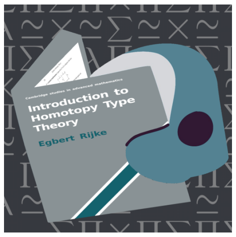

Worksheet 9 (Solved)

HoTTEST Summer School 2022

The HoTTEST TAs

5 August 2022

1 ($\star$)

(a) Show that $(A+B) \to C$ is equivalent to $(A \to C) \times (B \to C)$

We know that (up to currying)

$$\mathtt{ind}_{A+B} : (A \to C) \times (B \to C) \to ((A+B) \to C)$$

which gives us one direction of our equivalence. To go the other way, we compose with the inclusions

$$\lambda(f : (A+B) \to C).(f \circ \mathtt{inl}, f \circ \mathtt{inr}) : ((A+B) \to C) \to (A \to C) \times (B \to C)$$

It suffices to show that these functions compose to the identity. One direction follows from computation rules (in what follows, $p = (p_1, p_2)$, where $p_1 : A \to C$ and $p_2 : B \to C$).

$$\begin{aligned}
(\lambda f.(f \circ \mathtt{inl}, f \circ \mathtt{inr})) \circ \mathtt{ind}_{A+B} &\doteq \lambda(p : (A \to C) \times (B \to C)). \\
&\qquad ((\lambda f.(f \circ \mathtt{inl}, f \circ \mathtt{inr}))(\mathtt{ind}_{A+B}(p))) \\
&\doteq \lambda p.(\mathtt{ind}_{A+B}(p) \circ \mathtt{inl}, \mathtt{ind}_{A+B}(p) \circ \mathtt{inr}) \\
&\doteq \lambda p.(\lambda a.\mathtt{ind}_{A+B}(p)(\mathtt{inl}(a)), \lambda b.\mathtt{ind}_{A+B}(p)(\mathtt{inr}(b))) \\
&\doteq \lambda p.(\lambda a.p_1(a), \lambda b.p_2(b)) \\
&\doteq \lambda p.(p_1, p_2) \\
&\doteq \lambda p.p
\end{aligned}$$

while the other requires function extensionality. Indeed, computation rules tell us

$$\mathtt{ind}_{A+B} \circ (\lambda f.(f \circ \mathtt{inl}, f \circ \mathtt{inr})) \doteq \lambda g.\mathtt{ind}_{A+B}(g \circ \mathtt{inl}, g \circ \mathtt{inr})$$

We want to show this is equal to $\lambda g.g$, and by function extensionalilty it suffices to show, for each g, that $\mathtt{ind}_{A+B}(g \circ \mathtt{inl}, g \circ \mathtt{inr}) = g$.

But these are *also* functions! So by function extensionality *again* it suffices to show for each $x : A+B$ that $\mathtt{ind}_{A+B}(g \circ \mathtt{inl}, g \circ \mathtt{inr})(x) = g(x)$, but this is exactly the computation rule for $\mathtt{ind}_{A+B}$.
Indeed, if $x \doteq \mathtt{inl}(a)$, then,

$$\begin{aligned}\mathtt{ind}_{A+B}(g \circ \mathtt{inl}, g \circ \mathtt{inr})(x) &\doteq \mathtt{ind}_{A+B}(g \circ \mathtt{inl}, g \circ \mathtt{inr})(\mathtt{inl}(a)) \\ &\doteq (g \circ \mathtt{inl})(a) \\ &\doteq g(\mathtt{inl}(a)) \\ &\doteq g(x)\end{aligned}$$

The case of $x \doteq \mathtt{inr}(b)$ is similar.

(b) Formulate and prove a dependent version of part (a).

The dependent version should tell us that $\prod_{a:A} C(\mathtt{inl}(a)) \times \prod_{b:B} C(\mathtt{inr}(b))$ is equivalent to $\prod_{x:A+B} C(x)$.
Again, we see that $\mathtt{ind}_{A+B}$ and $\lambda f.(f \circ \mathtt{inl}, f \circ \mathtt{inr})$ provide the required inverse maps, and in fact the same proof from part (a) works here too.

For those in the know, the **universal property of the coproduct** says that maps $(A+B) \to C$ should be *uniquely determined by* a pair of maps $A \to C$ and $B \to C$.

Inside of type theory, this categorical property comes from the function $\mathtt{ind}_{A+B}$, which constructs a map $(A+B) \to C$ for each pair of maps $A \to C$ and $B \to C$. The purpose of this exercise is to show that (under `funext`!) such a map is uniquely determined, so that the coproduct in type theory really is a coproduct in the categorical sense[1].

2 (⋆⋆)

If $B(x)$ is a proposition (resp. is contractible) for each $x : A$, show that $\prod_{x:A} B(x)$ is a proposition (resp. is contractible).

Say $f, g : \prod_{x:A} B(x)$. We want to show that $f = g$.
By function extensionality, it suffices to check that $fx = gx$ for each x.
But this is definitely true, since fx and gx live in $B(x)$, which is a proposition!

Provided each $B(x)$ is contractible, with center of contraction c_x, then $f(x) = c_x$ witnesses $\prod_{x:A} B(x)$ as inhabited, so we're done since inhabited propositions are contractible.

[1]For those *really* in the know, the equivalence we've constructed shows that this is true in the ∞-categorical sense. Indeed, the usual way of stating the universal property of ∞-coproducts is by saying for all objects C the mapping spaces $(A \to C) \times (B \to C)$ and $(A+B) \to C$ are equivalent as simplicial sets.

3 $(\star\star\star)$

(a) Show that $\texttt{isContr}(A)$ is a proposition.

Let (a_1, p_1) and (a_2, p_2) have type $\texttt{isContr}(A)$[a].
We know $p_1(a_2) : a_1 = a_2$, so if we can show that $p_1(a_2)_* p_1 = p_2$ we'll be done. By function extensionality, it suffices to check that, for each $x : A$, $(p_1(a_2)_* p_1)(x) = p_2(x)$ as terms of type $a_2 = x$. But since A is contractible we know $a_2 = x$ is too (by a corollary from Lecture 8), making these two terms equal.

[a] recall this is an abbreviation for $\sum_{a:A} \prod_{x:A} a = x$

(b) Show that (for every k, for every type A) $\texttt{isTrunc}_k(A)$ is a proposition.

We induct on k.
When $k = -2$, truncation means contractibility, so this was part (a).
Assuming the claim for some fixed k, we want to show that $\texttt{isTrunc}_{k+1}(A)$ is a proposition for all A. That is, we want to show $\prod_{x,y:A} \texttt{isTrunc}_k(x = y)$ is a proposition. But, by induction, $\texttt{isTrunc}_k(x = y)$ is a proposition, so the claim follows by problem 2.

4 $(\star\star\star)$

Recall that (for some $f : A \to B$) we want $\texttt{isEquiv}(f)$ to have three properties:

1. $\sum_{g:B\to A}(g \circ f = \mathrm{id}_A) \times (f \circ g = \mathrm{id}_B)$ implies $\texttt{isEquiv}(f)$
2. $\texttt{isEquiv}(f)$ implies $\sum_{g:B\to A}(g \circ f = \mathrm{id}_A) \times (f \circ g = \mathrm{id}_B)$
3. $\texttt{isEquiv}(f)$ is a proposition

It should be obvious why we want the first two properties. They tell us that $\texttt{isEquiv}(f)$ should capture the intuitive notion of an equivalence between the types A and B, in the sense that f and its inverse g let us convert back and forth between A and B with no loss of information.

It may be less obvious why we want the third property. Hopefully it's starting to make sense that propositions are particularly simple types, which are particularly easy to handle. You might imagine that, for an idea as fundamental as equivalence, we would want it to be as easy to handle as possible. After all, we'll be using constantly! For this (admittedly informal) reason, it should be believable that requiring $\texttt{isEquiv}(f)$ to be a proposition will simplify later parts of the theory.

Now, inspired by conditions 1 and 2, we might (naively) try to define $\texttt{isEquiv}$ to simply *be* $\sum_g(gf = \mathrm{id}) \times (fg = \mathrm{id})$. Unfortunately, this fails condition 3.

One can show that if $\sum_g (gf = \mathrm{id}) \times (fg = \mathrm{id})$ is inhabited, then it is equivalent to $\prod_{x:A} x = x$ (this is Lemma 4.1.1 in the HoTT book), but this is not a mere proposition for $A \doteq S^1$ (do you see why?).

Then the race is on to find a type satisfying our three properties! We've already seen a handful of options and shown them to be equivalent to each other. For more details, see chapter 4 of the HoTT book, but for now you should try the following exercise:

Define $\texttt{isContr}(f)$ to mean $\prod_{y:B} \texttt{isContr}(\texttt{fib}_f(y))$. Show that $\texttt{isContr}(f)$ satisfies property 3 above. If you're feeling particularly ambitious, you might show that it also satisfies properties 1 and 2.

We know that $\texttt{isContr}(\texttt{fib}_f(y))$ is a proposition for each y by Exercise 3(a). Then by Exercise 2 again we know that $\prod_{y:B} \texttt{isContr}(\texttt{fib}_f(y))$ is a proposition.

As for properties 1 and 2, I *wasn't* feeling particularly ambitious, but you can find an indirect proof in Chapter 4 of the HoTT book.