Omega Constancy and Truncations

Nicolai Kraus

Functional Programming Lab, University of Nottingham

04/07/14
Consider a type $X$

... and two points $a, b : X$

$\Rightarrow$ we get a new type $a = b$.

Elements are paths, e.g.  
$p : a = b$ or $q : a = b$

$\Rightarrow$ we get a new type  
$p = q$, the inhabitants of which are 2-paths.

Definition: if $n$ such iterations always lead to a type that is isomorphic to the unit type, we say that $X$ is an $(n-2)$-type or $(n-2)$-truncated.
Propositional Truncation

The Truncation Monad

\((-1)\)-types are called *propositions*. They have the property that all their inhabitants are equal.

A type theory can have a monad \(\|\_\|\) which turns any type into a proposition (which says that this type is inhabited).

Concretely:

- \(\|A\|\) is propositional
- \(A \rightarrow \|A\|\)
- If \(B\) is propositional, then

\[
(A \rightarrow B) \rightarrow (\|A\| \rightarrow B).
\]

(That implies \(\|A\| \rightarrow (A \rightarrow \|B\|) \rightarrow \|B\|\).)
Goal: a function \( \|A\| \rightarrow B \). How do we get it?

If \( B \) is a proposition, then \( f : A \rightarrow B \) is enough. Interpretation:

“\( f(a) \) does not depend on the concrete choice of \( a : A \) because \( B \) does not have distinguishable elements anyway.”

This suggests that \( f \) should be constant (if we want to drop the condition on \( B \)).
But what is “constant”? 

First try:

\[
\text{const}_f \equiv \forall (x, y : A). f(x) = f(y)
\]

Indeed, we can prove:

Theorem

\[
(\sum_{f : A \to B} \text{const}_f) \simeq (\|A\| \to B)
\]

if \( B \) is a 0-type.
But look at this:
A function $1 \rightarrow \text{Torus}$ with a proof $\text{const}_f$:

We have to ask for more than that!
Coherence Conditions

The paths have to “fit together”.

Given \( f : A \rightarrow B \) and \( c : \text{const}_f \), define

\[
\text{coh}_{f,c} \equiv \forall (x, y, z : A). \, c(x, y) \cdot c(y, z) = c(x, z)
\]

We can prove:

**Theorem**

\[
(\sum_{f:A\rightarrow B} \sum_{c:\text{const}_f} \text{coh}_{f,c}) \simeq (\|A\| \rightarrow B)
\]

if \( B \) is a 1-type.
The General Case

- If we do not know anything about $B$, we need an “infinite $\Sigma$-type”.
- This can be done in a theory with certain Reedy-limits.
- Corollary: If $B$ is an n-type, the first $n$ conditions are sufficient.
- This case can be formalised in a proof assistant, for any $n$.
- Main contribution:
  
  a generalised universal property of the prop. truncation.

- P. Capriotti and I try to do the same for higher truncation, which requires a very different approach.
Questions?

Thank you!