Multiple Concepts of Equality in the New Foundations of Mathematics

by Vladimir Voevodsky

from the Institute for Advanced Study
in Princeton, NJ.

FOMUS, July 18, 2016, Bielefeld.

Let me start by saying a few general worlds about the concept that we refer to by the name “foundation of mathematics”.

I have formulated previously, in the my 2014 IAS Faculty Lecture the following description of the three main components any foundation of mathematics should, in my opinion, have:

The first component is a formal deduction system: a language and rules of manipulating sentences in this language that are purely formal, such that a record of such manipulations can be verified by a computer program.

The second component is a structure that provides a meaning to the sentences of this language in terms of mental objects intuitively comprehensible to humans.

The third component is a structure that enables humans to encode mathematical ideas in terms of the objects directly associated with the language.

In ZFC-based foundations the first component is the system corresponding to the predicate logic theory ZFC with one predicate of two variables, which we usually write using the infix notation “$x \in y$”, and the list of familiar axioms.

To specify the formal deduction system one has to specify the rules that allow one to ***prove*** sentences and there are some choices to be made here for example, of the sequent formulation versus the natural deduction style formulation, but they lead to equivalent concepts of provability.

We often confuse the terminology by using the name ZFC both for the predicate logic theory and for the foundations of mathematics that are based on it and I will do it here as well.

The second component in ZFC probably varies depending on a person.

I like to think of the objects that ZFC describes as of rooted trees without automorphisms and such that the length of each branch is finite. Let us call such trees Zermelo-Fraenkel objects.

One has "x∈y" if "x" is isomorphic, necessarily in a unique way, to a branch of "y" at the root, i.e., elements of "y" are its root branches.

The axiom of infinity postulates existence of such a tree with infinitely many root branches, which is easy to imagine - any tree whose root branches form an infinite set of pair-wise different Zermelo-Fraenkel objects will do.

The third component in the ZFC tells us how to encode mathematical concepts in term of the objects directly addressed by the ZFC sentences.

It starts with encoding definitions and constructions related to sets, which is probably why ZFC is often called a set theory.

This third component is very complicated. As an example, try to represent in the ZFC the function $x \mapsto 2x$ on natural numbers.

The univalent foundations started their growth supported by two needs.

The first one was felt most strongly by mathematicians who worked with categorical and then higher-categorical constructions. It was and still remains to be difficult to articulate but it certainly has something to do with our concept of equality of abstract objects.

The second need was felt mostly by people from very different slice of the academic community. Originally it was articulated as the need to be able to use computers to verify complex mathematical constructions for mistakes.

Later the growth of the univalent foundations became supported by a third need that is articulated as the need to have a better foundation for constructive mathematics.

It was and is felt most strongly by yet another slice of the academic community that has some intersection with the second one but also very little intersection with the first slice.

Returning to the first need I remember discussing it many times with Michael Kapranov back in late 1980's and a typical discussion we had then can be summarized in this form

"We need to have a language where we the objects of a category are never equal. …. But obviously X is equal to X so this is impossible."

at that the discussion would either loop or die out.

In all approaches to the formal deduction system of univalent foundations that are being developed today this conundrum is resolved by the existence of two concepts of equality.

The first one is the concept of the substitutional equality.

The second one is the concept of the transportational equality.

In the intensional Martin-Lof type theories there is one substitutional and one transportational equality.

The substitutional one is the definitional equality.

The transportational one is the one witnessed by elements of the Martin-Lof identity types.

Only substitutional equalities are equalities from the point of view of Leibniz "principle of substitution of equivalents".

In the Martin-Lof type theories definitional equality can not be required, but only checked.

The feeling that "objects of a category are never equal" is resolved into the fact that a true, substitutional, equality of two objects of a category can not be postulated.

The fact that "X=X" remains true as X is definitionally equal to itself, which can be easily checked.

From this one sees immediately that in the intensional Martin-Lof type theories the assertion that

"objects of a category are never equal"

can be extended to the assertion that

"elements of any type are never equal"

since one is not allowed to require substitutional equality between elements of any type.

This fact is the source of the main technical weakness of such theories.

The transportational equality in the MLTT is very different from what Leibniz would understand under the word "equality". In particular, there can be many different "equalities" between two elements.

This is what makes the Univalence Axiom possible, but this is also what makes the use of the transportational equality much more difficult.

The reason that the transportational equality in the Martin-Lof type theories is useful at all lies in the fact that if A is transportationally equal to B then B can be substituted for A directly in the type part of the sentence while the element part part must be replaced by a new expression using the so called transport function.

This transport function takes as one of its arguments the "witness" for the equality, but even more importantly it takes as another a "type family" that connects the substituted and unsubstituted type parts.

To be more precise, let us see what we have to do to make use of a witness to the transportational equality between elements A and B of type T, that is, of an element "e" of the Martin-Lof identity type "Id T A B".

Suppose that our goal is to construct an element of type P_1 and A occurs in the expression for P_1 such that we can view P_1 as P[A/X] where P is now a type family that depends on a parameter X:T.

We can use the transport function to change our goal from P_1 to P_2=P[B/X] because if R:P_2 is an element of P_2 then

$$(\text{transportb } X.P \text{ e } R):P_2$$

This is precisely what the "rewrite" tactic of the proof assistant Coq is trying to do when you run Coq over the command "rewrite e."

You may experiment with it and you will see that in many cases this tactic instead does nothing or returns an error message.

The main reason is that Coq does not know how to find a type family P such that P_l=P[A/X].

Quite often, simply replacing the subexpression A by a variable X and extending the context by declaring X as a variable of type T will lead to a badly formed expression P[X] because the type-checking of P requires more properties of A than the fact that it has type T.

A simple example is with the identity type itself.

For the expression Id T Y Y to be well-formed one must have Y:T. If I substitute a variable X of type U, where U is the universe, for T while leaving Y unchanged the expression will become badly formed because the type judgement

$$X:U \vdash Y:X$$

is invalid.

There is currently a lot of work being done in the direction of trying to extend a Martin-Lof type theory with stronger substitutional equality.

This is difficult, in part because one wants to preserve decidability of type-checking and that necessarily means decidability of the question whether or not two elements of a type are substitutionally equal.

I suggest that one may consider version of equality that is a transportational equality but of a very simple kind.

It comes with its own identity types Id0 but in order to use an element e:Id0 T A B one does not have to find and record a type family connecting P_1 and $P_2=P_1[B/A]$ and instead can write for $R:P_2$ the expression transportb0 R e that will have type P_1.