Solutions to selected exercises

October 24, 2023

Exercises from Chapter 1

Solution to Exercise 1.1

We of course know what composition of functions $f : A \to B$ and $g : B \to C$ should be, but let us see how we might derive it by considering available forms of construction. We want a term

$$g \circ f :\equiv (\square : A \to C),$$

where $\square : A \to C$ indicates that in place of $\square$ we would like to put something of type $A \to C$. Since we are defining a function whose domain is A, we expect it to be of the form

$$g \circ f :\equiv \lambda(x : A).\, (\square : C),$$

so now we are looking for something of type C, with x, f and g available. Of these g looks most promising as it lands in C:

$$g \circ f :\equiv \lambda(x : A).\, g(\square : B).$$

Now we repeat the same trick with f to get

$$g \circ f :\equiv \lambda(x : A).\, g(f(\square : A)).$$

Inside the abstraction x is available and has the type we need, so we define

$$g \circ f :\equiv \lambda(x : A).\, g(f(x)) : C \tag{0.1}$$

This baby example demonstrates how one often works with a proof assistant: look at what you need and what is available, and try to make some progress.

Now, suppose given also $h : C \to D$. We have, according to Eq. (0.1),

$$\begin{aligned} h \circ (g \circ f) &\equiv \lambda x.\, h((\lambda(y : A).\, g(f(y)))x) \\ &\equiv \lambda x.\, h(g(f(x))), \end{aligned}$$

and

$$\begin{aligned} (h \circ g) \circ f &\equiv \lambda x.\, (\lambda(y : A).\, h(g(y)))(f(x)) \\ &\equiv \lambda x.\, h(g(f(x))). \end{aligned}$$

They are equal, which establishes associativity of composition.

Solution to Exercise 1.2

If we suppose given only $\mathsf{pr}_1 : A \times B \to A$ and $\mathsf{pr}_2 : A \times B \to B$ satisfying $\mathsf{pr}_1((a,b)) \equiv a$ and $\mathsf{pr}_2((a,b)) \equiv b$, we can define $\mathsf{rec}'_{A\times B}$ by

$$\mathsf{rec}'_{A\times B}(C, g, x) :\equiv g(\mathsf{pr}_1 x)(\mathsf{pr}_2 x).$$

We can now verify, given $C : \mathcal{U}$, $g : A \to B \to C$ and $(a,b) : A \times B$,

$$\begin{aligned}\mathsf{rec}'_{A\times B}(C, g, (a,b)) &\equiv g(\mathsf{pr}_1(a,b))(\mathsf{pr}_2(a,b)) \\ &\equiv g(a)(b).\end{aligned}$$

For Σ-types we replace $A \times B$ above with $\sum_{(a:A)} B(a)$, but otherwise everything else stays the same:

$$\mathsf{rec}'_{\sum_{(x:A)} B(x)}(C, g, x) :\equiv g(\mathsf{pr}_1 x)(\mathsf{pr}_2 x).$$

Solution to Exercise 1.3

Quite naturally, we form

$$\mathsf{ind}''_{A\times B}(C, g, x) :\equiv g(\mathsf{pr}_1 x)(\mathsf{pr}_2 x),$$

of type

$$\prod_{C:A\times B\to\mathcal{U}} \Big(\prod_{(y:A)} \prod_{(z:B)} C((y,z)) \Big) \to \prod_{x:A\times B} C((\mathsf{pr}_1 x, \mathsf{pr}_2 x)).$$

This is not quite what we need because $\mathsf{ind}_{A\times B}$ has the type

$$\prod_{C:A\times B\to\mathcal{U}} \Big(\prod_{(y:A)} \prod_{(z:B)} C((y,z)) \Big) \to \prod_{x:A\times B} C(x).$$

Recall that we have the propositional uniqueness principle

$$\mathsf{uniq}_{A\times B} : \prod_{x:A\times B} ((\mathsf{pr}_1 x, \mathsf{pr}_2 x) =_{A\times B} x),$$

satisfying $\mathsf{uniq}_{A\times B}((a,b)) = \mathsf{refl}_{(a,b)}$. We can transport along $\mathsf{uniq}_{A\times B}(x)$ to get from $C((\mathsf{pr}_1 x, \mathsf{pr}_2 x))$ to $C(x)$:

$$\mathsf{ind}'_{A\times B}(C, g, x) :\equiv \mathsf{transport}^C(\mathsf{uniq}_{A\times B}(x), \mathsf{ind}''_{A\times B}(C, g, x)).$$

It remains to verify that $\mathsf{ind}'_{A\times B}(C, g, x)$ behaves as expected:

$$\begin{aligned}\mathsf{ind}'_{A\times B}(C, g, (a,b)) &\equiv \mathsf{transport}^C(\mathsf{uniq}_{A\times B}((a,b)), g(\mathsf{pr}_1(a,b))(\mathsf{pr}_2(a,b))) \\ &\equiv \mathsf{transport}^C(\mathsf{uniq}_{A\times B}((a,b)), g(a)(b)) \\ &\equiv \mathsf{transport}^C(\mathsf{refl}_{(a,b)}, g(a)(b)) \\ &\equiv g(a)(b).\end{aligned}$$

Now for Σ-types the exact same expressions work as well, except that the types change.

Exercises from Chapter 2

Solution to Exercise 2.4

In general, when defining an "n-foo", one should ensure that a "1-foo" is just a "foo" (e.g. a 1-category is just a category). In this case, a 1-path in some type A should just be a path, that is, an equality $x =_A y$ between some elements $x, y : A$. Then a 0-path should probably be an element of A. We also know that a 2-path, or homotopy, is a pair of 1-paths and an equality between them. Our definition should extrapolate from this pattern.

Define $C :\equiv \lambda n. \mathcal{U} \to \mathcal{U}$. By the induction principle for $\mathbb{N}$, it suffices to give terms

$$c_0 : \mathcal{U} \to \mathcal{U}$$
$$c_s : \prod_{(n:\mathbb{N})} \prod_{(f:\mathcal{U}\to\mathcal{U})} \mathcal{U} \to \mathcal{U}$$

Define these by

$$c_0 :\equiv \mathsf{id}_{\mathcal{U}}$$
$$c_s(n, f, A) :\equiv \sum_{x,y:f(A)} x = y$$

That is to say, an $(n+1)$-dimensional path should be a pair of n-dimensional paths, together with a path between them. More concisely:

$$\text{npath} : \mathbb{N} \to \mathcal{U} \to \mathcal{U}$$
$$\text{npath} :\equiv \mathsf{ind}_{\mathbb{N}}\Big(\lambda n. \mathcal{U} \to \mathcal{U}, \mathsf{id}_{\mathcal{U}}, \lambda n. \lambda f. \lambda A. \Big(\sum_{x,y:f(A)} x = y\Big)\Big)$$

What should the boundary of an n-path be? The boundary of a 1-path is a pair of points. The boundary of a 2-path (that is, a homotopy) is a pair of 1-paths. Perhaps the boundary of an $(n+1)$-path should be a pair of n-paths. We can get these by using the appropriate projections:

$$\text{nboundary} : \prod_{(n:\mathbb{N})} \prod_{(A:\mathcal{U})} \text{npath}(\mathsf{succ}(n), A) \to \text{npath}(n, A) \times \text{npath}(n, A)$$
$$\text{nboundary}(p) :\equiv (\mathsf{pr}_1(p), \mathsf{pr}_1(\mathsf{pr}_2(p)))$$

Exercises from Chapter 3

Solution to Exercise 3.1

Generally, we can use univalence to transform equivalences to equalities (paths), and use the principle indiscernability of identicals (transport) to show that any statement (family) that holds for one type holds for any type it is equivalent to.

Let $A, B : \mathcal{U}$, $s : \mathsf{isSet}(A)$, and $A \simeq B$. By univalence, there is a path $p : A =_{\mathcal{U}} B$. We can transport s across this path to obtain the desired result:

$$\mathsf{transport}^{X \mapsto \mathsf{isSet}(X)}(p, s) : \mathsf{isSet}(B)$$

Solution to Exercise 3.2

Let $A, B : \mathcal{U}$, and assume that A and B are sets. To show $\mathsf{isSet}(A + B)$, we must show that there is at most one path between elements of $A + B$, up to homotopy. Let $x, y : A + B$. We proceed by case analysis on x and y.

Consider the cases where $x \equiv \mathsf{inl}(a)$ and $y \equiv \mathsf{inr}(b)$ or $x \equiv \mathsf{inr}(b)$ and $y \equiv \mathsf{inl}(a)$ for some $a : A$ and $b : B$. By the characterization of paths in coproduct types (§2.12), these can't be equal. In particular, by (2.12.3), given $p : x = y$, we can conclude anything we like.

Now suppose that $x \equiv \mathsf{inl}(a_1)$ and $y \equiv \mathsf{inl}(a_2)$ for $a_1, a_2 : A$. Then by (2.12.1),

$$(x = y) \equiv (\mathsf{inl}(a_1) = \mathsf{inl}(a_2)) \simeq (a_1 = a_2).$$

Since A is a set, $a_1 = a_2$ is a mere proposition. It follows that $x = y$ is a mere proposition (one can use univalence and transport, as in the previous exercise). A symmetric proof shows that in the case that $x \equiv \mathsf{inr}(b_1)$ and $y \equiv \mathsf{inr}(b_2)$ for $b_1, b_2 : B$, $x = y$ is also a mere proposition. Therefore, $A + B$ is a set.

Solution to Exercise 3.4

($\Rightarrow$) Let $A : \mathcal{U}$ and $P : \mathsf{isProp}(A)$. To show that $A \to A$ is contractible, we must give a center of contraction and show that every other function $A \to A$ is equal to the center. Define our center to be id_A, and let $f : A \to A$. Define a homotopy $\mathsf{id}_A \sim f$ by

$$\alpha : \prod_{x:A} \mathsf{id}_A(x) = f(x)$$
$$\alpha(x) :\equiv P(\mathsf{id}_A(x), f(x))$$

Then by function extensionality, $\mathsf{id}_A = f$.

($\Leftarrow$) Assume $A \to A$ is contractible with center of contraction c, and let $x, y : A$. We want to show that $x = y$. Define $f : A \to A$ by $f(z) :\equiv y$. By contractability of $A \to A$, $\mathsf{id}_A = c = f$. Using happly on the equality between id_A and f, we obtain that $x \equiv \mathsf{id}_A(x) = f(x) \equiv y$, so $x = y$. We have shown that any two elements of A are equal, that is, A is a mere proposition.

Solution to Exercise 3.6

Assume A is a proposition, and let $x, y : A + (\neg A)$. We want to show that $x = y$. We proceed by cases using the induction principle for coproducts.

(i) Assume $x \equiv \mathsf{inl}(a)$ and $y \equiv \mathsf{inr}(n)$. Then $n(a) : \mathbf{0}$ gives us a contradiction.
(ii) Assume $x \equiv \mathsf{inr}(n)$ and $y \equiv \mathsf{inl}(a)$. Then $n(a) : \mathbf{0}$ gives us a contradiction.
(iii) Assume $x \equiv \mathsf{inl}(a_1)$ and $y \equiv \mathsf{inl}(a_2)$. By the characterization of paths in coproduct types (§2.12), we know $(x = y) \simeq (a_1 = a_2)$, and we have $a_1 = a_2$ since A is a proposition.
(iv) Finally, assume $x \equiv \mathsf{inr}(n_1)$ and $y \equiv \mathsf{inr}(n_2)$. Again by the characterization of paths in coproduct types, we know $(x = y) \simeq (n_1 = n_2)$, so it suffices to show $n_1 = n_2$. Since $\neg A \equiv A \to \mathbf{0}$ and $\mathbf{0}$ is a mere proposition, we have by Example 3.6.2 that $\neg A$ is a mere proposition. Therefore, any two elements of $\neg A$ are equal, and in particular, $n_1 = n_2$.

Solution to Exercise 3.7

Assume $h : \neg(A \times B)$, and let $x, y : A + B$. We want to show that $x = y$. We proceed by cases using the induction principle for coproducts.

(i) Assume $x \equiv \mathsf{inl}(a)$ and $y \equiv \mathsf{inr}(b)$ for $a : A$ and $b : B$. Then $h(a, b) : \mathbf{0}$, and we can use the destructor for $\mathbf{0}$ to conclude anything we wish.

(ii) Assume $x \equiv \mathsf{inr}(b)$ and $y \equiv \mathsf{inl}(a)$. Then $h(a, b) : \mathbf{0}$, and we're done.

(iii) Assume $x \equiv \mathsf{inl}(a_1)$ and $y \equiv \mathsf{inl}(a_2)$. By the characterization of paths in coproduct types (§2.12), we know $(x = y) \simeq (a_1 = a_2)$, and we have $a_1 = a_2$ since A is a proposition.

(iv) Assume $x \equiv \mathsf{inr}(b_1)$ and $y \equiv \mathsf{inr}(b_2)$. Just as above, we have $(x = y) \simeq (b_1 = b_2)$, and $b_1 = b_2$.

Solution to Exercise 3.19

The hypotheses imply that

$$\Big(\sum_{n:\mathbb{N}} P(n)\Big) \to \sum_{n:\mathbb{N}} \Big(P(n) \times \prod_{m:\mathbb{N}} \big((m < n) \to \neg P(m)\big)\Big).$$

In words, given n such that $P(n)$, we can find the least such n: we test every $m < n$ in turn, using decidability to do a case analysis, until we find the first one that satisfies $P(m)$. However, the right-hand side of the above implication is a mere proposition: if both n and n' are least numbers satisfying P then they must be equal. Therefore, we also have

$$\Big\Vert\sum_{n:\mathbb{N}} P(n)\Big\Vert \to \sum_{n:\mathbb{N}} \Big(P(n) \times \prod_{m:\mathbb{N}} \big((m < n) \to \neg P(m)\big)\Big)$$

from which the claim follows.

Exercises from Chapter 4

Solution to Exercise 4.2

First note that for any type A we have $\mathsf{isContr}(A) \simeq A \times \mathsf{isContr}(A)$. Thus

$$\begin{aligned}
\prod_{a:A} \mathsf{isContr}\Big(\sum_{b:B} R(a,b)\Big) &\simeq \prod_{a:A} \Big(\sum_{b:B} R(a,b)\Big) \times \mathsf{isContr}\Big(\sum_{b:B} R(a,b)\Big) \\
&\simeq \Big(\prod_{(a:A)} \sum_{(b:B)} R(a,b)\Big) \times \Big(\prod_{a:A} \mathsf{isContr}\Big(\sum_{b:B} R(a,b)\Big)\Big) \\
&\simeq \Big(\sum_{(f:A\to B)} \prod_{(a:A)} R(a, f(a))\Big) \times \Big(\prod_{a:A} \mathsf{isContr}\Big(\sum_{b:B} R(a,b)\Big)\Big)
\end{aligned}$$

using Theorem 2.15.7 at the last step. So the type given in the exercise is equivalent to

$$\sum_{(f:A\to B)} \sum_{(R:A\to B\to \mathcal{U})} \Big(\prod_{a:A} R(a, f(a))\Big) \times \Big(\prod_{a:A} \mathsf{isContr}\Big(\sum_{b:B} R(a,b)\Big)\Big) \times \Big(\prod_{b:B} \mathsf{isContr}\Big(\sum_{a:A} R(a,b)\Big)\Big).$$

It will therefore suffice to show that for any $f : A \to B$, the type

$$\sum_{R:A\to B\to\mathcal{U}} \Big(\prod_{a:A} R(a, f(a))\Big) \times \Big(\prod_{a:A} \mathsf{isContr}\Big(\sum_{b:B} R(a,b)\Big)\Big) \times \Big(\prod_{b:B} \mathsf{isContr}\Big(\sum_{a:A} R(a,b)\Big)\Big). \tag{0.2}$$

is equivalent to $\mathsf{isequiv}(f)$, or equivalently that it satisfies the three desiderata of $\mathsf{isequiv}(f)$.

Firstly, suppose f has a quasi-inverse g, and define $R(a,b) :\equiv (f(a) = b)$. For any a we have $\mathsf{isContr}(\sum_{(b:B)} R(a,b))$ by Lemma 3.11.8, and in particular we have $R(a, f(a))$. On the other hand, by Theorem 2.11.1 we have $R(a,b) \simeq (gf(a) = g(b))$, which is equivalent to $a = g(b)$, so Lemma 3.11.8 also implies that $\mathsf{isContr}(\sum_{(a:A)} R(a,b))$ for any $b : B$.

Secondly, suppose (0.2) is inhabited, i.e. we have $R : A \to B \to \mathcal{U}$ and witnesses $r : \prod_{(a:A)} R(a, f(a))$ and $c : \prod_{(a:A)} \mathsf{isContr}(\sum_{(b:B)} R(a,b))$ and $d : \prod_{(b:B)} \mathsf{isContr}(\sum_{(a:A)} R(a,b))$. Let $g(b) :\equiv \mathsf{pr}_1(\mathsf{pr}_1(d(b)))$, yielding $g : B \to A$; thus we have $R(g(b), b)$ for any $b : B$. Then for any $a_0 : A$ we have $R(a_0, f(a_0))$ and $R(g(f(a_0)), f(a_0))$; but $\sum_{(a:A)} R(a, f(a_0))$ is contractible, so $a_0 = g(f(a_0))$. Similarly, $b_0 = f(g(b_0))$ for any $b_0 : B$, so g is a quasi-inverse to f.

Finally, we must show that (0.2) is a mere proposition. Since $\prod_{(b:B)} \mathsf{isContr}\Big(\sum_{(a:A)} R(a,b)\Big)$ is a mere proposition by Example 3.6.2 and Lemma 3.11.4, by Lemma 3.5.1 we may ignore it and consider only the remainder:

$$\sum_{R:A\to B\to\mathcal{U}} \Big(\prod_{a:A} R(a, f(a))\Big) \times \Big(\prod_{a:A} \mathsf{isContr}\Big(\sum_{b:B} R(a,b)\Big)\Big).$$

Using Theorem 2.15.7 again, this is equivalent to

$$\prod_{(a:A)} \sum_{(R:B\to\mathcal{U})} R(fa) \times \mathsf{isContr}\Big(\sum_{b:B} R(b)\Big).$$

Thus it will suffice to show that for any $b_0 : B$, the type

$$\sum_{R:B\to\mathcal{U}} R(b_0) \times \mathsf{isContr}\Big(\sum_{b:B} R(b)\Big)$$

is a mere proposition. But in fact, this type is contractible; its center of contraction consists of $\lambda b.\,(b_0 = b)$ and refl_{b_0} and Lemma 3.11.8, and the contracting homotopy arises from Theorem 5.8.2(iv)⇒(iii) (together with univalence and function extensionality).

Solution to Exercise 4.7

An embedding clearly has properties (i) and (ii). Conversely, suppose f has properties (i) and (ii) and let $x, y : A$; we must show that $\mathsf{ap}_f : (x = y) \to (f(x) = f(y))$ is an equivalence. By Corollary 4.4.6, we are free to assume that $f(x) = f(y)$. Thus, by (i), we have some $p : x = y$. Now the following square commutes by Lemma 2.2.2:

$$\begin{tikzcd} \Omega(A,y) \arrow[r, "p \cdot -"] \arrow[d, "\mathsf{ap}_f"'] & (x = y) \arrow[d, "\mathsf{ap}_f"] \\ \Omega(B, f(y)) \arrow[r, "f(p) \cdot -"'] & (f(x) = f(y)). \end{tikzcd}$$

Both horizontal maps are equivalences by Exercise 2.6, while the left-hand vertical map is an equivalence by (ii). Thus, by Theorem 4.7.1, so is the right-hand vertical map, as desired.

As for examples, the unique map $\mathbf{2} \to \mathbf{1}$ satisfies (ii) but not (i), while the map $\lambda x. \mathbf{2} : \mathbf{1} \to \mathcal{U}$ satisfies (i) but not (ii).

Exercises from Chapter 6

Solution to Exercise 6.1

The torus T^2 is a higher inductive type generated by a point $b : T^2$, two paths $p : b = b$, $q : b = b$, and a 2-path $t : p \cdot q = q \cdot p$. The recursion principle thus says that given $C : \mathcal{U}$, for a function $f : T^2 \to C$ we require

- a point $b' : C$,
- a path $p' : b' = b'$,
- a path $q' : b' = b'$, and
- a 2-path $t' : p' \cdot q' = q' \cdot p'$.

The recursor $f : T^2 \to C$ then has the property that $f(b) \equiv b'$. Furthermore, there exist terms $\beta : f(p) = p'$ and $\gamma : f(q) = q'$ such that the following diagram commutes:

$$\begin{tikzcd}
f(p \cdot q) \arrow[r, dash, "\mathsf{ap}_{\mathsf{ap}_f}(t)"] \arrow[d, dash] & f(q \cdot p) \arrow[d, dash] \\
f(p) \cdot f(q) \arrow[r, phantom, "="] \arrow[d, dash, "\text{via } \beta, \gamma"'] & f(q) \cdot f(p) \arrow[d, dash, "\text{via } \beta, \gamma"] \\
p' \cdot q' \arrow[r, dash, "t'"'] & q' \cdot p'
\end{tikzcd}$$

The induction principle is more complicated; it says that given a family $P : T^2 \to \mathcal{U}$, for a section $f : \prod_{(x:T^2)} P(x)$ we require

- a point $b' : P(b)$,
- a path $p' : p_*(b') = b'$,
- a path $q' : q_*(b') = b'$, and
- a 2-path t' witnessing the equality of the following two paths from $(q \cdot p)_*(b')$ to b':

  $$\left(\mathsf{ap}_{\alpha \mapsto \alpha_*(b')}(t)\right)^{-1} \cdot \left(\mathsf{happly}_{\mathcal{I}_P(p,q)}(b') \cdot \mathsf{ap}_{q_*}(p') \cdot q'\right)$$
  $$\mathsf{happly}_{\mathcal{I}_P(q,p)}(b') \cdot \mathsf{ap}_{p_*}(q') \cdot p'$$

  where for any type family $B : A \to \mathcal{U}$ and paths $\alpha : x =_A y$ and $\alpha' : y =_A z$, the path

  $$\mathcal{I}_E(\alpha, \alpha') : (\alpha \cdot \alpha')_* = \lambda(u : B(x)).\, \alpha'_*(\alpha_*(u))$$

  is obtained by a path induction on α and α'.

The inductor $f : \prod_{(x:T^2)} P(x)$ then has the property that $f(b) \equiv b'$. Furthermore, there exist terms $\beta : \mathsf{apd}_f(p) = p'$ and $\gamma : \mathsf{apd}_f(q) = q'$ such that the 2-path

$$\mathsf{apd}_{\mathsf{apd}_f}(t) : \mathsf{transport}^{\alpha \mapsto \alpha_*(b') = b'}(t, \mathsf{apd}_f(p \cdot q)) = \mathsf{apd}_f(q \cdot p)$$

is equal to the 2-path

$$\begin{array}{c}
\mathsf{transport}^{\alpha \mapsto \alpha_*(b') = b'}(t, \mathsf{apd}_f(p \cdot q)) \\
\Big| \; \mathcal{T}^{b'}_{\alpha \mapsto \alpha_*(b')}(t, \mathsf{apd}_f(p \cdot q)) \\
\big(\mathsf{ap}_{\alpha \mapsto \alpha_*(b')}(t)\big)^{-1} \cdot \mathsf{apd}_f(p \cdot q) \\
\Big| \; \text{via } \mathcal{D}_f(p, q) \\
\big(\mathsf{ap}_{\alpha \mapsto \alpha_*(b')}(t)\big)^{-1} \cdot \big(\mathsf{happly}_{\mathcal{I}_P(p,q)}(b') \cdot \mathsf{ap}_{q_*}(\mathsf{apd}_f(p)) \cdot \mathsf{apd}_f(q)\big) \\
\Big| \; \text{via } \beta, \gamma \\
\big(\mathsf{ap}_{\alpha \mapsto \alpha_*(b')}(t)\big)^{-1} \cdot \big(\mathsf{happly}_{\mathcal{I}_P(p,q)}(b') \cdot \mathsf{ap}_{q_*}(p') \cdot q'\big) \\
\Big| \; t' \\
\mathsf{happly}_{\mathcal{I}_P(q,p)}(b') \cdot \mathsf{ap}_{p_*}(q') \cdot p' \\
\Big| \; \text{via } \beta^{-1}, \gamma^{-1} \\
\mathsf{happly}_{\mathcal{I}_P(q,p)}(b') \cdot \mathsf{ap}_{p_*}(\mathsf{apd}_f(q)) \cdot \mathsf{apd}_f(p) \\
\Big| \; \mathcal{D}_f(q, p)^{-1} \\
\mathsf{apd}_f(q \cdot p)
\end{array}$$

where for any $g : A \to B$, $c : B$, $\alpha : a =_A a'$ and $u : g(a) =_B c$, the path

$$\mathcal{T}^c_g(\alpha, u) : \mathsf{transport}^{x \to g(x) = c}(\alpha, u) = g(\alpha)^{-1} \cdot u$$

is obtained by a straightforward path induction on α. Similarly, for any $g : \prod_{(x:A)} B(x)$ and paths $\alpha : x =_A y$, $\alpha' : y =_A z$, the path

$$\mathcal{D}_g(\alpha, \alpha') : \mathsf{apd}_g(\alpha \cdot \alpha') = \mathsf{happly}_{\mathcal{I}_B(\alpha, \alpha')}(g(x)) \cdot \mathsf{ap}_{\alpha'_*}(\mathsf{apd}_g(\alpha)) \cdot \mathsf{apd}_g(\alpha')$$

is obtained by a path induction on α and α'.

Solution to Exercise 6.3

Logical equivalence between $\mathbb{S}^1 \times \mathbb{S}^1$ and T^2

We define a function $f : \mathbb{S}^1 \to T^2$ by circle recursion, mapping $\mathsf{base} \mapsto b$ and $\mathsf{loop} \mapsto p$. We define a function $F^{\to} : \mathbb{S}^1 \to \mathbb{S}^1 \to T^2$ again by circle recursion, mapping $\mathsf{base} \mapsto f$ and $\mathsf{loop} \mapsto \mathsf{funext}(H)$, where $H : \prod_{(x:\mathbb{S}^1)} f(x) = f(x)$ is defined by circle induction as follows. We map base to q and loop to the path

$$
\begin{array}{c}
\mathsf{transport}^{z \mapsto f(z)=f(z)}(\mathsf{loop}, q) \\
\Big| \; \mathcal{T}_1(\mathsf{loop}, q) \\
f(\mathsf{loop})^{-1} \cdot (q \cdot f(\mathsf{loop})) \\
\Big| \; \mathcal{I}_1(\delta) \\
q
\end{array}
$$

where for any $\alpha : x =_{\mathbb{S}^1} y$, and $u : f(x) = f(x)$, the path

$$\mathcal{T}_1(\alpha, u) : \mathsf{transport}^{z \mapsto f(z)=f(z)}(\alpha, u) = f(\alpha)^{-1} \cdot u \cdot f(\alpha)$$

is obtained by a straightforward path induction on α. For any $u : a =_A b$, $v : b =_A d$, $w : a =_A c$, $z : c =_A d$, we have functions

$$
\begin{aligned}
\mathcal{I}_1 &: (u \cdot v = w \cdot z) \to (u^{-1} \cdot w \cdot z = v) \\
\mathcal{I}_1^{-1} &: (u^{-1} \cdot w \cdot z = v) \to (u \cdot v = w \cdot z)
\end{aligned}
$$

defined by path induction on u and z, which form a quasi-equivalence. Finally, δ is the path

$$
\begin{array}{c}
f(\mathsf{loop}) \cdot q \\
\Big| \; \text{via } \beta_f \\
p \cdot q \\
\Big| \; t \\
q \cdot p \\
\Big| \; \text{via } \beta_f \\
q \cdot f(\mathsf{loop})
\end{array}
$$

where $\beta_f : f(\mathsf{loop}) = p$ witnesses the second computation rule for the circle.

Having defined a function $F^{\to} : \mathbb{S}^1 \to \mathbb{S}^1 \to T^2$, it is now straightforward to define a function $F^{\times} : \mathbb{S}^1 \times \mathbb{S}^1 \to T^2$. For the other direction, we define $G : T^2 \to \mathbb{S}^1 \times \mathbb{S}^1$ by torus recursion as follows. We map $b \mapsto (\mathsf{base}, \mathsf{base})$, $p \mapsto \mathsf{pair}^{=}(\mathsf{refl}_{\mathsf{base}}, \mathsf{loop})$, $q \mapsto \mathsf{pair}^{=}(\mathsf{loop}, \mathsf{refl}_{\mathsf{base}})$, and $t \mapsto \Phi_{\mathsf{loop},\mathsf{loop}}$, where for $\alpha : x =_A x'$ and $\alpha' : y =_A y'$,

$$\Phi_{\alpha,\alpha'} : \left(\mathsf{pair}^{=}(\mathsf{refl}_x, \alpha') \cdot \mathsf{pair}^{=}(\alpha, \mathsf{refl}_{y'})\right) = \left(\mathsf{pair}^{=}(\alpha, \mathsf{refl}_y) \cdot \mathsf{pair}^{=}(\mathsf{refl}_{x'}, \alpha')\right)$$

is defined by induction on α'.

This completes the definition of a logical equivalence between $\mathbb{S}^1 \times \mathbb{S}^1$ and T^2. Before we proceed to show that it is in fact a quasi-equivalence, we note a few key properties of the functions H, $F^{\times}$, G constructed above.

The 1-path computation rule for $F^{\rightarrow}$ gives us a term

$$\beta_{F^{\rightarrow}} : F^{\rightarrow}(\mathsf{loop}) = \mathsf{funext}(H)$$

The 1-path computation rules for G give us terms

$$\begin{aligned} \beta_G &: G(p) = \mathsf{pair}^{=}(\mathsf{refl}_{\mathsf{base}}, \mathsf{loop}) \\ \gamma_G &: G(q) = \mathsf{pair}^{=}(\mathsf{loop}, \mathsf{refl}_{\mathsf{base}}) \end{aligned}$$

The 2-path computation rule for G gives us the following commuting diagram:

$$\begin{tikzcd}
G(p \cdot q) \arrow[rr, dash, "\text{via } t"] \arrow[d, dash] & & G(q \cdot p) \arrow[d, dash] \\
G(p) \cdot G(q) \arrow[d, dash, "\text{via } \beta_G, \gamma_G"'] \arrow[rr, phantom, "(1)"] & & G(q) \cdot G(p) \arrow[d, dash, "\text{via } \beta_G, \gamma_G"] \\
\mathsf{pair}^{=}(\mathsf{refl}, \mathsf{loop}) \cdot \mathsf{pair}^{=}(\mathsf{loop}, \mathsf{refl}) \arrow[rr, dash, "\Phi_{\mathsf{loop},\mathsf{loop}}"'] & & \mathsf{pair}^{=}(\mathsf{loop}, \mathsf{refl}) \cdot \mathsf{pair}^{=}(\mathsf{refl}, \mathsf{loop})
\end{tikzcd}$$

For any $\alpha : x =_{T^2} x'$ and $\alpha' : y =_{T^2} y'$, we have path families

$$\begin{aligned} \mu(\alpha') &: F^{\times}\big(\mathsf{pair}^{=}(\mathsf{refl}_x, \alpha')\big) = F^{\rightarrow}(x)(\alpha') \\ \nu(\alpha) &: F^{\times}\big(\mathsf{pair}^{=}(\alpha, \mathsf{refl}_y)\big) = \mathsf{happly}_{F^{\rightarrow}(\alpha)}(y) \end{aligned}$$

defined by path induction on α and α'.

The function H is a homotopy between f and f. As such, for any path $\alpha : x =_{\mathbb{S}^1} y$, there exists a 2-path

$$\mathsf{nat}_H(\alpha) : f(\alpha) \cdot H(y) = H(x) \cdot f(\alpha)$$

defined by induction on α. In the case when $\alpha :\equiv \mathsf{loop}$, we can show that the following diagram commutes:

$$\begin{tikzcd}
f(\mathsf{loop}) \cdot q \arrow[r, dash, "\text{via } \beta_f"] \arrow[d, dash, "\mathsf{nat}_H(\mathsf{loop})"'] \arrow[dr, phantom, "(2)"] & p \cdot q \arrow[d, dash, "t"] \\
q \cdot f(\mathsf{loop}) \arrow[r, dash, "\text{via } \beta_f"'] & q \cdot p
\end{tikzcd}$$

To show this, we note that for any $\alpha : x =_{\mathbb{S}^1} y$, applying $\mathcal{I}_1^{-1}$ to the path

$$\begin{tikzcd}
f(\alpha)^{-1} \cdot H(x) \cdot f(\alpha) \arrow[d, dash, "\mathcal{T}_1(\alpha, H(x))^{-1}"] \\
\mathsf{transport}^{z \mapsto f(z) = f(z)}(\alpha, H(x)) \arrow[d, dash, "\mathsf{apd}_H(\alpha)"] \\
H(y)
\end{tikzcd}$$

yields precisely $\mathsf{nat}_H(\alpha)$ (by a simple path induction on α). The second computation rule for H tells us that $\mathsf{apd}_H(\mathsf{loop}) = \mathcal{T}_1(\mathsf{loop}, q) \cdot \mathcal{I}_1(\delta)$. Thus

$$\mathsf{nat}_H(\mathsf{loop}) = \mathcal{I}_1^{-1}\big(\mathcal{T}_1(\mathsf{loop}, q)^{-1} \cdot \mathsf{apd}_H(\mathsf{loop})\big) = \delta$$

which proves the commutativity of (2).

Equivalence between $\mathbb{S}^1 \times \mathbb{S}^1$ and T^2

Left-to-right We need to show that for any $x, y : \mathbb{S}^1$ we have $G(F^{\times}(x,y)) = (x,y)$. To use the circle induction, we first define a path family $\epsilon : \prod_{(y:\mathbb{S}^1)} G(f(y)) = (\mathsf{base}, y)$. The definition of ϵ itself proceeds by circle induction: we map base to the path $\mathsf{refl}_{(\mathsf{base},\mathsf{base})}$ and loop to the path

$$\begin{array}{c}
\mathsf{transport}^{z \mapsto G(f(z)) = (\mathsf{base}, z)}(\mathsf{loop}, \mathsf{refl}) \\
\Big| \; \mathcal{T}_2(\mathsf{loop}, \mathsf{refl}) \\
G(f(\mathsf{loop}))^{-1} \cdot \mathsf{refl} \cdot \mathsf{pair}^{=}(\mathsf{refl}, \mathsf{loop}) \\
\Big| \; \mathcal{I}_1(\kappa) \\
\mathsf{refl}
\end{array}$$

where for any $\alpha : x =_{\mathbb{S}^1} y$ and $u : G(f(x)) = (\mathsf{base}, x)$, the path

$$\mathcal{T}_2(\alpha, u) : \mathsf{transport}^{z \mapsto G(f(z)) = (\mathsf{base}, z)}(\alpha, u) = G(f(\alpha))^{-1} \cdot u \cdot \mathsf{pair}^{=}(\mathsf{refl}, \alpha)$$

is defined by path induction on α. Finally, κ is the path

$$\begin{array}{c}
G(f(\mathsf{loop})) \cdot \mathsf{refl} \\
\Big| \\
G(f(\mathsf{loop})) \\
\Big| \; \text{via } \beta_f, \beta_G \\
\mathsf{pair}^{=}(\mathsf{refl}, \mathsf{loop}) \\
\Big| \\
\mathsf{refl} \cdot \mathsf{pair}^{=}(\mathsf{refl}, \mathsf{loop})
\end{array}$$

This finishes the definition of ϵ. As before, for any $\alpha : x =_{\mathbb{S}^1} y$ we have a 2-path

$$\mathsf{nat}_{\epsilon}(\alpha) : G(f(\alpha)) \cdot \epsilon(y) = \epsilon(x) \cdot \mathsf{pair}^{=}(\mathsf{refl}, \alpha)$$

defined by induction on α. In the case $\alpha :\equiv \mathsf{loop}$, the following diagram commutes:

$$\begin{tikzcd}
G(f(\mathsf{loop})) \cdot \mathsf{refl} \arrow[r, no head] \arrow[d, no head, "\mathsf{nat}_\epsilon(\mathsf{loop})"'] \arrow[dr, phantom, "(3)"] & G(f(\mathsf{loop})) \arrow[d, no head, "\text{via } \beta_f, \beta_G"] \\
\mathsf{refl} \cdot \mathsf{pair}^{=}(\mathsf{refl}, \mathsf{loop}) \arrow[r, no head] & \mathsf{pair}^{=}(\mathsf{refl}, \mathsf{loop})
\end{tikzcd}$$

To show this, we note that for any $\alpha : x =_{\mathbb{S}^1} y$, applying $\mathcal{I}_1^{-1}$ to the path

$$\begin{tikzcd}
G(f(\alpha))^{-1} \cdot \epsilon(x) \cdot \mathsf{pair}^{=}(\mathsf{refl}, \alpha) \arrow[d, no head, "\mathcal{T}_2(\alpha, \epsilon(x))^{-1}"] \\
\mathsf{transport}^{z \mapsto G(f(z)) = (\mathsf{base}, z)}(\alpha, \epsilon(x)) \arrow[d, no head, "\mathsf{apd}_\epsilon(\alpha)"] \\
\epsilon(y)
\end{tikzcd}$$

yields precisely $\mathsf{nat}_\epsilon(\alpha)$ (by a simple path induction on α). The second computation rule for ϵ tells us that $\mathsf{apd}_\epsilon(\mathsf{loop}) = \mathcal{T}_2(\mathsf{loop}, \mathsf{refl}) \cdot \mathcal{I}_1(\kappa)$. Thus

$$\mathsf{nat}_\epsilon(\mathsf{loop}) = \mathcal{I}_1^{-1}\left(\mathcal{T}_2(\mathsf{loop}, \mathsf{refl})^{-1} \cdot \mathsf{apd}_\epsilon(\mathsf{loop})\right) = \kappa$$

which proves the commutativity of (3).

All that remains now is to prove that

$$\mathsf{transport}^{x \mapsto \prod_{(y:\mathbb{S}^1)} G(F^\times(x,y)) = (x,y)}(\mathsf{loop}, \epsilon) = \epsilon$$

The left endpoint can be expressed explicitly as the function

$$y \mapsto G\Big(\mathsf{happly}_{F^\to(\mathsf{loop})}(y)\Big)^{-1} \cdot \epsilon(y) \cdot \mathsf{pair}^{=}(\mathsf{loop}, \mathsf{refl})$$

as a generalization of loop to an arbitrary α and a subsequent path induction on α shows. By function extensionality it thus suffices to show that for any $y : \mathbb{S}^1$, we have

$$G\Big(\mathsf{happly}_{F^\to(\mathsf{loop})}(y)\Big)^{-1} \cdot \epsilon(y) \cdot \mathsf{pair}^{=}(\mathsf{loop}, \mathsf{refl}) = \epsilon(y)$$

The left endpoint can be simplified using $\beta_{F^\to}$ and the fact that happly and funext form a quasi-inverse:

$$G(H(y))^{-1} \cdot \epsilon(y) \cdot \mathsf{pair}^{=}(\mathsf{loop}, \mathsf{refl}) = \epsilon(y)$$

Showing the above is the same as showing

$$G(H(y)) \cdot \epsilon(y) = \epsilon(y) \cdot \mathsf{pair}^{=}(\mathsf{loop}, \mathsf{refl})$$

for any $y : \mathbb{S}^1$. We proceed yet again by circle induction. We map base to the path η below:

$$
\begin{array}{c}
G(q) \cdot \mathsf{refl} \\
\Big| \\
G(q) \\
\Big|\, \gamma_G \\
\mathsf{pair}^{=}(\mathsf{loop}, \mathsf{refl}) \\
\Big| \\
\mathsf{refl} \cdot \mathsf{pair}^{=}(\mathsf{loop}, \mathsf{refl})
\end{array}
$$

Now it remains to show that

$$\mathsf{transport}^{z \mapsto G(H(z)) \cdot \epsilon(z) = \epsilon(z) \cdot \mathsf{pair}^{=}(\mathsf{loop},\mathsf{refl})}(\mathsf{loop}, \eta) = \eta$$

For any $u : a =_A b$, $v : b =_A d$, $w : a =_A c$, $z : c =_A d$, we have functions

$$
\begin{aligned}
\mathcal{I}_2 &: (u \cdot v = w \cdot z) \to (v \cdot z^{-1} = u^{-1} \cdot w) \\
\mathcal{I}_2^{-1} &: (v \cdot z^{-1} = u^{-1} \cdot w) \to (u \cdot v = w \cdot z)
\end{aligned}
$$

defined by induction on u and z, which form a quasi-equivalence. For any $\alpha : x =_{\mathbb{S}^1} y$, let $\delta^\star(\alpha)$ be the path

$$
\begin{array}{c}
G(f(\alpha)) \cdot G(H_y) \\
\Big| \\
G(f(\alpha) \cdot H_y) \\
\Big|\, \text{via } \mathsf{nat}_H(\alpha) \\
G(H_x \cdot f(\alpha)) \\
\Big| \\
G(H_x) \cdot G(f(\alpha))
\end{array}
$$

Given any $\alpha : x =_{\mathbb{S}^1} y$ and $\eta' : G(H(x)) \cdot \eta(x) = \eta(x) \cdot \mathsf{pair}^{=}(\mathsf{loop}, \mathsf{refl})$, we can now express the path $\mathsf{transport}^{z \mapsto G(H(z)) \cdot \epsilon(z) = \epsilon(z) \cdot \mathsf{pair}^{=}(\mathsf{loop},\mathsf{refl})}(\alpha, \eta')$ explicitly as the following path:

$$
\begin{array}{c}
G(H_y) \cdot \epsilon_y \\
\Big| \\
G(H_y) \cdot \Big(G(f(\alpha))^{-1} \cdot G(f(\alpha))\Big) \cdot \epsilon_y \\
\Big| \\
\Big(G(H_y) \cdot G(f(\alpha))^{-1}\Big) \cdot \Big(G(f(\alpha)) \cdot \epsilon_y\Big) \\
\Big| \ \text{via } \mathcal{I}_2(\delta^\star(\alpha)) \\
\Big(G(f(\alpha))^{-1} \cdot G(H_x)\Big) \cdot \Big(G(f(\alpha)) \cdot \epsilon_y\Big) \\
\Big| \ \text{via } \mathsf{nat}_\epsilon(\alpha) \\
\Big(G(f(\alpha))^{-1} \cdot G(H_x)\Big) \cdot \Big(\epsilon_x \cdot \mathsf{pair}^=(\mathsf{refl}, \alpha)\Big) \\
\Big| \\
G(f(\alpha))^{-1} \cdot \Big(G(H_x) \cdot \epsilon_x\Big) \cdot \mathsf{pair}^=(\mathsf{refl}, \alpha) \\
\Big| \ \text{via } \eta' \\
G(f(\alpha))^{-1} \cdot \Big(\epsilon_x \cdot \mathsf{pair}^=(\mathsf{loop}, \mathsf{refl})\Big) \cdot \mathsf{pair}^=(\mathsf{refl}, \alpha) \\
\Big| \\
\Big(G(f(\alpha))^{-1} \cdot \epsilon_x\Big) \cdot \Big(\mathsf{pair}^=(\mathsf{loop}, \mathsf{refl}) \cdot \mathsf{pair}^=(\mathsf{refl}, \alpha)\Big) \\
\Big| \ \text{via } \Phi^{-1}_{\mathsf{loop},\alpha} \\
\Big(G(f(\alpha))^{-1} \cdot \epsilon_x\Big) \cdot \Big(\mathsf{pair}^=(\mathsf{refl}, \alpha) \cdot \mathsf{pair}^=(\mathsf{loop}, \mathsf{refl})\Big) \\
\Big| \\
G(f(\alpha))^{-1} \cdot \Big(\epsilon_x \cdot \mathsf{pair}^=(\mathsf{refl}, \alpha)\Big) \cdot \mathsf{pair}^=(\mathsf{loop}, \mathsf{refl}) \\
\Big| \ \text{via } \mathsf{nat}_\epsilon(\alpha)^{-1} \\
G(f(\alpha))^{-1} \cdot \Big(G(f(\alpha)) \cdot \epsilon_y\Big) \cdot \mathsf{pair}^=(\mathsf{loop}, \mathsf{refl}) \\
\Big| \\
\Big(G(f(\alpha))^{-1} \cdot G(f(\alpha))\Big) \cdot \Big(\epsilon_y \cdot \mathsf{pair}^=(\mathsf{loop}, \mathsf{refl})\Big) \\
\Big| \\
\epsilon_y \cdot \mathsf{pair}^=(\mathsf{loop}, \mathsf{refl})
\end{array}
$$

In the case $\alpha :\equiv \mathsf{loop}$ and $\eta' :\equiv \eta$ we thus have:

$$
\begin{array}{c}
G(q) \cdot \mathsf{refl} \\
\Big| \\
G(q) \cdot \Big(G(f(\mathsf{loop}))^{-1} \cdot G(f(\mathsf{loop})) \Big) \cdot \mathsf{refl} \\
\Big| \\
\Big(G(q) \cdot G(f(\mathsf{loop}))^{-1} \Big) \cdot \Big(G(f(\mathsf{loop})) \cdot \mathsf{refl} \Big) \\
\Big| \ \text{via } \mathcal{I}_2(\delta^\star(\mathsf{loop})) \\
\Big(G(f(\mathsf{loop}))^{-1} \cdot G(q) \Big) \cdot \Big(G(f(\mathsf{loop})) \cdot \mathsf{refl} \Big) \\
\Big| \ \text{via } \mathsf{nat}_\epsilon(\mathsf{loop}) \\
\Big(G(f(\mathsf{loop}))^{-1} \cdot G(q) \Big) \cdot \Big(\mathsf{refl} \cdot \mathsf{pair}^{=}(\mathsf{refl}, \mathsf{loop}) \Big) \\
\Big| \\
G(f(\mathsf{loop}))^{-1} \cdot \Big(G(q) \cdot \mathsf{refl} \Big) \cdot \mathsf{pair}^{=}(\mathsf{refl}, \mathsf{loop}) \\
\Big| \ \text{via } \eta \\
G(f(\mathsf{loop}))^{-1} \cdot \Big(\mathsf{refl} \cdot \mathsf{pair}^{=}(\mathsf{loop}, \mathsf{refl}) \Big) \cdot \mathsf{pair}^{=}(\mathsf{refl}, \mathsf{loop}) \\
\Big| \\
\Big(G(f(\mathsf{loop}))^{-1} \cdot \mathsf{refl} \Big) \cdot \Big(\mathsf{pair}^{=}(\mathsf{loop}, \mathsf{refl}) \cdot \mathsf{pair}^{=}(\mathsf{refl}, \mathsf{loop}) \Big) \\
\Big| \ \text{via } \Phi^{-1}_{\mathsf{loop},\mathsf{loop}} \\
\Big(G(f(\mathsf{loop}))^{-1} \cdot \mathsf{refl} \Big) \cdot \Big(\mathsf{pair}^{=}(\mathsf{refl}, \mathsf{loop}) \cdot \mathsf{pair}^{=}(\mathsf{loop}, \mathsf{refl}) \Big) \\
\Big| \\
G(f(\mathsf{loop}))^{-1} \cdot \Big(\mathsf{refl} \cdot \mathsf{pair}^{=}(\mathsf{refl}, \mathsf{loop}) \Big) \cdot \mathsf{pair}^{=}(\mathsf{loop}, \mathsf{refl}) \\
\Big| \ \text{via } \mathsf{nat}_\epsilon(\mathsf{loop})^{-1} \\
G(f(\mathsf{loop}))^{-1} \cdot \Big(G(f(\mathsf{loop})) \cdot \mathsf{refl} \Big) \cdot \mathsf{pair}^{=}(\mathsf{loop}, \mathsf{refl}) \\
\Big| \\
\Big(G(f(\mathsf{loop}))^{-1} \cdot G(f(\mathsf{loop})) \Big) \cdot \Big(\mathsf{refl} \cdot \mathsf{pair}^{=}(\mathsf{loop}, \mathsf{refl}) \Big) \\
\Big| \\
\mathsf{refl} \cdot \mathsf{pair}^{=}(\mathsf{loop}, \mathsf{refl})
\end{array}
$$

We can now use the commutativity of (3) and get rid of the extraneous identity paths to obtain:

$$
\begin{array}{c}
G(q) \cdot \mathsf{refl} \\
\Big| \\
G(q) \\
\Big| \\
G(q) \cdot \Big(G(f(\mathsf{loop}))^{-1} \cdot G(f(\mathsf{loop})) \Big) \\
\Big| \\
\Big(G(q) \cdot G(f(\mathsf{loop}))^{-1} \Big) \cdot G(f(\mathsf{loop})) \\
\Big| \text{ via } \mathcal{I}_2(\delta^\star(\mathsf{loop})) \\
\Big(G(f(\mathsf{loop}))^{-1} \cdot G(q) \Big) \cdot G(f(\mathsf{loop})) \\
\Big| \text{ via } \beta_f, \beta_G \\
\Big(G(f(\mathsf{loop}))^{-1} \cdot G(q) \Big) \cdot \mathsf{pair}^{=}(\mathsf{refl}, \mathsf{loop}) \\
\Big| \text{ via } \gamma_G \\
\Big(G(f(\mathsf{loop}))^{-1} \cdot \mathsf{pair}^{=}(\mathsf{loop}, \mathsf{refl}) \Big) \cdot \mathsf{pair}^{=}(\mathsf{refl}, \mathsf{loop}) \\
\Big| \\
G(f(\mathsf{loop}))^{-1} \cdot \Big(\mathsf{pair}^{=}(\mathsf{loop}, \mathsf{refl}) \cdot \mathsf{pair}^{=}(\mathsf{refl}, \mathsf{loop}) \Big) \\
\Big| \text{ via } \Phi^{-1}_{\mathsf{loop},\mathsf{loop}} \\
G(f(\mathsf{loop}))^{-1} \cdot \Big(\mathsf{pair}^{=}(\mathsf{refl}, \mathsf{loop}) \cdot \mathsf{pair}^{=}(\mathsf{loop}, \mathsf{refl}) \Big) \\
\Big| \\
\Big(G(f(\mathsf{loop}))^{-1} \cdot \mathsf{pair}^{=}(\mathsf{refl}, \mathsf{loop}) \Big) \cdot \mathsf{pair}^{=}(\mathsf{loop}, \mathsf{refl}) \\
\Big| \text{ via } \beta_G^{-1}, \beta_f^{-1} \\
\Big(G(f(\mathsf{loop}))^{-1} \cdot G(f(\mathsf{loop})) \Big) \cdot \mathsf{pair}^{=}(\mathsf{loop}, \mathsf{refl}) \\
\Big| \\
\mathsf{pair}^{=}(\mathsf{loop}, \mathsf{refl}) \\
\Big| \\
\mathsf{refl} \cdot \mathsf{pair}^{=}(\mathsf{loop}, \mathsf{refl})
\end{array}
$$

or equivalently:

$$
\begin{array}{c}
G(q) \cdot \mathsf{refl} \\
\Big| \\
G(q) \\
\Big| \\
G(q) \cdot \Big(G(f(\mathsf{loop}))^{-1} \cdot G(f(\mathsf{loop})) \Big) \\
\Big| \\
\Big(G(q) \cdot G(f(\mathsf{loop}))^{-1} \Big) \cdot G(f(\mathsf{loop})) \\
\Big| \ \text{via } \mathcal{I}_2(\delta^\star(\mathsf{loop})) \\
\Big(G(f(\mathsf{loop}))^{-1} \cdot G(q) \Big) \cdot G(f(\mathsf{loop})) \\
\Big| \ \text{via } \beta_f, \beta_G \\
\Big(G(f(\mathsf{loop}))^{-1} \cdot G(q) \Big) \cdot \mathsf{pair}^{=}(\mathsf{refl}, \mathsf{loop}) \\
\Big| \ \text{via } \gamma_G \\
\Big(G(f(\mathsf{loop}))^{-1} \cdot \mathsf{pair}^{=}(\mathsf{loop}, \mathsf{refl}) \Big) \cdot \mathsf{pair}^{=}(\mathsf{refl}, \mathsf{loop}) \\
\Big| \\
G(f(\mathsf{loop}))^{-1} \cdot \Big(\mathsf{pair}^{=}(\mathsf{loop}, \mathsf{refl}) \cdot \mathsf{pair}^{=}(\mathsf{refl}, \mathsf{loop}) \Big) \\
\Big| \ \text{via } \Phi^{-1}_{\mathsf{loop},\mathsf{loop}} \\
G(f(\mathsf{loop}))^{-1} \cdot \Big(\mathsf{pair}^{=}(\mathsf{refl}, \mathsf{loop}) \cdot \mathsf{pair}^{=}(\mathsf{loop}, \mathsf{refl}) \Big) \\
\Big| \\
\Big(G(f(\mathsf{loop}))^{-1} \cdot \mathsf{pair}^{=}(\mathsf{refl}, \mathsf{loop}) \Big) \cdot \mathsf{pair}^{=}(\mathsf{loop}, \mathsf{refl}) \\
\Big| \ \text{via } \beta_f, \beta_G \\
\Big(\mathsf{pair}^{=}(\mathsf{refl}, \mathsf{loop})^{-1} \cdot \mathsf{pair}^{=}(\mathsf{refl}, \mathsf{loop}) \Big) \cdot \mathsf{pair}^{=}(\mathsf{loop}, \mathsf{refl}) \\
\Big| \\
\mathsf{pair}^{=}(\mathsf{loop}, \mathsf{refl}) \\
\Big| \\
\mathsf{refl} \cdot \mathsf{pair}^{=}(\mathsf{loop}, \mathsf{refl})
\end{array}
$$

After some rearranging we get:

$$
\begin{array}{c}
G(q) \cdot \mathsf{refl} \\
\Big| \\
G(q) \\
\Big| \\
G(q) \cdot \Big(G(f(\mathsf{loop}))^{-1} \cdot G(f(\mathsf{loop})) \Big) \\
\Big| \\
\Big(G(q) \cdot G(f(\mathsf{loop}))^{-1} \Big) \cdot G(f(\mathsf{loop})) \\
\Big| \ \text{via } \beta_f, \beta_G \\
\Big(G(q) \cdot G(f(\mathsf{loop}))^{-1} \Big) \cdot \mathsf{pair}^{=}(\mathsf{refl}, \mathsf{loop}) \\
\Big| \ \text{via } \mathcal{I}_2(\delta^{\star}(\mathsf{loop})) \\
\Big(G(f(\mathsf{loop}))^{-1} \cdot G(q) \Big) \cdot \mathsf{pair}^{=}(\mathsf{refl}, \mathsf{loop}) \\
\Big| \ \text{via } \gamma_G \\
\Big(G(f(\mathsf{loop}))^{-1} \cdot \mathsf{pair}^{=}(\mathsf{loop}, \mathsf{refl}) \Big) \cdot \mathsf{pair}^{=}(\mathsf{refl}, \mathsf{loop}) \\
\Big| \ \text{via } \beta_f, \beta_G \\
\Big(\mathsf{pair}^{=}(\mathsf{refl}, \mathsf{loop})^{-1} \cdot \mathsf{pair}^{=}(\mathsf{loop}, \mathsf{refl}) \Big) \cdot \mathsf{pair}^{=}(\mathsf{refl}, \mathsf{loop}) \\
\Big| \\
\mathsf{pair}^{=}(\mathsf{refl}, \mathsf{loop})^{-1} \cdot \Big(\mathsf{pair}^{=}(\mathsf{loop}, \mathsf{refl}) \cdot \mathsf{pair}^{=}(\mathsf{refl}, \mathsf{loop}) \Big) \\
\Big| \ \text{via } \Phi^{-1}_{\mathsf{loop},\mathsf{loop}} \\
\mathsf{pair}^{=}(\mathsf{refl}, \mathsf{loop})^{-1} \cdot \Big(\mathsf{pair}^{=}(\mathsf{refl}, \mathsf{loop}) \cdot \mathsf{pair}^{=}(\mathsf{loop}, \mathsf{refl}) \Big) \\
\Big| \\
\Big(\mathsf{pair}^{=}(\mathsf{refl}, \mathsf{loop})^{-1} \cdot \mathsf{pair}^{=}(\mathsf{refl}, \mathsf{loop}) \Big) \cdot \mathsf{pair}^{=}(\mathsf{loop}, \mathsf{refl}) \\
\Big| \\
\mathsf{pair}^{=}(\mathsf{loop}, \mathsf{refl}) \\
\Big| \\
\mathsf{refl} \cdot \mathsf{pair}^{=}(\mathsf{loop}, \mathsf{refl})
\end{array}
$$

We now observe the following:

- For any paths $\alpha_u : u_1 =_{a=_A b} u_2$, $\alpha_v : v_1 =_{b=_A d} v_2$, $\alpha_w : w_1 =_{a=_A c} w_2$, $\alpha_z : z_1 =_{c=_A d} z_2$ and $\phi : u_1 \cdot v_1 = w_1 \cdot z_1$, $\phi' : u_2 \cdot v_2 = w_2 \cdot z_2$, we have

$$\begin{tikzcd} u_1 \cdot v_1 \arrow[r, dash, "\text{via } \alpha_v"] \arrow[d, dash, "\phi"'] & u_1 \cdot v_2 \arrow[r, dash, "\text{via } \alpha_u"] \arrow[d, phantom, "="] & u_2 \cdot v_2 \arrow[d, dash, "\phi'"] \\ w_1 \cdot z_1 \arrow[r, dash, "\text{via } \alpha_z"'] & w_1 \cdot z_2 \arrow[r, dash, "\text{via } \alpha_w"'] & w_2 \cdot z_2 \end{tikzcd}$$

  if and only if

$$\begin{tikzcd} v_1 \cdot z_1^{-1} \arrow[r, dash, "\text{via } \alpha_z"] \arrow[d, dash, "\mathcal{I}_2(\phi)"'] & v_1 \cdot z_2^{-1} \arrow[r, dash, "\text{via } \alpha_v"] \arrow[d, phantom, "="] & v_2 \cdot z_2^{-1} \arrow[d, dash, "\mathcal{I}_2(\phi')"] \\ u_1^{-1} \cdot w_1 \arrow[r, dash, "\text{via } \alpha_w"'] & u_1^{-1} \cdot w_2 \arrow[r, dash, "\text{via } \alpha_u"'] & u_2^{-1} \cdot w_2 \end{tikzcd}$$

  This follows at once by path induction on $\alpha_u, \alpha_v, \alpha_w, \alpha_z$ and the fact that $\mathcal{I}_2$ is an equivalence.

Next we want to show that the following diagram commutes:

$$\begin{tikzcd} G(f(\mathsf{loop})) \cdot G(q) \arrow[r, dash, "\text{via } \beta_f, \beta_G, \gamma_G"] \arrow[d, dash, "\delta^\star(\mathsf{loop})"'] \arrow[dr, phantom, "(4)"] & \mathsf{pair}^=(\mathsf{refl}, \mathsf{loop}) \cdot \mathsf{pair}^=(\mathsf{loop}, \mathsf{refl}) \arrow[d, dash, "\Phi_{\mathsf{loop},\mathsf{loop}}"] \\ G(q) \cdot G(f(\mathsf{loop})) \arrow[r, dash, "\text{via } \beta_f, \beta_G, \gamma_G"'] & \mathsf{pair}^=(\mathsf{loop}, \mathsf{refl}) \cdot \mathsf{pair}^=(\mathsf{refl}, \mathsf{loop}) \end{tikzcd}$$

This is the same as saying that the outer rectangle in the diagram below commutes:

$$\begin{tikzcd} G(f(\mathsf{loop})) \cdot G(q) \arrow[r, dash, "\text{via } \beta_f"] \arrow[d, dash] \arrow[dr, phantom, "\mathrm{A}"] & G(p) \cdot G(q) \arrow[r, dash, "\text{via } \beta_G, \gamma_G"] \arrow[d, dash] & \mathsf{pair}^=(\mathsf{refl}, \mathsf{loop}) \cdot \mathsf{pair}^=(\mathsf{loop}, \mathsf{refl}) \arrow[ddd, dash, "\Phi_{\mathsf{loop},\mathsf{loop}}"] \\ G(f(\mathsf{loop}) \cdot q) \arrow[r, dash, "\text{via } \beta_f"'] \arrow[d, dash, "\mathsf{nat}_H(\mathsf{loop})"'] \arrow[dr, phantom, "\mathrm{B}"] & G(p \cdot q) \arrow[d, dash, "\text{via } t"] \arrow[r, phantom, "\mathrm{D}"] & {} \\ G(q \cdot f(\mathsf{loop})) \arrow[r, dash, "\text{via } \beta_f"'] \arrow[d, dash] \arrow[dr, phantom, "\mathrm{C}"] & G(q \cdot p) \arrow[d, dash] & {} \\ G(q) \cdot G(f(\mathsf{loop})) \arrow[r, dash, "\text{via } \beta_f"'] & G(q) \cdot G(p) \arrow[r, dash, "\text{via } \beta_G, \gamma_G"'] & \mathsf{pair}^=(\mathsf{loop}, \mathsf{refl}) \cdot \mathsf{pair}^=(\mathsf{refl}, \mathsf{loop}) \end{tikzcd}$$

But this is clear: A and C obviously commute, B is precisely the diagram (2), and D is the diagram (1).

Since (4) commutes, by our earlier observation the following diagram commutes:

$$\begin{tikzcd}
G(q) \cdot G(f(\mathsf{loop}))^{-1} \arrow[r, dash, "\text{via } \beta_f, \beta_G, \gamma_G"] \arrow[d, dash, "\mathcal{I}_2(\delta^\star(\mathsf{loop}))"'] & \mathsf{pair}^{=}(\mathsf{loop}, \mathsf{refl}) \cdot \mathsf{pair}^{=}(\mathsf{refl}, \mathsf{loop})^{-1} \arrow[d, dash, "\mathcal{I}_2(\Phi_{\mathsf{loop},\mathsf{loop}})"] \\
G(f(\mathsf{loop}))^{-1} \cdot G(q) \arrow[r, dash, "\text{via } \gamma_G, \beta_f, \beta_G"'] & \mathsf{pair}^{=}(\mathsf{refl}, \mathsf{loop})^{-1} \cdot \mathsf{pair}^{=}(\mathsf{loop}, \mathsf{refl})
\end{tikzcd}$$

Our path can now be equivalently stated as:

$$
\begin{array}{c}
G(q) \cdot \mathsf{refl} \\
\Big| \\
G(q) \\
\Big| \\
G(q) \cdot \Big(G(f(\mathsf{loop}))^{-1} \cdot G(f(\mathsf{loop})) \Big) \\
\Big| \\
\Big(G(q) \cdot G(f(\mathsf{loop}))^{-1} \Big) \cdot G(f(\mathsf{loop})) \\
\Big| \text{ via } \beta_f, \beta_G \\
\Big(G(q) \cdot G(f(\mathsf{loop}))^{-1} \Big) \cdot \mathsf{pair}^{=}(\mathsf{refl}, \mathsf{loop}) \\
\Big| \text{ via } \beta_f, \beta_G \\
\Big(G(q) \cdot \mathsf{pair}^{=}(\mathsf{refl}, \mathsf{loop}) \Big) \cdot \mathsf{pair}^{=}(\mathsf{refl}, \mathsf{loop}) \\
\Big| \text{ via } \gamma_G \\
\Big(\mathsf{pair}^{=}(\mathsf{loop}, \mathsf{refl}) \cdot \mathsf{pair}^{=}(\mathsf{refl}, \mathsf{loop}) \Big) \cdot \mathsf{pair}^{=}(\mathsf{refl}, \mathsf{loop}) \\
\Big| \text{ via } \mathcal{I}_2(\Phi_{\mathsf{loop},\mathsf{loop}}) \\
\Big(\mathsf{pair}^{=}(\mathsf{refl}, \mathsf{loop})^{-1} \cdot \mathsf{pair}^{=}(\mathsf{loop}, \mathsf{refl}) \Big) \cdot \mathsf{pair}^{=}(\mathsf{refl}, \mathsf{loop}) \\
\Big| \\
\mathsf{pair}^{=}(\mathsf{refl}, \mathsf{loop})^{-1} \cdot \Big(\mathsf{pair}^{=}(\mathsf{loop}, \mathsf{refl}) \cdot \mathsf{pair}^{=}(\mathsf{refl}, \mathsf{loop}) \Big) \\
\Big| \text{ via } \Phi^{-1}_{\mathsf{loop},\mathsf{loop}} \\
\mathsf{pair}^{=}(\mathsf{refl}, \mathsf{loop})^{-1} \cdot \Big(\mathsf{pair}^{=}(\mathsf{refl}, \mathsf{loop}) \cdot \mathsf{pair}^{=}(\mathsf{loop}, \mathsf{refl}) \Big) \\
\Big| \\
\Big(\mathsf{pair}^{=}(\mathsf{refl}, \mathsf{loop})^{-1} \cdot \mathsf{pair}^{=}(\mathsf{refl}, \mathsf{loop}) \Big) \cdot \mathsf{pair}^{=}(\mathsf{loop}, \mathsf{refl}) \\
\Big| \\
\mathsf{pair}^{=}(\mathsf{loop}, \mathsf{refl}) \\
\Big| \\
\mathsf{refl} \cdot \mathsf{pair}^{=}(\mathsf{loop}, \mathsf{refl})
\end{array}
$$

which is equal to:

$$
\begin{array}{c}
G(q) \cdot \mathsf{refl} \\
\Big| \\
G(q) \\
\Big| \ \text{via } \gamma_G \\
\mathsf{pair}^{=}(\mathsf{loop}, \mathsf{refl}) \\
\Big| \\
\mathsf{pair}^{=}(\mathsf{loop}, \mathsf{refl}) \cdot \left(\mathsf{pair}^{=}(\mathsf{refl}, \mathsf{loop})^{-1} \cdot \mathsf{pair}^{=}(\mathsf{refl}, \mathsf{loop})\right) \\
\Big| \\
\left(\mathsf{pair}^{=}(\mathsf{loop}, \mathsf{refl}) \cdot \mathsf{pair}^{=}(\mathsf{refl}, \mathsf{loop})^{-1}\right) \cdot \mathsf{pair}^{=}(\mathsf{refl}, \mathsf{loop}) \\
\Big| \ \text{via } \mathcal{I}_2(\Phi_{\mathsf{loop},\mathsf{loop}}) \\
\left(\mathsf{pair}^{=}(\mathsf{refl}, \mathsf{loop})^{-1} \cdot \mathsf{pair}^{=}(\mathsf{loop}, \mathsf{refl})\right) \cdot \mathsf{pair}^{=}(\mathsf{refl}, \mathsf{loop}) \\
\Big| \\
\mathsf{pair}^{=}(\mathsf{refl}, \mathsf{loop})^{-1} \cdot \left(\mathsf{pair}^{=}(\mathsf{loop}, \mathsf{refl}) \cdot \mathsf{pair}^{=}(\mathsf{refl}, \mathsf{loop})\right) \\
\Big| \ \text{via } \Phi^{-1}_{\mathsf{loop},\mathsf{loop}} \\
\mathsf{pair}^{=}(\mathsf{refl}, \mathsf{loop})^{-1} \cdot \left(\mathsf{pair}^{=}(\mathsf{refl}, \mathsf{loop}) \cdot \mathsf{pair}^{=}(\mathsf{loop}, \mathsf{refl})\right) \\
\Big| \\
\left(\mathsf{pair}^{=}(\mathsf{refl}, \mathsf{loop})^{-1} \cdot \mathsf{pair}^{=}(\mathsf{refl}, \mathsf{loop})\right) \cdot \mathsf{pair}^{=}(\mathsf{loop}, \mathsf{refl}) \\
\Big| \\
\mathsf{pair}^{=}(\mathsf{loop}, \mathsf{refl}) \\
\Big| \\
\mathsf{refl} \cdot \mathsf{pair}^{=}(\mathsf{loop}, \mathsf{refl})
\end{array}
$$

We now make the following observation:

- For any paths $u : a =_A b$, $v : b =_A d$, $w : a =_A c$, $z : c =_A d$ and $\phi : u \cdot v = w \cdot z$, the path

$$
\begin{array}{c}
v \\
\Big| \\
v \cdot (z^{-1} \cdot z) \\
\Big| \\
(v \cdot z^{-1}) \cdot z \\
\Big| \text{ via } \mathcal{I}_2(\phi) \\
(u^{-1} \cdot w) \cdot z \\
\Big| \\
u^{-1} \cdot (w \cdot z) \\
\Big| \ \phi^{-1} \\
u^{-1} \cdot (u \cdot v) \\
\Big| \\
(u^{-1} \cdot u) \cdot v \\
\Big| \\
v
\end{array}
$$

is equal to the identity path at v. This follows by path induction on u and z.

Using the above observation, we can express our path simply as

$$
\begin{array}{c}
G(q) \cdot \mathsf{refl} \\
\Big| \\
G(q) \\
\Big| \text{ via } \gamma_G \\
\mathsf{pair}^{=}(\mathsf{loop}, \mathsf{refl}) \\
\Big| \\
\mathsf{refl} \cdot \mathsf{pair}^{=}(\mathsf{loop}, \mathsf{refl})
\end{array}
$$

which is precisely η.

Right-to-left We need to show that for any $x : T^2$ we have $F^{\times}(G(t)) = t$. We use torus induction, with $b' :\equiv \mathsf{refl}_b$. We let p' be the path

$$\begin{tikzcd}
\mathsf{transport}^{z \mapsto F^{\times}(G(z))=z}(p, \mathsf{refl}) \arrow[d, dash, "\mathcal{T}_3(p, \mathsf{refl})"] \\
F^{\times}(G(p))^{-1} \cdot \mathsf{refl} \cdot p \arrow[d, dash, "\zeta_p"] \\
\mathsf{refl}
\end{tikzcd}$$

where for any $\alpha : x =_{T^2} y$ and $u : F^{\times}(G(x)) = x$, the path

$$\mathcal{T}_3(\alpha, u) : \mathsf{transport}^{z \mapsto F^{\times}(G(z))=z}(\alpha, u) = F^{\times}(G(\alpha))^{-1} \cdot u \cdot \alpha$$

is defined by path induction on α and ζ_p is the path

$$\begin{tikzcd}
F^{\times}(G(p))^{-1} \cdot \mathsf{refl} \cdot p \arrow[d, dash] \\
F^{\times}(G(p))^{-1} \cdot p \arrow[d, dash, "\text{via } \beta_G"] \\
F^{\times}(\mathsf{pair}^{=}(\mathsf{refl}, \mathsf{loop}))^{-1} \cdot p \arrow[d, dash, "\text{via } \mu(\mathsf{loop})"] \\
f(\mathsf{loop})^{-1} \cdot p \arrow[d, dash, "\text{via } \beta_f"] \\
p^{-1} \cdot p \arrow[d, dash] \\
\mathsf{refl}
\end{tikzcd}$$

Similarly, let q' be the path

$$\begin{tikzcd}
\mathsf{transport}^{z \mapsto F^{\times}(G(z))=z}(q, \mathsf{refl}) \arrow[d, dash, "\mathcal{T}_3(q, \mathsf{refl})"] \\
F^{\times}(G(q))^{-1} \cdot \mathsf{refl} \cdot q \arrow[d, dash, "\zeta_q"] \\
\mathsf{refl}
\end{tikzcd}$$

where ζ_q is the path

$$
\begin{tikzcd}
F^\times(G(q))^{-1} \cdot \mathsf{refl} \cdot q \arrow[d, dash] \\
F^\times(G(q))^{-1} \cdot q \arrow[d, dash, "\text{via } \gamma_G"] \\
F^\times(\mathsf{pair}^=(\mathsf{loop}, \mathsf{refl}))^{-1} \cdot q \arrow[d, dash, "\text{via } \nu(\mathsf{loop})"] \\
\mathsf{happly}_{F^\to(\mathsf{loop})}(\mathsf{base})^{-1} \cdot q \arrow[d, dash, "\text{via } \beta_{F^\to}"] \\
\mathsf{happly}_{\mathsf{funext}(H)}(\mathsf{base})^{-1} \cdot q \arrow[d, dash, "\text{via } \mathsf{happly\text{-}funext\text{-}inv}(H)"] \\
q^{-1} \cdot q \arrow[d, dash] \\
\mathsf{refl}
\end{tikzcd}
$$

All that remains now is to show that the following diagram commutes:

$$
\begin{tikzcd}
\mathsf{transport}^{z \mapsto F^\times(G(z))=z}(p \cdot q, \mathsf{refl}) \arrow[rr, dash, "\text{via } t"] \arrow[d, dash] & & \mathsf{transport}^{z \mapsto F^\times(G(z))=z}(q \cdot p, \mathsf{refl}) \arrow[d, dash] \\
\mathsf{transport}^{z \mapsto F^\times(G(z))=z}(q, \mathsf{transport}^{z \mapsto F^\times(G(z))=z}(p, \mathsf{refl})) \arrow[d, dash, "\text{via } p'"] & & \mathsf{transport}^{z \mapsto F^\times(G(z))=z}(p, \mathsf{transport}^{z \mapsto F^\times(G(z))=z}(q, \mathsf{refl})) \arrow[d, dash, "\text{via } q'"] \\
\mathsf{transport}^{z \mapsto F^\times(G(z))=z}(q, \mathsf{refl}) \arrow[dr, dash, "q'"'] & & \mathsf{transport}^{z \mapsto F^\times(G(z))=z}(p, \mathsf{refl}) \arrow[dl, dash, "p'"] \\
& \mathsf{refl} &
\end{tikzcd}
$$

We make the following observation:

- For any $\alpha : x =_{T^2} y$, $\alpha' : y =_{T^2} z$, $u_x : F^\times(G(x)) = x$, $u_y : F^\times(G(y)) = y$, $u_z : F^\times(G(z)) = z$, and $\eta_y : F^\times(G(\alpha))^{-1} \cdot u_x \cdot \alpha = u_y$, $\eta_z : F^\times(G(\alpha'))^{-1} \cdot u_y \cdot \alpha' = u_z$, the path

$$
\begin{array}{c}
\mathsf{transport}^{z\mapsto F^\times(G(z))=z}(\alpha \cdot \alpha', u_x) \\
\Big| \\
\mathsf{transport}^{z\mapsto F^\times(G(z))=z}(\alpha', \mathsf{transport}^{z\mapsto F^\times(G(z))=z}(\alpha, u_x)) \\
\Big| \text{ via } \mathcal{T}_3(\alpha, u_x) \\
\mathsf{transport}^{z\mapsto F^\times(G(z))=z}(\alpha', F^\times(G(\alpha))^{-1} \cdot u_x \cdot \alpha) \\
\Big| \text{ via } \eta_y \\
\mathsf{transport}^{z\mapsto F^\times(G(z))=z}(\alpha', u_y) \\
\Big| \ \mathcal{T}_3(\alpha', u_y) \\
F^\times(G(\alpha'))^{-1} \cdot u_y \cdot \alpha' \\
\Big| \ \eta_z \\
u_z
\end{array}
$$

can be equivalently expressed as the path

$$
\begin{array}{c}
\mathsf{transport}^{z\mapsto F^\times(G(z))=z}(\alpha \cdot \alpha', u_x) \\
\Big| \ \mathcal{T}_3(\alpha \cdot \alpha', u_x) \\
F^\times(G(\alpha \cdot \alpha'))^{-1} \cdot u_x \cdot (\alpha \cdot \alpha') \\
\Big| \\
F^\times(G(\alpha'))^{-1} \cdot \left(F^\times(G(\alpha))^{-1} \cdot u_x \cdot \alpha\right) \cdot \alpha' \\
\Big| \text{ via } \eta_y \\
F^\times(G(\alpha'))^{-1} \cdot u_y \cdot \alpha' \\
\Big| \ \eta_z \\
u_z
\end{array}
$$

To prove this, it suffices to show that the outer rectangle in the diagram below commutes:

$$\begin{tikzcd}
\mathsf{transport}^{z\mapsto F^\times(G(z))=z}(\alpha\cdot\alpha', u_x) \arrow[rr, dash, "\mathcal{T}_3(\alpha\cdot\alpha', u_x)"] \arrow[d, dash] & & F^\times(G(\alpha\cdot\alpha'))^{-1}\cdot u_x\cdot(\alpha\cdot\alpha') \arrow[dd, dash] \\
\mathsf{transport}^{z\mapsto F^\times(G(z))=z}(\alpha', \mathsf{transport}^{z\mapsto F^\times(G(z))=z}(\alpha, u_x)) \arrow[d, dash, "\text{via } \mathcal{T}_3(\alpha, u_x)"] & \mathrm{A} & \\
\mathsf{transport}^{z\mapsto F^\times(G(z))=z}(\alpha', F^\times(G(\alpha))^{-1}\cdot u_x\cdot\alpha) \arrow[rr, dash, "\mathcal{T}_3(\alpha', F^\times(G(\alpha))^{-1}\cdot u_x\cdot\alpha)"'] \arrow[d, dash, "\text{via } \eta_y"] & & F^\times(G(\alpha'))^{-1}\cdot\left(F^\times(G(\alpha))^{-1}\cdot u_x\cdot\alpha\right)\cdot\alpha' \arrow[d, dash, "\text{via } \eta_y"] \\
\mathsf{transport}^{z\mapsto F^\times(G(z))=z}(\alpha', u_y) \arrow[rr, dash, "\mathcal{T}_3(\alpha', u_y)"'] & \mathrm{B} & F^\times(G(\alpha'))^{-1}\cdot u_y\cdot\alpha'
\end{tikzcd}$$

Both of the rectangles A and B are easily shown to commute by a suitable path induction.

Using the above observation, it suffices to show that the outer part of the following diagram commutes:

$$\begin{tikzcd}
\mathsf{transport}^{z\mapsto F^\times(G(z))=z}(p\cdot q, \mathsf{refl}) \arrow[rr, dash, "\text{via } t"] \arrow[d, dash, "\mathcal{T}_3(p\cdot q, \mathsf{refl})"] & \mathrm{A} & \mathsf{transport}^{z\mapsto F^\times(G(z))=z}(q\cdot p, \mathsf{refl}) \arrow[d, dash, "\mathcal{T}_3(q\cdot p, \mathsf{refl})"] \\
F^\times(G(p\cdot q))^{-1}\cdot\mathsf{refl}\cdot(p\cdot q) \arrow[r, dash, "\text{via } t"'] \arrow[d, dash] & F^\times(G(q\cdot p))^{-1}\cdot\mathsf{refl}\cdot(p\cdot q) \arrow[r, dash, "\text{via } t"'] & F^\times(G(q\cdot p))^{-1}\cdot\mathsf{refl}\cdot(q\cdot p) \arrow[d, dash] \\
F^\times(G(q))^{-1}\cdot\left(F^\times(G(p))^{-1}\cdot\mathsf{refl}\cdot p\right)\cdot q \arrow[d, dash, "\text{via } \zeta_p"] & \mathrm{B} & F^\times(G(p))^{-1}\cdot\left(F^\times(G(q))^{-1}\cdot\mathsf{refl}\cdot q\right)\cdot p \arrow[d, dash, "\text{via } \zeta_q"] \\
F^\times(G(q))^{-1}\cdot\mathsf{refl}\cdot q \arrow[dr, dash, "\zeta_q"'] & & F^\times(G(p))^{-1}\cdot\mathsf{refl}\cdot q \arrow[dl, dash, "\zeta_p"] \\
& \mathsf{refl} &
\end{tikzcd}$$

Since rectangle A clearly commutes, it suffices to prove that part B commutes.

We can equivalently express diagram B as

$$
\begin{tikzcd}
F^{\times}(G(p \cdot q))^{-1} \cdot \mathsf{refl} \cdot (p \cdot q) \arrow[r, dash, "\text{via } t"'] \arrow[d, dash] & F^{\times}(G(q \cdot p))^{-1} \cdot \mathsf{refl} \cdot (p \cdot q) \arrow[r, dash, "\text{via } t"'] & F^{\times}(G(q \cdot p))^{-1} \cdot \mathsf{refl} \cdot (q \cdot p) \arrow[d, dash] \\
\left(F^{\times}(G(p)) \cdot F^{\times}(G(q))\right)^{-1} \cdot \mathsf{refl} \cdot (p \cdot q) \arrow[d, dash] & & \left(F^{\times}(G(q)) \cdot F^{\times}(G(p))\right)^{-1} \cdot \mathsf{refl} \cdot (q \cdot p) \arrow[d, dash] \\
F^{\times}(G(q))^{-1} \cdot \left(F^{\times}(G(p))^{-1} \cdot \mathsf{refl} \cdot p\right) \cdot q \arrow[d, dash, "\text{via } \zeta_p"] & & F^{\times}(G(p))^{-1} \cdot \left(F^{\times}(G(q))^{-1} \cdot \mathsf{refl} \cdot q\right) \cdot p \arrow[d, dash, "\text{via } \zeta_q"] \\
F^{\times}(G(q))^{-1} \cdot \mathsf{refl} \cdot q \arrow[dr, dash, "\zeta_q"'] & & F^{\times}(G(p))^{-1} \cdot \mathsf{refl} \cdot q \arrow[dl, dash, "\zeta_p"] \\
& \mathsf{refl} &
\end{tikzcd}
$$

We make the following observation:

- For any $\alpha_u : u_1 =_{a =_A b} u_2$, $\alpha_v : v_1 =_{b =_A c} v_2$, the path

$$
\begin{tikzcd}
(u_1 \cdot v_1)^{-1} \cdot \mathsf{refl} \cdot (u_2 \cdot v_2) \arrow[d, dash] \\
v_1^{-1} \cdot (u_1^{-1} \cdot \mathsf{refl} \cdot u_2) \cdot v_2 \arrow[d, dash] \\
v_1^{-1} \cdot (u_1^{-1} \cdot u_2) \cdot v_2 \arrow[d, dash, "\text{via } \alpha_u"] \\
v_1^{-1} \cdot (u_2^{-1} \cdot u_2) \cdot v_2 \arrow[d, dash] \\
v_1^{-1} \cdot \mathsf{refl} \cdot v_2 \arrow[d, dash] \\
v_1^{-1} \cdot v_2 \arrow[d, dash, "\text{via } \alpha_v"] \\
v_2^{-1} \cdot v_2 \arrow[d, dash] \\
\mathsf{refl}
\end{tikzcd}
$$

  is equivalent to the path

$$
\begin{tikzcd}
(u_1 \cdot v_1)^{-1} \cdot \mathsf{refl} \cdot (u_2 \cdot v_2) \arrow[d, dash, "\text{via } \alpha_u"] \\
(u_2 \cdot v_1)^{-1} \cdot \mathsf{refl} \cdot (u_2 \cdot v_2) \arrow[d, dash, "\text{via } \alpha_v"] \\
(u_2 \cdot v_2)^{-1} \cdot \mathsf{refl} \cdot (u_2 \cdot v_2) \arrow[d, dash] \\
(u_2 \cdot v_2)^{-1} \cdot (u_2 \cdot v_2) \arrow[d, dash] \\
\mathsf{refl}
\end{tikzcd}
$$

This is clear by path induction.

Using this observation, it suffices to show that the outer part of the following diagram commutes:

$$
\begin{tikzcd}
F^{\times}(G(p \cdot q))^{-1} \cdot \mathsf{refl} \cdot (p \cdot q) \arrow[r, dash, "\text{via } t"'] \arrow[d, dash] & F^{\times}(G(q \cdot p))^{-1} \cdot \mathsf{refl} \cdot (p \cdot q) \arrow[r, dash, "\text{via } t"'] & F^{\times}(G(q \cdot p))^{-1} \cdot \mathsf{refl} \cdot (q \cdot p) \arrow[d, dash] \\
\left(F^{\times}(G(p)) \cdot F^{\times}(G(q))\right)^{-1} \cdot \mathsf{refl} \cdot (p \cdot q) \arrow[d, dash, "\text{via } \beta_G, \mu(\mathsf{loop}), \beta_f"] & \mathrm{A} & \left(F^{\times}(G(q)) \cdot F^{\times}(G(p))\right)^{-1} \cdot \mathsf{refl} \cdot (q \cdot p) \arrow[d, dash, "\text{via } \gamma_G, \nu(\mathsf{loop}), \beta_{F\to}, \mathsf{happly\text{-}funext\text{-}inv}(H)"'] \\
\left(p \cdot F^{\times}(G(q))\right)^{-1} \cdot \mathsf{refl} \cdot (p \cdot q) \arrow[d, dash, "\text{via } \gamma_G, \nu(\mathsf{loop}), \beta_{F\to}, \mathsf{happly\text{-}funext\text{-}inv}(H)"] & & \left(q \cdot F^{\times}(G(p))\right)^{-1} \cdot \mathsf{refl} \cdot (q \cdot p) \arrow[d, dash, "\text{via } \beta_G, \mu(\mathsf{loop}), \beta_f"'] \\
(p \cdot q)^{-1} \cdot \mathsf{refl} \cdot (p \cdot q) \arrow[r, dash, "\text{via } t"'] \arrow[d, dash] & (q \cdot p)^{-1} \cdot \mathsf{refl} \cdot (p \cdot q) \arrow[r, dash, "\text{via } t"'] & (q \cdot p)^{-1} \cdot \mathsf{refl} \cdot (q \cdot p) \arrow[d, dash] \\
(p \cdot q)^{-1} \cdot (p \cdot q) \arrow[dr, dash] & \mathrm{B} & (q \cdot p)^{-1} \cdot (q \cdot p) \arrow[dl, dash] \\
& \mathsf{refl} &
\end{tikzcd}
$$

Part B clearly commutes. This leaves us to show that part A commutes. We now make the following observation:

- For any 2-paths $\alpha_u^1 : u_1 =_{a=_A b} u_2$, $\alpha_u^2 : u_1 =_{a=_A b} u_3$, $\alpha_u^3 : u_3 =_{a=_A b} u_4$, $\alpha_u^4 : u_2 =_{a=_A b} u_4$, $\alpha_v : v_1 =_{c=_A d} v_2$ and path $w : b =_A c$ such that

$$\begin{tikzcd} u_1 \arrow[r, dash, "\alpha_u^2"] \arrow[d, dash, "\alpha_u^1"'] \arrow[dr, phantom, "="] & u_3 \arrow[d, dash, "\alpha_u^3"] \\ u_2 \arrow[r, dash, "\alpha_u^4"'] & u_4 \end{tikzcd}$$

the following diagram commutes:

$$\begin{tikzcd} u_1^{-1} \cdot w \cdot v_1 \arrow[r, dash, "\text{via } \alpha_u^2"] \arrow[d, dash, "\text{via } \alpha_u^1"'] & u_3^{-1} \cdot w \cdot v_1 \arrow[r, dash, "\text{via } \alpha_v"] & u_3^{-1} \cdot w \cdot v_2 \arrow[d, dash, "\text{via } \alpha_u^3"] \\ u_2^{-1} \cdot w \cdot v_1 \arrow[r, dash, "\text{via } \alpha_u^4"'] & u_4^{-1} \cdot w \cdot v_1 \arrow[r, dash, "\text{via } \alpha_v"'] & u_4^{-1} \cdot w \cdot v_2 \end{tikzcd}$$

This is clear by path induction on α_v, α_u^2, α_u^4.

Using this observation, it suffices to show that the following diagram commutes:

$$\begin{tikzcd} F^\times(G(p \cdot q)) \arrow[r, dash, "\text{via } t"] \arrow[d, dash] & F^\times(G(q \cdot p)) \arrow[d, dash] \\ F^\times(G(p)) \cdot F^\times(G(q)) \arrow[d, dash, "\text{via } \beta_G,\, \mu(\mathsf{loop}),\, \beta_f"] & F^\times(G(q)) \cdot F^\times(G(p)) \arrow[d, dash, "\text{via } \gamma_G,\, \nu(\mathsf{loop}),\, \beta_{F^\to},\, \mathsf{happly\text{-}funext\text{-}inv}(H)"'] \\ p \cdot F^\times(G(q)) \arrow[d, dash, "\text{via } \gamma_G,\, \nu(\mathsf{loop}),\, \beta_{F^\to},\, \mathsf{happly\text{-}funext\text{-}inv}(H)"] & q \cdot F^\times(G(p)) \arrow[d, dash, "\text{via } \beta_G,\, \mu(\mathsf{loop}),\, \beta_f"'] \\ p \cdot q \arrow[r, dash, "\text{via } t"'] & q \cdot p \end{tikzcd}$$

After some rearranging we get:

$$
\begin{tikzcd}
F^\times(G(p \cdot q)) \arrow[r, dash, "\text{via } t"] \arrow[d, dash] & F^\times(G(q \cdot p)) \arrow[d, dash] \\
F^\times(G(p) \cdot G(q)) \arrow[d, dash] & F^\times(G(q) \cdot G(p)) \arrow[d, dash] \\
F^\times(G(p)) \cdot F^\times(G(q)) \arrow[d, dash, "\text{via } \beta_G, \gamma_G"] & F^\times(G(q)) \cdot F^\times(G(p)) \arrow[d, dash, "\text{via } \beta_G, \gamma_G"] \\
F^\times(\mathsf{pair}^=(\mathsf{refl}, \mathsf{loop})) \cdot F^\times(\mathsf{pair}^=(\mathsf{loop}, \mathsf{refl})) \arrow[d, dash, "\text{via } \mu(\mathsf{loop}), \nu(\mathsf{loop})"] & F^\times(\mathsf{pair}^=(\mathsf{loop}, \mathsf{refl})) \cdot F^\times(\mathsf{pair}^=(\mathsf{refl}, \mathsf{loop})) \arrow[d, dash, "\text{via } \mu(\mathsf{loop}), \nu(\mathsf{loop})"] \\
f(\mathsf{loop}) \cdot \mathsf{happly}_{F^\to(\mathsf{loop})}(\mathsf{base}) \arrow[d, dash, "\text{via } \beta_{F^\to}, \mathsf{happly\text{-}funext\text{-}inv}(H)"] & \mathsf{happly}_{F^\to(\mathsf{loop})}(\mathsf{base}) \cdot f(\mathsf{loop}) \arrow[d, dash, "\text{via } \beta_{F^\to}, \mathsf{happly\text{-}funext\text{-}inv}(H)"] \\
f(\mathsf{loop}) \cdot q \arrow[d, dash, "\text{via } \beta_f"] & q \cdot f(\mathsf{loop}) \arrow[d, dash, "\text{via } \beta_f"] \\
p \cdot q \arrow[r, dash, "\text{via } t"'] & q \cdot p
\end{tikzcd}
$$

or equivalently:

$$
\begin{tikzcd}
F^{\times}(G(p \cdot q)) \arrow[rr, dash, "\text{via } t"] \arrow[d, dash] & & F^{\times}(G(q \cdot p)) \arrow[d, dash] \\
F^{\times}(G(p) \cdot G(q)) \arrow[d, dash, "\text{via } \beta_G,\, \gamma_G"] & & F^{\times}(G(q) \cdot G(p)) \arrow[d, dash, "\text{via } \beta_G,\, \gamma_G"] \\
F^{\times}(\mathsf{pair}^{=}(\mathsf{refl}, \mathsf{loop}) \cdot \mathsf{pair}^{=}(\mathsf{loop}, \mathsf{refl})) \arrow[d, dash] & & F^{\times}(\mathsf{pair}^{=}(\mathsf{loop}, \mathsf{refl}) \cdot \mathsf{pair}^{=}(\mathsf{refl}, \mathsf{loop})) \arrow[d, dash] \\
F^{\times}(\mathsf{pair}^{=}(\mathsf{refl}, \mathsf{loop})) \cdot F^{\times}(\mathsf{pair}^{=}(\mathsf{loop}, \mathsf{refl})) \arrow[d, dash, "\text{via } \mu(\mathsf{loop}),\, \nu(\mathsf{loop})"] & & F^{\times}(\mathsf{pair}^{=}(\mathsf{loop}, \mathsf{refl})) \cdot F^{\times}(\mathsf{pair}^{=}(\mathsf{refl}, \mathsf{loop})) \arrow[d, dash, "\text{via } \mu(\mathsf{loop}),\, \nu(\mathsf{loop})"] \\
f(\mathsf{loop}) \cdot \mathsf{happly}_{F^{\to}(\mathsf{loop})}(\mathsf{base}) \arrow[d, dash, "\text{via } \beta_{F^{\to}},\, \mathsf{happly\text{-}funext\text{-}inv}(H)"] & & \mathsf{happly}_{F^{\to}(\mathsf{loop})}(\mathsf{base}) \cdot f(\mathsf{loop}) \arrow[d, dash, "\text{via } \beta_{F^{\to}},\, \mathsf{happly\text{-}funext\text{-}inv}(H)"] \\
f(\mathsf{loop}) \cdot q \arrow[d, dash, "\text{via } \beta_f"] & & q \cdot f(\mathsf{loop}) \arrow[d, dash, "\text{via } \beta_f"] \\
p \cdot q \arrow[rr, dash, "\text{via } t"'] & & q \cdot p
\end{tikzcd}
$$

Finally, this diagram can be viewed as follows:

$$
\begin{tikzcd}
F^{\times}(G(p \cdot q)) \arrow[rr, dash, "\text{via } t"] \arrow[d, dash] & & F^{\times}(G(q \cdot p)) \arrow[d, dash] \\
F^{\times}(G(p) \cdot G(q)) \arrow[d, dash, "\text{via } \beta_G, \gamma_G"] & \mathrm{A} & F^{\times}(G(q) \cdot G(p)) \arrow[d, dash, "\text{via } \beta_G, \gamma_G"] \\
F^{\times}(\mathsf{pair}^{=}(\mathsf{refl}, \mathsf{loop}) \cdot \mathsf{pair}^{=}(\mathsf{loop}, \mathsf{refl})) \arrow[rr, dash, "\text{via } \Phi_{\mathsf{loop},\mathsf{loop}}"] \arrow[d, dash] & & F^{\times}(\mathsf{pair}^{=}(\mathsf{loop}, \mathsf{refl}) \cdot \mathsf{pair}^{=}(\mathsf{refl}, \mathsf{loop})) \arrow[d, dash] \\
F^{\times}(\mathsf{pair}^{=}(\mathsf{refl}, \mathsf{loop})) \cdot F^{\times}(\mathsf{pair}^{=}(\mathsf{loop}, \mathsf{refl})) \arrow[d, dash, "\text{via } \mu(\mathsf{loop}), \nu(\mathsf{loop})"] & \mathrm{B} & F^{\times}(\mathsf{pair}^{=}(\mathsf{loop}, \mathsf{refl})) \cdot F^{\times}(\mathsf{pair}^{=}(\mathsf{refl}, \mathsf{loop})) \arrow[d, dash, "\text{via } \mu(\mathsf{loop}), \nu(\mathsf{loop})"] \\
f(\mathsf{loop}) \cdot \mathsf{happly}_{F^{\to}(\mathsf{loop})}(\mathsf{base}) \arrow[rr, dash, "\mathsf{nat}_{\mathsf{happly}_{F^{\to}(\mathsf{loop})}}(\mathsf{loop})"] \arrow[d, dash, "\text{via } \beta_{F^{\to}}, \mathsf{happly\text{-}funext\text{-}inv}(H)"] & & \mathsf{happly}_{F^{\to}(\mathsf{loop})}(\mathsf{base}) \cdot f(\mathsf{loop}) \arrow[d, dash, "\text{via } \beta_{F^{\to}}, \mathsf{happly\text{-}funext\text{-}inv}(H)"] \\
f(\mathsf{loop}) \cdot q \arrow[rr, dash, "\mathsf{nat}_H(\mathsf{loop})"'] \arrow[d, dash, "\text{via } \beta_f"] & \mathrm{C} & q \cdot f(\mathsf{loop}) \arrow[d, dash, "\text{via } \beta_f"] \\
p \cdot q \arrow[rr, dash, "\text{via } t"'] & \mathrm{D} & q \cdot p
\end{tikzcd}
$$

where for any $\alpha : x =_{\mathbb{S}^1} y$, the 2-path

$$\mathsf{nat}_{\mathsf{happly}_{F^{\to}(\mathsf{loop})}}(\alpha) : f(\alpha) \cdot \mathsf{happly}_{F^{\to}(\mathsf{loop})}(y) = \mathsf{happly}_{F^{\to}(\mathsf{loop})}(x) \cdot f(\alpha)$$

is defined by induction on α.

Now rectangles A and D commute by diagrams (1) and (2) respectively. Rectangles B and C commute by a suitable path induction. □

Solution to Exercise 6.13

Let's call the given homotopy $p : \prod_{(x:A)}(f(x) = g(x))$. Since B may not be a mere proposition, we consider the contractible type $\sum_{(y:B)}(f(x) = y)$. For all $x : A$ we define a function $q_x : \mathbf{2} \to \sum_{(y:B)}(f(x) = y)$ by recursion on $\mathbf{2}$, with:

$$
\begin{aligned}
q_x(1_{\mathbf{2}}) &:\equiv (f(x), \mathsf{refl}_{f(x)}), \\
q_x(0_{\mathbf{2}}) &:\equiv (g(x), p(x)).
\end{aligned}
$$

By induction on truncation, since $\sum_{(y:B)}(f(x) = y)$ is a mere proposition, we get $\widetilde{p}_x : \|\mathbf{2}\| \to \sum_{(y:B)}(f(x) = y)$ satisfying $\widetilde{p}_x(|b|) \equiv q_x(b)$ for all $b : B$. Hence we can define $q : \|\mathbf{2}\| \to (A \to B)$ by

$$q(i) :\equiv \lambda x.\, \mathsf{pr}_1(\widetilde{p}_x(i))$$

Then we have $q(|1_{\mathbf{2}}|)(x) \equiv f(x)$ for all $x : A$, and hence $q(|1_{\mathbf{2}}|) \equiv f$. Similarly, $q(|0_{\mathbf{2}}|) \equiv g$, and hence $q(r) : f =_{(A \to B)} g$, where $r : |1_{\mathbf{2}}| =_{\|\mathbf{2}\|} |0_{\mathbf{2}}|$ exists since $\|\mathbf{2}\|$ is a mere proposition.

Exercises from Chapter 7

Solution to Exercise 7.5

Let $\bigcirc$ be a modality; it remains to show that if all maps into B out of a $\bigcirc$-connected type are constant, then B is $\bigcirc$-modal. We show under that hypothesis that $\eta : B \to \bigcirc B$ is an equivalence, by showing that its fibers are contractible.

Since η is always $\bigcirc$-connected, its fibers are $\bigcirc$-connected, and thus the inclusion of $\mathsf{fib}_\eta(z)$ into B is constant. Thus, there is an $a : B$ such that for any $b : B$ and $p : \eta b = z$ we have $q(b,p) : a = b$. Then $\mathsf{ap}_\eta(q(b,p)) \cdot p : \eta a = z$ for any $(b,p) : \mathsf{fib}_\eta(z)$, so we have a map $\mathsf{fib}_\eta(z) \to (\eta a = z)$. But $\bigcirc B$ is $\bigcirc$-modal, hence so is $\eta a = z$. Thus, by Corollary 7.5.9, this map is constant; hence we have $r : \eta a = z$ such that $\mathsf{ap}_\eta(q(b,p)) \cdot p = r$ for all (b,p). But using the characterization of paths in fibers, this means exactly that $(b,p) = (a,r)$ for all (b,p); hence $\mathsf{fib}_\eta(z)$ is contractible with center (a,r).

Solution to Exercise 7.6

If A is n-connected, then since $\|A\|_{-1} = \|\|A\|_n\|_{-1}$, also A is (-1)-connected. And since $\|a =_A b\|_{n-1} = (|a|_n =_{\|A\|_n} |b|_n)$ by Theorem 7.3.12 and the path spaces of a contractible type are contractible, each $a =_A b$ is $(n-1)$-connected.

Conversely, suppose A is (-1)-connected and all its path spaces are $(n-1)$-connected. Firstly, we claim $\|A\|_n$ is a mere proposition, i.e. that for all $x, y : \|A\|_n$, the type $x = y$ is contractible. Since contractibility of $x = y$ is a mere proposition, it suffices to assume that x and y are of the form $|a|_n$ and $|b|_n$ respectively. But $|a|_n =_{\|A\|_n} |b|_n$ is contractible by Theorem 7.3.12 and the assumption. Thus, $\|A\|_n$ is a mere proposition. Since it is also (-1)-connected by assumption, it is therefore contractible, so A is n-connected.

Solution to Exercise 7.7

Evidently LEM is the same as $\mathsf{LEM}_{-1,-1}$, so for the first part it suffices to assume LEM and prove $\mathsf{LEM}_{n,-1}$ and $\mathsf{LEM}_{-1,m}$ for all n, m. For the latter, note that if A is a mere proposition, then by Exercise 3.6 so is $A + \neg A$, and thus $\|A + \neg A\|_m = A + \neg A$ for any $m \geq -1$. For the former, note that assuming LEM, by Exercise 3.14 we have $\|B\|_{-1} = \neg\neg B$ for any B, while $\neg\neg(A + \neg A)$ is always true for any type A (Exercise 1.13).

For the second part, it suffices to derive a contradiction from $\mathsf{LEM}_{0,0}$; but the proof of Corollary 3.2.7 already uses an A that is a set (namely $\mathbf{2}$).

Solution to Exercise 7.10

Suppose the (-1)-connected AC, and by induction suppose also the n-connected AC. Let X be a set and $Y : X \to \mathcal{U}$ a family of $(n+1)$-connected types. By Exercise 7.6, to show that $\prod_{(x:X)} Y(x)$ is $(n+1)$-connected, it suffices to show that it is (-1)-connected and that all its path-types are n-connected. But also by Exercise 7.6, each $Y(x)$ is (-1)-connected and all its path types are

n-connected. Applying function extensionality to characterize the path types of $\prod_{(x:X)} Y(x)$, the claim follows from the (-1)-connected and n-connected axioms of choice.

Solution to Exercise 7.18

We distinguish the cases $k \leq n$ and $k \geq n$. If $k \leq n$, we know by Lemma 7.5.14 that $\|f\|_n : \|A\|_n \to \|B\|_n$ is an equivalence. We can now prove that $\|f\|_k$ is also an equivalence. This follows from the fact that $\|f\|_k$ is homotopic to $\|\|f\|_n\|_k$, under the equivalence given by Lemma 7.3.15. This homotopy is easily proven by truncation induction. Now since $\|f\|_k$ is an equivalence, it's clearly n-connected.

If $k \geq n$ we apply Lemma 7.5.7. It is sufficient to show that the map $\lambda s.\, s \circ \|f\|_k : \left(\prod_{(b:\|B\|_k)} P(b)\right) \to \left(\prod_{(a:\|A\|_k)} P(\|f\|_k(a))\right)$ has a section, for all $P : \|B\|_k \to \mathcal{U}$. We know that the map $\lambda s.\, s \circ f : \left(\prod_{(b:B)} Q(b)\right) \to \left(\prod_{(a:A)} Q(f(a))\right)$ has a section g, for $Q : B \to \mathcal{U}$ defined by $Q(b) :\equiv P(|b|_k)$. We define $h : \left(\prod_{(a:\|A\|_k)} P(\|f\|_k(a))\right) \to \left(\prod_{(b:\|B\|_k)} P(b)\right)$ by $h(t, |b|_k) :\equiv g(t \circ |{-}|_k, b)$. The function h is indeed a section, which means that we need to show that for $t : \prod_{(a:\|A\|_k)} P(\|f\|_k(a))$ we have $h(t) \circ \|f\|_k = t$. By function extensionality and truncation induction, it is sufficient to show that the functions have the same value when applied to $|a|_k$, for $a : A$. This follows from the following computation

$$\begin{aligned}
(h(t) \circ \|f\|_k)(|a|_k) &\equiv h(t, |f(a)|_k) \\
&\equiv g(t \circ |{-}|_k, f(a)) \\
&= (t \circ |{-}|_k)(a) && \text{(since } g \text{ is a section of } \lambda s.\, s \circ f) \\
&\equiv t(|a|_k).
\end{aligned}$$

Solution to Exercise 7.19

Let us denote by $\mathsf{isConn}(A)$ the proposition that a type A is connected. It is known that

$$\mathsf{isConn}(A) \simeq \|A\| \times \prod_{x,y:A} \|x =_A y\|$$

(see Exercise 7.6).

(i) Suppose a type A is connected. The goal $\prod_{(B,C:\mathcal{U})} \mathsf{isequiv}(e_{A,B,C})$ is a mere proposition, so we can assume $a : A$ and $\prod_{(x,y:A)} \|x = y\|$.

Let B, C be types. We want to construct $h : (A \to B + C) \to (A \to B) + (A \to C)$ such that $h \circ e_{A,B,C} = \mathsf{id}_{(A \to B)+(A \to C)}$ and $e_{A,B,C} \circ h = \mathsf{id}_{A \to B+C}$. Let $f : A \to B + C$ be an arbitrary function. Let us define $h(f) : (A \to B) + (A \to C)$ by case analysis on $f(a) : B + C$:

- if $f(a) \equiv \mathsf{inl}(u)$ for some $u : B$: Let us define $k_1 : B + C \to B$ by

$$\begin{aligned}
k_1(\mathsf{inl}(b)) &:\equiv b; \\
k_1(\mathsf{inr}(c)) &:\equiv u,
\end{aligned}$$

and let $h(f)$ be $\mathsf{inl}(k_1 \circ f)$.

- if $f(a) \equiv \mathsf{inr}(u)$ for some $u : C$: Let us define $k_2 : B + C \to C$ by

$$k_2(\mathsf{inl}(b)) :\equiv u;$$
$$k_2(\mathsf{inr}(c)) :\equiv c,$$

  and let $h(f)$ be $\mathsf{inr}(k_2 \circ f)$.

Let us prove that this h is indeed an inverse of $e_{A,B,C}$. First we prove $e_{A,B,C}(h(f)) = f$ for every $f : A \to B + C$ by case analysis on $f(a)$:

(a) if $f(a) \equiv \mathsf{inl}(u)$ for some $u : B$: We have $\mathsf{inl}(k_1(f(x))) = f(x)$ for every $x : A$ by case analysis on $f(x)$:
   - if $f(x) \equiv \mathsf{inl}(v)$ for some $v : B$: $\mathsf{inl}(k_1(f(x))) = \mathsf{inl}(k_1(\mathsf{inl}(v))) = \mathsf{inl}(v) = f(x)$,
   - if $f(x) \equiv \mathsf{inr}(v)$ for some $v : C$: we have $f(a) \neq f(x)$, which contradicts $\|a =_A x\|$.

   Then we have

$$e_{A,B,C}(h(f)) = e_{A,B,C}(\mathsf{inl}(k_1 \circ f)) = \lambda x. \mathsf{inl}(k_1(f(x))) = f.$$

(b) if $f(a) \equiv \mathsf{inr}(u)$ for some $u : C$: Similar to (a).

Next we prove $h(e_{A,B,C}(f)) = f$ for every $f : (A \to B) + (A \to C)$ by case analysis on f:

(a) if $f \equiv \mathsf{inl}(g)$ for some $g : A \to B$:

$$h(e_{A,B,C}(f)) = h(e_{A,B,C}(\mathsf{inl}(g))) = h(\lambda x. \mathsf{inl}(g(x))) = \mathsf{inl}(\lambda x. g(x)) = f,$$

(b) if $f \equiv \mathsf{inr}(g)$ for some $g : A \to C$:

$$h(e_{A,B,C}(f)) = h(e_{A,B,C}(\mathsf{inr}(g))) = h(\lambda x. \mathsf{inr}(g(x))) = \mathsf{inr}(\lambda x. g(x)) = f.$$

Therefore we have $\mathsf{isequiv}(e_{A,B,C})$.

(ii) (Categorical connectedness to isConn implies LEM): Assume for every type A,

$$\Big(\prod_{B,C:\mathcal{U}} \mathsf{isequiv}(e_{A,B,C})\Big) \to \mathsf{isConn}(A).$$

Assuming $\neg\neg P$ for a mere proposition P, we want to prove P. Let us define $A :\equiv \Sigma P$. Since $P \simeq (\mathsf{N} =_{\Sigma P} \mathsf{S}) \simeq \|\mathsf{N} =_{\Sigma P} \mathsf{S}\|$ (by Lemma 10.1.13 and since P is a mere proposition) and $\mathsf{isConn}(\Sigma P)$ implies $\|\mathsf{N} =_{\Sigma P} \mathsf{S}\|$ (by Exercise 7.6), it suffices to prove $\prod_{(B,C:\mathcal{U})} \mathsf{isequiv}(e_{\Sigma P,B,C})$. Before proving this, we prove that

$$\prod_{x,y:\Sigma P} \neg\neg(x =_{\Sigma P} y) \tag{0.3}$$

by case analysis on x and y:

(a) if $x \equiv \mathsf{N}$:

- if $y \equiv \mathsf{N}$: $\lambda k.\, k(\mathsf{refl}_{\mathsf{N}})$ has type $\neg\neg(\mathsf{N} = \mathsf{N})$.
- if $y \equiv \mathsf{S}$: $\neg\neg(\mathsf{N} = \mathsf{S})$ is equivalent to $\neg\neg P$, so it is inhabited.
- if y varies on $\mathsf{merid}(p)$ for some $p : P$: Since $\neg\neg(\mathsf{N} =_{\Sigma P} y)$ is a mere proposition for every y (see Example 3.6.2), we do not need to consider this case.

(b) if $x \equiv \mathsf{S}$: Similar to (a).

(c) if x varies on $\mathsf{merid}(p)$ for some $p : P$: Since $\prod_{(y:\Sigma P)} \neg\neg(x =_{\Sigma P} y)$ is a mere proposition for every x, we do not need to consider this case.

Let us prove $\prod_{(B,C:\mathcal{U})} \mathsf{isequiv}(e_{\Sigma P,B,C})$. Let B, C be types. We want to construct $h : (\Sigma P \to B + C) \to (\Sigma P \to B) + (\Sigma P \to C)$ such that $h \circ e_{\Sigma P,B,C} = \mathsf{id}_{(\Sigma P \to B)+(\Sigma P \to C)}$ and $e_{\Sigma P,B,C} \circ h = \mathsf{id}_{\Sigma P \to B+C}$.

Let $f : \Sigma P \to B + C$ be an arbitrary function. Let us define $h(f) : (\Sigma P \to B) + (\Sigma P \to C)$ by case analysis on $(f(\mathsf{N}), f(\mathsf{S})) : (B + C) \times (B + C)$:

- if $(f(\mathsf{N}), f(\mathsf{S})) \equiv (\mathsf{inl}(u_1), \mathsf{inl}(u_2))$ for some $u_1, u_2 : B$: Let us define $k_1 : B + C \to B$ by
$$k_1(\mathsf{inl}(b)) :\equiv b;$$
$$k_1(\mathsf{inr}(c)) :\equiv u_1,$$
and let $h(f)$ be $\mathsf{inl}(k_1 \circ f)$.
- if $(f(\mathsf{N}), f(\mathsf{S})) \equiv (\mathsf{inr}(u_1), \mathsf{inr}(u_2))$ for some $u_1, u_2 : C$: Let us define $k_2 : B + C \to C$ by
$$k_2(\mathsf{inl}(b)) :\equiv u_1;$$
$$k_2(\mathsf{inr}(c)) :\equiv c,$$
and let $h(f)$ be $\mathsf{inl}(k_2 \circ f)$.
- if $(f(\mathsf{N}), f(\mathsf{S})) \equiv (\mathsf{inl}(u), \mathsf{inr}(v))$ for some $u : B, v : C$: This case is impossible. $f(\mathsf{N}) \neq f(\mathsf{S})$ implies $\mathsf{N} \neq \mathsf{S}$, which contradicts (0.3).
- if $(f(\mathsf{N}), f(\mathsf{S})) \equiv (\mathsf{inr}(u), \mathsf{inl}(v))$ for some $u : C, v : B$: This case is also impossible.

Let us prove that this h is indeed an inverse of $e_{\Sigma P,B,C}$. First we prove $e_{\Sigma P,B,C}(h(f)) = f$ for every $f : \Sigma P \to B + C$. It is done by case analysis on $(f(\mathsf{N}), f(\mathsf{S})) : (B + C) \times (B + C)$:

(a) if $(f(\mathsf{N}), f(\mathsf{S})) \equiv (\mathsf{inl}(u_1), \mathsf{inl}(u_2))$ for some $u_1, u_2 : B$: We have $\mathsf{inl}(k_1(f(x))) = f(x)$ for every $x : \Sigma P$ by case analysis on $f(x)$:

- if $f(x) \equiv \mathsf{inl}(v)$ for some $v : B$: $\mathsf{inl}(k_1(f(x))) = \mathsf{inl}(k_1(\mathsf{inl}(v))) = \mathsf{inl}(v) = f(x)$,
- if $f(x) \equiv \mathsf{inr}(v)$ for some $v : C$: we have $f(\mathsf{N}) \neq f(x)$ and hence $\mathsf{N} \neq x$, which contradicts (0.3).

Then we have
$$e_{\Sigma P,B,C}(h(f)) = e_{\Sigma P,B,C}(\mathsf{inl}(k_1 \circ f)) = \lambda x.\, \mathsf{inl}(k_1(f(x))) = f.$$

(b) if $(f(\mathsf{N}), f(\mathsf{S})) \equiv (\mathsf{inr}(u_1), \mathsf{inr}(u_2))$ for some $u_1, u_2 : C$: Similar to (a).

Next we prove $h(e_{\Sigma P,B,C}(f)) = f$ for every $f : (\Sigma P \to B) + (\Sigma P \to C)$ by case analysis on f:

(a) if $f \equiv \mathsf{inl}(g)$ for some $g : \Sigma P \to B$:

$$h(e_{\Sigma P,B,C}(f)) = h(e_{\Sigma P,B,C}(\mathsf{inl}(g))) = h(\lambda x.\, \mathsf{inl}(g(x))) = \mathsf{inl}(\lambda x.\, g(x)) = f,$$

(b) if $f \equiv \mathsf{inr}(g)$ for some $g : \Sigma P \to C$:

$$h(e_{\Sigma P,B,C}(f)) = h(e_{\Sigma P,B,C}(\mathsf{inr}(g))) = h(\lambda x.\, \mathsf{inr}(g(x))) = \mathsf{inr}(\lambda x.\, g(x)) = f.$$

Therefore we have $\mathsf{isequiv}(e_{\Sigma P,B,C})$.

(LEM implies categorical connectedness to isConn): Assume LEM, $A : \mathcal{U}$ and

$$\prod_{B,C:\mathcal{U}} \mathsf{isequiv}(e_{A,B,C}).$$

We need to prove $\mathsf{isConn}(A)$. By LEM we have $\|A\|$, because if $\neg\|A\|$ we have $A \simeq \mathbf{0}$ and $\mathbf{1} + \mathbf{1} \simeq \mathbf{1}$, which is not the case. Also by LEM, for every $x, y : A$ we have $\|x = y\| + \neg\|x = y\|$. So if we fix $x : A$ we have a function $f_x : A \to \mathbf{1} + \mathbf{1}$ such that

$$f_x(y) :\equiv \begin{cases} \mathsf{inl}(\star) & \text{if } \|x = y\| \\ \mathsf{inr}(\star) & \text{if } \neg\|x = y\| \end{cases}$$

Applying the assumption to f_x, we obtain

$$e_{A,\mathbf{1},\mathbf{1}}{}^{-1}(f_x) : (A \to \mathbf{1}) + (A \to \mathbf{1})$$

but by definition of e and because $f_x(x) = \mathsf{inl}(\star)$, it must be equal to $\mathsf{inl}(\lambda y.\, \star) : (A \to \mathbf{1}) + (A \to \mathbf{1})$, which means we have $\prod_{(y:A)} \|x = y\|$.

Exercises from Chapter 8

Solution to Exercise 8.1

Remember that the fundamental group of a pointed space is defined as the 0-truncation of the loop space of that space. If we prove the equivalence without applying $\|-\|_0$ to both sides we are done. We use induction on the natural numbers, with base case 1.

Suppose we have two pointed types (A, a) and (B, b). For the base case, we have $\Omega(A \times B) \simeq \Omega(A) \times \Omega(B)$ because it is, by definition, the equivalence of pointed types:

$$\Big((a, b) = (a, b), \mathsf{refl}_{(a,b)}\Big) \simeq \Big((a = a \times b = b), (\mathsf{refl}_a, \mathsf{refl}_b)\Big).$$

By the characterization of Theorem 2.7.2 we see that it suffices to give a:

$$p : \Big((a, b) = (a, b)\Big) \simeq \Big((a = a) \times (b = b)\Big)$$

And to show that transporting $\mathsf{refl}_{(a,b)}$ along p is equal to $(\mathsf{refl}_a, \mathsf{refl}_b)$. To construct p just use Theorem 2.6.2. To show that transport respects the equality we use the definition of our function p, that is just an application of the projections and functoriality (see Eq. (2.6.1)).

For the inductive step we use the inductive hypothesis to get an equivalence: $\Omega^n(A \times B) \simeq \Omega^n(A) \times \Omega^n(B)$. Then we apply $\Omega(-)$ on both sides and use exactly the same propositions we used in the base case. This settles the inductive case, because of the inductive definition of $\Omega^{\mathsf{succ}(n)}(-)$.

Solution to Exercise 8.3

To solve this exercise we must first define the spheres as a type family $\mathbb{S}^- : \mathbb{N} \to \mathcal{U}$. Then we must define the inclusions that appear the in the diagram:

$$\mathbb{S}^0 \to \mathbb{S}^1 \to \mathbb{S}^2 \to \cdots$$

Then we will be able to define the colimit as a higher inductive type.

So, by induction on the natural numbers, we define $\mathbb{S}^- : \mathbb{N} \to \mathcal{U}$ with the base case being the two point type $\mathbf{2}$, and the inductive case being the iterated application of the suspension.

Now we define the inclusions $i_n : \mathbb{S}^n \to \mathbb{S}^{n+1}$. For the base case we have to give a function $i_0 : \mathbf{2} \to \Sigma\mathbf{2}$. This is easy: send one point to N and the other to S.

For the inductive case we notice that the domain of the function i_n that we have to define is the suspension $\Sigma\mathbb{S}^{n-1}$. So we can use the induction of $\Sigma\mathbb{S}^{n-1}$. The codomain is also a suspension, so we can use any of the constructors:

$$\begin{aligned} \mathsf{N}_{n+1} &: \mathbb{S}^{n+1} \\ \mathsf{S}_{n+1} &: \mathbb{S}^{n+1} \\ \mathsf{merid}_{n+1} &: \mathbb{S}^n \to \mathsf{N}_{n+1} = \mathsf{S}_{n+1} \end{aligned}$$

to define our function. We send $\mathsf{N}_n \mapsto \mathsf{N}_{n+1}$ and $\mathsf{S}_n \mapsto \mathsf{S}_{n+1}$. Now we must give a $\Sigma\mathbb{S}^{n-1}$-indexed family of equalities between N_{n+1} and S_{n+1}. By inductive hypothesis we have $i_{n-1} : \mathbb{S}^{n-1} \to \mathbb{S}^n$, so we can use:

$$\mathsf{merid}_{n+1} \circ i_{n-1}$$

It is also interesting to note that this construction works in a more general setting: any function $f : A \to B$ induces a function $\Sigma f : \Sigma A \to \Sigma B$.

Now is a good time to make a drawing to convince oneself that this is the right way to define the inclusions between consecutive spheres, and that this is the diagram intended in the exercise.

Let's define the type $\mathbb{S}^\infty$ as the colimit of the diagram we just constructed. The constructors are:

$$\begin{aligned} j_- &: \prod_{n:\mathbb{N}} \mathbb{S}^n \to \mathbb{S}^\infty \\ \mathsf{glue}_- &: \prod_{(n:\mathbb{N})} \prod_{(x:\mathbb{S}^n)} j_n(x) = j_{n+1}(i_n(x)) \end{aligned}$$

And the induction:

$$\mathsf{ind}_{\mathbb{S}^\infty} : \prod_{(C:\mathbb{S}^\infty \to \mathcal{U})} \prod_{(J_- : \prod_{(n:\mathbb{N})} \prod_{(x:\mathbb{S}^n)} C(j_n(x)))} \Big(\prod_{(n:\mathbb{N})} \prod_{(x:\mathbb{S}^n)} J_n(x) =^{C}_{\mathsf{glue}_n(x)} J_{n+1}(i_n(x)) \Big) \to \prod_{e:\mathbb{S}^\infty} C(e)$$

We also have the usual computational rules. Now, we can interpret the proof that we are going to give as follows. Each sphere is a meridian of the next one, as it passes through the N and S of the bigger sphere. The intuitive idea is that we can retract to a point a circle that is the equator of a sphere. Just as we can retract to a point two points that are the equator of a circle. This idea extends to all n-spheres. If we have all the spheres at the same time we can retract them all. But we have to be careful: the homotopies must be coherent for the total homotopy to be well defined. This can be done in classical homotopy theory using the fact that a CW complex is the colimit of the inclusions between its skeletons. In our case we constructed the colimit. So, if we define a homotopy for each sphere, and prove that this functions coincide in the glued parts (respect the equalities given by glue), by induction on the colimit we get a homotopy defined in the colimit.

The outline of the argument is as follows. It suffices to show that every point in $\mathbb{S}^\infty$ is equal to $j_0(\mathsf{N}_0)$, we prove it in two steps. The first step is to prove it only for the points of the form $j_n(\mathsf{N}_n)$. For the second step we will take an arbitrary $x : \mathbb{S}^\infty$ and, by induction, we will assume it comes from a $y : \mathbb{S}^n$ for some n, that is $x :\equiv j_n(y)$. Then we note that using the commutative diagram given by $\mathsf{glue}_n(x)$:

$$\begin{tikzcd} \mathbb{S}^n \arrow[r, "i_n"] \arrow[dr, "j_n"'] \arrow[r, phantom, shift right=4ex, "\overset{glue_n}{/\!/}"] & \mathbb{S}^{n+1} \arrow[d, "j_{n+1}"] \\ & \mathbb{S}^\infty \end{tikzcd}$$

we get a path $x :\equiv j_n(y) = j_{n+1}(i_n(y))$. But, as we will show, the inclusion i_n of $\mathbb{S}^n$ in $\mathbb{S}^{n+1}$ is nullhomotopic, every point in the image is equal to N_{n+1}. Thus, we are able to show that $j_{n+1}(i_n(y)) = j_{n+1}(\mathsf{N}_{n+1})$. Composing the proof of the first step with the proof of the second step we conclude the exercise.

To construct a $D_- : \prod_{(n:\mathbb{N})} j_n(\mathsf{N}_n) = j_0(\mathsf{N}_0)$ we proceed by induction on n. For the base case we can use $\mathsf{refl}_{j_0(\mathsf{N}_0)}$. For the inductive case, we have by inductive hypothesis $j_n(\mathsf{N}_n) = j_0(\mathsf{N}_0)$. By our definition of i_-, we have that $\mathsf{N}_{n+1} \equiv i_n(\mathsf{N}_n)$. So $j_{n+1}(\mathsf{N}_{n+1})$ equals $j_{n+1}(i_n(\mathsf{N}_n))$. By concatenation with $\mathsf{glue}_n(\mathsf{N}_n)$, we reduce our goal to the inductive hypothesis.

Let's now show that the inclusion of $\mathbb{S}^n$ in $\mathbb{S}^{n+1}$ can be continuously retracted to N_{n+1}. That is, let's construct a homotopy:

$$H_- : \prod_{(n:\mathbb{N})} \prod_{(x:\mathbb{S}^n)} i_n(x) = \mathsf{N}_{n+1}$$

For the case $n \equiv 0$ we know that $i_0(1_{\mathbf{2}}) \equiv \mathsf{N}_1$ and $i_0(0_{\mathbf{2}}) \equiv \mathsf{S}_1$. This is because we constructed the inclusion that way. So we can prove the equalities using $\mathsf{refl}_{\mathsf{N}_1}$ and $\mathsf{merid}_1(1_{\mathbf{2}}) : \mathsf{N}_1 = \mathsf{S}_1$.

For the inductive case we defined, previously, $i_n(\mathsf{N}_n) \equiv \mathsf{N}_{n+1}$ and $i_n(\mathsf{S}_n) \equiv \mathsf{S}_{n+1}$. So we can prove the equalities using $\mathsf{refl}_{\mathsf{N}_{n+1}}$ and $(\mathsf{merid}_{n+1}(\mathsf{N}_n))^{-1}$. Then we have to prove that the function respects merid_n:

$$\prod_{x:\mathbb{S}^{n-1}} \mathsf{refl}_{\mathsf{N}_{n+1}} =^{x\mapsto(i_n(x)=\mathsf{N}_{n+1})}_{\mathsf{merid}_n(x)} (\mathsf{merid}_{n+1}(\mathsf{N}_n))^{-1}$$

By Theorem 2.11.3 (and some straightforward computation) this reduces to:

$$i_n(\mathsf{merid}_n(x)) = \mathsf{merid}_{n+1}(\mathsf{N}_n)$$

But, by our definition of i_-, and the computation rule of the suspension induction $i_n(\mathsf{merid}_n(x))$ equals $\mathsf{merid}_{n+1}(i_{n-1}(x))$. And, by inductive hypothesis, $i_{n-1}(x) = \mathsf{N}_n$, which gives us the desired result.

Composing the two proofs we just gave we get a function:

$$J_n(x) :\equiv \mathsf{glue}_n(x) \cdot \mathsf{ap}_{j_{n+1}} H_n(x) \cdot D_{n+1} : \prod_{(n:\mathbb{N})} \prod_{(x:\mathbb{S}^n)} j_n(x) = j_0(\mathsf{N}_0).$$

We use this function and induction on $\mathbb{S}^\infty$ to derive the contractibility of the space. Now it remains to show that our function respects the gluing:

$$\prod_{(n:\mathbb{N})} \prod_{(x:\mathbb{S}^n)} J_n(x) =^{x\mapsto(x=j_0(\mathsf{N}_0))}_{\mathsf{glue}_n(x)} J_{n+1}(i_n(x))$$

By definition this is:

$$\prod_{(n:\mathbb{N})} \prod_{(x:\mathbb{S}^n)} \mathsf{transport}^{x\mapsto(x=j_0(\mathsf{N}_0))}(\mathsf{glue}_n(x), J_n(x)) = J_{n+1}(i_n(x))$$

The LHS is equal to $\mathsf{glue}_n(x)^{-1} \cdot J_n(x)$, which, by definition of $J_n(x)$, is:

$$\mathsf{glue}_n(x)^{-1} \cdot \mathsf{glue}_n(x) \cdot \mathsf{ap}_{j_{n+1}} H_n(x) \cdot D_{n+1}$$

Cancelling we get:

$$\mathsf{ap}_{j_{n+1}} H_n(x) \cdot D_{n+1}$$

We also use the definition of J_- in the RHS, and then the computation rule of D_-, giving us the equalities:

$$\begin{aligned}
&J_{n+1}(i_n(x)) \\
&= \mathsf{glue}_{n+1}(i_n(x)) \cdot \mathsf{ap}_{j_{n+2}} H_{n+1}(i_n(x)) \cdot D_{n+2} \\
&= \mathsf{glue}_{n+1}(i_n(x)) \cdot \mathsf{ap}_{j_{n+2}} H_{n+1}(i_n(x)) \cdot \mathsf{glue}_{n+1}(\mathsf{N}_{n+1})^{-1} \cdot D_{n+1}.
\end{aligned}$$

So it suffices to show:

$$\mathsf{ap}_{j_{n+1}} H_n(x) = \mathsf{glue}_{n+1}(i_n(x)) \cdot \mathsf{ap}_{j_{n+2}} H_{n+1}(i_n(x)) \cdot \mathsf{glue}_{n+1}(\mathsf{N}_{n+1})^{-1}$$

Or equivalently:

$$\mathsf{ap}_{j_{n+1}} H_n(x) \cdot \mathsf{glue}_{n+1}(\mathsf{N}_{n+1}) = \mathsf{glue}_{n+1}(i_n(x)) \cdot \mathsf{ap}_{j_{n+2}} H_{n+1}(i_n(x))$$

But we remember that we have the homotopy $\mathsf{glue}_{n+1} : j_{n+1} = j_{n+2} \circ i_{n+1}$, so, by a simple application of Lemma 2.4.3 and the functoriality of ap, we get a proof of the equality:

$$\mathsf{ap}_{j_{n+1}} H_n(x) \cdot \mathsf{glue}_{n+1}(\mathsf{N}_{n+1}) = \mathsf{glue}_{n+1}(i_n(x)) \cdot \mathsf{ap}_{j_{n+2}} \mathsf{ap}_{i_{n+1}} H_n(x)$$

So we reduced the goal to showing:

$$\mathsf{ap}_{i_{n+1}} H_n(x) = H_{n+1}(i_n(x))$$

This can be done easily by induction in $\mathbb{S}^n$ using the definition of H_-.

Solution to Exercise 8.4

First we write down the type of the induction principle explicitly:

$$\mathsf{ind}_{\mathbb{S}^\infty} : \prod_{(C:\mathbb{S}^\infty \to \mathcal{U})} \prod_{(n:C(\mathsf{N}))} \prod_{(s:C(\mathsf{S}))} \Big(\prod_{x:\mathbb{S}^\infty} C(x) \to n =^C_{\mathsf{merid}(x)} s \Big) \to \prod_{x:\mathbb{S}^\infty} C(x).$$

We take N as center of contraction. So we have to prove $\prod_{(x:\mathbb{S}^\infty)} \mathsf{N} = x$. For this we use induction on $\mathbb{S}^\infty$ taking:

$$C :\equiv (\lambda x. \mathsf{N} = x) : \mathbb{S}^\infty \to \mathcal{U}.$$

When x is N we just use $\mathsf{refl}_\mathsf{N} : \mathsf{N} = \mathsf{N}$. When x is S we use $\mathsf{merid}(\mathsf{N}) : \mathsf{N} = \mathsf{S}$. When x varies along merid we have to give a function of type:

$$\prod_{x:\mathbb{S}^\infty} \Big(\mathsf{N} = x \Big) \to \Big(\mathsf{refl}_\mathsf{N} =^C_{\mathsf{merid}(x)} \mathsf{merid}(\mathsf{N}) \Big).$$

So, given $x : \mathbb{S}^\infty$ and $p : \mathsf{N} = x$, we have to prove:

$$\mathsf{transport}^{x \mapsto (\mathsf{N}=x)}(\mathsf{merid}(x), \mathsf{refl}_\mathsf{N}) = \mathsf{merid}(\mathsf{N}).$$

By Lemma 2.11.2 it suffices to show $\mathsf{refl}_\mathsf{N} \cdot \mathsf{merid}(x) = \mathsf{merid}(\mathsf{N})$. Canceling refl_N and applying merid to p gets us the desired result.

Solution to Exercise 8.5

We know that every two points $y_1, y_2 : Y$ are merely equal, because Y is is connected. That is, we have a function $c : \prod_{(y_1,y_2:Y)} \|y_1 = y_2\|$. To prove this we can use the remark after Lemma 7.5.11. If we want to show that any pair of points $y_1, y_2 : Y$ are merely equal we can use the first point y_1 to get a pointed space (Y, y_1), and then use the remark.

We note that it suffices to show that for any $y_1, y_2 : Y$ we have $\big\|\mathsf{fib}_f(y_1) = \mathsf{fib}_f(y_2)\big\|$ because $\mathsf{fib}_f(y_1) = \mathsf{fib}_f(y_2)$ implies (using $\mathsf{idtoeqv}$) $\mathsf{fib}_f(y_1) \simeq \mathsf{fib}_f(y_2)$ and thus, by recursion on the truncation of $\mathsf{fib}_f(y_1) = \mathsf{fib}_f(y_2)$, we get that $\big\|\mathsf{fib}_f(y_1) = \mathsf{fib}_f(y_2)\big\|$ implies $\big\|\mathsf{fib}_f(y_1) \simeq \mathsf{fib}_f(y_2)\big\|$.

The type of $c(y_1, y_2)$ is a truncation, so we can use its recursion to prove the desired result. By recursion we can assume that $y_1 = y_2$, and in that case we obviously have $\big\|\mathsf{fib}_f(y_1) = \mathsf{fib}_f(y_2)\big\|$. We also have to show that the proposition we want to prove is -1-truncated, but that is straightforward because it is a -1-truncation.

Exercises from Chapter 9

Solution to Exercise 9.12

Define K to be the precategory with $K_0 :\equiv Y$ and $\hom_K(y_1, y_2) :\equiv (p(y_1) = p(y_2))$. Then $\mathrm{Desc}(A, p) :\equiv A^K$ is a good definition. Moreover, the obvious functor $K \to X$ (where X denotes the discrete category on itself) is a weak equivalence, so Lemma 9.9.2 and Theorem 9.9.4 yield the second and third parts. Finally, K is a strict category, so if it is a stack, then p has a section, while conversely if p has a section then $K \to X$ is a (strong) equivalence.

Exercises from Chapter 10

Solution to Exercise 10.7

Define $A < B$ iff $B \wedge \neg A$.

Solution to Exercise 10.8

Define $a < b$ iff $\exists (n : \mathbb{N}).\, (b_n < a_n)$.