Function extensionality

Samuel Mimram

2025

École polytechnique

Function extensionality

We can now construct useful non-trivial paths, such as $\mathsf{Bool} = \mathsf{Bool}$.

However, we cannot seem to be able to construct non-trivial paths between functions excepting simple ones.

Recall that there is a natural notion for comparing functions $f\ g : A \to B$:

$$f \sim g \quad \hat{=} \quad (x : A) \to f\,x = g\,x$$

Function extensionality

If we consider the negation

$$\mathsf{not} : \mathsf{Bool} \to \mathsf{Bool}$$

we can show

$$\mathsf{not} \circ \mathsf{not} \sim \mathsf{id}_{\mathsf{Bool}}$$

by reasoning by case analysis

- $\mathsf{not}\,(\mathsf{not}\ \mathsf{false}) = \mathsf{false}$
- $\mathsf{not}\,(\mathsf{not}\ \mathsf{true}) = \mathsf{true}$

But we cannot show

$$\mathsf{not} \circ \mathsf{not} = \mathsf{id}_{\mathsf{Bool}}$$

More generally, we would like to show that $f = g$ is the same as $f \sim g$.

Function extensionality

The only equalities we can easily show are the definitional ones.

For instance, for $f : A \to B$, we have

$$\mathsf{id} \circ f \mathrel{\hat{=}} (\lambda x.\, \mathsf{id}\, (f\, x)) \mathrel{\hat{=}} (\lambda x. f\, x) \mathrel{\hat{=}} f$$

and thus

$$\mathsf{refl} \quad : \quad \mathsf{id} \circ f = f$$

Equality vs homotopy

Given $f\ g : A \to B$, we have a canonical map

$$\mathsf{happly} : (f = g) \to (x : A) \to f\,x = g\,x$$

defined by path induction by

$$\mathsf{happly}\ \mathsf{refl} \quad \hat{=} \quad \lambda x.\,\mathsf{refl}$$

We will show that this map is an equivalence, in particular we will have an inverse

$$\mathsf{funext} : ((x : A) \to f\,x = g\,x) \to f = g$$

Equality vs homotopy

The equivalence $(f = g) \simeq (f \sim g)$ can be read as the fact that we have

- an elimination rule:

$$\mathsf{happly} : (f = g) \to (f \sim g)$$

- an introduction rule:

$$\mathsf{funext} : (f \sim g) \to (f = g)$$

- a computation rule: for $\alpha : f \sim g$ and $x : A$,

$$\mathsf{happly}\,(\mathsf{funext}\ \alpha)\,x = \alpha\,x$$

- a uniqueness rule: for $p : f = h$,

$$\mathsf{funext}\,(\lambda x.\,\mathsf{happly}\ p\,x) = p$$

Function extensionality

There is no simple way of constructing the map

$$\mathsf{funext} : (f \sim g) \to (f = g)$$

Intuitively, we should be able to do the following:

$$f \mathrel{\hat{=}} \lambda x.f\,x = \lambda x.g\,x \mathrel{\hat{=}} g$$

where the middle step is something like

$$\mathsf{ap}\,(\lambda y.\lambda x.y)\,(h\,x)$$

but without proper α-conversion...

Alternative plan

The idea is to show the from [Lic14] is to show the property for all f and g at once, i.e.

$$\Sigma(f\ g : A \to B).(f = g) \quad \simeq \quad \Sigma(f\ g : A \to B).(f \sim g)$$

If we consider the types

$$\begin{aligned} \mathsf{Path}(A) &\quad \hat{=} \quad \Sigma(x\ y : A).(x = y) \\ \mathsf{Homotopy}(A, B) &\quad \hat{=} \quad \Sigma(f\ g : A).(f \sim g) \end{aligned}$$

we want to show

$$\mathsf{Path}(A \to B) \quad \simeq \quad \mathsf{Homotopy}(A, B)$$

which we can do by

$$\mathsf{Path}(A \to B) \quad \simeq \quad A \to B \quad \simeq \quad A \to \mathsf{Path}(B) \quad \simeq \quad \mathsf{Homotopy}(A, B)$$

Alternative plan

In particular, we obtain a function

$$\mathsf{Funext} \quad : \quad \mathsf{Homotopy}(A, B) \quad \to \quad \mathsf{Path}(A \to B)$$

which to a homotopy associates a path

If we manage to show that the endpoints are respected, this induces

$$\mathsf{funext} \quad : \quad (f\ g : A \to B) \quad \to \quad f \sim g \quad \to \quad f = g$$

by

$$\mathsf{funext}\ f\ g\ \alpha \quad \hat{=} \quad \mathsf{snd}(\mathsf{Funext}(f, g, \alpha))$$

as required.

Equivalences

The equivalences are easily shown. We have

$$\begin{array}{rcl} \mathsf{Homotopy}(A,B) & \hat{=} & \Sigma(f\ g : A \to B).\Pi(x : A).(f\ x = g\ x) \\ & \simeq & \Pi(x : A).\Sigma(b_1\ b_2 : B).(b_1 = b_2) \\ & \hat{=} & A \to \mathsf{Path}(B) \end{array}$$

and

$$\mathsf{Path}(A) \quad \hat{=} \quad \Sigma(x : A).\Sigma(y : B).(x = y) \quad \simeq \quad \Sigma(x : A).1 \quad \simeq \quad A$$

(i.e. we contract paths on their source) and thus

$$\mathsf{Path}(A \to B) \quad \simeq \quad A \to B \quad \simeq \quad A \to \mathsf{Path}(B) \quad \simeq \quad \mathsf{Homotopy}(A,B)$$

from which we deduce

$$\mathsf{Funext} \quad : \quad \mathsf{Homotopy}(A,B) \quad \to \quad \mathsf{Path}(A,B)$$

Where did we use univalence???

Equivalences

Subtle point: in order to deduce $(A \to B) \simeq (A \to \mathsf{Path}(B))$ from $B \simeq \mathsf{Path}(B)$, we have used

Lemma
If $B \simeq B'$ then $(A \to B) \simeq (A \to B')$.

Proof (attempt).
Writing $g : B \simeq B'$, we define

$$\phi : (A \to B) \to (A \to B') \qquad f \mapsto g \circ f$$

$$\psi : (A \to B') \to (A \to B) \qquad f \mapsto g^{-1} \circ f$$

and we have

$$\psi \circ \phi(f) \doteq g^{-1} \circ (g \circ f) \doteq (g^{-1} \circ g) \circ f = \mathsf{id} \circ f \doteq f$$

Excepting that we do not have $g^{-1} \circ g = \mathsf{id}$ but only $g^{-1} \circ g \sim \mathsf{id}$... □

The proof of the lemma is rather the following one:

Lemma ([Uni13, Lemma 4.9.2])
If $B \simeq B'$ *then* $(A \to B) \simeq (A \to B')$.

Proof.
By univalence, from $e : B \simeq B'$ we deduce $\mathsf{ua}\ e : B = B'$ and thus

$$\mathsf{ap}\,(\lambda X.A \to X)\,(\mathsf{ua}\ e) \quad : \quad (A \to B) = (A \to B') \qquad \square$$

This proof is not entirely satisfactory, we would rather show that the previous

$$\phi : (A \to B) \to (A \to B')$$

is an equivalence, which can be done by equivalence induction (which requires univalence), see the labs. This works because we have $\mathsf{id} \circ f \mathrel{\hat{=}} f$!

Preservation of endpoints

Suppose given $f\,g : A \to B$, a homotopy $\alpha : f \sim g$ can be seen as an element

$$H \hat{=} (f, g, \alpha) \quad : \quad \mathsf{Homotopy}(A, B)$$

Consider the image of H under the equivalence

$$\begin{array}{ccccccc} \mathsf{Homotopy}(A,B) & \simeq & A \to \mathsf{Path}(B) & \simeq & A \to B & \simeq & \mathsf{Path}(A \to B) \\ (f, g, \alpha) & & \lambda x.(f\,x, g\,x, \alpha(x)) & & \lambda x.f\,x & & \lambda f.(f, f, \mathsf{refl}) \end{array}$$

It is the same as the image of

$$H_0 \hat{=} (f, f, \mathsf{id}) \quad : \quad \mathsf{Homotopy}(A, B)$$

By injectivity, we thus have $H_0 = H$ and thus $f = g$.

We have shown $(f \sim g) \to (f = g)$ as desired.

Function extensionality

We have shown **function extensionality**:

Theorem
Given $f\ g : A \to B$, *we have*

$$\mathsf{funext} \quad : \quad (f \sim g) \quad \to \quad (f = g)$$

This can be refined to

Theorem
Given $f\ g : A \to B$, we have

$$\mathsf{funext} \quad : \quad (f \sim g) \quad \simeq \quad (f = g) \quad : \quad \mathsf{happly}$$

Proof.
We have constructed an equivalence

$$\Sigma((f,g) : (A \to B)^2).(f \sim g) \quad \simeq \quad \Sigma((f,g) : (A \to B)^2).(f = g)$$

By previous theorem, this equivalence comes from a family of maps

$$\Sigma((f,g) : (A \to B)^2) \to (f \sim g) \to (f = g)$$

which ares equivalences (equivalences between total spaces are fiberwise so). □

Dependent function extensionality

We would like to generalize function extensionality

$$(f = g) \quad \simeq \quad f \sim g$$

to dependent functions

$$f\ g \quad : \quad (x : A) \quad \to \quad B\,x$$

The previous proof does not easily generalize.

But we can show that the non-dependent version implies the dependent one.

Dependent function extensionality

The plan is that

- **non-dependent function extensionality**
- implies **weak function extensionality**
- which implies (dependent) **function extensionality**

Weak function extensionality [Uni13, Definition 4.9.1]

The **weak function extensionality** principle is that for every $A : \mathcal{U}$ and $B : A \to \mathcal{U}$ we have

$$((x : A) \to \mathsf{isContr}(B\,x)) \to \mathsf{isContr}((x : A) \to B\,x)$$

This can be seen as a stronger form of the axiom of choice

$$((x : A) \to \|B\,x\|_{-1}) \to \|(x : A) \to B\,x\|_{-1}$$

Weak function extensionality

Theorem
The weak function extensionality principle holds:

$$((x : A) \to \mathsf{isContr}(B\,x)) \to \mathsf{isContr}((x : A) \to B\,x)$$

Proof (attempt).
Suppose $(x : A) \to \mathsf{isContr}(B\,x)$. For each $x : A$, we have an element $b_x : B\,x$.

We want to show that $(x : A) \to B\,x$ can be contracted to $b \mathrel{\hat{=}} \lambda x.b_x$.

Given $f : (x : A) \to B\,x$, we want to show $b_x = f$ from $(x : A) \to b_x = f\,x$.

But this is precisely dependent funext... □

Weak function extensionality

Theorem
The weak function extensionality principle holds:

$$((x : A) \to \mathsf{isContr}(B\,x)) \to \mathsf{isContr}((x : A) \to B\,x)$$

Proof.
Suppose $(x : A) \to \mathsf{isContr}(B\,x)$. Since $B\,x$ is contractible, we have $B\,x = 1$ and we are left with showing

$$\mathsf{isContr}((x : A) \to 1)$$

This can be contracted to $f : \lambda x.\star$. Namely, given $g : A \to 1$, we have $f\,x = g\,x$ and thus $f = g$ by non-dependent funext. □

Function extensionality

Theorem
We finally have ***function extensionality****:*

$$\mathsf{funext} \quad : \quad (f\ g : (x : A) \to B\,x) \quad \to \quad ((x : A) \to f\,x = g\,x) \quad \to \quad f = g$$

Proof.
Suppose given f and g such that $f \sim g$. f induces a function

$$\begin{aligned} f' \quad : \quad & (x : A) \to \Sigma(y : B).(f\,x = y) \\ & x \mapsto (f\,x, \mathsf{refl}) \end{aligned}$$

and similarly for g. The type $\Sigma(y : B).(f\,x = y)$ being contractible (singleton), by weak funext we have $\mathsf{isContr}((x : A) \to \Sigma(y : B).(f\,x = y))$ thus $f' = g'$ and thus $f = g$.

□

Function extensionality

With a bit more work one can show

Theorem ([Uni13, Theorem 4.9.5])
*We have **function extensionality**, i.e. an equivalence*

$$\mathsf{funext} \quad : \quad ((x : A) \to f\,x = g\,x) \quad \simeq \quad (f = g) \quad : \quad \mathsf{happly}$$

for $f\ g : (x : A) \to B\,x$.

Bibliography i

[Lic14] Dan Licata.
Another proof that univalence implies function extensionality.
Homotopy Type Theory blog post, 2014.
https://homotopytypetheory.org/2014/02/17/another-proof-that-univalence-implies-function-extensionality/.

[Uni13] The Univalent Foundations Program.
***Homotopy Type Theory: Univalent Foundations of Mathematics*.**
Institute for Advanced Study, 2013.
https://homotopytypetheory.org/book, arXiv:1308.0729.