Using function extensionality in Agda, (non-)computationally

Chuangjie Xu

j.w.w. Martín Hötzel Escardó

Ludwig-Maximilians-Universität München

Oberseminar Mathematische Logik, 6 December 2017

A short introduction to Martin-Löf type theory

A foundation for constructive math, based on the Curry-Howard correspondence

First order logic		Dependent type theory	
$\bot$	$\cong$	$\mathbb{0}$	
$\top$	$\cong$	$\mathbb{1}$	
$P \wedge Q$	$\cong$	$P \times Q$	
$P \vee Q$	$\cong$	$P + Q$	
$P \Rightarrow Q$	$\cong$	$P \to Q$	
$\forall(x:A).P(x)$	$\cong$	$\Pi(x:A).P(x)$	(dependent function)
$\exists(x:A).P(x)$	$\cong$	$\Sigma(x:A).P(x)$	(dependent pair)
$a = b$	$\cong$	$\mathrm{Id}_A(a,b)$	(identity type)

Axiom of choice?

$$\Pi(x:X).\Sigma(y:Y).A(x,y) \to \Sigma(f:X \to Y).\Pi(x:X).A(x,fx)$$

MLTT as a foundation: good or bad?

Intensional MLTT has good computational features (e.g. decidable type-checking, terminating normalisation, canonicity) and hence

- it can be regarded as a dependently typed programming language, and
- the design of a number of proof assistants and programming languages is based on certain variants of MLTT, including Agda, Coq and Lean.

However, certain difficulties arise in such type-theoretic development of math due to

- the presence of Proof Relevance (e.g. subsets/subtypes),
- the absence of Function Extensionality (funext)

$$\Pi(X:\mathcal{U})(Y:X\to\mathcal{U})(f,g:\Pi(x:X).Y(x)).\ (\Pi(x:X).fx=gx)\to f=g.$$

In this talk

I will use my thesis

> *A continuous computational interpretation of type theories*, doctoral dissertation, the University of Birmingham, 2015.

as an example to

- illustrate the difficulties of formalising math in MLTT,
- discuss some approaches to address the related issues, and
- execute some examples in Agda with (funext) used non-computationally and computationally.

Uniform continuity of functions $\mathbf{2}^{\mathbb{N}} \to \mathbb{N}$

$$\forall(f : \mathbf{2}^{\mathbb{N}} \to \mathbb{N}).\ \exists(n : \mathbb{N}).\ \forall(\alpha, \beta : \mathbf{2}^{\mathbb{N}}).\ \alpha =_n \beta \Rightarrow f(\alpha) = f(\beta)$$

- A finite prefix of an input of f is enough to compute the output.
- Not provable but consistent in HA^{ω},
  validated by the topological topos (using classical logic or other axioms).
- Its Curry–Howard interpretation
  $$\Pi(f : \mathbf{2}^{\mathbb{N}} \to \mathbb{N}).\ \Sigma(n : \mathbb{N}).\ \Pi(\alpha, \beta : \mathbf{2}^{\mathbb{N}}).\ \alpha =_n \beta \to f(\alpha) = f(\beta)$$
  is also consistent in MLTT.
- We have a constructive variation of the topological topos. The concrete objects in this model are called C-spaces (analogous to limit spaces).

$\mathbb{C}$-spaces and continuous maps

Def. A $\mathbb{C}$-topology on a set X is a collection P of probes $\mathbf{2}^{\mathbb{N}} \to X$ subject to the following probe axioms:

1. All constant maps are in P.
2. If $t\colon \mathbf{2}^{\mathbb{N}} \to \mathbf{2}^{\mathbb{N}}$ is uniformly continuous and $p \in P$, then $p \circ t \in P$.
3. For any two maps $p_0, p_1 \in P$, the unique map $p\colon \mathbf{2}^{\mathbb{N}} \to X$ defined by $p(i * \alpha) = p_i(\alpha)$ is in P.

A $\mathbb{C}$-space is a set X equipped with $\mathbb{C}$-topology.

A function $f\colon X \to Y$ of $\mathbb{C}$-spaces is continuous if $f \circ p \in P_Y$ whenever $p \in P_X$.

Example: All continuous maps from $\mathbf{2}^{\mathbb{N}}$ (with the usual topology) to any topological space X form a $\mathbb{C}$-topology on X. Any continuous map of topological spaces is continuous w.r.t. the above $\mathbb{C}$-topology.

Discrete C-spaces

Def. A map $p\colon \mathbf{2}^{\mathbb{N}} \to X$ into a set X is called locally constant iff $\exists(n : \mathbb{N}).\ \forall(\alpha, \beta : \mathbf{2}^{\mathbb{N}}).\ \alpha =_n \beta \Rightarrow f(\alpha) = f(\beta)$.

Lemma
The locally constant maps $\mathbf{2}^{\mathbb{N}} \to X$ form a C-topology which has the smallest amount of probes on X.

Def. A C-space X is discrete if all functions $X \to Y$ into any C-space Y are continuous.

Lemma
A C-space is discrete iff its probes are precisely the locally constant functions.

Def. We thus refer to the collection of all locally constant maps $\mathbf{2}^{\mathbb{N}} \to X$ as the discrete C-topology on X.

- The discrete C-topology on $\mathbf{2}$ or $\mathbb{N}$ is the set of uniformly continuous maps.
- The discrete space $\mathbf{2}$ is the coproduct of two copies of the terminal space.
- The discrete space $\mathbb{N}$ is the natural numbers object.

Yoneda Lemma and Fan functional

The monoid C of uniformly continuous $\mathbf{2}^{\mathbb{N}} \to \mathbf{2}^{\mathbb{N}}$ is a C-topology on $\mathbf{2}^{\mathbb{N}}$.

$$(\mathbf{2}^{\mathbb{N}}, \mathrm{C}) \;=\; \text{the exponential of the two discrete } \mathrm{C}\text{-spaces}$$

The Yoneda Lemma says that a map $\mathbf{2}^{\mathbb{N}} \to X$ into a C-space X is a probe iff it is continuous in the sense of the category of C-spaces.

Lemma
The exponential $\mathbb{N}^{\mathbf{2}^{\mathbb{N}}}$ is a discrete C-space.

Theorem
There is a continuous functional $\mathrm{fan}\colon \mathbb{N}^{\mathbf{2}^{\mathbb{N}}} \to \mathbb{N}$ that calculates (minimal) moduli of uniform continuity.

Modelling uniform continuity

$\mathbb{C}$-spaces provide a model of system T and dependent types:

1. Cartesian closed structure — simply typed λ-calculus.
2. Locally cartesian closed structure — dependent types.
3. Natural numbers object — base type and primitive recursion principle.

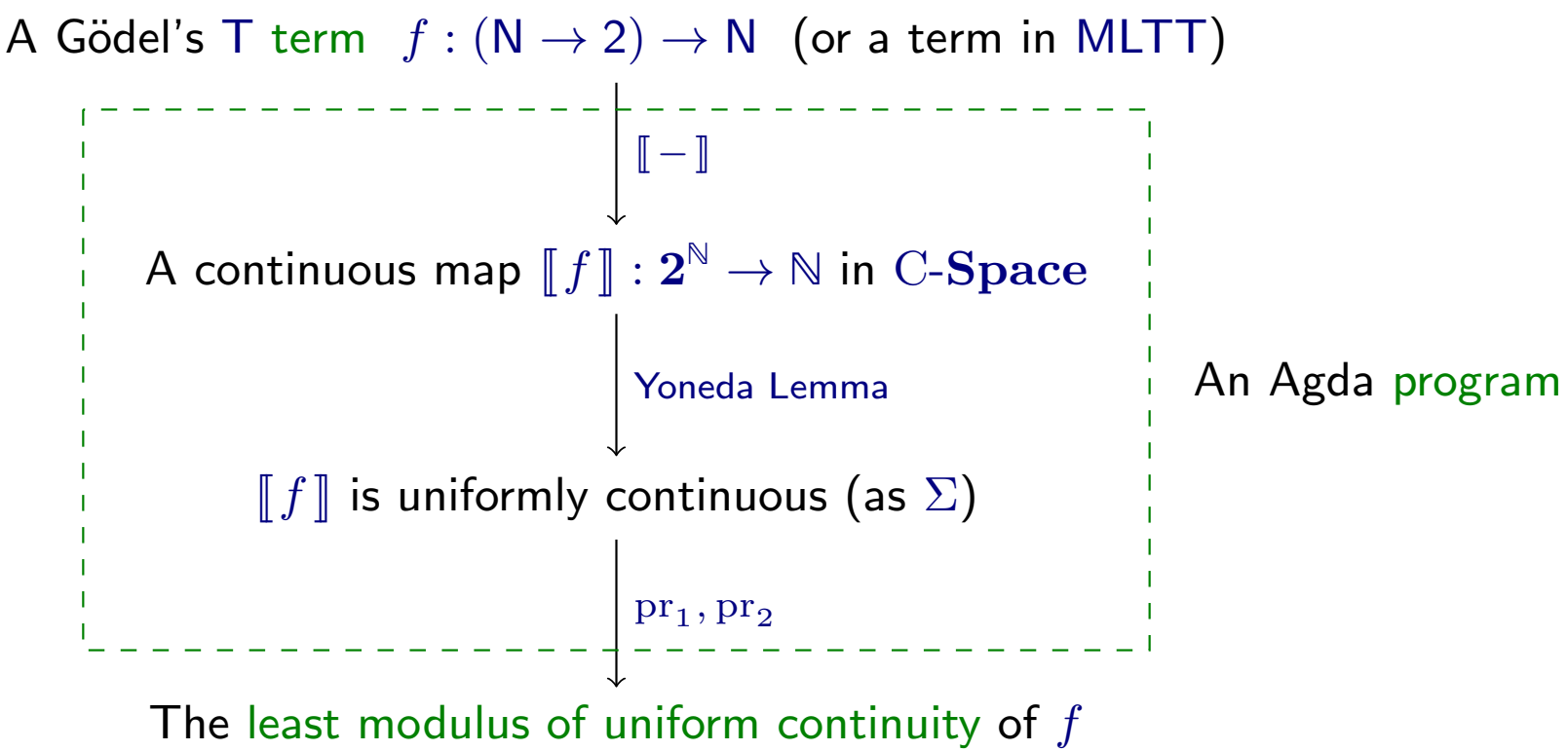
Theorem

The uniform continuity axiom

$$\forall(f : \mathbf{2}^{\mathbb{N}} \to \mathbb{N}).\ \exists(n : \mathbb{N}).\ \forall(\alpha, \beta : \mathbf{2}^{\mathbb{N}}).\ \alpha =_n \beta \Rightarrow f(\alpha) = f(\beta)$$

is validated by the fan functional.

Computing moduli of uniform continuity



A naive formulation of C-spaces

- Define a type family $\mathrm{C}\colon (\mathbf{2}^{\mathbb{N}} \to \mathbf{2}^{\mathbb{N}}) \to \mathcal{U}$ by
  $\mathrm{C}(t) :\equiv \Pi(m\colon\mathbb{N}).\ \Sigma(n\colon\mathbb{N}).\ \Pi(\alpha,\beta\colon\mathbf{2}^{\mathbb{N}}).\ \alpha =_n \beta \to t(\alpha) =_m t(\beta)$
  and write $t \in \mathrm{C}$ to denote $\mathrm{C}(t)$.

- Given $X\colon\mathcal{U}$ and $P\colon (\mathbf{2}^{\mathbb{N}} \to X) \to \mathcal{U}$, define $\text{probe-axioms}(X, P)$ to be
  $(\Pi(x\colon X).\ \lambda\alpha.x \in P)$
  $\times\ (\Pi(p\colon \mathbf{2}^{\mathbb{N}} \to X).\ p \in P \to \Pi(t\colon \mathbf{2}^{\mathbb{N}} \to \mathbf{2}^{\mathbb{N}}).\ t \in \mathrm{C} \to p \circ t \in P)$
  $\times\ (\Pi(p\colon \mathbf{2}^{\mathbb{N}} \to X).\ (\Sigma(n\colon\mathbb{N}).\ \Pi(s\colon\mathbf{2}^n).\ p \circ \mathrm{cons}_s \in P) \to p \in P)$

- Define the type of C-spaces by
  $\text{Space} :\equiv \Sigma(X\colon\mathcal{U}).\ \Sigma(P\colon (\mathbf{2}^{\mathbb{N}} \to X) \to \mathcal{U}).\ \text{probe-axioms}(X, P).$

Problems due to the absence of function extensionality

Working with this formulation of $\mathbb{C}$-spaces, some problems occur in the constructions of the following:

1. discrete $\mathbb{C}$-spaces,
2. exponentials of $\mathbb{C}$-spaces, and
3. the fan functional.

All of them are related to the fact that function extensionality (funext)

$$\Pi(X:\mathcal{U})(Y:X\to\mathcal{U})(f,g:\Pi(x:X).Y(x)).\ (\Pi(x:X).fx=gx)\to f=g$$

is not available in intensional Martin-Löf type theory.

The issues of (1) and (2) arise in the verifications of sheaf condition, which rely on the equation

$$\mathrm{cons}\ (\mathrm{take}\ \ n\ \alpha)\ (\mathrm{drop}\ \ n\ \alpha)=\alpha.$$

But it holds only up to pointwise equality.

Problems due to the presence of proof relevance

Function extensionality is needed, but insufficient to construct (3) the fan functional, because type theory is proof-relevant.

When showing that the domain $\mathbb{N}^{\mathbf{2}^{\mathbb{N}}}$ of the fan functional is a discrete C-space (*i.e.* any probe $\mathbf{2}^{\mathbb{N}} \to \mathbb{N}^{\mathbf{2}^{\mathbb{N}}}$ is locally constant), we need to prove equality of uniform-continuity witnesses of maps $\mathbf{2}^{\mathbb{N}} \to \mathbb{N}$. But there could be many uniform-continuity witnesses for a given map $\mathbf{2}^{\mathbb{N}} \to \mathbb{N}$.

We refine uniform continuity to mean that there exists a minimal modulus:

$$\mathrm{UC}(f) :\equiv \Sigma_{\mathrm{min}}(m{:}\mathbb{N}).\ \Pi(\alpha, \beta{:}\mathbf{2}^{\mathbb{N}}).\ \alpha =_m \beta \to f(\alpha) = f(\beta)$$

which becomes a proposition (that is a type with at most one element).

If two uniformly continuous maps are pointwise equal, then they have the same minimal modulus and yield the same morphism in the category of C-spaces.

To address the issues

We developed a few approaches:

1. Postulate (funext)
2. Use setoids
3. Add a probe axiom
4. Postulate (funext) in a computationally irrelevant field
5. Postulate the double negation of (funext)
6. Porting the development from MLTT to Cubical Type Theory

All of them have been implemented in Agda.

Construction by postulating (funext)

This is of course the easiest approach.

It gives a clean formalization of our informal development.

But we lose computational content:

Our Agda implementation can compute the modulus of uniform continuity of the constant map

$$\lambda\alpha.2.$$

But it can't compute the one of

$$\lambda\alpha.\,\mathrm{if}\ \ \mathrm{true}\ \ 2\ 2.$$

The normal form of its modulus produced by Agda has more than 200 lines (before my recent improvement it was more than 300 lines).

Construction by using setoids

Main adjustments:

- Any uniformly continuous map $t\colon \mathbf{2}^{\mathbb{N}} \to \mathbf{2}^{\mathbb{N}}$ is extensional, *i.e.* satisfying

$$\Pi(\alpha, \beta : \mathbf{2}^{\mathbb{N}}).\ (\Pi(i : \mathbb{N}).\alpha_i = \beta_i) \to \Pi(i : \mathbb{N}).(t\alpha)_i = (t\beta)_i.$$

- A C-space is a setoid X equipped with a C-topology P consisting of extensional probes.
- The C-topology P has to additionally satisfy

$$\Pi(p, q\colon \mathbf{2}^{\mathbb{N}} \to X).\ (\Pi(\alpha : \mathbf{2}^{\mathbb{N}}).p\alpha \sim_X q\alpha) \to p \in P \to q \in P.$$

- Continuous maps of C-spaces now have to be extensional.
  This defines a notion of equality of continuous maps that ignores continuity witnesses.

But it makes the constructions and proofs tedious, long and unreadable

Construction by adding a probe axiom

Motivated by the previous approach, we added the following probe axiom

$$\Pi(p, q\colon \mathbf{2}^{\mathbb{N}} \to X).\ (\Pi(\alpha\colon \mathbf{2}^{\mathbb{N}}).p\alpha = q\alpha) \to p \in P \to q \in P.$$

It avoids some uses of (funext) and hence preserves some amount of computational content.

However, (funext) is still needed in some constructions and proofs.

The canonicity of the theory is potentially destroyed.

Construction by postulating (funext) within a relevant field

We conjectured that (funext) is only used in computationally irrelevant contexts, and made use of Agda's irrelevant fields to verify it, and postulated (funext) within such an irrelevant field.

Then we observed that a stronger form of the additional probe axiom is needed:

$$\Pi(p, q : \mathbf{2}^{\mathbb{N}} \to X).\ (\Pi(\alpha : \mathbf{2}^{\mathbb{N}}).[p\alpha = q\alpha]) \to p \in P \to q \in P$$

where $[-]$ is an irrelevant field.

It computes. But one drawback is that it requires non-standard extension of type theory.

Construction by postulating $\neg\neg$(funext)

The only property of irrelevant fields we used is that they form a monad T satisfying $T\emptyset \to \emptyset$.

Double negation is the final such monad.

Hence we postulate $\neg\neg$(funext).

Similarly, we need a stronger form of the additional probe axiom

$$\Pi(p, q : \mathbf{2}^{\mathbb{N}} \to X).\ (\Pi(\alpha : \mathbf{2}^{\mathbb{N}}).\neg\neg(p\alpha = q\alpha)) \to p \in P \to q \in P.$$

Advantages:

- Though the model produced here is not the same as the original, it provides the same interpretation to simple (and dependent) types.
- It does not destroy computational content of the development! (The postulated axiom is consistent and in negative form.)

Porting the original development to Cubical Type Theory

Cubical Type Theory[1] allows one to directly manipulate n-dimensional cubes:

- it has path types with abstraction and application,
- (funext) has a very simple proof,
- Voevodski's univalence axiom is also provable, and
- some higher inductive types are available.

Now Agda has an experimental cubical mode. In Cubical Agda, path types and identity types are equivalent. Hence, theoretically one can derive (funext) for identity types from the provable one for paths. But it's not available yet.

To port our original, non-computing development to Cubical Agda, we

- replace (inductive) identity types by path identity types,
- replace usages of pattern matching on identity types by J, and
- do a little extra work to avoid the bugs.

[1] C. Cohen, T. Coquand, S. Huber, and A. Mörtberg, *Cubical Type Theory: a constructive interpretation of the univalence axiom* (2016). To appear in the proceedings of TYPES 2015.

Sample computations of least moduli of uniform continuity

We can compute the least moduli of uniform continuity of T-definable functions $\mathbf{2}^{\mathbb{N}} \to \mathbb{N}$ as follows:

1. Let a closed T-term F of type $(\mathbb{N} \to \mathbf{2}) \to \mathbb{N}$ be given.
2. Its interpretation is a continuous map $[\![F]\!]^{\mathrm{m}} : \mathbf{2}^{\mathbb{N}} \to \mathbb{N}$.
3. Using the fan functional, we get its least modulus of uniform continuity, that is a closed Agda term
$$\mathrm{pr}_1(\mathrm{fan})[\![F]\!]^{\mathrm{m}} : \mathbb{N}.$$
4. Then ask Agda to evaluate it!

Summary

- Some issues occur in the type-theoretic development of our $\mathbb{C}$-space model due to the lack of (funext).
- Directly postulating (funext) destroys the computation.
- We developed a few other approaches to obtain a computing implementation.
- We ported the original, non-computing development to Cubical Agda, and it successfully computes.
- Evaluation of paths in Cubical Agda is still very inefficient.