# UniMath: its origins, present, and future

Benedikt Ahrens

# What is UniMath?

"UniMath" may refer to several things, e.g.,

1. A univalent foundation of mathematics
2. A formal language (deductive system) (idealized UniMath)
3. A subset of the language of Coq, including $U : U$ (realized UniMath)
4. A library of mathematics formalized in realized UniMath, available on Github

At the origin of UniMath is Vladimir Voevodsky. Vladimir died on 30 September 2017, a few months before the First School and Workshop on Univalent Mathematics.

# In this talk

The development of the different UniMaths has happened in parallel. They are very much intertwined.

This week, you have learned about idealized UniMath (2) and realized UniMath (3). This talk is about

- the origins of UniMath;
- the difference between idealized (2) and realized (3) UniMath; and
- the contents of UniMath (the library).

# What is a foundation of mathematics?

A foundation of mathematics is specified by three things:

1. Syntax for mathematical objects
2. Notion of proposition and proof
3. Interpretation of the syntax into the world of mathematical objects

UniMath is the following foundation:

1. univalent type theory
2. logic of h-propositions
3. Interpretation of types as Kan complexes

We have not discussed point 3 this week.

# Outline

# Outline

# Vladimir's interest in computer-checked proofs

> *PS I am thinking again about the applications of computers to pure math. Do you know of anyone working in this area? I mean mostly some kind of a computer language to describe mathematical structures, their properties and proofs in such a way that ultimately one may have mathematical knowledge archived and logically verified in a fixed format.*

Email to Dan Grayson, Sept 2002

# Vladimir's ideas about computer-checked proofs I

*Let us start with what would be a perfect system of such sort [. . . ].*

*Ideally one wants a math oracle. A user inputs a type expression and the system either returns a term of this type or says that it has no terms. The type expression may be "of level 0" i.e. it can correspond to a statement in which case a term is a proof and the absence of terms means that the statement is not provable. It may be "of level 1" e.g. it might be the type of solutions of an equation. In that case the system produces a solution or says that there are non. It may be "of level 2" e.g. It might be the type of all solvable groups of order 35555. In that case the system produces an example of such a group etc.*

# Vladimir's ideas about computer-checked proofs II

*A realistic approximation [. . . ] may look as follows. Imagine a web-based system with a lot of users (both "creators" and "consumers") and a very large "database". Originally the database is empty (or, rather, contains only the primitive "knowledge" a-la axioms). User A (say in Princeton) inputs a type expression and builds up a term of this type either in one step (just types it in) or in many steps using the standard proof-assistant capabilities of the system. Both the original and all the intermediate type expressions which occur in the process are filed (in the real time) in the database. Enter user B (somewhere in Brazil), who inputs another type expression and begins the process of constructing a term of his type.*

# Vladimir's ideas about computer-checked proofs II

> *[. . . ] My homotopy lambda calculus is an attempt to create a system which is very good at dealing with equivalences. In particular it is supposed to have the property that given any type expression F(T) depending on a term subexpression t of type T and an equivalence t->t' (a term of the type Eq(T;t,t')) there is a mechanical way to create a new expression F' now depending on t' and an equivalence between F(T) and F'(T') (note that to get F' one can not just substitute t' for t in F – the resulting expression will most likely be syntactically incorrect).*

Email to Dan Grayson, Sept 2006

# Vladimir learning how to use Coq

> *I am thinking a lot these days about foundations of math and automated proof verification. My old idea about a "univalent" homotopy theoretical models of Martin-Lof type systems survived the verification stage an I am in the process of writing things up.*
>
> *I also took a course at the Princeton CS department which was for most part about Coq and was very impressed both by how much can be proved in it in a reasonable time and by how many young students attended (45, 35 undergrad + 10 grad!).*

Email to Dan Grayson, Dec 2009

# The Foundations library

In Feb 2010, Vladimir started writing the Coq library *Foundations*, making precise his ideas conceived during three years and collected in *A very short note on homotopy λ-calculus*.

```
Fixpoint isofhlevel (n:nat) (X:UU): UU:=
match n with
O => iscontr X |
S m => forall x:X, forall x':X, (isofhlevel m (paths _ x x'))
end.


Theorem hlevelretract (n:nat)(X:UU)(Y:UU)(p:X -> Y)(s:Y ->X)(eps: forall y:Y, paths _  (p (s y)) y): (isofhlevel n X) -> (isofhlevel n Y).
Proof. intro. induction n. intros. apply (contrl1' _ _ p s eps X0).
intros. unfold isofhlevel. intros. unfold isofhlevel in X0. assert (is: isofhlevel n (paths _ (s x) (s x'))).  apply X0.
set (s':= maponpaths _ _ s x x'). set (p':= pathssec2 _ _ s p eps x x'). set (eps':= pathssec3 _ _ s p eps x x'). apply (IHn _ _ p' s' eps' is). Defined.

Corollary  hlevelweqf (n:nat)(X:UU)(Y:UU)(f:X -> Y)(is: isweq _ _ f): (isofhlevel n X) -> (isofhlevel n Y).
Proof. intros.  apply (hlevelretract n _ _ f (invmap _ _ f is) (weqfg _ _ f is)). assumption. Defined.

Corollary  hlevelweqb (n:nat)(X:UU)(Y:UU)(f:X -> Y)(is: isweq _ _ f): (isofhlevel n Y) -> (isofhlevel n X).
Proof. intros.  apply (hlevelretract n _ _ (invmap _ _ f is) f (weqgf _ _ f is)). assumption. Defined.


Definition isaprop (X:UU): UU := isofhlevel (S O) X.
```

Other libraries were built on top of *Foundations*. . .

# Founding of UniMath (the library)

UniMath was founded in spring 2014, by combining three libraries:

- Foundations (Voevodsky)
- RezkCompletion (Ahrens, Kapulkin, Shulman) (started Feb 2013)
- Ktheory (Grayson) (started Oct 2013)

A library on p-adic numbers, written by Pelayo, Voevodsky, and Warren in 2012, was added to UniMath later as package *PAdics*.

# Outline

# Some information on the UniMath library

- ca. 150,000 loc (120,000 loc in Dec 2017)
- Two more repositories building on top of UniMath
  - TypeTheory (ca. 15,000 loc)
  - largecatmodules (ca. 10,000 loc)
- ca. 30 contributors (15 contributors in Dec 2017), plus many maintenance contributions from Coq developers
- Distributed under free software license

# The UniMath library

Organized in ‘packages’:

- Foundations
- More Foundations
- Combinatorics
- Algebra
- Number Systems
- Synthetic Homotopy Theory
- Real Numbers
- Category Theory
- Homological Algebra
- K-theory
- Topology
- Homological Algebra
- Substitution Systems
- . . .

# Package Foundations

Provides a specification of “univalent foundations”:

- Type and term constructors
- Definition of ‘weak equivalence’
- Definition of ‘h-level’ for types and functions
- Facts on propositions and sets
- Univalence axiom and consequences
- Arithmetic on natural numbers
- . . .

# Package MoreFoundations

Several variants of the same construction, e.g.,

$$(a,b) = (a',b') \quad \simeq \quad \sum_{p:a=a'} p_*(b) = b'$$

$$t = t' \quad \simeq \quad \sum_{p:t.1=t'.1} p_*(t.2) = t'.2$$

useful, but not interesting—dilutes Foundations.
Package MoreFoundations mirrors structure of Foundations:

- one variant of each construction in Foundations
- others in MoreFoundations

⇝ Foundations as a textbook on univalent foundations

# Package Combinatorics

- Standard finite sets
- Finite sets
- Finite sequences
- Ordered sets
- Wellordered sets
- Zermelo's wellordering theorem:
  AC $\Rightarrow$ every set can be well-ordered

# Package CategoryTheory

- (Univalent) categories, functors, natural transformations
- Category of sets and Yoneda lemma
- Adjunctions, equivalences
- Rezk completion
- (Co)Limits in general, direct definition of some special (co)limits
- Abelian categories
- Displayed categories
- NEW: Bicategory theory (called for by Vladimir in talk given at Big Proof programme at Newton Institute, Cambridge, in 2017)
- ...

# Package Algebra

- Lattices
- Binary operations
- Monoids and groups, abelian groups
- Rigs and rings;
- Ring of differences: construction of a (commutative) ring from a (commutative) rig (used to define the integers)
- Domains and fields
- Field of fractions: construction of a field from an integral domain with decidable equality (used to define the rationals)
- ...

# More specialized packages

- Folds: categories via univalent FOLDS
- Homological Algebra: 5-lemma for triangulated categories, naive homotopy category $K(A)$ is triangulated
- Ktheory: Torsors, the circle as $B(\mathbb{Z})$
- Number Systems: Integers and rationals via constructions in Algebra package
- SubstitutionSystems: theory of syntax with variable binding
- Tactics: tactics for proving results on monoids, groups, etc.

# How are packages created?

- Usually a person interested in topic X starts formalizing it in UniMath (e.g., Substitution Systems, Homological Algebra)
- For instance, in June 2016, Tomi Pannila, mathematician by training, joined the project with the goal of formalizing his Masters thesis.
- He wrote the package on Homological Algebra, with about 40,000 loc
- In May 2017, he then decided to stop formalizing in UniMath and writing a proof assistant himself.

# How are packages created?

- Usually a person interested in topic X starts formalizing it in UniMath (e.g., Substitution Systems, Homological Algebra)
- For instance, in June 2016, Tomi Pannila, mathematician by training, joined the project with the goal of formalizing his Masters thesis.
- He wrote the package on Homological Algebra, with about 40,000 loc
- In May 2017, he then decided to stop formalizing in UniMath and writing a proof assistant himself.
- We have not heard from him since.

# Outline

# Outline

# Propositional resizing

In idealized UniMath, there is a sequence $\mathsf{U}_0 : \mathsf{U}_1 : \mathsf{U}_2 : \mathsf{U}_3 : \ldots$ of universes.
In a talk at TYPES 2011, Vladimir suggested a set of **resizing rules**, in particular:

- If type $A : \mathsf{U}_i$ is a proposition, then $A$ lives in the lowest universe.
- For any universe $\mathsf{U}_i$, the type $\mathsf{hProp}(\mathsf{U}_i) = \sum_{X:\mathsf{U}_i} \mathsf{isaprop}(X)$ lives in the lowest universe.

Weakened versions of those rules—"up to equivalence"—are validated by Vladimir's simplicial set model.

# Use of resizing in (idealized) UniMath

Propositional resizing is needed to achieve that

- the propositional truncation of $A$,

$$\|A\| := \prod_{P:\mathsf{hProp}(\mathsf{U})} (A \to P) \to P$$

  lives in the same universe as $A$
- the set quotient of $(X, R)$ lives in the same universe as $X : \mathsf{U}_i$ (recall that elements of the quotient are equivalence classes $e : X \to \mathsf{hProp}(\mathsf{U}_k)$)

# Idealized vs realized UniMath

- In idealized UniMath, we have resizing rules available.
- In UniMath as currently implemented, on top of Coq, we use $\mathsf{U} : \mathsf{U}$ to achieve resizing.
- As per Thorsten's lecture, realized UniMath is inconsistent.
- In current UniMath (the library), we hence have to be careful to not use any paradox by accident.

Research on theory and implementation of resizing rules is ongoing. . .

# Research problems related to resizing

## Show consistency of resizing rules in univalent type theory

In the TYPES 2011 talk, Vladimir sketches a model of resizing that does not validate univalence.

## Implement a proof assistant with propositional resizing

- In UniMath, resizing is currently achieved by the inconsistent rule U : U
- Dan Grayson is currently working on isolating the uses of U : U into “resizing modules”

# Outline

# What to work on?

- UniMath aims to accommodate all of mathematics
- Anybody is welcome to contribute the mathematics they are interested in
- The purpose of this school/workshop has been to enable people to work on their own goals in UniMath

I am going to present some goals that have been suggested; you might pursue others!

# Vladimir's goals for UniMath

In a lecture in July 2017, Vladimir outlined three goals for the UniMath library:

# Vladimir's goals for UniMath I

" The first direction is the development and formalization of the mathematics surrounding the study of syntax and semantics of dependent type theories.

This direction itself has now branched into several subdirections. The most clearly aimed among those is the one whose goal is to formalize the construction of the univalent simplicial set model.

Its development progresses well. "

# Vladimir's goals for UniMath II

" Another direction is the one that I have stated in my Bernays lectures at the ETH in 2014 - to formalize a proof of Milnor's conjecture on Galois cohomology.

It has not been developing much. On the one hand, I discovered that formalizing it classically it is not very interesting to me because I am quite confident in that proof and in its extension to the Bloch-Kato Conjecture.

On the other hand, when planning a development of a constructive version of this proof one soon encounters a problem. The proof uses the so called Merkurjev-Suslin transfinite argument that relies on the Zermelo's well- ordering theorem that in turn relies on the axiom of choice for sets. "

# Vladimir's goals for UniMath III

" Finally, there is the third direction that I think UniMath can and should develop. It is the direction towards the modern theory of geometry and topology of manifolds.

The first step in this direction can be a definition of a univalent category of smooth manifolds. Univalence will force all the constructions relying on this category to be invariant, or maybe better to say equivariant, with respect to diffeomorphisms.

No one knows how much of the theory of smooth manifolds can be developed constructively which creates an additional challenge. "

# UniMath for education

Make UniMath suitable for teaching

- type theory and univalent foundations
- mathematics

To this end:

- restrict use of tactics to a small, well-defined, set
- simplify installation
- improve documentation
- modernize user interface
- anything else? Suggestions welcome!

Thanks

# Kudos to Coq

UniMath is a large codebase, with contributors coming and going. Its continued development would not be possible without the help of Coq developers, who

- implement features useful for Foundations/UniMath
- test Coq with UniMath to check for regressions
- ensure compatibility upon version upgrades
- make Coq faster with every version

# Thanks. . .

## to the mentors

In particular, to

- Langston Barrett
- Joj Helfer
- Tom de Jong

for their tutoring during the problem sessions.

## to you, the participants

for coming to Birmingham, and for your interest in, and contributions to, UniMath.

# Thanks. . .

## to the mentors

In particular, to

- Langston Barrett
- Joj Helfer
- Tom de Jong

for their tutoring during the problem sessions.

## to you, the participants

for coming to Birmingham, and for your interest in, and contributions to, UniMath.

I hope to see you again soon.

Thanks!

# References


- Vladimir's emails to Dan Grayson
  https://groups.google.com/forum/#!topic/homotopytypetheory/K_4bAZEDRvE
- Vladimir's library *Foundations*
  https://github.com/vladimirias/Foundations,
  archived at https://github.com/UniMath/Foundations
- Vladimir's talk at TYPES 2011
  https://www.math.ias.edu/vladimir/sites/math.ias.edu.vladimir/files/2011_Bergen.pdf
- Vladimir's talk on UniMath in July 2017
  https://www.newton.ac.uk/seminar/20170710113012301

# Specific tasks relating to universes in UniMath

**Theoretical:** Understand Coq's universe management and check that its rules are validated in the simplicial set model. E.g., with option `Polymorphic Inductive Cumulativity`, an element of `paths@i X x y` also has type `paths@j X x y`, provided `i` and `j` are universe levels at least as large as that of `X`.

**Practical:** Simplify universe management. E.g., would like to be able to define hSet for types at level i to itself be at level i+1, but instead hSet takes two parameters as result of Coq universe management.

Currently: Foundations, MoreFoundations, Combinatorics, and 4 of the Algebra files with have been converted to resizing, see `https://github.com/DanGrayson/UniMath/tree/file-by-file-type-in-type`