array}{cc} & (x : \alpha) \\ \alpha : \text{type} & \beta : \text{type} \end{array}}{(x : \alpha)\beta : \text{type}}$$

We have a ground type of sets, so set is a type. And, whenever we have a set, which is to say, an object A of type set, we may form a new type, namely the type of elements of A, elem(A). Most often, I will omit writing out elem(A) here and abbreviate that simply to A. It will always be possible from the context to see whether elem should really be inserted in front of A, hence for brevity we may write A instead of elem(A). So these are the two kinds of ground type that we have. Then we have the generalized rule for forming function types, which says that, if α is a type, and if β is a type, which possibly depends on a variable x ranging over a type α, then we may form the dependent function type, which I will denote $(x : \alpha)\beta$. This is the type of functions whose argument is of type α and whose value for argument x is of type β, which may depend on x as indicated.

We thus have a dependent type structure. Such a type structure was introduced, independently, by De Bruijn in AUTOMATH and by myself in type theory. The lists of assumptions, or variable declarations, $x_1 : \alpha_1, \dots , x_n : \alpha_n$, with the characteristic property that each type here may depend on the preceding variables, were also introduced independently in AUTOMATH and in type theory. De Bruijn called them contexts, and when I learned about that nice terminology, I immediately adopted it.

The dependent function type contains the usual non-dependent function type as a special case: you can simply define the non-dependent function type by means of the dependent one. Continuing with the natural-deduction style, we can write it like this:

$$\frac{\alpha : \text{type} \qquad \beta : \text{type}}{\begin{cases} (\alpha)\beta : \text{type} \\ (\alpha)\beta = (x : \alpha)\beta : \text{type} \end{cases}}$$

If α is a type, and β is a type, then the non-dependent function type $(\alpha)\beta$ is a type, and it is defined as follows: we choose a variable x that is distinct from all variables occurring in the assumptions, and then we define it simply as $(\alpha)\beta = (x : \alpha)\beta$: type. Here you have the first example of an abbreviative definition, in this case an abbreviative definition between types. ([B.G.S.:] "Do you use the eigen-terminology for the variable in this case, from Gentzen?" One can use that terminology. In that case, one would say that x is the eigenvariable of this inference here, whereas here there is no eigenvariable, there is simply a variable condition, namely that x must be distinct from all the variables on which you are dependent in your premisses.)

Let us see now what we can do with these new rules. I said last time what the two difficulties with the simple type structure are—or rather, I gave one of the difficulties and forgot the other one. The simple type structure is generated from the type of propositions and the type of individuals by means of the function type operation, so what is the difficulty of understanding the simple type structure? Well, if logic, either propositional logic or predicate logic, any kind of modern logic, is to work at all, we must take for granted that we can make sufficiently good sense of the notion of proposition. So there will be a well-defined type of propositions, we should not expect to have problems there. Similarly, whenever two types α and β are well-defined, surely we have sufficient trust in the notion of function that we think that the type of functions from α to β makes good sense. The difficulty with the simple type structure lies in understanding the ground type of individuals, which, as you remember from the first lecture, is Frege's universal type of Gegenstände. It just does not seem possible to make good sense of this idea of having one all-embracing universe of all objects, including numbers and truth-values and analytic functions and the sun and the moon and what not: it is one of the things in Frege's conception which it does not seem possible to make sense of.

How is this difficulty solved in first-order predicate logic? I mean, if anything works in logic, surely predicate logic must work, so how does one cope with that

difficulty there? One copes with it simply by not quantifying over any such universe of all objects. One limits oneself to a particular interpretation, and in any particular interpretation, the quantifiers range over the underlying set of that interpretation. One has a domain D, or set D, and takes the individuals simply to be elements of that set. In my notation I could write $\iota = \text{elem}(D) : \text{type}$, where D is some fixed set. It varies from interpretation to interpretation, but it is fixed once the interpretation is fixed. Once we have fixed such a set over which the quantifiers range, there is no problem here, because, surely, whenever we have a set, we must be able to talk about the elements of that set and also quantify over them. So the type of individuals limited in that way makes good sense.

But then I remarked that, once we limit the quantifiers to range over a fixed set, then the quantifier really has two arguments, namely a set, or a domain, D, and a propositional function P, that is, a proposition P that may depend on the variable x. In standard notation, this would be $(\forall x \in D)P = \forall(D, (x)P)$, the universal quantifier applied to the domain D and the propositional function $(x)P$—abstraction with respect to x of P—and similarly with the existential quantifier. Now, if you keep D fixed, then there is no problem: then the quantifiers can be typed, as I showed last time using the simple type structure. But when D varies, we have the problem that the second argument here, the propositional function, has a type that depends on the first argument. The first argument, D, is typed as a set, but the second argument, $(x)P$, has to be typed as a propositional function over that set, which, using D instead of elem(D), is written (D)set in the Schütte notation. The type of the second argument thus depends on the first argument. The universal and existential quantifiers therefore simply cannot be typed within the simple type structure: we need the dependent type structure that I just showed you.

Let us now see how these operators can be typed using the new type structure. So I will derive the appropriate type for the quantifiers in the new type structure:

(1) set : type (Ax)
(2) X : set (Ass)
(3) elem(X) : type
(4) prop : type (Ax)
(5) (elem(X))prop : type
(6) ((elem(X))prop)prop : type
(7) $(X : \text{set})((X)\text{prop})\text{prop} : \text{type}$

Begin by noting that, according to the first axiom, set is a type (1). Then, since set is a type, we can assume that X is an arbitrary set (2) (I have not given the precise rules for making assumptions yet, but I hope you are willing to follow me anyway). Then, by the second rule of type formation, elem(X) is a type (3). Fourthly, I now use the type of propositions, Church's o, as another ground type. It will eventually be identified with the type of sets, but I do not want to do that at this moment, it will come later. So, this would also be an axiom then (4): I introduce the type of propositions as a second ground type. Then, by the rule for forming function

types, ordinary function types, we can form the type of functions from elem(X) into prop (5), and that is a non-dependent function type. And then, sixth, we can again use the rule for forming non-dependent function types: since we have the type (elem(X))prop and the type prop, we can form the type of functions from the one to the other, so we get ((elem(X))prop)prop : type (6). I am now, of course, dependent—as I have been in several steps here—upon the assumption (1). Now I discharge that assumption, and I think I will allow myself to use my abbreviation, writing X instead of elem(X), so I get $(X : \text{set})((X)\text{prop})\text{prop} : \text{type}$ (7).

Here we have a new type, and it is the type of functions of two arguments whose first argument is a set and whose second argument is a propositional function over that set, and the value is a proposition. The quantifiers are exactly such functions, so again making a definition on the level of types, I may call this somewhat complicated type by a new name,

$$\begin{cases} \text{quantifier} : \text{type} \\ \text{quantifier} = (X : \text{set})((X)\text{prop})\text{prop} : \text{type} \end{cases}$$

So quantifier is a type which by definition is equal to $(X : \text{set})((X)\text{prop})\text{prop}$. I can then type the universal and the existential quantifier by saying that these are objects of type quantifier:

$$\forall : \text{quantifier} \qquad \exists : \text{quantifier}$$

This was meant to show you what the novelty is as far as the type structure is concerned.

Let us return to the forms of judgement, which I have formally displayed, but not given any semantical explanations of, so far.

In giving such semantical explanations, it turns out that you must proceed in an absolutely rigid order. You have to start, first of all, with the case where the context is empty. If n is the length of the context, we thus have to start with the case $n = 0$. Then we have four forms of judgement, namely, $\alpha : \text{type}$, $\alpha = \beta : \text{type}$, $a : \alpha$ and $a = b : \alpha$. Once these have been explained for a context of length zero, you can take a context of length one, and explain these four forms of judgement for that case. Then you are to take length two, and so on, until you reach arbitrarily long contexts. But even when you have fixed the length of the context, for instance, in the base case, taking $n = 0$, you cannot explain these four judgements in an arbitrary order. Each of them has certain presuppositions, and if a judgement has presuppositions, then the presuppositions have to be explained before the judgement, because they have to be satisfied in order for the judgement at all to make sense. The presupposition structure here is the following,

judgement	presupposition
$\alpha : \text{type}$	none
$\alpha = \beta : \text{type}$	$\alpha : \text{type}, \beta : \text{type}$
$a : \alpha$	$\alpha : \text{type}$
$a = b : \alpha$	$\alpha : \text{type}, a : \alpha, b : \alpha$

To say that α is a type, that clearly needs no presuppositions—Voraussetzungen in German, I do not think that there is really any terminological choice here: presupposition is the only sensible word to use in English. On the other hand, the second judgement, which says that the types α and β are identical, clearly has two presuppositions: we cannot say that α is the same type as β unless α and β are both types, so it has the two presuppositions that α is a type and that β is a type. Similarly, we cannot say meaningfully that a is an object of type α unless α is a type. This is indeed presupposed in the linguistic construction itself 'an object of the type α'. The last case is that a and b are identical objects of type α, or, as I will show further on, a is the same α as b. We cannot say that meaningfully unless a and b are both objects of type α, so certainly we have the presuppositions $a : \alpha$ and $b : \alpha$. We know from the previous line here that those two judgements carry with them the presupposition that α is a type, so surely that is also presupposed here. The last form of judgement therefore has three presuppositions.

This presupposition structure immediately imposes an order among the semantical explanations of these four forms of judgement: we must explain the presuppositions before we explain a certain judgement. We must therefore start with a form of judgement that has no presuppositions, and there is only one such, so our semantical explanations have to—there is absolutely no choice here, the first thing that has to be explained is what a judgement of the form α : type means, which is to say, what has to be explained is, What is a type?

Here I am naturally expressing myself by means of a form of locution that is so deeply rooted in our language that one often does not think about it. I mean, this is simply the old Socratic way of asking, What is? It is the Greek τί ἐστι(ν), and then comes goodness or virtue or whatever you are asking about. In this case I am asking about type: What is a type?

This is the old way of asking questions of this sort, and fortunately it is still living, it works just as well as any more modern way of giving the semantical explanations. Of course, we also have a more linguistic way of phrasing the same question, namely, by asking instead, What does it mean to be a type? Maybe even more explicit would be, What does a judgement of the form α : type mean? (Schematic letters, or schematic variables, are entirely replaceable by three dots or whatever you want, without changing the sense of it, . . . : type). This question can be turned around in numerous other ways as well, but this is already ample for what we need in the following.

What is a type? The simplest answer seems to me to be that a type is defined by what it means to be an object of the type as well as what it means for two objects of the type to be the same. The explanation thus consists of two parts: the first one says what it means, or what it is, to be an object of the type, and the second one says what it means, or what it is, for two objects of the type to be the same. For the first part Dummett in his Frege book has introduced a natural term, namely criterion of application—the criterion under which the general concept, here called type, is applicable to something. The second part of the explanation, which lays down what it means for two objects of the type to be the same, is naturally called

the criterion of identity associated with the type. The term criterion of identity comes, as you know, from Frege's *Grundlagen*, but that is only the term. Since Frege had not stratified his objects into types, but only had this all-embracing universal type of all Gegenstände, he had only one identity, namely, the identity relation over that universe, and hence it was never essential to say identical what: identical sets, identical numbers, identical functions or whatever it is. He could only say that two things were identical, since there was only one type of things.

The answer to the question, What is a type?, is not quite finished yet, because the identity criterion cannot be laid down in a completely arbitrary way. After all, the identity criterion defines the relation of identity between objects of the type, and the identity relation should surely have the usual properties of an equivalence relation. That is, it should be reflexive, symmetric, and transitive. It is therefore a requirement all the time that, whenever we lay down a criterion of identity, we must see to it that the identity relation that it defines is reflexive, symmetric, and transitive, which is to say that we must see to it that the following rules are valid:

$$\frac{a : \alpha}{a = a : \alpha} \qquad \frac{a = b : \alpha}{b = a : \alpha} \qquad \frac{a = b : \alpha \qquad b = c : \alpha}{a = c : \alpha}$$

If we take an arbitrary object of the type, the type that we are in the process of defining, then that object is identical to itself: the law of identity holds. Moreover, if a and b are identical objects, then b and a must be identical objects. Finally, if a and b are identical objects of type α, and b and c are identical objects of type α, then a and c are identical objects of type α. It is well known that symmetry and transitivity together are equivalent to a single rule, so if you prefer to have a single rule, take

$$\frac{a = b : \alpha \qquad a = c : \alpha}{b = c : \alpha}$$

From this we get symmetry by taking c to be a and use reflexivity. Once we have symmetry, of course, we can from this rule derive transitivity, and conversely. It is therefore a matter of choice if you want the three rules here or the two rules, reflexivity and this one.

I have stressed the identity criterion, that is, that it belongs to the definition of the type to say what it means for two objects of the type to be the same. In recent philosophical discussion the necessity of having a clear criterion of identity has been stressed, above all, I think, by Quine (in the article "Speaking of objects"), using the slogan

No entity without identity.

This principle is certainly correct, but when you think about it carefully, you will realize that it really is a conclusion reached in two steps. The first step is what we could call the doctrine of types,

No entity without type.

That is, if we have any entity, or object, whatsoever, it is always an object of a certain type. On the other hand, as I have just said, to the definition of a type,

there always should belong the identity criterion, so,

No type without identity.

I have not really spoken about the doctrine of types here, so things have not come in quite the right order. What is the fundamental idea underlying all kinds of type theory? Well, it is a very simple one. In mathematics, we certainly deal with mathematical objects, for instance natural numbers, integers, rational numbers, real numbers, complex numbers, analytic functions, groups and vector spaces, manifolds and so on, masses of mathematical objects. As you hear from these locutions, we never speak about an object just like that, an object tout court. An object, a mathematical object in particular, is always a something: it is either a natural number or a real number or an analytic function or whatever it is. It is always a something, and that something which it is, that is what we call its type. So whenever we have an object at all, it is always a something, that is, an object of a certain type. That is the fundamental idea underlying all kinds of type theories. This idea will seem quite credible, I think, if you just look at the way we ordinarily express ourselves. It is just never the case that we have an object in front of us, on the blackboard or wherever it is, and it is just an object, and we do not know what kind of object it is.

In the last sentence of this informal discussion, I used another word, namely kind: what kind of object it is. The notion of type is so basic that we have simple words that naturally come to mind already in natural, or non-technical, language, like kind or sort: an object is always of some kind, or sort, or other. It is precisely for this kind, or sort, that Russell introduced the technical term type, which was in a sense unfortunate, because type already had a well-determined meaning, especially in philosophy and in logic, because of the type/token terminology introduced by Peirce. Peirce was surely using the word in the ordinary sense, namely that a type is the same as a shape or a form as opposed to the various occurrences of that shape or form—various individual expressions having that shape or form. Russell used the word type in a new sense here, different from the traditional one, and, thinking of how basic the idea of type is, it is of course unthinkable that there should not be a word already in the tradition for what we here call types, and it is not difficult to find out what it is: you all probably know that the traditional word is category for what I here call type, the word category as it was introduced by Aristotle and heavily used by Kant. After the break I will show that the word, as used in the tradition, is entirely in line, coincides, with the way I am using it here.

What I have called the doctrine of types, namely that an object is always an object of a certain type, really goes back to Aristotle. If you consider his discussion of the notion of being in the philosophical dictionary, that is, book Δ in the *Metaphysics* (1017a22–23), you will find him saying that

> The senses of essential being are those which are indicated by the figures of predication,

τά σχήματα τῆς κατηγοριάς, so the categorical schemes, and scheme, σχήμα, as it is used by Aristotle, is what most closely corresponds to form as we use it. For instance, I have spoken about the forms of judgement and the forms of inference of a logical system, and in both cases the word σχήμα is used in Greek: the syllogistic schemes and here the categorical schemes, the forms of judgement as I have said. The sentence continues,

> for being has as many senses as there are ways of predication,

and the ways of predication are precisely the categories.

What does Aristotle want to say in this quotation? The term being is of course ambiguous, polysemic, has many quite different senses. Aristotle says that even if you concentrate on the verb 'to be' used as copula—when you say that S is P, or in the notation that I am using, $a : \alpha$—it still has as many senses as there are ways of predication. The judgement S is P must therefore not be analyzed as

$$(S) \text{ is } (P)$$

where you have two contents, S and P, which have their senses, and then you have a uniform meaning contribution coming form the copula 'is'. That is precisely what is not true. On the contrary, we must analyze it as

$$(S) \text{ is } P$$

We have just one hole in this judgement, filled by the subject S. The form of judgement has a different sense for each P, so we cannot say what 'is', just 'is', means, but we can say what it is to be a P, for an arbitrary fixed predicate or category P. You must remember that category is from the verb κατηγορεῖν, which is to predicate. Aristotle speaks, in many places, of categories of being, and it is then this notion of being, being a P, which is categorized.

In Kant, particularly in the *Critique of Pure Reason*, of course the notion of category has a central position, and category, Kategorie, in Kant is synonymous with pure concept of understanding (reiner Verstandesbegriff). Instead of Verstandesbegriff you can sometimes see him using Verstandesform, and also Verstandesfunktion is sometimes used—somewhat less fortunate, but also less common—and sometimes you can find Gedankenform. The form that you have here corresponds to Aristotle's scheme, σχήμα. That Kant's notion of category is also in tune with the notion of type as I am using it here is clear from the fact that Kant had his peculiar way of arriving at his categories, which was simply to look at the forms of judgement of logic as it was current at his time and extracting his categories from the forms of judgement. As you have seen, as I am using the word type or category here, it simply is a form of judgement: it is the form of judgement '... is an α'.

It is thus completely clear that the word used in the tradition for type is category, so why not use category in philosophical contexts? There is a problem, of course, since these logical systems are called type theories, and if we change type into category, then we get category theory, which is also the the name of a certain part of algebra invented, in the meantime, by Saunders Mac Lane. He took the word category from Kant and the word functor from Carnap, thinking that it was

safe, so to say, because he could not think that these philosophers could, so to say, have anything living still to offer. One could thus safely snatch these terms from them and put them to some good use, which he did. Unfortunately, we still need the notion of category in the original sense of the word, and then we have the option of using type here, but surely category is really the right word.

This was about the first part of the definition of a category, namely, the criterion of application. The second part, which is the criterion of identity—I have already mentioned Quine's dictum, but the one who really has brought the category dependence of the notion of identity into focus is Geach, in *Reference and Generality*, from 1962, and in an article called "Identity", from 1967. In *Reference and Generality*, Geach says,

> I maintain that it makes no sense to judge whether things are 'the same', or a thing remains 'the same', unless we add or understand some general term – 'the same F'. That in accordance with which we judge whether identity holds I call a criterion of identity.

Geach's point is that we should not just say that a is the same as b, but we have to say, a is the same F as b, in order for the identity statement to make good sense. In the later paper, "Identity" from 1967, he formulates the same idea in the following way:

> I am arguing for the thesis that identity is relative. When one says 'x is identical with y', this, I hold, is an incomplete expression; it is short for 'x is the same A as y', where 'A' represents some count noun understood from the context of the utterance,

In the first sentence here, Geach says that he is arguing for the thesis that identity is relative, and hence this notion of identity has come to be called relative identity: it is relative to the category, or sort, and so often also called sortal identity.

Now, when you start to investigate this idea—it is surely a very important insight that the notion of identity has to belong to the definition of a category, it is something that I first met in Dummett's Frege book again—one begins of course to wonder whether, if this is so basic, maybe it is actually present much earlier? I do not think it is to be found in modern logic, for the reason that I just gave, namely that Frege had his universal type of all objects and the identity relation over that, which was the only identity he had access to. On the other hand, if you look very carefully in Aristotle, in the *Metaphysics*, you will actually find this category relativity stated already there. It is again in book Δ, but now the passage is at 1018a35-39,

> And since one and being have various meanings,

that is, depending on the category. Being I have already discussed, and one is the same: if you ask what one means, the counter-question is, One what? One apple, one orange or whatever—it is always one something. So Aristotle says that,

> Since one and being have various meanings, all other terms which are used in relation to one and being must vary in meaning with them, and so same,

here we have it

> other and contrary must so vary, and so must have a separate meaning in accordance with each category.

So he himself uses the word category here. ([B.G.S:] "Perhaps, if I may interject, the point about count words, one and so on, being sensitive to the categories, is a point that Frege makes quite explicitly, in the *Grundlagen*, § 27, I think [§ 29]." So there it is relative to? "Well he says that ein. . . " He needs a criterion of identity in order for them to make sense? "Yes, the example he uses is, ein Weiser cannot be analysed as somebody who is Ein and also is weiser." [inaudible])

You also find this category relativity of the notion of identity in Locke's *Essay*, Book II, the book of Ideas, Chapter 27, § 7— this is the chapter on identity and diversity—and Locke says,

> 'Tis not therefore Unity of Substance that comprehends all sorts of *Identity*, or will determine it in every case: But to conceive, and judge of it,

that is, identity,

> aright, we must consider what *Idea* the Word it is applied to stands for: It being one thing to be the same *Substance*, another the same *Man*, and a third the same *Person*, if *Person*, *Man*, and *Substance* are three names standing for three different *Ideas*; for such as is the *Idea* belonging to that Name, such must be the *Identity*.

The only difference here is that Locke uses the term idea rather than category.

This is what I had to say about the category relativity of the notion of identity. Remember, then, where we were: I had defined what a type is and required that the identity criterion be such that the identity relation it defines is reflexive, symmetric and transitive, and in formulating those rules I forgot to say that they are the first inference rules that I have written up. Why are they valid? Clearly because it is part of the definition of a type that they are valid, so they are valid for purely conceptual reasons.

Before moving on to the next form of judgement, I should say something about the word criterion. Dummett has proposed this term, criterion of application, and we have from Frege the term criterion of identity. The word criterion here seems to me to be absolutely perfect. What a criterion gives is the explanation of what a judgement of the form in question means, which is to say that it gives the criterion for when you have the right to make a judgement of the form in question. Criterion here is from κρίνειν, to judge, and κριτήριον is just the standard by means of which you judge. So no word could be better than criterion here for the semantic explanation that is associated with a particular form of judgement: it is the criterion for making a judgement of that form. One could preferably also use criterion instead of condition. I mean, it is customary to speak of assertion condition or judgemental condition, but judgemental criterion seems even better.

The question to be answered in connection with the second form of judgement, $\alpha = \beta$: type, is, What does it mean for two types to be the same? The answer is that two types are the same if an object of the one type is the same as an object of the other type, and also to be identical objects of the one type is the same as to be identical objects of the other type, and vice versa. Spelt out more schematically, this means the following. If a is an object of type α, then it should also be an object of type β, and vice versa (and I am going sometimes to use the device of writing double lines when the inference goes in both directions). And if a and b are identical objects of type α, then they are also identical objects of type β, and vice versa. So the identity of the types simply means that you have the right to infer in both directions in these two rules:

$$\frac{a : \alpha}{\overline{a : \beta}} \qquad \frac{a = b : \alpha}{\overline{a = b : \beta}}$$

This is the identity criterion for types.

You can see that this is important by considering types introduced by explicit definition. I have given only one definition so far, namely the definition of the type quantifier. Let me give an even simpler one first, namely the definition of the type of classes of elements of a set A:

$$\frac{A : \text{set}}{\begin{cases} \text{class}(A) : \text{set} \\ \text{class}(A) = (A)\text{prop} : \text{type} \end{cases}}$$

Here you have the second example of a definition, and it is at the type level again. The other example was the quantifier example, where we had quantifier defined to be a type, quantifier : type, by the definition quantifier $= (X : \text{set})((X)\text{prop})\text{prop}$: type. If I have defined class(A) in this way, I can instead define quantifier as follows:

$$\begin{cases} \text{quantifier} : \text{type} \\ \text{quantifier} = (X : \text{set})(\text{class}(X))\text{prop} \end{cases}$$

Here we thus have two very simple examples of identities between types.

You can now immediately see why we need the notion of identity between types: we want to be able to make abbreviative definitions of this sort, and when we have made such an abbreviative definition and we have proved, for a particular Q, that it is of the type $(X : \text{set})(\text{class}(X))\text{prop}$, then surely, we must be allowed to conclude that Q is a quantifier,

$$\frac{Q : (X : \text{set})(\text{class}(X))\text{prop}}{Q : \text{quantifier}}$$

Why? Well, because the type quantifier was defined to be identical to the type $(X : \text{set})(\text{class}(X))\text{prop}$, so surely we must be able to make the inference from $Q : (X : \text{set})(\text{class}(X))\text{prop}$ to Q : quantifier, and vice versa. A formal step is involved here, of course, and we must have rules that allow us to perform those steps, and the formulation of those rules requires this notion of identity between types.

The explanation that I just gave of what it means for two types to be the same immediately justify the rules of reflexivity, symmetry, and transitivity, between types,

$$\frac{\alpha : \text{type}}{\alpha = \alpha : \text{type}} \qquad \frac{\alpha = \beta : \text{type}}{\beta = \alpha : \text{type}} \qquad \frac{\alpha = \beta : \text{type} \qquad \beta = \gamma : \text{type}}{\alpha = \gamma : \text{type}}$$

Why are these rules obvious? Well, we just have to recall what equality between two types, say α and β, means, namely that we can conclude from a's being an object of type α to a's being an object of type β, and vice versa, and from a and b's being equal objects of type α to a and b's being identical objects of type β, and vice versa. In the first case, suppose that α is a type. Is α equal to itself? Can we conclude in both directions here when β is α? Well, surely, we have the same judgement in the premiss and in the conclusion. Next, can we interchange the roles of α and β, as in the symmetry rule? Well, surely, because we have required this in both directions, so we can turn these around. And finally, suppose that α is equal to β and that β is equal to γ. What does it mean for α to be equal to γ? That means that we can infer in both directions here, from α to γ, and vice versa. But we can infer in both directions here, from α to β, and vice versa, and in both direction here, from β to γ, and vice versa.

We have now made a few more rules evident under the meaning explanations that I have given to this particular form of judgement. This finishes the second form of judgement, which says that two types are the same.

Let us now go over to the third form of judgement, which says that a is an object of type α, which has the presupposition that α is a type, and the fourth form of judgement, a and b are identical objects of type α, which has the presuppositions that a is an object of type α and b is an object of type α and that α is a type,

	presuppositions
$a : \alpha$	$\alpha : \text{type}$
$a = b : \alpha$	$a : \alpha, b : \alpha, \alpha : \text{type}$

In explaining what a judgement of either of these two forms means, we may assume that the presuppositions are already known. So we assume that we know α to be a type, and we have to answer the question, What is an object of type α?, respectively, What does it mean for two objects of type α to be the same? Here the answer is very simple because by our presupposition, α is a type. That α is a type means that it has been fixed, or laid down, what it means to be an object of that type and what it means for two objects of that type to be the same: that is what defines the type. So if α is a type, then we know the answer to both of these questions, because it is precisely the answers to these two questions which explain the type.

The situation here is very similar to when you are dealing with the notions of proposition and truth. In the usual classical definition of the notion of proposition you say that a proposition is defined by its truth condition, and if you then later come to ask, What does it mean for proposition P to be true?, the answer is, Well, that depends on P. It is precisely that which defines P. Similarly here: the

explanation of what it means to be an object of type α and identical objects of type α depends on the type. It is precisely that which defines the type.

This finishes the semantical explanation of the four forms of judgement in the case that the context is empty, that is, in the case $n = 0$, and I will deal with the case when assumptions are present next time. Now I want to remark on this rigid order that we have seen, the order in which the semantical explanations have to come.

The question is, first of all, what we shall call this order. What comes most naturally, to me at least, is to call it the order of conceptual priority, or the conceptual order. The realization that we have such a rigid order between our concepts is, again, certainly an age-old one, the question is only how it has been called from time to time. If you look again in Aristotle, you will see that this order is called πρότερον respectively ὕστερον κατὰ τὸν λόγον, which in high scholastic philosophy was translated into prior respectively posterior secundum rationem, ratio here being the translation of the Greek λόγος.

The λόγος that is talked about here is the definition of the essence of the thing. When you ask the question, What is something?, the Socratic question, then the answer is given in language, and the answer is called a λόγος. The λόγος is simply the definition of the essence of a thing. So the order here, which I call the order of conceptual priority, might also be called the essential order, the Wesensordnung.

I want to finish by remarking one important thing about this order. The judgement $a : \alpha$, as we saw, carries with it the presupposition that α is a type. Now, if it carries with it that presupposition, that means that, surely, α : type must be known prior to $a : \alpha$. That is, there cannot be any question of talking of an object of type α unless α is a type, so α—the type—must come before a—the object—in this conceptual order. And, surely, the α here is the universal, or general, concept under which the object a falls. The conclusion is therefore that the general concept, the universale, is conceptually prior to the object which falls under it. This principle had a succinct scholastic formulation, as you probably all know: it is the universalia ante rem principle.

Lecture 3, 7.10.93

Something fell away during last lecture and made another point obscure, so let me repair that. Remember that I explained what it means to be a type as well as what it means for two types to be the same. Then I remarked that reflexivity, symmetry and transitivity are obvious between objects of a type, because it is part of the definition of a type that equality between objects of the type is an equivalence relation. And I gave the detailed explanations for identity between types, that it is reflexive, symmetric and transitive. What I forgot to state, most embarrassingly, was the identity of types rules, that is, the two rules,

$$\frac{a : \alpha \qquad \alpha = \beta : \text{type}}{a : \beta} \qquad\qquad \frac{a = b : \alpha \qquad \alpha = \beta : \text{type}}{a = b : \beta}$$

These two rules are of fundamental importance for the whole system. You may say that they are somehow the navel around which the whole system turns. It is therefore of vital importance that the semantical explanations are such that these rules are validated. And, of course, they are validated on the explanation that I have given of what it means for two types to be the same. That two types are the same means precisely that an object of the one type is also an object of the other type, and vice versa, and that identical objects of the one type are identical objects of the other type, and vice versa. So clearly, these identity of types rules hold.

We can already see why these rules are important—that is why I gave the example with the type of quantifiers. I actually gave two definitions of types, even three, last time. One was the definition of the non-dependent function type by means of the dependent function type. The other was the definition of the type of classes of a set A as the type of propositional functions over A,

$$\text{class}(A) = (A)\text{prop} : \text{type}, \text{ where } A : \text{set}$$

And I gave a third definition in defining the type quantifier, which was to take a set and a propositional function over that set into a proposition,

$$\text{quantifier} = (X : \text{set})((X)\text{prop})\text{prop}$$

Now you see the importance of the identity of types rules. Namely, if I have proved that, let us say, Q is a quantifier, then of course, I also know that Q is an object of type $(X : \text{set})((X)\text{prop})\text{prop}$, because quantifier was defined to be that type. Formally, I need a rule to make that step, however, and that is what I pointed out last time, with this bit missing, hence it was a bit incomprehensible

why I said it. To conclude from Q being a quantifier to Q being an object of type $(X : \text{set})((X)\text{prop})\text{prop}$, and also of course in the other direction, I need to use the identity of types rule, to change the type into an identical type,

$$\frac{Q : \text{quantifier}}{Q : (X : \text{set})((X)\text{prop})\text{prop}}$$

Similarly with respect to the definition of class: if I have a class of elements of A, then of course I need to be able to conclude that it is a propositional function over A, because that is what a class is, and vice versa,

$$\frac{B : \text{class}(A)}{B : (A)\text{prop}}$$

That is done by means of the identity of types rules once the definition here is given.

([Robert Laterveer:] "Excuse me, do you not want to make a difference between definitional equality and the equality which is expressed in the judgement that α and β are the same type?" Yes, you mean the equality which is used when you state the definition? "Right." Yes, I agree completely that you have to have some special mark when you state the definition, and of course that is normally done by writing, say def., before or after, so here we could have def., but... "But then you do not need these rules, in case of the definitions, because it is in the definitional equality." Well, but, that is, you see, even if this is the definition itself, of course I need a rule to make this step, and the rule is precisely this one. I can never do without that. "But..." I agree that it is self-evident, but that is what rules are, they should be self-evident, but nevertheless, it is a step that needs to be made. [B.G.S.:] "The use of definitional abbreviations would be impossible without such a rule." Yes, exactly, you see, we have, first of all abbreviative definitions, and then we will have more elaborate kinds of definition, like recursive definitions, and the rules of the language must be such that we can always replace a definiendum with its corresponding definiens, and vice versa, because, after all, the definiendum and the definiens, looked at linguistically, express the same object, they are only notationally different. So we need rules which allow us everywhere to replace a definiendum with a corresponding definiens, and vice versa, and it turns out to require much labour to set up the language in such a way that it is obvious that you are allowed to make those replacements. This is the first place where it comes in, that we may perform a definitional replacement here, to the right of the colon.)

I have spoken about identity here all the time for good reasons. Namely, considered as types or as objects, what you have on the two sides of the identity sign are the same type or the same object, they are only notationally different. If you want to qualify the term identity here by some adjective, so as to make it absolutely clear what kind of identity it is, the best adjective, to my mind, is definitional: identity between types and between objects of a type is all the time definitional identity.

Let me say already at this stage what the rules are that govern the definitional identity relation. There is a very limited number of rules that may be used to prove definitional identities, and they are the following. First of all there are the rules of

reflexivity, symmetry and transitivity,

so that definitional identity is an equivalence relation. Secondly, there is a rule which states that a definiendum is identical to its corresponding definiens,

definiendum = definiens

Finally, we have the principle that you may substitute, or replace, equals for equals. I should say identicals instead, perhaps, so

substitution (replacement) of identicals for identicals.

What that compact formulation means is that, if you have an expression, then you may replace a part of that expression by a definitionally identical one, and the whole remains definitionally identical. It is the principle that Frege formulated in *Begriffsschrift* § 8 by saying,

sodaß man überall an die Stelle von A B setzen kann und umgekehrt.

So that was what was needed to clarify the obscure point during the last lecture.

At the end of the last lecture I very briefly said something about the order of conceptual priority, or the conceptual order, and I thought that, since I have begun to say something about it, it would be better if I said essentially all I have to say about it at this stage. The reason I introduced it last time was only because I wanted to remark that the presupposition structure that I showed you was such that the judgement $a : \alpha$ has as presupposition that α is a type. Before you can make a judgement to the effect that a is an object of type α, you must therefore judge α to be a type—which is to say that the type is prior to the object in the order of conceptual priority. This, as I remarked last time, is a precise form of the scholastic principle universalia ante rem, because the type is a universal, and the object is a thing, a res. In order to say this, I needed to refer to the order of conceptual priority, and before coming to the order of conceptual priority, you need to reflect a bit on this very way of questioning that I used last time: What is a type? What is an object of a type? etc., this way of putting the question, What is?, the Socratic question, which in Greek is the question, τί ἐστι(ν), and translated into Latin, either Quid est? or Quid sit?

What this question asks for is the essence of something, and the answer to the question, that is, the verbal formulation of the answer to the question, is called the definition. This is to say that the essence is what is defined by the definition. Essence is thus the word we use, or one of the words we use, for the Aristotelian τὸ τί ἐστι, which is simply that which answers the question, What is? This notion of essence is so important that Aristotle has unusually many names for it. Another, slightly longer one, is τὸ τί ἦν εἶναι, and the meaning of this is, at least as I understand it, to be that which is the answer to the question, and now since the question was put previously, we have an imperfect here instead, so to the question What was? Moreover, Aristotle uses ουσία. It is used in many senses, as he himself notes in the philosophical dictionary, book Δ in the *Metaphysics*, but one of the senses of ουσία is precisely this one. He also uses, although less commonly and less formally, φύσις, in this sense.

These are the four terms used by Aristotle. In Latin, τὸ τί ἐστι became quod quid est, and τὸ τί ἦν εἶναι became quod quid erat esse. Then they had the splendid idea of introducing a simple technical term for this, since these are so long, namely quidditas, that which answers the question Quid sit? The basic translation of οὐσία is substantia, but here also they were clever and started using essentia as well, so as to have a difference between the two senses of οὐσία as substance and as essence. And φύσις is of course natura.

In English, this has remained the same: we have quiddity for quidditas and essence for essentia, and nature—nature in the sense in which you ask for the inner nature of a thing, or the intrinsic nature of a thing—is the same as the essence.

Essence is what makes a thing what it is, so it is simply that which answers the question, What is? As I said, the answer is given in a definition, so what is called a definition is the verbal formulation that you use to answer this question. In Aristotle's words,

> A definition is a phrase signifying a thing's essence.

There are many places where you can see that this is how he uses the word. This one, which is particularly clear, is from *Topics* I.5 (101b39). And you can see that this is a relatively modern translation, because the traditional translation here—the word to use here—is formula rather than phrase. There is no difference in sense, of course, it is just a question of how you translate the Greek λόγος.

Now we come to a notion that is very closely connected with the notion of essence, namely the notion of definition. Definition is the Greek ὁρισμός. On a few occasions, Aristotle also uses ὅρος in the sense of definition, but ὅρος is the word for term—in his forms of judgement, all A are B, etc., the A's and B's are called terms, ὅροι in Greek—so it is overloading to use it also for definition. Normally, definition is called ὁρισμός and translated into Latin definitio, whereas ὅρος is normally translated into terminus, except in the few places where he uses it for definition.

A definition is thus a phrase signifying a thing's essence. Therefore, when I gave my explanation of what a type is, for instance, the phrase that I used then—if you remember what it was—is simply the definition of the notion of type, of what a type is. I have actually given several definitions. This was one example, but I have also given the definitions of quantifier and of class(A), and those definitions are of course given in a completely different way from the definition of what a type is. The last two are given in the form of defining equations, where the thing to be defined is on the left, and the defining expression is on the right side, whereas the definition of what a type is cannot be put in that form at all. It was instead given in the form of a meaning explanation, namely the meaning explanation associated with the form of judgement α : type.

The realization that definitions take these two radically different forms is also an old one and to be found in Aristotle. It is the distinction between what in the tradition is called a real definition as opposed to a nominal definition. The distinction goes back to a single place, as far as I know, in Aristotle, namely *Posterior*

Analytics, II.7, 92b26–28. There, Aristotle talks about these two different kinds of definition, namely one which is a definition of the essence of something, that is, answers the question, What is?, by a direct explanation, and the other is the kind of definition where you explain the meaning of a name. The first is what we now call real definition, and the second is what we now call nominal definition. The terms real and nominal definition were introduced in Latin, and the real definition is definitio quid rei, what thing, or, definitio quid nominis, what name. Ockham, for instance, says

definitio dicens quid rei, definitio dicens quid nominis

so, expressing what thing and expressing what name. In the *Port-Royal Logic* it has become so standardized that the quid is away here, so there you have definitio rei, and definitio nominis. But this is the most common formulation, or definitio (with an adjective instead) realis and nominalis.

A nominal definition—and we have already seen, as I have said, a few examples of nominal definitions—has the form $a = b$. In general, these may be typed, so it should be colon followed by type, or these are objects of some type α, in which case it should be colon followed by α here. Talking about it more informally now, I will just write $a = b$. It is not so easy to find good things written about definitions, nominal definitions and definitional equality. Interestingly enough, there is unusually much in the Peano school. This notation for instance,

$$a = b \qquad \text{Df.}$$

which most of us know from the *Principia*, Russell took directly from Peano's *Formulaire*. It appears that an even more common notation,

$$a =_{\text{Def.}} b$$

very often used by mathematicians, was introduced slightly earlier by Burali-Forti, also a member of the Peano school. Yet another notation, very prominent among mathematicians, is

$$a \equiv b$$

Mathematicians, so to say, feel immediately that there is something special about definitional equalities as compared with all other equalities, of which there are a great many, and so commonly write three bars when it is a definitional equality. That is also a notation coming from the Peano school. I think it was first used by Pieri. Quite common at present is also to use capital delta in order to qualify the equality as a definitional equality,

$$a \stackrel{\Delta}{=} b$$

However we write it, this is a definitional equality. Associated with this there is some terminology, namely, a particular way of calling the left- and right-hand member, as well as the definition as a whole. That terminology has changed. The old way of talking about definitions was this:

$$\text{definitum} = \text{definitio}$$

If you remember the Aristotelian formulation of what a definition is, namely that a definition is what defines the essence of something, it is quite natural that, when you wrote $a = b$, then b was called the definition, whereas a was what was defined, that is, the definitum,

Apparently, one did not feel the need to have a special name for the whole equation. Since one sometimes needs such a name, however, one started calling the whole equation the definition. Then we can still call what is defined the definitum, but we need a new word for the defining expression. The present participle, definiens, is then used instead,

$$\underbrace{\text{definitum} = \text{definiens}}_{\text{definition}}$$

That is the standard old terminology, and this is from the modern era. This is, however, still not what we are used to, because we are used to calling the left-hand member here, not the defined expression, but that which is to be defined, that is, the definiendum,

$$\underbrace{\text{definiendum} = \text{definiens}}_{\text{definition}}$$

This, on the other hand, is a very recent thing. The first place I know of is in *Principia*. Hamilton, for instance, in his *Logic* still uses definitum here, and the Italians surely use definito and definiente, if I am not mistaken. So the definiendum has come in here as a kind of mistake rather: the natural words to use is of course the defining expression and the defined expression... ([B.G.S.:]"And did they use a term for the act of defining?" Ja, definitio—"Has got the suitable ambiguity"—has got the ὁρίζεσθαι for the verb and το ὁριζόμενον for that which is defined, definitum.)

This is what I had to say about nominal definitions. This old terminology, real versus nominal definitions, has a lot to it, and actually, all my meaning explanations can be categorized into one of these two boxes. That is, whenever I give a meaning explanation, either it is a nominal definition or it is a meaning explanation of something primitive. That a term is primitive means precisely that it is not the left-hand member of a defining equation. It is what you come to when you make definitional replacements, that is, successively replace a definiendum by its corresponding definiens. If the definitions have been properly made, then you must in that way eventually come to something which is not defined again, and that is precisely what we call something primitive. It is of primitive form, and hence the meaning of the primitive terms cannot be explained by means of nominal definitions, but there we have to give what I think it is natural to call direct meaning explanations, of the kind that I gave when I defined what a type is, for instance. Similarly, we will come to the notion of set eventually, we will come to the particular set of natural numbers, we will come to zero, we will come to the successor operation, and so on. They are all primitive, hence their meaning explanation has to take the form of a real definition as opposed to a nominal definition.

Concerning the notion of essence, I would like to add that—or maybe I could say what I had to say about the order of conceptual priority first. Very rapidly

last time I introduced the order of conceptual priority—conceptual order or order of conceptual priority. Again this is such a fundamental notion that you find it from early on. I had time to say, in the end of the last lecture, that it is the order which is referred to by Aristotle as πρότερον respectively ὕστερον κατὰ τὸν λόγον, so here we have the λόγος that I referred to, either κατα τὸν λόγον or τῷ λόγῳ, with the same meaning. The best reference for that order is when he talks about prior and posterior in *Metaphysics* Δ.11 1018b30-37. So, this is prior and posterior according to definition, or according to formula, that is, the formula that constitutes the definition of a thing. As I said the last time, it was translated into the high scholastic philosophy, which was dominated by Aristotle, as prius/posterius secundum rationem, ratio being the translation of λόγος here, or ratione, as the translation for λόγῳ. It is very natural here also to say prior or posterior according to the definition, because the definition is precisely this formula, so you also have the expression prius/posterius secundum definitionem, or definitione. These terms are used in many places by Thomas, for instance. (I am referring to Thomas here simply because it is so much easier to check Thomas because of the abundance of good dictionaries.) The English translation of this is, as I said, prior/posterior according to formula, using formula here as the translation of λόγος, or in formula, and then for the second two here it will be according to definition instead, and in definition.

Looking at the last formulation here—according to definition—it is of course very natural to call this order, not only as I did last time, conceptual order, but also definitional order, or order of definition, because this is the order in which our definitions have to be given. It is then very natural that we have yet another name for this order, because what is it that the definition defines? The definition defines the essence of a thing, so no wonder that this order is called also the essential order, or order of essence, Wesensordung. I suppose you have it in Dutch too. Thomas, for instance, uses both ordo essentiae, order of essence, and ordo essentialis.

So these are the main terms used for this order. The question is, do you find exactly 'conceptual order'? There is actually such a term—ordo intellectum—although I do not think it is the common term used. One must then remember the many senses of intellectus, one of which is as synonymous with conceptus, that is, your conception of a thing, your concept of a thing. Understanding intellectus in this sense, the proper translation of ordo intellectum is precisely conceptual order, in German, Ordnung der Begriffe, or Begriffsordnung, or begriffliche Ordnung. As I said, this is not a common term, as far as I know, but it can be found in *De Potentia*, VII.11.

The notion of order had a very important role in Greek and in medieval thinking, and the terms that I have given here for the order of conceptual priority are basically from the Aristotelian high scholastic tradition. You also have the Platonic–Augustinian tradition, and there you find the order which you perhaps think of first of all when you hear the word order, namely the order of things, ordo rerum. I

think the reason why this even has gone into natural language, so to say—it is so common that we hardly even notice it—is because of Augustine's *De Ordine*. Augustine's *De Ordine* is really about ordo rerum—the term occurs already at the very beginning of it. It was the authority of Augustine's *De Ordine* which made the concept of order so important in scholastic thinking.

You also have a counterpart to ordo rerum, namely the ordo idearum, and you see here that this is essentially from the Platonic tradition in the duality between thing and idea, res and idea. The ordo rerum is, as I said, from Augustine's book. Ordo idearum is rather from the beginning of the modern era, I think, when it became common to speak about ideas in the sense of concepts, so that the order of ideas here is to be taken as the order of ideas in the modern sense, that is, the order of concepts. That is confirmed by the fact that—at least, the only obvious place that I know of where it occurs is the famous proposition in Spinoza's *Ethics*,

ordo et connexio idearum idem est ac ordo et connexio rerum,

that is, the order and connection between ideas is the same as the order and connection between things. (Here you have a beautiful example of the old way of using proposition in a mathematical—in this case, ethical—text, and not in the modern sense.) This was also quoted by Husserl in the first volume of *Ideen*.

Upon hearing that Spinozistic formulation, my first question is, If the order between ideas is exactly the same as the order between things, what is really the relation between ideas and things? Maybe mathematical concepts, or ideas, are just the same as mathematical things, or objects? It would be a very natural conclusion to draw, and it is actually exactly what will emerge here in the following. This finishes what I had to say about order, and I will continue to speak about it as the order of conceptual priority.

As you have already seen, I am all the time asking this question, either in its old form, What is?, or in its linguistic form, What does it mean? So, the question, What is a type?—that I take to be obvious—is the same question as, What does it mean to be a type?, or even more explicitly, What does a judgement of the form α : type mean? Then I have turned the old Socratic question into linguistic form, that is, into the form of asking for the meaning of certain linguistic expressions or forms of expression, and that is the way that I am going to turn this, or going to answer this question all the time: I am going to answer the question by giving simultaneously a syntactic and a semantic analysis. You might call it the syntactical–semantic method, and it is really a method of answering this old question. Remembering that the old question was a question for the essence of things, you see that what I am doing here in a modern form with my syntactical–semantic analysis is really giving a theory of essences, Wesenslehre. It was the realization of this which made Husserl call his way of proceeding, his method, in the *Logical Investigations*, Wesensschau. My method is therefore a precise way of going about investigating essences, of coming to see essences.

As I said, the Socratic question has got a linguistic turn, a linguistic tinge, or it has been subjected to the linguistic turn, most clearly by Frege. The method of

displaying linguistic form and always giving semantical explanations corresponding to those linguistic forms was pursued by Frege quite consciously, and surely that was at least an influence on Husserl, particularly in the early stage of the *Logical Investigations*. Of course, the old way of asking for the essences of things was not only limited to linguistic things, that is, linguistically expressed things: the old question could certainly be put about anything, I mean, What is it, this particular thing? So it is quite natural that Husserl widened this method—if I could call it linguistic method, or syntactical-semantic method—from the linguistic field to all of reality, to any kind of objects. That generalization from the linguistic field to the non-linguistic, that is, to the parts of reality that are not primarily linguistically constituted, is certainly characteristic of phenomenology. It is the step taken by Husserl from the *Logical Investigations* to the *Ideen*, between 1901 and 1913.

I am not dependent here on whether this syntactical–semantic method can in fact be generalized also to parts of reality that are not linguistically constituted, because I will be concerned with the mathematical world. Mathematical objects, which in their totality make up the mathematical world, are linguistically expressed, which is why this syntactical–semantic method is quite sufficient for my purposes in its original and proper home so to say, in the linguistic domain. That makes it possible for me to avoid the discussion whether it was a good idea in phenomenology to try to generalize the method outside the narrow area of language. I think it was a very good and fruitful idea, but what I will be doing here does not depend on it.

I will not say more at this stage about the syntactical–semantic method that I am using, because it is much more illuminating, to begin with, to see it at work. I can perhaps later try to say something in general about the method itself, but let us go back now to the forms of judgement, which I explained last time in the assumption-free case. Now we have to deal with the case when assumptions are present.

You will remember the context, which had length n, and I dealt with the case $n = 0$ last time. In case the judgement $\alpha : \text{type}$ has assumptions, it looks like this:

$$\alpha : \text{type} \quad (x_1 : \alpha_1, \ldots, x_n : \alpha_n)$$

Now you see that I have reverted to my own old notation, which I somehow find more natural to use: this is the same notation as what I wrote last time, with either the Gentzen arrow or the Frege turnstile,

$$x_1 : \alpha_1, \ldots, x_n : \alpha_n \to a : \text{type}$$
$$x_1 : \alpha_1, \ldots, x_n : \alpha_n \vdash a : \text{type}$$

My old notation is much more close to ordinary mathematical writing, and of course exactly the same information is contained. In any case, this form of judgement is read either as α is a type depending on these arguments or, if you want, α is a family of types, which is what comes most naturally to a mathematician. (I think it is Bourbaki who has this terminology, famille d'ensembles, in this case family of types.)

What are the presuppositions in this case? I always have to make clear the presupposition structure, so the presuppositions are—well I should maybe first of all say what is the subject and what is the predicate in the ordinary grammatical sense in this judgement: α is the subject and all the rest is the predicate. The presuppositions are then

$$\begin{aligned} &\alpha_1 : \text{type} \\ &\quad\vdots \\ &\alpha_n : \text{type} \quad (x_1 : \alpha_1, \ldots, x_{n-1} : \alpha_{n-1}) \end{aligned}$$

To begin with, α_1 must be a type, and it cannot depend on anything, because there are no variables preceding it, and then finally, α_n must be a type in the preceding context, which is now one unit shorter, fortunately, because otherwise my explanations would be circular here.

In explaining what a judgement of this form means for a particular value of n, I will have to assume, of course, that the presuppositions are already known. I have already dealt with the case $n = 0$, and now I am dealing with the case of n, assuming that it is known for all smaller values of n what a judgement of this form means.

So, what does it mean? That question is the same as the question, What is a family of types of n arguments? What does it mean to be a family of types? The explanation is that a judgement of this form means that when you assign objects of the argument types to the variables, or substitute objects of the argument types for the variables, you get a type,

$$\begin{aligned} &\alpha(a_1/x_1, \ldots, a_n/x_n) : \text{type} \\ \text{provided} \quad &a_1 : \alpha_1 \\ &\quad\vdots \\ &a_n : \alpha_n(a_1/x_1, ..., a_{n-1}/x_{n-1}) \end{aligned}$$

That is the first condition, that when we assign objects of the appropriate types as values to the variables, we should get a type. But it is also part of the definition that identity should be preserved, so there is a second condition here, namely,

$$\begin{aligned} &\alpha(a_1/x_1, \ldots, a_n/x_n) = \alpha(b_1/x_1, \ldots, b_n/x_n) : \text{type} \\ \text{provided} \quad &a_1 = b_1 : \alpha_1 \\ &\quad\vdots \\ &a_n = b_n : \alpha_n(a_1/x_1, ..., a_{n-1}/x_{n-1}) \end{aligned}$$

This is the explanation of what a family of types is.

I am here using a certain notation for simultaneous or multiple substitution which I hope you are familiar with from predicate logic. Take for instance the elimination rule for the universal quantifier. I hope you have seen it in both of these forms:

$$\frac{\forall x A(x)}{A(a)} \qquad \frac{\forall x A}{A(a/x)}$$

I am using the notation to the right here for multiple substitution.

That finishes the explanation of what a family of types of n arguments is. As you see, first we deal with $n = 0$, then having dealt with that, it makes sense for $n = 1$, and then after that, $n = 2$, etc. It is really an unlimited number of forms of judgement that we have here, one for each n.

Next we have to deal with identity between types in a context,

$$\alpha = \beta : \text{type} \quad (x_1 : \alpha_1, \ldots, x_n : \alpha_n)$$

What does it mean for α and β to be identical families of types? Here the answer is simply that, when we assign objects of the appropriate types to the variables, we get identical types,

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

$$\begin{aligned} \text{provided} \quad & a_1 : \alpha_1 \\ & \vdots \\ & a_n : \alpha_n(a_1/x_1, ..., a_{n-1}/x_{n-1}) \end{aligned}$$

I forgot to say what the presuppositions are. The presuppositions in this case are that both α and β are types in the context $x_1 : \alpha_1, \ldots, x_n : \alpha_n$, and then of course that the context is well-formed, which is the same set of presuppositions as I wrote with respect to the previous form of judgement. So we have the same presuppositions as in the previous case plus two new ones, namely that α is a type in this context and that β is a type in this context.

Now we come to the most important meaning explanation, namely the meaning explanation which answers the question, What is a function in the old-fashioned sense? Since I will need to distinguish between functions in two different senses, I will use the expressions 'in the old-fashioned sense' and 'in the modern sense'

What do I mean by function in the old-fashioned sense and in the modern sense? The precise definitions will come later, but let me give an intuitive formulation before getting that far.

If you have something like, say, $x^2 + 3y$, where x and y range over natural numbers—0, 1, etc.—then this is a function in one sense of the word function, namely a function of the two variables x and y. I will call that a function in the old-fashioned sense. A function in the old-fashioned sense is thus the same as a function of variables, and such a function is always expressed by an expression built up from constants and variables. That is the traditional formulation, and other examples of functions in this sense are $\sin(x)$ and $\log(x)$. These are all functions in the old-fashioned sense.

Now, in order to make it possible for functions to have functions as arguments—as Frege had, for instance, when he introduced his hierarchy of functions of higher type over his domain of objects, Gegenstände—then it is no good any longer to have functions in the old-fashioned sense. One passes then instead to a corresponding function in the modern sense. In this case, we would call it, say, f, where f is defined by the equation

$$f(x, y) = x^2 + 3y$$

Then $f(x, y)$ is a function in the old-fashioned sense, since it is the same function as $x^2 + 3y$, but if you just write f, it has lost its arguments and become a function in the modern sense. In this sense, then, it is not really $\sin(x)$ which is the function, but sin or log, etc. which is the function, as we are carefully told nowadays.

Of course, the notion of function has changed meaning here, and I need to have some terminology to distinguish between these two, because it will be important in everything that follows. Church tried to face this terminological problem by using function in the modern sense and then using the word form for function in the old-fashioned sense. That is very puzzling when you read Church's *Introduction to Mathematical Logic*: what on earth is this thing that he calls a form? It clashes completely with the way you usually use the word. We would normally talk about a proposition of conjunction form, for instance: A & B is a proposition of conjunction form. It has conjunction form with the two parts A and B, and hence to begin to call $x^2 + 3y$, for instance, a form is very confusing. I think Church had in mind the way you use the word form when you speak of quadratic forms, forms of third degree, and so on. Mathematicians call, for instance, $x^3 + 3x^2y + y^2$ a form of the third degree, so that is, I guess the explanation for Church's use of the word form here.

I try to manage this problem by talking about function in the modern sense and function in the old-fashioned sense. The reason for that is of course that function always meant function of variables from the time when it was introduced in the 1690s up until the late 18th century. It is not so easy to find out exactly who introduced functions in the modern sense, but surely in Frege and Dedekind you find them.

This was just an informal explanation to begin with in order to clarify what I mean by function in the old-fashioned sense. I will talk more about the function concept after I have given my explanations here.

To ask, What is a function in the old-fashioned sense?, is the same as asking what is meant by a judgement of this form:

$$a : \alpha \quad (x_1 : \alpha_1, \ldots, x_n : \alpha_n)$$

What does a judgement of this form mean? Well, this says precisely that a is a function in the old-fashioned sense of the variables $x_1, \ldots, x_n$, whose value is of type α, which itself may depend on the arguments. The presuppositions are the same as for the first form of judgement, that is, such as make the context well-formed, and, of course, that α is a type in that context: I could not say this unless α was a type in the context $x_1 : \alpha_1, \ldots, x_n : \alpha_n$. So suppose that all of these presuppositions are known already. Then I say that a judgement of this form means that, if I assign to the variables objects of the appropriate types, then I get as value an object of type α under the same assumptions,

$$a(a_1/x_1, \ldots, a_n/x_n) : \alpha(a_1/x_1, \ldots, a_n/x_n)$$

$$\begin{array}{rl} \text{provided} & a_1 : \alpha_1 \\ & \vdots \\ & a_n : \alpha_n(a_1/x_1, ..., a_{n-1}/x_{n-1}) \end{array}$$

That is the first condition, that it gives an object of the right type as value. But there should also be what one normally would call the extensionality condition, that if we substitute identical objects we should get identical values. If our language is going to be referentially transparent—it was Quine, I guess, who introduced that term—then we must see to it that all the constructs are referentially transparent. That does not come for free, of course: it has to be checked all the time. In this particular case, the referential transparency of a function means that we should be able to replace its arguments by identical arguments and get identical values. So that must be added as a further condition here:

$$a(a_1/x_1, \ldots, a_n/x_n) = a(b_1/x_1, \ldots, b_n/x_n) : \alpha(a_1/x_1, \ldots, a_n/x_n)$$

$$\begin{array}{rl} \text{provided} & a_1 = b_1 : \alpha_1 \\ & \vdots \\ & a_n = b_n : \alpha_n(a_1/x_1, ..., a_{n-1}/x_{n-1}) \end{array}$$

Here it does not matter if I write $\alpha(a_1/x_1, \ldots, a_n/x_n)$ or $\alpha(b_1/x_1, \ldots, b_n/x_n)$, because I already know that α is extensional. So that is the extensionality condition.

There remains only the final form of judgement, the one which says that two functions are the same. What does it mean for two functions in the old-fashioned sense to be the same? The answer to that question is the semantic explanation associated with this form of judgement:

$$a = b : \alpha \quad (x_1 : \alpha_1, \ldots, x_n : \alpha_n)$$

What are the presuppositions here? First of all, there are the presuppositions which guarantee that $x_1 : \alpha_1, \ldots, x_n : \alpha_n$ is a well-formed context, then the presupposition that α is a type in that context, and the two presuppositions that a is of type α in that context and that b is of type α in that context. So there are $n + 3$ presuppositions. And the meaning is that for whatever objects of the appropriate types we take as arguments, the values are identical,

$$a(a_1/x_1, \ldots, a_n/x_n) = b(a_1/x_1, \ldots, a_n/x_n) : \alpha(a_1/x_1, \ldots, a_n/x_n)$$

$$\begin{array}{rl} \text{provided} & a_1 : \alpha_1 \\ & \vdots \\ & a_n : \alpha_n(a_1/x_1, ..., a_{n-1}/x_{n-1}) \end{array}$$

That finishes my meaning explanations for the forms of judgement.

Maybe I could just use the last two minutes to say that I have here systematically used—as I said when I referred to the universal elimination rule, you have the choice between this kind of notation,

$$\frac{\forall x A(x)}{A(a)}$$

which is the old-fashioned one, all the time used in mathematics, whereas the other notation,

$$\frac{\forall x A}{A(a/x)}$$

is no doubt used only by logicians. I do not think any mathematician even has seen this notation, so why am I using it here? Well, it is for good reasons. Of course I could use the first kind of notation, maybe better for other reasons, but the trouble with it is that, if I have a higher-type language, where I will need to write functional application as $b(a)$, say, then I am in trouble. There are two possibilities. Either you will write something which depends on x as $b(x)$, and then the result of substituting a for x, or giving x the value a, is written $b(a)$. That is the traditional way. The other is to write this quantity which depends on x simply by one letter b, and then the result of giving a as value to x will have to be written $b(a/x)$, or $b[x = a]$, or, as a computer scientist would write, $b[x := a]$. So these are the two notational possibilities here:

$$(1)\ \ b(x),\ \ b(a) \qquad\qquad (2)\ \ b,\ \ b(a/x)$$

The trouble with the first of these is that the notation $b(a)$ then becomes ambiguous, because you do not know whether $b(a)$ means the result of substituting a for x in $b(x)$ or the result of applying the function b in the modern sense to the argument a. If the language is a higher-type language, where we will constantly have to deal with application of functions in the modern sense, it becomes too ambiguous to overload this notation. That is particularly clear in the rule of lambda-conversion,

$$(1)\ \ ((x)b(x))(a) = b(a) \qquad\qquad (2)\ \ ((x)b)(a) = b(a/x)$$

You see that the first alternative does not really work any longer here, because you do not know now whether $b(x)$ is b applied to x or a notation for an expression which depends on x.

Lecture 4, 14.10.93

I finished last time by giving the semantical explanations of the four forms of judgement, including the case where a context is present, that is, where we have dependency and variables. I was all the time then using a notation for substitution, simultaneous or multiple substitution,

$$(a_1/x_1, \ldots, a_n/x_n)$$

This notation comes from Curry, except he uses square brackets and a prefix rather than a postfix notation, as I was using.

The notion of substitution is an old one, and the ordinary mathematical notation is, of course,

$$(x_1 = a_1, \ldots, x_n = a_n)$$

That is ordinary mathematics notation, and because of the great importance of the clause $x = a$ in computer science, particularly in so-called imperative languages, they have invented an even more explicit notation,

$$(x_1 := a_1, \ldots, x_n := a_n)$$

which I think is from ALGOL originally. What is denoted in either case we call a simultaneous or multiple substitution.

The Latin substituere, respectively, substitutio, was used in translating a particular passage in Aristotle's *Prior Analytics*, where the Greek word is μεταλαμβάνω, or the infinitive -λαμβάνειν, and the verbal noun is μετάληψις. If you look at the way in which the verb, μεταλαμβάνω, and the substantive, μετάληψις, are used in the *Prior Analytics*, you will see that they are used for what we now call replacement. You have some complex linguistic expression, and you want to replace some part of it by another expression: that is the operation which is called substitution. This was, surely, the original sense of substitution. Consider, for instance, the Leibnizian definition of identity, that two terms are identical if the one can be substituted for the other salva veritate in all contexts. By substitution there Leibniz surely means what we now call replacement—if one can be replaced by the other salva veritate. Similarly for the well-known principle of substituting equals for equals, that we should everywhere be able to replace an expression by one which is identical, or equal, to it: we call it the principle of substituting equals for equals, but it is really the principle of replacing equals for equals.

The historical situation is thus that substitution originally meant replacement,

$$\text{replacement} = \text{substitution in the original sense}$$

We therefore have two possibilities here. Either we change the terminology, as is almost universally done—in modern logic we use replacement for substitution in this old sense, and then we are free to use substitution for what we write as $(a_1/x_1, \ldots, a_n/x_n)$. But there is also a perfect word that we can use for this, if we want, distinct from substitution, namely, assignment, the word that is actually used in computer science,

$$\text{assignment} = \text{substitution in the modern sense}$$

So this is about terminology and history. As for the notation itself, it is, strangely enough, difficult to pin a mathematician down on the exact notation for substitution. He would probably say that he has no standard notation for it, but if he were really forced to write something, he would usually write something like this:

$$b|_{x=a}$$

We have some expression b, and we want to set x equal to a in b. If it is a simultaneous substitution, he might use a notation like this:

$$b|_{x_1=a_1,\ldots,x_n=b_n}$$

I think the tendency to write something of this sort on the part of a mathematician comes from the standard notation introduced in connection with definite integrals,

$$\int_b^a f(x)\,dx = F(x)|_{x=a}^{x=b} = \underbrace{F(b)}_{F(x)|_{x=b}} - \underbrace{F(a)}_{F(x)|_{x=a}}$$

When you calculate a definite integral, it is standard to use a notation such as this one. You find the primitive function, $F(x)$, for $f(x)$, and then you have to evaluate that between x equal to a and x equal to b. This notation was introduced almost immediately after Fourier had introduced the notation for the definite integral itself. What does the notation mean? It means, of course, that you have to evaluate $F(x)$ at b and subtract the value of $F(x)$ at a. If you use Fourier's notation for the definite integral, it is quite natural to use this other notation as well. I therefore think that this is the reason for the tendency of a mathematician to use a notation such as $b|_{x=a}$ for substitution.

There is an old notation in a paper by Martin Ohm from round 1830,

$$(b_x)_a,$$

but for these very scarce occurrences of an explicit notation in informal mathematics, one had to wait until this century to have an absolutely formal notation for substitution. The earliest one is from Russell, not from the *Principia*, but from the paper on type theory from 1908, and the notation is this:

$$b/(x_1, \ldots, x_n)\text{;}(a_1, \ldots, a_n)$$

We want to substitute in the expression b, and we want to substitute for the variables $x_1, \ldots, x_n$, and what you substitute for them is $a_1, \ldots, a_n$. So this rather strange notation, with its uplifted semi-colon, is Russell 1908.

A notation which comes very close to one of the notations that we are still using, derives, as Göran has pointed out to me, from Julius König, in a little-known book from 1914,

$$\mathrm{S}\,(\,{}^{x_1,\ldots,x_n}_{a_1,\ldots,a_n}\,)\,b$$

Of course, this notation is inspired by the standard notation for permutations in mathematics, an old notation,

$$\begin{pmatrix} 1 & 2 & 3 & 4 & 5 \\ 3 & 5 & 4 & 2 & 1 \end{pmatrix}$$

If you want to permute the integers from 1 to 5, say, you write a permutation like this: 1 is replaced by 3, 2 is replaced by 5, 3 is replaced by 4, 4 is replaced by 2, 5 is replaced by 1. König's notation was taken over almost without change by von Neumann in 1927, but, presumably in order to stress that this is not a notation in the formal language, but an informal notation belonging to the metalanguage, he expanded S here into Subst,

$$\mathrm{Subst}\,(\,{}^{x_1,\ldots,x_n}_{a_1,\ldots,a_n}\,)\,b$$

This notation was taken over by Gödel in his well-known 1931 paper, except that he changed the order,

$$\mathrm{Subst}\,b\,(\,{}^{x_1,\ldots,x_n}_{a_1,\ldots,a_n}\,)$$

Church's notation, the notation used systematically by Church in all of his work in logic, is a variant, no doubt derivative, of this notation, namely

$$\mathrm{S}\,{}^{x_1\ldots x_n}_{a_1\ldots a_n}\,b|$$

So this is Church's notation, found already in his 1932 paper. Heyting, in the second of his formalization papers in 1930, had the unusual idea of not treating substitution as an informal notation in the metalanguage, but rather as a notation in the language itself. He could not give all the necessary rules for it, but since he had it as a notation in the language, he had to have a formally precise substitution notation, and the one that he used, strangely enough, has inverted the lines here, so it is now

$$(\,{}^{a_1,\ldots,a_n}_{x_1,\ldots,x_n}\,)\,b$$

It is likely that Curry's notation,

$$[a_1/x_1, \ldots, a_n/x_n]b$$

is derived almost immediately from Heyting's notation

There are more notations, certainly, but these are the most common ones. In computer science, as usual, one has, so to say, started more or less from scratch and invented one's own notations, but you will see that they contain exactly the same information. The computer science notations that I have in mind are

$$\text{let } x_1 = a_1, \ldots, x_n = a_n \text{ in } b$$

and

$$b \text{ where } x_1 = a_1, \ldots, x_n = a_n$$

These are both from Landin 1964, as so much good terminology and notation in computer science is from Landin.

Maybe I should say something about how I am treating substitution here. As we have seen, I need substitution almost immediately when explaining what a family of types is and what a function in the old-fashioned sense is, but how do I understand it? Do I understand it as an informal notation belonging to the metalanguage, as is customary, or do I understand it as a formal notation belonging to the language itself?

Let us recall the definition of a family of types of one argument. The judgement

$$\beta : \text{type} \quad (x : \alpha)$$

means that

$$\beta(a/x) : \text{type for } a : \alpha$$
$$\beta(a/x) = \beta(c/x) : \text{type for } a = c : \alpha$$

Since I have said nothing about expressions and free and bound variable occurrences and how we substitute and so on, it is clear that, at this stage, this can only be taken as something purely formal. That is, we write b and let it be followed by the substitution postscript, (a/x). However, if this is to be done in a completely satisfactory way, we have to treat it as a notation that belongs to the language itself, for which we have to give exact rules, just as for all other notation of the language.

That has actually been done by now: it is what I talked about at the symposium here in September last year. It turns out to be much more laborious, and therefore interesting, than you might at first think. If I were to deal with substitution in that, in the end, only really satisfactory way, my time here would therefore be almost entirely consumed by the substitution problem, and Göran has advised me wisely not to do that, but rather relax the rigour and just not formalize substitution. We normally do not do that anyway in a presentation of predicate logic or lambda calculus or the like: we leave the substitution informal, maybe with some explanation that we must avoid variable clashes, or something like that. I am going to deal with it with exactly that level of precision. I hope then that you will take this in a, maybe, benevolent way, so to say, because you are already familiar with substitution, and you know how to manipulate the substitution prefix, or—in my case—postfix. But I wanted to say this, that if it is to be entirely satisfactory, you should really give exact substitution rules also, although I will not do it.

Then there is the terminological question that arose last time concerning the notion of function, which I am used to solving by speaking of functions in the old-fashioned sense versus functions in the modern sense. If you think that this terminology—function in the old-fashioned sense—is a bit too home-made, we have alternatives, and I would like to point out what they are. Instead of function in the old-fashioned sense, which can be either a type-valued or an object-valued function, we could speak about dependent type, respectively, object: it is a type or an object

depending on certain variables. Or—and this is terminology that I have actually used already—we could speak about a family of types, and if we do that, we could also speak of families of objects, if we want to, that is, an object that depends on some parameters, or is parametrized by some variables.

This shows that the notion of function in the old-fashioned sense is nothing but the idea of, in traditional terminology, one quantity depending on other quantities. Quantity, the old word here, which we no longer feel like using, of course plays no other role than object: an object depending on other objects—that is what we would say. If we use this terminology, then we will get the word function free to use in the modern sense only, which is certainly how we use function nowadays in mathematics. So if we use dependent type or object, or family of types or objects, then we could take a function in the modern sense to be simply called a function.

I surmise that this way of speaking about dependent types or objects could be very convenient actually, but I am used to this home-made idea of function in the old-fashioned sense, and I think one of the reasons is that, since this is really the basic notion of function, as we shall see, one so to say want to have the term function associated to it. I think that is the reason why I tend to speak of function in the old-fashioned sense. ([B.G.S.:] "In connection with dependent object, it would not be unnatural to speak about indicised object." About? "Indicised object or indexed object." Indexed object, yes, that is another possibility. And also, that is another notational possibility to use. Instead of writing $\beta : \text{type}$, as I did before, we could indicate the variable dependencies by an index. So we could use the notation

$$\beta_x : \text{type} \quad (x : \alpha)$$

and then one would have the notation β_a for substitution, distinct from application. Index notation has its problems also, so I will not write it—but it is one possibility.)

We must now start to explain some rules. We are done with the forms of judgement, but I have not really considered any rules yet, although at an early stage I displayed the rules of type formation, just so that you could see what the difference in the type structure is as compared with the simple type structure. Remember what those rules of type formation were,

$$\text{set} : \text{type} \qquad \frac{A : \text{set}}{\text{elem}(A) : \text{type}} \qquad \frac{\begin{array}{c}(x : \alpha)\\ \alpha : \text{type} \qquad \beta : \text{type}\end{array}}{(x : \alpha)\beta : \text{type}}$$

These three rules, which are the only three rules of type formation, have corresponding identity rules which show that identity between types is preserved under these rules. They are

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

$$\frac{\begin{array}{c}(x : \alpha)\\ \alpha = \gamma : \text{type} \qquad \beta = \delta : \text{type}\end{array}}{(x : \alpha)\beta = (x : \gamma)\delta : \text{type}}$$

The judgement set = set : type is an instance of reflexivity of identity between types, so we can forget about that, but the next rule is important and will eventually have to be justified: if A and B are the same set, then they are also the same as type. For the function type former, we have the rule that, if α is identical to γ, the same type as γ, and if β is the same type as δ, depending on a variable x ranging over α—and here it does not matter if I write α or γ—then the corresponding function types are identical. This rule says that the function type former does not introduce any opaque contexts: we can always make replacements inside α and β and preserve identity.

All of these rules have to be explained semantically. In one sense, it would be more natural to start by explaining the ground types and then proceed to the function type former, but I have decided to take the opposite order here. Although we do not have any specific types yet, I will begin by explaining the function type former and everything that is connected with it. The other rules will then come later.

So here we have—it is really the first rule now of the system, since previously we had only the forms of judgement,

$$\frac{\alpha : \text{type} \qquad \beta : \text{type}}{(x : \alpha)\beta : \text{type}}$$

Fortunately, both the premisses and the conclusion have the form of two of our forms of judgement: the form of judgement that something is a type and the form of judgement that something is a family of types over a type. We know already what a judgement of each one of those forms means, so what remains for us here? Well, assuming that we are in the situation that a type α and a family of types β over α are known to us—and we know what that means because of the explanations of the forms of judgement—then I have to explain to you what type the dependent function type $(x : \alpha)\beta$ is. To explain that, I have to explain—because of the general explanation of what a type is—first of all, what an object of that type is, that is, I have to give the criterion of application, and I also have to explain what it means for two objects of that type to be the same, that is, I have to give the criterion of identity for this particular type.

In giving the criterion of application I am, as a matter of fact, answering the question, What is a function in the modern sense? Objects of this type are namely functions in the modern sense. What is a function in the modern sense? Or, turned more linguistically, What does it mean to be a function in the modern sense? Or, even more explicitly, What does a judgement of the form $f : (x : \alpha)\beta$ mean? The answer to that question is precisely the criterion of application for the type $(x : \alpha)\beta$.

The meaning of this judgement is that, when I take an object a of the argument type α and apply f to a, then I get an object, $f(a)$, of type $\beta(a/x)$—remember that β in general depends on x,

$$f(a) : b(a/x) \text{ for } a : \alpha \tag{1}$$

Just as in the case of substitution that I commented on previously, the notation $f(a)$ is to be understood purely formally: we take f, and we put the left parenthesis, insert a and put the right parenthesis. We have no problem now whether this is an informal notation in the metalanguage or belongs to the language itself: this is a notation in the language itself.

Does this explanation make sense? It presupposes, first of all, that I know what α is. But I do know that because of the premiss that α is a type. Moreover, I must know what an object of the type $\beta(a/x)$ is. But I know that because of the second premiss of the rule, so surely this explanation here makes good sense.

Moreover, it is required, as always, that identity should be preserved. So we also require that $f(a)$ is identical to $f(c)$, which are both objects of type $\beta(a/x)$—since β is a family of types over α, we know, because of the extensionality condition for β, that this is the same type as $\beta(c/x)$, and hence these are both objects of that same type—and we require that this should be the case provided a and c are identical objects of type α,

$$f(a) = f(c) : \beta(a/x) \text{ provided } a = c : \alpha \tag{2}$$

Here, as always, I am sloppy about making all presuppositions explicit, because my habit, and I think anybody's habit, is to—whenever you judge something, it is implicitly understood that all presuppositions have also been judged. Of course, in this case what I need is that a and c are objects of type α, but since these are the presuppositions of the judgement $a = c : \alpha$, I allow myself to omit it.

This is the semantical explanation of what a function in the modern sense is. ([B.G.S.:] "Perhaps you could comment something on the generality involved in the use of a here, because, often in the literature, this gets confused with that of the universal quantifier." Right.) So the generality involved here must in no way be thought to be analysable by means of the universal quantifier. We could call it, perhaps, schematic generality, since it is expressed by means of a schematic variable, or what Quine calls schematic letter. How is it then to be understood, if it is not by means of the universal quantifier? It is to be understood by means of rules. You see, from these two semantic explanations, it is immediately clear that the following rule is valid,

$$\text{(App)} \qquad \frac{f : (x : \alpha)\beta \qquad a : \alpha}{f(a) : \beta(a/x)}$$

This is the second rule of the system that I am showing you. Why is it valid? It is valid because the major premiss, $f : (x : \alpha)\beta$, means precisely that we may infer $f(a) : \beta(a/x)$ from $a : \alpha$. If you want to use the terminology of inference licence, or inference ticket—I do not know who introduced it, [B.G.S.:] "Ryle," yes—we could say that a judgement of the form $f : (x : \alpha)\beta$ is precisely an inference licence that gives us the right to infer $f(a) : \beta(a/x)$ from $a : \alpha$. Then of course the validity of the rule is clear, and that also answers Göran's question about how the generality is to be understood here: it is, as I said, a schematic generality, the kind of generality we have in all logical rules. Rule generality would be another word for it.

Similarly, from the second part of the meaning explanation here, we see immediately that another rule is valid, namely the rule

$$\text{(App-id)} \qquad \frac{f : (x : \alpha)\beta \qquad a = c : \alpha}{f(a) = f(c) : \beta(a/x)}$$

The reason is the same: it is included in the meaning of $f : (x : \alpha)\beta$. It is the second part of the meaning of this that you may infer $f(a) = f(c) : \beta(a/x)$ from $a = c : \alpha$.

Now I have written the same thing up twice here. When I write it as in (1) and (2), I really mean the same as (App) and (App-id). There is no difference in meaning between (1) and (App), and between (2) and (App-id), so I could actually allow myself to speak approximately as Gentzen did when he said that the meanings of the logical operations are determined by the introduction rules. Of course, I have not reached them yet, but he said that their meaning is determined by certain rules. I could allow myself to say, in this case, that if you want to grasp the meaning of this judgement, $f : (x : \alpha)\beta$, then look at these two rules. It is from these two rules that you can read off what this must mean, because the judgement is simply loaded with the weakest meaning it can have so that these two rules become valid.

I think that this is quite a natural and convenient way of expressing oneself, to say that the meaning is determined by certain rules, and then one points to the rules which are the meaning determining rules for the construct in question. It is quite analogous to—and here I am using a simile of Wittgenstein's—explanations of how a complicated machinery works. A complicated machinery consists of masses of little parts, and similarly language—type theory, in this case—consists of masses of little parts, which are put together in a particular way. In order to understand the language, it is not enough just to see formally how the parts are assembled. You also have to have an account of how the whole machinery works, and that account of how the machinery works must consist of explanations given in a particular order. In the simile, the account of how the machinery works is the semantic account of the functioning of the language, and the order in which the explanations must be given is what I have called the order of conceptual priority.

If you want to explain how a car engine works, for instance, you cannot profitably start by the—What is it called? I do not know the English words here... combustion coming out in the end and the thing you have there to prevent noise—you cannot start in that end if you are going to explain how a combustion engine works. You will have to start by saying that here you have the fuel tank and the pipe from the fuel tank feeding the fuel into the carburettor, and you have the battery with the electricity coming to the plugs, and so on. That is the order in which you have to explain how the whole machinery works, and similarly here.

The definition of the function type is not complete, because I have only given the criterion of application. I also have to give the explanation of when two functions in the modern sense are the same, which is to say that I have to explain what a

judgement of this form means:

$$f = g : (x : \alpha)\beta$$

Here it is of course presupposed that α is a type, that β is a family of types over α, and that f and g are objects of this function type. A judgement of this form then means that if we take an arbitrary argument, that is, an arbitrary object a of type α, and apply f and g both to that argument, we get as result two objects of type $\beta(a/x)$, and what is required is that those two objects should be identical,

$$\frac{f = g : (x : \alpha)\beta \qquad a : \alpha}{f(a) = g(a) : \beta(a/x)}$$

Since β is presupposed to be a family of types over α, that means that $\beta(a/x)$ is a type, so it has, not only a criterion of application, but also a criterion of identity associated with it. Hence we can speak of identity between these two objects, $f(a)$ and $g(a)$, of that type.

Now you see that I have made things a bit briefer, because I immediately gave the explanation in the form of a rule. Then it is clear, of course, that the rule is valid, because I could give the meaning explanation simply by saying that $f = g : (x : \alpha)\beta$ means that $f(a) = g(a) : \beta(a/x)$ may be inferred from the minor premiss, $a : \alpha$.

We have now defined completely the function type, and thereby completely justified the rule for forming function types that I started with. Let us next compare the two notions of function—function in the old-fashioned sense and function in the modern sense—and see what the difference is. Let us do it in as simple as possible a case, namely, a function of a single argument with a non-dependent range,

function	
old-fashioned	modern
$b : \beta \quad (x : \alpha)$	$f : (\alpha)\beta$
means that	
(1) $b(a/x) : \beta$ provided $a : \alpha$	(1) $f(a) : \beta$ provided $a : \alpha$
(2) $b(a/x) = b(c/x) : \beta$ provided $a = c : \alpha$	(2) $f(a) = f(c) : \beta$ provided $a = c : \alpha$

Here I hope that your reaction is, at least with respect to the second column here, Do I need to be told this? Is this not what we all know, that this is what a function is? Possibly forgetting about the second condition here—if one is less careful, one does not stress the identity-preserving character of a function—surely, the first condition we all know and have seen stated in one form or the other in our textbooks in mathematics.

But this is in the nature of things. I mean, it should be like that. If you already know the mathematical language, if you have been trained in it, if you have an implicit knowledge of it, then, in one sense, you certainly know that this is what a function is. It is just that you may not have seen it explicitly stated. But once you see it stated, you say, Yes, of course, that is what I have always known. Your

experience then is essentially what Plato introduced the term ἀνάμνεσις for. You are just reminded of something that you already knew. That is, you already knew it implicitly, but you now see it stated explicitly, what you knew all along. There is no need to think that you learned this before birth or anything like that: it is something that you have learned through your training in mathematics. But you have not had it made explicit to yourself perhaps before. In this case, I am sure that everybody has had it made explicit in one way or another, but with most of the principles, I am absolutely sure that you have not seen those made explicit before. Nevertheless, once you see them, they are utterly self-evident, as they should be, since they are the axioms and the primitive rules of inference, and they should be, if anything, self-evident.

On the other hand, there is something novel here as compared with modern presentations of the function concept, namely that I have been stressing all the time the old-fashioned notion of function. As we will see, the old-fashioned notion of function can never be avoided, so it is very strange that it has, so to say, disappeared from mathematics. Here we have the old-fashioned notion of function, so to say, restored. I would like to say outright that it is conceptually prior to the modern notion of function, but we will see, maybe next time, that this is not true in an entirely unqualified sense, so I have to be careful. What I can say for sure is that we cannot do without it. Without the notion of old-fashioned function, we would not get any functions in the modern sense, because whenever we define a function in the modern sense, we do it in terms of a function in the old-fashioned sense, so we need this notion.

The difference between the two is very small, although crucial. The difference is essentially only that, whereas a function in the modern sense is supplied with its argument by application, a function in the old-fashioned sense is supplied with its argument by substitution. Substitution thus plays with respect to old-fashioned functions the same role as application plays with respect to functions in the modern sense. (Instead of argument and value, which I am speaking about naturally in mathematical terminology, we could use the terminology of input and output, as is common in computer science.)

This would be a good place to say something about the history of the function concept. As you probably know, the very term or idea of function is due to Leibniz. I think the first appearance of the term is in a paper from 1692, where you have the Latin functio appearing, but not with any attempt at giving a definition. Already two years after, in 1694, you have stated what we normally think of as the Euler definition of function by Johan Bernoulli. By the Euler definition, I mean the norm-giving definition of function that was given in the first calculus, or analysis, text, which was Euler's *Introductio in analysin infinitorum* from 1748. The canonical formulation there, which generations have known more or less by heart, was

> Functio quantitatis variabilis est expressio analytica quomodo-cumque composita ex illa quantitate variabili et numeris seu quantitatibus constantibus,

that is,

> A function of variable quantities is an analytical expression composed in any way from these variable quantities and numbers and constant quantities.

I have been used to thinking that this is the Euler definition of function, but as a matter of fact, as I said, it ought to be called the Bernoulli definition of function, because already in 1694 you had the formulation

> Let n be any quantity composed in any way out of variables and constants,

and in a more elaborate treatment from 1718, for instance, by Johan Bernoulli, you have also,

> On appelle ici Fonction d'une grandeur variable, une quantité composée de quelque manièere que ce soit de cette grandeur variable & de constantes.

What I have been used to calling the Euler definition of function is thus the Bernoulli definition of function, and since the early reference here is just at the time of the first appearance of the term function, it means that it comes from the Leibniz circle. (Johan Bernoulli was among the people who were in direct contact with Leibniz.) So this was Johan Bernoulli, and then comes Euler. In his most well-known book, the *Introductio* from 1748, as I have said, it is interesting to see a mathematics text beginning by explaining, first, what is a variable, second, what is a constant, then, third, that you may assign a constant as value to a variable, and then, fourth comes the definition of function that I just quoted, which is the Bernoulli definition of function.

This definition of function, as an analytic expression composed in an arbitrary way from constants and variables, was no doubt norm-giving. It was almost universally accepted up to the 1820s or 1830s. Then, in connection with discussions of whether an arbitrary real-valued function could be developed into a Fourier series, there arose a feeling—essentially among mathematical physicists, so not in pure mathematics—that this Euler definition of function was much too restrictive. We could think of, say, the temperature in Leiden as a function of time or something like that, and why should we think of that as being given by an analytic expression? We should rather let it be determined in some completely arbitrary way, and that notion of function—that a function is an arbitrary assignment of a value to an arbitrarily given argument—I, at least, have been used to calling the Dirichlet notion of function. Dirichlet, in 1837, in a paper precisely about the problem of developing an arbitrary function in a Fourier series, says quite explicitly that he rejects the idea that a function should be given by an analytical expression: it should rather be an arbitrary assignment, or law of correspondence, or whatever word you want to use. So it was the development in mathematical physics that gave rise to this change, and—according to the source that Göran gave me—Dirichlet was not the first to say this, but it is rather Fourier that takes the step for the first time of rejecting the idea of an analytical expression.

It appears that Euler—and I think it was a Swedish historian of mathematics, the only one we have really, Eneström, who discovered this—had not only this well-known definition, the Bernoulli definition, but he actually had two definitions of function: one in the *Introductio* and another one a few years later in the 1750s, which is his book on the differential calculus called *Institutiones calculi differentialis*. There one finds what I would say is an almost perfect definition of what a function in the old-fashioned sense is. In translation it reads,

> If, however, some quantities depend on others in such a way that if the latter are changed, the former undergo changes themselves, then the former quantities are called functions of the latter quantities. This is a very comprehensive notion and comprises in itself all the modes through which one quantity can be determined by others. If, therefore, x denotes a variable quantity, then all the quantities which depend on x in any manner whatever or are determined by it are called its functions.

This is a very good definition of what a function is in the old-fashioned sense. It is precisely that idea which is captured in a somewhat more formal way in the explanation that I gave last time. But, as I said, although this formulation is as early as 1755, the 1748 definition kept on at least until the 1820s and was only in the 1830s, approximately, replaced by the Dirichlet definition.

As I said last time, you have to go to the latter part of the last century to find the notion of function in the modern sense. The sources that immediately come to my mind at least are, on the one hand, Dedekind's *Was sind und was sollen die Zahlen?*, from 1887, and, on the other hand, Frege, in the *Begriffsschrift* and in the first part of *Grundgesetze*. It is clear that they both have the modern notion of function in mind, and Dedekind's formulation, for instance, is very typical. I mean, it is not better stated in any modern text,

> By a mapping of a system S,

system is Dedekind's word for set,

> a law is understood, in accordance with which to each determinate element s of the system S there is associated a determinate object, which is called the image of s and is denoted by $f(s)$; we say, too, that $f(s)$ corresponds to the element s, that $f(s)$ is caused or generated by the mapping,

the first appearance of the word mapping,

> f out of S, that s is transformed by the mapping f into $f(s)$.

This is very clearly the modern notion of function.

As for Frege, the situation is a little bit complicated. It is clear that in the *Begriffsschrift*, he stuck to the first Euler definition, that is, the Bernoulli definition, and spoke about a function as an expression. A function in the *Begriffsschrift* is thus an analytical expression, whereas he corrected that in the *Grundgesetze*, where he distinguished carefully between function and functional expression.

If we ask whether Frege had the old-fashioned notion of function or the modern notion of function, it is not entirely straightforward to answer. In spirit, it seems to me quite clear that it is the modern notion of function that he had or aimed at, because he said that the function is the part that remains, so to say, when you take the argument away from the expression. It is the expression with holes in it, and he used this term, by chemical analogy, that the function or the functional expression is unsaturated as opposed to saturated, and when you supply the function with its arguments you saturate it, in his terminology.

From all that it seems clear that it is the modern notion of function that he had in mind. But it is not so straightforward, because when it comes to having a notation for functions, he after all writes up functional expressions of the familiar old-fashioned kind. What example did I have last time? Say $x^2 + 3y$. When Frege wants to have an expression for such a function, he does not do what we now are doing, say define $f(x, y) = x^2 + 3y$ and then speak of f as the function in the modern sense. This is the natural thing to do now, but Frege does not have a satisfactory notation for functions in the modern sense. The way he handles it is by introducing these special Greek symbols, ξ and ζ, and he takes $\xi^2 + 3\zeta$ to be the expression of the function. Now of course he has after all put these Greek letters into the argument places, so it does not have holes in it any longer, it just looks like the usual expression for an old-fashioned function. What he rather wants is the expression with holes in it, something like $\bigcirc^2 + 3\bigcirc$. [Audience:] "Different holes." Yes that is the problem: different holes. You have to indicate that in some way, and Frege's way was precisely to insert something into the holes again. Quine had the idea of putting numbers there, $❶^2 + 3❷$. He called them stencils, but it is exactly Frege's way of doing it. The point here is that Frege's expression for a function is after all a saturated expression: it has been saturated by these Greek variables.

I therefore think one must admit that Frege did not have a completely satisfactory way of denoting functions in the modern sense. It is not that it does not work, because he does manage with this notation. My guess is, however, that it was precisely in order to analyze with full formal rigour Frege's functional notation that Church was led to introduce the idea of functional abstraction. The complexity of the substitution process in Frege's functional hierarchy is considerable, and a good formal notation helps greatly. So Church had the splendid idea to denote the function in the modern sense by, in his notation, $(\lambda x, y)(x^2 + 3y)$, or as I am going to write it, without the lambda, $(x, y)(x^2 + 3y)$. This is not the notation used by mathematicians, because they universally use a notation such as f, defined by $f(x, y) = x^2 + 3y$, which also works fine. However, surely this is a great improvement over Frege, to introduce a systematic notation for functional abstraction.

([B.G.S.:] "Of course Frege does use the spiritus lenis notation, which is yet a further complication because that corresponds to yet another thing in your system." Yes, so maybe that could be a reason to make another comment.) As I have said last time, the modern notion of function is really needed only when you begin to consider functions of functions, that is, functionals. When did one begin to consider

functions which have functions as arguments? That was, of course, in calculus: the first time you had such notations for functions with functions as arguments was in the differential and integral calculus. Here is an example:

$$\int_a^b f(x)dx$$

Analyzed in the modern way, that is, in the style of combinatory logic and lambda calculus, you could write, say, with I for integral,

$$I(f, a, b)$$

if you choose the alternative of not showing the argument of f explicitly here, so that you just write f—but f may depend on x, and then you will have to put abstraction with respect to x here,

$$I((x)f, a, b)$$

Having functions as arguments of other functions is thus tantamount to having variable binding operations. Whenever you have a variable binding operation—like the integral here or with the notation for the derivative—it means that you have a function as argument of a function.

This great notational and conceptual invention is again from Leibniz, in the notations for the integral and the derivative. It is, however, only with Frege that you have many other variable-binding operations. First of all, he invented the quantifiers, which are variable binders, precisely because they have as arguments propositional functions, and take them into propositions. He also had the course-of-values operator, written $\dot{\alpha}\Phi(\alpha)$, which Göran mentioned. It takes a function Φ from objects to objects into an object. If I use, say, λ for that function, I could write it as $\lambda(\Phi)$, because that is a function in the modern sense from objects to objects into an object. Frege did, however, not have the idea of pure abstraction, because if I use this notation for his Wertverlauf operator, then you would have to write here either $\lambda(\Phi)$ or $\lambda((x)\Phi(x))$. He did not have this idea of just making the pure functional abstraction. It is only with Church that you will have that.

I have a few minutes left, and they can be used to justify the remaining rule for forming function types, namely the rule which says that the function type former is extensional, that is,

$$\frac{\alpha = \gamma : \text{type} \qquad \beta = \delta : \text{type} \quad (x : \alpha)}{(x : \alpha)\beta = (x : \gamma)\delta : \text{type}}$$

If α and γ are identical types, and β and δ are identical types depending on x, then the function types are identical.

I have explained already what identity between types means. Two types are identical, $\alpha = \beta$: type, remember, means that if a is an object of type α, then it is also an object of type β, and vice versa, and if a and b are identical objects of type

α, then they are identical objects of type β, and vice versa,

$$\frac{a : \alpha}{\overline{a : \beta}} \qquad \frac{a = b : \alpha}{\overline{a = b : \beta}}$$

That defines identity between types.

So assume you know the premisses $\alpha = \gamma$: type and $\beta = \delta$: type $(x : \alpha)$. I have to convince you of the conclusion, which is to say that I have to convince you that if f is an object of the one function type, then you may conclude that it is an object of the other function type, and similarly, if f and g are identical objects of the one function type, then they are also identical objects of the other function type,

$$\frac{f : (x : \alpha)\beta}{\overline{f : (x : \gamma)\delta}} \qquad \frac{f = g : (x : \alpha)\beta}{\overline{f = g : (x : \gamma)\delta}}$$

It is sufficient of course to do this in one direction. So assume that f is a function of the dependent type $(x : \alpha)\beta$. I have to explain why it is also a function of type $(x : \gamma)\delta$. What does this mean? It means that, if I take an arbitrary object, a say, of type γ, and apply f to that, I get something of the type $\delta(a/x)$. But if a is of type γ, then, because of the premiss $\alpha = \gamma$: type, I certainly know also that a is an object of type α. And because of the premiss $f : (x : \alpha)\beta$, then $f(a)$ is an object of type $\beta(a/x)$. Now use the second premiss, $\beta = \delta$: type $(x : \alpha)$. If I take an object, in this case a of type α, then I know that $\beta(a/x)$ is identical to the type $\delta(a/x)$, and hence by equality of types, I can conclude that $f(a)$ is indeed an object of type $\delta(a/x)$. So that was one way with a single argument here.

Then of course we have also to prove the second property here, so let us do that. Assume you know the premiss, that is, that f and g are identical functions of the function type $(x : \alpha)\beta$. We want to convince ourselves that they are also identical functions of type $(x : \gamma)\delta$. What does that mean? It means that, if we take an object a of type γ, then the values, $f(a)$ and $g(a)$, are identical objects of the type $\delta(a/x)$. But if a is of type γ, it is also of type α, because of the premiss $\alpha = \gamma$: type. Hence, because of the premiss $f = g : (x : \alpha)\beta$, we know that $f(a)$ and $g(a)$ are identical objects of type $\beta(a/x)$. And now, since $\beta(a/x)$ is identical to $\delta(a/x)$, because of this property of identity between types, we also have $f(a)$ identical to $g(a)$ as objects of type $\delta(a/x)$, and that is precisely what we want to see here in the conclusion. So this finishes the semantical justification of the extensionality rule.

And I will continue in the... [B.G.S.:] "...only in the preservation of $a = c$, so to say." Eh. "That equal objects are mapped into equal objects." I have forgotten that one, yes. So we should also check that if $a = c : \gamma$, then $f(a) = f(c) : \delta(a/x)$, since this is part of what $f : (x : \gamma)\delta$ means. Now, if a and c are identical objects of type γ, then, because of the first premiss, they are also identical objects of type α, and then, because of the premiss $f : (x : \alpha)\beta$, we can conclude that $f(a)$ is equal to $f(c)$ of type $\beta(a/x)$. But since $\beta(a/x)$ is identical with $\delta(a/x)$, and identical types enjoy this property, we can also take the last step here.

So next time I will speak about how we get objects in the function types, that is, the rules of functional abstraction, and I will also talk about the scheme for

explicit definition. As yet we have no objects—we have only types—but we will get objects next time.

Lecture 5, 21.10.93

I finished last time by explaining why the function type former is identity preserving and said that, so far, we have no objects. That is what I will be concerned with today: what are the ways in which we can build up objects of the various types?

The first rule of object formation is the assumption rule. In natural-deduction style it could be written as follows:

$$\frac{\alpha : \text{type}}{x : \alpha}\ \text{by assumption}$$

If you have already constructed a type α, then you may assume that x is an arbitrary object of type α. You introduce a variable x, which is a new symbol that must not clash with any of the variables on which α depends. Since this is natural-deduction style, the premiss here in general depends on certain assumptions that have been made already, and the chosen variable x must be distinct from all the variables on which α depends. Of course, when we judge x to be an object of type α, we are now judging that in a new context. Namely, the judgement $x : \alpha$ depends, not only on all assumptions on which α depends, but it also depends on this new assumption itself.

This is an example of a rule where it is helpful to write out all assumptions explicitly, because the things I have said here is not shown in the figure. So let me write it out in this case,

$$\frac{\alpha : \text{type} \quad (x_1 : \alpha_1, \ldots, x_n : \alpha_n)}{x : \alpha \quad (x_1 : \alpha_1, \ldots, x_n : \alpha_n, x : \alpha)}$$

This is just a more explicit way of writing what I just explained in natural-deduction style.

What is the semantical justification of this rule? Suppose you know the premiss, and I will try to make the conclusion evident to you. What does the conclusion mean? It means that if we take objects

$$\begin{aligned} a_1 &: \alpha_1 \\ &\vdots \\ a_n &: \alpha_n(a_1/x_1, \ldots, a_{n-1}/x_{n-1}) \end{aligned}$$

and an object

$$a : \alpha(a_1/x_1, \ldots, a_n/x_n)$$

and make the substitution $(a_1/x_1, \ldots, a_n/x_n, a/x)$ in x, then we get an object of type $\alpha(a_1/x_1, \ldots, a_n/x_n, a/x)$. According to the substitution laws which are not

formalized here, the effect of performing that substitution is a, of course, so we have to see to it that a is an object of type $\alpha(a_1/x_1, \ldots, a_n/x_n, a/x)$. Why is that the case? Well, it is because of the last premiss here, $a : \alpha(a_1/x_1, \ldots, a_n/x_n)$. There is a difference between the substitution $(a_1/x_1, \ldots, a_n/x_n)$ and the substitution $(a_1/x_1, \ldots, a_n/x_n, a/x)$, but since x had to be chosen distinct from all the variables on which α depends, α does not depend on x, and hence this last clause, a/x, has no effect. The type $\alpha(a_1/x_1, \ldots, a_n/x_n, a/x)$ is therefore the same type as $\alpha(a_1/x_1, \ldots, a_n/x_n)$, and we know already that a is an object of that type. I am thus using the fact that α does not depend on x.

That is only half of what the conclusion $x : \alpha$ $(x_1 : \alpha_1, \ldots, x_n : \alpha_n, x : \alpha)$ means, if we are to be careful: it also means that if we take identical objects of the argument types,

$$\begin{aligned} a_1 &= c_1 : \alpha_1 \\ &\vdots \\ a_1 &= c_n : \alpha_{n-1}(a_1/x_1, \ldots, a_{n-1}/x_{n-1}) \end{aligned}$$

and identical objects

$$a = c : \alpha(a_1/x_1, \ldots, a_n/x_n)$$

and we perform both the substitution $(a_1/x_1, \ldots, a_n/x_n, a/x)$ and the substitution $(c_1/x_1, \ldots, c_n/x_n, c/x)$ on x, then we get identical objects of the corresponding type. If we perform the first substitution on x, we get a, of course, and if we perform the second substitution on x, we get c, so we have to see to it that a and c are identical objects of type $\alpha(a_1/x_1, \ldots, a_n/x_n, a/x)$. Again, for the same reason, this is just the last premiss but for the substitution of a for x, which has no effect, because α does not depend on it. ([B.G.S.:] "In the substitution calculus you mentioned, then, you make this precise, what it means that these two substitutions coincide?" Yes, but you can imagine that there is a lot of complexity if you are to make this completely formal.)

Now we have at least something to start building objects from, namely variables. Our syntax is such that any kind of object or function will be built up from variables and constants by means of application and abstraction. The constants will have to wait, because at the present moment I am in the process of explaining everything that has to do with the type structure, whereas the constants have to do with the particular set-theoretical and logical constructs that we have, and they will have to come later.

What I shall deal with today is the means for constructing new objects out of old ones, namely application and abstraction. Application I have actually dealt with last time already, so I will just need to recall what the rules were. The first rule is that we can form the application $f(a)$ provided f is of a function type, so f is a function in the modern sense, and a is an object of its argument type,

$$\frac{f : (x : \alpha)\beta \qquad a : \alpha}{f(a) : b(a/x)}$$

This is one of the rules that we justified last time. Moreover, we have to check, as always, that application is identity preserving, and I also did that last time. Its

identity-preserving character to the right was this rule:

$$\frac{f : (x : \alpha)\beta \qquad a = c : \alpha}{f(a) = f(c) : \beta(a/x)}$$

This follows immediately from the meaning of the major premiss, the left premiss. Then we had the identity preserving character with respect to the function, which said that

$$\frac{f = g : (x : \alpha)\beta \qquad a : \alpha}{f(a) = g(a) : \beta(a/x)}$$

Again this follows immediately from the meaning of the major premiss. I just want to remark here that instead of having these two rules, you can condense them into one rule, if you want, and that rule would be

$$\frac{f = g : (x : \alpha)\beta \qquad a = c : \alpha}{f(a) = f(c) : \beta(a/x)}$$

This rule implies both of those because of reflexivity, and conversely, those two imply this one. So this is all there is to say about application.

Now we come to abstraction, functional abstraction. Here there will be a novelty in my presentation, since the rule for functional abstraction will actually be derived from two underlying rules. The usual rule for functional abstraction is as follows:

$$\begin{array}{c}(x : \alpha)\\ \vdots\\ \dfrac{b : \beta}{(x)b : (x : \alpha)\beta}\end{array}$$

If b is an object of type β, depending on a variable x ranging over type α, then we may abstract b with respect to x, thereby getting an object of the function type $(x : \alpha)\beta$, that is, thereby getting a function in the modern sense. Abstraction is thus the operation which takes a function in the old-fashioned sense into the corresponding function in the modern sense. This is the usual rule of functional abstraction, or lambda abstraction.

I am going to introduce a novelty in the presentation by making the step from the premiss to the conclusion here as two steps of some more basic new rules. So I will only arrive at the rule of functional abstraction after having presented these new rules.

The first new rule could conveniently be called argument removal. It says that if we have proved $f(x) : \beta$ under the assumption $x : \alpha$, and f does not depend on x, which syntactically means no free occurrence of x in f, then I may leave out the argument x and conclude $f : (x : \alpha)\beta$,

$$\begin{array}{c}(x : \alpha)\\ \vdots\\ \dfrac{f(x) : \beta}{f : (x : \alpha)\beta}\end{array}$$

Strangely enough, this is a rule which I, at least, have never seen stated before.

If you remember what I did informally at one stage, in my discussion of the difference between a function in the old-fashioned sense and a function in the modern

sense: I took a function in the old-fashioned sense, $x^2 + 3y$, and, as in our ordinary mathematical practice, to get a notation for the corresponding function in the modern sense, I made the definition $f(x, y) = x^2 + 3y$, and then the corresponding function in the modern sense is f simply.

If you reflect upon this, you see that two things go on. We have the function in the old-fashioned sense expressed by an open expression. Then I, firstly, make an explicit definition and, secondly, take away the arguments, x and y, which I can do now since they are in standard position. You see, why can I not just leave the arguments out in $x^2 + 3y$? It is because they are deep inside the expression, and we have the problem that Frege had and that we discussed last time. It does not work just to leave them out from $x^2 + 3y$. We have to bring them into what we could call standard position first. When they are in standard position, we can leave them out and get something unsaturated in Frege's terminology, namely f.

As usual, it is so difficult just to see what one is actually doing, I mean, reflect properly upon what one is actually doing. In that little example I was displaying very clearly what goes on here, and there should be a formal rule which allows you to do precisely what I did informally there. That formal rule—or, there are two formal rules: one is the rule of explicit definition, which will come maybe after the break, and the other is this rule of passing from a function in the old-fashioned sense that has got its argument in standard position to the corresponding function in the modern sense, and that is exactly what the argument removal rule allows you to do. Since I have decided to use the Schönfinkel device of representing functions of several arguments as iterated functions of one argument, I can do here with just one argument. In the definition of f above, instead of the two arguments x and y, I would have to write $f(x)(y)$, and it would need two applications of the rule of argument removal.

As you have seen, all rules come in pairs, because there is always a corresponding identity rule. In this case, the corresponding identity rule will say that if we have $f(x) = g(x) : \beta \quad (x : \alpha)$, then we can conclude that f and g are identical functions of the function type $(x : \alpha)\beta$,

$$\frac{\begin{array}{c}(x : \alpha)\\ f(x) = g(x) : \beta\end{array}}{f = g : (x : \alpha)\beta}$$

This rule, on the other hand, is well-known from combinatory logic: it is the typed version of the rule that is labelled ζ in combinatory logic and usually referred to as the extensionality rule.

With respect to combinatory logic, maybe I should say once and for all that what I am doing here is, if you want, dependently typed lambda calculus and combinatory logic. There was a design choice made at a very early stage in combinatory logic, namely of trying to have the convertibility relation to be untyped. You had an untyped universe of expressions, called obs in combinatory logic, and you had the convertibility relation, which corresponds to my identity relation, as a relation over that untyped universe of, in general, meaningless expressions. That was a design

choice. One hoped to separate the problems that had to do with the convertibility relation from the problems that had to do with typing, so Curry consciously started to work on the convertibility relation first and then hoped to include the typing later on. In retrospect we can see that that was not the right design choice, because the very idea of, not only forming meaningless expressions, but even talking about meaningless expressions, is not a good one. I mean, it is in the very nature of an expression that it should express something: it is not an expression unless it expresses something, which is to say that it is meaningful. In order to avoid at all forming meaningless expressions, we have to type also the identity relation, as I call it, or convertibility relation, as it is called in combinatory logic. I have therefore to redo, so to say, a lot of combinatory logic and lambda calculus here in the typed fashion. Of course, this does not mean that what was done in combinatory logic is not of great value, because we can take over a lot from combinatory logic. There is, however, this fundamental difference, that the identity is now typed all the time. I hope that my discussion of why identity has to be taken to be typed—type- or category relative—should have convinced you of that necessity.

What about the justification of these two rules? The simplest way to justify them for me would be slightly to change the definition of what a function in the modern sense is, and simply say

$$f : (x : \alpha)\beta \text{ means that } f(x) : \beta \quad (x : \alpha)$$

If I give that meaning explanation, then it is clear that I may make the step in the first rule, because the conclusion has the same meaning as the premiss. Correspondingly, if I had explained identity between functions simply by saying

$$f = g : (x : \alpha)\beta \text{ means that } f(x) = g(x) : \beta \quad (x : \alpha)$$

then the extensionality rule is trivially valid.

I think this would be the preferable way of doing it, so we now have to check that these meaning explanations agree with the ones that I gave last time. We then merely have to recall what a function in the old-fashioned sense was. So what does the judgement $f(x) : \beta \quad (x : \alpha)$ mean? Well, it means, first of all, that, if we take an arbitrary object a of type α, and substitute a for x in $f(x)$, and here f does not depend on x, that was a condition in the rule, then we just get $f(a)$ of type $\beta(a/x)$. The second condition was that if a and c are identical objects of type α, then when we substitute them for x in $f(x)$, we should get identical objects. What do we get under that substitution? Well, when we substitute a for x, we get $f(a)$, and when we substitute c for x, we get $f(c)$. These should be identical objects of type $\beta(a/x)$. This is, however, exactly the definition, or meaning explanation, that I gave last time of what a function in the modern sense is. So surely, these two agree.

Of course, if I choose this alternative, then the very notion of function in the modern sense becomes conceptually posterior to the notion of function in the old-fashioned sense, because I explain the meaning of f of type $(x : \alpha)\beta$ in terms of $f(x)$ of type β, depending on x of type α, which is a function in the old-fashioned

sense. You may say that it looks as if I, so to say at any cost, want to make it look as if the function in the modern sense is posterior to the notion of function in the old-fashioned sense. If you want to avoid that conclusion for as long as you can, then define function in the modern sense directly as I did last time. Then, at least, that definition is not posterior to the definition of function in the old-fashioned sense, because they do not depend on each other. Neither depends on the other if you make that choice. But we shall see in a while that, nevertheless—and I think I said that already last time—the notion of function in the old-fashioned sense is the more basic notion, because even if we try to avoid it at this point, we will see in a while that we will never get any function in the modern sense: we cannot construct any functions in the modern sense unless we have functions in the old-fashioned sense to construct them from.

([B.G.S.:] "So the expression $f(x)$, at that stage, that can only be understood as the expression consisting of f, left application bracket, x, right application bracket, and that is purely formal in that sense." Yes, and application is purely formal in that sense, not only when you have a variable there: in general, it is purely formal.) Yes, this is an important point, actually. You see, there is a fundamental difference between, say $f(a)$, and if you take some logical operation, say $A \supset B$, or some quantifier, $(\forall x)A$. When you have an operator here, that operator gives a meaning contribution to the whole. The parts here will have certain meanings, and then the whole will get a certain meaning dependent on the meanings of the parts, but there will also be a meaning contribution from the operator. If you analyze implication, for instance, using the standardized notation that I have, Polish notation, $\supset(A, B)$, then we still of course have a meaning contribution coming from the implication: this is just a notational difference. There is, however, no meaning contribution coming from the application as such: there is, so to say, no meaning contribution coming from the parentheses here, from the parentheses and the comma. You must not think of $f(a)$ here as: you have the meaning of a and the meaning of f, and then you get a new meaning contribution when you apply f to a. That is not the case, because, if that were the case, then we ought to write some operator, apply say, of f and a, apply(f, a), to indicate that we have a meaning contribution coming from this operation. At some stage, this has to stop, because otherwise we would get a new operator with a new meaning contribution all the time. At some stage we shall simply have to have a way of putting our expressions together without getting any meaning contribution from just putting them together, and application is already that. It is not an operation in the sense of an operator, but a process of putting f and a together into a whole.

By changing the meaning explanation for the notion of function in the modern sense in the way I suggested here, the two new rules of argument removal were trivially validated. Those two rules do, however, not suffice to get the rule of lambda abstraction, or functional abstraction. In order to derive that, we need another new rule, which could naturally be called the argument move rule. As you saw in the example with $x^2 + 3y$, before we could get the function in the modern sense, we had to move the argument places labelled by x and y to the standard

position, and that is what this new rule allows you to do,

$$\frac{\begin{array}{cc}(x:\alpha)\\ a:\alpha \qquad b:\beta\end{array}}{\begin{cases}((x)b)(a):\beta(a/x)\\ ((x)b)(a)=b(a/x):\beta(a/x)\end{cases}}$$

If you have an object a of type α, and you have a function b of type β, depending on x of type α, then I introduce a new combination of signs here, $((x)b)(a)$, and I say that this is to be an object of type $\beta(a/x)$—is to be, because as yet you cannot know what object of type $\beta(a/x)$ this is. That is explained by the second conclusion here. There are therefore really two rules involved here. The premisses are the same, but the conclusion of the first rule is $((x)b)(a) : \beta(a/x)$, and the conclusion of the second rule is $((x)b)(a) = b(a/x) : \beta(a/x)$. So I am making a nominal definition here with $((x)b)(a)$ as definiendum and $b(a/x)$ as definiens. In order for that nominal definition to work, we must of course know already that the definiens is indeed an object of type $\beta(a/x)$. But we know that from the two premisses and the substitution rule. So this is indeed an object of type $\beta(a/x)$. I also know what it means to be the same object of type $\beta(a/x)$, because $\beta(a/x)$ is a type, and it belongs to the definition of a type that we know what it means for two objects of the type to be the same. I cannot say that $((x)b)(a)$ is to be the same object of type $\beta(a/x)$ as $b(a/x)$ unless I know what same means here. But we do know what same means, because $\beta(a/x)$ is a type.

So this is the argument move rule, and as in the argument removal rules the second one was well-known, namely, the ζ-rule, in this case also, the second one here is well-known: it is the rule of β-conversion, the rule labelled β in combinatory logic. But the first rule is new. It is not new in the sense that you cannot derive this rule from the rule of abstraction and the rule of application. You certainly can do that, but I am reversing the order here: I am saying that this rule is the more basic one and that we should derive abstraction by means of this new rule.

([B.G.S.:] "It seems that this has got something to do with the fact that one forgets the application criteria. I mean, Geach, in his exposition of the dependence of identity on a type, only stresses the identity criteria, but forgets about the application criteria, and therefore gives very strange examples such as the surman in Leeds and so on. You postulate an equivalence relation, and you forget about the equivalence class. In the previous case it was also just the identity rule, and the application rule came first, but that was not known in combinatory logic, so I think that it is a well-known, well not a well-known phenomenon, but the philosophical debate shows the same predilection for the identity criteria above the application criteria." So the application criteria always have to come first. "Yes.") Yeah, and I have put the brace here, of course, to show the interconnectedness of these two conclusions. On the one hand, the second conclusion here serves as the semantical explanation of the first. When you read the first conclusion, you ask yourself immediately, Oh, what object of type $\beta(a/x)$ is this? I have never seen it before,

so tell me what it is! The explanation then comes below: it is the same object as this one, which you already are familiar with. So that is the dependence in this direction. On the other hand, we have of course also a converse dependence, because a judgement of the form $a = b : \alpha$ always has as presuppositions that $a : \alpha$ and that $b : \alpha$, so this is a presupposition of the second conclusion, and that is why they should be listed in this order.

After the break, I will derive the abstraction rule from the new rules of argument removal and of argument move, and I would also like to ask if there is anyone of you who knows even the slightest thing about Chomsky's last theory from about ten years ago, where the word 'move' plays a crucial role. I would be very glad to know.

What are the rules of abstraction? First of all, if b is of type β in the context $x : \alpha$, then $(x)b$ is an object of the function type. As usual, we must have the corresponding identity rule, which says that if b and d are identical objects of type β, then $(x)b$ and $(x)d$ are identical objects of the function type, so that we know that we may make replacements under the abstraction operator,

$$\frac{\begin{array}{c}(x:\alpha)\\ b:\beta\end{array}}{(x)b : (x:\alpha)\beta} \qquad \frac{\begin{array}{c}(x:\alpha)\\ b = d:\beta\end{array}}{(x)b = (x)d : (x:\alpha)\beta}$$

The second rule here has the name ξ in combinatory logic, if I remember correctly, and the first rule is the typed version of the rule of lambda abstraction, or functional abstraction.

It seems to me appropriate still to call it lambda abstraction, although there is no λ left in the notation I am using. I am using the pure abstraction notation of combinatory logic, $(x)b$, rather than Church's λxb, except that in combinatory logic you have square brackets, $[x]b$. Just as in the application case that we discussed at some length before the break, there is no meaning contribution coming from the abstraction itself. The abstraction operation should rather be thought of as some kind of rearranging of the expression. That is the reason why I do not want to have an operator symbol like λ for abstraction. Then we have the symbol λ left to use as a particular constant, namely the analogue of Frege's Wertverlauf operator.

The two rules governing abstraction can be derived, as I have said, from the argument move and removal rules. Let us begin with the first one,

$$\dfrac{\dfrac{\begin{array}{cc}x:\alpha\ (2) & \qquad b:\beta\end{array}}{\left\{\begin{array}{l}((x)b)(x):\beta\\ ((x)b)(x) = b:\beta\end{array}\right.}\ (1)}{(x)b : (x:\alpha)\beta}\ (2)$$

$$(x:\alpha)\ (1)$$

First of all, I have the premiss $b : \beta$ under the assumption $x : \alpha$. Then I make an argument move, so I get $((x)b)(x)$ of type $\beta(x/x)$. One of our substitution

rules says that β is the same as $\beta(x/x)$, so that substitution has no effect. The second conclusion is that $((x)b)(x)$ is equal to $b(x/x)$, thus b, so this is what the rule of argument move gives us. At this inference, the assumption labelled (1) is discharged. Now proceed from the first conclusion, $((x)b)(x) : \beta$, by the argument removal rule to $(x)b : (x : \alpha)\beta$, discharging the assumption labelled (2).

For the second rule, we have of course to use the second conclusion, $((x)b)(x) = b : \beta$. As presuppositions of the premiss $b = d : \beta$, we have both $b : \beta$ and $d : \beta$ under the assumption $x : \alpha$. We can derive the same conclusion here with d instead of b, so we also have $((x)d)(x) = d : \beta$, under the assumption $x : \alpha$. Now use the premiss, symmetry and transitivity, and you get $((x)b)(x) = ((x)d)(x) : \beta$ under the assumption $x : \alpha$. The ζ-rule, the extensionality rule, now allows you to remove the argument, so we get the desired conclusion of the ξ-rule. Both the rules of λ-conversion, or λ-abstraction, are thus valid as derived rules.

Since this is the place where abstraction has come into the language, maybe it would be a good point to say a little bit more, in addition to the discussion we had last time, about where the idea of pure functional abstraction comes from. Functional abstraction is tied to the idea of function in the modern sense, because abstraction is the operation which allows you to go from a function in the old-fashioned sense to the corresponding function in the modern sense. Functions in the modern sense are needed—maybe too strong to say that they are needed, but at least they help, the concept helps greatly—when you come to have functions as arguments of other functions, that is, when you come to deal with functionals. It is therefore clear that one has to go back and see when functions of functions were introduced for the first time, and that is the same question as asking when variable-binding operations were introduced for the first time. As I said in my previous lecture, that is certainly in the notations from the differential and integral calculus introduced by Leibniz. There is no doubt, I think, that that is the first place where you have this extremely powerful notational innovation of variable-binding operators.

To see the relation between Leibniz's notation and the modern notation coming from lambda calculus and combinatory logic, let me show you how to write

$$\frac{df(x)}{dx}$$

in a modern notation. We analyze this as a differential operator, D, operating on the function $(x)f(x)$, so that is the derived function, $D((x)f(x))$ and then we have to take the value of the derived function at x itself again, $D((x)f(x), x)$. If we abbreviate $(x)f(x)$ as f, we get $D(f, x)$, and this, of course, is the same as the notation $f'(x)$,

$$\frac{df(x)}{dx} = D((x)f(x), x) = D(f, x) = f'(x)$$

A strange thing about the traditional notation for the derivative is that there is no free occurrence of x, so if we want to have the derivative, not at x, but at some point a, we know that the only way we can handle it with this old-fashioned notation is

to write something like

$$\left.\frac{df(x)}{dx}\right|_{x=a}$$

This is of course the same as $D(f,x)(a/x)$, which is $D(f,a)$, or $f'(a)$.

Likewise, the definite integral,

$$\int_a^b f(x)dx$$

is analyzed, as I said last time, as an integral operator, which is a functional, having the function in the modern sense, $(x)f(x)$, as first argument, and a and b as the second and third argument. So you get $I((x)f(x),a,b)$, which we can write simply as $I(f,a,b)$.

This is essentially all Leibniz, and, of course, there is no trace here of the pure abstraction operation, so surely we had nothing like this in the notations for the derivative and the integral. This modern analysis of the notation belongs, as I said, to lambda calculus and combinatory logic. It is to be found in Curry and Feys, *Combinatory Logic*, but surely also already earlier in the 1940s, or maybe in Church's papers from the 1930s.

Then there was the question, Did Frege have functional abstraction, the pure functional abstraction operation? I think I have said quite correctly that he introduced more variable-binding operations, but he did not have pure functional abstraction. In modern notation, he introduced the universal quantifier, $(\forall x)\Phi(x)$, which in this modern analysis we should think of as an operation, $\forall$, applied to a function in the modern sense, $(x)\Phi(x)$, which then we can write just as $\forall(\Phi)$, and similarly with the existential quantifier. The other new variable-binding operation that he invented was the Wertverlauf operation. He introduced the notation $\dot{\varepsilon}\Phi(\varepsilon)$, where Φ is a function from objects to objects in his terminology, from Gegenstände to Gegenstände, and this is to be a new object, a new Gegenstand. It should therefore really be analyzed as some operator, and now I call it λ, which takes the function Φ, which is a unary function from objects to objects, into an object, $\lambda((x)\Phi(x)) = \lambda(\Phi)$. Frege did not have the symbol λ, but the spiritus lenis is awkward when it stands alone. So I think one must say that Frege introduced new variable-binding operations, but that he did not have the idea of pure functional abstraction.

The first time that functional abstraction and its dual operation of application is recognized is in connection with class abstraction, that is, abstraction of a propositional function. That is Peano, and then Russell and Whitehead took it over in the *Principia*. If we have some proposition P depending on a variable x, then Peano writes

$$(x \ni P(x))$$

for the class of all x's such that $P(x)$. In the *Principia*, this sign was changed, just a symbolic change, to $\hat{x}P(x)$. The winning notation, however, in the sense that most mathematicians use it nowadays, is

$$\{x \mid P(x)\}$$

where there can also be a colon or semi-colon. The modern analysis of this, that is, the analysis in the notation of combinatory logic and lambda calculus, would be $(x)P(x)$, so here we have the pure abstraction operation.

Peano consciously chose $\ni$, because it was epsilon, $\in$, the other way around. By taking a class C and an individual a, you get the proposition $a \in C$, which says that a belongs to the class C. The modern analysis of the epsilon is just application, so the modern analysis of $a \in C$ is $C(a)$, the propositional function C applied to a, the function in the modern sense applied to a. If for C we choose $(x \ni P(x))$, what we get is $((x)P(x))(a)$, and that is the result of substituting a for x in $P(x)$, which is $P(a)$, or we could get it by saying that $(x)P(x)$ is just P. In any event, the proposition

$$a \in (x \ni P(x))$$

is just $P(a)$.

So here we have the pure functional abstraction. Of course, there is the question now, Did they think about classes in this way? I mean, surely this is the good way of dealing with classes, but one would have to check if their way of dealing with classes is more in this way... ([B.G.S.:] "I think that in those days, the two notions of class, that of extension of a property and set built out of something, were conflated to such a degree that one could say that they had both these notions conflated into one, and this is mustering out the extension of a property notion of a class." Yes.) Anyway, the dual character of the abstraction and application was clearly recognized here. At least the modern analysis of Peano's notations here would be in terms of the pure functional abstraction and application, and we recognize then the law of β-conversion in the equation

$$a \in (x \ni P(x)) = P(a)$$

But certainly it was only abstraction of propositional functions, it was not abstraction of functions in general. The idea of the pure functional abstraction in general, not only of propositional functions, that is clearly due to Church, and to my mind Church's greatest contribution to logic.

There now remains only one law, which is the law of η-conversion. I have applied it here several times already, namely in identifying $((x)P(x))(a)$ with $P(a)$, $(x)P(x)$ with P, and $\lambda((x)\Phi(x))$ with $\lambda(\Phi)$. Again it is a derived law, so no new axioms here. It says that—and now I will take advantage of my new argument removal rule—from the premiss $b(x) : \beta$, for x of type α, where b does not depend on x, we can conclude that $(x)b(x)$ is identical to b as objects of the function type $(x : \alpha)\beta$,

$$\frac{\begin{array}{c}(x : \alpha)\\ b(x) : \beta\end{array}}{(x)b(x) = b : (x : \alpha)\beta}$$

This is the well-known rule which is called η in combinatory logic, the rule of η-conversion. (I will comment on the use of the word conversion here later on when I discuss synonymy.)

How do you derive it? First of all, a remark. Because of the argument removal rule, the premiss is inter-derivable with $b : (x : \alpha)\beta$. Namely, from $b(x) : \beta \;\; (x : \alpha)$ I get $b : (x : \alpha)\beta$ by the argument removal rule, and conversely from this, assuming $x : a$, I get $b(x) : \beta$ by application. So these are inter-derivable, which is one of the reasons for this new rule of argument removal, that we can pass not only in that direction but also conversely. And how do we prove it? Well, apply the argument move rule to the premisses $b(x) : \beta \;\; (x : \alpha)$ and $x : \alpha$. We can then conclude that $((x)b(x))(x)$ is of type β with x substituted for x, and the second conclusion is that by β-conversion this is equal to $b(x)$, with x substituted for x, that is just $b(x)$, which is of type β. And when applying the argument move rule we discharge the assumption $x : \alpha$ to the right,

$$\frac{\begin{array}{cc} & (x : \alpha) \\ x : \alpha & b(x) : \beta \end{array}}{\begin{cases} ((x)b(x))(x) : \beta \\ ((x)b(x))(x) = b(x) : \beta \end{cases}}$$

So we now have a proof of the last line here from the assumption $x : \alpha$. By the extensionality rule, the ζ-rule, we can then take away the arguments here and conclude that $(x)b(x)$ and b are identical functions in the modern sense, that is, objects of the function type $(x : \alpha)\beta$.

Now I have derived all the usual laws of combinatory logic, and there remain fifteen minutes to be devoted to the subject of explicit, or abbreviative, definitions. We have that on two levels here. Firstly, we have judgements saying that something is a type and that two types are identical, and since we have the possibility of saying that types are identical, we have the possibility of making abbreviations of types. Secondly, we have the corresponding object-level judgements which say that two objects of a type are identical, and since we have that identity between objects of a type, we also have the possibility of introducing abbreviations for objects. Let me begin with the type level.

It should be said immediately here that what I will present is not one rule, because whenever you make an abbreviative definition, you introduce a new rule into your language, which means a new axiom or a new rule. The system thus expands each time you introduce an abbreviative definition. On the other hand, we have a general description of the pattern according to which we are allowed to make such extensions, and I tend to use the word scheme for this, rather than rule, since it is, so to say, one level above rule: it is a scheme that has rules as instances.

It is therefore the scheme for making explicit definitions that I am describing here, and the scheme, on the type level, is this, that if you have a type β depending on certain variables x_1 of type α_1 up to x_n of type α_n, and no others—so this is important here: I am displaying all the variables on which β may depend—then I can introduce an abbreviation for β. The first way of doing it would be to say that now I introduce, say, an abbreviation Φ for β, but I have to display the arguments

on which β depends, and if I have opted for the Schönfinkel device of writing many-place application as iterated one-place application, I have to write $\Phi(x_1)\dots(x_n)$, although it may look horrible to you. (Maybe I will devote a lecture after the break to explaining what this notation looks like if we take many-place abstraction and application as primitive and do not use the Schönfinkel device, because, surely, one does not like this, though it is undeniably elegant.) So we would say that we introduce $\Phi(x_1)\dots(x_n)$ as an abbreviation of β, which is to say that this is to be a type, and its meaning is given by saying that it is to be the same type as β,

$$\frac{\begin{array}{c}(x_1:\alpha_1,\dots,x_n:\alpha_n)\\ \beta:\text{type}\end{array}}{\begin{cases}\Phi(x_1)\dots(x_n):\text{type}\\ \Phi(x_1)\dots(x_n)=\beta:\text{type}\end{cases}}$$

The problem with formulating the scheme in this way is that that would break a fundamental principle in the formulation of the rules of a language, namely, that all rules should be closed under substitution. In particular, all defining equations should be closed under substitution, so we may substitute for the x's, but then the result would no longer be an instance of this scheme. The proper way of formulating the scheme is therefore not like this, but—since we also have to perform a substitution here—as follows:

$$\frac{\begin{array}{cc}(x_1:\alpha_1,\dots,x_n:\alpha_n)\\ a_1:\alpha_1\ \dots\ a_n:\alpha_n(a_1/x_1,\dots,a_{n-1}/x_{n-1}) & \beta:\text{type}\end{array}}{\begin{cases}\Phi(a_1)\dots(a_n):\text{type}\\ \Phi(a_1)\dots(a_n)=\beta(a_1/x_1,\dots,a_n/x_n):\text{type}\end{cases}}$$

Of course, this contains the previous rule as a special case: we just choose $x_1,\dots,x_n$ for $a_1,\dots,a_n$, and then that substitution has no effect on β.

So here is the scheme: whenever I have a type β depending upon certain variables, I allow myself to introduce a new symbol Φ and say that Φ applied to n arguments of the appropriate types is a type, namely, the same type as β with that substitution carried out. One can give the same comments here as I gave before the break. Whenever you have a nominal definition—and this is of course a nominal definition, it has this structure: first a typing clause, and then an identity clause—and when you read the typing clause, your first question is, Oh, here is a new notation that I do not know, and it is said that this is a type, so what type is it? The answer is that it is the same type as β with the substitution carried out, and I can say that because this is a type by the premisses, and I have already laid down what identity between types means.

What is it, really, that I am saying contentually when I am putting $\Phi(a_1)\dots(a_n)$ by definition equal to $\beta(a_1/x_1,\dots,a_n/x_n)$? Well, remember what it means for two types to be the same. The form of judgement $\alpha=\beta$: type meant that we can

conclude in both directions here:

$$\frac{a : \alpha}{a : \beta} \qquad \frac{a = b : \alpha}{a = b : \beta}$$

With the terminology that Göran told us last time is due to Ryle, we can say that $\alpha = \beta$: type is an inference ticket which licenses inferences in both directions here. The meaning of the second clause here is precisely that we are licensed to conclude in both directions here:

$$\frac{a : \Phi(a_1) \ldots (a_n)}{a : \beta(a_1/x_1, \ldots, a_n/x_n)} \qquad \frac{a = b : \Phi(a_1) \ldots (a_n)}{a = b : \beta(a_1/x_1, \ldots, a_n/x_n)}$$

That is the meaning of setting $\Phi(a_1) \ldots (a_n)$ identical to $\beta(a_1/x_1, \ldots, a_n/x_n)$. Now you see how that ties together with my comments about the importance of the rule of type identity in the very beginning, because it is the rule of type identity that, together with this defining equation, permits you to make those inferences.

([B.G.S.:] "You do not consider it necessary to indicate that this is a stipulative equality rather than something which has to be proved or understood?" Yes, so I think the natural way of indicating that is the way used in ordinary mathematics, namely, you write

$$\text{Def. } \Phi(a_1) \ldots (a_n) = \beta(a_1/x_1, \ldots, a_n/x_n) : \text{type}$$

to indicate that this is the place where the new symbol is introduced. But observe, there is a typing part also, so there is the question whether all of this should be called the definition or just the second clause. It is probably better to call all of it the definition.)

Then there is the observation that one abbreviation that I have used on the type level is not an instance of this general scheme, and that is the abbreviation that I used for the non-dependent function type. Remember I introduced here the abbreviation

$$\frac{\alpha : \text{type} \qquad \beta : \text{type}}{\begin{cases} (\alpha)\beta : \text{type} \\ (\alpha)\beta = (x : \alpha)\beta : \text{type} \end{cases}}$$

If α is a type, and β is a type, then I introduce $(\alpha)\beta$ as an abbreviation of $(x : \alpha)\beta$, where I have to choose x so that it does not clash with the variables on which α and β depend. I just want to remark that this is not an instance of the general scheme, and the reason is that α and β are types, and in the general scheme what can occur as arguments under Φ are only objects, not types. If I wanted this to be an instance of the general scheme, then I would have to have type variables, since the variables on which $(\alpha)\beta$ depends would have to range over types, and that is something I do not have. So this definition, although clearly correct, is not an instance of the general scheme.

And then finally just a few examples of this scheme. Without stating the general scheme, I already gave a couple of instances. Let us recall what they were.

One was this:

$$\frac{\begin{array}{cc} & (X : \text{set}) \\ A : \text{set} & (X)\text{prop} : \text{type} \end{array}}{\begin{cases} \text{class}(A) : \text{type} \\ \text{class}(A) = (A)\text{prop} : \text{type} \end{cases}}$$

If we assume that X is a set, then (X)prop is a type. (Here I am using the abbreviation I just talked about.) Then, if A is a set, I can use the scheme to introduce a new symbol and say that class(A) is, first of all, a type, and it is by definition equal to (A)prop. That was my definition of class(A).

A second example could be

$$\frac{(\text{prop})(\text{prop})\text{prop} : \text{type}}{\begin{cases} \text{connective} : \text{type} \\ \text{connective} = (\text{prop})(\text{prop})\text{prop} : \text{type} \end{cases}}$$

If I have the ground type prop, of propositions, then surely I can form the type (prop)(prop)prop. The scheme with $n = 0$, that is, no arguments, no parameters, then allows me to introduce a new symbol, connective say, and say that it is a type that by definition is equal to (prop)(prop)prop.

And then we had my last example—I just want to see to it that it is an instance of the scheme—when I introduced quantifier as an abbreviation of a certain complex type,

$$\frac{(X : \text{set})(\text{class}(X))\text{prop} : \text{type}}{\begin{cases} \text{quantifier} : \text{type} \\ \text{quantifier} = (X : \text{set})(\text{class}(X))\text{prop} : \text{type} \end{cases}}$$

Here we have a type, $(X : \text{set})(\text{class}(X))$prop, a complex type, which is so long that it would be tiring to write it out all the time. Since it does not depend on any variables, I can introduce a new constant, quantifier, and say that quantifier is a type, namely a type that by definition is equal to the one given here.

So then more about explicit definitions on the object level next time, and some general comments of a philosophical character on the meaning of nominal definitions: are nominal definitions definitions of the meaning of an expression or definitions of an object?

Lecture 6, 4.11.93

If I remember correctly, we ended last time with the scheme of explicit or abbreviative definitions of types. There is a corresponding scheme on the object level which looks as follows:

$$\frac{\begin{array}{ll} & (x_1 : \alpha_1, \ldots, x_n : \alpha_n) \\ a_1 : \alpha_1 \quad \ldots \quad a_n : \alpha_n(a_1/x_1, \ldots, a_{n-1}/x_{n-1}) & b : \beta \end{array}}{\begin{cases} f(a_1) \ldots (a_n) : \beta(a_1/x_1, \ldots, a_n/x_n) \\ f(a_1) \ldots (a_n) = b(a_1/x_1, \ldots, a_n/x_n) : \beta(a_1/x_1, \ldots, a_n/x_n) \end{cases}}$$

Firstly, b is an object of type β, depending on some variables, say x_1 of type α_1 up to x_n of type α_n—and here it is essential that these are all the variables on which b and β depend, so no hidden variables. Secondly, we have objects a_1 of type α_1 up to a_n of type $\alpha_n(a_1/x_1, \ldots, a_{n-1}/x_{n-1})$. Then we may introduce a new symbol, a defined constant, let us call it f, and give it the following definition: $f(a_1) \ldots (a_n)$—again we have this unnatural feature that I am using, the Schönfinkel device—is an object of type $\beta(a_1/x_1, \ldots, a_n/x_n)$, and it is by definition the same object as $b(a_1/x_1, \ldots, a_n/x_n)$, which, because of the right-hand premiss, is an object of type $\beta(a_1/x_1, \ldots, a_n/x_n)$.

I have already said that in the right-hand premiss, you must not have any dependence on variables other than those that are explicitly shown. For the remaining premisses, however, there may be dependence on hypotheses that I have not shown explicitly here to simplify matters. This is therefore natural-deduction style writing: there may be hidden assumptions.

Let us take three small examples of this scheme to show how it works. Of course, I have no constants yet, but I will take for granted that I have already introduced a few of them. In particular, in the first two examples here, I presuppose that I already have access to the set of natural numbers and the particular natural number 0 and the successor operation,

$$\mathbf{N} : \text{set} \qquad 0 : \mathbf{N} \qquad \frac{a : \mathbf{N}}{\mathbf{s}(a) : \mathbf{N}}$$

These rules allow us to derive $\mathbf{s}(0) : \mathbf{N}$, and then we can use the scheme of explicit definition, which will allow us to introduce a new symbol, 1, and say that it is typed by $\mathbf{N}$, and moreover that it is by definition equal to the already constructed natural

number $\mathbf{s}(0)$,

$$\frac{\mathbf{s}(0) : \mathbf{N}}{\begin{cases} 1 : \mathbf{N} \\ 1 = \mathbf{s}(0) : \mathbf{N} \end{cases}}$$

Once we have introduced 1 in this way, we can do the same thing again, that is, we can apply the successor rule with a chosen to be 1 and get $\mathbf{s}(1) : \mathbf{N}$. Then I can say: let us introduce a new symbol to abbreviate that one! It will get the type $\mathbf{N}$, of course, and by definition it is equal to $\mathbf{s}(1)$,

$$\frac{\mathbf{s}(1) : \mathbf{N}}{\begin{cases} 2 : \mathbf{N} \\ 2 = \mathbf{s}(1) : \mathbf{N} \end{cases}}$$

So here are two examples of the scheme of explicit definition. In both examples, you have $n = 0$, for the n that appears as index in the scheme.

The next example is probably well known to all of you. Suppose we have the type of propositions, and that we have also introduced the particular proposition absurdity, $\bot$, and that we have introduced implication,

$$\text{prop} : \text{type} \qquad \bot : \text{type} \qquad \frac{A : \text{prop} \qquad B : \text{prop}}{A \supset B : \text{prop}}$$

I can then make the following abbreviative definition:

$$\frac{A : \text{prop}}{\begin{cases} \neg A : \text{prop} \\ \neg A = A \supset \bot : \text{prop} \end{cases}}$$

If A is a proposition, then, by the above rules, $A \supset \bot$ is a proposition, hence I can introduce an abbreviation, $\neg A$, of that. It is a proposition, and by definition it is the same proposition as $A \supset \bot$. If I were to adhere strictly to the Schönfinkel notation, I would have to write parentheses here, $\neg(A)$.

The newly introduced f in the scheme of explicit definition is what we call a defined function constant or, in Curry's terminology, a combinator,

$$\text{combinator} = \text{defined function constant}$$

The whole combination $f(x_1)\ldots(x_n)$ is of course a name of the possibly complex function b that we have in the premiss, which is of type β and depends on the variables $x_1, \ldots, x_n$. If we omit the arguments here, however, and just look at f itself, then that is a name of the corresponding function in the modern sense which we get by abstracting b with respect to its variables, which is an object of the function type $(x_1 : \alpha_1)\ldots(x_n : \alpha_n)\beta$,

$$\begin{aligned} f(x_1)\ldots(x_n) \;\; &\text{name of} \;\; b : \beta \;\; (x_1 : \alpha_1, \ldots, x_n : \alpha_n) \\ f \;\; &\text{name of} \;\; (x_1)\ldots(x_n)b : (x_1 : \alpha_1)\ldots(x_n : \alpha_n)\beta \end{aligned}$$

What I previously called explicit, or abbreviative, definition is thus the same as what we call naming.

If we have access to functional abstraction, that is, the possibility of forming $(x)b$, then it is sufficient to have the scheme of explicit definition for $n = 0$, that is, without any arguments in the scheme. Why is that so? Well, if we have access to functional abstraction, we may pass from the premiss

$$b : \beta \quad (x_1 : \alpha_1, \ldots, x_n : \alpha_n),$$

to

$$(x_1) \ldots (x_n)b : (x_1 : \alpha_1) \ldots (x_n : \alpha_n)\beta$$

Now there is no dependence on any variables any longer, so the scheme of explicit definition with $n = 0$ allows us to introduce f as follows:

$$\begin{cases} f : (x_1 : \alpha_1) \ldots (x_n : \alpha_n)\beta \\ f = (x_1) \ldots (x_n)b : (x_1 : \alpha_1) \ldots (x_n : \alpha_n)\beta \end{cases}$$

If we introduce f in this way, then, clearly, if we apply f to the objects which are given as the left-hand premiss in the rule of explicit definition—we had a_1 of type α_1 up to a_n of type $\alpha_n(a_1/x_1, \ldots, a_{n-1}/x_{n-1})$—then, first of all, we get

$$f(a_1) \ldots (a_n) : \beta(a_1/x_1, \ldots, a_n/x_n)$$

and, moreover, $f(a_1) \ldots (a_n)$ is definitionally equal to $((x_1) \ldots (x_n)b)(a_1) \ldots (a_n)$. If we now use β-conversion, which we already have access to, we find that this is equal to $b(a_1/x_1, \ldots, a_n/x_n)$ of type $\beta(a_1/x_1, \ldots, a_n/x_n)$,

$$\begin{aligned} f(a_1) \ldots (a_n) &= ((x_1) \ldots (x_n)b)(a_1) \ldots (a_n) \\ &= b(a_1/x_1, \ldots, a_n/x_n) : \beta(a_1/x_1, \ldots, a_n/x_n) \end{aligned}$$

So you see, if you have access to functional abstraction, you can make do with the scheme of abbreviative definition with $n = 0$, but that is only if you have access to functional abstraction. I will explain in a moment what we do if we use Curry's approach with combinators, in which case we do not have functional abstraction. Then it is absolutely necessary that the scheme is formulated for arbitrary n.

In the scheme of explicit definition I had the requirement that all the variables on which the expression which is to be abbreviated depends must be shown explicitly in the definiendum. We had b of type β, and I said that $x_1, \ldots, x_n$ must be all the variables on which it depends. I am, however, not, of course, requiring that all those variables actually occur in b. For instance, if I thin the context by introducing extra variables, then b will not depend on those. There is absolutely no harm in this. The important thing is that all the variables on which b depends are shown explicitly in the definiendum, that is, when you write the definition as

$$f(x_1) \ldots (x_n) = b : \beta \quad (x_1 : \alpha_1, \ldots, x_n : \alpha_n)$$

The reason why that is essential is that I have opted for the usual approach of treating substitution as an informal operation belonging to the metalanguage, rather than as a formal operation in the language. If I were to introduce an abbreviative definition without showing explicitly all the variables that occur in the definiens, then I would be in trouble, because there would be no argument places

in the definiendum corresponding to those variables. For instance, if I have, say $\mathbf{s}(x)$, successor of x, and I introduce some abbreviative definition for that, say $\mathsf{one} = \mathbf{s}(x) : \mathbf{N}$, then I would be in difficulty, of course, because one here would depend on x, although the variable dependence is not shown explicitly. If I were then later to substitute for x, then I would get $\mathsf{one}(a/x)$, but I have no way of carrying out this substitution. The more general scheme of explicit definition, where you do not show the variable dependence explicitly in the definiendum requires a calculus where substitutions are made explicit, that is, as an explicit notation in the calculus. It is one of the several places where you can see the interest of developing such a calculus. Since I have opted for the standard procedure here, however, I must make this requirement.

Frege, in *Grundgesetze* § 33, the final point 7, actually required more here: he required that, not only should all the variables in the definiens also occur in the definiendum, but also vice versa. That simply does not work, because you need constant functions, for instance. You need the function of one variable which is identically equal to 0, for instance, $f(x) = 0 \ \ (x : \mathbf{N})$, and here you have a variable in the definiendum but not in the definiens. If Frege himself—who quantified over functions—had proved something for all functions, he would want to be able to insert as an instance of that universal quantifier a constant function. One thus definitely needs the scheme in this more general form, and that was corrected actually in *Principia*, in the one page essentially about definitions, page 11 to 12. ([B.G.S.:] "But you can restore the Fregean requirement by taking $0 \cdot x$, or in the general case, the general constant function, it would say $((x + 1) - x) \cdot n$. You can fiddle it if you want to." Yes, but you will not get intensionally the same function. You get only something which is extensionally the same. "True.")

Another remark I want to make is that, because of the rule of argument removal, it does not matter if I write, in the scheme of explicit definition,

$$f(x_1) \ldots (x_n) : \beta \quad (x_1 : \alpha_1, \ldots, x_n : \alpha_n)$$

or if I write

$$f : (x_1 : \alpha_1) \ldots (x_n : \alpha_n)\beta,$$

because these two are interderivable. To get the first one from the second, you just apply f to $x_1, \ldots, x_n$. In the converse direction, the rule of argument removal allows you successively to eliminate the arguments $x_1, \ldots, x_n$ in favour of the function type former on the right-hand side. So, by n argument removals, you get the corresponding conclusion here.

In particular, this means that it does not matter when I give, for instance, the definition of negation—you see, the way you spontaneously give that definition, is no doubt as I did it here before,

$$\frac{A : \text{prop}}{\begin{cases} \neg A : \text{prop} \\ \neg A = A \supset \bot : \text{prop} \end{cases}}$$

If A is a proposition, I introduce $\neg A$, which is a new proposition that abbreviates $A \supset \bot$. That is no doubt how you naturally write the definition of negation. However, you want to be able to derive that negation, by itself, is a function which takes propositions into propositions, that is, it is an object of type (prop)prop,

$$\begin{cases} \neg : (\text{prop})\text{prop} \\ \neg = (X)(X \supset \bot) : (\text{prop})\text{prop} \end{cases}$$

It should be immaterial whether we write the definition as I first did or if we do it in this way.

In one direction, it is easy to go, because if I would have defined negation in this latter way, by a higher type definition, then of course I get from this by application that $\neg A$ is a proposition and that $\neg A$ is equal to $((X)(X \supset \bot))(A)$, which by β-conversion is equal to $A \supset \bot$. So that is easy.

However, since the first definition is the natural way of doing it, we ought to be able to derive the latter from that one as well, and we shall see that that is not possible without the rule of argument removal. Applying that scheme with the premiss chosen to be the assumption $X : \text{prop}$, I get

$$\begin{cases} \neg(X) : \text{prop} \quad (X : \text{prop}) \\ \neg(X) = (X \supset \bot) : \text{prop} \quad (X : \text{prop}) \end{cases}$$

So that is by using the first definition. Argument removal then allows me to infer

$$\neg : (\text{prop})\text{prop},$$

Moreover, by β-conversion I have

$$((X)(X \supset \bot))(X) = X \supset \bot : \text{prop}$$

So $\neg X$ and $((X)(X \supset \bot))(X)$ are both equal to $X \supset \bot$, hence they are equal to each other, hence by argument removal, that is, by the ζ-rule, or extensionality rule, $\neg$ is equal to $(X)(X \supset \bot)$. In order to show the equivalence of these, I thus definitely need the rule of argument removal. It would be very awkward actually if one did not have access to it.

The next point I want to explain is the relation between the approach that Church took with his calculus of lambda abstraction and Curry's approach by means of combinators. These two approaches are equivalent in a precise sense, and I want to explain how. I especially want to explain that because—as all those of you who are trained in mathematics no doubt have observed—in ordinary mathematics, there is no established notation for functional abstraction. It is one of the mysterious things, really, that mathematicians have managed without a notation for functional abstraction, and we must understand why.

The question is therefore, What is the relation between lambda abstraction, which is Church's approach, and combinators?

$$\text{lambda abstraction (Church)} \Longleftrightarrow \text{combinators (Curry)}$$

In one direction, I have already shown it, namely, the reduction of combinators to functional abstraction. I proved a few minutes ago that if a combinator f is introduced by explicit definition—so, $f(x_1)\dots(x_n)$ was set equal to an expression b, or rather a function b in the old-fashioned sense, of type β, depending on the variables $x_1,\dots,x_n$—then that f is actually definitionally equal to $(x_1)\dots(x_n)b$, which are both of the function type,

$$f = (x_1)\dots(x_n)b : (x_1 : \alpha_1)\dots(x_n : \alpha_n)\beta$$

This shows that the combinators, Curry's combinators—which are here in a typed framework, so they are typed combinators—are expressible by means of lambda abstraction. On both sides here you have application, so you do not have to show how to interpret application.

It is the other direction which is the interesting one: it is the fact that this is possible which is the theoretical explanation why mathematicians have managed, and are still managing, without ever having seen a precise notation for functional abstraction. How do we express functional abstraction by means of combinators? Suppose that we have some function abstract, $(x)b$, of function type, $(x : \alpha)\beta$. Of course, the b here may depend on other variables besides the x which we are abstracting, so we have to be careful and show all those variables explicitly,

$$\frac{b : \beta \quad (x_1 : \alpha_1, \dots, x_n : \alpha_n, x : \alpha)}{(x)b : (x : \alpha)\beta \quad (x_1 : \alpha_1, \dots, x_n : \alpha_n)}$$

How do I express this without abstraction, using combinators instead? Well, by using the premiss instead as a premiss for an explicit definition, we get

$$\frac{b : \beta \quad (x_1 : \alpha_1, \dots, x_n : \alpha_n, x : \alpha)}{\begin{cases} f(x_1)\dots(x_n)(x) : \beta \quad (x_1 : \alpha_1, \dots, x_n : \alpha_n, x : \alpha) \\ f(x_1)\dots(x_n)(x) = b : \beta \quad (x_1 : \alpha_1, \dots, x_n : \alpha_n, x : \alpha) \end{cases}}$$

So I am making an explicit definition here, defining a new function constant, f, that is, a combinator in Curry's terminology, with $n+1$ arguments. Then I can use the rule of argument removal, and that is precisely what mathematicians do: when an argument stands in standard position, we just omit it, so we use argument removal and get

$$f(x_1)\dots(x_n) : (x : \alpha)\beta \quad (x_1 : \alpha_1, \dots, x_n : \alpha_n)$$

This is now within Curry's syntax, it is a combinator term, and we just have to see to it that that combinator term is identical to the abstract, $(x)b$. That follows because $f(x_1)\dots(x_n)(x)$ is equal to b,

$$f(x_1)\dots(x_n)(x) = b \quad (x_1 : \alpha_1, \dots, x_n : \alpha_n, x : \alpha)$$

and, on the other hand, by β-conversion, we know that $((x)b)(x)$, obtained by an argument move in my terminology, is equal to b of type β, in the same context,

$$((x)b)(x) = b : \beta \quad (x_1 : \alpha_1, \dots, x_n : \alpha_n, x : \alpha)$$

hence they are equal among themselves, by symmetry and transitivity, and hence, again by argument removal, that is, the ζ-rule, the extensionality rule, we can remove the argument x and get

$$f(x_1)\dots(x_n) = (x)b \quad (x_1 : \alpha_1, \dots, x_n : \alpha_n)$$

This shows that the function abstract can be expressed by means of combinators, using now crucially two things: explicit definition and argument removal. I need explicit definition to define the combinator f, and I need argument removal to remove the last argument.

This, as I said, is the theoretical explanation why we manage, as we still do in mathematics, without a notation for functional abstraction. The closest, maybe, that one comes to a notation for functional abstraction is in the Bourbaki notation for defining functions in the modern sense, a notation which has been widespread in the teaching of elementary mathematics in the 1960s.

Bourbaki hoped systematically to do away with the old-fashioned notion of function, hence it was thought no good to write, as one spontaneously would do—for, say, the function of two arguments that I had before, one would spontaneously write, in old-fashioned notation,

(D) $$f(x, y) = x^2 + 3y$$

(I omit type information in this discussion.) If one had not seen the Bourbaki notation, one would certainly say: let f be the function in the modern sense defined by this equation. We then have two things: we have $f(x, y)$, which is a function in the old-fashioned sense, and then we have just f, which is the corresponding function in the modern sense. Bourbaki did not want to write this, because they thought it led the thoughts in the wrong direction, namely, in the direction of functions in the old-fashioned sense. They therefore introduced this new notation:

(B) $$f : x, y \mapsto x^2 + 3y$$

The purpose of introducing this new notation was precisely to separate f from its arguments, that is, to make it absolutely clear that it is a function in the modern sense that is being defined.

That is laudable, of course, if it is a function in the modern sense that you want. But, what is really the logic of this notation? It is that you first have the function $x^2 + 3y$, which is a function in the old-fashioned sense, and then you get the associated function in the modern sense which we call f. That is the logic of it, and what does that correspond to in my symbolism? It corresponds to introducing, by explicit definition, $f(x, y)$, which is now an abbreviation of $x^2 + 3y$, and then making an argument removal to get the corresponding function in the modern sense. Surely, that is the logic behind this. You can do that either by writing definition (D) and making those two argument removals, or else, if you necessarily want to have the symbol f standing alone in order to show that it is a function in the modern sense, you can of course write also, by the scheme of explicit definition,

$$f = (x, y)(x^2 + 3y)$$

Since this uses Church's notation for functional abstraction, which one does not have access to, Bourbaki instead writes (B), where you have, as you see, exactly the same information, because in the part

$$x, y \mapsto$$

you have precisely the information that corresponds to the variable binder. ([B.G.S:] "One could say with Wittgenstein that those two expressions have exactly the same Mannigfaltigkeit." The? "The same Mannigfaltigkeit, multiplicity is the English translation." Yeah. "Do they comment that the variables x and y are bound in Bourbaki?" No, they avoid talking about syntax as much as possible.)

I am now done with everything that has to do with the function type. Maybe you think that it has been a little too much, so to say, a lot of symbolism here. It is, however, not so strange: after all, I have been dealing with the concept of function, and dealt with it in the syntactic–semantic manner in which I am dealing with all of these problems. One could maybe even say that the notion of function is the most important concept in mathematics. Of course, we cannot do with just the notion of function: we need a little bit more. We need the concept of set, and we need maybe the concept of proposition to deal with logic. But surely, there are not more notions than a few, two or three or so, at this very basic level, that is, the level of the type structure, and of those notions the notion of function is perhaps the most central one, of importance not only to mathematics, but also to linguistics. It is therefore no wonder, if the notion of function is so central, that one has to devote at least some time to explaining syntactically and semantically the precise notation that has been introduced for functions.

The lecture two weeks ago finished with a question: does a nominal definition primarily define the meaning of a linguistic expression, or does it define, on the object level, an object or, on the type level, a type? I think the most common thing to say is that it is the meaning of a name that is defined by a nominal definition, as is indicated by the very name nominal definition: nominal for nomen, that is, name. But let us reflect on what we have spontaneously said in connection with the examples that we have considered.

Let us take, on the type level, one of our examples. I introduced the type connective by the explicit definition

$$\begin{cases} \text{connective} : \text{type} \\ \text{connective} = (\text{prop})(\text{prop})\text{prop} : \text{type} \end{cases}$$

What is it that I am saying here? I am saying that connective is a type, namely, the type which is defined by saying that it is to be the same type as the type of binary propositional operators. I am not using the word expression here at all: I am speaking of types all the time. I say, connective is a type, and you ask, Oh, this, I have never seen that before, what type is it? And I am saying, well, it is the

same type as (prop)(prop)prop, which we have already introduced. Clearly, I am speaking of types here, and not type expressions.

On the object level, when I introduced the abbreviation for negation, surely I said this: let A be a proposition, then I introduce a new proposition, $\neg A$—new proposition, speaking in objective terms—so a notation you may not have seen before, so you say, What proposition is it? Well, it is the same proposition as $A \supset \bot$, which you know already to be a proposition, provided you know absurdity and implication,

$$\frac{A : \text{prop}}{\begin{cases} \neg A : \text{prop} \\ \neg A = A \supset \bot : \text{prop} \end{cases}}$$

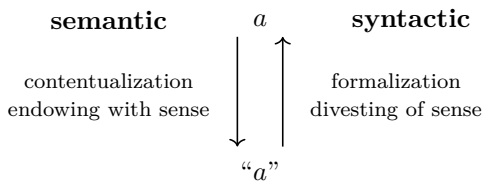
In the first case, it is a type that is defined, and in the second case, it is a proposition—or a unary propositional operation—that is defined, by the nominal definition. Therefore, if we believe that when we speak spontaneously, as I have done here, we are not speaking improperly, then no doubt, this is an indication that what is defined is not primarily the meaning of a linguistic expression, but what is defined is primarily a type, respectively an object. Of course, in defining a type or an object, we are also secondarily defining the meaning of a linguistic expression. Above, we thus defined, in the one case, the linguistic expression "connective", and in the other case, the linguistic expression "$\neg A$". That is, however, only in a secondary, and so to say somewhat trivial, sense, because of a fundamental principle which I think I will spell out now.

The principle really belongs in the part of my lectures about semantics, but a few things are needed already at this point. I will use the standard device of using quotation marks to indicate when I am speaking of expressions rather than objects. Let me then state the following general principle that, if you have a linguistic expression, "a" say, then a is the meaning, a itself is the meaning of that expression, and, conversely, a within quotation marks is the expression of a,

$$\begin{gathered} a \text{ is the meaning of } \text{“}a\text{”} \\ \text{“}a\text{” is the expression of } a \end{gathered}$$

(In the first sentence here, you have a beautiful example of an opaque context. This place, when you say that a is the meaning of "a", is an opaque context: you cannot replace a by anything which does not look exactly like a in that place.)

I am beginning to speak about expressions here, so I need to say something already at this point about expressions and objects. I mean, I have not said anything so far about what expressions are as opposed to objects, and I think the following figure is helpful here:



Taken in the ordinary way in which we take language when we use it—whether doing mathematics or reading a newspaper or whatever it is, we take it in the object directed or object oriented way, to use an expression from Husserl. We are directed, not towards the linguistic expression, but towards what it expresses, or what it says. It requires a very conscious effort in reading the newspaper to actually look at the words which are there and not at what is expressed by means of those words. ([Georgos Konstantis:] "At the meaning or at the object?" At the meaning. I am speaking of the meaning, the meaning or the object.) So surely, in the natural attitude, another expression from Husserl—or maybe I should use the German here: natürliche Einstellung—we are object directed, gegenständlich gerichtet.

Object thus, so to say, comes first here: it is from this that we start. I mean, we live in a world with tables and chairs and kitchen tools—and numbers and functions and so on in mathematics. It is a world with objects that we are in, and it requires conscious effort to change the natural attitude, in order to pay attention, not to the object, but to the look of the object, that is, to consider the object purely formally. This is the process of formalization, or divesting of sense.

[G.K.:] "What do you mean by formalization?" Well, it is the word we use in connection with mathematics and with language in general. If we start with some mathematical object, expressed by some complicated formula, or with the newspaper text, it is not in the least place easy to take this step and find the expression that expresses that meaning, because originally we were object directed and now we must look at this as a formal structure, rather than as what is meant by that formal structure. In mathematics that is comparatively easy, if you do not think of the English parts of a mathematical text, but think of the formulae. Formulae are written in such a standardized notation that it is usually quite easy to understand how they are structured. For instance, if you write $10^{10!}$ or something like that, everybody knows how this is to be structured: it has an outermost form, which is the form of exponentiation, and the base is 10 and the exponent is 10!, which has the factorial form and a part which is 10 again, and then there can be discussion as to how 10 is to be structured.

This structuring of the expression is thus usually a simple thing in the case of mathematical formulae, but if you take natural language, like the newspaper article, then this is, so to say, the big problem of linguistics: to give a grammatical analysis of the sentence, let us say one sentence from that newspaper article, to parse it or to give its grammatical analysis. It is no problem, if it is typed say, to structure the sentence as a string of typewriter signs. That is no problem at all, because they are clearly distinguished. Everybody agrees, however, that the expression as a string of typewriter signs is something completely different from the analysis of the expression that goes with its meaning decomposition, and it is only that decomposition that we are interested in from a linguistic point of view. ([B.G.S.:] "Everybody ought to agree, most certainly not everybody does agree.") Well, so in the case of a natural language, this is, so to say, the main problem, namely to find the correct grammatical analysis of what we in the first place understood without any difficulty at all. The mystery is, How can we come to understand all these

wonderful things through the newspaper? It is precisely that mechanism that we want to lay bare.

In the case of natural language, it is thus this step which is the most difficult one. If, on the other hand, you take formal languages, of which we have a plethora constructed, say from Frege onwards, then the formal structure is absolutely explicit, if those languages meet the high requirements that have been set in modern logic. Structuring the expressions is then no problem at all, because it is made explicit in the rules. On the other hand, some of these languages have semantical difficulties. That is, there is a question, although we have a complete grasp of these formal structures: is there really any object that is expressed by those formal structures? That is the semantical problem. For instance, Frege's own calculus was formally absolutely precise, but it was inconsistent, so there were errors somewhere, which means that there were expressions in his language that did not make good sense. This step, from the expression to the meaning, could therefore not be taken, and what is this step? It is the converse of formalization, so if we use the form/content duality, which is natural here, we might say that this is contentualization. The converse of formalization we might call contentualization, or endowing with sense, and here Husserl had two favourite expressions, namely, Sinnbelebung and Sinnbeseelung, that is, animation with sense.

From this figure you already see what the syntactic–semantic clarification method that I am using amounts to. You start from something here which is not clear. It is still a mystery for us how we have come to grasp this complex thing, whether it is a complicated mathematical object or a newspaper article or whatever linguistic item it is. We want to get clarification here, that is, we want to get insight into the mechanism through which we have come to grasp this object. So the original question that we are putting is very naturally put in the way that I have done it all the time, in the way of a Socratic question: What is a? Simply, What is this a here? What is 0? What is the set $\mathbf{N}$ of natural numbers? etc. The answer to the question is given by syntactic–semantic clarification, namely, by structuring a, which, as I said, is sometimes not at all trivial.

It is a great problem: structuring a so that it becomes a complex of atoms, a complex of primitives. Then we give a meaning explanation for all those primitives and see to it that those atoms are put together in such a way that the whole gets its meaning from the way it is composed. That is Frege's compositionality principle, I mean, the general linguistic compositionality principle. So, clearly, this involves two parts: it involves a syntactic part, and it involves a semantic part. These two combined actually provide an answer to the question which is originally put in terms of essences. With the question, What is a? we are asking in traditional terms for the essence of a, and the question is answered, that is, the essence of a is explained, by this combined syntactic–semantic analysis.

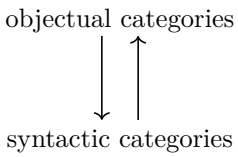
Since you asked about Husserl: there is a great affinity, at least, to—I do not dare to say whether it is exactly the same as—Heidegger's distinction between the two attitudes of Zuhandenheit and Vorhandenheit. The Zuhandenheit attitude towards the things around you, which may be ordinary tools, in metal and so on,

but it may also be our linguistic tools, which is what we are discussing here—the ordinary attitude is the Zuhandenheit attitude, where you just use them without thinking about what their formal structure or formal look is: you just take them and use them. But, of course, you can also change your attitude and, so to say, step back and look at this thing that you use daily and pay attention to its form and why it is formed in the particular way it is.

As I said, the primary thing, what we start from, is without doubt the object. It therefore has priority over the expression, and that is why Husserl insisted very strongly on the fact that an expression is not an expression unless it expresses something, that is, unless it has a sense or meaning. It is an expression in the proper sense of the word only if it has been obtained by the process of formalization from something which made sense to start with. That is why, whenever we have an expression, we can unhesitatingly talk about its meaning, because the meaning is simply the object from which the expression was obtained by the process of formalization.

I have spoken all the time here about an object and used the letter a, which I ordinarily use for object. If I had a word which combined both object and type, like an object which may be either general—that is, a type—or particular—that is, what I call object—I would have used that word. If you speak of concepts, for instance, you have a good terminological possibility, because you can speak of general concepts as opposed to specific concepts, and general concepts would be the types and specific concepts would be what I call objects here. Had I had such a terminology, I would have preferred it, but now I have to say instead that all of what I have said here applies just as well with respect to types and type expressions as with respect to objects and object expressions. The only difference would be that I would put an α here instead.

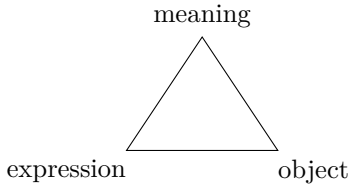
In the process of formalization, a type is turned into a type expression and an object is turned into an object expression. Most interestingly, the categories that we have on the object level, which are categories of objects, or objectual categories, they are, through this process of formalization, turned into categories of corresponding expressions instead. The category of sets, for instance, is turned, through the process of formalization, into the category of set expressions, which is a syntactic category as the ordinary terminology is. Objectual categories are thus turned into syntactic categories in this process,



This also immediately answers a question that arises most naturally in connection with syntactic and semantic categories, namely, Do we have two types of categories here, syntactic and semantic categories, or do we have three types, namely, syntactic categories, semantic categories, and what I call here objectual categories, that is,

categories of objects? With the view that I am expounding here, the meaning of the expression "a" is the object. I am using the word object in such a way here that the meaning of the expression "a" is the object from which it was obtained by formalization, that is, a is the meaning of "a", and "a" is the expression of a. It is therefore clear that, on this understanding of things, the objectual categories, categories of objects, and semantic categories are the same, because the objects are the meanings of the corresponding linguistic expressions. There are thus two types of categories, and not three.

[G.K.:] "The meaning is the same as the object, the meaning of the expression is the object?" I know the problem here. I know full well the terminological problems that we have at this point. The traditional picture, as codified in the semantic triangle, is that we have the object, the meaning and the name, or the expression,



This is the standard way of expressing oneself. For Husserl, for instance, the expression is the Ausdruck, and for meaning he uses both Sinn and Bedeutung, and then in *Ideen*, Noema, and for object he uses Gegenstand. I agree with you, this is the standard picture. But, you see, I am gradually beginning to change things. Surely, there is something in this picture. I mean, this picture has been with us through the whole tradition, essentially from Aristotle to the present day, so it is kind of inconceivable that nothing is right in it. It would be completely wild to hope that this picture was to be changed by us in such a way that we could not recognize it in whatever new way of drawing the figure that we might have. But I am going to redraw this picture a little bit.

The problem is then that to get the words—if you are changing this picture a little bit, you will have problems with some words, because they do not quite fit any longer. It is precisely such a problem that we have at the present moment. You see, I have used object for this all the time, whereas you have, quite understandably, and correctly from, so to say, a traditional point of view, tried to change that into a unit of meaning, or a meaning complex, or something like that. I will just agree that you are right: what I have in mind here is a meaning unit or a meaning complex, but I happen to use the word object for it. I excuse myself for it, but I hope you will see that, in the end, there is little to do about it because of what I want to say. Anyway, it is this that I have in mind.

([B.G.S.:] "Perhaps one point about your terminology, about syntactic, semantic and objectual categories here, because there is a traditional notion of Bedeutungskategorie, from Husserl and from Leśniewski, which became very important in Polish logic, Ajdukiewicz and so on, but that is the category of expressions, and it is not on the level of—so the semantical notion there is really what you would

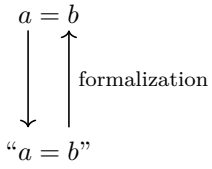
call a syntactic category. It is important to see this: the semantic categories of Ajdukiewicz and so on classify expressions, and also the Bedeutungskategorien of Husserl." But still, the Bedeutungskategorien will correspond to—of course, the English translation is semantical categories, and what I am saying is that semantical categories and objectual categories will collapse into one and the same notion for me, so we shall have syntactic categories and semantic or objectual categories. [B.G.S.:] "Yes, no, but I think that what you call syntactic categories was called semantic categories by them, that is the unfortunate thing, but I think that your terminology is the better one, and so that is the unfortunate terminological choice, for instance used by Tarski in 'Der Wahrheitsbegriff'." So that would have to be studied carefully. "Yes." But the way Husserl speaks about the Bedeutungskategorien—it is certainly this in our terminology.)

I will have to say more about the, so to say, general semantic picture here, and I will do so in due course. I now only said as little as I could in order to clarify the question that I began by posing, namely, Does a nominal definition define primarily an object or the meaning of a linguistic expression? It is clear that, according to what I have said here, it primarily defines an object. However, if the object is nothing but the meaning of the expression of the object, then it indirectly, or so to say in two steps, also defines the meaning of that expression, but primarily it defines an object. This may sound paradoxical, because after all I am now discussing what is called nominal definitions, that is, name definitions, and the outcome of the discussion is that even a nominal definition defines an object. That is, in this sense, even a nominal definition is a real definition, that is a definition of a thing, rather than of the meaning of an expression, primarily. Of course, I do not mean to propose that for this reason we should abandon the term nominal definition. It is a very good term that we should continue to use for definitions of defined notions, that is, for definitions which take the form of a definiendum set equal to a corresponding definiens. Despite the name, however, I think the conceptual situation is as I have described it.

When one says that it is the meaning of an expression that is defined, then one very naturally says that the definition itself states a synonymy, that is, it states that the meaning of the left-hand member is to be the same as the meaning of the right-hand member. Let us now change this picture by looking at two objects, a and b, instead of two expressions. Then we have—without writing out any type information in this informal discussion—two objects a and b, and $a = b$ says that a and b are the same objects of whatever type they are. When they are formalized, we have instead the expressions of the objects, "a" and "b". What happens with the relation of identity between objects under the process of formalization, when you trace it backwards along the formalization arrow? Well, most naturally, it becomes the relation of synonymy between the linguistic expressions, because, after all, what does synonymy mean? Well, two expressions are synonymous if they have the same meaning, that is, if their meanings are the same.

As I have already said, the meaning of the expression "a" is the object a from which it was obtained through the process of formalization, and the meaning of the

expression "b" is the object b from which it was obtained by formalization, hence to say that the meanings of "a" and "b" are the same is precisely to say that a and b are identical.



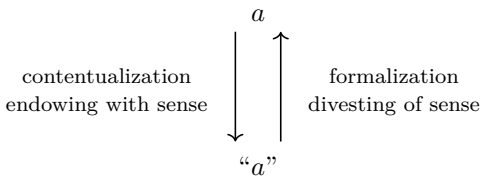
If one looks at a nominal definition, not as one should, as stating that an object a is set equal to an already present object b, but one looks upon it taking the linguistic attitude instead, then what it says is therefore that the meaning of the definiendum is set equal to the meaning of the definiens. That is, it states a synonymy. That, as far as I can see in the literature, is the usual way of accounting for nominal definitions, but, as I have said, a nominal definition states primarily an identity between objects, and not synonymy between linguistic expressions.

I will continue next time with introducing the base categories. We so far only have the function category former, and we need the category of sets and the category of propositions.

Lecture 7, 11.11.93

Last time we got into somewhat deep and important philosophical problems, so I think it would be better that I continued a bit with that before going over to explaining the basic categories.

Remember the picture that we had:



The object a here could be $2 + 2$ for instance, in which case it would be a number. The process of paying attention to the form of the object is the process of formalization, or divesting of sense, and in the opposite direction, you have the process of endowing an expression with sense, or, if you want, contentualization, to have a word dual to formalization. I gave Husserl's terms for the latter, namely Sinnbelebung and Sinnbeseelung, but I forgot the corresponding term on the other side, namely Sinnentleerung.

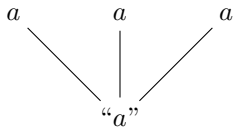
Concerning this picture I want to say two things. Firstly, there is a metaphor that is quite helpful in trying to clarify the difference between the object and its form, or the formalization of it. The problem of distinguishing these two is that they look the same: the expression "a" has precisely the look of the object a, if we just pay attention to what it looks like, to its shape or form. It is precisely because they look the same that you tend to use some device like the quotation marks to indicate when you have in mind the one and when you have in mind the other.

A simile that I find helpful here is the simile of the magnet: a is like the magnet which attracts or repels other pieces of metal, whereas "a" is like the piece of iron demagnetized. The magnet and the demagnetized piece of iron look exactly the same, but they behave quite differently with respect to other things, in particular to other pieces of paramagnetic material.

That is a simile for this situation, but it is clear that you have the corresponding situation, not only in the linguistic field, but also outside of this linguistic context. Think, for instance, of a piece of mineral that someone brings home from a mountaineering expedition. That piece of mineral may be put into the mineralogist's collection, in a little box, say, and in that way it acquires a certain sense within

the context of that collection of minerals. But someone else might have brought the same piece of mineral home simply as a memory, as a visible memory, of this particular expedition of the particular mountain on which he was, and then it would function quite differently. It would be kept at home in some drawer, for instance, or maybe on a table, as that little memory. He might, however, also start to use that piece of mineral, for instance as a paperweight, if it is of a suitable size, and then it will function in a particular way in his familiar working surroundings.

Here we have three different senses that something which looks exactly the same acquires depending on how it is used. It is a prime example of ambiguity, or equivocality, in the non-linguistic field. Maybe the natural picture for that is that we have something which looks one and the same way, but it acquires different senses:

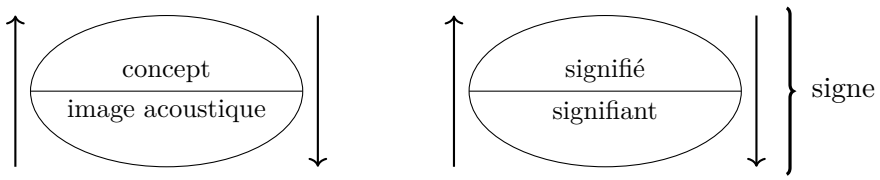


We thus have different objects in these three situations that I referred to: the objects are different, but the expression, or the form, is the same.

For yet another example of the same sort: if you have a piece of plank that you find on the shore, you can take that home as a log to put in your fireplace. But you might also, on your way home, look at it and say, Oh, that is a very nice piece of wood actually, so why not use it as a chopping-board? If it then becomes the chopping board where you cut your bread daily, it of course assumes a quite different sense, so it becomes a different object.

Maybe the most obvious example of this sort is the bread as used in ordinary, daily eating as compared with the bread in its quite different function in the Mass, in the Eucharist, where it acquires a completely different sense. In most cases, one uses a special kind of bread in the latter situation, so it has a different form even and is not an example of this ambiguity situation. If, however, you use the same sort of bread, as you do in the Russian Orthodox Church at least, then you have exactly this situation. The baker may bake a lot of loaves in the morning, and all of those loaves are distributed to various shops to be sold and so on, but maybe there is one loaf that goes to the church for the Mass. That loaf of bread assumes of course a quite different sense, because it is used quite differently in its function as host in the Mass.

I also want to point out that, as I understand it, it is precisely this figure that Saussure is concerned with when—well, he draws it in this way:

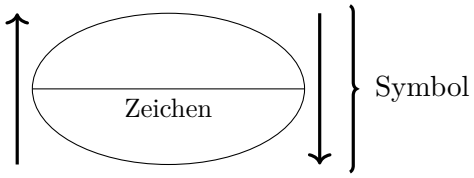


He says that the sign consists of two parts: one is the acoustic image, and the other is the concept expressed by that acoustic image. He even draws both arrows like

that, so this is what the figure looks like. All of this is the sign in his terminology, so he uses the terminology signifiant and signifié for the signifier part and the signified part of the sign.

As far as I can understand, this is exactly the same picture as the one I drew to begin with, where the upper part corresponds to a and the lower part to "a". The only thing that seems unfortunate, or difficult, for me is to use the word sign for this whole thing, because normally we use sign only for what signifies. That is, we would normally use sign for the signifiant rather than for this whole thing.

It is entirely the same situation as with Wittgenstein's use of the terms symbol and sign, Symbol and Zeichen, in the *Tractatus*,



Wittgenstein says (TLP 3.32),

> Das Zeichen ist das sinnlich Wahrnehmbare am Symbol,

so the lower part would be the Zeichen and all of this would be the Symbol. Remember also another passage (TLP 3.326):

> Um das Symbol am Zeichen zu erkennen, muß man auf den sinnvollen Gebrauch achten.

The symbol for Wittgenstein is thus the sign in its use, with its meaning, or in its use, and that is exactly like Saussure's signe. ([B.G.S.:] "May I interject something here, because with Wittgenstein it is difficult to draw this exactly as you have done, because Wittgenstein also says that it includes the projection method, aber nicht das Projizierte." The symbol, you mean? "Yes, so it would include the arrows, so to say, but not the concept." But not the concept. "Yes, and so that is..." So there is some difference here. "There is some difference, yes.")

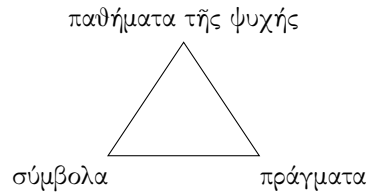
The other thing that I would like to take up at this point, because we needed it last time, was the semiotic triangle. As I said last time, it is a pattern of thought that goes through the whole of our tradition, and the origin of it—the earliest place where you have it very clearly—is right at the beginning of *De Interpretatione*. In a very well-known passage Aristotle says

> Spoken sounds are symbols of affections in the soul...

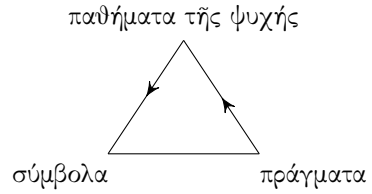
Maybe impressions in the soul would be a better translation, though the Greek word is παθήματα, so it is really passions in the soul, if you are to translate it literally.

> ...and written marks symbols of spoken sounds. And just as written marks are not the same for all men, neither are spoken sounds. But what these are in the first place signs of—affections of the soul [impressions in the soul]—are the same for all; and what these affections [impressions] are likenesses of—actual things—are also the same.

Here we have very clearly the trichotomy which is captured in the semiotic triangle. The words used by Aristotle are as follows: For the things, the Greek word is πράγματα. What is in the soul, the impressions in the soul, or passions of the soul, are the παθήματα τῆς ψυχής, and the expressions that you use to give expression to these impressions in the soul are called σύμβολα.

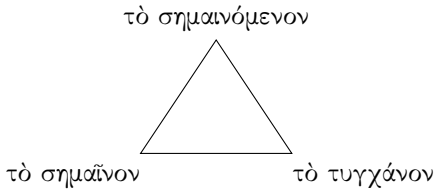


So here you have the semantic triangle, and there is a natural direction to the arrows here. You have the exterior things that give rise to impressions in the soul, and then we give expression to these impressions in the soul, verbal expressions one could say,



You thus naturally think of the arrows going that way, and the relation between the symbol, or the expression, and the thing goes via the impressions in the soul. There is thus this mental link between the expression and the thing.

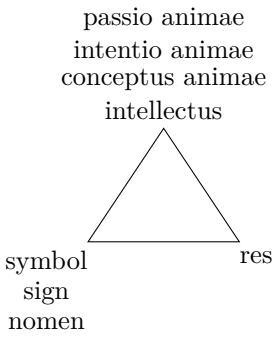
That was Aristotle, and, although there is a great difference between Aristotle's logic and Stoic logic, this thought pattern is so basic that it was exactly the same for the Stoics, although they used different words. It is particularly interesting that the words the Stoics used are the ones that we still use and exactly the ones used by Saussure. In their terminology, what is in the soul, that is, what is signified, or the signifié in Saussure's terminology, is τὸ σημαινόμενον, which in Latin is significatum, whereas the signifying thing is τὸ σημαῖνον, signifiant as Saussure says, or significans in Latin. Finally, there is—and strangely enough, they do not use πρᾶγμα for the thing, but the Stoics used the word τὸ τυγχάνον for the exterior thing, but it is exactly the same trichotomy that is involved here,



The most well-known passage to clarify this is from Sextus Empiricus *Adversus Mathematicos*, VIII, 11-12:

> The champions of the first opinion were the Stoics who said that "Three things are linked together, the thing signified and the thing signifying and the thing existing"; and of these the thing signifying is the sound ("Dion," for instance); and the thing signified is the actual thing indicated thereby, and which we apprehend as existing in dependence on our intellect, whereas the barbarians although hearing the sound do not understand it; and the thing existing is the external real object, such as Dion himself.

This thought pattern was taken over by the Scholastics:

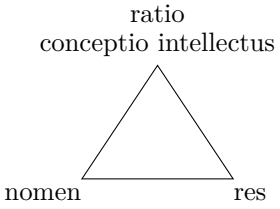


There is no doubt that the external thing is called res in the scholastic terminology. For the expression you had both symbol, as in Aristotle, and sign, as well as the special case of a name, nomen, but there are many possible choices here. The most important thing to become clear about is what was used in scholastic terminology for the thing signified, that is, the meaning, as we would say. There is a very clarifying passage in Ockham's Logic where he says that this is called either passio animae—so that is exactly from the passage that I read from Περὶ ἑρμενείας, the passion, or impression, of the soul—or what this passion became when it was translated via the Arabic: as Aristotle was passed down to us, it became intentio animae.[1] This is the original meaning of intention with "t": it is this impression in the soul. Intellectus is Boethius' terminology, which he used in his commentary on the Περὶ ἑρμενείας. Conceptus, the word that we very generally use—and which you saw in Saussure's first picture, where you have the image acoustique and the concept—corresponds to the Greek νοήμα, the term that Husserl also began to use as a technical term. The Greek νοήμα was translated intellectus by Boethius and conceptus by William of Moerbeke, as far as I know.

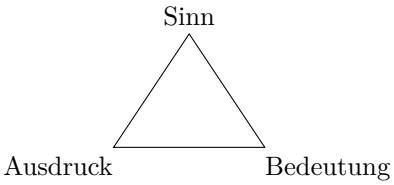
If you look at how Thomas expresses himself about this, he uses also ratio, and he uses conceptio intellectus, and res—always res here—and in the particular case

[1]Ockham, *Summa logicae* I.1.6: "Terminus conceptus est intentio seu passio animae aliquid naturaliter significans vel consignificans, nata esse pars propositionis mentalis, et pro eodem nata supponere."

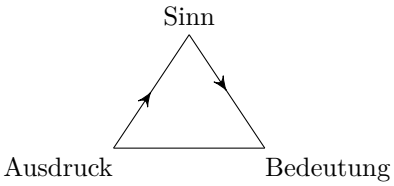
that he is discussing, namely the names of God, the expression is called the nomen,



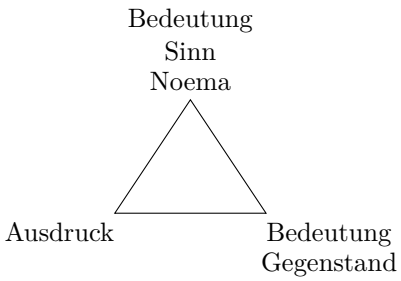
This is the triangle during the scholastic time, but we are, of course, normally familiar with it from Frege in the first place. We all know what words Frege used here: Ausdruck, Sinn, and Bedeutung,



If you want to put directions on the sides of the triangle here, it is interesting to note that we tend to put the arrows in completely the opposite direction to the Aristotelian picture: we say that it is the expression which expresses its sense and refers to its reference via the sense, so we would now put the arrows like this:



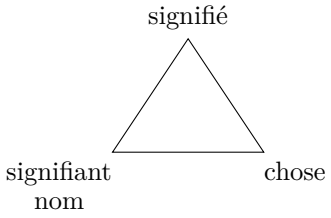
Husserl on this point is very close to Frege:



As you probably know, Husserl thought it was not so good an idea to make a technical distinction between Sinn und Bedeutung, so he tends to use Bedeutung and Sinn interchangeably. The only difference is that he tends to use Bedeutung when he is concerned with linguistic meaning, linguistic in the narrow sense, whereas if the concept of meaning is generalized also to non-linguistic situations, as in the examples that I gave here in the beginning, he uses Sinn in that more general sense. But there is no important difference between Sinn and Bedeutung for Husserl. In

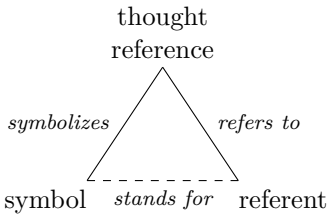
the *Ideen*, he introduced the new word Noema, which is just the Greek νόημα, from νοεῖν, the verb for thinking, so this is simply the thought, what is thought, and then he does not have the option of using Bedeutung for the thing which is referred to, so there he uses the traditional word Gegenstand, and for expression he uses Ausdruck.

And then we have Saussure:



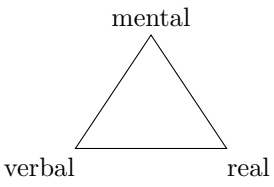
The words I showed you belong to the lower left node and the top node, so Saussure uses signifiant and signifié there, and then chose. If it is a name, of course, nom can be used instead of signifiant.

Finally there is those who had the idea of putting this conceptual scheme in a triangular form, namely Ogden and Richards in their book *The Meaning of Meaning* from 1923. There the picture is this:



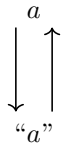
This is the first time in modern philosophy, as far as I know, where the verb refer and the substantive reference are used systematically, albeit somewhat differently from how we use them now. At the top you have the thought, thought or reference—very strange—and then they say that the symbol symbolizes the thought and that the thought, or reference, refers to the referent. They moreover say that the symbol stands for the referent, and they make clear that the symbol stands for the referent only in this indirect way that it symbolizes the thought which refers to the referent, so there is no direct path from the symbol to the referent, but the path passes via the intermediate mental thing, the thought.

So this is the semantic triangle, and, of course, what it reflects is really a maybe even more fundamental thought pattern, namely the trichotomy into the mental, the verbal and the real,



As you have seen from the passages that I have read here, this trichotomy is there from the very beginning. In Aristotle, for instance, and in scholastic times—we know that scholastic thinking was, so to say, wholly framed in this trichotomy. If you think of the debate about the nature of universals, for instance: it was largely seen as a debate between three different positions corresponding to the verbal, the mental and the real, namely, a nominalist position, a conceptualist position and a realist position. [B.G.S.:] "Looking at the pictures that you have drawn, Frege is a bit of an outsider in the sense that his Sinn definitely is not mental." Yes, so now we come to that point.

I drew the semiotic triangle last time in order to clarify a certain point. In this picture that I am using,



I said that we start from an object, and then we look at it purely formally, and in that way we get to the expression. I said that "a" is the expression of the object a, and conversely, a is the meaning of the expression "a". Georgos then quite rightly said that this sounds strange, or unusual. If you compare this picture with the semiotic triangle, "a" corresponds to the lower left node, and a corresponds to the top node. We would therefore normally want to call a a concept or a thought or something like that, whereas I said that we start from an object here, and object is of course the word that we normally associate with the lower right corner of the semiotic triangle.

There is, however, something inevitable in speaking about a as an object, and it has to do with what Göran just said. What would be a particular example of this a? As I said, we might start with some mathematical object, say $2 + 2$, which is a number, so surely the word that we ordinarily use there is precisely an object, a mathematical object. We could, however, just as well speak of it as a mathematical concept. It does not matter, you see, whether we speak of mathematical concepts or mathematical objects, though it is most natural to use the term object. Points, lines, planes, circles, ellipses and so on in geometry: surely, we speak of them as mathematical objects. In modern mathematics, we speak of natural numbers, rational numbers, real numbers, complex numbers, analytic functions and so on as various kinds of mathematical objects. It thus seems inevitable that we start from an object. We then consider it purely formally, and in that way we reach the expression for the object. In the case of $2+2$, for instance, we merely pay attention to the fact that it has plus form and that it has two parts, both of which are "2", that is 2 considered purely formally.

Göran said that, for Frege, the top node would not be mental: Frege's Sinn would not be mental. That is true, of course—and maybe not only beginning with Frege, because before Frege you have Bolzano's conception of Vorstellungen an sich, objective representations, if you want, which are very much like Frege's Sinne. We

have a precursor of that notion in the conceptus obiectivus in late scholastic time: the objective concept, not the concept as something that is in my mind, understood in some psychological sense, but the objective concept. The earliest place where you have something similar to this is in Stoic logic, namely the concept of λεκτόν, which literally means that which is said, that is, that which is expressed by a linguistic expression. It is thus what we call the meaning, or the concept, or the thought, which is expressed by the expression.

As Göran said, Frege certainly did not consider this to be mental in some psychological sense of mental, where psychological then refers to the psychology, largely empirical psychology, of his time. Everything of that sort should be avoided in speaking about meanings. One might say in a pictorial way that Frege, and maybe also people before him, expelled the thoughts from the mind. The question is then, If they are not mental, what kind of entities are they? They are clearly not real, or exterior, or external, Frege thought, since they are obviously different from tables and chairs and so on. On the other hand, since Frege expelled them from the mind, neither are they mental, or internal. Of course, this metaphorical use of inner/outer, or internal/external, or interior/exterior, is exactly the same as the distinction between the mental and the real. [B.G.S.:] "It is important that it is not the same real as there: Frege's real was wirklich, that which was causally active." No, I am not sure that this is not the same, but anyway, he said that the Sinne are neither real nor mental, so he had to invent this third realm, his drittes Reich, as you may remember from the paper called "Der Gedanke". So he invented the third realm. . . ([B.G.S.:] "No, he did not invent it. It existed long before. It was a common term in Neo-Kantianism, and there is a whole lemma in Eisler, in 1907, on drittes Reich, where you find quotes from Rickert and other people." Right, was that by Frege himself? "No, 1907, it is long before "Der Gedanke", which is 1919." Right, but was that also by Frege from 1907? "No, no, that is a lemma in Eisler." Oh, in Eisler. "In an early edition of Eisler, there is a lemma on the drittes Reich with a long list of quotes from Neo-Kantians." Oh, I see, so clearly then the very term here—third realm—for this realm of meanings, or Sinne, was current apparently then at the time in Neo-Kantianism. "In precisely Frege's philosophical surroundings and among his associates." Right.)

Then we are at the stage where the thoughts, or the meanings, have been expelled from the mind and expelled to this third realm. If you have a special third realm here, you still have the trichotomy, but the question is how much difference that makes: whether not the same effect, so to say, is achieved by saying that by mental you mean, not something which is mental in the sense of empirical psychology, but you have another, more objective conception of the mental, and then there might be no harm in saying that they are mental.

The question thus arises, When the concepts, or thoughts—and meanings are concepts, or thoughts—are expelled from the mind, do we really need to invent some new third realm for them? Is it not rather that they come to belong to the

same realm as a lot of things that we are already familiar with, so that in the semiotic triangle, where you have concepts, or thoughts, at the top, and objects, or things, down to the right, these two collapse into one realm, which is to say that there is no categorical difference between concepts and objects? To my mind, this is exactly how it is, or at least, how it comes out in connection with type theory.

To my mind, there is no categorical difference between, say, mathematical concepts, the meanings of mathematical expressions, and other human artifacts. Here I could say human constructions, since the word that is often used, especially in intuitionistic mathematics, is that of mathematical constructions. There is no fundamental difference, that is, no difference in the kind, or realm, that they belong to, as compared with human constructions like the knife and the fork and the spoon and the garlic press and the steam engine and the combustion engine and the aeroplane and so on: all of these are human constructions. Some of our constructions, or some of our tools, are linguistic in nature, but that does not place them in a different realm.

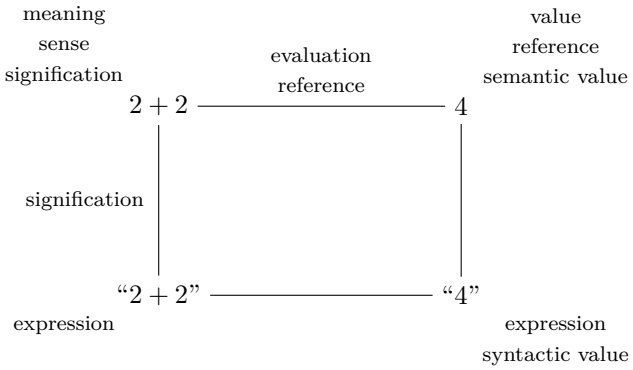
Since I promised in the very beginning that I would discuss in detail how the sense/reference distinction comes out in connection with type theory, I think this would be the natural point to do so. We will then see more clearly what I just stated, namely that there is no categorical difference between what appears at the top in the semiotic triangle and what appears in the lower right-hand corner: they will be of the same category, that is, objects of the same sort. That is why it is not so strange that I tend to use the word object for something where Georgos correctly said that it sounded strange because one would normally call it a concept rather than an object: concepts and objects will be on a par—concepts, of course, in this objective sense.

About the sense/reference distinction, let me first of all explain the terminological situation. As I already said, Frege's terms were Sinn and Bedeutung, and the translation of these into English has posed problems. I know at least of the following proposals. The first translation was by Russell, who used meaning and indication. That is Russell in the appendix to *The Principles of Mathematics* from 1903. Two years later he changed indication to denotation, conscious that he did not use denote then in the way that it was used by Mill, but in a new way. The most common translation, the most successful one perhaps, is the sense/reference translation, which is from Black, 1948. Then you have Feigl's translation from 1949, which follows Carnap's proposal to use sense and nominatum. Church then used sense and denotation in "On the logic of sense and denotation" from 1951. Finally, in the last translation of Frege, the *Posthumous Writings* from 1979 by Long, White and Hargreaves, sense and meaning is used, and that is no doubt the closest to Frege's original terms.

That is about the terminology. You can see that the difficulty is really with the second term here. In ordinary English, we have plenty of words for the meaning, namely signification, meaning and sense, and they are quite interchangeable in ordinary usage, but we need a term for the thing, or the object, referred to. We have seen the distinction in the semantic triangle, and the semantic triangle has

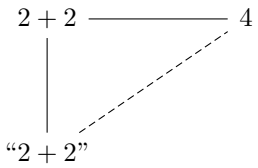
been there all the time, so one may wonder why there is no word to use here. The reason is that in scholastic, or in traditional, usage 'signify', that is, significare, was used both for the signification of the meaning, and for the signification of the thing signified. You saw immediately whether one had in mind the meaning signified or the thing signified if you had the object there, I mean object in the grammatical sense, so there was no trouble: one could always use significare. There is, however, no doubt that we are in need of a good word for the relation to the thing signified, and the most successful proposal, as far as I can see, is to use reference. It comes, as we saw, from Ogden and Richards, who, however, used referent rather than reference, which is very unfortunate, since the referent is the referring thing rather than what is referred to: it should not be referent, but relatum, though if we use the usual act/object ambiguity, we can of course also call it reference as we have done.

Let us now turn to how the sense/reference distinction comes out in connection with type theory.



We start with some object here, like $2 + 2$, and then we have the expression that expresses it, "$2 + 2$", and the relation between the expression and the object, which goes in both directions, that is, we can pass from the expression to the object and from the object to the expression. But we have also another ingredient here, namely the passage from the expression to its value, that is, the evaluation relation. Surely, this number here, $2 + 2$, is not given in primitive form, if we think of the numbers as determined by the first two Peano axioms, namely that 0 is a natural number and that if a is a natural number, then the successor of a is a natural number. It has to be calculated before we get it into computed form. If we use the decimal number system, the value is 4, and if we would use the unary number system, the value would be $\mathbf{s}(\mathbf{s}(\mathbf{s}(\mathbf{s}(0))))$. Let us just say that we use the decimal notation system. Then I can of course perform exactly the same linguistic reflection on the value and look at 4 purely formally, that is, look at the expression for this object, which is "4", where I indicate the change of attitude by quotes. We thus get a square now rather than a triangle.

The triangle is, however, contained in this square, namely, it is the upper left-hand corner of the square,



This is the semiotic triangle, because "2 + 2" is an expression, more precisely a numerical expression, and 2 + 2 is the object, that is to say, the number that it expresses, and 4 is the value of that number.

If you want to direct arrows in the semantic square, we naturally direct the upper side in the direction towards the value,

$$2 + 2 \longrightarrow 4$$

The upper right-hand corner will thus be called the value or the reference. The lower left-hand corner is the expression, and the upper left-hand corner is the signification, the sense, or the meaning, of the expression. The passage from the expression to the meaning is what we call signification. Of course, signification has the usual act/object duality: it stands both for the process, the actual process of signifying, and for what is signified. If we want to avoid that ambiguity, we can use significatum for the latter.

As I said, the vertical side here really goes in both directions: you can think of it both in the direction from the object to the expression for the object and from the expression to the object that it expresses, that is, what we call its meaning. In the horizontal direction we have a completely different relation, which is what we call reference, reference in the act or process sense, or evaluation in the mathematical case. In the lower right-hand corner we also have an expression, namely the expression of the reference or value. There is an absolutely fundamental difference between the upper and lower horizontal levels, which are related by the signification relation, and the left and right vertical levels, which are related by the reference, or evaluation, relation.

As you see now, there is no categorical difference between what you have in the upper two corners here: they are both numbers in this case, and if you were dealing with, say, propositions instead, then these would both be propositions. For the semiotic triangle, this means that there is no categorical difference between what appears at the top and in the right-hand lower corner: they are objects of the same sort, or the same category.

Now I should say something about the evaluation relation. What is evaluation? Well, it is simply this, that every object will be either primitive or defined, and if it is a defined object, then it has the form of a definiendum, and I can pass to the corresponding definiens. If that is already in primitive form, then the computation is over in one step, or it may again be defined, that is, again have the form of a definiendum with a corresponding definiens, and so on. So we will get a chain starting from some object, let us call it a, or a_0. If it is not already primitive, it

will have the form of a definiendum with the corresponding definiens, say a_1. If this is again defined, it will have a definiens, a_2, and so on. Of course, if the definitions have been properly made, then this sequence of definitional reductions must eventually terminate with something which is of primitive form, say a_n. We call such a sequence a definitional chain, and what appears at the end of the definitional chain, which is primitive, can be called either—with a good terminology introduced by Curry—the ultimate definiens, or a usual terminology is of course to call it the value, the result, or value, of the computation,

$$\underbrace{a_0 = a_1 = a_2 = \ldots = a_n}_{\text{definitional chain}} \text{ (ultimate definiens, value)}$$

This is the relation between the object in the upper left-hand corner and in the upper right-hand corner of the semantic square. Reference thus comes out as computation, and that shows what a basic position the idea of computation has: it is something that comes in already at this basic semantic level. Not all computer scientists maybe would agree or at least would say that this is not the only way one can look at computation. I would agree with that, but at least it is one very clear way in which one can consider the phenomenon of computation.

([B.G.S.:] "May I interpose? In the standard translations, the Black one, Frege would say that the expression signifies its Bedeutung and expresses its sense." Yes, expresses its... "It expresses its sense and signifies its reference, so that does not quite work out right here on the..." Well, Frege says that it bedeutet seine Bedeutung..."And bezeichnen also, Bezeichnung he used in that connection too." Drückt seinen Sinn aus und bedeutet seine Bedeutung? "Yes, and bezeichnen gets used in the same context too." Anyway, bedeuten or bezeichnen, but of course that is express, so it would be unfortunate to translate that by signify. It is better to use refers: it refers to its reference, and here of course it is also the signification relation.)

Dummett often uses semantic value as a synonym for reference, and that is a very natural terminology, and then we also have a term, if we want, for the lower right-hand corner, namely the syntactic value.

I have so far only looked at the objects and the expressions that express them. Let us now look at the notion of identity as well. We know that you cannot have an object without its being an object of a certain category, and to the category belongs an identity criterion. What identities are involved here?

In the upper half of the square we have objects, so they belong to some objectual category. That category is equipped with its identity criterion, so we have a notion of identity between these objects, and clearly that identity is such that if an object is defined, then its value is equal, or identical, to the object itself. In the usual way of speaking, it is thus the same object that we have here which is presented, or expressed, in two different ways: it is the same object that here, to the left, is defined and here, to the right, has been reduced to primitive form. On this level, we thus have identity between objects of whatever category they are, in this case identity between numbers.

In the lower half of the figure, by contrast, we have expressions for objects, and those expressions belong to the corresponding syntactical category, in this case the category of numerical expressions. The reason why it is appropriate to use the term syntactical category is that, again, it is not only equipped with a criterion of application, which tells you what an expression of that syntactic category is, but it is also equipped with a criterion of identity. What is that criterion of identity associated with the syntactic category? That is of course syntactic identity, where two expressions are syntactically identical if they have the same form and their parts are again syntactically the same. On this level, we thus have another identity, namely identity between expressions, which is what we call syntactic identity.

As I have said, in the vertical direction here you can pass in both directions, both from the object to the expression that expresses it and from the expression to the object from which it was obtained by formalization. Since we can pass in both directions, the identity that we have between objects can be traced back downwards in the square to get another relation between expressions, and syntactic identity can be traced upwards in the square, yielding a stricter identity between objects. We will thus actually get four equality, or identity, relations. Originally we have two that come naturally, but since each one of them can be traced back along the arrows in question here, we will get four relations: two on the level of objects and two on the level of expressions.

What are these new relations? When we move the identity between expressions to the object level, then we get the relation which holds between two objects if they are identically expressed, which is to say that they are as identical as they could possibly be: they simply look exactly the same. When we move the identity between objects downwards here to the level of expressions, then two expressions are the same with respect to that relation if the objects that they express are identical. The objects that they express is what we call their meanings, so the two expressions are identical with respect to this relation if their meanings are the same. It is therefore clear what relation we get here: it is the relation of sameness of meaning, that is, of synonymy,

"a" is synonymous with "b" if a is identical to b.

This is the way, it seems to me, that one naturally expresses oneself in connection with synonymy. Since a is the meaning of "a", and b is the meaning of "b", this relation here says that the meanings are identical, and identity of meaning is what we ordinarily call synonymy.

On the other hand, it is clear that this does not square with the way in which Frege speaks about synonymy, or sameness of meaning. So let us compare now with Frege. Although the picture that I have just sketched here is, as it seems to me, the simple and natural picture, it is not exactly what Frege's theory looks like. He takes the sense of an expression rather like I have done, so the meaning, or sense, of "a" is a, but his identity between senses is so strict as to coincide with the notion of syntactical identity traced upwards to the level of objects. For Frege, "a" and "b" have the same meaning if their meanings, which is to say a and b, are

the same according to the other identity relation between objects. If I just wrote here, a and b are the same objects, then we would have the synonymy as I defined it previously on the blackboard, but that is not what Frege does. He actually takes them to have the same meaning only if a and b are the same with respect to the stricter identity here, and that was the identity which was obtained by tracing the syntactical identity upwards, so he gets,

"a" and "b" have the same meaning if a and b are identically expressed.

This is only a complicated way of saying something very simple, namely that "a" and "b" have the same meaning if they are syntactically identical. Synonymy in Frege's sense therefore boils down to syntactic identity.

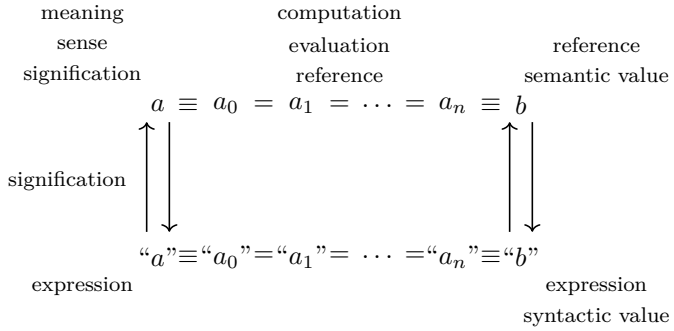
The crucial question here is, whether in a definition,

$$\text{definiendum} = \text{definiens}$$

are the expression for the definiendum and the expression for the definiens synonymous or not? It is clear that, as I have explained the situation, they are synonymous, because the definiendum is identical to the definiens, hence the expression on the left-hand side has the same meaning as the expression on the right-hand side. On the other hand, Frege thought of meaning as the way in which an object is presented, namely the way in which the reference, or value, is presented—Art des Gegebenseins—and, of course, the reference, or value, is differently presented, or differently expressed, by the definiens and the definiendum: they look different, and there is one computation step from the definiendum to the definiens. With Frege's notion of synonymy, the definiendum is therefore not synonymous with the definiens. This does not square with everything that Frege says about this, but at least I hope to have captured the essentials of the situation, this quite complicated situation.

Lecture 8, 18.11.93

To remind you what we reached last time: for closed expressions of ground type we have the semantic square,



We have an expression "a" and its meaning a, and they are in this mutual relation vertically here. Horizontally, we have what I call a definitional chain, unless a is already primitive. If a is not primitive, then it is defined, and let us call it a_0—here you see I am spontaneously using three bars to indicate that a_0 is expressed in the same way as a. We have a definitional chain, which is to say that a is a definiendum with a definiens a_1, which in turn is a definiendum with a definiens a_2, and so on. If the definitions have been correctly made, that is, for instance, are not circular, then this definitional chain must eventually end with something primitive, something no longer defined, call it b. That is what we call—in mathematics, the only word really is the value, or in Fregean semantics, reference. Of course, this mutual relationship between an object and its expression is the same for all objects in this definitional chain, so we have the same mutual relation between a_0 and its expression, between a_1 and its expression, and so on up to b and its expression.

We have no terminological problems with "a", because expression is the universally used word for this, whereas for its meaning, we have these several possibilities: either its meaning or its sense or its signification. As I said last time, if we want to have a word for the expression of the value, we might call it the syntactic value, to stress the fact that it is considered linguistically, and then for the opposite, the semantic value.

This square, which we might call the semantic square, contains the semantic, or semiotic, triangle in its upper left-hand corner. And I want to stress, first of all, that the reference, or the evaluation, or the computation, relation here holds

primarily between a and b, and only indirectly do we talk about b as the value, or reference, of the expression "a". This is therefore not a theory of direct reference, but a theory of indirect reference: there is no direct path from the expression "a" to the reference b, but it goes via its meaning, or sense. We do, however, allow ourselves to say that b is the reference of the expression "a": this is simply the convention that if a has value b, or if a refers to b, then we also allow ourselves to say that the expression does so, namely, indirectly.

I stressed last time the fundamental difference between the horizontal and the vertical relations here, that is, the signification relation, which is the vertical one, and the reference, or evaluation, relation, which is the horizontal one. With Frege, we can say that the expression "a" means its meaning—means, or expresses, or signifies, of course, if you use signification—whereas a refers to, has as value, evaluates to, b. Frege says "drückt seinen Sinn aus" for the first one and "bedeutet oder bezeichnet"—both terms—its Bedeutung, for the second,

$$\text{“}a\text{”} \left\{ \begin{array}{c} \text{signifies} \\ \text{means} \\ \text{expresses} \\ \hline \text{ausdrücken} \end{array} \right\} \; a \; \left\{ \begin{array}{c} \text{has value} \\ \text{refers to} \\ \text{evaluates to} \\ \hline \text{bedeuten} \\ \text{bezeichnen} \end{array} \right\} \; b$$

Next I want to say a few more words about the evaluation relation. How are objects, respectively expressions, evaluated? The evaluation relation, a has value b, or as I have been saying, a evaluates to b, or a refers to b, is defined somewhat more explicitly by saying

(1) If a is primitive, then a has value a.

(2) If a is a definiendum, with definiens b, and b has value c, then a has value c.

Whether something is primitive or not is very easily ascertained in type theory, namely simply by looking at its outermost form and seeing whether that form is a primitive or a defined form. For instance, if it is 0, it is primitive. If it is the successor of something, it is primitive, because successor is primitive. If it is the pair of two things, then since the pair form is primitive, it is primitive. We do not care at all about what the parts look like, whether they are primitive or defined: we just look at the outermost form.

This defines inductively the computation relation. Using the equality arrow, $\Rightarrow$, for the evaluation relation, we can give the second rule in a kind of schematic form,

$$\frac{a =_{\text{def}} b \qquad b \Rightarrow c}{a \Rightarrow c}$$

Together with the first clause, this defines the reference relation.

Now observe, what I did not reach last time, that since we have this definitional chain between a and b,

$$a \equiv a_0 = a_1 = \ldots = a_n \equiv b,$$

where b is the reference, and a is the sense, or meaning, of the expression "a", the sense is equal to the reference considered as objects. The sense differs from the reference only in the way they are expressed. All of $a_0, \ldots, a_n$ are of a certain

category, α say, and with that category is associated a criterion of identity, and according to that criterion of identity, we certainly have that sense is equal, or identical, to the reference,

$$\text{sense} = \text{reference} \quad \text{(as objects)}$$

Sense and reference will even be syntactically the same in one case, namely when a is already primitive, because then the length of this computational chain is zero, and the beginning of the chain coincides with its end. In that case, then, the sense is even syntactically the same as the reference. If a is defined, on the other hand, then of course this computational chain has length at least one, and in that case, a and b are no longer syntactically identical, because b is primitive, hence its outermost form is primitive, whereas a is defined, which is to say that its outermost form is defined, and no form can be both primitive and defined. In that case, therefore, the sense is syntactically different from the reference, although they are equal as objects. Using the three bars for syntactical identity, we have

$$\text{sense} \not\equiv \text{reference, unless } a \text{ is already primitive}$$

What I have said here is quite familiar to computer scientists, and I would like to put it in those terms. I think I said briefly last time that this is really the, so to say, deep reason why computer scientists have been interested in type theory, and through it also in these philosophical questions: in type theory, the very notion of computation enters at this basic level, whereas in, say, classical set theory, or a language like that, there just is no talk about computation at any place in the system in question. Computation really has a semantic significance in type theory.

The situation here can be expressed, and is usually expressed in computer science, by saying that during a computation, there is something that changes all the time—namely the expression changes all the time,

$$\text{“}a\text{”} \not\equiv \text{“}a_0\text{”} \not\equiv \ldots \not\equiv \text{“}a_n\text{”} = \text{“}b\text{”}$$

where b is the result of the computation—but there is another thing which is invariant, namely the meaning, that is, the object. The expressions change during the computation, but the object is invariant, and the object is the meaning of the expressions.

This picture that I have given holds for closed expressions of ground type. Of course, I have not explained the ground types yet, but I have at least said what they are: the type of sets, and for an arbitrary set A, the type of elements of A, and we also have the ground type of propositions, and for each proposition A, the type of proofs, in the sense of the intuitionistic proof objects, of A. Eventually I will identify the type of propositions with the type of sets, but let us suppose for now that we have not made that identification yet. The picture that I have shown is, in any event, the same for objects of all four kinds, or two kinds after the identification.

I should also say that I am in agreement with Dummett in his talk at the meeting here in Leiden in September last year, as far as individual terms are concerned,

which is to say elements of sets, for instance numbers or numerical terms. There is no difference in substance in what I have been saying here as compared with Dummett on that point.

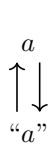
On the other hand, Dummett says that we do not have, constructively, a distinction between sense and reference for propositions, and on that point I do not agree: for all closed expressions of ground type, in particular for sentences, that is, for propositional expressions, we have this picture as well. Why? Well, simply because we have defined propositions. Of course, we usually do not have any very elaborate schemes for defining propositions—in type theory we have, but, say, in predicate logic, we do not—but at least we have abbreviative definitions, like when we define $\neg A$ to be $A \supset \bot$, or we define the logical equivalence of A and B, $A \supset\subset B$ by saying that it is $(A \supset B) \wedge (B \supset A)$. As soon as we make such definitions, then also propositions begin to be defined, like these. The two definitions just given do, of course, not give rise to any long computational chains, because in one step we reach something primitive, provided implication and conjunction are primitive, but in type theory, as I said, we have much more complicated schemes of definition. Hence we will have exactly the same picture for propositions as we have for, say, numbers, or elements of sets in general. So we certainly have a sense/reference distinction for propositions.

What has radically changed, of course, as we shall see later on, in comparison with classical semantics, or Fregean semantics, is that the references of propositions cannot possibly be taken to be truth values. We cannot compute the proposition so far as just to collapse into a truth value, simply because the passage from a proposition to its truth value is not an effective one. I will elaborate on that later on. Although we have a sense/reference distinction for propositions, the references are therefore not truth values, as they are in classical semantics. Rather, they are primitive propositions, for example such as we have in $A \supset \bot$ and $(A \supset B) \wedge (B \supset A)$. When in the computation of a proposition we reach a proposition with a primitive logical operator as outermost sign, no further computation is possible.

[B.G.S.:] "You call that logical equivalence, but is not material equivalence the usual?" Material equivalence? "Yes, because logical equivalence would mean that the material equivalence was a logical truth." I see, so then material equivalence. "I had one more question on the diagram you began with, namely your first square, that in the bottom row you used ordinary equality signs between the expressions too, but you have not commented on that relation yet." Yes, so maybe I will insert that discussion here then.

At the very end of my last lecture, I talked about the four identities we have, but there was not much time left, so I could not elaborate very much on it. Let me

draw a figure to clarify the situation.



	$\equiv$	$=$
objects	identity of expression = syntactic identity	identity
expressions	identity	identity of meaning = semantical identity = synonymy

As usual, we have these two levels: the level of expressions and the level of objects. What identities do we have? Between objects we have the identity that belongs to the category that the objects are objects of and which I have denoted by two bars. Here we therefore simply have identity between objects. Similarly, for expressions we have the relation of identity that belongs to the syntactical category to which they belong, namely the relation of syntactical identity. In this lecture I started to use three bars for that, a natural notation—when you put more bars, they get more equal: that is the logic of the mathematician's notation.

These are—in the first place, at least—the only two clear identities that we have here, which is to say, we have no other clear identity criteria than these two, that is, either the objectual identity criterion or the syntactical identity criterion. However, because of this mutual relation between the expressions and their meanings, we can carry each of these identities over to the other side. Because of the arrow that points from an expression to its meaning, we can carry the identity between objects backwards to an identity relation between expressions. This yields the relation that holds between two expressions if their meanings are identical. We thus get the relation of identity of meaning here, and of course that is what has been called synonymy since the beginning of time, essentially, I mean, since Aristotle at least. Synonymy—the term synonymy—has been used between expressions if their meanings are the same, and on that point there are, so to say, no terminological difficulties. At least, that is always said to be what synonymy is, so I will use synonymy. It is common also to say that they are—rather than identical in meaning—equivalent in meaning, but there is no difference in substance when you say that.

On the other hand, the arrow that goes in the other direction, that is, the arrow that allows us, whenever we have an object, to look at its linguistic form allows us to carry identity between expressions backwards along that arrow and get a stricter identity between objects. Two objects are identical in that sense if their expressions are identical. This is identity of expression, which it is natural to call syntactic identity. Then of course, identity of meaning could also be called semantical identity, so the expressions are semantically the same.

Out of our original two identity criteria we get in this way four by transferring them to the other side, giving us, on the one side what we have identified as the notion of synonymy and on the other side the very strict identity that holds between two objects if they are identically expressed.

In this way I am, so to say, meeting the challenge of Quine. Quine has, so to say, discredited everything that has to do with meaning, synonymy, analyticity and so forth. How could one counter that challenge except by very carefully explaining what the meanings of our expressions in this particular case in type theory are and what it means for them to be synonymous? The first thing is then to explain what is the meaning of an expression. You have seen what my explanation is, namely that the meaning of an expression is the object which it expresses, that is, from which it has been obtained by changing from the objectual to the linguistic attitude. That explains what the meaning is, and then we have to give the criterion of identity for meanings, and, how do we get that? Well, the meaning is the object, and an object is always an object of a certain category, and with the category there always goes an identity criterion, and because of that identity criterion it makes perfectly good sense to speak about the identity of the meanings of two expressions, which is to say, of the synonymy of two expressions. As for the notion of analyticity, I think several of you have already seen my Kant article, where I deal with the notion of analyticity.

([B.G.S.:] "Would you have another term in the case of syntactic identity for the case that the objects are not presented via linguistic expressions, so to say, when there would be an object of perception or something like that?" Yes, then syntactic would not fit very well, but then it would be identity which holds if they look the same, if they have the same appearance, or the same look. [Maria van der Schaar:] "Identity of form?" Yes, identity of form.)

I was at the point of saying that the original picture that I showed is correct for closed expressions of ground type. There remain of course the higher types, which is to say, there remain functional expressions: open expressions, which are expressions of functions in the old-fashioned sense, and higher type expressions, which are expressions of functions in the modern sense. Both open and higher type expressions are functional expressions.

Maybe this would be a natural point to say that the distinction between functional and non-functional expressions, or better, incomplete and complete expressions, is a distinction that traditionally was made in another terminology, namely as the distinction between categorematic and syncategorematic expressions. This was the scholastic terminology. Syncategorematic expressions are like "not" and "and", which need to be completed to "not A" and "A and B", respectively, in order to become a—well, a meaningful whole, one would then have said, but as I am using the terms, syncategorematic expressions are meaningful even in isolation, though in order to refer they need to be completed. The old terminology here was thus categorematic/syncategorematic, universally used in scholastic time, for instance, and it goes back, apparently, to Priscian.

Frege did not adhere to this old terminology, but introduced his own, relying on a chemical analogy: he used gesättigt/ungesättigt, or instead of ungesättigt, ergänzungsbedürftig. Husserl continued to use, on the one hand, the old terminology, kategorematische/synkategorematische Ausdrücke, but he also used—and that is the first place that I know of—vollständige/unvollständige Ausdrücke, that

is, complete and incomplete expressions. This is for expressions, and as we know, the expressions are precisely as complete or incomplete as their meanings—a functional expression is incomplete because a function is incomplete and vice versa—but for the meanings, Husserl used rather selbständige/unselbständige Bedeutungen, and he also used Frege's term ergänzungsbedürftig. So this is Husserl in *Logische Untersuchungen* IV §§ 4-5. Marty, in his grammatical work from 1908, invented yet another terminology, for reasons unknown to me: autosemantische/synsemantische Zeichen. These are then the terminologies that we have to choose between for complete and incomplete expressions.

I have begun to talk about incomplete expressions in order to say that, for them, the picture is different, because a functional expression does not have a value. Let me take two simple examples to make the point clear. If you take, to begin with, a functional expression in the old-fashioned sense, say $x^2 + 3y$, and ask, What is its value?, then I think your immediate reaction is, Oh, for what values of x and y do you want it? You cannot ask for the value of $x^2 + 3y$ without giving the arguments of the function. It is similar if we have a closed functional expression, that is, a closed expression of higher type. For instance, suppose that we define the factorial function in the usual way,

$$\begin{cases} \mathrm{fac}(0) & = 1 \\ \mathrm{fac}(n+1) & = \mathrm{fac}(n) \cdot (n+1) \end{cases}$$

Then $\mathrm{fac}(n)$ is of course an expression of ground type, but if we remove the argument and just have the notation for the factorial function in the modern sense, that is, fac, you have a closed expression of higher type. If I now ask you, What is the value of fac?, then of course the reaction is the same: which number do you want the factorial of? If you give me an argument, 5 say, then of course I can tell you what the value of fac(5) is, but it does not make sense to ask for the value of fac in isolation.

It thus seems quite clear that on the conception of reference, or value, that I have explained for closed expressions of ground type, functional expressions, that is, expressions for either old-fashioned functions or functions in the modern sense, just do not have a value until they have been supplied with their arguments. This is entirely similar with the situation in natural language, which I guess is universally admitted. Suppose you have some linguistic expression that contains indexical elements, like "He gave the book to her" or something like that. If you look at that in isolation, it certainly makes good sense, we understand what it means, just as of course $x^2 + 3y$ and fac in isolation make good sense. They do not refer, however. That linguistic expression has to be put in a context where the pronouns he and her refer before it makes sense to begin asking if the proposition that we then obtain is true or false, for instance, and pronouns in natural language correspond to variables in the mathematical language.

The question that I am in the process of discussing is, Do functional expressions have reference? As you have seen, my answer is, No, they do not have value, or reference, until you have supplied them with an argument. Now, this is a notoriously difficult point in the interpretation of Frege. Dummett, for instance, devotes a whole long chapter—chapter 7—in his Frege book to this question, the title of which is "The reference of incomplete expressions". The reason I think why Frege was anxious to have a reference associated also with functional expressions is that although functional expressions do not have a value, we certainly need to account for the fact that one and the same function may be expressed in many different ways. Frege accounted for that by saying that functional expressions may have different senses although they have the same reference, that is, in his terminology, they refer to the same function.

It seems that there is no difference in substance here, it is just that what Frege wanted to account for by means of the sense/reference distinction, we have to account for in a terminologically different way, although it is in substance the same. The solution has more or less been given already in distinguishing the various identity relations for objects and expressions,

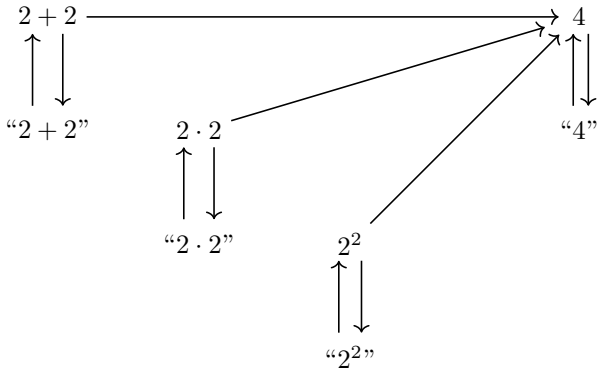
	$\equiv$	$=$
objects	identity of expression = syntactic identity	identity
expressions	identity = Sinnesgleichheit	identity of meaning = synonymy = Bedeutungsgleichheit

For expressions we have two equalities: one is the syntactical identity, and the other is the relation of synonymy, or identity of meaning. The best way to compare this with Frege is, I think, to say that Frege's notion of sense corresponds completely to the notion of meaning, or sense, that I am using, but that he identifies senses by means of the stricter identity, namely the relation of identity of expression, or syntactic identity. For two expressions "a" and "b" to have the same sense in Frege's sense means that their senses, which is to say a and b, are identical with respect to the stricter identity relation, which simply means that their expressions are the same. Frege's notion of Sinnesgleichheit therefore boils down to syntactic identity, whereas his Bedeutungsgleichheit is the relation of identity of meaning, or synonymy.

You see then that we do not need the notion of reference of functional expressions to make the distinction between Sinnesgleichheit and Bedeutungsgleichheit, because it fits into the picture we already have without talking at all about Bedeutung. The simple way of accounting for the phenomenon that Frege accounted for by means of his distinction between sense and reference for functional expressions is simply to say that one and the same function may be expressed in different ways. You may have f and g that are equal as functions, but differently expressed, which is to say that "f" and "g" may not be syntactically the same. In that way, we account for the phenomena, but in a terminologically different way. I will come

back to this once more in a moment, in the course of a discussion of three points in the *Grundgesetze*.

The first one of these is actually the point that I have just been discussing for functional expressions, but now for the simple case of complete expressions. In *Grundgesetze*, § 2 already, I think, Frege says that expressions such as "2 + 2", "2·2" and "2^2" are examples of expressions which have different sense but the same reference. That is how Frege expresses this situation. The sense of the expression "2 + 2", according to the account that I am giving, is simply 2 + 2, so that is the passage from the expression to its sense, and similarly for the other expressions. All these of course also have the same reference, or value, and the value may also be considered to be 4 here, in which case we have the expression "4",



Frege accounted for this situation by saying that these three expressions have different senses, but the same reference. It is exactly the same data here—or, the same situation can be accounted for by saying that we have here one and the same number expressed in three different ways: 2+2, 2·2 and 2^2 are identical considered as numbers. It is one and the same number, but it is expressed in different ways.

It seems to me that this is a much simpler, and more natural, way of expressing this situation. Whereas I start with the objects in accounting for this situation, Frege starts with the expressions. Then he associates with the expressions, to begin with, their senses, which are the numbers that they express, and then he says that these numbers, that is, the senses, are different with respect to his relation of Sinnesgleichheit, which is nothing but saying that the expressions are different. So that is concerning the senses. On the other hand, he says that they all have the same reference. Remember, however, that the reference is not reached directly from the expression, but the reference is only reached indirectly via the sense, so the reference of these expressions is the value of these numbers.

Frege thus always starts at the expressions, but by moving the point of gravity from the expressions to the objects, we obtain a much more natural way of expressing this situation: we just say that the objects are the same considered as objects, but they are differently expressed. That solution works uniformly for complete expressions and incomplete expressions, hence there is no need of a notion of Bedeutung, or value, for functional expressions in order to be able to make this distinction for functional expressions.

That was the first point in connection with Frege. The second point is from § 27, the paragraph where he writes about abbreviative definitions. Frege says,

> Wir führen durch eine Definition einen neuen Namen ein, indem wir bestimmen, dass er denselben Sinn und dieselbe Bedeutung haben solle wie ein aus bekannten Zeichen zusammengesetzter.

He thus says, "denselben Sinn und dieselbe Bedeutung". In the next sentence, he says, as he does in so many other places,

> Dadurch wird nun das neue Zeichen gleichbedeutend mit dem erklärenden,

so he uses the term gleichbedeutend, and here it seems that gleichbedeutend is used as a rather non-technical term. It is the way we ordinarily express ourselves when we do not have to contort our minds into some difficult technical terminology. Gleichbedeutend thus just means that they have the same meaning, which is to say synonymous,

$$\text{gleichbedeutend} = \text{synonymous}$$

You see that the inclination in connection with a nominal definition, which has the form that a definiendum is set equal to the definiens,

$$\text{definiendum} = \text{definiens}$$

is to say, as I think we universally do, that it has as effect that the meaning of what stands to the left is equated with the meaning of what stands to the right, which is to say that the two members in a nominal definition are synonymous. I have not been able to find any place where one does not say that, and the inclination to say that is so strong that even Frege says that they should have the same sense and the same reference, whereas according to his own conception of the distinction between sense and reference, he should certainly say that they have the same reference—and that fits also, since Bedeutung goes together with gleichbedeutend—but, according to his own conception, he should not say that they have the same sense, because that is precisely what an abbreviative definition gives you: it gives you a new way of referring to an object. You have some complicated expression, and it is so complicated that it is tiresome to repeat it all the time: we need a more convenient way of referring to that object, and we get that by introducing an abbreviative symbol. This new symbol is certainly a new way of referring to the object, of presenting the object, hence according to Frege's own conception, he ought to say that definiendum and definiens have different senses, but the same reference.

([Martien Wijers:] "May I ask something?" Yes. "Maybe I understand you incorrectly, but is this passage not a proof that your interpretation of Frege is too strong, that it is not a syntactical identity that he is looking for when he is talking about identity of sense? I think that he has another identity criterion." I agree with that. I agree with what you say in the sense that there is a slight mismatch between the pattern that I am describing here and Frege's pattern. The point that I want to make here, however, and that I am putting in Frege's terms, is that, if we take seriously this idea that the sense is the mode of presentation

of the object, then surely we get a new presentation of the object when we make a definition. [B.G.S.:] "If I may support Martien also, we could also think of the definitional chain as being the mode of presentation, and so everything which is in that definitional chain would share the same mode of presentation." Well, but then, if the definitional chain ends in the reference, according to that criterion, it boils down to sameness of reference. [B.G.S.:] "Yes, if one takes this stance, we need to introduce the third identity relation, which has not yet come into play. Certain of Frege's uses of identity will definitely be that of the third notion of identity, which you have not yet brought into the discussion."

Yes, so that should not be brought in at this stage, but, I mean, I am directing a criticism really wholly in terms of Frege's own picture here. If we accept Frege's statement here, we are in the difficult situation that they should have the same sense and the same reference, because then we have to have, besides syntactical identity and his Bedeutungsgleichheit... [M.W.:] "I understand your point, but I am questioning whether you are taking Frege in his own words, because I do not think that the relation between what it looks like syntactically and what is the mode of presentation, that you can identify them in Frege." No, so Frege has the problem then, if the syntactical expression is not the mode of presentation... "Yes, of course, he has then the problem to make clear what is the identity criterion for sense." Yes, exactly. "But then that would be another identity criterion"—another identity criterion lying in between these two, and that is something which we just do not have at the present time. In that sense, Quine is right in his critique that we do not have a clear identity criterion, because Frege does not have it, and no one has been able to come up with it since then.

I agree with you, however, that I am presenting here a kind of simplified picture to account for essentially the same phenomena, and I know that it does not match completely with Frege. On the other hand, what the Frege industry, I mean, the industry in Frege interpretation, indicates is of course precisely that it does not fit quite well together: there are changes that have to be made, and above all, there is new constructive work that is needed. It is not just a matter of adapting a little here and there, because at the most basic points, namely in explaining what a proposition is, for instance, and what it means for a proposition to be true, we have to make something radically new in order to be able to fit it all together. I take the Frege industry as an indication of the fact that it does not work completely, and of course we know that it did not work since it was inconsistent. We have to make changes, and I am suggesting a simplified picture, but at the same time, I venture today to try to relate it to Frege, since that is what everybody knows in the group.)

That was in § 27, and in §§ 28–29 we come into the question of—and that has to do with functional expressions. First of all, in § 28 he says that "Rechtmässig gebildete Namen müssen immer etwas bedeuten." He thus says "etwas bedeuten" there, whereas the way we naturally express exactly the same substance nowadays is no doubt by saying that well-formed expressions must be meaningful. So this etwas bedeuten here is what we naturally express by saying *to be meaningful*, and note the shift here from Bedeutung to sense, since to be meaningful is to have sense.

So that is the first point, and then comes the point in § 29 which has to do with functional expressions. It is when he explains when an expression for a function of the first level, that is, a function from objects to objects, has Bedeutung. He says,

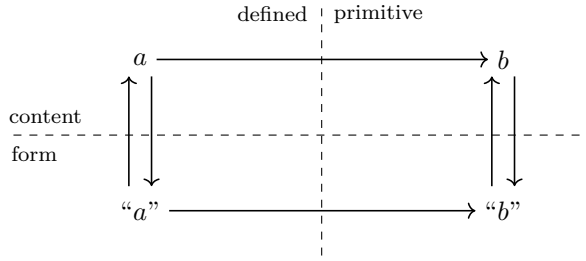
> Ein Name einer Function erster Stufe mit einem Argumente hat dann eine B e d e u t u n g (b e d e u t e t etwas, ist b e d e u t u n g s v o l l), wenn der Eigenname, der aus diesem Functionsnamen dadurch entsteht, dass die Argumentstellen mit einem Eigennamen ausgefüllt werden, immer dann eine Bedeutung hat, wenn dieser eingesetzte Name etwas bedeutet.

Here comes again, just as in the previous sentence I quoted, this expression, to have a reference, or bedeutet etwas, to refer to something, or bedeutungsvoll would be. . . [B.G.S.:] "Significant." A terrible translation. "Yeah, but it is what they use." I think I will put German here instead: eine Bedeutung haben, etwas bedeutet, or, bedeutungsvoll sein—these are the three synonymous expressions he uses in this point, and you see it is exactly the same phrasing as in this sentence, "müssen immer etwas bedeuten". As I said in connection with that sentence, this is what we naturally express as saying *is meaningful*, and hence we have again this shift from something which is expressed in terms of Bedeutung to meaning.

Now compare this explanation of Frege's with my explanation of what a function in the old-fashioned sense is. Frege says that, if you have a functional expression "b", then this is meaningful in case what results from "b" by inserting an expression which has reference—that is, with this change in terminology, when inserting a meaningful object-expression for its variable into its argument places, "$b(a/x)$"—if this becomes a meaningful object-expression. That is what he says in that passage. Now compare that with my explanation of what a judgement of this form means. Let A be a fixed set to replace Frege's universe of objects. What does it mean to say that b is an element of A provided that x ranges over A, $b : A \;\; (x : A)$? Well, according to the meaning explanation, this means that $b(a/x)$ is an element of A, provided a is an element of A, and then there is a corresponding identity condition which we can forget about at this moment, since Frege does not have it. If the set A here corresponds to Frege's universe of objects, then this is essentially the same explanation, but again you see the shift that Frege expresses things in terms of expressions. He is explaining when a functional expression is meaningful, whereas I am explaining what a function is.

From this it is clear that Frege's notion of Bedeutung for functional expressions really corresponds to the notion of sense, meaning or sense, in my terminology. This is in complete agreement with the shift here from Bedeutung to meaning, and also in agreement with what I have said before, namely that Frege's distinction between sameness of sense and sameness of reference is taken over by the distinction between syntactical identity and identity between objects. We can therefore account for the same phenomena here without having a notion of reference for functional expressions.

That finishes the comments I wanted to make on Frege, and then there is only one more figure I wanted to show you in connection with this, namely this one:



The semantic square can be split in two ways, either horizontally or vertically. If we split it horizontally, then it is the distinction between expression and meaning, or between form and content. On the other hand, we cannot always split it vertically. First of all, for functional expressions we do not have the right half of this picture: for functional expressions, the only picture we have is this one:

Secondly, also when the computational chain here has length zero, of course the right half and the left half of the figure collapse, and so we are again reduced to this situation, and that is precisely when a here is primitive already. But if that is not the case, that is, if a is defined, then we indeed have this picture, and the splitting is precisely then into the left half, where the expression and the object that it expresses are defined, and the right half, where they are primitive.

This dichotomy—primitive and defined—assumes a very important role in everything that I have to say, hence I just wanted to comment here on the terminological possibilities. I think primitive/defined is maybe the most natural terminology, but there are three other pairs of terms that are used. One is, instead of primitive, to say direct, or directly presented, and instead of defined, to say indirectly presented or indirectly given. Another terminology was introduced by Dummett in connection with proofs in the sense of intuitionistic proof objects, namely between canonical and non-canonical. Canonical used in this way is essentially synonymous with normal, so normal and non-normal is the third option, a terminology used both in combinatory logic and in proof theory. Both of these two last are, moreover, used in mathematics, when you speak about the Jordan normal form of the matrix, for instance, canonical form or normal form. ([B.G.S.:] “I wonder, did Brouwer not use canonical in connection with the proof of the Bar Theorem, 1927?” It is possible.)

The final thing I wanted to say is in response to Göran’s asking me, if possible, to say something about whether these ideas are generalizable from type theory, or from a strictly linguistic framework, also to cover non-mathematical situations, situations which are not entirely linguistic. In type theory, the reference of the

expressions are also linguistically constituted, because they are numbers and sets and so on, and they are certainly linguistically constituted, and this is a natural point where one can see that there is such a generalization. I have talked about abbreviative definitions in the context of type theory and noted that an abbreviative definition there is nothing but what we normally call naming. Surely we have naming also outside of mathematics. I mean, we have it in the first place outside of mathematics, namely in the naming, on the one hand, of persons and dogs and cats and places and so on, and, on the other hand—and there the analogy is really very close—the naming of species, biological species, like you have the genus swan, say, and you have the species mute swan, whooper swan and Bewick's swan. There you have naming of species, not of individuals but of species, and of course that corresponds much more closely to the mathematical situation, because in mathematics, we are never talking about individual things, but we always are talking about species.

In all of these cases, the situation seems very similar to when we make an abbreviative definition. In the case of a simple naming situation, you have the person there already, in flesh and blood, and you introduce a name for him. Suppose this is my son Andrew, so we have the name "Andrew" and we give meaning to that name by associating it with him in flesh and blood,

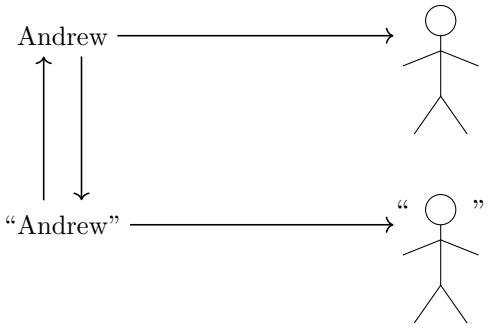
As usual, the association from the name, or the expression, to, in this case, the bearer of the name passes over via the sense. So the meaning of the name, or expression, "Andrew" is Andrew simply, and we have the usual mutual relation between the name and its meaning. On the other hand, we give meaning to it precisely by saying that this is to serve, if not as an abbreviation of him, at least it is to serve as a way of referring to him. So it seems that naming bears a great resemblance to abbreviative definition.

([B.G.S.:] "There is an example which is given in reference discussions. One says that in the case of the name of a person, it is very unnatural to ask what the meaning of the name is, but one asks, Who is Andrew? Not, What is the meaning of Andrew? but, Who is Andrew? That is the question to ask, so that brings this out beautifully" Yes, so, Who is Andrew?, I would say that asks for... "When we are in the situation that we have got the name, but we cannot evaluate it to the flesh and blood person." Yes.)

Now you see that language has become linked to what we ordinarily call non-linguistic things. Language is here grafted onto non-linguistic reality, hence it is clear that if we are to make a semantic account of such parts of language that are

grafted onto non-linguistic reality, then we cannot keep the language apart from that reality. We certainly have to include the reference relation in our picture and also the references, that is, what we refer to. It is only, so to say, the combined system of the language and the non-linguistic things—that whole system—of which we can give a semantic account. It is impossible to draw this sharp distinction between linguistic and non-linguistic things, as has been pointed out forcefully by Wittgenstein in the beginning of the *Philosophical Investigations*.

Also another thing is visible here, namely that if the picture that I have been drawing in the mathematical case is to generalize to this non-mathematical situation, then we have to take seriously the phenomenological idea, and I think it is only in phenomenology that it began to appear, that there is an analogue of linguistic reflection, that is, paying attention to the linguistic form of something, there is an analogue of that also in the non-linguistic realm, namely of paying attention, in this case, to the look of the person, or the appearance of the person, or the form, as Marietje said, of the person. Even in this case, therefore, we have, not only a semantic triangle, but a semantic square:



If we call the upper right-hand corner here the reference, the object referred to, then the lower right-hand corner would be the object phenomenon, or the appearance of the object. You see, moreover, that the passage from the meaning of the name to the bearer is one computation step, or corresponds to one computation step.

Lecture 9, 25.11.93

We have now reached the ground types, that is, the type of sets and the type of propositions, and we have to explain them and the rules associated with them. Let me begin with the type of sets.

The axioms and rules associated with the type of sets are the axiom that set is a type, and that if A is a set, then A is a type, and a corresponding equality rule that if A and B are the same set, then they are also the same as types. The equality rule corresponding to the ground type of set is of course that set = set : type, which is an instance of reflexivity.

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

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

Let us begin with the first axiom, set : type. According to the general explanation of what a type is, we have to answer two questions in order to explain that axiom, namely,

- What is a set?
- What does it mean for two sets to be the same?

The answer to the first question is the criterion of application, and the answer to the second question is the criterion of identity.

Let me begin with the first question, which is no doubt a difficult question. The confusions surrounding the notion of set when set theory was created, so to say in one blow, essentially about a hundred years ago by Cantor—the difficulties were enormous: the difficulties in getting a sufficiently clear concept so that one could see what axioms hold for it. At least part of the difficulties is that the notion of set is a mixture of three different concepts. In order to clarify the notion of what I call set here, we had therefore better first of all become clear about what these three different concepts are.

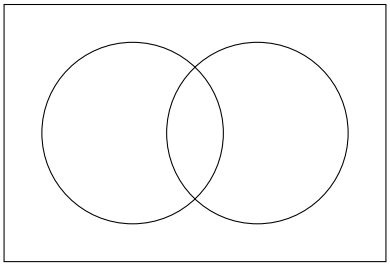
The three concepts I have in mind is, first of all, the notion of class, where a class is always a class of individuals belonging to some universe of discourse, so it is a subclass of a universe of discourse. The notion of class is of course the central notion of the Boolean tradition. When Cantor created set theory in the 1880s, class theory was already there, which makes it quite clear that Cantor's set theory cannot be class theory, because that would not have been any essential novelty at the time.

About the notion of class I need to say essentially nothing, I think, because, for any set A, I have defined class(A) as the type of propositional functions over A,

$$\frac{A : \text{set}}{\begin{cases} \text{class}(A) : \text{type} \\ \text{class}(A) = (A)\text{prop} : \text{type} \end{cases}}$$

The notion of class can thus be introduced by an explicit definition. The class operations—union, intersection, complement, etc.—are also given by explicit definitions, so class theory is really something quite trivial. There is no power in it at all. It just consists of a number of explicit definitions that you have to learn how to manipulate.

This is class theory, but observe that class theory already presupposes something, because, as I have said, a class is always a class of individuals belonging to some universe of discourse. Usually we draw figures like this:



There are two classes, and the square symbolizes the universe of discourse. The universe of discourse itself can of course not be a class, because, if it were to be a class, it would have to be a class of elements, or individuals, belonging to some other universe of discourse, and that cannot go on indefinitely: eventually the universe of discourse must be something which is not a class.

The question is then, of course, What is a universe of discourse? That question leads us to the notion of set as introduced by Cantor in the 1880s, which is, to an unusually high degree, as far as I understand, the conception, or the result, of a single person's conception. True, Cantor was inspired by Bolzano, and the very term set was already there. There was some hesitation to begin with what term to use, and there were three terms around: Mannigfaltigkeit, Inbegriff and Menge. All of these were used by Bolzano—as far as I know, Menge appears for the first time in Bolzano's *Paradoxien des Unendlichen* from 1851—and of these words, which were used interchangeably to begin with, Menge was the winning one, so it became the standard term. ([B.G.S.:] "What is the etymology behind "set?" Set, oh it is like a set of porcelain or a set of tools or something like that. It is that idea.)

Since classes were already there and are unproblematic, Cantor's notion of set should rather be thought of as an attempt to make precise what a universe of discourse is. That Cantor's set theory is not class theory is clear, first of all, if you look at his attempted definitions of the notion of set—they do not say that a set is a subset of some universe of discourse—but it is most clear if you look at what are the basic operations on sets that Cantor has. The basic operations on classes are

union, intersection and complement, but those are not Cantor's basic set-theoretic operations. Rather, his basic set-theoretic operations are those that correspond to to the addition, multiplication and exponentiation of cardinalities, which he thought of as generalizing the ordinary arithmetical operations for numbers. Cantor's three most basic set-theoretic operations were to form the union of two disjoint sets, or as we would now say, the disjoint union of two sets, corresponding to addition of cardinalities, and given two sets A and B, to form the Cartesian product of A and B, corresponding to multiplication of cardinalities, and given two sets A and B, to form the set of all functions from A to B, corresponding to exponentiation of cardinalities. From this, it is completely clear that what Cantor had in mind is something different from class theory.

Cantor's set theory, in the form that he presented it, was not formalized. It was developed intuitively. It is, so to say, a giant attempt at conceptual creation. Although Cantor's work was largely simultaneous with Frege's, there was therefore a very noticeable difference: although they were both concerned with the conceptual foundations of mathematics, Cantor's work is done in the object-oriented attitude, which is the natural one for a working mathematician, so he was dealing with objects throughout, essentially never any talk about linguistic expressions—and quite poor notation, one would say, as a logician—whereas Frege had developed this new method, which I have called the syntactical-semantic method, of being absolutely precise about the notation all the time and explaining what the notations mean. There is thus this profound difference in spirit, so to say, between Cantor's and Frege's work. Frege's virtues are obvious—to achieve the formal precision that Frege achieved was a great feat at that time—but we must remember that not everything is, so to say, on the side of Frege here, because, after all, Cantor was the one who had the better sense of what was needed for mathematics. When it comes to providing a workable foundation for mathematics, it was set theory that had the greater influence.

As I have said, Cantor himself did not formalize set theory: that was for the next generation coming after Cantor beginning with Zermelo. We then have the curious fact that the set theory that was formalized by Zermelo, Skolem and Fraenkel, and by Gödel and Bernays—that set theory formalized yet another notion of set, distinct both from class and from Cantor's notion of set: it formalized what we now call the iterative, or the cumulative-hierarchy, notion of set. ([B.G.S.:] "Was the omission of von Neumann in the list of names you mentioned deliberate?" No, no.) This is the idea of having sets whose elements are sets, which thus again have elements, and so on, in a well-founded manner. There is no trace, as far as I can tell, of that idea in Cantor. If you look at Cantor's definition of set, of which there are three versions, there is absolutely no trace of the idea that we have sets of sets of sets, etc., so that is already one indication that this is a different notion of set from Cantor's. Another is—let us look at what the basic set-theoretic operations are. We have already seen what Cantor's basic set-operations were. What are the basic operations for forming iterative sets? Well, it is pairing and union, in the sense of the cumulative hierarchy, and a few other operations, separation and replacement,

and infinity also. It does not matter exactly what they are, but it is quite clear that the basic set-theoretic operations in Zermelo–Fraenkel set theory are not Cantor's fundamental operations of disjoint union, Cartesian product and exponentiation.

We have therefore actually three distinct notions of set here. I have already said that classes presuppose a universe of discourse which is not itself a class. What about Zermelo–Fraenkel set theory? There the picture is particularly instructive, because we have all the three notions coming in at the same time. First of all, we have the universe of discourse of set theory, which is what is called the set-theoretic universe, V as it is usually called. That is not itself an iterative set. The elements of the set-theoretic universe are the iterative sets, whereas the set-theoretic universe itself, which is what the quantifiers in set theory range over, is not itself an iterative set. Second, the iterative sets are, as I have said, rather the elements, or individuals, in this set-theoretic universe, the elements of V. And third, since V, the set-theoretic universe, is a universe of discourse, we have of course classes of elements of V, as we have with any universe of discourse. So we have classes, and that is what is called classes simpliciter in connection with Zermelo–Fraenkel and Gödel–Bernays set theory. They are even treated formally in the Gödel–Bernays formulation of set theory as opposed to in the Zermelo–Fraenkel formalization.

In Zermelo–Fraenkel set theory, all three notions thus come in. Classes I have already talked about, it is an unproblematic notion, but the notion of iterative set itself needs careful clarification. Within the present type-theoretical conceptual framework, that belongs to Aczel, who has given an admirable series of papers dealing with the clarification of the cumulative hierarchy within this conceptual framework. What remains then is again the universe. Just as with classes, the problem is, What is the universe of discourse? What kind of thing is the set-theoretic universe, which the quantifiers range over in set theory?

This formulation of the question leads me to the other starting point here, which is to look at the quantifiers. We all think that the invention of the quantifiers was, if not the, then at least a very crucial step in the development of modern logic. We certainly count the quantifiers to logic: not only the propositional connectives, but also the quantifiers we count as logical operations. The question then arises, What do the quantifiers range over?

We may form a quantified statement, $(\forall x \in D)P$: this is a proposition, provided D is a quantificational domain and P is a proposition depending on a variable x, namely an element of D,

$$\frac{D \text{ domain} \qquad \begin{matrix}(x \in D)\\ P \text{ prop}\end{matrix}}{(\forall x \in D)P \text{ prop}}$$

This is the rule for forming universally quantified propositions.

What I call domain here, or quantificational domain, is the same as what we call individual domain, because the elements of such a domain is what we call individuals, although the word individual here is particularly unfortunate. It comes

from the Boolean tradition, as far as I know, but individual species is a well-established, Aristotelian terminology. If one were to choose between individual and species here, surely it should be a domain of species rather than individuals, because in logic we do not talk about individual things—this particular occurrence of 0 or something like that, individual occurrence of 0—we always talk about the number 0, or, if we are talking about the expressions, it is the type rather than the token. So individual is an unfortunate term here, but we have to make do with it. Now, individuals are the elements of the universe of discourse in Boolean logic, so an individual domain is just the same as what I have called the universe of discourse.

We also have quantification over classes, but that is again an unproblematic notion because it is just given by the definitions of the relativized quantifiers:

$$(\forall x \in (x \in D|P))Q = (\forall x \in D)(P \supset Q)$$
$$(\exists x \in (x \in D|P))Q = (\exists x \in D)(P \wedge Q)$$

This reduces quantification over classes to quantification over a universe of discourse, but surely, if we are dealing with predicate logic, we have to have this notion of domain, or universe, of discourse, hence that notion needs clarification.

In connection with this, let me also say that this shows that quantificational logic presupposes set theory, because we have at least to have so much set theory—or domain theory—that we have one quantificational domain: we can never get around needing to have at least some set theory, and that set theory is prior to quantificational logic. There is therefore no reduction of set theory to pure logic: it is the other way around. Hence, if logicism is thought of as the doctrine that everything should be reduced to logic, so that also the notion of set should be reduced to purely logical notions, then logicism is wrong, because we must have at least one set. This fits the common view that the conceptual basis of mathematics is a combination of logic and set theory. It is a logico-set-theoretical basis that we have.

We thus have to clarify this notion of quantificational domain, or individual domain, or universe of discourse, and that is what I propose to use the word set for: set as distinct from class, and as distinct from the iterative, or the cumulative-hierarchy, notion of set.

quanificational domain = domain of individuals = universe of discourse = set

It is clear that, on this point, we are not going to be helped by studying Frege. Frege had a domain, namely his domain of Gegenstände, and one might wonder, How did Frege explain semantically his objects? This is the point where Frege's semantics is most obviously flawed. We know that his formal system is flawed, since it is inconsistent. Hence, since his semantics is meant to justify, or explain, the formal system, there must be something wrong somewhere in his semantics. The most obvious point where it is just fatally wrong is in his explanation of what—he does not put it in the terms of saying what a Gegenstand is, but, as usual, he says what it means for a Gegenstandsname to denote. So we shall have no help from there. It is of course interesting to study Cantor's attempted definitions of a set, but they

are not of much help either. We have to give a much more precise definition, or precise account, of what we understand by sets.

I shall not quote those Cantorian definitions, but I do want, because of what we talked about during the lecture last time, hence for another reason, to quote the end of his first definition of set, which is from 1880. There he says that a set has to be defined, first, by explaining what an element of a set is—now I am using my terminology—and, secondly, we have to explain what it means for two elements of the set to be equal. There is thus very clearly both the criterion of application and the criterion of identity in his definition of set. This last part about the criterion of identity reads as follows:

> Ob zwei zur Menge gehörige Objekte trotz formaler Unterschiede
> in der Art des Gegebenseins einander gleich sind oder nicht.

Here we thus have the term Art des Gegebenseins from 1880, in a text which we know that Frege read, because surely he studied all of Cantor's work in set theory. It is clear that the simple-minded interpretation of Art des Gegebenseins that I have been using is what Cantor has in mind here. He speaks about "trotz formaler Unterschiede in der Art des Gegebenseins," which is what we ordinarily express by saying that we have, say, one and the same object, but we have formal differences in the way it is presented, as in

$$2 + 2,\ 2 \cdot 2,\ 2^2$$

Here we have the same number presented in formally three different ways: the differences here are formal differences. Formal difference is what you ordinarily would express by saying that they are just notationally different, and that is the same as saying that they are different as far as their expressions are concerned, but the object is one and the same.

It is thus clear that the Art des Gegebenseins of the object is the presentation, or the expression, of the object: mode of presentation or mode of expression, it does not change the substance. That also explains why Frege was so careful when defining his notion of sense in "Über Sinn und Bedeutung". He does not say that sense is the Art des Gegebenseins, because that would, so to say, conflate expression and meaning, but he says that sense is "das worin eine Art des Gegebenseins enthalten ist." The natural interpretation is that sense is the object including its expression. Frege's notion of sense is, I think, best understood as the object together with its expression. Since the expression is included, a sense can be identified by the expression, so that identity of sense comes down to identity of expression, as I have said. This may sound a bit forced, but it somehow seems more logical than to say that the Sinn simply is the expression. The objection against that is that it would be to conflate the expression and its meaning, and that distinction is so firmly rooted that Frege did not want to do that.

There is, however, another place where you actually find that carried out to its radical conclusion, and that is in Husserl, whose thinking about Sinn and Bedeutung and Noema is at least partly inspired by Frege and very close to Frege's. In *Ideen*, which is a late text in dealing with these matters, Husserl comes to say

the following, absolutely puzzling and shocking thing when you see it the first time—moreover, it is spelled out in double spaces, so it stares you in the face: "L o g i s c h e B e d e u t u n g i s t e i n A u s d r u c k." This is in *Ideen* I, § 124. You wonder, when you see this, What on earth has happened? How can we contort our minds so that we can understand what is meant here? You then go to the previous page to see what is meant by logische Bedeutung, and there you find that Husserl speaks about " 'l o g i s c h e' oder 'a u s d r ü c k e n d e' B e d e u t u n g." It is thus the Bedeutung which expresses. That is completely in line with the previous sentence here, because if it is the Bedeutung which expresses—normally, of course, it is the expression which expresses—then that fits at least with the Bedeutung being the expression.

Here we have an example of someone who, so to say, carried this way of thinking about sense to its logical conclusion. Of course, that logical conclusion is at the same time a reductio ad absurdum of this way of expressing things, because, if anything is certain in semantics, it is that we have to make a distinction between the linguistic expression and what it means, its meaning, that is, its Sinn or Bedeutung (Husserl did not make any distinction between Sinn and Bedeutung). What Husserl writes about this may well be internally coherent, but it provokes some thoughts about what you can allow yourself in doing philosophy. The trouble for us philosophers is that the unphilosophical language is after all more secure: it is a firmer basis, so that if we begin to express ourselves in a way that blatantly conflicts with the way everybody normally expresses himself, then it is we who are in trouble and not ordinary people, so to say. Here is a particularly clear case of that: Husserl is certainly in trouble if he reaches the conclusion that the Bedeutung is the expression. That is why I have been so careful in explaining a more natural way of accounting for the same phenomena, that is, the phenomena that have to do with the distinction between Sinn and Bedeutung. It is no doubt a more natural way to say that we have the same object expressed in different ways. That is why identity statements are informative and so on: it is because the expressions are not the same on the two sides of the identity, or the equality, sign.

This was just a parenthesis, accidental parenthesis, because of the quotation from Cantor's first set definition. ([B.G.S.:] "The tripartite distinction that you drew with respect to sets, is that completely new? The bifurcation into, should we say, set as extensional property or class or something like that versus the cumulative hierarchy is well-known, for instance from Hao Wang's book, but I have never seen this squeezing Cantor's notion in between the two others. That seems to me to be new." Yes, what has amazed me is that Zermelo–Fraenkel set theory is taken to be a formalization of Cantor's set theory. It just seems to be generally accepted that it is an adequate formalization, and that seems to me not correct at all. There is no trace of a cumulative hierarchy in Cantor's writings, and still he developed a quite considerable amount of set theory.)

Now I have separated the notion of class and the cumulative-hierarchy notion of set from the notion of—whatever we want to call it—universe of discourse, quantificational domain, individual domain, or, as I shall say, set. The question is then

how we are going to explain this notion of set. What is a set? What does it mean for two sets to be the same? Let us first consider some examples. We have some familiar sets, and we can see how they are defined.

In Cantor's set theory, the most important basic sets were the finite sets and the set of natural numbers, so let us see how we define them. For the finite sets, let us take the example of a set with two elements, which is what computer scientists call bool,

$$\text{bool} : \text{set}$$

Suppose someone has not seen this notation before, so he is puzzled and asks: what set is bool? Your answer is then, Well, bool is just the set whose two elements are true and false, or 1 and 0, or whatever you want to call the two truth values. It is clear then how the explanation proceeds: since it is a finite set, I explain it by simply saying outright what its elements are, so I say

$$\text{true} : \text{bool} \quad \text{and} \quad \text{false} : \text{bool}$$

Or you use 1 and 0, if you prefer, which are the truth values used in the Boolean tradition, whereas true and false are Frege's notation for the truth values.

If we are more careful, as we know that we have to be, namely in also giving the criterion of identity for the elements, then nothing could be simpler: the criterion of identity is that true is equal to true and false is equal to false, and they must not be identified. The identity criterion is thus given simply by saying

$$\text{true} = \text{true} : \text{bool} \quad \text{and} \quad \text{false} = \text{false} : \text{bool}$$

When you write this, it is tacitly understood that there are no other rules for specifying when two elements of bool are the same.

It is quite clear that this is how we explain a finite set. Let us therefore go to the other, more complicated, example of the set of natural numbers. We must have a set of natural numbers, $\mathbf{N}$, and the question is, How is it defined? If someone asks you, What are the natural numbers?, you will either say, Well, it is 0, 1, 2, etc., or if you are a bit more logically trained, you will say that they are generated by the two principles that 0 is a natural number, and whenever you have a natural number, you can form its successor,

$$\mathbf{N} : \text{set} \qquad 0 : \mathbf{N} \qquad \frac{a : \mathbf{N}}{\mathbf{s}(a) : \mathbf{N}}$$

By application of these rules we get that $\mathbf{s}(0)$ is a natural number and then again $\mathbf{s}(\mathbf{s}(0))$ is a natural number and so on.

The set of natural numbers is thus defined by the first two Peano axioms: 0 is a natural number, and if a is a natural number, then the successor of a is a natural number. If we are more careful, then we also have to give the criterion of identity for natural numbers. We say that 0 is to be identical to itself and that if a and b

are identical, then $\mathbf{s}(a)$ and $\mathbf{s}(b)$ are identical,

$$0 = 0 : \mathbf{N} \qquad \frac{a = b : \mathbf{N}}{\mathbf{s}(a) = \mathbf{s}(b) : \mathbf{N}}$$

We have now defined what the natural numbers are and what it means for two natural numbers to be identical. Let us take one further example of a set, and let us not then take a basic set: let us take instead a set-theoretic operation, the simplest one, Cartesian product,

$$\frac{A : \text{set} \qquad B : \text{set}}{A \times B : \text{set}}$$

Suppose someone is in his first course in the university and does not know what the Cartesian product of two sets is. How do we explain to him what it is? Well, we say that it is the set whose elements are ordered pairs whose first component is an element of A and whose second component is an element of B. That is what we say, and again we must take it seriously that this is how we define a set like this, so we give the rule

$$\frac{a : A \qquad b : B}{(a, b) : A \times B}$$

If a is an element of A, and b is an element of B, then we form the ordered pair, for which we have a notation either like this, (a, b), or another kind of bracket, $\langle a, b \rangle$, and that is an element of $A \times B$. As always, it is to be understood that this is the only way in which we form elements of the Cartesian product. And again, if one is more careful and have realized the necessity of identity criteria, then we also have to say when two such elements are identical, but nothing could be simpler: the pair (a, b) will be identical to the pair (c, d) precisely if the components are identical,

$$\frac{a = c : A \qquad b = d : B}{(a, b) = (c, d) : A \times B}$$

I have thus explained the Cartesian product by displaying outright how its elements are formed and when two elements formed in that way are the same.

I think these examples are enough now to see the general pattern here. There are many sets and set-forming operations, but the explanations follow a common pattern. A general answer to the first question, What is a set?, could thus be that a set is defined by prescribing how its elements are formed as well as how identical elements are formed—formed, constructed or generated: you may vary the formulations.

There is a comment I want to make first of all, and I will make it in connection with the definition of natural number here. The first rule, which says that $\mathbf{N}$ is a set, is what I call a formation rule, a rule for forming a set, whereas the following rules are called the introduction rules. Another formulation of the answer to the question, What is a set?, is that a set is defined by its introduction rules, because the introduction rules tell you how the elements of the set are formed and how equal elements of the set are formed. What I wanted to remark is that there is an interdependence between the formation rule and the introduction rules. The introduction rules serve as the semantical explanation of the formation rule, but at the same time, in this case, that $\mathbf{N}$ is a set is a presupposition in the introduction

rules, because the form of judgement $a : A$ has as a presupposition A : set, as we saw in the very beginning. Since the form of judgement $a : \mathbf{N}$ appears in the introduction rules, they have as a presupposition that $\mathbf{N}$ is a set. You thus have, on the one hand, that the introduction rules presuppose the formation rule, and on the other hand, the meaning of $\mathbf{N}$ is read off from the introduction rules. So you have an interdependence here.

The situation is in fact very similar to the situation in a nominal definition. Let us take negation,

$$\frac{A : \text{prop}}{\begin{cases} \neg A : \text{prop} \\ \neg A = A \supset \bot : \text{prop} \end{cases}}$$

This is a typical example of a nominal definition, and the natural reading of it is that $\neg A$ is a proposition, namely, the proposition defined by saying that it is to be the same as $A \supset \bot$. The interdependence here is that $\neg A$ is defined by the judgement $\neg A = A \supset \bot$: prop, which in turn has as a presupposition that $\neg A$ is a proposition. This interdependence is the reason for the bracket, indicating that the two judgements have to come simultaneously in a typing part and a definitional part. Similarly in the definition of a set, we say that $\mathbf{N}$ is a set, namely, the set generated by the following rules, and then we give the introduction rules.

There is another similarity between a nominal definition and the definition of a set, and that is that the defining part of the nominal definition is stipulatory in character: it is a stipulation which says that $\neg A$ is to be the same proposition as $A \supset \bot$. It is similar in an inductive definition. We say that $\mathbf{N}$ is a set, namely the set generated by the first two Peano axioms, and they simply stipulate that 0 is to count as a natural number and that if a already counts as a natural number, then so does the successor of a. If someone asks me to explain, or to give a semantical justification, of the first two Peano axioms, the answer is that they are just stipulations. Here the explanations end, just as with a nominal definition.

That was the first comment, and the other comment is an objection that immediately comes to mind: can this be all the natural numbers? What about $2 + 2$? That cannot be formed by means of these rules and is clearly a natural number. Some qualification is therefore needed here, and the answer is that the way I have expressed myself in giving these examples is the way we ordinarily express ourselves. We are speaking about objects all the time, not caring about how they are expressed, that is, we are not caring about the Art des Gegebenseins at all. We are imagining that we can deal with the objects directly, and surely there are no other objects, that is, no other elements, of the set of natural numbers than these, thinking of them as objects, though they may be presented, or referred to, in various ways. So surely, all of these are also natural numbers:

$$2 + 2,\ 2 \cdot 2,\ 2^2 : \mathbf{N}$$

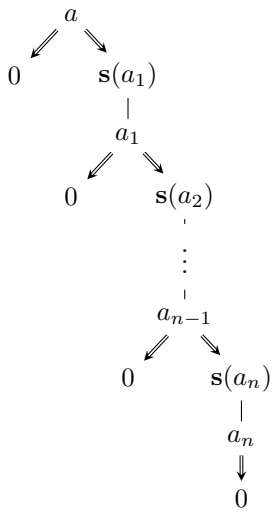
In defining the natural numbers by means of the first two Peano axioms, we are therefore taking it tacitly for granted that whatever evaluates to one of these

two forms, that is 0 or $\mathbf{s}(a)$, is also a natural number. Let us take addition, defined, although I have not done it so far, by the usual equations,

$$\begin{cases} a + 0 = a : \mathbf{N} \\ a + \mathbf{s}(b) = \mathbf{s}(a + b) : \mathbf{N} \end{cases}$$

The number 1 was defined as $\mathbf{s}(0)$, and 2 was defined as $\mathbf{s}(1)$. Then $2 + 2$ is an element of $\mathbf{N}$, because when you calculate it—so how does that calculation proceed? You have to see if the second argument is 0 or of successor form, so you have to look at the second argument. Since it is not, you have to look at the definition of 2, putting $\mathbf{s}(1)$ there instead, so that is $2 + \mathbf{s}(1)$, and now we use the second clause to get that this equals $\mathbf{s}(2 + 1)$, which is in successor form, hence it can be gotten by the second generation rule. Then when we pick out $2 + 1$, why is it a natural number? Well, it is a natural number because when we start computing it—and now 1 is neither 0 nor of successor form, so we have to use the definition of 1 and get $2 + \mathbf{s}(0)$, which is $\mathbf{s}(2 + 0)$. Now it has been brought to successor form, so it remains to see that $2 + 0$ is a natural number. That follows because $2 + 0 = 2$, by the first clause, and 2, on the other hand, is equal to $\mathbf{s}(1)$, so now it has been brought to successor form. Then we take out 1, which is still not of 0 or successor form, but it is sufficient to use the definition of 1 to get $\mathbf{s}(0)$, and then 0 is not computable, so this is why $2 + 2$ is also in $\mathbf{N}$.

When we define the set of natural numbers by the first two Peano axioms, we are thus tacitly ignoring the fact that we may have to make definitional replacements in general to get something of successor form or of the form 0. The more elaborate way is therefore to say that, What does $a : \mathbf{N}$ mean? It means that a, when you evaluate it, either has value 0 or it has value $\mathbf{s}(a_1)$. Then, when I pick out a_1 and begin to calculate it, it will have value either 0 or $\mathbf{s}(a_2)$, and so on, until, when I pick out a_{n-1}, it will have value either 0 or $\mathbf{s}(a_n)$, and then when I pick out a_n, it will be computed to 0,



So this is really how the inductive definition is to be understood if you take account of the fact that the objects in general need not be presented in primitive

form, but in defined form. Once you have said this, however, in the future we may always forget about it and stick to the usual way of expressing ourselves.

I think this is all I have to say as an answer to the question, What is a set? All these explanations are given essentially correctly in my little book *Intuitionistic Type Theory*.

We do not, however, yet have the type, or the category, of sets, because we also need the identity criterion. Remember what rules we are trying to justify:

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

I have given the criterion of application for the category, or type, of sets. The second rule here is also clear now, because the definition of a set stipulates how its elements are formed and how equal elements are formed—if we are more careful we say how its primitive, or canonical, elements are formed and how equal primitive, or canonical, elements are formed. The definition of the set simply stipulates that, hence we then know what an element of the set is and what it means for two elements of the set to be the same. That is what we must know in order to know the type A, because a type is always defined by saying what an object of that type—in this case an element of the set—is and what it means for two objects of that type—in this case two elements of the set—to be the same. That is, however, stipulated by the definition of the set. The second rule is therefore clear.

The identity criterion for the type, or category, of sets is then simply given in such a way as to validate the third rule. We have to see to it that if A and B are identical as sets, they are identical as types. The simplest way to achieve that is to say that they are identical as sets by definition if the types they give rise to are the same. Then the rule is obviously validated. What does that amount to? If you remember, we defined α to be identical to β if we may infer both ways by these rules:

$$\frac{a : \alpha}{a : \beta} \qquad \frac{a = b : \alpha}{a = b : \beta}$$

That was the definition of α being equal to β, so in this particular case α is the set A, and β is the set B, so the definition of $A = B : \text{set}$, of A and B being identical sets, is that it means that we may infer both ways by these rules:

$$\frac{a : A}{a : B} \qquad \frac{a = b : A}{a = b : B}$$

This fixes the identity criterion for the category of sets. Thereby the definition of the category, or the type, of sets and, for each set, the definition of the type of elements of that set are complete.

I will now turn to the other ground type that we have met, namely the type of propositions. We have used several times already the axiom $\text{prop} : \text{type}$. This axiom has been presupposed, so to say, in all logic that is based on the notion of proposition, that is to say propositional logic, to begin with, and later in predicate logic. Surely, it was always assumed that we have some objects that we call propositions that we can let the logical operations operate on. If objects always have to

have a type, that means that a type of propositions was presupposed, but explicitly and formally this type appears only in the *Principia*.

We thus have to answer the two usual questions again, What is the criterion of application?, which is to say, What is a proposition?, and, What is the criterion of identity?, that is, What does it mean for two propositions to be the same? Observe that without the identity criterion, we just do not have a type, or category.

Let us begin with the first question: What is a proposition? Here there are several answers, some in conflict with others and some compatible with others in the sense that one may be considered as an elaboration of, rather than genuinely different from, the others.

Let me begin—although it is not historically the first—with the Boolean conception. On the Boolean conception, a proposition is just a truth value. A proposition is just an element of the set bool, one of the two truth values. That is the Boolean conception, and that is the notion of proposition that Frege tried to build *Grundgesetze* on—not *Begriffsschrift*, because there we have *beurteilbare Inhalte*, but in *Grundgesetze* he simply takes the truth values to be the Bedeutungen of sentences, which means that, in effect, he is treating propositions on the level of Bedeutung as nothing but truth values. The affirmative judgement, A is true, is then simply analyzed as $A = \text{true}$, and the judgement A is false is simply understood as $A = \text{false}$.

On this conception, certain things work fine, namely the propositional connectives, because on this conception to define the propositional connectives we simply have to give the usual truth tables, which were given around 1920, explicitly as such by Wittgenstein and Post and Łukasiewicz. The problem is that we do not only have propositional logic: we also have predicate logic, so we have to make sense of the quantifiers. In particular, we have to make sense of the rule for forming quantified propositions that I wrote up earlier, namely the rule

$$\frac{D \text{ domain} \qquad \begin{matrix}(x \in D)\\ P \text{ prop}\end{matrix}}{(\forall x \in D)P \text{ prop}}$$

If anything is certain in quantificational logic, it is that we must have this rule.

On the Boolean conception of proposition the premisses here mean that we have a domain D and a Boolean function over D, which is to say that for each element d in D we have—let me write out the variable dependence of the function—$P(d)$ equal to 0 or 1, or true or false. There are no problems with that, of course, but in order to explain the quantifiers, we must explain what truth value the proposition in the conclusion denotes, or evaluates to. That is usually done by saying that, in the universal case, $(\forall x \in D)P(x)$ is equal to true—it is a case definition—if $P(d)$ is equal to true for every element d in the quantificational domain, and it is false otherwise,

$$(\forall x \in D)P(x) = \begin{cases} \text{true} & \text{if } P(d) = \text{true for every } d \in D \\ \text{false} & \text{otherwise} \end{cases}$$

For the existential quantifier there is a corresponding clause,

$$(\exists x \in D)P(x) = \begin{cases} \text{true} & \text{if } P(d) = \text{true for some } d \in D \\ \text{false} & \text{otherwise} \end{cases}$$

This is the analogue of the truth tables for the quantifiers.

These definitions have, as usual, a left-hand side, a definiendum, and a right-hand side, a definiens. The problem with them, however, is that we cannot pass from the definiendum to the definiens in general, namely, unless the domain D is given by an explicit list of its elements, which means that it has to be a finite set whose elements are explicitly listed. Then we can pass from the definiendum to the definiens by simply calculating $P(d)$ for all elements of D. However, if D is not a finite set given by an explicit list of its elements, but an infinite set like the set of natural numbers, then this does not work.

Simple minded as it may seem, this is really the crucial objection to two-valued semantics. Why does two-valued semantics not work? It does not work because we have to give up something which is so fundamental that we are rather willing to give up two-valued semantics than giving up it, and that is the principle that definitions should be eliminable: whenever we make a definition, so that something is defined rather than primitive, then we should be able to eliminate the definition. If it is done in several steps, that gives rise to a definitional chain, but that definitional chain must eventually lead us to something primitive, and in this case it means that the definitional chain must lead us to one of the truth values, either true of false. The problem with the definitional chains now is that we have a step which we just cannot bridge: it is impossible to get effectively from the definiendum to the definiens. [B.G.S.:] ("This is very much Kronecker's way of putting the objections to classical analysis, in the sense that it leads to incalculable functions.")

Yes, so that is the basic objection, and, as far as I can see, the criticism cannot be reduced to anything more convincing than that. It is objectional on semantical grounds to have non-eliminable definitions, which is what I have stressed here, but if you need somehow more down-to-earth arguments, it would be this. Think of how mathematics is used. Mathematics is, after all, something that we use in daily life, or in human life, and the way it is used in science, in natural science, in physics in particular, is that we build our mathematical models and, instead of carrying out an experiment and just seeing what the result will be, we make computations in our mathematical model and calculate beforehand what the result is going to be. We are sure that the rocket is going to land in the right place, or whatever it is, by calculation. Clearly then, if calculations cannot be carried out any longer, that is, if we reach steps in our calculation where we cannot proceed any further, then mathematics cannot fulfil that need, and if you think that that is perhaps the most important use of mathematics, then you will agree that it is objectionable to prevent functions from being calculable.

For these reasons—both semantical reasons and practical reasons, if you want—the Boolean conception of propositions just does not work when it comes to quantification over infinite domains. So something else must be tried, and the first

natural step then is to go back to something which is earlier than the Boolean conception. You find it in Stoic logic, and you find it in Ockham, for instance, that the meanings of "and" and "or" and so on—the meanings of the propositional connectives—are explained by giving the truth conditions: you say that $A \,\&\, B$ is true, provided A is true and B is true, and similarly for disjunction, implication, and so on.

That is now the answer to the question, What is a proposition? It says, not that a proposition is a truth value, but that a proposition is defined by its truth condition. Let me write these truth conditions in a perhaps somewhat unusual form:

$$\frac{A \text{ true} \qquad B \text{ true}}{A \,\&\, B \text{ true}} \qquad \frac{A \text{ true}}{A \vee B \text{ true}} \qquad \frac{B \text{ true}}{A \vee B}$$

$$\frac{\begin{array}{c}(A \text{ true})\\ B \text{ true}\end{array}}{A \supset B \text{ true}} \qquad -$$

$$\frac{\begin{array}{c}x \in D\\ P(x) \text{ true}\end{array}}{(\forall x \in D)P(x) \text{ true}} \qquad \frac{d \in D \qquad P(d) \text{ true}}{(\exists x \in D)P(x) \text{ true}}$$

For conjunction it is that if A is true and B is true, then $A \,\&\, B$ is true. For disjunction it is that, if A is true, then $A \vee B$ is true, and similarly, if B is true, then $A \vee B$ is true. For implication it is that, if B is true, provided that A is true, then $A \supset B$ is true. Falsehood is true on no condition, so there is no truth condition for falsity, which is different from saying that truth conditions are not given for falsity: I am giving the truth condition for falsity by saying that the number of clauses in its truth definition is zero. Then we have the quantifiers. If $P(x)$ is true for an arbitrary x in the domain, then $(\forall x \in D)P(x)$ is true, and finally, if we have some element d from the domain, and P is true for that element, then the existential proposition is true.

These are the standard truth conditions for the logical operations, and on this conception of proposition, they are to be taken as defining the propositions in question. The meaning of the proposition is defined by saying under what condition it is true, what it means for it to be true. As I have said, such explanations have been there from the very beginning, but the term, the explicit formulation of this conception of what a proposition is, using the term truth condition, is in the *Tractatus*. ([B.G.S.:] "In § 32 of the *Grundgesetze*, you will also find that a proposition will be determined by a truth condition, and the thought expressed is the thought that this condition is fulfilled." Aha, so almost... so condition at least... "Condition in connection with determining truth is given by Frege." But not the word Wahrheitsbedingung. So the Frege–Wittgenstein conception we could say.)

This is quite different from the Boolean conception. The intuitionistic conception of proposition that I shall reach, although it is in blatant contradiction with the Boolean conception—it is just a complete rejection of the Boolean conception—is

not in conflict with the idea that a proposition is defined by its truth condition. It is rather an elaboration, a more precise elaboration, of the idea that a proposition is defined by its truth condition.

What are the difficulties with the conception that a proposition is defined by its truth condition? One very clear difficulty is the following. In order to have the type, or category, of propositions, we have to explain, not only what a proposition is: we must also explain what it means for two propositions to be the same. If all we know about the proposition is its truth condition, which is to say, what it means for it to be true—that is, the only thing we know is what A is true means—then, if we are going to define what it means for two propositions to be the same, we have nothing else to go on than this. How could we then give the identity criterion if not by saying that, A is identical to B if we may infer both ways, from A is true to B is true and vice versa?

$$A = B : \text{prop} \quad \Leftrightarrow \quad \frac{A \text{ is true}}{B \text{ is true}}$$

This notion of equality between propositions is, by definition, logical equivalence, and if identity between propositions is logical equivalence, then any two true propositions are identical according to that definition. An interesting proposition, like Euclid's, say, about the infinity of the prime numbers, $(\forall x : \mathbf{N})(\exists y : \mathbf{N})(x \leq y \ \& \ \text{Prim}(y))$ is logically equivalent to $\top$. (I have not introduced $\top$, but just as we introduced absurdity by saying that it is true on no condition, you define $\top$ outright by giving its truth condition simply as true.) Hence if identity between propositions is logical equivalence, then these two would be identical, and of course that is very counterintuitive.

([B.G.S.:] "There are various refinements possible, because here you would have really only the truth of the material equivalence. This would not be really logical equivalence, because you have also the logical truth of the equivalence. There is a difference in strength here: we would not say that 'The rose is red' and Euclid's theorem are logically equivalent, but their material equivalence is true, albeit not a logical truth." So should I call it material equivalence? "I think you should call it material equivalence. Wittgenstein shows us exactly the logical equivalence as its truth condition, so all tautologies, which are logically equivalent, are equal to each other, but that is less strong than to say that all propositions which are materially equivalent are equal. It is still too strong, but it is more refined.") Next time I will get out of this conundrum, get out of this difficulty.

Lecture 10, 2.12.93

We are in the course of explaining the axiom that propositions form a type, and by the general explanation of what a type is that means that we have to answer the two questions,

- What is a proposition?
- What does it mean for two propositions to be the same?

Last time I dealt with the first attempt at answering the first question, namely, the Boolean conception of a proposition as a truth value, also adopted by Frege in the *Grundgesetze*. As I said, that conception simply has to be rejected because of the difficulties of then interpreting quantification over infinite domains.

The second attempt that I started on was the explanation of a proposition as being defined by its truth condition, and then I rapidly wrote down the usual truth conditions: A & B is true if A is true and B is true, etc. Then I said that if the only thing we know about a proposition is the condition under which it is true—which is the same as what it means for it to be true, so we know what A true means—then it seems that there is no way of answering the second question, that is, of laying down the criterion of identity between propositions, than by saying that A and B are the same proposition if one is true if and only if the other is true, which is to say, if and only if we may infer both ways by the following rule:

$$\frac{A \text{ is true}}{B \text{ is true}}$$

We have, so to say, no other information about a proposition than what it means for it to be true, so what else could we say here?

The notion of equivalence between two propositions which holds when one of them is true if and only if the other is true is a well-known notion, as Göran corrected me last time: it is the notion of material equivalence. I said last time, mistakenly, logical equivalence, but in one sense, there was no risk, because the notion of logical equivalence as we ordinarily understand it just does not make sense at this stage. The only thing we know about A and B here is what it means for A to be true and what it means for B to be true. There is absolutely nothing said about the internal structure of A and B or anything like that: A and B are purely schematic letters here, so the usual notion of logical equivalence, that is, that the equivalence holds under all interpretations or something like that, is simply not available to us at this stage. It is therefore clearly material equivalence that we are dealing with here.

Material equivalence identifies all true propositions, and it also identifies all false propositions. Hence, as I said, any interesting true proposition, like Euclid's proposition that there are infinitely many prime numbers, gets identified with any other true proposition, which may have a completely different content, and this is of course very counterintuitive. Euclid's proposition and the proposition $\top$, which is defined simply by stipulating that it is to be true, are both materially equivalent, yet they have a totally different meaning from each other.

This was then the difficulty that we ended with last time. How do we get an acceptable criterion of identity between propositions? Strangely enough, the help here comes from intuitionism. This is, so to say, help that was not available earlier. It was not available to Frege, for instance. Intuitionists define the notion of truth in terms of two underlying more basic notions. They introduce, for an arbitrary proposition A, the notion of proof of the proposition A, hence they introduce a new form of judgement,

$$a \text{ is a proof of } A$$

This is now proof in a sense that is distinct from the usual sense of proof when we talk about a proof of a mathematical theorem, because the proof here is an object, a mathematical object. In terms of this new notion of proof of a proposition, the notion of truth is defined by saying that there exists a proof of A,

$$A \text{ true } = \text{ there exists a proof of } A$$

Truth is thus defined in terms of the notion of proof and the notion of existence.

The notion of proof I have already introduced, and now comes the notion of existence here. Both of these have to be discussed, and let me begin with the notion of existence. One's first instinct may be that existence is a well-known notion expressed by means of the existential quantifier. The notion of existence that enters here, however, cannot be analyzed by means of the existential quantifier.

I will use the formal notation, $a : A$, for the relation of a's being a proof of A. The logic of this is that if we are treating proofs as mathematical objects, then they have to have a type, because every object has to have a type by the doctrine of types, and then the question is what type does a proof of a proposition have? The simple solution is to take the proposition itself to serve as the type of its proofs, so we lay down the following rule, a second rule here:

$$\frac{A : \text{prop}}{A : \text{type}}$$

If A is proposition, then A itself is a type, and then we can express that a is a proof of A in this usual notation, $a : A$.

To return to the notion of existence here: if you tried to express this by means of the existential quantifier naively, then you would begin writing $\exists x : A$—but then what to write next? If there came some B here, $(\exists x : A)B$, then this would be an existential proposition, but there does not come any B, so this is clearly not an existential proposition. If you tried to complete $\exists x : A$ into an existential proposition by writing $\top$ after it—the proposition $\top$ which is defined to be true—then you just get a new proposition, $(\exists x : A)\top$. The question is then, first of all,

what a proof of it is, and when you ask what it means for this proposition to be true, you get again that there exists a proof of it, and that would continue: you would have an infinite regress. That solution does therefore not help either.

The situation is in fact quite simple: the notion of existence that enters here is a different notion of existence from that expressed by means of the existential quantifier. There is nothing strange with that: we have an ambiguity in the word existence. We still have to explain carefully what we mean by existence in this new sense, since we cannot rely upon our old knowledge of the existential quantifier. I therefore introduce a new form of judgement,

$$\alpha \text{ exists}$$

where it is presupposed that α is a type—if you remember, every form of judgement has its presuppositions, and this new form of judgement has the presupposition that α is a type.

Since this is a new form of judgement, it is incumbent on me to explain what it means, or what I mean by it. The meaning can be, so to say, read off from the following rule for it:

$$\frac{a : \alpha}{\alpha \text{ exists}}$$

If a is an object of type α, then α exists. From this rule, you can, so to say, read off, or guess, what this new form of judgement means. The best way to understand this new form of judgement is that it is a kind of abbreviated judgement, an incomplete or abbreviated judgement. It is not an abbreviative definition, or nominal definition, in the sense that I dealt with at length before, but it is an abbreviation nevertheless. Namely, I leave out the object a of type α and retain only the information that there is such an object, without caring about exactly which object it is. I may not need that information at the moment, so I just give the information that there is such an object.

The terminology introduced by Hermann Weyl, the references to which Göran knows better than I do, was that of an incomplete judgement, or judgement abstract, Urteilsabstrakt, I think, ([B.G.S.:] "Ja."): we have abstracted from the object a and retain only the information that there exists such an a. ([B.G.S.: "Also the term partial Urteil is used in this connection." Ah, partial judgement. "I think incomplete might be Hilbert with incomplete communication." Yes, incomplete communication. Is that from Hilbert? "That is from Hilbert, I think, yes, but I am not certain, I have got everything upstairs if you. . . " Right, incomplete judgement, or incomplete communication. And maybe partial judgement would be. . . "Kleene made the incomplete communication well-known in the realizability formulations." Yes, the incomplete communication terminology is very natural here, and has, so to say, sifted into the language that we ordinarily use nowadays to talk about this situation.)

Paraphrasing the explanation slightly here, you could say that what the condition is that must be fulfilled for you to make the judgement, α exists, is that you actually possess the object a. You have the object, so that on request you can come up with it. It is somewhat like having a driver's licence, since there are not many

indirect ways that we accept here. It is not acceptable here, for instance, to say that if you run a certain program that requires millions of years to run, then you will find a. If you say $10^{10^{10}} : \mathbf{N}$, then we know that, by running this program, the end result will eventually be written out in the form $\mathbf{s}(\mathbf{s}(\ldots \mathbf{s}(0) \ldots))$, although it requires such a long time that it is not feasible for us to reach that end product: there is none of this kind of potentiality involved in the coming up, or presentation, of the witnessing object a here. It is rather like a driver's licence: it is okay if you do not have it with you at the very moment when the control is being made, but you should at least be able to come up with it at the police station the next day or something like that. It must be available even if you do not have it in your hand or in your mind at the moment. Hence, whenever you make the judgement α exists, you could, if you wanted to, always fill in the object a that is claimed to exist.

This should be enough about the meaning of this new form of judgement, α exists. Since we have the rule that if A is a proposition, then A is a type, we can put A for α in this new form of judgement, which then becomes the same in meaning as A true, because it says that there exists a proof of A,

$$A \text{ exists } = A \text{ true } (\text{provided } A : \text{prop})$$

I have already introduced the peculiarly intuitionistic notion of proof as mathematical object. The very idea of treating proofs as mathematical constructions, or mathematical objects, comes from Brouwer, but it remained for Heyting to apply that idea to the logical systems that were available at the time, notably to propositional logic and predicate logic, so as to get the intuitionistic interpretation of those two calculi. Again, Göran knows the history much better than I do here, but it was certainly new with intuitionism that proofs began to be treated as mathematical objects in such a way that it eventually affected our logical calculi.

When we introduce this notion of proof, then of course we still have our usual proofs, proofs of geometrical theorems and so on, and our usual proofs are certainly not mathematical objects: our usual proofs are what convinces us that a theorem is correct. We therefore now have two notions of proof: proof of a proposition and proof of a judgement. (A mathematical theorem is a judgement, and a proof of a mathematical theorem is therefore a proof of a judgement.) We can separate these notions by talking precisely about a proof of a proposition as opposed to a proof of a judgement, but it is also quite convenient to use the term verification for proof in this new sense, as a mathematical object,

$$\text{proof of a proposition } = \text{ verification}$$

If we mean a proof in the good old sense, then we also have a good old word for it, namely, demonstration. After all, demonstratio is the old Latin term for proof in this sense, the translation of the Greek ἀπόδειξις,

$$\text{proof of a judgement } = \text{ demonstration}$$

By introducing proof objects, we do not make this notion of proof superfluous: we always need that.

Since the proofs of a proposition, or the verifications, are objects, we will have the distinction that we have for all objects, and also for types, namely, between primitive and defined, or in this case, the terminology most commonly used is that of canonical versus non-canonical proof. This terminology comes from Brouwer—does it? [B.G.S.:] "Yes." Canonical proof? "Yes." So it goes back to Brouwer's 1927 paper on the Bar Theorem, and in recent discussion it is Dummett who has introduced it. If we use the word verification, then we have a good terminology that was introduced in the 1930s when the verification principle was extensively discussed, namely the notion of a direct versus an indirect verification.

This was just to make clear that we still have our good old notion of proof, proof as demonstration. I am not putting it into focus right now, though it is always needed in the background. If it will be necessary, I will say something about it, but for now we just have to take it as we are used to.

There is one more thing I wanted to say to clarify the new form of judgement α exists and that is to comment on a mistake that is commonly made—and which intuitionists are particular prone to—in explaining that form of judgement. It is very easy to say that α exists means that an object of type α has been found and that this is the intuitionistic notion of existence,

$$\alpha \text{ exists means that an object of type } \alpha \text{ has been found}$$

But this is quite incorrect. I mean, you have to be very benevolent to accept this way of speaking, because what is said in the judgement α exists is definitely not that an object of type α has been found, which is the same as is known: that is not what is said in the judgement. It is rather by saying that α exists, that is, that there exists an object of type α, that I show—and now I am using Wittgenstein's terminology here, say versus show—that I have found an object of type α. That I know an object of type α is not contained in the content of the judgement, that is, it is not in the verbal, or the locutionary part, of the judgement α exists: it is in the illocutionary part of the judgement. By making the judgement I show that I know such an object.

Since A true is a special case of α exists, exactly the same remark applies here: it is not correct to explain the notion of truth intuitionistically by saying that A true means that a proof of A has been constructed. The judgement does not mean that. What it means is precisely what I have said, namely that there exists a proof of the proposition A, but when I say that, that is, when I make the judgement A true, then I show that I have found such a proof. What entitles me to make the judgement is that I am in the possession of such a proof.

This is a mistake which intuitionists are very prone to, and it is almost impossible not to find this mistake, I think, when you look into intuitionistic texts. It has to do with the fact that the notion of existence that I am explaining here, and the correlative notion of truth, are not present in the literature before, at least not in any clearly distinguished way. ([B.G.S.:] "The slipperiness of the notion seems to lie in the fact that it gets explained by making use of assertoric force, but the

notion that gets explained is itself without assertoric force." Right.) So, enough about this.

Now I would like in connection with the notion of existence, and therefore also with the notion of truth for propositions, to talk about actuality and potentiality. As I have said, α exists means that there exists an object a of type α in the sense that I have just made clear,

$$\alpha \text{ exists } = \text{ there exists an } a \text{ of type } \alpha \tag{1}$$

From existence in this sense, which we might call bare existence, or existence simpliciter, we must distinguish what the intuitionists, as I have said, often confuse with it, namely the notion of actual existence. That α exists actually means that an object of type α is known—you may vary the formulation endlessly here: is known, has been found, constructed,

$$\begin{aligned} \alpha \text{ exists actually } &= \text{an object of type } \alpha \text{ is known} \\ &\quad \text{(has been found, constructed)} \end{aligned} \tag{2}$$

From both of these we must distinguish the notion of potential existence. That α exists in the potential sense means that an object of type α can be found, or constructed, or can be known, to use the same as in (2),

$$\begin{aligned} \alpha \text{ exists potentially } &= \text{an object of type } \alpha \text{ can be known} \\ &\quad \text{(found, constructed)} \end{aligned} \tag{3}$$

These are three quite distinct notions. The notions of actuality and potentiality that enter here are the good old notions discussed at such length by Aristotle, that is, the notion of ἐνέργεια and δύναμις. Also the qualification of existence as either actual or potential is from Aristotle, it is also from the discussion of existence in the *Metaphysics*.

There is an absolutely clear order of conceptual priority between these three notions of existence. Existence in sense (1) is presupposed in (2), because to say that α exists actually is the same as to say that the judgement α exists is known, and hence that α exists in sense (1) is contained as a component in α exists actually. On the other hand, there is a similar phenomenon between (2) and (3), because to say that α exists potentially is to say that the judgement α exists can be known, and that it can be known means that it can be known actually, it can actually be known. You will find this remark in Aristotle also, that when you say that something can happen, or that something can be, then what it means is that it can happen actually, or it can be actually. The notion of actual being, or actual existence, is therefore conceptually prior to the notion of potential existence. The can here is an additional operator, if you want, a modal operator, that operates on the notion of actual existence. This was expressed by Aristotle by the sentence, so often quoted in scholastic time, which says that πρότερον ἐνέργεια δυνάμεώς ἐστιν, actuality is prior to potentiality. The order in which it is prior is precisely the order that I talked about at such length in the beginning of this series, namely the order of conceptual priority. The Greek term is, as I remarked then, τῷ λόγῳ, according to formula, or definition. This was translated in the comments on this passage in

the *Metaphysics*—by Thomas, for instance, but maybe already by Albertus—into actus est prior potentia, and the translation of τῷ λόγῳ was ratione. The order of conceptual priority is therefore: (1) is prior to (2), and (2) is prior to (3).

Since A is true is a particular case of α exists, we just need to change α into the proposition A, and we get the same distinctions between A is true, A is actually true and A is potentially true.

For A is true we may say A is true simpliciter, and it means simply that there exists a proof of A,

$$A \text{ is true } = \text{ there exists a proof of } A \tag{1}$$

That A is actually true means that A is known to be true, as we would say if we do not have the proof object yet, but only the truth condition. Making the proof object explicit, actual truth means that a proof, in the sense of verification, of A is known, or has been found,

$$\begin{aligned} A \text{ is actually true } &= A \text{ is known to be true} \\ &= \text{ a proof (verification) of } A \text{ has been found} \end{aligned} \tag{2}$$

That A is potentially true means that A can be known to be true, which is the same as to say that a proof, in the sense of a verification, of A can be found,

$$\begin{aligned} A \text{ is potentially true } &= A \text{ can be known to be true} \\ &= \text{ a proof (verification) of } A \text{ can be found} \end{aligned} \tag{3}$$

This you may be familiar with from my metaphysics paper.

As I have said, the most important novelty here is the notion of bare truth, or truth simpliciter. You will find the notion of actual truth especially in the intuitionistic literature. The notion of potential truth has been very clearly introduced in a paper by Prawitz, for instance, from the late 1970s, and it is the notion of truth that Putnam aims at with his notion of warranted assertibility. You also find it in Dummett's writings from the 1970s, I believe. I have not traced the formulation "A is true if there exists a proof of A", but surely you can find it, though I doubt that you can find a clear distinction between (1) and (3). ([B.G.S.:] "I think you find formulation (1) in Dummett's *Elements of Intuitionism*, which dates from 1974, the first edition of that." Yes, but the question is what the existence here means and that you have to distinguish these three senses.)

The interesting situation is here that we somehow vindicate Frege after all. If you remember from the introduction to the *Grundgesetze*, he says

> Kann man ärger den Sinn des Wortes 'wahr' fälschen, als wenn man eine Beziehung auf den Urtheilenden einschliessen will!

Approximately: could you falsify the notion of truth more than by including a relation to the knowing subject? He is right on this point: the notion of bare truth, or truth simpliciter, contains no epistemic content at all. The epistemic content is only added on top of that, in (2) in the actual sense and in (3) in the potential sense. Perhaps one could also say here, concerning (2) and (3), that the intuitionists who are more idealistically inclined have been putting the stress on

(2), whereas those who are more Platonistically, or realistically, inclined have put the emphasis on (3). One might therefore call (2) the idealistic notion of truth and (3) the realistic notion of truth, but maybe it is better not to use those words. The actual order of conceptual priority between them—I think there is no doubt that (2) is conceptually prior to (3).

The notion of truth that enters into the truth conditions which are meaning determining for the logical operations is the notion of bare truth, or truth simpliciter. That is particularly clear in the case of implication, because the condition for an implication, $A \supset B$, to be true is that B is true provided that A is true,

$$\begin{array}{c}(A \text{ true}) \\ \vdots \\ \dfrac{B \text{ true}}{A \supset B \text{ true}}\end{array}$$

Above the inference bar, we have a logical consequence—or maybe I should say consequence—which says that B is true provided that A is true. It is quite clear that you can have no epistemic operators in the antecedent of that consequence. If you are to have an epistemic operator here, saying that something is known or is knowable, then it would have to apply to all of $A \supset B$: all of this could be known or knowable, but you certainly cannot have any epistemic operators in the antecedent or consequent of that logical consequence. The notion of truth that enters there is quite clearly the notion of bare truth, or truth simpliciter.

When truth of a proposition is explained as existence of proof, then truth conditions come to be replaced by proof conditions. Instead of truth conditions we thus have proof conditions. The previous explanation that a proposition is defined by its truth condition is then changed into the intuitionistic explanation of what a proposition is, namely that a proposition is defined by laying down how its proofs are formed as well as how identical proofs of the proposition are formed.

Only now that we have given that explanation do we have the right to the rule that I wrote up before, namely, the rule that if A is a proposition then A is a type,

$$\frac{A : \text{prop}}{A : \text{type}}$$

The conclusion here says that A is a type, hence we must know what an object of that type is as well as what it means for two objects of that type to be the same. The objects of a propositional type are its proofs, so we must know what a proof of the proposition is as well as when two proofs of the proposition are the same, but that is precisely what defines any proposition, as I have just said. When in the truth-conditional theory you ask, What does it mean for a proposition to be true?, the answer is, Well, that depends upon the proposition, because it is precisely that feature of the proposition which defines it as such. It is similar here: if you ask, What is a proof of a proposition?, and, What does it mean for two proofs of a proposition to be the same?, you say, That depends on the proposition, it is precisely that which defines the proposition.

This is thus a more refined explanation of what a proposition is, I mean, refined as compared with the truth-conditional explanation. As I said last time, it is not in conflict with the truth-conditional explanation, it is just an elaboration.

Now we can, finally, solve the problem that we were left with last time, namely the problem how to define identity between propositions. We are here in the process of dealing with the axiom prop : type, and in that process we have dealt with the above rule, if A : prop, then A : type. Since we all the time have to see to it that all operations are compatible with identity, in particular here with the passage from a proposition to the corresponding type, we have to see to it that that passage preserves identity, which is to say that we have to see to it that the following rule is satisfied,

$$\frac{A = B : \text{prop}}{A = B : \text{type}}$$

If A and B are identical propositions, then they are identical as types. It is this rule which is the clue to the notion of identity between propositions, because we have already fixed the criterion of identity for types,

$$\alpha = \beta : \text{type} \quad \text{means that} \quad \frac{a : \alpha}{a : \beta} \quad \frac{a = b : \alpha}{a = b : \beta}$$

Two types are the same if we may infer both ways by the rules that if a is of type α, then a is of type β and the rule that if a and b are identical objects of type α, then they are also identical objects of type β. That explains what this means, and then it is clear how we are more or less forced to define the notion of identity of propositions: we have to define it in such a way that this rule is validated. The simple solution is to define the propositions A and B to be identical if they are identical as types. We therefore have to take these two conditions with A inserted for α and B inserted for β,

$$A = B : \text{prop} \quad \text{means that} \quad \frac{a : A}{a : B} \quad \frac{a = b : A}{a = b : B}$$

We may infer both ways by the rule that if a is a proof of A, then a is a proof of B and the rule that if a and b are identical proofs of A, then a and b are also identical proofs of B. This fixes the criterion of identity between propositions.

Quine has been right in his criticism of the notion of proposition on the grounds that we lack a clear identity criterion. If propositions are to form a type, or category, then we have to lay down a clear identity criterion. In that sense, Quine is right, but Quine's conclusion has been the opposite one, namely that we ought to dispense with the notion of proposition, whereas I am meeting his challenge by providing, first of all, the criterion of application, so that we know what a proposition is, but also providing this identity criterion. As you see, the difficulties in providing it has been that earlier, that is, before intuitionism, one only had the notion of proposition as determined by its truth condition. If we only have the truth condition, it seems hopeless to get anything but material equivalence, whereas now that we have the proof objects, we get, in a perfectly natural way, this criterion of identity between propositions.

We now have three equalities between propositions:

(1) syntactical identity
(2) identity, $A = B$: prop
(3) material equivalence, $A \supset\subset B$ true

The first is syntactical identity, the notion of identity that we get from the expressions by transferring it over to the object. This is the notion of sameness of sense that Frege favoured. My analysis of Frege's notion of Sinnesgleichheit reached the conclusion that that was a very strict notion which boils down to syntactical identity, whereas we now have the notion of identity simpliciter between propositions. This is identity between the objects, that is in this case between propositions, and it is the notion of identity that has been lacking all the time. Thirdly we have material equivalence, which we may express as the bi-implication's being true, $A \supset\subset B$ true.

These three levels correspond, in the case of sentences, to Frege's threefold division into Ausdruck, Sinn and Bedeutung. The first is the notion of identity between the propositional expressions, that is, the Sätze in Frege's terminology. The second is the notion of identity between their senses, which is to say, between the Gedanken, and the third Frege took to be equality between the truth values. We no longer have access to the notion of truth value—that is a classical conception that we just have to dismiss—but we do have the notion of equality that corresponds to equality of truth values, namely the notion of material equivalence: it makes perfectly good sense, except it is expressed differently by means of the bi-implication. We do not have any new object in the step from (2) to (3): it is the same old propositions that we have here. The objects here are the same, but the identity criterion has changed, as we ordinarily say, from the intensional to the extensional identity criterion.

Once we have this notion of identity between propositions, we have the same for classes. Remember how classes were defined,

$$\frac{A : \text{set}}{\begin{cases} \text{class}(A) : \text{type} \\ \text{class}(A) = (A)\text{prop} : \text{type} \end{cases}}$$

Now that the definition of the type, or category, of propositions is complete, this is a good type, and we have already defined what it means to be an object of that type and what it means for two such objects to be the same. The identity criterion has already been laid down for the function type, and that means that we now have exactly the same threefold division here for classes,

(1) syntactical identity, $B \equiv C$: class(A)
(2) identity, $B = C$: class(A)
(3) extensional equality, $(\forall x : A)(B(x) \supset\subset C(x))$ true

We certainly have the notion of syntactical identity: two classes are the same in that sense if they are identically expressed. We also have the notion of identity simpliciter between classes now, that B and C are identical as classes. Remember what identity means on a function type: $B = C$: class(A) means that $B(a)$ is the same proposition—you see, we did not have access to this before we had identity between

propositions—as $C(a)$, for an arbitrarily chosen element a of the set A. That is the notion of identity between classes, and this is of course to be distinguished from extensional equality between classes. (I have a tendency to use "equality" rather than "identity" when it comes to extension at least. Does extensional identity make sense? Can one say that? It would be nice to use "identity" here, but what comes naturally for me is "extensional equality", because you do not want to use the word "identity" when they are not as equal as they possibly can be, and here they are certainly not.) What is then extensional equality between two classes B and C? That can be expressed by saying that for all elements x of the universe of discourse A, $B(x)$ is materially equivalent to $C(x)$.

As a notation for the syntactical identity, one could use the convention that when you put quotation marks around a type, the objectual type is transformed into the corresponding syntactical type, or category. In our case, you would say that "B" and "C" are expressions of the syntactical category, and they are identical as expressions, "B" = "C" : "class(A)". This is, however, unnecessarily complicated, I think. I am speaking of three different senses of identity between two classes, and I can write simply $B \equiv C : \mathrm{class}(A)$, because I have introduced the three bars precisely to denote syntactical identity.

I have carefully avoided until now the words intensional and extensional precisely because the patterns that we see coming up here do not fit completely with what we are used to. By now, however, it should be clear what is the most natural way of using the terms intensional and extensional here. Surely, we want to use extensional equality for this last relation between classes. That is absolutely clear, and if that is the extensional equality, or the extensional identity, then surely it is the second which deserves to be called the intensional one as opposed to the extensional one. That fits with Gödel's use of the terms intensional and definitional, because in a footnote to his *Dialectica* paper, from 1958, he says, intensional—with an s—or definitional equality.[1] I have stressed all the time that the term definitional is fine, because it is absolutely unambiguous, and the identity that I have for objects, and also for types, is properly called definitional identity. Calling it intensional identity therefore fits with this equating of intensional and definitional equality of Gödel's.

This again then is a meeting of Quine's challenge, now with respect to classes. We have, not only the notion of extensional equality between classes, which has made good sense all the time, but we also have this notion of intensional identity fixed by a clear identity criterion.

Since we have these three equalities, the notion of transparency and opacity has to be accordingly trichotomized. The notion of transparency in this connection comes from Russell, Appendix C in the second edition of *Principia*, the one from 1925, I believe. The corresponding notion of opacity, or opaque, was introduced by Quine. As you know, the question is whether in a certain context you may replace one expression by another expression salva something, salva veritate or

[1] "Identität zwischen Funktionen ist als intensionale oder Definitionsgleichheit zu verstehen."

salva intentione. Now we have three senses of identity here, and with respect to syntactical identity, it is easy. With respect to syntactical identity every context is transparent, because you do not change anything, so we get two senses here of transparency and opacity, namely an intensional and an extensional one. A context is intensionally transparent if you can replace an expression in that context—or maybe I could better say, an object in that context—by one which is intensionally identical to it, salva, whatever it is, veritate or something. It is extensionally transparent if I can replace that object by one which is extensionally equal to it, and now it is most certainly salva veritate.

All the time here I have taken care to see to it that every operation, be it a type-forming operation or an object-forming operation, preserves identity. It preserves intensional identity, identity in sense (2), salva what, ja, salva identitate, if that is the correct Latin form, because all the time I have seen to it that if the parts are identical, then what I form from those parts is again identical. We have seen countless examples of that. When I formed the function type, for instance, we had the rule

$$\frac{\alpha = \gamma : \text{type} \qquad \beta = \delta : \text{type}}{(x : \alpha)\beta = (x : \gamma)\delta}$$

Whenever I have some operation of forming types or forming objects I have seen to it that the operation is identity preserving, which is to say that the context that is created by this new operation is intensionally transparent salva identitate, so the whole language is in this sense transparent.

On the other hand, so-called propositional attitudes create, in general, contexts which are, not only extensionally, but even intensionally opaque. I am not going to discuss propositional attitudes, but we have at least one already implicitly introduced, namely that of knowing, because I have spoken about knowing a proposition to be true. Suppose we put in a person now, P, who knows that A is true, where A is presupposed to be a proposition,

$$\text{P knows that } A \text{ is true}$$

This is not even intensionally transparent, because A may be intensionally the same as another proposition B, and yet P may not have proved, or demonstrated that, he may not be aware of this intensional identity. In particular, he may not have judged that B is true, so the only thing we can safely replace A by in this context is by a B which is actually syntactically the same as A, that is, the kind of transparency that we always have. This is thus not even intensionally transparent.

I think that the fact that we now have a clear notion of intensional identity as opposed to the extensional one, and hence this dichotomy between intensional and extensional transparency and opacity, may be of considerable importance in clearing up the problems that have to do with opaque contexts.

So we now have reached our identity criterion, and we have reached the new definition, or the intuitionistic definition, of what a proposition is, namely that it is defined by its proof condition rather than its truth condition. We should

therefore have a look and see what these conditions look like for the standard logical operations. I already wrote up the truth conditions last time, and we now need to see how they are changed into proof conditions.

Let us begin with absurdity, $\bot$. It has no truth conditions—absurdity is true on no condition—hence it has no proof condition either: there are no conditions which stipulate how a proof of absurdity is formed.

Then let us take conjunction,

$$\frac{A \text{ true} \qquad B \text{ true}}{A \,\&\, B \text{ true}} \qquad\qquad \frac{a : A \qquad b : B}{(a, b) : A \,\&\, B}$$

The truth condition said that if A is true and if B is true, then $A \,\&\, B$ is true, so $A \,\&\, B$ is true on the condition that both A is true and B is true. The judgement A true here is treated as an abbreviated, or incomplete, form of judgement indicating that there is a proof of A. This therefore turns into the rule that if there exists a proof of A and if there exists a proof of B, then there exists a proof of $A \,\&\, B$. That is then further analyzed as follows. Let a be the proof of A that is said to exist in the first premiss, and let b be the proof of B that is said to exist in the second premiss. Then we have to get a proof of $A \,\&\, B$, so we need an operation here which takes the two proofs of the premisses into a proof of the conjunction. That operation is just to take the pair of those two. With the usual pair notation from mathematics we can write it (a, b), or we can write explicitly, as one sometimes does, $\mathrm{pair}(a, b)$.

This rule is a refinement of the truth condition, and it defines the meaning of conjunction. It is really the Heyting explanation of the meaning of conjunction, which says that a proof of the conjunction consists of a proof of A and a proof of B, or even more explicitly, is a pair whose components are a proof of A and a proof of B—only we have to be a bit more careful because we must not forget the identity criterion, but that is easy. In the case of the logical operations here, the identity criterion is always that two proofs are the same if they have the same form and their parts are the same,

$$\frac{a = c : A \qquad b = d : B}{(a, b) = (c, d) : A \,\&\, B}$$

I think we can leave those out most of the time in the following. So this is for conjunction, and these rules are simply meaning determining for conjunction.

Let us pass to disjunction,

$$\frac{A \text{ true}}{A \vee B \text{ true}} \qquad \frac{B \text{ true}}{A \vee B \text{ true}} \qquad \frac{a : A}{i(a) : A \vee B} \qquad \frac{b : B}{j(b) : A \vee B}$$

The truth conditions were that if A is true, then $A \vee B$ is true, and if B is true, then $A \vee B$ is true. Again this means that if there exists a proof of A, there should exist a proof of $A \vee B$. That is made even more explicit intuitionistically by saying that we should have an operation that takes a proof of A into a proof of $A \vee B$. Let us call that operation $i(a)$. Similarly for the second rule: we should have an operation that takes a proof of B into a proof of the disjunction. The identity criteria are the obvious ones,

$$\frac{a = c : A}{i(a) = i(c) : A \vee B} \qquad \frac{b = d : B}{j(b) = j(d) : A \vee B}$$

That is the disjunctive case, and now we come to implication,

$$\frac{(A \text{ true})}{\dfrac{B \text{ true}}{A \supset B \text{ true}}} \qquad \frac{(x : A)}{\dfrac{b : B}{(\lambda x)b : A \supset B}}$$

The standard truth condition is that $A \supset B$ is true if B is true provided that A is true. That B is true provided that A is true is now analyzed as meaning that there should exist a function in the old-fashioned sense, b, which is a proof of B depending on an arbitrary proof of A, that is, depending on a variable x that varies over the proofs of A. The analysis is thus that there exists a function which takes an arbitrary proof of A into a proof of B. Then we need an operation which given such a function gives us a proof of $A \supset B$, and we denote that operation by lambda abstraction, which we can think of as built up in two parts: first we pass from the function b here in the old-fashioned sense to the corresponding function in the modern sense, $(x)b$, and then we apply the lambda operator to that, $\lambda((x)b)$. There is also a similar identity clause here.

These are the intuitionistic meaning explanations of the sentential operators, and now only the quantifiers remain. Let us take existence first,

$$\frac{a : A \qquad B(a) \text{ true}}{(\exists x : A)B(x) \text{ true}} \qquad \frac{a : A \qquad b : B(a)}{(a, b) : (\exists x : A)B(x)}$$

The standard truth condition says that if a is an element of the quantificational domain, say A, and $B(a)$ is true—if we have some element of the quantificational domain and B is true for that element—then the existential proposition $(\exists x : A)B(x)$ is true. For the left-hand premiss, we have nothing to do, but $B(a)$ is true is analyzed as the existence of a proof of $B(a)$. From that proof and the element a we must get a proof of the existential statement. Here again one usually denotes that operation by pairing, as in the conjunctive case, and the corresponding identity rule is similar.

Then finally, the universal quantifier,

$$\frac{(x : A)}{\dfrac{B(x) \text{ true}}{(\forall x : A)B(x) \text{ true}}} \qquad \frac{(x : A)}{\dfrac{b : B(x)}{(\lambda x)b : (\forall x : A)B(x)}}$$

The truth-conditional explanation is that if for an arbitrary x in the quantificational domain, $B(x)$ is true, then $(\forall x : A)B(x)$ is true. The premiss is analyzed intuitionistically as the existence of a function b—and let us take it in the old-fashioned sense here as in the implicational case—that is a proof of $B(x)$ depending on x. From that function, we should get a proof of the universal proposition, and the operation which achieves that is usually again denoted by lambda abstraction. We use $(\lambda x)b$ as a proof of the universal proposition, and there is a corresponding identity clause.

By filling in the proof objects in the standard truth conditions, which of old are taken to be meaning determining for the logical operations, we thus arrive at Heyting's intuitionistic meaning explanations for the logical operations.

Now two more lectures remain, and I shall try, I think, to put some technical material—we have not seen any constants yet, and I have to show you at least some rules here. Even if this is a course about the philosophical aspects of type theory, some of the formal machinery has to be shown, and I will choose to concentrate on those operations which are immediately related to Fregean themes, that is, to Frege's Wertverlauf operator and to his notion of identity. I hope I will concentrate that more technical, more formulae-requiring, material into the next lecture and save the last one for some more general philosophical remarks.

Lecture 11, 9.12.93

We finished last lecture by filling in proof objects in the truth conditions for the logical operators, thereby transforming the truth conditions into proof conditions, which are exactly Heyting's meaning explanations of the logical constants.

		polymorphic notation	monomorphic notation
$\bot$	$-$	$-$	$-$
$A \,\&\, B$	(a, b)	$\&\mathrm{I}(a, b)$	$\&\mathrm{I}(A, B, a, b)$
$A \vee B$	$i(a), j(b)$	$\vee\mathrm{I}_1(a), \vee\mathrm{I}_2(b)$	$\vee\mathrm{I}_1(A, B, a),\ \vee\mathrm{I}_2(A, B, b)$
$A \supset B$	$(\lambda x)b$	$\supset\mathrm{I}((x)b)$	$\supset\mathrm{I}(A, B, (x)b)$
$(\exists x : A)B$	(a, b)	$\exists\mathrm{I}(a, b)$	$\exists\mathrm{I}(A, (x)B, a, b)$
$(\forall x : A)B$	$(\lambda x)b$	$\forall\mathrm{I}((x)b)$	$\forall\mathrm{I}(A, (x)B, (x)b)$

The logical operations are denoted in the usual way, and the notation that I introduced last time for the proof objects was that given in the second column here. It is the notation that comes most naturally to mind, at least for a mathematician, and it is close to what Heyting himself used. For the conjunction we had the pair, for the disjunction we had either $i(a)$ or $j(b)$, for the implication I used lambda abstraction, for the existential proposition we again had the pair, and for the universally quantified proposition we again had lambda abstraction.

There is a more systematic notation here which conforms to the notation used for the rules of inference in natural deduction, namely the notation used in the third column: conjunction introduction, &I, applied to the proofs a and b of the premisses; disjunction introduction—since there are two rules of disjunction introduction we have to distinguish them in some way, so $\vee\mathrm{I}_1$ and $\vee\mathrm{I}_2$—applied, in the one case, to the proof of A and, in the other case, to the proof of B; implication introduction, $\supset$I, applied to the abstraction of b with respect to x, $(x)b$, so that x is bound; existential introduction, $\exists$I, applied to the individual a and a proof of $B(a)$; and universal introduction, $\forall$I, applied to the abstraction with respect to x of b. This is a more systematic notation, but clearly it is just a change of notation.

As written here, this notation is usually called the polymorphic notation, using a terminology introduced by Robin Milner, the computer scientist. In the case of &I, polymorphic refers to the fact that I am using the same operator independently of what A and B are. One might, however, wonder, Should not &I depend also on A and B, so that it actually gets four arguments, and similarly in the other cases?

Such a notation is, in contrast, called the monomorphic notation. It reads as in the fourth column. For each pair of propositions, A and B, we have an operation which depends on A and B and which we might write as $\&\mathrm{I}_{A,B}$, or, when applied to two arguments, as $\&\mathrm{I}_{A,B}(a,b)$, but of course, writing the A and B as subscripts or writing them as arguments has the same meaning, so the systematic notation is $\&\mathrm{I}(A,B,a,b)$. Here the dependence on the propositions is made explicit, and similarly in the other cases: Disjunction introduction depends on the two propositions A and B, and we apply it, in the one case, to the proof of A, yielding $\vee\mathrm{I}_1(A,B,a)$, and, in the other case, to the proof of B, yielding $\vee\mathrm{I}_2(A,B,b)$. Implication introduction depends on the two propositions A and B, and it is applied to the proof b of the premiss, binding the variable x. Existential introduction depends on the domain A, to begin with, and secondly on the propositional function that we have over that domain, and that propositional function is the abstraction of B with respect to x, $(x)B$, and then we have the two arguments a and b, the individual and the proof of the premiss. Universal introduction also depends on the domain A and on the propositional function, $(x)B$, that is quantified, and then the argument is $(x)b$, as we already had before. This is the full monomorphic notation, and morph here is of course for form.

If we use the polymorphic notation, then $\&\mathrm{I}(a,b)$ will be a proof of $A \,\&\, B$, for instance, but we may get syntactically the same proofs of different propositions, and that is excluded with the monomorphic notation. How much type information to include in the notation has been a design problem, but we now have a very clear guiding principle for that, namely that so much type information must be included that type checking can be effectively performed, which is to say that we can check mechanically the well-formedness of the expressions in the language. It turns out that for type checking, the monomorphic notation is in general necessary. At least to start with, the best thing is therefore to put in all possible type information, and then one can investigate if one can do without some of it. ([B.G.S:] "I wonder about the (a,b) notation: would it not be advantageous to use the hooked brackets, so as to avoid, for instance, that the same thing occurs within the round brackets and the hooked ones?" Yes, it has become popular to use the hooked brackets for pairs, but I am sufficiently old to have been brought up with the round brackets.)

This is the formal notation that is used for the proof objects in type theory. The only previous notation available for proofs considered as objects was the notation for proofs introduced by Gentzen in his system of natural deduction. Heyting introduced proof objects in his meaning explanations for the logical operators, but he did not introduce proof objects into the formal systems, that is, into the system of propositional and predicate logic. They were just informally part of the semantical explanations.

I thought I would show you, at least for the introduction rules, how the Gentzen notation, which I am sure you have seen, corresponds to the type-theoretic linear notation. Gentzen's notation is a notation in tree form. The rule of conjunction

introduction, for instance, looks as follows:

$$\dfrac{\begin{array}{cc}\vdots\ a & \vdots\ b\\ A & B\end{array}}{A\ \&\ B}\ \&\mathrm{I} \qquad\qquad \&\mathrm{I}(A,B,a,b)$$

We have a proof of A and a proof of B—here now is a notation for proofs, a rather scanty notation, but at least it is sufficiently workable to begin with, even if not for all purposes—and then we apply conjunction introduction and conclude $A\ \&\ B$. In the linear notation of type theory, this is written as $\&\mathrm{I}(A,B,a,b)$, and the correspondence between the tree-like notation and the linear notation of type theory is that the a here is the proof of A, and b, similarly, is the proof of B. This thus shows the exact correspondence between the tree-like notation and the linear notation.

For disjunction introduction, we have

$$\dfrac{\begin{array}{c}\vdots\ a\\ A\end{array}}{A\vee B}\ \vee\mathrm{I}_1 \qquad \dfrac{\begin{array}{c}\vdots\ b\\ B\end{array}}{A\vee B}\ \vee\mathrm{I}_2 \qquad\qquad \vee\mathrm{I}_1(A,B,a),\ \vee\mathrm{I}_2(A,B,b)$$

If we have a proof of A, we may apply the rule $\vee\mathrm{I}_1$ to get a proof of $A\vee B$, and similarly from a proof of B we get a proof of $A\vee B$. The notation that I just introduced was $\vee\mathrm{I}_1(A,B,a)$, so a corresponds to the proof of A, and in $\vee\mathrm{I}_2(A,B,b)$, b corresponds to the tree-like proof of B that we have in Gentzen's notation.

Similarly in the case of implication,

$$\dfrac{\begin{array}{c}\overset{x}{(A)}\\ \vdots\ b\\ B\end{array}}{A\supset B}\ \supset\mathrm{I} \qquad\qquad \supset\mathrm{I}(A,B,(x)b)$$

By implication introduction, we conclude $A\supset B$, having a proof of B from the assumption A, discharging this assumption A. In the linear notation we have $\supset\mathrm{I}$ depending on the propositions A and B and applied to the hypothetical proof of the premiss B from the assumption A, which corresponds to the tree-like configuration in Gentzen's notation. Moreover, we discharge zero, one or two or more occurrences of the assumption A at this inference, and it has to be indicated then at this inference exactly what assumptions are discharged. That index, usually taken to be a number or something, corresponds to the variable which is being bound in b, so if we use x as a label for the assumptions that are discharged, then the correspondence is as here.

There now only remain the two quantifier rules.

$$\dfrac{\begin{array}{cc}a:A & \vdots\ b\\ & B(a/x)\end{array}}{(\exists x:A)B}\ \exists\mathrm{I} \qquad\qquad \exists\mathrm{I}(A,(x)B,a,b)$$

If we have an individual a and a proof of $B(a/x)$, and we conclude $(\exists x:A)B$ by existential introduction, then that corresponds to $\exists\mathrm{I}$ applied to A, to the propositional function $(x)B$ that is being quantified, to the individual a, and to the proof

b of the instance $B(a/x)$. The a is already explicit in the natural-deduction rule as I have written it—observe that a is written linearly even in predicate logic, not as a tree, whereas the proof of $B(a/x)$ is written in tree-like form, and it is that tree which is linearly encoded into b.

Finally, the universal case,

$$\dfrac{\begin{array}{c}(x:A)\\ \vdots\ b\\ B\end{array}}{(\forall x:A)B}\ \forall\mathrm{I} \qquad\qquad \forall\mathrm{I}(A,(x)B,(x)b)$$

From the assumption that x is an arbitrary element of A, we prove B to be true, and then we conclude $(\forall x:A)B$, by universal introduction, discharging this assumption, which is to say, binding the variable x in this proof. In the linear notation, we have $\forall$I depending on the domain A, the propositional function $(x)B$, and then we have the linear encoding of this tree-like proof, let us call it b, and now we have to indicate which variable is being bound by this inference, and in this case it is x, as indicated in the abstraction here.

What I have shown to you here is a translation that actually goes in both directions: from Gentzen's tree-like notation into the linear notation, but you can also pass in the other direction, so it is simply a notational isomorphism between the two sides. Once you have seen this, it is clear that Heyting's meaning explanations of the logical constants—in accordance with the principle that a proposition is defined by explaining how its proofs, more precisely canonical or direct proofs, are formed—as an explanation of what a proposition is in general and of what the particular propositions are that we form by means of the logical operations is in complete agreement with Gentzen's formulation. Gentzen, in the same paper where he introduced the calculus of natural deduction, said in one sentence that—and I have to take this from memory—"The meaning of a logical constant is, so to speak, determined by its introduction rules." Once you have seen the correspondence here between the introduction rules and these operators, which are the possible forms that a canonical proof, or a primitive proof, of the proposition in question can take, you see that there is complete agreement between this explanation of Gentzen's and Heyting's explanation.

As I have said, for Heyting the proofs entered into the meaning explanations, but not into the formal systems, and it is this which is the novelty of type theory, or one of the novelties of type theory: to make the proof objects part of the symbolism. They should be there, of course. If they are necessary in the meaning explanations, we should introduce a precise notation for them and import them into the formal system itself. Gentzen's system of natural deduction, especially as it was developed by Prawitz, who studied normalization of proofs in Gentzen's system of natural deduction, may be regarded as an intermediate step. There, at least some notation—although a rather cumbersome notation—is introduced for proofs, actually the first one, and then that is linearized in type theory, which means that proof objects are treated on a par with all other objects. In the case of the existential quantifier, it is clear that a is an object, namely an object of the individual domain.

In the minor premiss, on the other hand, you have a proof object, so you see that the notation for these objects is very incongruous. Since the proof that you get for the existential statement has to contain both as parts, it is very natural that you have somehow to harmonize the notations for them, and that is what is done in type theory.

Since we have an isomorphism here between the two sides, one might have gone in the opposite direction, introducing a tree-like notation for everything rather than a linear notation. Since it is good to have different notations to experiment with, I thought I would briefly show what would happen if we systematically used the tree-like notation that we have in natural deduction. Then we would have to write our individual terms also as trees. In particular, if our domain is the domain of natural numbers, we would have to write the numerical terms as trees,

$$\frac{}{\mathbf{N}}\,0 \qquad \frac{\mathbf{N}}{\mathbf{N}}\,\mathbf{s} \qquad \frac{\mathbf{N}\qquad\mathbf{N}}{\mathbf{N}}\,+ \qquad \frac{\mathbf{N}\qquad\mathbf{N}}{\mathbf{N}}\,\cdot \qquad \frac{\mathbf{N}}{\mathbf{N}}\,!$$

Instead of writing 0, which is a natural number, like that, we would have to treat it as a rule of inference by which we may conclude $\mathbf{N}$ from no premisses, since 0 has no arguments. The rule is thus called 0. The successor operation will be an operation that takes natural numbers into natural numbers, so it looks like a rule of inference from $\mathbf{N}$ to $\mathbf{N}$, and that rule of inference is called $\mathbf{s}$. Plus is a binary operation which takes a natural number and another natural number into a natural number, and + is the name of that operation, and similarly for times we will have a rule which looks exactly the same except that it is called $\cdot$. The factorial function, as a final example, is a unary function taking numbers into numbers, and it is called !, which we write as a logical rule as above.

If we take a complex numerical term which we can easily read, like $5! + 2$, then how does it look? We have a plus as the outermost operator, then we have a 2 with no premisses, no further parts, and here we have 5!, which has been obtained by the factorial rule from 5,

$$5! + 2 \qquad \sim \qquad \frac{\dfrac{\dfrac{}{\mathbf{N}}\,5}{\mathbf{N}}\,! \qquad \dfrac{}{\mathbf{N}}\,2}{\mathbf{N}}\,+$$

([B.G.S.:] "How would you compute that to a tower of hundred... it would be a tower of 122 $\mathbf{N}$'s on top of each other!" Oh, yes. [Laughter.]) So when we make the definition, the nominal definition that I have made, $1 = \mathbf{s}(0)$, then that would take the following form,

$$1 = \mathbf{s}(0) \qquad \sim \qquad \frac{}{\mathbf{N}}\,1 = \frac{\dfrac{}{\mathbf{N}}\,0}{\mathbf{N}}\,\mathbf{s}$$

When we define $2 = \mathbf{s}(1)$, similarly, we get

$$2 = \mathbf{s}(1) \qquad \sim \qquad \frac{}{\mathbf{N}}\,2 = \frac{\dfrac{}{\mathbf{N}}\,1}{\mathbf{N}}\,\mathbf{s}$$

We thus have a complete isomorphism here. For a computer scientist, this is very natural, of course. He primarily thinks of an expression as it is stored in the

computer memory, namely as a tree structure and not in a linear notation. If you think of it that way, you have, so to say, a more abstract way of looking at the notation, and from that point of view there is no difference at all here between the two sides. But what is absolutely essential here is the uniformization that I just talked about, namely that we treat the elements of our individual domains—of which there are many in type theory—in the same way as we treat the proof objects. We treat them on a par with the proof objects, which is to say that we should have the same kind of notation, either linear or tree-like, for the two. We should be able to deal with them in the same way, because it is this fact, that we introduce proof objects into the formal system, that makes it possible to strengthen the elimination rules for the logical operators in the way it is done in type theory.

It is remarkable by itself that we have—Brouwer's criticism of the classical laws was most remarkable, of course. It simply challenged some of the classical laws and said that we ought to change those laws. In the step into type theory, however, we have, not a change in some already available laws—the intuitionistic laws that were already there remain valid, they are derivable in type theory—but what is remarkable is that those laws are strengthened. Not the introduction rules, but the elimination rules are strengthened, and that strengthening is possible precisely because the proof objects have been imported into the system, that is, made part of the system itself.

Maybe I should show how that strengthening is possible in one case, namely, for the existential quantifier. The usual rule of existential-quantifier elimination says this:

$$\frac{(\exists x : A)B(x) \text{ true} \qquad \begin{array}{c}(x : A, B(x) \text{ true}) \\ C \text{ true}\end{array}}{C \text{ true}}\ \exists\text{E}$$

If we have a proof of the existential proposition, and we have another proposition, C, which we prove to be true under the assumption that x varies over the domain A and $B(x)$ is true, then we may conclude that C is true, discharging those two assumptions. This is the usual rule of existential elimination. Now suppose that we fill in the proof objects,

$$\frac{c : (\exists x : A)B(x) \qquad \begin{array}{c}(x : A, y : B(x)) \\ d : C\end{array}}{\exists\text{E}(A, B, C, (x, y)d, c) : C}$$

That we have a proof of the existential proposition means that we have a proof object, c say, of it. That we have a proof of C from the two assumptions $x : A$ and $B(x)$ true means that we have a proof, d say, of C, which depends on x and depends on an assumed proof, y say, of $B(x)$. Then we get a proof of C here, which in the monomorphic notation is $\exists$E applied to A, B, and C, and then we have d, whose two variables, x and y, become bound, and then we have finally c. This becomes our proof of C.

So far I have done nothing but to give you another example of the passage from the tree-like notation in natural deduction to the linear notation of type theory, but

the fact that the proof object, c here, is made explicit makes it immediately possible to strengthen this rule by assuming, not as we have done here, that C is an arbitrary proposition—which is the usual assumption, of course—but we can take C to be a proposition that depends on an arbitrary proof of the existential proposition. Instead of taking just the proposition, we take a propositional function over the proofs of the major premiss. Writing $\exists(A, B)$ for the existential proposition, we therefore get $C : (\exists(A, B))\text{prop}$. We thus replace C by a propositional function, and that will include the case when C is a constant, because then we take this propositional function to be the one which is constantly equal to the old C.

What changes do we have to make then in the rule of existential-quantifier elimination? Well, we look upon C now as a property of proofs of the existential proposition. In the conclusion we get that that property holds for the particular proof c, and we conclude that from knowing that C holds for an arbitrary proof of that proposition of introduction form, or canonical form. In the simple notation that I first introduced, the canonical form is just the pair (x, y), but if I write it out in the full monomorphic notation, it is $\exists\text{I}(A, B, x, y)$,

$$\frac{\begin{array}{cc} & (x : A, y : B(x)) \\ c : (\exists x : A)B(x) & d : C(\exists\text{I}(A, B, x, y)) \end{array}}{\exists\text{E}(A, B, C, (x, y)d, c) : C(c)}$$

This strengthened rule is of course valid. The second premiss here, the minor premiss, says that C holds of an arbitrary proof of the form $\exists\text{I}(A, B, x, y)$, where x and y are arbitrary. By the meaning explanations, however, a proof—canonical proof more precisely—of an existential proposition has the form $\exists\text{I}(A, B, a, b)$ for some a and b, and since we know this for arbitrary x and y, we can choose x to be a and y to be b, hence C must hold for an arbitrary proof c here.

This is one example showing how the usual elimination rule can be strengthened when you pass form ordinary predicate logic into type theory, and this strengthening can be done for all elimination rules, including the universal quantifier and the implication. Again, this is not a rejection of previous rules, but a strengthening of them. Moreover, this is not only of interest as a theoretical possibility, so to say, but it immediately makes it possible to prove the axiom of choice: once you have strengthened the existential elimination rule in this way, you can prove the axiom of choice. I will not give that proof—it is given in my little book, for instance—but I would like to comment that it was stated by Bishop in his book from 1967 that the axiom of choice is clearly valid on the intuitionistic interpretation of the logical operations.

The formulation of the axiom of choice that I have in mind now is the following one,

$$(\forall x : A)(\exists y : B(x))C(x, y) \supset (\exists f : (\Pi x : A)B(x))(\forall x : A)C(x, \text{app}(f, x))$$

If for all x in a set A, there exists a y in a set B, which may even depend on x, such that some relation C holds between x and y, then there exists a function, that is, an element of the Cartesian product $(\Pi x : A)B(x)$—I will come to the

Cartesian product in a moment here—such that for all x in A, C holds between x and the value of this function for the argument x, for which I use an operator, app—which will come in a moment—for application of f to x. (Here I have used the polymorphic notation. In the full monomorphic notation we have to show the dependency on A and B also, so it becomes $\mathrm{app}(A, B, f, x)$.)

Bishop noted, and maybe it was noted already by Brouwer or Heyting, that the axiom of choice is clearly valid on the intuitionistic interpretation of the logical operations. Howard, in 1969, then observed that it can be derived in a formal system of terms which is on the way to type theory, so to say: a system that forms an intermediate step on the way from the standard predicate calculus to type theory.

The axiom of choice, as you know, was introduced in the very beginning of the century by Zermelo, and curious things were soon proved with it, so curious that there arose a lively discussion, even among mathematicians—Borel, Lebesgue, Baire and Hadamard, in particular—whether this axiom was valid. Could we use an axiom that leads to such strange consequences? The strange consequences were all of the sort that you—well, it was realized that by means of this axiom you could easily construct functions that could not be computed, so there was the question, were they really good functions then, if you could not compute them? In that discussion, it was universally thought that it was the axiom of choice that gave rise to these uncomputable functions, and it was only Brouwer who realized, in 1908, that the problem is really with classical logic. It is the law of excluded middle in combination with the axiom of choice that give rise to these strange uncomputable functions. Moreover, the law of excluded middle itself, alone, is sufficient to give such uncomputable functions, as we have seen in my discussion of the Boolean interpretation of the notion of proposition: certain propositional functions become uncomputable as soon as we allow quantification over infinite domains. Brouwer thus understood what was really the source of the problems, namely classical logic.

Interestingly enough, once classical logic is replaced by intuitionistic logic, then the axiom of choice is valid, it is simply provable, which is very fortunate, since it is a most important principle used in many, many places in the development of mathematics. This is then the first solid justification that we have of the axiom of choice, the proof that we have of it in type theory from the strengthened elimination rule for existence.

It should already be clear here that the difference between the domains that we quantify over, that is, what I call sets and propositions, is not so big any longer, because we have introduced a uniform notation that works as well for elements of domains as it does for proofs of propositions. Moreover, we have seen the following rules:

$$\begin{array}{c|c} \mathrm{set} : \mathrm{type} & \mathrm{prop} : \mathrm{type} \\ \dfrac{A : \mathrm{set}}{A : \mathrm{type}} & \dfrac{A : \mathrm{prop}}{A : \mathrm{type}} \\ \dfrac{A = B : \mathrm{set}}{A = B : \mathrm{type}} & \dfrac{A = B : \mathrm{prop}}{A = B : \mathrm{type}} \end{array}$$

For sets we had, first of all, the axiom that set is a type, and then the rules that if A is a set, then A is a type and, moreover, if we have equal, or identical, sets, then they are also identical as types. On the other hand, we had the axiom that propositions form a type and the rules that each proposition gives rise to a type and that identical propositions give rise to identical types. (This last rule was the clue to the identity criterion of the category of propositions.) Sets and propositions thus begin to look as if they behave in a very similar way, and you may ask whether maybe a further unification is possible here. I think I will just finish before the break by writing up the following table and then comment upon it after the break.

logic	set theory
A is a prop	A is a set
a is a proof of A	a is an element of A
A is true	A is nonempty (inhabited)
$\bot$	$\emptyset$
$A \,\&\, B$	$A \times B$
$A \vee B$	$A + B$
$A \supset B$	$B^A = A \to B$
$(\exists x \in A)B(x)$	$\sum_{x \in A} B_x = (\Sigma x \in A)B(x)$
$(\forall x \in A)B(x)$	$\prod_{x \in A} B_x = (\Pi x \in A)B(x)$

We let the judgement that A is a proposition correspond to the judgement that A is a set, and instead of saying that a is a proof of the proposition A, we say that a is an element of the set A. The judgement A is true is compared with the judgement that the set A is non-empty, or inhabited, as the intuitionists prefer to say in order to stress that it is a positive notion. We compare absurdity with the empty set, the conjunction of two propositions with the Cartesian product of two sets, the disjunction of two propositions with the disjoint union of two sets, the implication of two propositions, A and B, with the set of functions from A to B, what mathematicians ordinarily like to write as B^A, or a more common notation is $A \to B$. For the quantifiers, $(\exists x \in A)B(x)$ shall be compared with the disjoint union of a family of sets, which a mathematician normally writes as $\sum_{x \in A} B_x$, using the old-fashioned notation where the dependence is usually written as an index, and if we linearize that notation we get $(\Sigma x \in A)B(x)$. Finally, the universal quantifier, $(\forall x \in A)B(x)$ should be compared to the Cartesian product of a family of sets, $\prod_{x \in A} B_x$, and again linearizing the notation, $(\Pi x \in A)B(x)$.

These are well-known set-theoretical constructs, and this table could be made longer—though I will not do so here—by listing proofs of propositions on the left and the corresponding set-theoretical constructs on the right. I have already done that to some extent earlier, so let me finish here.

You may understand this table in different ways. One way is to understand it as an interpretation of logical notions by means of set-theoretical notions. If

you look at the correspondences for the logical operations, they fit exactly with Heyting's meaning explanations. You may say that Heyting's meaning explanation for conjunction, for instance, says that the set of proofs of $A \,\&\, B$ is the Cartesian product of the set of proofs of A and the set of proofs of B, that the set of proofs of an implication is the set of functions which take a proof of A into a proof of B, and similarly in all other cases. If you look upon it that way, then you come to see Heyting's meaning explanations, or Heyting's interpretation of propositional and predicate logic, as a set-theoretical interpretation of it.

This set-theoretical interpretation of logic validates all the rules, and that somehow goes completely counter to logicism, if logicism was the idea of reducing all mathematical notions, in particular the notion of natural number, to purely logical notions. I have already said that to attempt such a reduction is futile, since we need at least one set to quantify over, hence we cannot do wholly without set-theoretical notions, but if you view this table in the way that I have just suggested, then it, so to say, goes entirely counter to logicism: now we have a set-theoretical reduction of all the logical laws, so that, really, set theory becomes more basic than logic.

I prefer, however, to view the table differently. I prefer to view it rather as a complete isomorphism between the two sides, which is to say that there is no fundamental difference between the notion of proposition and the notion of set, no fundamental difference between the notion of proof of a proposition and that of an element of a set, no fundamental difference between truth of a proposition and inhabitedness of a set, and no fundamental difference between the logical operations and the set-theoretical operations.

The proposal is simply to say that proposition and set are basically one and the same notion. How are you to justify such a claim? Well, you have to look very carefully at how the question, What is a proposition?, is answered, and at how the question, What is a set?, is answered, to see whether the answers are the same. If they are the same, then propositions and sets indeed are the same—not entirely the same, of course, but the same except for wording. Surely, there is a difference between the term proposition and the term set, but the question is whether that is just a difference in nomenclature rather than in the meaning.

Let us therefore go back to our basic meaning explanations. How did we answer the question, What is a set? We said that a set was defined by explaining how its elements, more precisely its canonical elements, are formed, as well as how identical canonical elements are formed—you will remember the examples I gave of natural numbers and Cartesian product, for instance. When we came to analyze the notion of proposition, after rejecting the Boolean notion and passing via the explanation of a proposition as defined by its truth conditions, we ended up with Heyting's explanation, which, if one formulates it carefully, becomes: a proposition is defined by laying down how its proofs, more precisely its canonical proofs, are formed, as well as how identical canonical proofs are formed. It is those conditions, which is what we call the introduction rules, which determine the proposition in question.

You see that these explanations are totally isomorphic if you replace the word proposition by the word set and the word proof of a proposition by the term element

of a set, so it seems that we really have to do with two notions that—so to say on a deep level, not on the surface level of how we express ourselves, but on the deep level of what we mean by those terms—seem to be fused into one and the same notion. That is why I have shown all these little bits and pieces of the similarity between rules that had to do with sets and rules that had to do with propositions. I even showed you how to write numbers as if they were proofs in order to stress the similarity between the two notions.

This is therefore how I prefer to look at the above table: rather than simply seeing it as a set-theoretical reduction of logic running completely counter to logicism, I look upon it as a fusion of set theory and logic that has been accomplished by means of, or via, the intuitionistic interpretation of logic. This, to my mind, is something quite remarkable if you think of the origins of, on the one hand, modern logic and, on the other hand, set theory. Of course, logic has behind it a long history. The propositional part of modern logic was there already with the Stoics essentially, and it was intensively studied and taught during the medieval period. It does, however, not really become logic as we think of it now until the quantifiers were introduced by Frege around 1880. At precisely the same time, set theory was developed by Cantor, starting, I believe, from some hints of Bolzano's. I have not studied Bolzano's writings on the notion of set, but I think it is generally agreed that set theory to an unusually great extent was the creation of a single person, namely Cantor, in the 1880s. At one and the same time, therefore, we have modern logic being developed by Frege and set theory being developed in an essentially logic-free way by Cantor—he did not rely upon any pre-existing logic or anything of that sort, but just went about it directly. So we have these two quite independent creations at approximately the same time, and in the present light, they have simply been fused into one conceptual framework by means of the isomorphism that I have spelled out in this table.

Let me then say something about the history of this isomorphism which we now usually refer to as the Curry–Howard isomorphism. I think that the real source, I mean, the most fundamental source of it, is Heyting's meaning explanations for intuitionistic logic, so it is Heyting 1931. As I have said, Heyting's meaning explanations can be phrased simply by saying that the proofs of the conjunction, for instance, form the Cartesian product of the proofs of A and the proofs of B, and similarly in the other cases. There you already have a correspondence between the logical operations and the set-theoretic operations, but surely it took a long time until it crystallized into the form that I have presented here. The very idea of treating proofs as mathematical objects Heyting had from Brouwer, and Curry, already in 1934, labelled the axioms of the positive implicational calculus in a way that shows that he had seen this correspondence to some extent at least.

The axioms of the positive implicational calculus, that is, the intuitionistic implicational calculus, are axioms such as these:

(PI) $A \supset A$
(PK) $A \supset (B \supset A)$
(PS) $(A \supset (B \supset C)) \supset ((A \supset B) \supset (A \supset C))$

The labels are from Curry, and the reason for them is that P is Curry's notation for implication, and I,K and S are his combinators, defined by

$$\mathrm{I}x = x, \;\; \mathrm{K}xy = x, \;\; \mathrm{S}xyz = xz(yz)$$

Curry had thus clearly seen a correspondence here: You may look upon the combinator I as the proof of the implication $A \supset A$ which takes a proof of A into itself. You may look upon K as the proof of $A \supset (B \supset A)$ which takes a proof x of A and a proof y of B into x. And you may look upon S as the operation which takes a proof x of $(A \supset (B \supset C))$, a proof y of $A \supset B$ and a proof z of A into the proof of C that you get by first applying x to z and then applying xz to the proof yz that you get by applying y to z. So Curry had clearly seen this correspondence already in the 1930s, and in his book from 1958 it is spelled out quite clearly in the case of the positive implicational calculus.

Then we have the paper by Howard that I mentioned, from 1969. It was a set of notes that were widely circulated and finally published in 1980, in the Festschrift to Curry, where it was taken quite a step further. Howard of course knew Curry and was inspired by him, so these seem to me to be the main references.

Here in Holland you have De Bruijn, who, independently of Howard, in the first papers on AUTOMATH from 1968, had part of this, but not the full correspondence between logic and set theory that I have shown you. De Bruijn noticed that in formalizing mathematics, you make assumptions of two kinds: one is when you say, Let x be an arbitrary element of the set A, another is when you assume some proposition B to be true. De Bruijn noticed that it was very natural to think of those two different kinds of assumptions as the same, because assuming that B is true may be interpreted as assuming that we have an arbitrary proof of B. De Bruijn thus introduced proof variables in that way, and of course he knew Heyting, so surely, as I have said earlier, it is Heyting's introduction of the proof object that is the real source.

If you turn the assumption that B is true into an assumption that we have an arbitrary proof of B, and we prove something later on, C true, say, then that also has to be interpreted as our having a proof, or having constructed a proof, of C. We can thus view a proof that a proposition is true as an implicit construction of a proof object of the proposition in question. In this way you moreover get a functional notation for proofs, which is essentially of the kind that I have shown you. This is thus the degree to which, as far as I know, De Bruijn had the Curry–Howard correspondence, but, as I have said, not the complete correspondence between logic and set theory.

There are also other things that one could mention from approximately the same time, but I think this will be enough for us. This is therefore all I have to say about the Curry–Howard isomorphism, which has led to the fusion of logic and set theory in intuitionism.

Now it remains for me to give some constants of type theory. I certainly will not be able to give all the rules of type theory, but I at least want to give the rules associated with the Cartesian product. In the Curry–Howard correspondence,

that corresponds to the universal quantifier and includes also, as special cases, the function set construct, on the one hand, and the implication, on the other hand, on the logical side. I also want to give the rules for identity, since I have talked so much about the different notions of identity—intensional identity and extensional identity, for instance—and now we have a further identity here, the one that corresponds to the identity you have in predicate logic when we speak of predicate logic with identity. That will give me occasion to comment on the correspondence between type theory and Frege's system on certain points.

Here comes now the first constant, Π, which is the Cartesian product operator. With the type structure that I have dealt with at such length, the axiom for Π can be simply written as the typing of Π, which is to say, I just say that it is an operation which takes a set X and a family of sets over X into a set,

$$\text{(Π-form)} \qquad \Pi : (X : \text{set})(Y : (X)\text{set})\text{set}$$

If we think of Π as the universal quantifier, then we would have prop instead of set here, so Y would be a propositional function over the set X, and we would get a proposition out, but we have now already performed the identification.

Here is thus an axiom, and it needs to be explained, as all axioms do. We therefore have to remember what it means. According to the general meaning explanation, this axiom is tantamount to the validity of the following rules:

$$\frac{A : \text{set} \qquad B : (A)\text{set}}{\Pi(A, B) : \text{set}} \qquad\qquad \frac{A = C : \text{set} \qquad B = D : (A)\text{set}}{\Pi(A, B) = \Pi(C, D) : \text{set}}$$

If A is a set, and B is a family of sets over A, then $\Pi(A, B)$ is a set, and moreover, we always have the identity criterion, which I will soon forget about, but at least here I think you should remember it: if A and C are the same set, and B and D are the same families of sets over A, then we must also check that $\Pi(A, B)$ is the same set as $\Pi(C, D)$. By the general meaning explanations that I have given long ago, the axiom (Π-form) simply means that we have the right to infer by these rules. We must therefore see to it that both of these rules are valid.

Let us begin with the first one. Assume the premisses that A is a set and that B is a family of sets over A. I have to explain to you what set $\Pi(A, B)$ is. How do I define a set? Well, I define a set by explaining how its canonical elements are formed. In this case, how are they formed? Well, take—in addition to the premisses that we already have—a function, b say, from A with values in $B(x)$, then the introduction rule for the Π-operator states that, and now I am using the monomorphic notation, $\lambda(A, B, b)$ is an element of $\Pi(A, B)$,

$$\text{(Π-intro)} \qquad \frac{A : \text{set} \qquad B : (A)\text{set} \qquad b : (x : A)B(x)}{\lambda(A, B, b) : \Pi(A, B)}$$

There is a corresponding equality rule, of course, which maybe I should write out in this case,

$$\frac{A = C : \text{set} \qquad B = D : (A)\text{set} \qquad b = d : (x : A)B(x)}{\lambda(A, B, b) = \lambda(C, D, d) : \Pi(A, B)}$$

If A is identical to C, and B to D, and b is identical to d, then $\lambda(A, B, b)$ is identical to $\lambda(C, D, d)$, which are both in $\Pi(A, B)$.

We may say that these two introduction rules simply define the set $\Pi(A, B)$, because they say how the canonical elements are formed and how equal canonical elements are formed. At the level of precision with which one normally works, this would be enough to say, but there is one point here which I will mention this time and never again, I think, and that is that we have to satisfy also the second rule here, namely that if A and C are identical, and B and D are identical, then $\Pi(A, B)$ and $\Pi(C, D)$ should be the same. The explanation here cannot be strictly correct, because if these are to be the same, then $\lambda(A, B, b)$ must also be a canonical element of $\Pi(C, D)$, because of how identity between sets is explained. The meaning explanation as I gave it is thus not strictly correct. Instead, you must say that a canonical element of $\Pi(C, D)$ has the form $\lambda(A, B, b)$, where A is identical to C, B is identical to D, and b is a function of type $(x : A)B(x)$. Likewise you must adjust the explanation of how equal canonical elements are formed.

That fixes the meaning explanation of the Cartesian-product operator and, moreover, we now have the λ-operator, which allows us to form elements of the Cartesian product. The λ-operator is the counterpart in type theory to Frege's Wertverlauf operator. If we pass now, for the moment, to the polymorphic notation and leave out the two first arguments to λ, then a canonical element of the Cartesian product looks like $\lambda(b)$. By the η-rule, this is the same as $\lambda((x)b(x))$, or in a more usual notation, $(\lambda x)b(x)$, and this is now what corresponds to Frege's x with spiritus lenis on top of it followed by $b(x)$,

$$\lambda(b) = \lambda((x)b(x)) = (\lambda x)b(x) = \grave{x}b(x) : \Pi(A, B)$$

Frege did not normally use x and b here—he always used small Greek letters in the Wertverläufe and instead of b he used Φ or Ψ or something, $\grave{\varepsilon}\Phi(\varepsilon)$—but that is only a notational difference.

What then is the difference here? The difference is that Frege put everything into his universal domain of all objects, and I have already argued at length why it is necessary to cut it up. We have to have one domain for the truth values, that was my set bool, another domain for the natural numbers, which was the set $\mathbf{N}$ of natural numbers, and then we will have many other domains as well. When we come to the Wertverläufe, then the function b that we take the course of values of will have a domain, A say, which it is defined over, and it will take values in some other domain, B, which may even in general depend on x. Since there is no universal domain, whenever we have an element, it has to be an element of some domain. In particular, the arguments of b must belong to some domain and the values to some other domain, and where are we going to put this Wertverlauf, $(\lambda x)b(x)$? Well, we simply have to create a new domain to have somewhere to put it. We do not have a universal domain that we can put it in, so we create a new domain, and that is precisely $\Pi(A, B)$ here. That is a new domain, a new set, and its elements are precisely the Wertverläufe of functions of the type that I mentioned.

We also have an eliminatory operation, dual to the introductory operation, and that eliminatory operation, in full generality, looks as follows—I have introduced two constants so far, Π and λ, and now comes the third one. I should also say that, with the systematic notation that I have been using before, λ is the introductory operator corresponding to Π, so $\lambda = \Pi\mathrm{I}$, sometimes also written Πintr. Now, then, comes the corresponding eliminatory operator, $\Pi\mathrm{E}$, or Πelim, which is typed as follows:

$$\Pi\mathrm{E} : (X : \mathrm{set})(Y : (X)\mathrm{set})(Z : (\Pi(X, Y))\mathrm{set})$$
$$(w : (y : (x : A)B(x))Z(\lambda(X, Y, y)))(z : \Pi(X, Y))Z(z)$$

In the monomorphic notation we have to show all the arguments: the function $\Pi\mathrm{E}$ takes a set X, a family of sets Y over X, and a family Z which, because of the strengthening of the elimination rules that I have talked about, will in general depend upon a proof, in this case, of $\Pi(X, Y)$, so Z is a family sets over $\Pi(X, Y)$, and then the next argument is a function, let us call it w, which takes a function from A into B, where B may depend on the argument, and finally it has as last argument the argument that fits for Z, so z in $\Pi(X, Y)$, and what we get out now is something of type Z applied to z.

I am well aware of the impossibility of handling a notation like this if you have not trained yourself in it, but this is inevitable: there is absolutely no other way to learn a new notation than by using it, sitting down and using it. I know this particularly with respect to type theory, because even excellent mathematicians just cannot read the notation if they are not used to it, and they complain about this mystical notation that they have not seen before, but it is of course no more mystical than any other notation, for instance the notation in the *Grundgesetze* or even the notation in predicate logic. They are also things that one has to learn at some stage, and of course the type-theoretical notation can be learned just as well, but I am fully aware of the fact that it is not possible to take it in if you have not had some training.

What makes this notation somehow readable and manageable is that the type-forming operation may be thought of as a kind of universal quantifier. You may thus think of $(x : \alpha)\beta$ somehow as for all α β, and you think of the special case when β does not depend on x, $(\alpha)\beta$, approximately as α implies β. Since we have training in handling the universal quantifier and implication, we can, so to say, transfer that to this level. What you read here then is that Z is a property of proofs that holds for all proofs of the form $\lambda(X, Y, y)$, but this is the form of a canonical proof, hence Z must hold for an arbitrary proof of this proposition.

This is the Π-elimination operator written out as a constant. If we write it out as a rule instead, it is somewhat more human,

$$(\Pi\text{-elim}) \qquad \frac{\begin{array}{ccc} A : \mathrm{set} & B : (A)\mathrm{set} & C : (\Pi(A, B))\mathrm{set} \\ d : (y : (x : A)B(x))C(\lambda(A, B, y)) & & c : \Pi(A, B) \end{array}}{\begin{array}{c} \Pi\mathrm{E}(A, B, C, d, c) : C(c) \\ \Pi\mathrm{E}(A, B, C, d, \lambda(A, B, b)) = d(b) : C(\lambda(A, B, b)) \end{array}}$$

Here we have a new rule, and there is a corresponding equality rule for it, which it is now definitely not necessary to exhibit, but we have to justify this rule. So assume the premisses, and I have to tell you what element of the set $C(c)$ $\Pi\mathrm{E}(A, B, C, d, c)$ is. The explanation is given by the corresponding definitional rule, which tells us how $\Pi\mathrm{E}(A, B, C, d, c)$ is evaluated when c has introductory form, or canonical form, which in this case is $\lambda(A, B, b)$. When the argument has that canonical form, then we can evaluate this and get as result $d(b)$, which is an element of $C(\lambda(A, B, b))$.

Here we have a new nominal definition. It is nominal because you have a definiens and a corresponding definiendum, but it is not explicit any longer, because you are looking at the form of the last argument to $\Pi\mathrm{E}$. It is the first example, I believe, of a nominal definition that is not explicit, and then there will be recursive definitions and transfinite recursive definitions and so on, many more definitions than just the explicit ones.

Next time I will begin by showing how this eliminatory operator contains the operation app, which I used in the formulation of the axiom of choice and which monomorphically looks like $\mathrm{app}(A, B, b, a)$ and polymorphically like $\mathrm{app}(b, a)$. It is the counterpart in type theory to Frege's $a^\frown b$ that you know from *Grundgesetze*. ([B.G.S.:] "Did you give the type marking of λ, did you write that out?" Yes, or did I forget that? "I think you did.") Oh, maybe I forgot the type marking of λ, so let me write it on the blackboard,

$$\lambda = \Pi\mathrm{I} : (X : \mathrm{set})(Y : (X)\mathrm{set})(y : (x : X)Y(x))\Pi(X, Y)$$

I gave it in rule form, but did not write it out as a type marking. I have some formal material then left for the first hour next time, and hopefully the second hour will be informal.

Lecture 12, 16.12.93

We are at the stage where we have introduced the operator Π, which is at the same time the universal quantifier, $\Pi = \forall$. We have introduced the Π-introduction operator, $\Pi\mathrm{I} = \lambda$, which is the analogue of Frege's Wertverlauf operator, and I finished last time by introducing the corresponding eliminatory operator,

$$\text{(Π-elim)} \qquad \frac{\begin{array}{c} A : \mathrm{set} \qquad B : (A)\mathrm{set} \qquad C : (\Pi(A,B))\mathrm{set} \\ d : (y : (x : A)B(x))C(\lambda(A,B,y)) \quad c : \Pi(A,B) \end{array}}{\begin{array}{c} \Pi\mathrm{E}(A,B,C,d,c) : C(c) \\ \Pi\mathrm{E}(A,B,C,d,\lambda(A,B,b)) = d(b) : C(\lambda(A,B,b)) \end{array}}$$

Given a set A and a family B of sets over A, and given a family of sets over $\Pi(A,B)$—which we can think of as a property, because of the identification of propositions and sets—and given what is most naturally thought of as a proof that for all functions y from A to B, the property C holds of the corresponding Wertverlauf, that is, of $\lambda(A,B,y)$, and finally, given an arbitrary c from the product $\Pi(A,B)$, then we can conclude that when we apply the operator $\Pi\mathrm{E}$ to A, B, C, d and c, we get an element, or a proof, of $C(c)$.

The operator $\Pi\mathrm{E}$ has to be defined, of course, and it is enough to define it when the last argument is a canonical element of $\Pi(A,B)$, namely when it has λ-form. The definition is then that $\Pi\mathrm{E}$ of A, B, C, d and—now we put in an argument of canonical form—$\lambda(A,B,b)$ is to be equal to $d(b)$. We have to check that this is of the right type, that is, of type $C(\lambda(A,B,b))$, but that is clear, since d is of type $(y : (x : A)B(x))C(\lambda(A,B,y))$, and b is of type $(x : A)B(x)$. This is thus the definition of the Π-elimination operator.

The operator $\Pi\mathrm{E}$ has a particularly important special case, namely the application operator, which must be distinguished from the ordinary function application operation—I will comment on that in a moment. I define app to be $\Pi\mathrm{E}$ with a particular choice of C and a particular choice of d, as follows:

$$\mathrm{app}(A,B,c,a) = \Pi\mathrm{E}(A,B,(z)B(a),(y)y(a),c) : B(a)$$

You take C to be the family over $\Pi(A,B)$ that is constantly equal to $B(a)$. As your d, you take the function that takes an arbitrary y—a function of type $(x : A)B(x)$—into $y(a)$. For any y, $y(a)$ is an element of $B(a)$, as required by how we have determined the family C here. The type of the whole becomes C of the final

argument c, but C is constantly equal to $B(a)$, no matter what the argument is, so we get something which is an element of $B(a)$.

If we define application that way, then it satisfies the following rules:

$$\frac{A : \text{set} \qquad B : (A)\text{set} \qquad a : A \qquad c : \Pi(A, B)}{\text{app}(A, B, c, a) : B(a)}$$

$$\frac{A : \text{set} \qquad B : (A)\text{set} \qquad a : A \qquad b : (x : A)B(x)}{\text{app}(A, B, \lambda(A, B, b), a) = b(a) : B(a)}$$

In the polymorphic notation, the conclusion in the first rule is simply $\text{app}(c, a) : B(a)$, and likewise in the second rule, which shows the definitional equality that the app-operation satisfies. We have to tell what $\text{app}(c, a)$ equals when c has canonical form, or introduction form, so c has to be λ of some function b from A to B. Using our definitions, we find, in the polymorphic notation, that $\text{app}(\lambda(b), a)$ must be $b(a)$, which is indeed an element of $B(a)$,

$$\begin{aligned} \text{app}(\lambda(b), a) &= \Pi\text{E}((y)y(a), \lambda(b)) \\ &= ((y)y(a))(b) \\ &= b(a) \qquad\qquad : B(a) \end{aligned}$$

If I had not strived for utmost generality, I could have skipped the general Π-elimination operator and just introduced the application operation, but I wanted to give the rule following the general pattern that is to be seen for all other rules. Moreover, I will actually need the Π-elimination rule in connection with the discussion of identity in a while. ([B.G.S.: "The easier form can be found in your little Bibliopolis book also." Right, and the more general form is in the preface of the Bibliopolis book.)

We have now recovered one more analogue in type theory of a component of Frege's *Grundgesetze*. I have already explained that λ corresponds to the Wertverlauf operation. The application operation, app, is the converse operation, the operation which Frege wrote as $\xi^\frown\zeta$ and which is a binary function over his universe of all objects. What is important about this function is that it satisfies the following equation:

$$\Delta^\frown\dot{\varepsilon}\Phi(\varepsilon) = \Phi(\Delta)$$

Applying this function—the hook—to an object Δ as first argument and to a course of values, say $\dot{\varepsilon}\Phi(\varepsilon)$ in Frege's notation, as second argument, we get an object which is identical, or equal, to $\Phi(\Delta)$. This is the important property of the hook function.

Frege does not take the hook as a primitive function. Rather, he defines it by means of a definite description, a slightly complicated definite description, and then he has to prove—which takes a couple of pages—that it satisfies this equation.

Frege's Wertverlauf is thus turned into the λ of some function b—maybe in this discussion I will use the polymorphic notation, it fits more closely, and maybe I should make it even more close by writing $(\lambda x)b(x)$, which is the same as $\lambda((x)b(x))$, which is just an η-expansion of b. This is thus nothing but $\lambda(b)$, but I have written it so that it is as close as possible to Frege's $\dot{\varepsilon}\Phi(\varepsilon)$. Frege's hook corresponds to my

application operation, and since I write the second argument as a rather than Δ, my $b(a)$ corresponds to his $\Phi(\Delta)$,

$$\mathrm{app}((\lambda x)b(x), a) = b(a) : B(a)$$

This equation corresponds to the crucial equation that Frege had to prove. The correspondence is therefore close, but there is an important difference: we do not have a universe of all objects. Frege's universe of all objects has been cut up into many, many different domains, hence the application function is not defined as a function of two arguments over that universe. Rather we have to tell exactly what domains its arguments belong to. In the present case, those domains are A, on the one hand, and $\Pi(A, B)$, on the other hand, for the functional argument. This shows what has happened to Frege's application function, the hook function, in the move to type theory.

Observe that we now have yet another concept of function. We have had so far two notions of function, namely function in the old-fashioned sense and function in the modern sense, but we now also have function in the sense of an element of a product, $\Pi(A, B)$.

Let me limit myself to functions of one argument here for simplicity. Then we have, first of all, a function in the old-fashioned sense, b say, which takes values in $B(x)$ when x ranges over A,

$$b : B(x) \quad (x : A)$$

I should say that I presuppose in this discussion that A is a set and that B is a family of sets over A. If we abstract a function in the old-fashioned sense, then we get a function in the modern sense, and let us again call it b,

$$b : (x : A)B(x)$$

This is clearly a function in the modern sense, but now we have the possibility of doing one thing more, namely to take the Wertverlauf, the course of values, of this function, so as to arrive at λ of this, which becomes an element of the function set $\Pi(A, B)$. Let us yet again call it b,

$$b : \Pi(A, B)$$

This is a function in the sense of an element of a function set, or a product. In the special case where B does not depend on the argument, this becomes B^A, or, as I have used, $A \to B$. This is exactly what is properly called a power set—I mean, I would like to call it a power set, and it ought to be called power set, were it not for the fact that power set in ZF set theory is used for something slightly different. Anyway, this is what replaces the power set operation in type theory, so we should definitely call it B raised to the power A.

If we want to quantify over functions, then there is only one of these cases that fits the formation rules for the quantifiers, namely the last one. The universal or existential quantifier always ranges over a set—you cannot quantify over anything but over a set, hence if we want to quantify over functions, they have to be construed as elements of a function set. We do not, however, have only the universal and the

existential quantifier, $(\forall x : A)B(x)$ and $(\exists x : A)B(x)$. If we want to quantify over functions, then we have to take the range of the quantifier to be a product set, but we also have the function space former, $(x : \alpha)\beta$, which, as I have said, we may manipulate approximately like a quantifier. If we regard this also as a kind of universal quantifier, or something analogous to the universal quantifier, then we have greater liberty, because α here is a type, whereas in the universal quantifier properly, it is a set. For α we can, for instance, put in any function type we want. In this sense, therefore, we can also quantify over functions in the modern sense. On the other hand, we cannot quantify over functions in the old-fashioned sense in any way, under any circumstances.

Observe also the three different ways in which functions of these three different kinds are supplied with their arguments. In each case, an element of the set A would serve as a good argument for the function, so let a be an element of A.

$$\begin{array}{ll} b : B(x) \quad (x : A) & b(a/x) : B(a) \\ b : (x : A)B(x) & b(a) : B(a) \\ b : \Pi(A, B) & \mathrm{app}(b, a) : B(a) \end{array}$$

The function in the old-fashioned sense is supplied with its argument in the following way: we take the variable that b is a function of, namely x, and substitute the argument for x, thereby getting an element, $b(a/x)$, of $B(a)$. The function in the modern sense, remember, is supplied with its argument by functional application in the original sense. Finally, a function in the sense of an element of a function set is supplied with its argument by means of the binary—counting also the parameters, it would be quaternary—application function, $\mathrm{app}(b, a)$, or $\mathrm{app}(A, B, b, a)$. As I think I have said before, application in the second sense is logically prior to application in the third sense, because I am using application in the second sense when I form $\mathrm{app}(b, a)$ ([B.G.S.:] "The first two lines also explain beautifully why functions in the modern sense and in the old-fashioned sense get confused, because often one leaves away the strokes, the slash x there, so the same notation gets used for both substitution and functional application." Ja.)

We can pass back and forth between functions in all these senses: if we have a function in one sense, we can get a function in any of the other senses. For instance, if we have a function b in the old-fashioned sense, then we get a function in the modern sense by abstraction, and if I want to get a function as an element of the function set, then I go one step further and get $\lambda(A, B, (x)b)$ as an element of $\Pi(A, B)$,

$$\dfrac{\dfrac{b : B(x) \quad (x : A)}{(x)b : (x : A)B(x)}}{\lambda(A, B, (x)b) : \Pi(A, B)}$$

This shows how to go from the first to the second and from the first to the third. From the second to the first, if we have a function in the modern sense, we can, just by applying it to a variable x, get a function in the old-fashioned sense,

$$\dfrac{b : (x : A)B(x)}{b(x) : B(x) \quad (x : A)}$$

In the other direction, if we have a function in the modern sense, we can apply the λ-operator to it and get an element of the function set,

$$\frac{b : (x : A)B(x)}{\lambda(A, B, b) : \Pi(A, B)}$$

Finally, if we start with an element of the function set, then we can get a function in the old-fashioned sense by applying it in the app-sense to a variable x to get an element of $B(x)$, where x ranges over A, and if we want to get a function in the modern sense from this, we can use the rule of argument removal,

$$\frac{\dfrac{b : \Pi(A, B)}{\mathrm{app}(A, B, b, x) : B(x) \quad (x : A)}}{\mathrm{app}(A, B, b) : (x : A)B(x)}$$

Now the function concept has been dealt with completely, that is, in all of its senses. If one asks what corresponds most closely to the concept of function as it is presented in the set-theoretical preliminaries that you find in almost any mathematics textbook, then it is no doubt the last one—element of a product set—because a function is usually defined either as a functional relation between two sets or as a set of ordered pairs of elements from two sets. That definition has been chosen because you can, by means of the power set axiom, get that the functions in this sense again form a set, so that you can quantify over it, for instance. Of course, one has a need of something corresponding to this also in type theory, and it is precisely for that purpose that the function set is used. Functions in this third sense can, so to say, be collected into a function set so that it makes sense to quantify over them, but the way that gets done is quite different. This third notion of function is the least fundamental notion, so to say, and strangely enough, functions in the two prior senses have more or less disappeared, I think, in current mathematics texts.

I will now take up one more operator besides the Cartesian product, and that is the identity operator. Again I have chosen that in order to be able to comment upon the relation to Frege's treatment of identity. Frege in the *Grundgesetze* had his universal domain of all objects, and then he had equality as a binary relation over that universe. Now that that universal domain has been replaced by a host of different domains, also the equality has to depend on the domain in question. We will therefore have, for each set A, and each pair of elements from A, an identity proposition,

(**I**-form) $$\frac{A : \mathrm{set} \qquad a : A \qquad b : A}{\mathbf{I}(A, a, b) : \mathrm{prop}}$$

Because of the identification of sets and propositions, we can think of $\mathbf{I}(A, a, b)$ as a set, but it is no doubt easier for us to think of it as a proposition. I am showing explicitly the dependence on the set A here, which could be achieved also by writing something like $a =_A b$. In a context where the underlying set is kept fixed all the time, one could even drop the A, and then this would look exactly as in Frege's notation.

In the compact notation where I just declare a constant, the formation rule is written

$$\mathbf{I} : (X : \text{set})(X)(X)\text{set}$$

This single axiom has the same effect as the rule (**I**-form) plus the corresponding identity rule,

$$\frac{A = C : \text{set} \qquad a = c : A \qquad b = d : A}{\mathbf{I}(A, a, b) = \mathbf{I}(C, c, d) : \text{set}}$$

The meaning of that single axiom is precisely that these two rules are valid.

To explain the meaning of the identity relation, since it is expressed in this proposition, $\mathbf{I}(A, a, b)$, we have to define that proposition, which is to say that we have to lay down how the canonical proofs of it are formed as well as how identical canonical proofs are formed. The rule is the following one—maybe I will do something more pedagogical here, namely to start from the rules of identity in predicate logic that we are familiar with and see exactly how one passes from that familiar formulation to the type-theoretical formulation. What are the rules of identity in predicate logic? Well, there are two. The introduction rule, which is simply the law of identity, says that an arbitrary element a of A is identical to itself,

(**II**) $\qquad \mathbf{I}(A, a, a)\ \text{true} \qquad \text{law of identity}$

The corresponding elimination rule says that if a and b are identical, and we have some property C which holds of a, then C holds also of b, and this is usually called the indiscernibility of identicals,

(**IE**) $\qquad \dfrac{\mathbf{I}(A, a, b)\ \text{true} \qquad C(a)\ \text{true}}{C(b)\ \text{true}} \qquad \text{indiscernibility of identicals}$

How are these rules converted into type theory? The first one is simple: since this is the introduction rule, we just have to introduce a corresponding operator into type theory, named **II** in our systematic notation. It depends on the underlying set A and on the element a, so $\mathbf{II}(A, a)$ is the canonical proof that a is identical to itself,

$$\mathbf{II}(A, a) : \mathbf{I}(A, a, a)$$

I happen to have called this operator r, for reflexivity,

$$\text{r}(A, a) : \mathbf{I}(A, a, a)$$

If we want to introduce it in the higher-order notation, we write that r is a constant that takes a set X and an element x of that set into a proof of the identity proposition $\mathbf{I}(X, x, x)$,

$$\text{r} : (X : \text{set})(x : X)\mathbf{I}(X, x, x)$$

This is how the law of identity is converted into type theory.

The more difficult—and interesting—thing is to convert the law of indiscernibility of identicals into type-theoretical notation, because there, as in all elimination rules, there is a proper generalization that goes on. The introduction rule is converted just by introducing another notation, a linear notation, for the proof object, but the elimination rule comes with a proper generalization. Let me first write the

rule of indiscernibility of identicals in a slightly different way,

$$\frac{\begin{array}{cc}\vdots\, c & \vdots\, d\\ \mathbf{I}(A,a,b) & C(a)\end{array}}{C(b)}\,\mathbf{I}\mathrm{E}$$

The rule **I**E takes a proof of the first premiss, call it c, and a proof of the second premiss, call it d, into a proof of the conclusion—of course, it will also depend on A, on a and b, and on C, so we write that now in the functional notation as

$$\mathbf{I}\mathrm{E}(A,a,C,d,b,c) : C(b)$$

It is thus applied to six arguments, and there is no other choice but to begin with the argument A. After A we have to take a, but then you can move the arguments around a bit: it does not matter exactly which order you take them in as long as you do not violate what the typing restrictions require. I will write them in this order: C for the family, then d for the proof of the minor premiss, then b and then c, and the whole thing is a proof of $C(b)$.

This is a first approximation to the elimination rule for identity in type theory, but it is just a passage from the tree-like two-dimensional notation to the linear functional notation. As I have said, however, once we have introduced the proof objects, in particular the proof c of the major premiss, then a generalization is possible. Namely, we may let C also depend upon that proof of the major premiss. Instead of supposing—as we did and as one ordinarily does in predicate logic—that C is a propositional function over the set A, we may assume instead that C has a further argument, namely the proof of the major premiss. It now therefore has two arguments, say y in A, which is the one we had before, but also an argument of type $\mathbf{I}(A,a,y)$, that is, a proof that a is identical to y. So the type of C has been changed from $(A)\mathrm{prop}$ to $(y:A)(\mathbf{I}(A,a,y))\mathrm{prop}$.

What change is then needed to **I**E? The first argument to C is a. That is therefore our y, so we have to put in as second argument to C a canonical proof of $\mathbf{I}(A,a,a)$, which we denoted by $\mathrm{r}(A,a)$. We then conclude in the conclusion that C holds of b, as before, and of c, which is the proof of the major premiss,

$$\frac{\begin{array}{cc}\vdots\, c & \vdots\, d\\ \mathbf{I}(A,a,b) & C(a,\mathrm{r}(A,a))\end{array}}{C(b,c)}\,\mathbf{I}\mathrm{E}$$

Here is then the generalized identity-elimination operator,

$$\mathbf{I}\mathrm{E}(A,a,C,d,b,c) : C(b,c)$$

This new operator has of course to be defined by the appropriate nominal definition, which tells you what it is equal to provided its major argument, the last argument, is of introductory form. The definitional equality rule is this:

$$\frac{A:\mathrm{set} \qquad a:A \qquad C:(y:A)(\mathbf{I}(A,a,y))\mathrm{prop} \qquad d:C(A,\mathrm{r}(A,a))}{\mathbf{I}\mathrm{E}(A,a,C,d,a,\mathrm{r}(A,a)) = d : C(a,\mathrm{r}(A,a))}$$

If A is in set, a is in A, and we have C in $(y : A)(\mathbf{I}(A, a, y))$prop and a proof d in $C(a, \mathrm{r}(A, a))$—under these premisses, $\mathbf{I}$E takes A, a, C and d, and now the last two arguments have to be as in the introduction rule, that is, they have to be a and $r(A, a)$. What then should this be set equal to? Well, first of all, what has its type to be? The last two arguments were a and $\mathrm{r}(A, a)$, so it has to be in $C(a, \mathrm{r}(A, a))$. We therefore have to put it equal to d, because d is what we have of that type. This is then the corresponding equality rule, which is a nominal definition, but again not an explicit one, because it depends on the form of the major argument.

This formulation of the identity rules is due to Christine Paulin-Mohring. It is the one that you get if you start from the usual, so to say, Leibnizian identity rules and convert them into type theory. I had the habit of using another form, which was symmetrical in the two arguments. As you have noticed here, it is not symmetrical in a and b, but Christine Paulin-Mohring has proved that these are equivalent. It is not quite trivial to do that, and it requires further axioms. The most natural thing, however, is no doubt to take the axiom directly in this form that corresponds to the usual identity rules.

So we now have access to identity, and then after the break I will comment on how that relates to Frege's identity.

I have already remarked that between propositions and classes we have actually three identity relations: syntactical identity, intensional identity and what I call extensional identity. After having introduced the propositional identity, the binary propositional function $\mathbf{I}$, we have the same situation between elements of sets, on the one hand, and on the other, in the logical interpretation, between proofs of propositions. Namely, first of all, the relation of syntactical identity I denote by three bars,

$$a \equiv b : A$$

Secondly, we have the relation of intensional identity,

$$a = b : A$$

Intensional identity is expressed in a judgement, and now we also have the propositional identity, $\mathbf{I}(A, a, b)$. This is expressed in a proposition, whereas the intensional one is expressed in a judgement, but to make them comparable here I will transform that proposition into a judgement by writing true after it,

$$\mathbf{I}(A, a, b) \text{ true}$$

It is this identity which deserves to be called, or which is the analogue of, the extensional identity that I introduced before for propositions and classes. Remember, for propositions it was material equivalence, and for classes it was the usual extensional equality between classes.

It is this last identity which, as I have said, corresponds to Frege's identity in the *Grundgesetze*. He had a binary function, $\xi = \zeta$, for the binary identity relation, and that corresponds in my notation to $\mathbf{I}$ of A—we have to fix some domain now,

since we do not have a universal domain—and then we have x and y, both elements of A,

$$\mathbf{I}(A, x, y) : \text{prop} \quad (x : A, y : A)$$

This is the analogue of Frege's identity relation in the *Grundgesetze*, and when he inserts two objects there to get $\Gamma = \Delta$, that corresponds to my proposition $\mathbf{I}(A, a, b)$.

These three identities are between elements of sets, or proofs of propositions. Similarly we get the analogous three identities between element-valued functions: syntactical, intensional and extensional. Assume that A is a set and that B is a family of sets over A—just as I dealt only with classes, but not with relations of several arguments, I will do this for functions of one argument, since the generalization to several arguments is completely trivial. Suppose that we have two functions, f and g. It does not matter for this discussion if we take them in the modern sense or in the old-fashioned sense, but let me take them in the modern sense, hence $f, g : (x : A)B(x)$. First of all, f and g may be syntactically the same,

$$f \equiv g : (x : A)B(x)$$

Or they may be—which is really where the point of gravity is in type theory, it is always on this second level, the mid level, the intensional level—identical as functions,

$$f = g : (x : A)B(x)$$

Finally, they may be extensionally equal, and the extensional equality is again expressed in a proposition—a universal proposition in this case: for all elements x of A, the values are identical in the extensional sense,

$$(\forall x : A)\mathbf{I}(B(x), f(x), g(x)) \text{ true}$$

I forgot to say that they are of decreasing strength, because you can always conclude in this direction. ([B.G.S.:] "If we step over to elements of the function set, then we can also have the identity proposition $\mathbf{I}(\Pi(A, B), f, g)$." That is a special case of element of set, because the function set is a set, so it is included in that case.)

Those are Frege's axioms that had to do with identity. He has, what is it, six axioms. Some of them are just propositional axioms, which cause no problem. Then he has axioms for the universal quantifier, also his higher level quantifier, but the axiom looks the usual way. Then he has his axiom III, which is just his way of stating the identity rules, that is, the law of identity, on the one hand, and the indiscernibility of identicals, on the other hand. His axiom IV is just something that has to do with the amalgamation of horizontals or something,[1] but his axiom V really deserves discussion. In a slightly modernized notation, it says that two Wertverläufe are equal if and only if the functions are extensionally equal,

$$\dot{\varepsilon} f(\varepsilon) = \dot{\alpha} g(\alpha) \supset\subset (\forall x)(f(x) = g(x))$$

[1]Editor's note: Frege's basic law IV is his axiom governing negation:

$$\vdash \begin{array}{l} (\text{—}\ a) = (\text{—}\ b) \\ \vdash\!\!\!\lrcorner\ (\text{—}\ a) = \left(\neg\, b\right) \end{array}$$

One can see Frege's care here, since he was clear about α-conversion, that is, changing bound variables. What does this axiom correspond to in type theory? The analogue is this:

$$\mathbf{I}\big(\Pi(A,B),(\lambda x)f(x),(\lambda x)g(x)\big) \supset\subset (\forall x:A)\mathbf{I}\big(B(x),f(x),g(x)\big)$$

On the left-hand side we have $\mathbf{I}$ of $\Pi(A,B)$, since we are dealing with Wertverläufe, and then instead of two Wertverläufe, we have $(\lambda x)f(x)$ and $(\lambda x)g(x)$, using the polymorphic notation. On the right-hand side we have for all x in A—we have to fix some domain of quantification—and then we have the identity between $f(x)$ and $g(x)$, which are elements of $B(x)$.

Now this simply does not hold in type theory in both directions. It holds in one direction, from left to right. Since $f(x)$ is definitionally equal to $\mathrm{app}((\lambda x)f(x),x)$, and $g(x)$ is definitionally equal to $\mathrm{app}((\lambda x)g(x),x)$, these can be replaced by each other. It does not matter if I write them one way or the other: that is the point about the intensional identity, that it is just a notational change. Write them therefore in the alternative way and consider the relation

$$(\forall x:A)\mathbf{I}(B(x),\mathrm{app}(w,x),\mathrm{app}(z,x)) \quad (w:\Pi(A,B),z:\Pi(A,B))$$

This is clearly a reflexive relation, hence if you have two elements, $(\lambda x)f(x)$ and $(\lambda x)g(x)$, of $\Pi(A,B)$ which are identical, then that reflexive relation must hold between them—that follows by the $\mathbf{I}$-elimination rule, the indiscernibility of identicals. It is therefore clear that you may pass from the left-hand side to the right-hand side.

The implication from right to left, however, is not possible to derive in type theory. Frege's axiom V therefore fails. It does not fail in the sense that one can prove its negation, because, although I think no one has investigated it in detail, surely one would expect that you can model this axiom, which is to say that it does not lead to a contradiction, that it is still consistent to have it. There is, however, absolutely no justification for it, hence there is no justification for Frege's axiom V when you interpret equality on the two sides here as extensional equality. On the right-hand side you have the extensional equality between the functions, and on the left-hand side you have the extensional equality between the Wertverläufe, and that fails.

On the other hand, if we look instead at the corresponding intensional identities, which Frege did not have, then the law does hold. The corresponding intensional identities say, on the right-hand side, that f and g are intensionally identical, that is, as functions, and on the left-hand side you have that $\lambda(f)$, which is the same as $(\lambda x)f(x)$, is intensionally equal to $\lambda(g)$,

$$\frac{f=g:(x:A)B(x)}{\lambda(f)=\lambda(g):\Pi(A,B)}$$

Why does this hold in both directions? In the downwards direction it holds simply by definition of what identity between canonical elements in the set $\Pi(A,B)$ means. Two canonical elements, $\lambda(f)$ and $\lambda(g)$, in $\Pi(A,B)$ are by definition identical if f and g are identical. Now if that is the definition, then we can go also in the other

direction—also formally, if you want to go formally in the other direction: just apply both $\lambda(f)$ and $\lambda(g)$ to x via the app-operation to get $f(x)$ equal to $g(x)$, and then perform an abstraction to get f equal to g, using η-conversion. The intensional law corresponding to Frege's axiom V is therefore valid, but not the extensional one.

I should also say that in referring to Frege here, I have all the time referred to Frege's treatment of identity in the *Grundgesetze*. There he had a consistently extensional interpretation of identity, whereas in the *Begriffsschrift* he had what he called Inhaltsgleichheit and denoted by three bars. The very name Inhaltsgleichheit, and what Frege says about it, makes it clear that that corresponds to my intensional identity—note also that Frege uses three bars for the Inhaltsgleichheit and not for syntactical identity. Frege therefore changed from the intensional identity in the *Begriffsschrift* to the extensional one, the binary propositional function, in the *Grundgesetze*. One can say perhaps that that is part of a general tendency to a consistently extensional approach in the *Grundgesetze*. It is the same with the passage from beurteilbare Inhalte in the *Begriffsschrift*, which are like my propositions, and which we would say naturally are intensional objects, to the truth values that come to replace them in the *Grundgesetze*. That is again a passage from intension to extension.

I would like to let this be the end of the technical discussion, and I would like to use the time that remains to say something about how type theory, that is, all that I have talked about, is related to the traditional points of view in the foundations of mathematics that go under the names of logicism, intuitionism and formalism. It became common to structure foundational discussions in mathematics under these three headings in connection with a meeting in 1931, where talks were given by representatives of all these three directions, and I think also by Waismann on Wittgenstein's philosophy ([B.G.S.:] "The lecture was prepared, but was never delivered." Oh, was never delivered, I see. "Or at least, if delivered, then never printed." I think delivered, but not printed. "Ja, it is now available in Waismann's Nachlass.").

It seems to me that we have at present a kind of revival of interest in these foundational problems concerning mathematics. The foundational discussion was of course very lively before the war, in particular in the 1920s, when the Hilbert–Brouwer controversy reached its peak, and still in the 1930s, as witnessed by this particular conference, for instance, in 1931. At that time, the foundational discussion was lively, and it involved, not only philosophers specialized in the philosophy of mathematics, but also mathematicians in general. It is easy to mention many working mathematicians, outstanding working mathematicians, at that time who had a lively interest in these foundational questions. After the Second World War I think it is fair to say that the interest in these more philosophical questions essentially vanished, and the development in mathematics was largely technical and did not concern these foundational matters.

Now we see a new interest in these matters, perhaps not among mathematicians, I mean, usual mathematicians, but more among computer scientists. There is a resurge of interest in questions concerning the foundations of mathematics, and

all my work on type theory has been directed towards getting some clarification of these foundational problems. It therefore seems fair now, sixty years after the trichotomy was introduced, to say something about how the present situation is related to these three old labels of logicism, intuitionism and formalism.

Let me begin with logicism. Logicism stands as a label, as we know, essentially for the attitude taken by Frege and by Russell and Whitehead in the *Principia*. It has many ingredients, and we have to take them separately. One ingredient in logicism was the desire to reduce arithmetic to pure logic. That was Frege's avowed goal in the *Grundlagen*: to put arithmetic on a secure logical basis. He thought of achieving that by simply reducing, that is, defining the arithmetical notions in such a way that all the arithmetical axioms became provable from purely logical principles. That idea in logicism has turned out not to be a good one. I mean, everybody agrees that that simply does not work, and that idea has certainly been abandoned. Russell still tried to stick to it in the *Principia*, but he did not succeed there either, as pointed out in detail by Gödel, for instance, in his paper on Russell's mathematical logic. So we have certainly abandoned that idea.

On the other hand, the landscape has changed so much that the very interest in the question somehow has disappeared, or is not as great any longer. The formulation of the logical and set-theoretical axioms has progressed, or improved, to such an extent that we now cannot see how this reduction idea could attract so much attention. The axioms for arithmetic have by now been put into a form that is completely analogous to the axioms for the logical operators: we have the same pattern of formation rules, introduction rules and elimination rules in the case of the arithmetical axioms as we have for the logical operators.

The picture that we have now is clear from the following table,

$\emptyset$	$A \times B$	$A + B$	B^A	$\Sigma(A,B)$	$\Pi(A,B)$	$\mathbf{I}(A,a,b)$	$\mathbf{N}$	$\mathrm{List}(A)$	$\mathrm{W}(A,B)$
$\bot$	$A \,\&\, B$	$A \vee B$	$A \supset B$	$\exists(A,B)$	$\forall(A,B)$				

The product corresponds to conjunction, the disjoint union corresponds to disjunction, the function set, the power, B^A, corresponds to implication, the disjoint union of a family of sets corresponds to the existentially quantified proposition, and the product of a family of sets, $\Pi(A,B)$, corresponds to the universally quantified proposition, $\forall(A,B)$. There is always the discussion whether we should count identity among the purely logical notions or not. Frege did count it among the notions that he was allowed to use in defining the natural numbers. Because of the Curry–Howard Isomorphism, however, you see that the rest of these notions are as much logical as they are set-theoretical. That holds, for instance, for the natural numbers and for the set of lists of elements of A, for every set A. There are yet other operators—the natural numbers and lists are both given by what is called ordinary inductive definitions, but there are also operators defined by generalized inductive definitions, of which one is called $\mathrm{W}(A,B)$. I could even make the list longer here.

All of these operators, logical and set-theoretical operators, have axioms that follow the same general pattern that I have displayed in a few places. To put a

dividing line somewhere here and say that those to the left are more basic than those to the right, and that we ought to try to reduce those on the right to those on the left, is no longer something that we would be interested in. Moreover, we know that it is impossible. I mean, we definitely know that we cannot reduce the notion of natural numbers to the purely logical notions that we have here, for instance. To put such a dividing line somewhere and say that this is logical and this is non-logical thus does not fit our present understanding. What we have on both sides of the dashed line here is equally logical, only that the axioms that we get to the right, since we are dealing with inductive definitions and even generalized inductive definitions, are slightly more complicated than the purely logical ones, because recursion or induction comes in, which is not present in the purely logical axioms. There is, however, no difference in principle. This whole question of reducing arithmetic to logic has—well, first of all, it has failed and secondly, it has lost its meaning, its interest.

There is another component of logicism, and that is the anti-psychologism. Frege—and this is characteristic of the whole school, so to say, beginning with Bolzano and ending with Frege and Russell and Husserl—had the desire to make a clear distinction, as Frege said in the *Grundlagen*, between the psychological and the logical. Frege's discussion of this distinction makes it clear that he understood it as the same as that between the subjective and the objective, which in turn is the same as the cleavage that I usually refer to as the act/object cleavage,

psychological	logical
subjective	objective
act	object

The hope of the logicists was that one could deal with logic entirely in an objective fashion, that is, putting it entirely on the right-hand side here, so to say, and expel everything that is psychological in nature. (I should mention that Cantor was also very strongly in in this objectivist tradition, so it is a trend in German philosophy, of which Bolzano is, so to say, the origin.)

This anti-psychologism is of course very understandable, and it was probably a fight that was necessary to fight at the time. The intrusion of psychology in an empirical sense—you might call it empirical psychology or psychology thought of as a kind of natural science—into logic was certainly a big error, as was pointed out with such eloquence and emphasis by Frege and Husserl, for instance. That was certainly a fight that was worth fighting with all the intensity with which it was fought. But also on this point, the situation has changed a bit. Even if we try with this maximally objective attitude, that is, try to put logic wholly on the right-hand side in the table above, and not speak about anything psychological or subjective, we just do not succeed.

It is entirely in the spirit of logicism when negation has become an object of type (prop)prop. We have a logical object here, and similarly the quantifiers have become logical objects, and that is entirely in the spirit of logicism. It is precisely how Russell, for instance, wanted to have it. What, however, does it mean to say that something is an object of type (prop)prop, say, or of type $(X : \mathrm{set})((X)\mathrm{set})\mathrm{set}$?

Let us take another example, the successor function. What does the judgement $\mathbf{s} : (\mathbf{N})\mathbf{N}$ mean? It means that if a is a natural number, then $\mathbf{s}(a)$ is a natural number, and if a and b are identical natural numbers, then $\mathbf{s}(a)$ and $\mathbf{s}(b)$ are identical natural numbers,

$$\frac{a : \mathbf{N}}{\mathbf{s}(a) : \mathbf{N}} \qquad \frac{a = b : \mathbf{N}}{\mathbf{s}(a) = \mathbf{s}(b) : \mathbf{N}}$$

In this meaning explanation you say, if a is a natural number, then $\mathbf{s}(a)$ is a natural number. The question is, What if-then is it that occurs here? Since $a : \mathbf{N}$ is a judgement, it is clearly a relation between judgements. I have taken great care, when giving these explanations, to stress that the if-then that is involved here is the inference relation, so that we may regard the axiom $\mathbf{s} : (\mathbf{N})\mathbf{N}$ as an inference ticket which says that we are allowed to conclude that $\mathbf{s}(a)$ is a natural number from the premiss that a is a natural number, and similarly in the case of identity.

When you come to explain what this judgement means, it is therefore inevitable that you must talk about inference, and when you reach inference, you are at a logical notion, maybe the most basic of all, that just cannot be treated in purely objective terms. The rules of inference are really what specify a logical system, they form the backbone of a logical system. A rule of inference says that if you have proved the premisses, then you may proceed to the conclusion. Here there is an inevitable reference to something subjective: we ourselves, who make these proofs, come into the picture when we talk about inference. That is unavoidable. If logicism was really the idea that you could expel everything that has to do with the subjective, or to do with us, from logic, then it fails, because at the most basic level, we are forced into introducing a reference to those who make the proofs, who draw the inferences, or make the proofs ([B.G.S.:] "It is symptomatic and typical that Frege, Russell, Bolzano all deal with the logical truth of propositions rather than with the validity of inference." Right, and the notion of evidence is so to say expelled from their treatment.)

What is the conclusion of this? The reference to what we are doing, that is, to our drawing inferences, is inevitable. On the other hand, we do not want to fall into psychologism of the kind that Frege and Husserl criticized so severely. They were entirely right that logic does not rest upon some kind of empirical psychology or psychology in the style of a natural science. Logic is a self-sustaining area of its own, and it just does not rest on anything of that sort. What is then the conclusion? Well, we need to talk about the subjective, or acts, here, the acts of inference, for instance, but we need to do it in, so to say, a non-psychological sense—non-psychological in the sense that it has nothing to do with empirical psychology or natural-scientific psychology.

Maybe we could actually get something valuable for psychology out of logic, out of our experience from logic, namely that it is possible to study the soul, or psychic processes, in the same way that we logicians study reasoning processes. After all, our reasoning processes are only some of our psychic processes. In logic we have a particular way of studying reasoning process, and maybe that way of studying reasoning is generalizable to other kinds of activities than reasoning, maybe to all

kinds of psychic activities. This would be, so to say, the germ of something one might call a logical psychology, where we study the psychic after the model of logic.

There was a generally accepted distinction at the time of Kant—and it clearly comes from the time before Kant, either from Wolff or German philosophy just before Kant—between, on the one hand, empirical psychology, and, on the other hand, rational or pure or transcendental psychology. Maybe the proposal that I am making here of a logical study of other activities than reasoning, other psychic activities than reasoning, has some similarity with that old idea of a rational, or pure, psychology as opposed to an empirical psychology. Husserl's conclusion at the very end of his life, in the *Krisis*, went precisely in this direction. In the last paragraph of *Krisis*, § 72, he says,

> Eine reine Psychologie als positive Wissenschaft, eine Psychologie, die die in der Welt lebenden Menschen als reale Tatsachen in der Welt universal erforschen will, so wie andere positive Wissenschaften, Naturwissenschaften und Geisteswissenschaften, gibt es nicht. Es gibt nur eine transzendentale Psychologie, die identisch ist mit der transzendentalen Philosophie.

His view was thus that a pure, or rational, psychology simply becomes phenomenology itself. ([M.v.d.S.:] "It changed completely, because in the *Logical Investigations* he says that "transzendentale Psychologie ist eben auch Psychologie," as a criticism of Kant, so this has changed completely." It is a complete change that he performs at the very end, I mean, in the last paragraph of the last thing that he wrote, after having distinguished so carefully between transcendental phenomenology and transcendental psychology all the time.)

There is another indication here of the necessity that we cannot keep the subjective out. Suppose that we start with the best of will to try to develop logic in an entirely objective way. We formulate all the logical laws, including laws like, if A is proposition and B is a proposition, then $A \,\&\, B$ is a proposition, etc. We set up this whole system of laws that govern the mathematical objects, or the mathematico-logical objects, which in their totality is the mathematical world. It is, so to say, the mathematical world order, the ordo mundi of the mathematical world. When we have formulated these laws, however, and look at them and ask, What are they?, the answer is that they are the rules of inference.

However objective we wanted to be when talking about mathematical objects and the laws that they satisfy, we end up with laws that govern our reasoning. They are the rules of reasoning, or the laws of reasoning, which means that, as much as they make up the mathematical world order, they make up the life order of the mathematicians. They are the rules that mathematicians follow, or are guided by, in their mathematical activity. Even having started in an entirely objective attitude, the final product—and the final product of logicians is always a logical system, or a logical calculus, that is, a system of rules—is just as much ordo vivendi as it is ordo mundi.

We thus reach the somehow striking conclusion that the mathematical world order is the same as the system of rules that the mathematicians follow in what they are doing,

$$\text{ordo mundi} \;=\; \text{ordo vivendi}$$

That shows again that it is impossible to deal with logic in an entirely objective fashion. The act/object duality comes in essentially, and the rules are as much rules governing our activity as they are rules governing the objects. ([B.G.S.:] "You can see that too from the fact that Wittgenstein's *Tractatus*, which perhaps is the most sustained attempt to have a truly objectivized logic, fails, because whether something is a valid inference there depends upon whether a certain proposition is a tautology, and it was meant to be checked "am Symbol allein", and we know now that that cannot be done.)

Concerning the relation between logic and psychology, something has therefore shifted, a bit at least, and maybe we should not be so hostile against the intrusion of the subjective, provided it is properly understood, that is, not in the sense of empirical science. Maybe, as I have said, instead we can transfer our experience from logic also to processes other than the reasoning process.

These were my comments about logicism. Then we have intuitionism. With intuitionism, it is evident that there are certain ingredients which have proved to be very valuable, and all of this work is resting on them. There are, however, other ingredients that just have to be rejected—there is, really, no one who believes in them any longer. To start with, what everybody seems to reject are the ideas that mathematical objects are primarily mental constructions and that mathematics is a languageless activity. These ideas of Brouwer's are of course comprehensible against the background of formalism. Brouwer saw more clearly than anybody else the hollowness of Hilbert's formalist programme, and as a reaction to that he wanted to stress the contrary of formalism to the utmost, which is to say that he wanted to stress content rather than form—one could perhaps call it contentualism, had that been a word.

At the same time, Brouwer was tied to the very traditional idea of a linguistic expression as the expression of a thought, a thought in turn being something inner, or mental, that thus gets exteriorized through the linguistic act. On that view, the thought, that is, the mental thing, is precisely the meaning, or the content, of the linguistic expression. The duality between form and content, or the formal and the contentual, thus comes to coincide for Brouwer with the duality between the real and the mental, or—if you want to use this terminology—the outer and the inner, or the external and the internal. Hence in opposing—rightly—the idea that mathematical objects are just formal expressions that we manipulate, instead of stressing that we should think of the objects, the mathematical objects, which are the contents of the mathematical expressions, he came to the conclusion that we should look at the thoughts, that is, the mental things, of which the mathematical expressions are the outer expressions.

Brouwer thus arrived at the idea that it is the mental constructions that is all that matters, whereas nowadays we, or at least I, would not naturally say this. I would say that what needs to be stressed is that we deal with mathematical objects, and not with expressions, and that these objects are the meanings, or contents, of the mathematical expressions, but they are not mental. Unfortunately, Brouwer could not benefit from Frege—he never read Frege, as far as I know—who was absolutely clear on that point, that these meanings are not mental. Similarly then with the idea that mathematics is a languageless activity. Brouwer wanted to oppose formalism, which made mathematics into a manipulation of formal symbols, but—since the content is mental and whenever the mental is expressed it becomes language and formal—in stressing the content, he instead came to say such a very paradoxical thing as that mathematics is a languageless activity, which I think nobody believes any longer.

On the other hand, there are other ingredients in intuitionism which are absolutely crucial—there are so many that I hardly need to mention them, but particularly the idea of introducing proof objects, the Heyting explanations of the logical operations, and Kolmogorov's and Heyting's, not only semantical explanations, but also formalization of intuitionistic logic, which Brouwer would not have thought of himself because of his prejudices against formalism.

Finally, let me say a few words about formalism. Formalism as a philosophy of mathematics was undermined completely by Gödel's results, and the one who understood best what the situation was, already before Gödel, was no doubt Brouwer. Although formalism as a foundational scheme for mathematics was refuted in the 1930s, the pursuit of the Hilbert program was certainly not without value. The work that I have been talking about here is heavily based on, first of all, the improvement of logical symbolism that was effected by Gentzen in the 1930s in the pursuit of the Hilbert program, and later work in proof theory, such as Gödel's *Dialectica* paper, Prawitz's study of Gentzen's system of natural deduction, Tait's proof of the normalizability of the terms of Gödel's *Dialectica* theory and Howard's isomorphism, which I talked about at length—all of this work has come out in the proof-theoretical tradition, that is, in an attempt to pursue the Hilbert program, and that work has been the immediate basis of type theory. I should also say that in the proof-theoretical tradition there has been a great sensitivity to what principles of reasoning you are using in a particular branch of mathematics, particularly in metamathematics, and that sensitivity to what principles you are using has also been of great importance for the development of type theory.

I think this is all I wanted to say about logicism, intuitionism and formalism. Nowadays we are in the unusual situation that the greatest interest in these foundational questions is among computer scientists. At least for me, that was totally unexpected. I mean, it was not at all what I was aiming at when I started this work, and one may well wonder what effect this will have in the long run. It is characteristic of type theory that it uses precise symbolism very heavily, and a symbolism which probably those who are not used to working with a formal symbolism find forbidding. Computer scientists, on the other hand, are working with formal

symbolism—it is their medium, since they need to write programs. We might run into the situation where the pure mathematicians like to continue to pursue mathematics in the usual, informal but rigorous, way, without adopting any kind of formal symbolism. One can have a lot of sympathy with that attitude of wanting, so to say, to retain mathematics as it has always been. It may, however, be that we run into a situation which is similar to when the calculus was introduced. The symbolism of the infinitesimal calculus was also forbidding when it was first introduced. One tried then to continue doing mathematics in a geometrical fashion, but doing so meant, in fact, lagging behind seriously. We might have a similar situation that these formal methods actually prove so powerful that in the end everyone simply has to learn them. That is, however, something that we know nothing about yet, and since nothing is foreseeable beforehand, surely this is not foreseeable either. It will be very interesting to see what comes out of it eventually. Thank you.