A PUSHOUT OF SETS OVER AN INJECTION IS A SET

1. The 'pushout is a set' theorem presented

Theorem 1. *Given mere sets* $\mathtt{A}$, $\mathtt{B}$, *and* $\mathtt{C}$ *configured in a span*

$$\mathtt{C} \xleftarrow{g} \mathtt{A} \xrightarrow{f} \mathtt{B}$$

where f *is monic, then the pushout* $\mathtt{B} +_{\mathtt{A}} \mathtt{C}$ *is a mere set.*

DANIEL: In the following paragraph, we define $\mathtt{P}$ which encodes those id-types of $\mathtt{B} +_{\mathtt{A}} \mathtt{C}$ that are propositions. The pushout $\mathtt{B}+_{\mathtt{A}}\mathtt{C}$ is a set when $\mathtt{P}$ contains all id-types in the pushout. So we define $\mathtt{Q}$ to encode all id-types of $\mathtt{B} +_{\mathtt{A}} \mathtt{C}$, which means that showing that $\mathtt{B} +_{\mathtt{A}} \mathtt{C}$ is a set is the same as showing that $\mathtt{Q}$ is equivalent to $\mathtt{P}$.

Denote the canonical pushout maps as $\mathtt{inl} \colon \mathtt{B} \to \mathtt{B} +_{\mathtt{A}} \mathtt{C}$ and $\mathtt{inr} \colon \mathtt{C} \to \mathtt{B} +_{\mathtt{A}} \mathtt{C}$. Consider the types

$$\mathtt{P} :\equiv \{(x, y) \colon (\mathtt{B} +_{\mathtt{A}} \mathtt{C}) \times (\mathtt{B} +_{\mathtt{A}} \mathtt{C}) | x =_{\mathtt{B}+_{\mathtt{A}}\mathtt{C}} y \text{ is a mere proposition}\}$$

and

$$\mathtt{Q} :\equiv (\mathtt{B} +_{\mathtt{A}} \mathtt{C}) \times (\mathtt{B} +_{\mathtt{A}} \mathtt{C}).$$

To prove the theorem, it is sufficient to show that the inclusion $\mathtt{P} \hookrightarrow \mathtt{Q}$ is an equivalence.

Lemma 1. *The inclusion* $\mathtt{P} \to \mathtt{Q}$ *is an embedding.*

Proof. We need to show that, for each $(x_0, y_0) \colon \mathtt{Q}$, the homotopy fibre

$$\mathtt{F} :\equiv \sum_{(x,y) \colon \mathtt{P}} (x, y) =_{\mathtt{Q}} (x_0, y_0)$$

is a mere proposition. Any $p \colon \mathtt{F}$, is a pair of paths $p_x \colon x =_{\mathtt{B}+_{\mathtt{A}}\mathtt{C}} x_0$ and $p_y \colon y =_{\mathtt{B}+_{\mathtt{A}}\mathtt{C}} y_0$. The existance of such a p implies that $(x_0, y_0) \colon \mathtt{P}$ by β-reducing (x, y) to (x_0, y_0). Therefore, $\mathtt{refl}_{(x_0,y_0)} \colon \mathtt{F}$. The cube of paths

$$\begin{tikzcd}
x \arrow[rr, dash] \arrow[dr, dash, "p_x"] \arrow[dd, dash, "p_x"'] & & y \arrow[dr, dash, "p_y"] \arrow[dd, dash] & \\
& x_0 \arrow[rr, dash, "p_y"] \arrow[dd, dash] & & y_0 \arrow[dd, dash] \\
x_0 \arrow[rr, dash] \arrow[dr, dash] & & y_0 \arrow[dr, dash] & \\
& x_0 \arrow[rr, dash] & & y_0
\end{tikzcd}$$

commutes trivially and is filled because $\mathtt{B} +_{\mathtt{A}} \mathtt{C}$ is at most a 1-type. Hence, we have that $p =_{\mathtt{F}} \mathtt{refl}_{(x_0,y_0)}$ and $\mathtt{F}$ is contractible.

DANIEL: Where do we use that 'being a proposition' is a proposition? □

To prove that $\mathtt{P} \hookrightarrow \mathtt{Q}$ is an equivalnece, it remains to show that $\mathtt{P}$ contains a point in every connected component of $\mathtt{Q}$. Note that every connected component of $\mathtt{B} +_{\mathtt{A}} \mathtt{C}$ contains a point of form $\mathtt{inl}b$ or $\mathtt{inr}c$. Therefore, if $\mathtt{inl}b =_{\mathtt{B}+_{\mathtt{A}}\mathtt{C}} x$ and $\mathtt{inr}c =_{\mathtt{B}+_{\mathtt{A}}\mathtt{C}} x$ are always mere propositions, then $\mathtt{P}$ contains a point in every component of $\mathtt{Q}$ as desired.

To actually do this, we employ the *encode-decode method*. The strategy is to introduce types

$$\prod_{x\colon \mathtt{B}+_{\mathtt{A}}\mathtt{C}} \mathtt{code}(\mathtt{inl}b, x) \text{ and } \prod_{x\colon \mathtt{B}+_{\mathtt{A}}\mathtt{C}} \mathtt{code}(\mathtt{inl}c, x).$$

After showing these are mere propositions—an easier task than proving the identity types in $\mathtt{B} +_{\mathtt{A}} \mathtt{C}$ are mere propositions—we build an equivalence between

$$\prod_{x\colon \mathtt{B}+_{\mathtt{A}}\mathtt{C}} \mathtt{code}(\mathtt{inl}b, x) \text{ and } \prod_{x\colon \mathtt{B}+_{\mathtt{A}}\mathtt{C}} \mathtt{inl}b =_{\mathtt{B}+_{\mathtt{A}}\mathtt{C}} x,$$

and likewise between

$$\prod_{x\colon \mathtt{B}+_{\mathtt{A}}\mathtt{C}} \mathtt{code}(\mathtt{inr}c, x) \text{ and } \prod_{x\colon \mathtt{B}+_{\mathtt{A}}\mathtt{C}} \mathtt{inr}c =_{\mathtt{B}+_{\mathtt{A}}\mathtt{C}} x.$$

1.1. Defining $\prod_{x\colon \mathtt{B}+_{\mathtt{A}}\mathtt{C}} \mathtt{code}(\mathtt{inl}b, x)$.

To define

$$\prod_{x\colon \mathtt{B}+_{\mathtt{A}}\mathtt{C}} \mathtt{code}(\mathtt{inl}b, x),$$

we use structural induction on x. Thus we compute the values

- $\mathtt{code}(\mathtt{inl}b, \mathtt{inl}b')$,
- $\mathtt{code}(\mathtt{inl}b, \mathtt{inr}c)$, and
- $\mathtt{code}(\mathtt{inl}b, \mathtt{glue}a)$.

We also show that the first two are mere propositions.

Define $\mathtt{code}(\mathtt{inl}b, \mathtt{inl}b')$ as the pushout of the span

$$\begin{tikzcd}
\sum_{a,a'\colon \mathtt{A}}(b =_{\mathtt{B}} fa) \times (b' =_{\mathtt{B}} fa') \times (b =_{\mathtt{B}} b') \arrow[r, "\alpha"] \arrow[d, "\beta"'] & (b =_{\mathtt{B}} b') \\
\sum_{a,a'\colon \mathtt{A}}(b =_{\mathtt{B}} fa) \times (b' =_{\mathtt{B}} fa') \times (ga =_{\mathtt{C}} ga') &
\end{tikzcd}
\qquad
\begin{tikzcd}
(p,q,r) \arrow[r, mapsto, "\alpha"] \arrow[d, mapsto, "\beta"'] & r \\
(p,q,\mathtt{ap}_g(\widehat{p^{-1};r;q})) &
\end{tikzcd}$$

where $\widehat{p^{-1};r;q}$ is the preimage of the path $p^{-1};r;q\colon fa =_{\mathtt{B}} fa'$ Its existance and uniqueness follows from the injectivity of f.

To distinguish these pushout maps from those associated to $\mathtt{B} +_{\mathtt{A}} \mathtt{C}$, we denote them by $\mathtt{inl}'$ and $\mathtt{inr}'$.

In the following lemma, we use the denotation $\mathtt{X} :\equiv \sum_{a,a':\mathtt{A}}(b =_{\mathtt{B}} fa) \times (b' =_{\mathtt{B}} fa') \times (ga =_{\mathtt{C}} ga')$, $\mathtt{Y} :\equiv \sum_{a,a':\mathtt{A}}(b =_{\mathtt{B}} fa) \times (b' =_{\mathtt{B}} fa') \times (b =_{/} Bb')$, and $\mathtt{Z} :\equiv b =_{\mathtt{B}} b'$.

Lemma 2. *The type* $\mathtt{code}(\mathtt{inl}b, \mathtt{inl}b')$ *is a mere proposition.*

Proof. The feet of the span are mere propositions; $\mathtt{Z}$ trivially so. Showing that $\mathtt{X}$ is also a mere proposition requires some work. Pick $(r, s, t)\colon (b =_{\mathtt{B}} fa) \times (b' =_{\mathtt{B}} fa') \times (ga =_{\mathtt{C}} ga')$ and $(r', s', t')\colon (b =_{\mathtt{B}} fa'') \times (b' =_{\mathtt{B}} fa''') \times (ga'' =_{\mathtt{C}} ga'''')$. Then $r^{-1}; r'\colon fa =_{\mathtt{B}} fa''$ and $s^{-1}; s'\colon fa' =_{\mathtt{B}} fa'''$, along with the injectivity of f, ensure the existance of paths $\hat{r}\colon a =_{\mathtt{A}} a''$ and $\hat{s}\colon a' =_{\mathtt{A}} a'''$. Beta-reduction provides that $(b =_{\mathtt{B}} fa) \times (b' =_{\mathtt{B}} fa') \times (ga =_{\mathtt{C}} ga')$ is equivalent to $(b =_{\mathtt{B}} fa'') \times (b' =_{\mathtt{B}} fa''') \times (ga'' =_{\mathtt{C}} ga'''')$, both of which are contractable because $\mathtt{B}$ and $\mathtt{C}$ are mere sets. It follows that $(r, s, t) = (r', s', t')$.

If $\mathtt{code}(\mathtt{inl}b, \mathtt{inl}b')$ is not empty, then one of the span's feet must also be non-empty. This leads to three cases:

- $\mathtt{X}$ is empty and $\mathtt{Z}$ is non-empty. This forces $\mathtt{Y}$ to also be empty and so the lemma holds;
- $\mathtt{X}$ is non-empty and (Z) is empty. This again forces $\mathtt{Y}$ to be empty, so the lemma holds;
- $\mathtt{X}$ and $\mathtt{Z}$ are non-empty. It is quick to check that this forces $\mathtt{Y}$ to be non-empty. We dedicate the remainder of this proof to showing $\mathtt{code}(\mathtt{inl}b, \mathtt{inl}b')$ is a proposition in this case.

A point in the apex of the span provides a witness to $fa =_{\mathtt{B}} fa'$ which in turn provides a witness to $a =_{\mathtt{A}} a'$. Beta-reduction plus univalence equates

$$\mathtt{Y} = \sum_{a\colon \mathtt{A}} (b =_{\mathtt{B}} fa) \times (b' =_{\mathtt{B}} fa) \times (b =_{\mathtt{B}} b')$$

and also

$$\mathtt{X} = \sum_{a\colon \mathtt{A}} (b =_{\mathtt{B}} fa) \times (b' =_{\mathtt{B}} fa) \times (ga =_{\mathtt{C}} ga)$$

The third terms of each of these types are redudant because $\mathtt{B}$ and $\mathtt{C}$ are mere sets. Applying beta-reduction and univalence again equates

$$\mathtt{Y} = \sum_{a\colon \mathtt{A}} (b =_{\mathtt{B}} fa) \times (b' =_{\mathtt{B}} fa)$$

and

$$\mathtt{X} = \sum_{a\colon \mathtt{A}} (b =_{\mathtt{B}} fa) \times (b' =_{\mathtt{B}} fa),$$

hence $\mathtt{X} = \mathtt{Y}$. It follows that $\mathtt{code}(\mathtt{inl}b, \mathtt{inl}b')$ is equivalent to the pushout of the span

$$\begin{tikzcd}
\mathtt{X} \arrow[r, "!"] \arrow[d, "\mathrm{id}"'] & \mathtt{1} \\
\mathtt{X}
\end{tikzcd}$$

which is a mere set. ∎

This completes our current work with $\mathtt{code}(\mathtt{inl}b, \mathtt{inl}b')$.

Next, define

$$\mathtt{code}(\mathtt{inl}b, \mathtt{inr}c) :\equiv \sum_{a:\mathtt{A}} (b =_{\mathtt{B}} fa) \times (c =_{\mathtt{C}} ga).$$

Proposition 1. $\mathtt{code}(\mathtt{inl}b, \mathtt{inr}c)$ *is a mere proposition.*

Proof. Indeed, if there does not exist an $a : A$ such that $b =_B f(a)$ and $c' =_C g(a)$ are both populated, then $\mathtt{code}\,(\mathtt{inl}(b), \mathtt{inr}(c'))$ is empty. If there exists a single $a : A$ such that $b =_B f(a)$ and $c' =_C g(a)$ are both populated, then because they are each equivalent to $\mathtt{1}$, $\mathtt{code}\,(\mathtt{inl}(b), \mathtt{inr}(c'))$ is also equivalent to $\mathtt{1}$. If there is $a, a' : A$ such that $b =_B f(a)$ and $c' =_C g(a)$, and also $b =_B f(a')$ and $c' =_C g(a')$, then the injectivity of f and $f(a) =_B b =_B f(a')$ implies that $a =_A a'$ which also gives us that $\mathtt{code}\,(\mathtt{inl}(b), \mathtt{inr}(c'))$ is equivalent to $\mathtt{1}$. ∎

Finally, we need to construct a witness to

$$\mathtt{code}(\mathtt{inl}b, \mathtt{ap}_{\mathtt{glue}}a) \colon \mathtt{code}(\mathtt{inl}b, \mathtt{inl}fa) = \mathtt{code}(\mathtt{inl}b, \mathtt{inr}ga).$$

Because both sides of the equation are mere propositions, it suffices to show that they are simultaneously empty or populated.

Lemma 3. *There exist functions* $\mathtt{code}(\mathtt{inl}b, \mathtt{inl}fa) \to \mathtt{code}(\mathtt{inl}b, \mathtt{inr}ga)$ *and* $\mathtt{code}(\mathtt{inl}b, \mathtt{inl}fa) \to \mathtt{code}(\mathtt{inl}b, \mathtt{inr}ga)$*. Moreover, they form an equivalence.*

Proof. Suppose $\mathtt{code}(\mathtt{inl}b, \mathtt{inl}ga)$ is inhabited. This point is equal $(p, \mathtt{refl}_{ga})$ and p is pushed forward to populate $\mathtt{code}(\mathtt{inl}b, \mathtt{inl}fa)$.

suppose that $\mathtt{code}(\mathtt{inl}b, \mathtt{inl}fa)$ is inhabited. This implies that either $p \colon b =_{\mathtt{B}} fa$ or $(q, r, s) \colon (b =_{\mathtt{B}} fa') \times (fa =_{\mathtt{B}} fa'') \times (ga' =_{\mathtt{C}} ga'')$ in the first case, $(p, \mathtt{refl}_{ga}) \colon \mathtt{code}(\mathtt{inl}b, \mathtt{inr}ga)$ in the second case, the injectivity of f allows us to beta-reduce a to a''. That is, we actually have that $(q, r, s) \colon (b =_{\mathtt{B}} fa') \times (fa =_{\mathtt{B}} fa) \times (ga' =_{\mathtt{C}} ga)$ hence $(q, s^{-1}) \colon \mathtt{code}(\mathtt{inl}b, \mathtt{inr}ga)$.

the functions form an equivalence because both $\mathtt{code}(\mathtt{inl}b, \mathtt{inl}fa)$ and $\mathtt{code}(\mathtt{inl}b, \mathtt{inr}ga)$ are propositions. □

1.2. **Define** $\prod_{x \colon \mathtt{B}+_{\mathtt{A}}\mathtt{C}} \mathtt{code}(\mathtt{inr}c, x)$**.** To define $\prod\limits_{x \colon \mathtt{B}+_{\mathtt{A}}\mathtt{C}} \mathtt{code}(\mathtt{inr}c, x)$ requires us to compute three values:

- $\mathtt{code}(\mathtt{inr}c, \mathtt{inl}b)$,
- $\mathtt{code}(\mathtt{inr}c, \mathtt{inr}c')$, and
- $\mathtt{code}(\mathtt{inr}c, \mathtt{glue}a)$,

the first two of which we show are mere propositions.

Define

$$\mathtt{code}(\mathtt{inr}c, \mathtt{inl}b) :\equiv \sum_{a:\mathtt{A}} (c =_{\mathtt{C}} ga) \times (b =_{\mathtt{B}} fa).$$

Lemma 4. *The type* $\mathtt{code}(\mathtt{inr}c, \mathtt{inl}b)$ *is a proposition.*

Proof. See lemma 4. □

Define

$$\mathtt{code}(\mathtt{inr}c, \mathtt{inr}c') :\equiv \sum_{a,a':a} (c =_{\mathtt{C}} ga) \times (c' =_{\mathtt{C}} ga') \times (fa =_{\mathtt{B}} fa').$$

Lemma 5. *The type* $\mathtt{code}(\mathtt{inr}c, \mathtt{inr}c')$ *is a mere proposition.*

Proof. Because f is in injection, $a =_{\mathtt{A}} a'$ is populated so $\mathtt{code}(\mathtt{inr}c, \mathtt{inr}c')$ beta-reduces to

$$\sum_{a,a' \colon \mathtt{A}} (c =_{\mathtt{C}} ga) \times (c' =_{\mathtt{C}} ga) \times (a =_{\mathtt{A}} a)$$

which further reduces to

$$\mathtt{X} :\equiv \sum_{a \colon \mathtt{A}} (c =_{\mathtt{C}} ga) \times (c' =_{\mathtt{C}} ga)$$

because $\mathtt{A}$ is a set. Let (p, q) and (p', q') be in $\mathtt{X}$. The cube

$$\begin{tikzcd}
ga \arrow[rr, dash] \arrow[dr, dash, "p"] \arrow[dd, dash, "p'"'] & & ga \arrow[dr, dash, "q"] \arrow[dd, dash, "q'" near end] & \\
& c \arrow[rr, dash, crossing over] \arrow[dd, dash] & & c \arrow[dd, dash] \\
ga' \arrow[rr, dash] \arrow[dr, dash] & & ga' \arrow[dr, dash] & \\
& c \arrow[rr, dash] & & c
\end{tikzcd}$$

commutes and is filled because $\mathtt{C}$ is a set. Therefore $(p, q) = (p', q')$ □

DANIEL: Is the lemma below sufficient? It says the two codes are equivalent, but couldn't they still be empty?

We now construct a point

$$\mathtt{code}(\mathtt{inr}c, \mathtt{ap}_{\mathtt{glue}}a) \colon \mathtt{code}(\mathtt{inr}c, \mathtt{inl}fa) = \mathtt{code}(\mathtt{inr}c, \mathtt{inl}ga).$$

because both sides of the equality are propositions, a unique definition for $\mathtt{code}(\mathtt{inr}c, \mathtt{ap}_{\mathtt{glue}}a)$ exists if we construct an equivalnce.

Lemma 6. *The functions*

$$\mathtt{code}(\mathtt{inr}c, \mathtt{inl}fa) \to \mathtt{code}(\mathtt{inr}c, \mathtt{inr}ga),\ (p, q) \mapsto p$$

and

$$\mathtt{code}(\mathtt{inr}c, \mathtt{inr}ga) \to \mathtt{code}(\mathtt{inr}c, \mathtt{inl}fa),\ r \mapsto (r, \mathtt{refl}_{\mathtt{inl}fa})$$

form an equivalence.

Proof. First, let us show these actually are functions.

Let $(p, q) \colon \mathtt{code}(\mathtt{inr}c, \mathtt{inl}fa)$. since f is injective, we have that $a =_{\mathtt{A}} a'$. beta-reducing ga' to ga, gives us that $p \colon \mathtt{code}(\mathtt{inr}c, \mathtt{inr}ga)$. this provides the first function.

Given $r \colon \mathtt{code}(\mathtt{inr}c, \mathtt{inr}ga)$, r is equal to a point in $c =_{\mathtt{C}} ga$. this provides a point $(r, \mathtt{refl}_{fa}) \colon \mathtt{code}(\mathtt{inr}c, \mathtt{inl}fa)$. this gives the second function.

The functions form an equivalence because both $\mathtt{code}(\mathtt{inr}c, \mathtt{inl}fa)$ and $\mathtt{code}(\mathtt{inr}c, \mathtt{inr}ga)$ are propositions. □

The next stage in proving theorem 1 is showing that $\mathtt{inl}b =_{\mathtt{B}+_{\mathtt{A}}\mathtt{C}} x$ and $\mathtt{inr}c =_{\mathtt{B}+_{\mathtt{A}}\mathtt{C}} x$ are mere propositions for any x. To do so, we construct equivalences between, respectively, $\mathtt{code}(\mathtt{inl}b, x)$ and $\mathtt{code}(\mathtt{inr}c, x)$ which we know are mere propositions. Specifically, we define maps

$$\mathtt{encode}(\mathtt{inl}b) \colon \prod_{x \colon \mathtt{B}+_{\mathtt{A}}\mathtt{C}} (\mathtt{inl}b =_{\mathtt{B}+_{\mathtt{A}}\mathtt{C}} x) \to \mathtt{code}(\mathtt{inl}b, x)$$

$$\mathtt{encode}(\mathtt{inr}c) \colon \prod_{x \colon \mathtt{B}+_{\mathtt{A}}\mathtt{C}} (\mathtt{inr}c =_{\mathtt{B}+_{\mathtt{A}}\mathtt{C}} x) \to \mathtt{code}(\mathtt{inr}c, x)$$

$$\mathtt{decode}(\mathtt{inl}b) \colon \prod_{x \colon \mathtt{B}+_{\mathtt{A}}\mathtt{C}} \mathtt{code}(\mathtt{inl}b, x) \to (\mathtt{inl}b =_{\mathtt{B}+_{\mathtt{A}}\mathtt{C}} x)$$

$$\mathtt{decode}(\mathtt{inr}c) \colon \prod_{x \colon \mathtt{B}+_{\mathtt{A}}\mathtt{C}} \mathtt{code}(\mathtt{inr}c, x) \to (\mathtt{inr}c =_{\mathtt{B}+_{\mathtt{A}}\mathtt{C}} x)$$

with corresponding $\mathtt{encode}$ and $\mathtt{decode}$ pairs forming mutual equivalences.

1.3. **defining $\mathtt{encode}$.** We define $\mathtt{encode}(\mathtt{inl}b)$ and $\mathtt{encode}(\mathtt{inr}c)$ by inducting on $x\colon \mathtt{B}+_{\mathtt{A}}\mathtt{C}$. In doing so, we make use of path induction.

Define

$$\mathtt{encode}(\mathtt{inl}(b))\colon \prod_{x\colon \mathtt{B}+_{\mathtt{A}}\mathtt{C}} (\mathtt{inl}(b) =_{\mathtt{B}+_{\mathtt{A}}\mathtt{C}} x) \to \mathtt{code}(\mathtt{inl}(b), x)$$

by the $\mathtt{refl}_{\mathtt{inl}b} \mapsto \mathtt{refl}_{\mathtt{inl}'b}$. Recall, $\mathtt{inl}$ corresponds to the pushout map into $\mathtt{B} +_{\mathtt{A}} \mathtt{C}$ and $\mathtt{inl}'$ to the pushout map into $\mathtt{code}(\mathtt{inl}b, \mathtt{inl}b')$.

Define

$$\mathtt{encode}(\mathtt{inr}c)\colon \prod_{x\colon \mathtt{B}+_{\mathtt{A}}\mathtt{C}} (\mathtt{inr}c =_{\mathtt{B}+_{\mathtt{A}}\mathtt{C}} x) \to \mathtt{code}(\mathtt{inr}c, x)$$

by the assignemnt $\mathtt{refl}_{\mathtt{inr}c} \mapsto \mathtt{refl}_c$. Note, we use that $\mathtt{code}(\mathtt{inr}c, \mathtt{inr}c)$ is equivalent to $c =_{\mathtt{C}} c$ as shown in Lemma 5.

Define

$$\mathtt{ap}_{\mathtt{encode}(\mathtt{inl}b)}\colon ((b =_{\mathtt{B}+_{\mathtt{A}}\mathtt{C}} fa) \to \mathtt{code}(\mathtt{inl}b, \mathtt{inl}fa)) = ((b =_{\mathtt{B}+_{\mathtt{A}}\mathtt{C}} ga) \to \mathtt{code}(\mathtt{inl}b, \mathtt{inr}ga))$$

to be the diagram

$$\begin{tikzcd}
(b =_{\mathtt{B}+_{\mathtt{A}}\mathtt{C}} fa) \arrow[r, "\mathtt{encode}(\mathtt{inl}b)(\mathtt{inl}fa)"] \arrow[d, "(-);\mathtt{glue}a"'] & \mathtt{code}(\mathtt{inl}b, \mathtt{inl}fa) \arrow[d, "\mathtt{code}(\mathtt{inl}b, \mathtt{glue}a)"] \\
(b =_{\mathtt{B}+_{\mathtt{A}}\mathtt{C}} ga) \arrow[r, "\mathtt{encode}(\mathtt{inl}b)(\mathtt{inr}ga)"] & \mathtt{code}(\mathtt{inl}b, \mathtt{inr}ga)
\end{tikzcd}$$

which commutes because $\mathtt{code}(\mathtt{inl}b, ga)$ is a mere proposition.

Define

$$\mathtt{ap}_{\mathtt{encode}(\mathtt{inr}c)}\colon ((c =_{\mathtt{B}+_{\mathtt{A}}\mathtt{C}} fa) \to \mathtt{code}(\mathtt{inr}c, \mathtt{inl}fa)) = ((c =_{\mathtt{B}+_{\mathtt{A}}\mathtt{C}} ga) \to \mathtt{code}(\mathtt{inr}c, \mathtt{inr}ga))$$

to be the diagram

$$\begin{tikzcd}
(c =_{\mathtt{B}+_{\mathtt{A}}\mathtt{C}} fa) \arrow[r, "\mathtt{encode}(\mathtt{inr}c)(\mathtt{inl}fa)"] \arrow[d, "(-);\mathtt{glue}a"'] & \mathtt{code}(\mathtt{inr}c, \mathtt{inl}fa) \arrow[d, "\mathtt{code}(\mathtt{inr}c, \mathtt{glue}a)"] \\
(c =_{\mathtt{B}+_{\mathtt{A}}\mathtt{C}} ga) \arrow[r, "\mathtt{encode}(\mathtt{inr}c)(\mathtt{inr}ga)"] & \mathtt{code}(\mathtt{inr}c, \mathtt{inr}ga)
\end{tikzcd}$$

which commutes because $\mathtt{code}(\mathtt{inr}c, ga)$ is a mere proposition.

1.4. **Defining $\mathtt{decode}$.** To define the map

$$\mathtt{decode}(\mathtt{inl}b)\colon \prod_{x\colon \mathtt{B}+_{\mathtt{A}}\mathtt{C}} \mathtt{code}(\mathtt{inl}b, x) \to (\mathtt{inl}b =_{\mathtt{B}+_{\mathtt{A}}\mathtt{C}} x) \tag{1}$$

we use induction on $x\colon \mathtt{B} +_{\mathtt{A}} \mathtt{C}$ which, in practice, means that we need two function types

- $\mathtt{decode}(\mathtt{inl}b)(\mathtt{inl}b')\colon \mathtt{code}(\mathtt{inl}b, \mathtt{inl}b') \to (\mathtt{inl}b =_{\mathtt{B}+_{\mathtt{A}}\mathtt{C}} \mathtt{inl}b')$
- $\mathtt{decode}(\mathtt{inl}b)(\mathtt{inr}c)\colon \mathtt{code}(\mathtt{inl}b, \mathtt{inr}c) \to (\mathtt{inl}b =_{\mathtt{B}+_{\mathtt{A}}\mathtt{C}} \mathtt{inr}c$

and a witness

$$\begin{aligned}\mathtt{ap}_{\mathtt{decode}(\mathtt{inl}b)}(\mathtt{glue}a) \colon (\mathtt{code}(\mathtt{inl}b, \mathtt{inl}fa) &\to (\mathtt{inl}b =_{\mathtt{B}+_{\mathtt{A}}\mathtt{C}} \mathtt{inl}fa) \\ = (\mathtt{code}(\mathtt{inl}b, \mathtt{inr}ga) &\to (\mathtt{inl}b =_{\mathtt{B}+_{\mathtt{A}}\mathtt{C}} \mathtt{inr}ga)\end{aligned}$$

Define

$$\mathtt{decode}(\mathtt{inl}b) \colon \mathtt{code}(\mathtt{inl}b, \mathtt{inl}b') \to (\mathtt{inl}b =_{\mathtt{B}+_{\mathtt{A}}\mathtt{C}} \mathtt{inl}b')$$

by inducting on $\mathtt{code}(\mathtt{inl}b, \mathtt{inl}b')$ This requires three values.

- Let $\mathtt{decode}(\mathtt{inl}b)(\mathtt{inl}b')(\mathtt{inl}'p) \colon (\mathtt{inl}b =_{\mathtt{B}+_{\mathtt{A}}\mathtt{C}} \mathtt{inl}b')$, where $p \colon b =_{\mathtt{B}} b'$, be $\mathtt{ap}_{\mathtt{inl}}(p)$;
- Let $\mathtt{decode}(\mathtt{inl}b)(\mathtt{inr}c)(\mathtt{inr}'(q,r,s)) \colon (b = \mathtt{B} +_{\mathtt{A}} \mathtt{C}b')$, where $(q,r,s) \colon \sum_{a,a' \colon \mathtt{A}} (b =_{\mathtt{B}} fa) \times (b' =_{\mathtt{B}} fa') \times (ga =_{\mathtt{C}} ga')$, be $ap_{\mathtt{inl}}q; \mathtt{glue}a; \mathtt{ap}_{\mathtt{inr}}s; \mathtt{glue}^{-1}a'; r^{-1}$ and
- The path

  $$\begin{aligned}\mathtt{ap}_{\mathtt{decode}(\mathtt{inl}b)}(\mathtt{glue}(t,u,v)) \colon (\mathtt{decode}(\mathtt{inl}b)(\mathtt{inl}'(\alpha(t,u,v))) &=_{\mathtt{inl}b =_{\mathtt{B}+_{\mathtt{A}}\mathtt{C}} \mathtt{inl}b'} \\ \mathtt{decode}(\mathtt{inl}b)(\mathtt{inr}'(\beta(t,u,v))),\end{aligned}$$

  where $(t,u,v) \colon (b =_{\mathtt{B}} fa) \times (b' =_{\mathtt{B}} fa') \times (b =_{\mathtt{B}} b')$, is trivial because $\mathtt{code}(\mathtt{inl}b, \mathtt{inl}b')$ is a mere set.

To define

$$\mathtt{decode}(\mathtt{inl}b) \colon \mathtt{code}(\mathtt{inl}b, \mathtt{inr}c) \to (\mathtt{inl}b =_{\mathtt{B}+_{\mathtt{A}}\mathtt{C}} \mathtt{inr}c),$$

recall that

$$\mathtt{code}(\mathtt{inl}b, \mathtt{inr}c) = \sum_{a \colon \mathtt{A}} (b =_{\mathtt{B}} fa) \times (c =_{\mathtt{C}} ga)$$

Define

$$\mathtt{decode}(\mathtt{inl}b)(p,q) :\equiv \mathtt{inl}p; \mathtt{glue}a; \mathtt{inr}q$$

Right now, we define

$$\begin{aligned}\mathtt{ap}_{\mathtt{decode}(\mathtt{inl}b)}(\mathtt{glue}a) \colon (\mathtt{code}(\mathtt{inl}b, \mathtt{inl}fa) &\to (\mathtt{inl}b =_{\mathtt{B}+_{\mathtt{A}}\mathtt{C}} \mathtt{inl}fa) \\ = (\mathtt{code}(\mathtt{inl}b, \mathtt{inl}ga) &\to (\mathtt{inl}b =_{\mathtt{B}+_{\mathtt{A}}\mathtt{C}} \mathtt{inl}ga)\end{aligned}$$

To define this is to construct a commuting square

$$\begin{tikzcd} \mathtt{code}(\mathtt{inl}b, \mathtt{inl}fa) \arrow[r, "\mathtt{decode}(\mathtt{inl}b)(\mathtt{inl}b')"] \arrow[d, "\theta"'] & (\mathtt{inl}b =_{\mathtt{B}+_{\mathtt{A}}\mathtt{C}} \mathtt{inl}fa) \arrow[d, "\theta'"'] \\ \mathtt{code}(\mathtt{inl}b, \mathtt{inl}ga) \arrow[r, "\mathtt{decode}(\mathtt{inl}b)(\mathtt{inl}c)"] & (\mathtt{inl}b =_{\mathtt{B}+_{\mathtt{A}}\mathtt{C}} \mathtt{inl}ga) \end{tikzcd}$$

We have already defined the two $\mathtt{decode}$ maps. The map θ' is post-composition by $\mathtt{glue}a$. The map θ requires induction to define because we are mapping out of a pushout. Therefore, we need the following three values to define θ:

- $\theta(\mathtt{inl}'p)$ for $p\colon b =_{\mathtt{B}} fa$
- $\theta(\mathtt{inr}'(q,r,s))$ for $(q,r,s)\colon \sum_{a',a''\colon \mathtt{A}} (b =_{\mathtt{B}} fa') \times (fa =_B fa'') \times (ga' =_C ga'')$
- $\mathtt{ap}_\theta(\mathtt{glue}(t,u,v))\colon \theta(\mathtt{inl}'(\alpha(t,u,v))) = \theta(\mathtt{inr}'(\beta(t,u,v)))$.

A quick remark on notation: since f is monic and $\mathtt{B}$ is a set, we can pull back any path of form $r\colon fa =_{\mathtt{B}} fa'$ to a determined path $\hat{r}\colon a =_{\mathtt{A}} a'$.

Define $\theta(\mathtt{inl}'p)$ for $p\colon b =_{\mathtt{B}} fa$ to be $(p, \mathtt{refl}_{ga})$. Define $\theta(q,r,s)$ to be $(q, \mathtt{ap}_g(\hat{r}); s^{-1})$.

Now we know that $\mathtt{ap}_\theta(\mathtt{glue}(t,u,v))$ is a path from $\theta(\mathtt{inl}'(v)) = (v, \mathtt{refl}_{ga})$ to $\theta(\mathtt{inr}'(\beta(t,u,v)))$ But that can be reduced:

$$\begin{aligned}\theta(\mathtt{inr}'(t,u,\mathtt{ap}_g(\hat{v}))) &= (t, \mathtt{ap}_g(\hat{u}); \mathtt{ap}_g(\hat{v})^{-1}) \\ &= (t, \mathtt{ap}_g(\hat{u}; \hat{v}^{-1})).\end{aligned}$$

With this in mind, we define $\mathtt{ap}_\theta(\mathtt{glue}(t,u,v))$ to be post-composition by

$$(\mathtt{ap}_f(\hat{u}; \hat{v}^{-1}), \mathtt{ap}_g(\hat{u}; \hat{v}^{-1})).$$

Now we need to check that the diagram commutes. Since $\mathtt{code}(\mathtt{inl}b, \mathtt{inl}fa)$ is a set, it suffices to check that

$$\theta'(\mathtt{decode}(\mathtt{inl}b)(\mathtt{inl}fa)(x)) =_{\mathtt{inl}b =_{\mathtt{B}+_{\mathtt{A}}\mathtt{C}} \mathtt{inr}ga} \mathtt{decode}(\mathtt{inl}b)(\mathtt{inr}ga)(\theta(x))$$

where $x\colon \mathtt{code}(\mathtt{inl}b, \mathtt{inl}fa)$ takes values

- $\mathtt{inl}'p$ for $p\colon b =_{\mathtt{B}} fa$, and
- $\mathtt{inr}'(q,r,s)$ for $(q,r,s)\colon \sum_{a',a''\colon \mathtt{A}} (b =_{\mathtt{B}} fa') \times (fa =_{\mathtt{B}} fa'') \times (ga' =_{\mathtt{C}} ga'')$.

We also need to check that

$$\mathtt{ap}_{\theta';\mathtt{decode}(\mathtt{inl}b)(\mathtt{inl}fa))}(\mathtt{glue}(t,u,v)) =_{\mathtt{inl}b =_{\mathtt{B}+_{\mathtt{A}}\mathtt{C}} \mathtt{inr}ga} \mathtt{ap}_{\mathtt{decode}(\mathtt{inl}b)(\mathtt{inr}ga);\theta}(\mathtt{glue}(t,u,v))$$

Set $x :\equiv \mathtt{inl}'p$. Then

$$\begin{aligned}\theta'(\mathtt{decode}(\mathtt{inl}b)(\mathtt{inl}fa)(\mathtt{inl}'p)) &= \theta'((\mathtt{ap}_{\mathtt{inl}'}p)) \\ &= \mathtt{ap}_{\mathtt{inl}'}p; \mathtt{glue}a\end{aligned}$$

Also,

$$\begin{aligned}\mathtt{decode}(\mathtt{inl}b)(\mathtt{inr}ga)(\theta(\mathtt{inl}'p)) &= \mathtt{decode}(\mathtt{inl}b)(\mathtt{inr}ga)((p, \mathtt{refl}_{ga})) \\ &= \mathtt{ap}_{\mathtt{inl}'}p; \mathtt{glue}a; \mathtt{ap}_{\mathtt{inr}'}\mathtt{refl}_{ga} \\ &= \mathtt{ap}_{\mathtt{inl}'}p; \mathtt{glue}a; \mathtt{refl}_{\mathtt{inr}'ga} \\ &= \mathtt{ap}_{\mathtt{inl}'}p; \mathtt{glue}a\end{aligned}$$

Therefore, the square commutes on $\mathtt{inl}'p$

Set $x :\equiv \mathtt{inr}'(q,r,s)$. Then

$$\begin{aligned}\theta'(\mathtt{decode}(\mathtt{inl}b)(\mathtt{inl}fa)(\mathtt{inr}'(q,r,s))) &= \theta'(\mathtt{ap}_{\mathtt{inl}}q; \mathtt{glue}a'; \mathtt{ap}_{\mathtt{inr}}s; \mathtt{glue}a''; \mathtt{ap}_{\mathtt{inl}}r^{-1}) \\ &= \mathtt{ap}_{\mathtt{inl}}q; \mathtt{glue}a'; \mathtt{ap}_{\mathtt{inr}}s; \mathtt{glue}^{-1}a''; \mathtt{ap}_{\mathtt{inl}}r^{-1}; \mathtt{glue}a \\ &= \mathtt{ap}_{\mathtt{inl}}q; \mathtt{glue}a'; \mathtt{ap}_{\mathtt{inr}}s; \mathtt{glue}^{-1}a''; \mathtt{ap}_{\mathtt{inl}}\mathtt{ap}_f\hat{r}^{-1}; \mathtt{glue}a\end{aligned}$$

Also,

$$\begin{aligned}
\mathtt{decode}(\mathtt{inl}b)(\mathtt{inr}ga)(\theta(\mathtt{inr}'(q,r,s))) &= \mathtt{decode}(\mathtt{inl}b)(\mathtt{inr}ga)(q, \mathtt{ap}_g(\hat{r}; s^{-1}))\\
&= \mathtt{ap}_{\mathtt{inl}}q; \mathtt{glue}a'; (\mathtt{ap}_{\mathtt{inr}}(\mathtt{ap}_g\hat{r}; s^{-1})^{-1})\\
&= \mathtt{ap}_{\mathtt{inl}}q; \mathtt{glue}a'; \mathtt{ap}_{\mathtt{inr}}(\mathtt{ap}_g(\hat{r}); (\mathtt{ap}_g(s^{-1}))^{-1})\\
&= \mathtt{ap}_{\mathtt{inl}}q; \mathtt{glue}a'; \mathtt{ap}_{\mathtt{inr}}(s^{-1})^{-1}; \mathtt{ap}_g(\hat{r})^{-1}\\
&= \mathtt{ap}_{\mathtt{inl}}q; \mathtt{glue}a'; \mathtt{ap}_{\mathtt{inr}}(s^{-1})^{-1}; \mathtt{ap}_g(\hat{r})^{-1}\\
&= \mathtt{ap}_{\mathtt{inl}}q; \mathtt{glue}a'; \mathtt{ap}_{\mathtt{inr}}s; \mathtt{ap}_g(\hat{r})^{-1}
\end{aligned}$$

We just need to check that

$$\mathtt{ap}_g(\hat{r})^{-1} = \mathtt{glue}^{-1}a''; \mathtt{ap}_{\mathtt{inl}}\mathtt{ap}_f\hat{r}^{-1}; \mathtt{glue}a$$

But this follows from the continuity of $\mathtt{glue}$, that is, it preserves paths. Therefore, the square commutes on $\mathtt{inr}'(q,r,s)$.

It remains to check that

$$\mathtt{ap}_{\theta';\mathtt{decode}(\mathtt{inl}b)(\mathtt{inl}fa))}(\mathtt{glue}(t,u,v)) =_{\mathtt{inl}b =_{\mathtt{B}+_{\mathtt{A}}\mathtt{C}} \mathtt{inr}ga} \mathtt{ap}_{\mathtt{decode}(\mathtt{inl}b)(\mathtt{inr}ga);\theta}(\mathtt{glue}(t,u,v))$$

for $(t,u,v)\colon \sum_{a',a''\colon \mathtt{A}} (b =_{\mathtt{B}} fa') \times (fa =_{\mathtt{B}} fa'') \times (b =_{\mathtt{B}} fa)$. Here, we make some reductions.

- Because f is monic, we get that $fa =_{\mathtt{B}} fa''$ reduces to $fa =_{\mathtt{B}} fa$. Since $\mathtt{B}$ is a set, we can take $u = \mathtt{refl}_{fa}$.
- The type $b =_{\mathtt{B}} fa'$ reduces to $fa =_{\mathtt{B}} fa$ because $\mathtt{B}$ is monic, so $t = \mathtt{refl}_{fa}$..
- The type $b =_{\mathtt{B}} fa$ reduces to $fa =_{\mathtt{B}} fa$ because $\mathtt{B}$ is monic, so $v = \mathtt{refl}_{fa}$.

Without loss of generality, we can take $(t,u,v) = (\mathtt{refl}_{fa}, \mathtt{refl}_{fa}, \mathtt{refl}_{fa})$. Therefore, it suffices to check that

$$\begin{aligned}
&\mathtt{ap}_{\theta';\mathtt{decode}(\mathtt{inl}b)(\mathtt{inl}fa))}(\mathtt{glue}(\mathtt{refl}_{fa}, \mathtt{refl}_{fa}, \mathtt{refl}_{fa}))\\
&=_{\mathtt{inl}b =_{\mathtt{B}+_{\mathtt{A}}\mathtt{C}} \mathtt{inr}ga} \mathtt{ap}_{\mathtt{decode}(\mathtt{inl}b)(\mathtt{inr}ga);\theta}(\mathtt{glue}(\mathtt{refl}_{fa}, \mathtt{refl}_{fa}, \mathtt{refl}_{fa})).
\end{aligned}$$

But because the square commutes on points as shown above, the two paths on the left and right of the above equation are certainly parallel. Then functorality gives us that $\mathtt{refl}_{fa}$ is preserved. Hence the square commutes.

To define the map

$$\mathtt{decode}(\mathtt{inr}c)\colon \prod_{x\colon \mathtt{B}+_{\mathtt{A}}\mathtt{C}} \mathtt{code}(\mathtt{inr}c, x) \to \mathtt{inr}(c =_{\mathtt{B}+_{\mathtt{A}}\mathtt{C}} x)$$

we use induction on $\mathtt{B} +_{\mathtt{A}} \mathtt{C}$ which, in practice, means that we require the two function types

- $\mathtt{decode}(\mathtt{inr}c)(\mathtt{inl}b)\colon \mathtt{code}(\mathtt{inr}c, \mathtt{inl}b) \to (\mathtt{inr}c =_{\mathtt{B}+_{\mathtt{A}}\mathtt{C}} \mathtt{inl}b)$
- $\mathtt{decode}(\mathtt{inr}c)(\mathtt{inr}c')\colon \mathtt{code}(\mathtt{inr}c, \mathtt{inr}c') \to (\mathtt{inr}c =_{\mathtt{B}+_{\mathtt{A}}\mathtt{C}} \mathtt{inr}c')$

and a witness

$$\begin{aligned}
\mathtt{ap}_{\mathtt{decode}(\mathtt{inr}c)}(\mathtt{glue}a)\colon &(\mathtt{code}(\mathtt{inr}c, \mathtt{inl}fa) \to (\mathtt{inr}c =_{\mathtt{B}+_{\mathtt{A}}\mathtt{C}} \mathtt{inl}fa)\\
&= (\mathtt{code}(\mathtt{inr}c, \mathtt{inl}ga) \to (\mathtt{inr}c =_{\mathtt{B}+_{\mathtt{A}}\mathtt{C}} \mathtt{inl}ga)
\end{aligned}$$

First, let us define the map

$$\mathtt{decode}(\mathtt{inr}c)(\mathtt{inl}b)\colon \mathtt{code}(\mathtt{inr}c, \mathtt{inl}b) \to (\mathtt{inr}c =_{\mathtt{B}+_{\mathtt{A}}\mathtt{C}} \mathtt{inl}b).$$

Recall that $\mathtt{code}(\mathtt{inr}c, \mathtt{inl}b) :\equiv \sum_{a:\,\mathtt{A}} (c =_{\mathtt{C}} ga) \times (b =_{\mathtt{B}} fa)$. Thus for any (p,q) in $\mathtt{code}(\mathtt{inr}c, \mathtt{inl}b)$, we define a path $\mathtt{decode}(\mathtt{inr}c)(\mathtt{inl}b)(p,q)$ in $\mathtt{B} +_{\mathtt{A}} \mathtt{C}$ from $\mathtt{inr}c$ to $\mathtt{inl}b$. Take this path to be $\mathtt{inr}p; \mathtt{glue}^{-1}a; \mathtt{inl}q^{-1}$.

Next, we define

$$\mathtt{decode}(\mathtt{inr}c)(\mathtt{inr}c') : \mathtt{code}(\mathtt{inr}c, \mathtt{inr}c') \to (\mathtt{inr}c =_{\mathtt{B}+_{\mathtt{A}}\mathtt{C}} \mathtt{inr}c').$$

Recall that

$$\mathtt{code}(\mathtt{inr}c, \mathtt{inr}c') :\equiv \sum_{a:\,\mathtt{A}} (c =_{\mathtt{C}} ga) \times (c' =_{\mathtt{C}} ga') \times (fa =_{\mathtt{B}} fa').$$

Thus for any (p,q,r) in $\mathtt{code}(\mathtt{inr}c, \mathtt{inr}c')$, we define a path $\mathtt{decode}(\mathtt{inr}c)(\mathtt{inr}c')(p,q,r)$ in $\mathtt{B} +_{\mathtt{A}} \mathtt{C}$ from $\mathtt{inr}c$ to $\mathtt{inr}c'$. Take this path to be $\mathtt{inr}p; \mathtt{glue}^{-1}a; \mathtt{inl}r\mathtt{glue}a'; \mathtt{inr}q^{-1}$.

Finally, we define a path

$$\begin{aligned}\mathtt{ap}_{\mathtt{decode}(\mathtt{inr}c)}(\mathtt{glue}a) : (\mathtt{code}(\mathtt{inr}c, \mathtt{inl}fa) &\to (\mathtt{inr}c =_{\mathtt{B}+_{\mathtt{A}}\mathtt{C}} \mathtt{inl}fa) \\ &= (\mathtt{code}(\mathtt{inr}c, \mathtt{inl}ga) \to (\mathtt{inr}c =_{\mathtt{B}+_{\mathtt{A}}\mathtt{C}} \mathtt{inl}ga).\end{aligned}$$

Replacing the two $\mathtt{code}$ expressions with their definitions, this path is constructed as a commuting square

$$\begin{tikzcd}
\sum_{a':\,\mathtt{A}} (c =_{\mathtt{C}} ga') \times (fa =_{\mathtt{B}} fa') \arrow[r, "\mathtt{decode}"] \arrow[d, "\theta"'] & (\mathtt{inr}c =_{\mathtt{B}+_{\mathtt{A}}\mathtt{C}} \mathtt{inl}fa) \arrow[d, "\theta'"] \\
\sum_{a',a'':\,\mathtt{A}} (c =_{\mathtt{C}} ga') \times (ga =_{\mathtt{C}} ga') \times (fa' =_{\mathtt{B}} fa'') \arrow[r, "\mathtt{decode}"] & \mathtt{inr}c =_{\mathtt{B}+_{\mathtt{A}}\mathtt{C}} \mathtt{inl}ga
\end{tikzcd}$$

The easier map to define is θ' which concatenates with $\mathtt{glue}a$. Then θ is given by $(p,q) \mapsto (p, \mathtt{refl}_{ga}, q)$. Now we check whether the square commutes. We have

$$\begin{aligned}\theta'(\mathtt{decode}(\mathtt{inr}c, -)(\mathtt{inl}fa)(p,q)) &= \theta'(\mathtt{inr}p; \mathtt{glue}^{-1}a'; \mathtt{inl}q^{-1}) \\ &= \mathtt{inr}p; \mathtt{glue}^{-1}a'; \mathtt{inl}q^{-1}; \mathtt{glue}a.\end{aligned}$$

We also have that

$$\begin{aligned}\mathtt{decode}(\mathtt{inr}c, -)(\mathtt{inl}ga)(\theta(p,q)) &= \mathtt{decode}(\mathtt{inr}c, -)(\mathtt{inl}ga)(p, \mathtt{refl}_{ga}, q) \\ &= \mathtt{inr}p; \mathtt{glue}^{-1}a'; \mathtt{inl}q^{-1}; \mathtt{glue}a; \mathtt{inr}\mathtt{refl}_{ga} \\ &= \mathtt{inr}p; \mathtt{glue}^{-1}a'; \mathtt{inl}q^{-1}; \mathtt{glue}a\end{aligned}$$

Hence the square commutes.

1.5. **Composing encode and decode.** Consider the composite

$$\mathtt{encode}; \mathtt{decode} : \prod_{x,y:\,\mathtt{B}+_{\mathtt{A}}\mathtt{C}} x =_{\mathtt{B}+_{\mathtt{A}}\mathtt{C}} y x =_{\mathtt{B}+_{\mathtt{A}}\mathtt{C}} y$$

To show that this map is the identity up to homotopy, we compute both

- $\mathtt{encode}; \mathtt{decode}(\mathtt{inl}b) : \prod_{x:\,\mathtt{B}+_{\mathtt{A}}\mathtt{C}} (\mathtt{inl}b =_{\mathtt{B}+_{\mathtt{A}}\mathtt{C}} x) \to (\mathtt{inl}b =_{\mathtt{B}+_{\mathtt{A}}\mathtt{C}} x)$, and

- $\mathtt{encode};\mathtt{decode}(\mathtt{inr}c)\colon \prod_{x\colon \mathtt{B}+_{\mathtt{A}}\mathtt{C}} \mathtt{inr}c =_{\mathtt{B}+_{\mathtt{A}}\mathtt{C}} x \to (\mathtt{inr}c =_{\mathtt{B}+_{\mathtt{A}}\mathtt{C}} x)$.

But computing these values actually requires the computation of the following four maps:

- $\mathtt{encode};\mathtt{decode}(\mathtt{inl}b)(\mathtt{inl}b')\colon \mathtt{code}(\mathtt{inl}b, \mathtt{inl}b') \to (\mathtt{inl}b =_{\mathtt{B}+_{\mathtt{A}}\mathtt{C}} \mathtt{inl}b')$. By path induction, it suffices to check $\mathtt{refl}_{\mathtt{inl}b}$:
$$\begin{aligned}\mathtt{encode};\mathtt{decode}(\mathtt{inl}b)(\mathtt{inl}b')(\mathtt{refl}_{\mathtt{inl}b}) &= \mathtt{encode}(\mathtt{ap}_{\mathtt{inl}'}\mathtt{refl}_{\mathtt{inl}b})\\ &= \mathtt{encode}(\mathtt{refl}_{\mathtt{inl}'b})\\ &= \mathtt{ap}_{\mathtt{inl}}\mathtt{refl}_{\mathtt{inl}'b}\\ &= \mathtt{refl}_{\mathtt{inl}b}.\end{aligned}$$
So this one checks out.
- $\mathtt{encode};\mathtt{decode}(\mathtt{inl}b)(\mathtt{inr}c)\colon \mathtt{code}(\mathtt{inl}b, \mathtt{inr}c) \to (\mathtt{inl}b =_{\mathtt{B}+_{\mathtt{A}}\mathtt{C}} \mathtt{inl}c)$. **DANIEL:** glue(a) *is* the only thing to check here, right? This is only non-trivial when $b =_{\mathtt{B}} fa$ and $c =_{\mathtt{C}} ga$. Hence
$$\mathtt{encode};\mathtt{decode}(\mathtt{inl}b)(\mathtt{inr}c)(\mathtt{glue}a) = \mathtt{decode}(\mathtt{inl}b)(\mathtt{inr}c)(\mathtt{refl}_{fa}, \mathtt{refl}_{ga}) = \mathtt{ap}_{\mathtt{inr}}\mathtt{refl}_{ga};\mathtt{glue}a;\mathtt{ap}_{\mathtt{inl}}\mathtt{re}$$
This one checks out too.
- $\mathtt{encode};\mathtt{decode}(\mathtt{inr}c,-)(\mathtt{inl}b)\colon \mathtt{code}(\mathtt{inr}c, \mathtt{inl}b) \to (\mathtt{inr}c =_{\mathtt{B}+_{\mathtt{A}}\mathtt{C}} \mathtt{inl}b)$. This is symmetric to the above case, so checks out.
- $\mathtt{encode};\mathtt{decode}(\mathtt{inr}c,-)(\mathtt{inr}c')\colon \mathtt{code}(\mathtt{inr}c, \mathtt{inr}c') \to (\mathtt{inr}c =_{\mathtt{B}+_{\mathtt{A}}\mathtt{C}} \mathtt{inr}c')$ is, by path induction,
$$\mathtt{encode};\mathtt{decode}(\mathtt{inr}c,-)(\mathtt{inr}c')(\mathtt{refl}_{\mathtt{inr}c}) = \mathtt{decode}(\mathtt{inr}c,-)(\mathtt{inr}c')(\mathtt{refl}_c) = \mathtt{ap}_{\mathtt{inr}}\mathtt{refl}_c = \mathtt{refl}_{\mathtt{inr}c}$$

And so, we have proved that $\mathtt{decode}$ is a section for $\mathtt{encode}$. The opposite direction remains.

Now, we look at the composite
$$\mathtt{decode};\mathtt{encode}\colon \prod_{x,y\colon \mathtt{B}+_{\mathtt{A}}\mathtt{C}} \mathtt{code}(x,y) \to \mathtt{code}(x,y)$$
We can compute this composite by computing the values

- $\mathtt{decode};\mathtt{encode}(\mathtt{inl}b)(\mathtt{inl}b')\colon \mathtt{code}(\mathtt{inl}b, \mathtt{inl}b') \to \mathtt{code}(\mathtt{inl}b, \mathtt{inl}b')$,
- $\mathtt{decode};\mathtt{encode}(\mathtt{inl}b)(\mathtt{inr}c)\colon \mathtt{code}(\mathtt{inl}b, \mathtt{inr}c) \to \mathtt{code}(\mathtt{inl}b, \mathtt{inr}c)$,
- $\mathtt{decode};\mathtt{encode}(\mathtt{inr}c)(\mathtt{inl}b)\colon \mathtt{code}(\mathtt{inr}c, \mathtt{inl}b) \to \mathtt{code}(\mathtt{inr}c, \mathtt{inl}b)$, and
- $\mathtt{decode};\mathtt{encode}(\mathtt{inr}c)(\mathtt{inr}c')\colon \mathtt{code}(\mathtt{inr}c, \mathtt{inr}c') \to \mathtt{code}(\mathtt{inr}c, \mathtt{inr}c')$

But these maps are all identity since $\mathtt{code}(x,y)$ is a proposition for $x,y\colon \mathtt{B}+_{\mathtt{A}}\mathtt{C}$.

We now know that $\mathtt{encode}$ and $\mathtt{decode}$ are mutual inverses. Therefore, $\mathtt{inl}b =_{\mathtt{B}+_{\mathtt{A}}\mathtt{C}} x$ and $\mathtt{inr}c =_{\mathtt{B}+_{\mathtt{A}}\mathtt{C}} x$ are mere propositions for any $x\colon \mathtt{B}+_{\mathtt{A}}\mathtt{C}$. As discussed above, the embedding $\mathtt{P} \hookrightarrow \mathtt{Q}$ is actually an equivlance. Theorem 1 follows.