The equivalence of the torus and the product of two circles in homotopy type theory

KRISTINA SOJAKOVA, Carnegie Mellon University

Homotopy type theory is a new branch of mathematics which merges insights from abstract homotopy theory and higher category theory with those of logic and type theory. It allows us to represent a variety of mathematical objects as basic type-theoretic constructions, higher inductive types. We present a proof that in homotopy type theory, the torus is equivalent to the product of two circles. This result indicates that the synthetic definition of torus as a higher inductive type is indeed correct.

CCS Concepts:•**Theory of computation** → **Type theory;**

Additional Key Words and Phrases: homotopy type theory, torus, unit circle, higher inductive type

ACM Reference Format:
Kristina Sojakova, 2015. The equivalence of the torus and the product of two circles in homotopy type theory. *ACM Trans. Comput. Logic* V, N, Article A (January YYYY), 19 pages.
DOI: http://dx.doi.org/10.1145/0000000.0000000

1. INTRODUCTION

Homotopy type theory (HoTT) [The Univalent Foundations Program, Institute for Advanced Study 2013] is a new branch of mathematics which merges insights from abstract homotopy theory and higher category theory with those of logic and type theory. A number of well-known results in algebraic topology have been established within HoTT and formally verified using the proof assistants Agda [Norell 2007] and Coq [Team 2012]; these include the calculation of $\pi_n(\mathbf{S}^n)$ ([Licata and Shulman 2013; Licata and Brunerie 2013]); the Freudenthal Suspension Theorem [The Univalent Foundations Program, Institute for Advanced Study 2013]; the Blakers-Massey Theorem [The Univalent Foundations Program, Institute for Advanced Study 2013], the van Kampen theorem [The Univalent Foundations Program, Institute for Advanced Study 2013], and the Mayer-Vietoris theorem [Cavallo 2014].

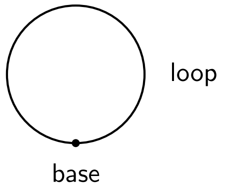
As a formal system, HoTT is an extension of Martin-Löf's dependent type theory with two new concepts: Voevodsky's *univalence axiom* ([Kapulkin et al. 2012; Voevodsky 2011]) and *higher inductive types* ([Lumsdaine 2011; Shulman 2011]). The univalence axiom can be paraphrased as stating that *equivalent types are equal*, and hence we can reason about them using the identity elimination principle. While we do not make an explicit use of the axiom in this paper, we use one of its most important consequences - the function extensionality principle - which states that two pointwise equal functions are in fact equal ([Gambino 2011], Ch. 4.9 of [The Univalent Foundations Program, Institute for Advanced Study 2013]).

The second main feature of HoTT, higher inductive types, are a higher-dimensional generalization of ordinary inductive types which allows us to declare constructors in-

I gratefully acknowledge the support of the Air Force Office of Scientific Research through MURI grant FA9550-15-1-0053. Any opinions, findings and conclusions or recommendations expressed in this material are those of the authors and do not necessarily reflect the views of the AFOSR.
Author's address: K. Sojakova, Computer Science Department, Carnegie Mellon University.

DOI: http://dx.doi.org/10.1145/0000000.0000000

volving the *path spaces* of the type X being defined, rather than just X itself. This means that we can define the higher inductive type X *e.g.*, by means of the constructors $\mathsf{base} : X$, $\mathsf{loop} : \mathsf{base} =_X \mathsf{base}$. While base is an ordinary nullary constructor, akin to the constant 0 in the definition of natural numbers, loop is a term of an identity type over X, not X itself. Intuitively, we can draw the type X as consisting of the point base and a loop from base to base - also known as the circle:



This is not an isolated occurrence: higher inductive types turn out to be well suited for representing a wide variety of mathematical objects, and the definitions generally require very little prior development. Most of the difficult work then lies in showing that such a "synthetic" definition is indeed the "right" one, in the sense that the higher inductive type representing, *e.g.*, the circle or the torus does possess the expected mathematical properties. For instance, we would like to be able to show that in HoTT, the fundamental group of the circle is the group of integers, and that the torus is the product of two circles.

The former result was shown by Licata and Shulman [2013] and notably, the proof they give is much more concise than its homotopy-theoretic counterpart. In this paper, we present the full proof of the latter result that the torus T^2 is equivalent, in a precise sense, to the product $\mathbf{S}^1 \times \mathbf{S}^1$ of two circles. This problem was brought to the author's attention during the Special Year on Univalent Foundations at the Institute for Advanced Study in 2012/2013. During that time, the author gave a sketch of the proof[1] and a year later expanded it into a full writeup [Sojakova 2014], which was included in the HoTT Book exercise solutions file but never published. Later, in summer of 2014, Dan Licata and Guillaume Brunerie produced a similar, formalized proof of the result which builds upon their cubical library for the Agda proof assistant. This proof later appeared in a published paper [Licata and Brunerie 2015]. Licata [2015c] later presented a proof of the same result in cubical type theory. This proof is much simpler since the cubical type theory seems better suited for arguments involving higher paths; however, this new theory is itself still under development.

The main difference between our approach and the one of Licata and Brunerie [2015] is that our proof is essentially self-contained and requires no knowledge of cubical methods. It also provides an example of nontrivial higher path algebra, which might serve as a template for similar future proofs. The proof by Licata and Brunerie uses their cubical library developed for Agda. A number of the intermediate steps in our proof have rough analogues in lemmas found in the cubical library, but here we present them in a form tailored to our specific case. In the conclusion we provide a more detailed comparison of how the proof presented here compares to the one by Licata and Brunerie.

2. PRELIMINARIES

Summarizing from The Univalent Foundations Program, Institute for Advanced Study [2013], HoTT is a dependent type theory with

[1]In personal correspondence, P. Lumsdaine stated he also had a sketch of a proof, which has not been made public.

— dependent pair types $\Sigma_{x:A}B(x)$ and dependent function types $\Pi_{x:A}B(x)$. The non-dependent versions are denoted by $A \times B$ and $A \to B$.
— intensional identity types $x =_A y$. We have the usual formation and introduction rules, where the identity path on $x : A$ will be denoted by 1_x. The elimination and computation rules are recalled below:

$$\frac{E : \Pi_{x,y:A} x =_A y \to \mathtt{type} \qquad d : \Pi_{x:A} E(x, x, 1_x)}{\mathsf{J}(E, d) : \Pi_{x,y:A} \Pi_{p:x=_A y} E(x, y, p)}$$

$$\frac{E : \Pi_{x,y:A} x =_A y \to \mathtt{type} \qquad d : \Pi_{x:A} E(x, x, 1_x) \qquad a : A}{\mathsf{J}(E, d)(a, a, 1_a) \equiv d(a) : E(a, a, 1_a)}$$

As usual, these rules are applicable in any context Γ, which we generally omit. If the type $x =_A y$ is inhabited, we call x and y *equal*. If we do not care about the specific equality witness, we often simply say that $x =_A y$. A term $p : x =_A y$ will be often called a *path* and the process of applying the identity elimination rule will be referred to as *path induction*. Definitional equality between $x, y : A$ will be denoted as $x \equiv y : A$.

Proofs of identity behave much like paths in topological spaces: they can be reversed, concatenated, mapped along functions, etc. Below we summarize a few of these properties:

— For any path $p : x =_A y$ there is a path $p^{-1} : y =_A x$, and we have $(1_x)^{-1} \equiv 1_x$.
— For any paths $p : x =_A y$ and $q : y =_A z$ there is a path $p \cdot q : x =_A z$, and we have $1_x \cdot 1_x \equiv 1_x$.
— Associativity of composition: for any paths $p : x =_A y$, $q : y =_A z$, $r : z =_A u$ we have $(p \cdot q) \cdot r = p \cdot (q \cdot r)$.
— We have $1_x \cdot p = p$ and $p \cdot 1_y = p$ for any $p : x =_A y$.
— For any $p : x =_A y$, $q : y =_A z$ we have $p \cdot p^{-1} = 1_x$, $p^{-1} \cdot p = 1_y$, and $(p^{-1})^{-1} = p$, $(p \cdot q)^{-1} = q^{-1} \cdot p^{-1}$.
— For any $P : A \to \mathtt{type}$ and $p : x =_A y$ there is a function $\mathtt{trans}^P(p) : P(x) \to P(y)$ called the *transport*. We furthermore have $\mathtt{trans}^P(1_x) \equiv \lambda_{x:P(x)} x$.
— We have $\mathtt{trans}^P(p \cdot q) = \mathtt{trans}^P(q) \circ \mathtt{trans}^P(p)$ for any $P : A \to \mathtt{type}$ and $p : x =_A y$, $q : y =_A z$.
— For any function $f : A \to B$ and path $p : x =_A y$, there is a path $\mathsf{ap}_f(p) : f(x) =_B f(y)$ and we have $\mathsf{ap}_f(1_x) \equiv 1_{f(x)}$.
— We have $\mathsf{ap}_f(p^{-1}) = \mathsf{ap}_f(p)^{-1}$ and $\mathsf{ap}_f(p \cdot q) = \mathsf{ap}_f(p) \cdot \mathsf{ap}_f(q)$ for any $f : A \to B$ and $p : x =_A y$, $q : y =_A z$.
— Given a dependent function $f : \Pi_{x:A}B(x)$ and path $p : x =_A y$, there is a path $\mathsf{apd}_f(p) : \mathsf{trans}^B(p, f(x)) =_{B(y)} f(y)$ and we have $\mathsf{apd}_f(1_x) \equiv 1_{f(x)}$.
— All constructs respect propositional equality.

Definition 2.1. For $f, g : \Pi_{x:A}B(x)$, we define the type

$$f \sim g :\equiv \Pi_{a:A}(f(a) =_{B(a)} g(a))$$

and call it the *type of homotopies* between f and g.

Definition 2.2. For $f, g : X \to Y$, $p : x =_X y$, $\alpha : f \sim g$, there is a path

$$\mathsf{nat}_\alpha(p) : \mathsf{ap}_f(p) \cdot \alpha(y) = \alpha(x) \cdot \mathsf{ap}_g(p)$$

defined in the obvious way by induction on p and referred to as the *naturality* of the homotopy α. Pictorially, we have

$$\begin{tikzcd}
f(x) \arrow[r, dash, "\alpha(x)"] \arrow[d, dash, "\mathsf{ap}_f(p)"'] \arrow[dr, phantom, "\mathsf{nat}_\alpha(p)"] & g(x) \arrow[d, dash, "\mathsf{ap}_g(p)"] \\
f(y) \arrow[r, dash, "\alpha(y)"'] & g(y)
\end{tikzcd}$$

A crucial concept in HoTT is that of an equivalence between types.

Definition 2.3. A map $f : A \to B$ is called an *equivalence* if it has both a left and a right inverse (not necessarily identical):

$$\mathsf{iseq}(f) :\equiv \big(\Sigma_{g:B\to A}(g \circ f \sim \mathsf{id}_A)\big) \times \big(\Sigma_{h:B\to A}(f \circ h \sim \mathsf{id}_B)\big)$$

We define

$$(A \simeq B) :\equiv \Sigma_{f:A\to B}\mathsf{iseq}(f)$$

and call A and B *equivalent* if the above type is inhabited.

A much weaker notion is that of *logical equivalence:* we call A and B logically equivalent if there are functions $f : A \to B$, $g : B \to A$. Obviously, we can prove that A and B are equivalent by showing that they are logically equivalent and that the functions f and g in fact compose to identity on both sides; in this case we say that f and g form a *quasi-equivalence* between A and B and call them *quasi-inverses* of each other. This is also a necessary condition: given an equivalence $f : A \to B$ with a left inverse $g : B \to A$ and a right inverse $h : B \to A$, it is not hard to show that $g \sim h$. Thus, f and g (as well as f and h) are quasi-inverses. These observations are summarized in the following lemma:

LEMMA 2.4. *Two types A and B are equivalent if and only if there exist functions $f : A \to B$ and $g : B \to A$ such that $g \circ f \sim \mathsf{id}_A$ and $f \circ g \sim \mathsf{id}_B$.*

It is important to note, however, that the type of quasi-equivalences between A and B is in general *not* equal to $A \simeq B$ (see chapter 4 of The Univalent Foundations Program, Institute for Advanced Study [2013]), hence the word "quasi". From this we can easily show:

LEMMA 2.5. *Equivalence of types is an equivalence relation.*

Many "diagram-like" operations on paths turn out to be equivalences. For instance:

— For any $u : a =_A b$, $v : b =_A d$, $w : a =_A c$, $z : c =_A d$, as in the diagram

$$\begin{tikzcd}
a \arrow[r, dash, "u"] \arrow[d, dash, "w"'] & b \arrow[d, dash, "v"] \\
c \arrow[r, dash, "z"'] & d
\end{tikzcd}$$

we have functions

$$\mathcal{I} : (u \cdot v = w \cdot z) \to (u^{-1} \cdot w \cdot z = v)$$
$$\mathcal{I}^{-1} : (u^{-1} \cdot w \cdot z = v) \to (u \cdot v = w \cdot z)$$

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

Finally, we show how to construct paths in pair and function types. Given two pairs $c, d : A \times B$, we can easily construct a function

$$\mathsf{proj}^{=}_{c,d} : (c = d) \to (\pi_1(c) = \pi_1(d)) \times (\pi_2(c) = \pi_2(d)).$$

We can show:

LEMMA 2.6. *The map* $\mathsf{proj}^{=}_{c,d}$ *is an equivalence for any* $c, d : A \times B$.

We will denote the quasi-inverse of $\mathsf{proj}^{=}_{c,d}$ by $\mathsf{pair}^{=}_{c,d}$. For brevity we will often omit the subscripts.

Analogously, given two functions $f, g : \Pi_{x:A} B(x)$, we can construct a function

$$\mathsf{hap}_{f,g} : (f = g) \to (f \sim g)$$

Showing that this map is an equivalence (or even constructing a map in the opposite direction) is much harder, and is in fact among the chief consequences of the univalence axiom:

LEMMA 2.7. *The map* $\mathsf{hap}_{f,g}$ *is an equivalence for any* $f, g : \Pi_{x:A} B(x)$.

PROOF. See Ch. 4.9 of The Univalent Foundations Program, Institute for Advanced Study [2013]. □

We will denote the quasi-inverse of $\mathsf{hap}_{f,g}$ by $\mathsf{funext}_{f,g}$.

3. THE CIRCLE $\mathbf{S}^1$ AND THE TORUS T^2

The circle $\mathbf{S}^1$ is a higher inductive type generated by the constructors

$$\mathsf{base} : \mathbf{S}^1,$$
$$\mathsf{loop} : \mathsf{base} = \mathsf{base}.$$

The recursion principle says that given a type $C : \mathtt{type}$ and terms

$$b : C,$$
$$l : b = b$$

there exists a function $f : \mathbf{S}^1 \to C$ for which $f(\mathsf{base}) \equiv b$ and $\mathsf{ap}_f(\mathsf{loop}) = l$. We note that the latter "computation rule" is propositional rather than definitional. This is mostly a matter of historical convention, which arose from the fact that the map ap is a defined notion rather than a primitive one, and as such could have several different (but propositionally equal) acceptable forms.

The induction principle says that given a family $E : \mathbf{S}^1 \to \mathtt{type}$ and terms

$$b : E(\mathsf{base}),$$
$$l : \mathsf{trans}^E(\mathsf{loop}, b) = b$$

there exists a function $f : \Pi_{x:\mathbf{S}^1} E(x)$ for which $f(\mathsf{base}) \equiv b$ and $\mathsf{apd}_f(\mathsf{loop}) = l$.

The torus T^2 is a higher inductive type generated by the constructors

$$\mathtt{b} : T^2,$$
$$\mathtt{p} : \mathtt{b} = \mathtt{b},$$
$$\mathtt{q} : \mathtt{b} = \mathtt{b},$$
$$\mathtt{t} : \mathtt{p} \cdot \mathtt{q} = \mathtt{q} \cdot \mathtt{p}$$

as pictured below:

$$\begin{tikzcd}
\mathtt{b} \arrow[r, dash, "\mathtt{p}"] \arrow[d, dash, "\mathtt{q}"'] \arrow[dr, phantom, "\Downarrow \mathtt{t}"] & \mathtt{b} \arrow[d, dash, "\mathtt{q}"] \\
\mathtt{b} \arrow[r, dash, "\mathtt{p}"'] & \mathtt{b}
\end{tikzcd}$$

The recursion principle says that given a type C : $\mathtt{type}$ and terms

$$\begin{aligned}
b' &: C, \\
p' &: b' = b', \\
q' &: b' = b', \\
t' &: p' \cdot q' = q' \cdot p',
\end{aligned}$$

there exists a function $f : T^2 \to C$ for which $f(\mathtt{b}) \equiv b'$ and for which there exist paths $\beta : \mathsf{ap}_f(\mathtt{p}) = p'$ and $\gamma : \mathsf{ap}_f(\mathtt{q}) = q'$ making the following diagram commute:

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

The paths β and γ witness the "computational rules" for the constructors $\mathtt{p}$ and $\mathtt{q}$, and the commutativity of the above diagram corresponds to a similar "computation rule" for the constructor $\mathtt{t}$. Here, each edge represents an equality between its vertices. Unlabeled edges stand for the "obvious" equalities which follow from the basic properties of identity types, such as the path from $\mathsf{ap}_f(\mathtt{p} \cdot \mathtt{q})$ to $\mathsf{ap}_f(\mathtt{p}) \cdot \mathsf{ap}_f(\mathtt{q})$. Edges labeled with, *e.g.*, "via β, γ" stand for an application of congruence: here β is a path from $\mathsf{ap}_f(\mathtt{p})$ to p' and γ is a path from $\mathsf{ap}_f(\mathtt{q})$ to q'. Since path concatenation respects equality, combining β and γ in a straightforward fashion yields a path from $\mathsf{ap}_f(\mathtt{p}) \cdot \mathsf{ap}_f(\mathtt{q})$ to $p' \cdot q'$.

We note that there may be several natural ways how to implement, *e.g.*, the congruence of path concatenation with respect to path equality: we can perform path induction on the first argument, on the second, or on both. For our purposes the exact definition is immaterial as they are all equal up to a higher path, which is why we only specify the arguments (in this case β and γ). From now on, all paths and diagrams will be annotated in this style.

The induction principle for T^2 is quite a bit more complicated; it says that given a family $E : T^2 \to \mathtt{type}$, in order to get a function $f : \Pi_{x:T^2} E(x)$ we require terms

$$
\begin{aligned}
&b' : E(\mathtt{b})\\
&p' : \mathsf{trans}^E(\mathtt{p}, b') = b'\\
&q' : \mathsf{trans}^E(\mathtt{q}, b') = b'\\
&t' : \mathcal{T}_f(E, \mathtt{p}, \mathtt{q}, b') \cdot \mathsf{ap}_{\mathtt{trans}^E(\mathtt{q})}(p') \cdot q' = \mathsf{ap}_{\alpha \mapsto \mathsf{trans}^E(\alpha, b')}(\mathtt{t}) \cdot \mathcal{T}_f(E, \mathtt{q}, \mathtt{p}, b') \cdot \mathsf{ap}_{\mathtt{trans}^E(\mathtt{p})}(q') \cdot p'
\end{aligned}
$$

where for any family $E : T^2 \to \mathtt{type}$, paths $\alpha : x =_{T^2} y$, $\alpha' : y =_{T^2} z$ and point $u : E(x)$, the path

$$\mathcal{T}_f(E, \alpha, \alpha', u) : \mathsf{trans}^E(\alpha \cdot \alpha', u) = \mathsf{trans}^E(\alpha', \mathsf{trans}^E(\alpha, u))$$

is obtained by path induction on α and α'. The function f then has the property that $f(b) \equiv b'$ and there exist paths $\beta : \mathsf{apd}_f(\mathtt{p}) = p'$ and $\gamma : \mathsf{apd}_f(\mathtt{q}) = q'$. Formally, there is an additional higher coherence law representing the "computation rule" on the constructor $\mathtt{t}$. However, this law is not needed to establish the equivalence of $\mathcal{T}^2$ with $\mathbf{S}^1 \times \mathbf{S}^2$ and, more importantly, becomes redundant once we show this equivalence, since it will automatically follow from the fact that $\mathcal{T}^2$ is a 1-type. That result will in turn follow from the fact that the circle $\mathbf{S}^1$, and hence the product $\mathbf{S}^1 \times \mathbf{S}^1$, is a 1-type (Licata and Shulman [2013]).

The term t' can be understood as a dependent counterpart to $\mathtt{t}$, which intends to relate the "composition" of p' and q' with the "composition" of q' with p'. Of course, such compositions only make sense after we insert the necessary transports, as shown in the diagram below; t' then witnesses its commutativity.

$$\begin{tikzcd}
\mathsf{trans}^E(\mathtt{p} \cdot \mathtt{q}, b') \arrow[rr, dash, "\text{via } \mathtt{t}"] \arrow[d, dash] & & \mathsf{trans}^E(\mathtt{q} \cdot \mathtt{p}, b') \arrow[d, dash] \\
\mathsf{trans}^E(\mathtt{q}, \mathsf{trans}^E(\mathtt{p}, b')) \arrow[d, dash, "\text{via } p'"] & & \mathsf{trans}^E(\mathtt{p}, \mathsf{trans}^E(\mathtt{q}, b')) \arrow[d, dash, "\text{via } q'"] \\
\mathsf{trans}^E(\mathtt{q}, b') \arrow[dr, dash, "q'"'] & & \mathsf{trans}^E(\mathtt{p}, b') \arrow[dl, dash, "p'"] \\
& b' &
\end{tikzcd}$$

4. LOGICAL EQUIVALENCE BETWEEN $\mathbf{S}^1 \times \mathbf{S}^1$ AND T^2

Left-to-right. We define a map $\mathbf{f} : \mathbf{S}^1 \to T^2$ by circle recursion, mapping $\mathsf{base} \mapsto \mathtt{b}$ and $\mathsf{loop} \mapsto \mathtt{p}$. Thus, we have a definitional equality $\mathbf{f}(\mathsf{base}) \equiv \mathtt{b}$ and a path $\beta_{\mathbf{f}} : \mathsf{ap}_{\mathbf{f}}(\mathsf{loop}) = \mathtt{p}$. We define a map $\mathcal{F}^{\to} : \mathbf{S}^1 \to \mathbf{S}^1 \to T^2$ again by circle recursion, mapping $\mathsf{base} \mapsto \mathbf{f}$ and $\mathsf{loop} \mapsto \mathsf{funext}(\mathcal{H})$, where $\mathcal{H} : \Pi_{x:\mathbf{S}^1} \big(\mathbf{f}(x) = \mathbf{f}(x)\big)$ is defined by circle induction as follows. We map $\mathsf{base} \mapsto \mathtt{q}$ and loop to the path

$$\begin{tikzcd}
\mathsf{trans}^{z\mapsto \mathbf{f}(z)=\mathbf{f}(z)}(\mathsf{loop},\mathsf{q}) \arrow[d, dash, "\mathcal{T}_1(\mathbf{f},\mathbf{f},\mathsf{loop},\mathsf{q})"] \\
\mathsf{ap}_{\mathbf{f}}(\mathsf{loop})^{-1} \cdot \mathsf{q} \cdot \mathsf{ap}_{\mathbf{f}}(\mathsf{loop}) \arrow[d, dash, "\mathcal{I}(\gamma)"] \\
\mathsf{q}
\end{tikzcd}$$

where for any $f, g : A \to B$, $\alpha : x =_A y$ and $u : f(x) = g(x)$, the path

$$\mathcal{T}_1(f, g, \alpha, u) : \mathsf{trans}^{z\mapsto f(z)=g(z)}(\alpha, u) = \mathsf{ap}_f(\alpha)^{-1} \cdot u \cdot \mathsf{ap}_g(\alpha)$$

is obtained by a straightforward path induction on α, the equivalence $\mathcal{I}$ is as defined on page 4, and γ is the path

$$\begin{tikzcd}
\mathsf{ap}_{\mathbf{f}}(\mathsf{loop}) \cdot \mathsf{q} \arrow[d, dash, "\text{via } \beta_{\mathbf{f}}"] \\
\mathsf{p} \cdot \mathsf{q} \arrow[d, dash, "\mathsf{t}"] \\
\mathsf{q} \cdot \mathsf{p} \arrow[d, dash, "\text{via } \beta_{\mathbf{f}}^{-1}"] \\
\mathsf{q} \cdot \mathsf{ap}_{\mathbf{f}}(\mathsf{loop})
\end{tikzcd}$$

Having defined a function $\mathcal{F}^{\to} : \mathbf{S}^1 \to \mathbf{S}^1 \to T^2$, it is now straightforward to define its uncurried version $\mathcal{F} : \mathbf{S}^1 \times \mathbf{S}^1 \to T^2$. We note that $\mathcal{F}^{\to}(\mathsf{base}) \equiv \mathbf{f}$, and in particular $\mathcal{F}(\mathsf{base}, \mathsf{base}) \equiv \mathsf{b}$. Furthermore, we have a path $\beta_{\mathcal{F}^{\to}} : \mathsf{ap}_{\mathcal{F}^{\to}}(\mathsf{loop}) = \mathsf{funext}(\mathcal{H})$. Since hap and funext form a quasi-equivalence, we have a path

$$\beta^*_{\mathcal{F}^{\to}} : \mathsf{hap}(\mathsf{ap}_{\mathcal{F}^{\to}}(\mathsf{loop})) = \mathcal{H}$$

By definition, the function $\mathcal{H}$ is a homotopy between $\mathbf{f}$ and $\mathbf{f}$ such that $\mathcal{H}(\mathsf{base}) \equiv \mathsf{q}$. Furthermore, the diagram in figure 1 commutes. To show this, we note that for any $f, g : A \to B$, $\alpha : x =_A y$ and $H : f \sim g$, applying $\mathcal{I}^{-1}$ to the path

$$\begin{tikzcd}
\mathsf{ap}_f(\alpha)^{-1} \cdot H(x) \cdot \mathsf{ap}_g(\alpha) \arrow[r, dash, "\mathcal{T}_1(f,g,\alpha,H(x))^{-1}"] & \mathsf{trans}^{z\mapsto f(z)=g(z)}(\alpha, H(x)) \arrow[r, dash, "\mathsf{apd}_H(\alpha)"] & H(y)
\end{tikzcd}$$

yields $\mathsf{nat}_H(\alpha)$: this follows by a path induction on α and a subsequent generalization and path induction on $H(x)$. The second computation rule for $\mathcal{H}$ tells us that

$$\mathsf{apd}_{\mathcal{H}}(\mathsf{loop}) = \mathcal{T}_1(\mathbf{f}, \mathbf{f}, \mathsf{loop}, \mathsf{q}) \cdot \mathcal{I}(\gamma)$$

Thus

$$\mathsf{nat}_{\mathcal{H}}(\mathsf{loop}) = \mathcal{I}^{-1}\big(\mathcal{T}_1(\mathbf{f}, \mathbf{f}, \mathsf{loop}, \mathsf{q})^{-1} \cdot \mathsf{apd}_{\mathcal{H}}(\mathsf{loop})\big) = \gamma$$

which proves the commutativity of the diagram in figure 1.

$$\begin{tikzcd}
\mathsf{ap}_{\mathbf{f}}(\mathsf{loop}) \cdot \mathtt{q} \arrow[rr, dash, "\mathsf{nat}_{\mathcal{H}}(\mathsf{loop})"] \arrow[d, dash, "\text{via } \beta_{\mathbf{f}}"'] & & \mathtt{q} \cdot \mathsf{ap}_{\mathbf{f}}(\mathsf{loop}) \arrow[d, dash, "\text{via } \beta_{\mathbf{f}}"] \\
\mathtt{p} \cdot \mathtt{q} \arrow[rr, dash, "\mathtt{t}"'] & & \mathtt{q} \cdot \mathtt{p}
\end{tikzcd}$$

Fig. 1. Commuting diagram for the homotopy $\mathcal{H}$

$$\begin{tikzcd}
\mathsf{ap}_{\mathcal{G}}(\mathtt{p} \cdot \mathtt{q}) \arrow[rr, dash, "\text{via } \mathtt{t}"] \arrow[d, dash] & & \mathsf{ap}_{\mathcal{G}}(\mathtt{q} \cdot \mathtt{p}) \arrow[d, dash] \\
\mathsf{ap}_{\mathcal{G}}(\mathtt{p}) \cdot \mathsf{ap}_{\mathcal{G}}(\mathtt{q}) \arrow[d, dash, "\text{via } \beta^p_G,\ \beta^q_{\mathcal{G}}"'] & & \mathsf{ap}_{\mathcal{G}}(\mathtt{q}) \cdot \mathsf{ap}_{\mathcal{G}}(\mathtt{p}) \arrow[d, dash, "\text{via } \beta^q_G,\ \beta^p_{\mathcal{G}}"] \\
\mathsf{pair}^{=}(1, \mathsf{loop}) \cdot \mathsf{pair}^{=}(\mathsf{loop}, 1) \arrow[rr, dash, "\Phi_{\mathsf{loop},\mathsf{loop}}"'] & & \mathsf{pair}^{=}(\mathsf{loop}, 1) \cdot \mathsf{pair}^{=}(1, \mathsf{loop})
\end{tikzcd}$$

Fig. 2. Commuting diagram for the map $\mathcal{G}$

Finally, we note that for any $\alpha : x =_{\mathbf{S}^1} x'$ and $\alpha' : y =_{\mathbf{S}^1} y'$, we have path families

$$\begin{aligned}
\mu_x(\alpha') &: \mathsf{ap}_{\mathcal{F}}(\mathsf{pair}^{=}(1_x, \alpha')) = \mathsf{ap}_{\mathcal{F}^{\rightarrow}(x)}(\alpha') \\
\nu_y(\alpha) &: \mathsf{ap}_{\mathcal{F}}(\mathsf{pair}^{=}(\alpha, 1_y)) = \mathsf{hap}(\mathsf{ap}_{\mathcal{F}^{\rightarrow}}(\alpha), y)
\end{aligned}$$

defined by path induction on α' and α respectively.

Right-to-left. We define a function $\mathcal{G} : T^2 \to \mathbf{S}^1 \times \mathbf{S}^1$ by torus recursion as follows. We map $\mathtt{b} \mapsto (\mathsf{base}, \mathsf{base})$, $\mathtt{p} \mapsto \mathsf{pair}^{=}(1_{\mathsf{base}}, \mathsf{loop})$, $\mathtt{q} \mapsto \mathsf{pair}^{=}(\mathsf{loop}, 1_{\mathsf{base}})$, and $\mathtt{t} \mapsto \Phi_{\mathsf{loop},\mathsf{loop}}$, where for any $\alpha : x =_A x'$, $\alpha' : y =_B y'$, the path

$$\Phi_{\alpha,\alpha'} : \Big(\mathsf{pair}^{=}(1_x, \alpha') \cdot \mathsf{pair}^{=}(\alpha, 1_{y'})\Big) = \Big(\mathsf{pair}^{=}(\alpha, 1_y) \cdot \mathsf{pair}^{=}(1_{x'}, \alpha')\Big)$$

is defined by induction on α and α'.

Then we have a definitional equality $\mathcal{G}(\mathtt{b}) \equiv (\mathsf{base}, \mathsf{base})$ and paths

$$\begin{aligned}
\beta^p_{\mathcal{G}} &: \mathsf{ap}_{\mathcal{G}}(\mathtt{p}) = \mathsf{pair}^{=}(1_{\mathsf{base}}, \mathsf{loop}) \\
\beta^q_{\mathcal{G}} &: \mathsf{ap}_{\mathcal{G}}(\mathtt{q}) = \mathsf{pair}^{=}(\mathsf{loop}, 1_{\mathsf{base}})
\end{aligned}$$

which make the diagram in figure 2 commute.

5. EQUIVALENCE BETWEEN $\mathbf{S}^1 \times \mathbf{S}^1$ AND T^2

Left-to-right. We need to show that for any $x, y : \mathbf{S}^1$ we have $\mathcal{G}(\mathcal{F}(x, y)) = (x, y)$. We do this by circle induction on x. We need a path family $\varepsilon : \Pi_{y:\mathbf{S}^1}\big(\mathcal{G}(\mathbf{f}(y)) = (\mathsf{base}, y)\big)$. The definition of ε itself proceeds by circle induction: we map $\mathsf{base} \mapsto 1_{(\mathsf{base},\mathsf{base})}$ and loop to the path

$$\begin{tikzcd}
\mathsf{trans}^{y \mapsto \mathcal{G}(\mathbf{f}(y))=(\mathsf{base},y)}(\mathsf{loop}, 1) \arrow[d, dash, "\mathcal{T}_2^l(\mathcal{F}, \mathcal{G}, \mathsf{base}, \mathsf{loop}, 1)"] \\
\mathsf{ap}_{\mathcal{G}}(\mathsf{ap}_{\mathcal{F}}(\mathsf{pair}^{=}(1, \mathsf{loop})))^{-1} \cdot 1 \cdot \mathsf{pair}^{=}(1, \mathsf{loop}) \arrow[d, dash, "\mathcal{I}(\delta)"] \\
1
\end{tikzcd}$$

where for any maps $f : A \times B \to C$, $g : C \to A \times B$ and paths $\alpha : x =_A x'$, $\alpha' : y =_B y'$, $u : g(f(x, y)) = (x, y)$, the paths

$$\mathcal{T}_2^l(f, g, x, \alpha', u) : \mathsf{trans}^{z \mapsto g(f(x,z))=(x,z)}(\alpha', u) = \mathsf{ap}_g(\mathsf{ap}_f(\mathsf{pair}^{=}(1_x, \alpha')))^{-1} \cdot u \cdot \mathsf{pair}^{=}(1_x, \alpha')$$
$$\mathcal{T}_2^r(f, g, y, \alpha, u) : \mathsf{trans}^{z \mapsto g(f(z,y))=(z,y)}(\alpha, u) = \mathsf{ap}_g(\mathsf{ap}_f(\mathsf{pair}^{=}(\alpha, 1_y)))^{-1} \cdot u \cdot \mathsf{pair}^{=}(\alpha, 1_y)$$

are defined by path induction on α' and α respectively, and δ is the path

$$\begin{tikzcd}
\mathsf{ap}_{\mathcal{G}}(\mathsf{ap}_{\mathcal{F}}(\mathsf{pair}^{=}(1, \mathsf{loop}))) \cdot 1 \arrow[d, dash] \\
\mathsf{ap}_{\mathcal{G}}(\mathsf{ap}_{\mathcal{F}}(\mathsf{pair}^{=}(1, \mathsf{loop}))) \arrow[d, dash, "\text{via } \mu_{\mathsf{base}}(\mathsf{loop})"] \\
\mathsf{ap}_{\mathcal{G}}(\mathsf{ap}_{\mathbf{f}}(\mathsf{loop})) \arrow[d, dash, "\text{via } \beta_{\mathbf{f}}"] \\
\mathsf{ap}_{\mathcal{G}}(\mathsf{p}) \arrow[d, dash, "\beta_{\mathcal{G}}^{p}"] \\
\mathsf{pair}^{=}(1, \mathsf{loop}) \arrow[d, dash] \\
1 \cdot \mathsf{pair}^{=}(1, \mathsf{loop})
\end{tikzcd}$$

This finishes the definition of ε; we note that in particular, $\varepsilon(\mathsf{base}) \equiv 1_{(\mathsf{base},\mathsf{base})}$. We now need to prove that

$$\mathsf{trans}^{x \mapsto \Pi(y:\mathbf{S}^1)(\mathcal{G}(\mathcal{F}(x,y))=(x,y))}(\mathsf{loop}, \varepsilon) = \varepsilon$$

By function extensionality, it suffices to show that for any $y : \mathbf{S}^1$ we have

$$\mathsf{trans}^{x \mapsto \Pi(z:\mathbf{S}^1)(\mathcal{G}(\mathcal{F}(x,z))=(x,z))}(\mathsf{loop}, \varepsilon)\ y = \varepsilon(y)$$

Using the characterization of transports in Π-types, we get the goal

$$\mathsf{trans}^{x \mapsto \mathcal{G}(\mathcal{F}(x,y))=(x,y)}(\mathsf{loop}, \varepsilon(y)) = \varepsilon(y)$$

$$\begin{tikzcd}
\mathsf{trans}^{x\mapsto\mathcal{F}(\mathcal{G}(x))=x}(\mathsf{p},1) \arrow[d, dash, "\mathcal{T}_3(\mathcal{G},\mathcal{F},\mathsf{p},1)"] \\
\mathsf{ap}_{\mathcal{F}}(\mathsf{ap}_{\mathcal{G}}(\mathsf{p}))^{-1} \cdot 1 \cdot \mathsf{p} \arrow[d, dash, "\mathcal{I}(\kappa_p)"] \\
1
\end{tikzcd}$$

a) p'

$$\begin{tikzcd}
\mathsf{trans}^{x\mapsto\mathcal{F}(\mathcal{G}(x))=x}(\mathsf{q},1) \arrow[d, dash, "\mathcal{T}_3(\mathcal{G},\mathcal{F},\mathsf{q},1)"] \\
\mathsf{ap}_{\mathcal{F}}(\mathsf{ap}_{\mathcal{G}}(\mathsf{q}))^{-1} \cdot 1 \cdot \mathsf{q} \arrow[d, dash, "\mathcal{I}(\kappa_q)"] \\
1
\end{tikzcd}$$

b) q'

Fig. 3. The paths p' and q'

By $\mathcal{T}_2^r(\mathcal{F},\mathcal{G},y,\mathsf{loop},\varepsilon(y))$ it suffices to show

$$\mathsf{ap}_{\mathcal{G}}(\mathsf{ap}_{\mathcal{F}}(\mathsf{pair}^=(\mathsf{loop},1_y)))^{-1} \cdot \varepsilon(y) \cdot \mathsf{pair}^=(\mathsf{loop},1_y) = \varepsilon(y)$$

By $\nu_y(\mathsf{loop})$ and $\mathsf{hap}(\beta^*_{\mathcal{F}^\to},y)$, we have $\mathsf{ap}_{\mathcal{F}}(\mathsf{pair}^=(\mathsf{loop},1_y)) = \mathsf{hap}(\mathsf{ap}_{\mathcal{F}^\to}(\mathsf{loop}),y) = \mathcal{H}(y)$. It thus suffices to show

$$\mathsf{ap}_{\mathcal{G}}(\mathcal{H}(y)) \cdot \varepsilon(y) = \varepsilon(y) \cdot \mathsf{pair}^=(\mathsf{loop},1_y)$$

We proceed again by circle induction, this time on y. We map base to the path η below:

$$\begin{tikzcd}
\mathsf{ap}_{\mathcal{G}}(\mathsf{q}) \cdot 1 \arrow[d, dash] \\
\mathsf{ap}_{\mathcal{G}}(\mathsf{q}) \arrow[d, dash, "\beta^q_{\mathcal{G}}"] \\
\mathsf{pair}^=(\mathsf{loop},1) \arrow[d, dash] \\
1 \cdot \mathsf{pair}^=(\mathsf{loop},1)
\end{tikzcd}$$

All that now remains to show is

$$\mathsf{trans}^{y\mapsto \mathsf{ap}_{\mathcal{G}}(\mathcal{H}(y)) \cdot \varepsilon(y)=\varepsilon(y) \cdot \mathsf{pair}^=(\mathsf{loop},1_y)}(\mathsf{loop},\eta) = \eta$$

However, this follows at once from the fact that the circle $\mathbf{S}^1$, and hence the product $\mathbf{S}^1 \times \mathbf{S}^1$, is a 1-type (as shown by Licata and Shulman [2013]): this means that for any two points $x, y : \mathbf{S}^1 \times \mathbf{S}^1$, any two paths $\alpha, \alpha' : x = y$, and any two higher paths $\gamma, \gamma' : \alpha = \alpha'$, we necessarily have $\gamma = \gamma'$.

Right-to-left. We need to show that for any $x : T^2$ we have $\mathcal{F}(\mathcal{G}(x)) = x$. We use torus induction with $\mathsf{b}' :\equiv 1_{\mathsf{b}}$. We let p' and q' be the paths in figure 3, where for any maps $f : A \to B$, $g : B \to C$ and paths $\alpha : x =_A y$, $u : g(f(x)) = x$, the path

$$\mathcal{T}_3(f,g,\alpha,u) : \mathsf{trans}^{z\mapsto g(f(z))=z}(\alpha,u) = \mathsf{ap}_g(\mathsf{ap}_f(\alpha))^{-1} \cdot u \cdot \alpha$$

is defined by path induction on α, and κ_p, κ_q are the paths in figure 4.

$$\begin{tikzcd}
\mathsf{ap}_{\mathcal{F}}(\mathsf{ap}_{\mathcal{G}}(\mathtt{p})) \cdot 1 \arrow[d, dash] \\
\mathsf{ap}_{\mathcal{F}}(\mathsf{ap}_{\mathcal{G}}(\mathtt{p})) \arrow[d, dash, "\text{via } \beta^{p}_{\mathcal{G}}"] \\
\mathsf{ap}_{\mathcal{F}}(\mathsf{pair}^{=}(1, \mathsf{loop})) \arrow[d, dash, "\mu_{\mathsf{base}}(\mathsf{loop})"] \\
\mathsf{ap}_{\mathbf{f}}(\mathsf{loop}) \arrow[d, dash, "\beta_f"] \\
\mathtt{p} \arrow[d, dash] \\
1 \cdot \mathtt{p}
\end{tikzcd}$$

a) κ_p

$$\begin{tikzcd}
\mathsf{ap}_{\mathcal{F}}(\mathsf{ap}_{\mathcal{G}}(\mathtt{q})) \cdot 1 \arrow[d, dash] \\
\mathsf{ap}_{\mathcal{F}}(\mathsf{ap}_{\mathcal{G}}(\mathtt{q})) \arrow[d, dash, "\text{via } \beta^{q}_{\mathcal{G}}"] \\
\mathsf{ap}_{\mathcal{F}}(\mathsf{pair}^{=}(\mathsf{loop}, 1_{\mathsf{base}})) \arrow[d, dash, "\nu_{\mathsf{base}}(\mathsf{loop})"] \\
\mathsf{hap}(\mathsf{ap}_{\mathcal{F}^{\rightarrow}}(\mathsf{loop}), \mathsf{base}) \arrow[d, dash, "\mathsf{hap}(\beta^{*}_{\mathcal{F}^{\rightarrow}}, \mathsf{base})"] \\
\mathtt{q} \arrow[d, dash] \\
1 \cdot \mathtt{q}
\end{tikzcd}$$

b) κ_q

Fig. 4. The paths κ_p and κ_q

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

$$\begin{tikzcd}
\mathsf{trans}^{x \mapsto \mathcal{F}(\mathcal{G}(x))=x}(\mathtt{p} \cdot \mathtt{q}, 1) \arrow[rr, dash, "\text{via } \mathtt{t}"] \arrow[d, dash] & & \mathsf{trans}^{x \mapsto \mathcal{F}(\mathcal{G}(x))=x}(\mathtt{q} \cdot \mathtt{p}, 1) \arrow[d, dash] \\
\mathsf{trans}^{x \mapsto \mathcal{F}(\mathcal{G}(x))=x}(\mathtt{q}, \mathsf{trans}^{x \mapsto \mathcal{F}(\mathcal{G}(x))=x}(\mathtt{p}, 1)) \arrow[d, dash, "\text{via } p'"] & & \mathsf{trans}^{x \mapsto \mathcal{F}(\mathcal{G}(x))=x}(\mathtt{p}, \mathsf{trans}^{x \mapsto \mathcal{F}(\mathcal{G}(x))=x}(\mathtt{q}, 1)) \arrow[d, dash, "\text{via } q'"] \\
\mathsf{trans}^{x \mapsto \mathcal{F}(\mathcal{G}(x))=x}(\mathtt{q}, 1) \arrow[dr, dash, "q'"'] & & \mathsf{trans}^{x \mapsto \mathcal{F}(\mathcal{G}(x))=x}(\mathtt{p}, 1) \arrow[dl, dash, "p'"] \\
& 1 &
\end{tikzcd}$$

We proceed in four steps.

Step 1. The goal of the first step is to get rid of all but the uppermost two transports in the above diagram. We will do this by introducing and proving a generalization of our current situation. Let $\zeta_{G,F}(\alpha_1, \alpha_2, \alpha'_1, \alpha'_2, u_x, u_y, u_z, \eta_1, \eta_2)$ be the path in figure 5.

$$\begin{tikzcd}
\mathsf{ap}_F(\mathsf{ap}_G(\alpha_1 \cdot \alpha_2)) \cdot u_z \arrow[d, dash] \\
\Big(\mathsf{ap}_F(\mathsf{ap}_G(\alpha_1)) \cdot \mathsf{ap}_F(\mathsf{ap}_G(\alpha_2))\Big) \cdot u_z \arrow[d, dash] \\
\mathsf{ap}_F(\mathsf{ap}_G(\alpha_1)) \cdot \Big(\mathsf{ap}_F(\mathsf{ap}_G(\alpha_2)) \cdot u_z\Big) \arrow[d, dash, "\text{via } \eta_2"] \\
\mathsf{ap}_F(\mathsf{ap}_G(\alpha_1)) \cdot (u_y \cdot \alpha_2) \arrow[d, dash] \\
\Big(\mathsf{ap}_F(\mathsf{ap}_G(\alpha_1)) \cdot u_y\Big) \cdot \alpha_2 \arrow[d, dash, "\text{via } \eta_1"] \\
(u_x \cdot \alpha_1) \cdot \alpha_2 \arrow[d, dash] \\
u_x \cdot (\alpha_1 \cdot \alpha_2)
\end{tikzcd}$$

Fig. 5. The path $\zeta_{G,F}(\alpha_1, \alpha_2, \alpha'_1, \alpha'_2, u_x, u_y, u_z, \eta_1, \eta_2)$

Here

- $G : A \to B$ and $F : B \to A$,
- $\alpha_1 : x =_A y$ and $\alpha_2 : y =_A z$ and $\alpha'_1 : a =_A b$ and $\alpha'_2 : b =_A c$,
- $u_x : F(G(x)) = x$ and $u_y : F(G(y)) = y$ and $u_z : F(G(z)) = c$,
- $\eta_1 : \mathsf{ap}_F(\mathsf{ap}_G(\alpha_1)) \cdot u_y = u_x \cdot \alpha'_1$ and $\eta_2 : \mathsf{ap}_F(\mathsf{ap}_G(\alpha_2)) \cdot u_z = u_y \cdot \alpha'_2$.

We claim that the path

$$\begin{tikzcd}
\mathsf{trans}^{w \mapsto F(G(w))=w}(\alpha_1 \cdot \alpha_2, u_x) \arrow[d, dash] \\
\mathsf{trans}^{w \mapsto F(G(w))=w}(\alpha_2, \mathsf{trans}^{w \mapsto F(G(w))=w}(\alpha_1, u_x)) \arrow[d, dash, "\text{via } \mathcal{T}_3(G, F, \alpha_1, u_x) \cdot \mathcal{I}(\eta_1)"] \\
\mathsf{trans}^{w \mapsto F(G(w))=w}(\alpha_2, u_y) \arrow[d, dash, "\mathcal{T}_3(G, F, \alpha_2, u_y) \cdot \mathcal{I}(\eta_2)"] \\
u_z
\end{tikzcd}$$

is equal to the path

$$\begin{tikzcd}
\mathsf{trans}^{w\mapsto F(G(w))=w}(\alpha_1 \cdot \alpha_2, u_x) \arrow[d, dash, "\mathcal{T}_3(G,F,\alpha_1\cdot\alpha_2,u_x)"] \\
\mathsf{ap}_F(\mathsf{ap}_G(\alpha_1\cdot\alpha_2))^{-1}\cdot u_x\cdot(\alpha_1\cdot\alpha_2) \arrow[d, dash, "\mathcal{I}(\zeta_{G,F}(\alpha_1,\alpha_2,u_x,u_y,u_z,\eta_1,\eta_2))"] \\
u_z
\end{tikzcd}$$

for any

— $G : A \to B$ and $F : B \to A$,
— $\alpha_1 : x =_A y$ and $\alpha_2 : y =_A z$,
— $u_x : F(G(x)) = x$ and $u_y : F(G(y)) = y$ and $u_z : F(G(z)) = c$,
— $\eta_1 : \mathsf{ap}_F(\mathsf{ap}_G(\alpha_1)) \cdot u_y = u_x \cdot \alpha_1$ and $\eta_2 : \mathsf{ap}_F(\mathsf{ap}_G(\alpha_2)) \cdot u_z = u_y \cdot \alpha_2$.

To show this, we proceed by path induction on α_1 and α_2. Hence we have to establish the claim for $\alpha_1 :\equiv 1_x$, $\alpha_2 :\equiv 1_x$, $u_x, u_y, u_z : F(G(x)) = x$, and $\eta_1 : 1_{F(G(x))} \cdot u_y = u_x \cdot 1_x$, $\eta_2 : 1_{F(G(x))} \cdot u_z = u_y \cdot 1_x$.

We note, however, that the types of η_1, η_2 are equivalent to $u_x = u_y$ and $u_y = u_z$ respectively. Hence it suffices to show that given $u_x, u_y, u_z : F(G(x)) = x$, $\eta_1' : u_x = u_y$, $\eta_2' : u_y = u_z$, we can establish the claim for the special case when $\eta_1 :\equiv \mathcal{I}_{1,1}\big((\eta_1')^{-1}\big)$ and $\eta_2 :\equiv \mathcal{I}_{1,1}\big((\eta_2')^{-1}\big)$, where for any $u, v : a =_X b$, the map $\mathcal{I}_{1,1} : (u = v) \to (1_a \cdot u = v \cdot 1_b)$ is the obvious equivalence.

We can now perform path induction on η_1' and η_2', giving us $u_x : F(G(x)) = x$ and $\eta_1' :\equiv 1_{u_x}$, $\eta_2' :\equiv 1_{u_x}$. At this point, all terms involving a transport or an ap construct have reduced to reflexivities. Furthermore, the paths η_1, η_2 have reduced to the path $\mathcal{I}_{1,1}(1_{u_x})$ and the paths $\mathcal{T}_3(G, F, \alpha_1, u_x)$, $\mathcal{T}_3(G, F, \alpha_2, u_y)$, $\mathcal{T}_3(G, F, \alpha_1 \cdot \alpha_2, u_x)$ have all reduced to the path $\mathcal{I}\big(\mathcal{I}_{1,1}(1_{u_x})\big)^{-1}$ from u_x to $1_{F(G(x))} \cdot u_x \cdot 1_x$. It is thus possible to generalize the left endpoint of u_x and perform a final path induction, finishing the proof.

By what we have just shown, it suffices to prove that the following diagram commutes:

$$\begin{tikzcd}
\mathsf{trans}^{x\mapsto \mathcal{F}(\mathcal{G}(x))=x}(\mathsf{p}\cdot\mathsf{q}, 1) \arrow[rr, dash, "\text{via } \mathsf{t}"] \arrow[d, dash, "\mathcal{T}_3(\mathcal{G},\mathcal{F},\mathsf{p}\cdot\mathsf{q},1)"] & & \mathsf{trans}^{x\mapsto \mathcal{F}(\mathcal{G}(x))=x}(\mathsf{q}\cdot\mathsf{p}, 1) \arrow[d, dash, "\mathcal{T}_3(\mathcal{G},\mathcal{F},\mathsf{q}\cdot\mathsf{p},1)"] \\
\mathsf{ap}_{\mathcal{F}}(\mathsf{ap}_{\mathcal{G}}(\mathsf{p}\cdot\mathsf{q}))^{-1}\cdot 1\cdot(\mathsf{p}\cdot\mathsf{q}) \arrow[dr, dash, "\mathcal{I}\big(\zeta_{\mathcal{G},\mathcal{F}}(\mathsf{p},\mathsf{q},\mathsf{p},\mathsf{q},1,1,1,\kappa_p,\kappa_q)\big)"'] & & \mathsf{ap}_{\mathcal{F}}(\mathsf{ap}_{\mathcal{G}}(\mathsf{q}\cdot\mathsf{p}))^{-1}\cdot 1\cdot(\mathsf{q}\cdot\mathsf{p}) \arrow[dl, dash, "\mathcal{I}\big(\zeta_{\mathcal{G},\mathcal{F}}(\mathsf{q},\mathsf{p},\mathsf{q},\mathsf{p},1,1,1,\kappa_q,\kappa_p)\big)"] \\
 & 1 &
\end{tikzcd}$$

Step 2. The goal of the second step is to get rid of the remaining transports as well as the applications of $\mathcal{I}$. We first prove the following generalization: given terms

— $G : A \to B$ and $F : B \to A$,
— $\alpha, \alpha' : x =_A y$ and $\theta : \alpha = \alpha'$,
— $u_x : F(G(x)) = x$ and $u_y : F(G(y)) = y$,
— $\eta : \mathsf{ap}_F(\mathsf{ap}_G(\alpha)) \cdot u_y = u_x \cdot \alpha$ and $\eta' : \mathsf{ap}_F(\mathsf{ap}_G(\alpha')) \cdot u_y = u_x \cdot \alpha'$,

the commutativity of the diagram

$$\begin{tikzcd}
\mathsf{trans}^{w \mapsto F(G(w))=w}(\alpha, u_x) \arrow[rr, dash, "\text{via } \theta"] \arrow[d, dash, "\mathcal{T}_3(G,F,\alpha,u_x)"] & & \mathsf{trans}^{w \mapsto F(G(w))=w}(\alpha', u_x) \arrow[d, dash, "\mathcal{T}_3(G,F,\alpha',u_x)"] \\
\mathsf{ap}_F(\mathsf{ap}_G(\alpha))^{-1} \cdot u_x \cdot \alpha \arrow[dr, dash, "\mathcal{I}(\eta)"'] & & \mathsf{ap}_F(\mathsf{ap}_G(\alpha'))^{-1} \cdot u_x \cdot \alpha' \arrow[dl, dash, "\mathcal{I}(\eta')"] \\
& u_y &
\end{tikzcd}$$

is equivalent to the commutativity of the diagram

$$\begin{tikzcd}
\mathsf{ap}_F(\mathsf{ap}_G(\alpha)) \cdot u_y \arrow[r, dash, "\text{via } \theta"] \arrow[d, dash, "\eta"'] & \mathsf{ap}_F(\mathsf{ap}_G(\alpha')) \cdot u_y \arrow[d, dash, "\eta'"] \\
u_x \cdot \alpha \arrow[r, dash, "\text{via } \theta"'] & u_x \cdot \alpha'
\end{tikzcd}$$

To show this, we proceed by path induction on θ and a subsequent path induction on α. After simplifying, it remains to prove that for $u_x, u_y : F(G(x)) = x$ and $\eta, \eta' : 1_{F(G(x))} \cdot u_y = u_x \cdot 1_x$, we have $(\mathcal{I}(\eta) = \mathcal{I}(\eta')) \simeq (\eta = \eta')$. But this follows since $\mathcal{I}$ is an equivalence.

By what we have just shown, it suffices to prove that the following diagram commutes:

$$\begin{tikzcd}
\mathsf{ap}_{\mathcal{F}}(\mathsf{ap}_{\mathcal{G}}(\mathtt{p} \cdot \mathtt{q})) \cdot 1 \arrow[r, dash, "\text{via } \mathtt{t}"] \arrow[d, dash, "\zeta_{\mathcal{G},\mathcal{F}}(\mathtt{p},\mathtt{q},\mathtt{p},\mathtt{q},1,1,1,\kappa_p,\kappa_q)"'] & \mathsf{ap}_{\mathcal{F}}(\mathsf{ap}_{\mathcal{G}}(\mathtt{q} \cdot \mathtt{p})) \cdot 1 \arrow[d, dash, "\zeta_{\mathcal{G},\mathcal{F}}(\mathtt{q},\mathtt{p},\mathtt{q},\mathtt{p},1,1,1,\kappa_q,\kappa_p)"] \\
1 \cdot (\mathtt{p} \cdot \mathtt{q}) \arrow[r, dash, "\text{via } \mathtt{t}"'] & 1 \cdot (\mathtt{q} \cdot \mathtt{p})
\end{tikzcd}$$

Step 3. The goal of the third step is to transform the vertical paths in the above diagram into a more suitable form. As before, we proceed by introducing a generalization. For $k \in \{1,2\}$, let the following terms be given:

— $G : A \to B$ and $F : B \to A$,
— $x_1, x_2, x_3 : A$,
— $\alpha_k^1 : x_k = x_{k+1}$ and $\alpha_k^2 : G(x_k) = G(x_{k+1})$ and $\alpha_k^3, \alpha_k^4 : F(G(x_k)) = F(G(x_{k+1}))$,
— $\iota_k^1 : \mathsf{ap}_G(\alpha_k^1) = \alpha_k^2$ and $\iota_k^2 : \mathsf{ap}_F(\alpha_k^2) = \alpha_k^3$ and $\iota_k^3 : \alpha_k^3 = \alpha_k^4$.

Then the path $\zeta_{G,F}\left(\alpha_1^1, \alpha_2^1, \alpha_1^4, \alpha_2^4, 1, 1, 1, \eta_1, \eta_2\right)$, where η_k is the path in **a)** below, is equal to the path in **b)**. This follows right away by a one-sided path induction on $\iota_k^1, \iota_k^2, \iota_k^3$ and a subsequent path induction on α_k^1.

$$\begin{tikzcd}
\mathsf{ap}_F(\mathsf{ap}_G(\alpha_k^1)) \cdot 1 \arrow[d, dash] \\
\mathsf{ap}_F(\mathsf{ap}_G(\alpha_k^1)) \arrow[d, dash, "\text{via } \iota_k^1"] \\
\mathsf{ap}_F(\alpha_k^2) \arrow[d, dash, "\iota_k^2"] \\
\alpha_k^3 \arrow[d, dash, "\iota_k^3"] \\
\alpha_k^4 \arrow[d, dash] \\
1 \cdot \alpha_k^4
\end{tikzcd}$$

a)

$$\begin{tikzcd}
\mathsf{ap}_F(\mathsf{ap}_G(\alpha_1^1 \cdot \alpha_2^1)) \cdot 1 \arrow[d, dash] \\
\mathsf{ap}_F(\mathsf{ap}_G(\alpha_1^1 \cdot \alpha_2^1)) \arrow[d, dash] \\
\mathsf{ap}_F\left(\mathsf{ap}_G(\alpha_1^1) \cdot \mathsf{ap}_G(\alpha_2^1)\right) \arrow[d, dash, "\text{via } \iota_1^1, \iota_2^1"] \\
\mathsf{ap}_F(\alpha_1^2 \cdot \alpha_2^2) \arrow[d, dash] \\
\mathsf{ap}_F(\alpha_1^2) \cdot \mathsf{ap}_F(\alpha_2^2) \arrow[d, dash, "\text{via } \iota_1^2, \iota_2^2"] \\
\alpha_1^3 \cdot \alpha_2^3 \arrow[d, dash, "\text{via } \iota_1^3, \iota_2^3"] \\
\alpha_1^4 \cdot \alpha_2^4 \arrow[d, dash] \\
1 \cdot (\alpha_1^4 \cdot \alpha_2^4)
\end{tikzcd}$$

b)

By what we have just shown, it suffices to prove that the outer rectangle in the following diagram commutes:

$$\begin{tikzcd}
\mathsf{ap}_{\mathcal{F}}(\mathsf{ap}_{\mathcal{G}}(\mathtt{p}\cdot\mathtt{q}))\cdot 1 \arrow[rr, dash, "\text{via } \mathtt{t}"] \arrow[dd, dash] & & \mathsf{ap}_{\mathcal{F}}(\mathsf{ap}_{\mathcal{G}}(\mathtt{q}\cdot\mathtt{p}))\cdot 1 \arrow[dd, dash] \\
& \text{A} & \\
\mathsf{ap}_{\mathcal{F}}(\mathsf{ap}_{\mathcal{G}}(\mathtt{p}\cdot\mathtt{q})) \arrow[rr, dash, "\text{via } \mathtt{t}"] \arrow[d, dash] & & \mathsf{ap}_{\mathcal{F}}(\mathsf{ap}_{\mathcal{G}}(\mathtt{q}\cdot\mathtt{p})) \arrow[d, dash] \\
\mathsf{ap}_{\mathcal{F}}\Big(\mathsf{ap}_{\mathcal{G}}(\mathtt{p})\cdot\mathsf{ap}_{\mathcal{G}}(\mathtt{q})\Big) \arrow[d, dash, "\text{via } \beta^{p}_{\mathcal{G}},\, \beta^{q}_{\mathcal{G}}"] & \text{B} & \mathsf{ap}_{\mathcal{F}}\Big(\mathsf{ap}_{\mathcal{G}}(\mathtt{q})\cdot\mathsf{ap}_{\mathcal{G}}(\mathtt{p})\Big) \arrow[d, dash, "\text{via } \beta^{q}_{\mathcal{G}},\, \beta^{p}_{\mathcal{G}}"'] \\
\mathsf{ap}_{\mathcal{F}}\Big(\mathsf{pair}^{=}(1,\mathsf{loop})\cdot\mathsf{pair}^{=}(\mathsf{loop},1)\Big) \arrow[rr, dash, "\text{via } \Phi_{\mathsf{loop},\mathsf{loop}}"] \arrow[d, dash] & & \mathsf{ap}_{\mathcal{F}}\Big(\mathsf{pair}^{=}(\mathsf{loop},1)\cdot\mathsf{pair}^{=}(1,\mathsf{loop})\Big) \arrow[d, dash] \\
\mathsf{ap}_{\mathcal{F}}(\mathsf{pair}^{=}(1,\mathsf{loop}))\cdot\mathsf{ap}_{\mathcal{F}}(\mathsf{pair}^{=}(\mathsf{loop},1)) \arrow[d, dash, "\text{via } \mu_{\mathsf{base}}(\mathsf{loop}),\, \nu_{\mathsf{base}}(\mathsf{loop})"] & \text{C} & \mathsf{ap}_{\mathcal{F}}(\mathsf{pair}^{=}(\mathsf{loop},1))\cdot\mathsf{ap}_{\mathcal{F}}(\mathsf{pair}^{=}(1,\mathsf{loop})) \arrow[d, dash, "\text{via } \nu_{\mathsf{base}}(\mathsf{loop}),\, \mu_{\mathsf{base}}(\mathsf{loop})"'] \\
\mathsf{ap}_{\mathbf{f}}(\mathsf{loop})\cdot\mathsf{hap}(\mathsf{ap}_{\mathcal{F}\to}(\mathsf{loop}),\mathsf{base}) \arrow[rr, dash, "\mathsf{nat}_{\mathsf{hap}(\mathsf{ap}_{\mathcal{F}\to}(\mathsf{loop}))}(\mathsf{loop})"'] \arrow[dd, dash, "\text{via } \beta_{\mathbf{f}},\, \mathsf{hap}(\beta^{*}_{\mathcal{F}\to},\mathsf{base})"] & & \mathsf{hap}(\mathsf{ap}_{\mathcal{F}\to}(\mathsf{loop}),\mathsf{base})\cdot\mathsf{ap}_{\mathbf{f}}(\mathsf{loop}) \arrow[dd, dash, "\text{via } \mathsf{hap}(\beta^{*}_{\mathcal{F}\to},\mathsf{base}),\, \beta_{\mathbf{f}}"'] \\
& \text{D} & \\
\mathtt{p}\cdot\mathtt{q} \arrow[rr, dash, "\mathtt{t}"'] \arrow[dd, dash] & & \mathtt{q}\cdot\mathtt{p} \arrow[dd, dash] \\
& \text{E} & \\
1\cdot(\mathtt{p}\cdot\mathtt{q}) \arrow[rr, dash, "\text{via } \mathtt{t}"'] & & 1\cdot(\mathtt{q}\cdot\mathtt{p})
\end{tikzcd}$$

Step 4. It suffices to prove that each of the inner rectangles commutes. Rectangles A and E commute obviously. Rectangle B is just the diagram from figure 2 transported along $\mathsf{ap}_{\mathcal{F}}$, and hence commutes. Rectangle C commutes by the following generalization: for any $\alpha : x =_{\mathbf{S}^1} y$, the diagram below commutes by path induction on α:

$$\begin{tikzcd}
\mathsf{ap}_{\mathcal{F}}\Big(\mathsf{pair}^{=}(1_x,\alpha)\cdot\mathsf{pair}^{=}(\alpha,1_y)\Big) \arrow[r, dash, "\text{via } \Phi_{\alpha,\alpha}"] \arrow[d, dash] & \mathsf{ap}_{\mathcal{F}}\Big(\mathsf{pair}^{=}(\alpha,1_x)\cdot\mathsf{pair}^{=}(1_y,\alpha)\Big) \arrow[d, dash] \\
\mathsf{ap}_{\mathcal{F}}(\mathsf{pair}^{=}(1_x,\alpha))\cdot\mathsf{ap}_{\mathcal{F}}(\mathsf{pair}^{=}(\alpha,1_y)) \arrow[d, dash, "\text{via } \mu_x(\alpha),\, \nu_y(\alpha)"] & \mathsf{ap}_{\mathcal{F}}(\mathsf{pair}^{=}(\alpha,1_x))\cdot\mathsf{ap}_{\mathcal{F}}(\mathsf{pair}^{=}(1_y,\alpha)) \arrow[d, dash, "\text{via } \nu_x(\alpha),\, \mu_y(\alpha)"'] \\
\mathsf{ap}_{\mathcal{F}^{\to}(x)}(\alpha)\cdot\mathsf{hap}(\mathsf{ap}_{\mathcal{F}^{\to}}(\alpha),y) \arrow[r, dash, "\mathsf{nat}_{\mathsf{hap}(\mathsf{ap}_{\mathcal{F}^{\to}}(\alpha))}(\alpha)"'] & \mathsf{hap}(\mathsf{ap}_{\mathcal{F}^{\to}}(\alpha),x)\cdot\mathsf{ap}_{\mathcal{F}^{\to}(y)}(\alpha)
\end{tikzcd}$$

It remains to show that rectangle D commutes. Consider the following diagram:

$$\begin{tikzcd}
\mathsf{ap}_{\mathsf{f}}(\mathsf{loop})\cdot\mathsf{hap}(\mathsf{ap}_{\mathcal{F}^{\to}}(\mathsf{loop}),\mathsf{base}) \arrow[r, dash, "\mathsf{nat}_{\mathsf{hap}(\mathsf{ap}_{\mathcal{F}^{\to}}(\mathsf{loop}))}(\mathsf{loop})"'] \arrow[d, dash, "\text{via } \mathsf{hap}(\beta^{*}_{\mathcal{F}^{\to}},\mathsf{base})"] & \mathsf{hap}(\mathsf{ap}_{\mathcal{F}^{\to}}(\mathsf{loop}),\mathsf{base})\cdot\mathsf{ap}_{\mathsf{f}}(\mathsf{loop}) \arrow[d, dash, "\text{via } \mathsf{hap}(\beta^{*}_{\mathcal{F}^{\to}},\mathsf{base})"'] \\
\mathsf{ap}_{\mathsf{f}}(\mathsf{loop})\cdot\mathtt{q} \arrow[r, dash, "\mathsf{nat}_{\mathcal{H}}(\mathsf{loop})"'] \arrow[d, dash, "\text{via } \beta_{\mathsf{f}}"] \arrow[ur, phantom, "\mathrm{D}_1"] & \mathtt{q}\cdot\mathsf{ap}_{\mathsf{f}}(\mathsf{loop}) \arrow[d, dash, "\text{via } \beta_{\mathsf{f}}"'] \\
\mathtt{p}\cdot\mathtt{q} \arrow[r, dash, "\mathtt{t}"'] \arrow[ur, phantom, "\mathrm{D}_2"] & \mathtt{q}\cdot\mathtt{p}
\end{tikzcd}$$

Commutativity of the outer rectangle clearly implies the commutativity of D. It thus remains to show that D_1 and D_2 commute. The rectangle D_2 is precisely the diagram in figure 1, which commutes. Rectangle D_1 commutes by the following generalization: let $f, g : A \to B$, $h_1, h_2 : f \sim g$, $\gamma : h_1 = h_2$ and $\alpha : x =_A y$ be given. Then the following diagram commutes by path induction on γ and α:

$$\begin{tikzcd}
\mathsf{ap}_f(\alpha)\cdot h_1(y) \arrow[r, dash, "\mathsf{nat}_{h_1}(\alpha)"] \arrow[d, dash, "\text{via } \mathsf{hap}(\gamma,y)"'] & h_1(x)\cdot\mathsf{ap}_g(\alpha) \arrow[d, dash, "\text{via } \mathsf{hap}(\gamma,x)"] \\
\mathsf{ap}_f(\alpha)\cdot h_2(y) \arrow[r, dash, "\mathsf{nat}_{h_2}(\alpha)"'] & h_2(x)\cdot\mathsf{ap}_g(\alpha)
\end{tikzcd}$$

This finishes the proof.

6. CONCLUSION

We have presented a homotopy-type theoretic proof that the torus T^2 is equivalent to the product of two circles $\mathbf{S}^1 \times \mathbf{S}^1$. To compare the proof described here to the one given by Licata and Brunerie [2015], we first note that the definitions of the back-and-forth functions between T^2 and $\mathbf{S}^1 \times \mathbf{S}^1$ are exactly the same. When proving that the functions compose to the identity on $\mathbf{S}^1 \times \mathbf{S}^1$ we used the fact that the circle $\mathbf{S}^1$ is a 1-type. This simplification is not used by Licata and Brunerie; the lines *75-76, 82-86* in [Licata 2015a] comprise the path algebra which would be avoided by the aforementioned simplification. On the other hand, in this fashion Agda is able to automatically infer the terms loop1-case and loop2-case, which in our notation correspond to the paths η and δ respectively (of course a paper proof offers no such opportunity).

Similarly, when proving that the functions compose to the identity on $\mathcal{T}^2$, the terms p-case and q-case, which in our proof correspond to the paths κ_q and κ_p, are inferred automatically. Steps 1 and 2 of our proof roughly correspond to lines *403-441* in [Licata 2015b] and *51-57* in [Licata 2015a]; in both proofs, the purpose of these steps is to mediate between a diagram involving transports (a "square-over") and an equivalent diagram which does not (a "cube"). Steps 3 and 4 then roughly correspond to lines *60-67* in [Licata 2015a]; the commuting diagrams (or "cubes") established in Step 4 are composed together, using a reordering of operations that is justified by Step 3.

ACKNOWLEDGMENT

The author would like to thank her advisors, Profs. Steve Awodey and Frank Pfenning, for their help.

REFERENCES

E. Cavallo. 2014. The Mayer-Vietoris sequence in homotopy type theory. (2014). Talk at the Oxford Workshop on Homotopy Type Theory.

N. Gambino. 2011. The univalence axiom and function extensionality. (2011). Talk at the Oberwolfach Mini-Workshop on the Homotopy Interpretation of Constructive Type Theory. Proof due to V. Voevodsky; notes by K. Kapulkin and P. Lumsdaine available at http://www.pitt.edu/~krk56/UA_implies_FE.pdf.

C. Kapulkin, P. Lumsdaine, and V. Voevodsky. 2012. The Simplicial Model of Univalent Foundations. (2012). Available at arxiv.org as arXiv:1211.2851v1.

D. Licata. 2015a. Agda repository. (2015). Available at https://github.com/dlicata335/hott-agda/blob/master/homotopy/TS1S1.agda.

D. Licata. 2015b. Agda repository. (2015). Available at https://github.com/dlicata335/hott-agda/blob/master/lib/cubical/Cube.agda.

D. Licata. 2015c. Cubical type theory. (2015). Talk at the homotopy type theory seminar, Carnegie Mellon University. Slides available from the author's website.

D. Licata and G. Brunerie. 2013. $\Pi_n(\mathbf{S}^n)$ in Homotopy Type Theory. In *Certified Programs and Proofs (LNCS)*, Vol. 8307. Springer, 1–16.

D. Licata and G. Brunerie. 2015. A Cubical Approach to Synthetic Homotopy Theory. In *Logic in Computer Science (LICS 2015)*. IEEE, 92–103.

D. Licata and M. Shulman. 2013. Calculating the Fundamental Group of the Circle in Homotopy Type Theory. In *Logic in Computer Science (LICS 2013)*. IEEE Computer Society, 223–232.

P. Lumsdaine. 2011. Higher inductive types: a tour of the menagerie. (2011). Post on the Homotopy Type Theory blog.

U. Norell. 2007. *Towards a practical programming language based on dependent type theory*. Ph.D. Dissertation. Chalmers University of Technology.

M. Shulman. 2011. Homotopy Type Theory, VI. (2011). Post on the n-category cafe blog.

K. Sojakova. 2014. The torus T^2 is equivalent to the product $\mathbf{S}^1 \times \mathbf{S}^1$ of two circles. (2014). Available at http://ncatlab.org/homotopytypetheory/files/torus.pdf.

Coq Development Team. 2012. *The Coq Proof Assistant Reference Manual, version 8.4pl3*. INRIA. Available at coq.inria.fr.

The Univalent Foundations Program, Institute for Advanced Study. 2013. *Homotopy Type Theory - Univalent Foundations of Mathematics*. Univalent Foundations Project.

V. Voevodsky. 2011. Univalent foundations of mathematics. (2011). Invited talk at the Workshop on Logic, Language, Information and Computation (WoLLIC 2011).