UniMath - a library of mathematics formalized in the univalent style

by Vladimir Voevodsky
from the Institute for Advanced Study in Princeton, NJ.

ICMS 2016, Jul. 14, 2016, Berlin.

When I was in Oxford earlier this year I saw the title

"Constructive mathematics"

on the daily schedule of classes. Somewhat surprised I went to my computer and found the syllabus for this class. It turned out that the class was about algorithms and their use in mathematics.

UniMath is a library of constructive mathematics but it is not about algorithms.

Users do not see UniMath in the form of compiled code that they execute on their computers and whose value lies in its ability to provide different outputs for different user inputs.

UniMath exists in the form of the source code . To "use" UniMath is to write new source code extending the existing one. To do it one has to read and understand what was written before.

Different people may use UniMath with different goals in mind.

Some may want to verify a complex mathematical proof.

Some may want to explore constructive ramifications of some classical area of mathematics.

Some may use it is as a teaching tool to teach students what rigorous mathematical proofs are.

To explain UniMath I should start by explaining the principles of the univalent foundations.

Any such explanation will necessarily be incomplete and in places hard to understand to one part of the audience while in other places hard to understand to another.

But I will try.

Consider element-level, set-level and higher level mathematics.

- Element-level mathematics works with elements of "fundamental" mathematical sets mostly numbers of different kinds.

- Set-level mathematics works with structures on abstract sets.

- Higher level mathematics is known today as category-level mathematics. It works with structures on collections whose elements are sets with structures and with higher level collections.

Since the university years I heard and later was able to "feel" that objects of categories such as the category of sets or groups do not form a set.

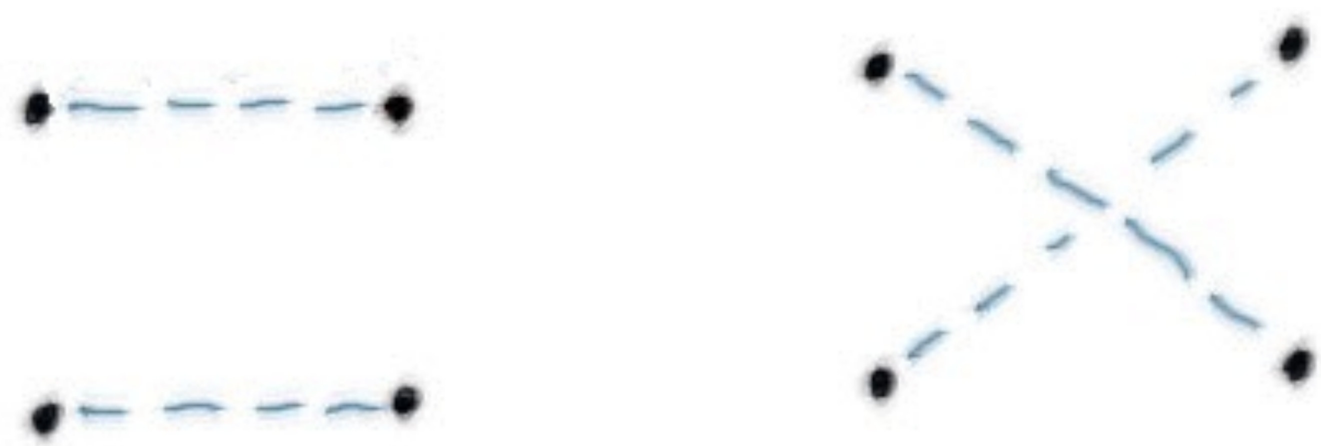
For many years I, following everybody else, assumed that objects of the category of sets do not form a set because there are too many sets.

The first important idea of the univalent foundations is that “all sets” do not form a set for two reasons that can be separated from each other.

One is that there are too many sets.

Another one is that unlike numbers or elements of an abstract set two sets can be equal to each other in more than one way.

For example, two two-element sets can be equal to each other in two ways depending on which element of the first set is identified by this equality with which element of the second:



Therefore, two abstract sets do not form a set, or at least do not always form a set, and therefore there are *small* collections that are not sets.

In the world of the univalent foundations, which you can work with in the UniMath, this idea that there are sets and then there are collections that are more complex than sets is made precise by the concept of *h-level*.

If we accept that there can be more than one equality between two elements then we are led to the picture where the equalities between two elements of a collection form a collection.

Let ***T*** be a collection. Let us denote the fact that ***a*** is an element of ***T*** by the expression ***a:T***. Let

$$Id\ T\ a\ a'$$

be the collection of equalities between the elements ***a*** and ***a'*** of ***T***.

- A collection ***T*** is said to be of h-level 0 if it has exactly one element.

- A collection ***T*** is said to be of h-level n+1 if for any two elements ***a, a’*** the collection ***Id T a a’*** is of h-level n.

A collection is of h-level 1 if for any two elements ***a, a'*** the collection of equalities ***Id T a a'*** has exactly one element. That is, any two elements of ***T*** are equal and in exactly one way.

In classical mathematics that are two such collections - the collection with exactly one element and the empty collection.

These two collections are traditionally used to represent the truth values -

True and False.

A collection is of h-level 2 if for any two elements ***a, a'*** the collection of equalities ***IdT a a'*** is either empty or has exactly one element. That is, if two elements are either not equal or equal in exactly one way.

All collections in the ZFC are such collections as are all collections that we intuitively think of as sets.

That is, sets are collections of the h-level 2.

We already saw that it is natural to consider the collection ***T*** that is formed by two two-element sets as being different.

If we decide that equalities between two sets are bijections we will find out that for any ***a a' :T*** the collection ***Id T a a'*** has two elements.

These two-element collections ***Id T a a'*** behave as collections that we are familiar with from the ZFC, that is they are of h-level 2.

Therefore, according to our definition, ***T*** itself is a collection of h-level 3.

The definition of collections of h-level ***n*** given above is a straightforward definition by induction on ***n*.**

The only difficulty is with the case ***n=0***.

Indeed, what does it mean that a collection contains exactly 1 element?

Does the collection that consists of one two-element set satisfies this definition?

It might seem so, but the collection of equalities between a two-element set and itself is, as we have seen, of h-level 2 and, therefore, the collection that consists of one two-element set is of h-level 3.

It is difficult to make the definition of collections of h-level 0 precise without having some formalism in which to do it.

It can be done in ZFC, but we have already seen that all collections in the ZFC have h-level less or equal to 2 and that ZFC does not render faithfully the intuitive understanding of collections of *abstract* sets.

It can be done using much more advanced concepts of the ZFC-based mathematics - the concepts of the homotopy theory.

In this rendering a collection is represented by a space, simplicial set or an object of some other class that can be used to model homotopy types.

A collection of h-level ***0*** is represented by a *contractible* homotopy type and ***Id T a a'*** by the space of paths between points ***a*** and ***a'*** of ***T.***

A collection of h-level 1 will then be representable by a homotopy type that is contractible or empty and a collection of h-level 2 by a homotopy type that is homotopy equivalent to a discrete set.

This is very useful for the development of intuition about h-levels and higher-level collections but not very useful from the foundational perspective.

A remarkable fact on which the type-theoretic formalization of mathematics of the univalent foundations is based is that ***Id T a a'*** and collections of h-level 0 can be faithfully defined in the so called Martin-Lof type theories.

Collections in these theories are called types.

Types of h-level 0 are called contractible types and the definition of being contractible is called in all libraries based on the original "Foundations" library ***iscontr*** or ***isContr*** and has the form

Definition iscontr (T :Type) := Σ (x :T) , Π (y : T) , Id T y x .

Understanding this definition and how it appears in each particular library is the key to the understanding of the univalent formalization style in general and of the specifics of a given library.

In the UniMath this definition takes the form:

Definition iscontr (T : UU) :=

total2 (fun x : T => forall (y :T) , paths y x) .

Types of h-level 1 are usually called propositions instead of truth values.

The corresponding definition in the notation of the original Martin-Lof type theory has the form

Definition isaprop (T :Type) := Π (x y :T) , iscontr (Id T x y)

and in the notation of the UniMath the form

Definition isaprop (T :Type) := forall (x y :T), iscontr (paths x y) .

Maybe the next thing that I may suggest to understand is the proof and the importance of the following theorem

Theorem isapropiscontr (T :Type) := isaprop (iscontr T) .

This theorem has the same form in the original Martin-Lof notation and in the UniMath, however it can be proved in UniMath that assumes the Univalence Axiom, and, as its corollary, the so called function extensionality axiom, but not in the original Martin-Lof Type Theory.

The end of slides.