

## MEETING NOTES

1. DEC 1 2017

### 1.1. Chandrika's notes.

1.1.1. *Rushed Introduction and summary.* Consider the following diagram, where  $A, B, C$  are sets, and  $D$  is the homotopy pushout of  $B$  and  $C$  along  $A$ .

$$\begin{array}{ccc} A & \xrightarrow{f} & B \\ \downarrow g & & \downarrow \\ C & \longrightarrow & D \end{array}$$

We would like to show that  $D$  is a homotopy set as well using the encode-decode method. Therefore, we defined  $\text{code} : D \rightarrow D \rightarrow \text{Type}$  so that  $\text{code}(x, y)$  is a proposition.

Amelia spoke of a picture to think of for  $D$ . This is my understanding of the picture.  $D$  is the following topological space (“the double mapping cylinder”).  $(B \cup A \times [0, 1] \cup C) / \sim$ , where  $\sim$  is defined by:

- $(a, 1) \sim b \iff f(a) = b, \forall a \in A, b \in B$
- $(a, 0) \sim c \iff g(a) = c, \forall a \in A, c \in C$

We have attempted to define  $\text{code}(x, y)$  to be the homotopy structure of the paths from  $x$  to  $y$  in this double mapping cylinder.

In Utah, we successfully defined  $\text{encode}$  and began to define  $\text{decode} : \text{code} \rightarrow \text{Id}_D$ . We defined  $\text{decode}(\text{code}(x, y))$  for all  $x, y \in (B \cup C) \subset D$ .

1.1.2. *What's next?* It remains to define  $\text{ap}_{\text{decode}}$ . I think that these are the types of the various parts of  $\text{ap}_{\text{decode}}$ .

$$\begin{aligned}
& \text{ap}_{\text{decode}}\left(\text{ap}_{\text{code}}(\text{in}b, \text{glue}(a))\right) : \left(\text{decode}\left(\text{code}(\text{in}b, \text{in}f(a))\right) = \text{decode}\left(\text{code}(\text{in}b, \text{in}rg(a))\right)\right) \\
& \text{ap}_{\text{decode}}\left(\text{ap}_{\text{code}}(\text{glue}(a), \text{in}b)\right) : \left(\text{decode}\left(\text{code}(\text{in}f(a), \text{in}b)\right) = \text{decode}\left(\text{code}(\text{in}lg(a), \text{in}b)\right)\right) \\
& \text{ap}_{\text{decode}}\left(\text{ap}_{\text{code}}(\text{glue}(a), \text{in}rc)\right) : \left(\text{decode}\left(\text{code}(\text{in}f(a), \text{in}rc)\right) = \text{decode}\left(\text{code}(\text{in}lg(a), \text{in}rc)\right)\right) \\
& \text{ap}_{\text{decode}}\left(\text{ap}_{\text{code}}(\text{in}rc, \text{glue}(a))\right) : \left(\text{decode}\left(\text{code}(\text{in}rc, \text{in}f(a))\right) = \text{decode}\left(\text{code}(\text{in}rc, \text{in}rg(a))\right)\right) \\
& \text{ap}_{\text{decode}}\left(\text{ap}_{\text{code}}(\text{glue}(a), \text{glue}(a))\right) : \text{isContr}\left(\text{decode}\left(\text{code}(\text{in}f(a'), \text{in}f(a))\right) = \text{decode}\left(\text{code}(\text{in}rg(a'), \text{in}rg(a))\right)\right)
\end{aligned}$$

2. DEC 8 2017

## 2.1. Chandrika's notes.

### 2.1.1. Questions to think about and things to do for next time.

- (1) Internalize and share if possible.
- (2) Do we have to deal with a push out when defining  $\text{ap}_{\text{code}}(b, b)$ ? Understand why/why not.
  - My memory is that while  $\text{code}(b, b)$  is a push out, this is not an issue because for  $\text{ap}_{\text{code}}$  the relevant situation is when  $b$  is in the image of  $A$ .
  - What happens to  $\text{code}(b, b)$  when we restrict to  $b$  being in the image of  $A$ ?
- (3) Why isn't defining  $\text{decode}$  as simple as defining  $\text{encode}$ ? Why can't we use path induction like we did for  $\text{encode}$ ?
  - Refer to Michael Schulman's comment: "The point is that the domain of  $\text{code}$  is a \*colimit\* of infinity-groupoids, so to define  $\text{code}$  we have to give a coherently commuting cocone. The part that looks like "defining it on objects" is defining the \*morphisms\* in that cone, each of which is automatically an infinity-groupoid morphism. The part that looks like "defining  $\text{ap}_{\text{code}}$ " is showing that that cone commutes (up to coherent homotopy), which amounts to "defining  $\text{ap}_{\text{code}}$ " \*only\* on the "new" higher morphisms that are "glued in universally" by the colimit. It's a little hard to distinguish in this case because the infinity-groupoids going into the colimit are all discrete, so they don't have any higher morphisms themselves, but in general there could be some, and we wouldn't have to define  $\text{ap}$  on those; it would come for free once we defined the morphisms in the cone.

For  $\text{encode}$ , we are using the universal property of equality types, which is essentially a colimit-like construction. But there are no homotopies being glued in, and hence no "ap" cases to do.

For  $\text{decode}$ , we again use the universal property of the domain of  $\text{code}$ , so we have ap cases to deal with.

I hope this helps. You may also find it useful to look at chapters 6 and 8 of the HoTT Book.”

### 3. DEC 19 2017

Goal for next time: to understand our definition for  $\text{ap}_{\text{code}}$  and begin to define  $\text{ap}_{\text{decode}}$ .