On the Calculation of Fundamental Groups in Homotopy Type Theory by Means of Computational Paths

Tiago Mendonça Lucena de Veras¹, Arthur F. Ramos², Ruy J. G. B. de Queiroz², and Anjolina G. de Oliveira²

¹ Departamento de Matemática
Universidade Federal Rural de Pernambuco
tiago.veras@ufrpe.br

² Centro de Informática
Universidade Federal de Pernambuco
afr@cin.ufpe.br
ruy@cin.ufpe.br
ago@cin.ufpe.br

Abstract. One of the most interesting entities of homotopy type theory is the identity type. It gives rise to an interesting interpretation of the equality, since one can semantically interpret the equality between two terms of the same type as a collection of homotopical paths between points of the same space. Since this is only a semantical interpretation, the addition of paths to the syntax of homotopy type theory has been recently proposed by De Queiroz, Ramos and De Oliveira [7,12]. In these works, the authors propose an entity known as ‘computational path’, proposed by De Queiroz and Gabbay in 1994 [8], and show that it can be used to formalize the identity type. We have found that it is possible to use these computational paths as a tool to achieve one central result of algebraic topology and homotopy type theory: the calculation of fundamental groups of surfaces. We review the concept of computational paths and the \textit{LND}_{EQ}-\textit{TRS}, which is a term rewriting system proposed by De Oliveira in 1994 [2] to map redundancies between computational paths. We then proceed to calculate the fundamental group of the circle, cylinder, Möbius band, torus and the real projective plane. Moreover, we show that the use of computational paths make these calculations simple and straightforward, whereas the same result is much harder to obtain using the traditional code-encode-decode approach of homotopy type theory.

Keywords. Fundamental group, computational paths, homotopy type theory, algebraic topology, term rewriting system.

1 Introduction

One of the most interesting and intriguing concepts of Martin-Löf’s Type Theory is the identity type. It has been thoroughly studied since the discovery of
the so-called Univalent Models by Vladimir Voevodsky in 2005 [14]. This discovery gave rise to a new area of research known as homotopy type theory. It is based on a groundbreaking semantic interpretation of the identity type: a witness $p : Id_A(a,b)$ can be seen as a homotopical path between the points $a$ and $b$ within a topological space $A$. As one can see in [13,1], this interpretation generated groundbreaking results. Nonetheless, one has to assert the fact that this interpretation is a semantical one, i.e., those paths are not part of the syntax of type theory.

Based on the fact that paths are seen only as semantic entities in homotopy type theory, De Queiroz, De Oliveira and Ramos have recently proposed a way of adding those paths to the syntax of type theory. They have been proposing the addition of an entity called computational path to type theory. This entity would work as the syntactic counterpart of semantic paths. Thus, one could interpret the identity type as syntactic paths between two terms of a given type. The full development of this theory can be seen in two recently published papers, [7] and [12].

Here the theory of computational paths is going to be thoroughly used. Our purpose is to further explore the study of fundamental groups of many surfaces using homotopy type theory. Nevertheless, we find that the main technique currently used in homotopy type theory to study fundamental groups looks too complex and seems to ask for more clarity. We are referring to the code-encode-decode technique, extensively used in many proofs of traditional homotopy type theory, as one can see in [13]. Since paths are not present in the syntax of traditional type theory, one needs to use this technique to simulate the path-space. Thus, we found in the theory of computational paths an effective way of calculating the fundamental groups without the use of what seems to be overly complex techniques such as the code-encode-decode one. With that in mind, we are going to review the main concepts of the theory of computational paths, as proposed in [7,12], but we shall stop short of diving into more advanced concepts of this theory.

Since the notion of fundamental groups of surfaces is one of the central topics of interest in algebraic topology, the main topic of interest of this work is the calculation of these fundamental groups by means of computational paths. Given a surface $S$, the fundamental group is obtained by studying the paths on the surface starting and ending at the same point $x_0 \in S$. These paths are called loops $x_0$ and $x_0 \in S$ is the base point. But, as it will become clear later on, we are not interested in all kinds of loops, but only the ones that are not homotopic to the point $x_0$ (since they are homotopically the same and this path will be denoted by $\rho_{x_0}$, meaning ‘the reflexivity path’).

First we will calculate the fundamental group of the circle, $S^1$. We will use its proof to obtain the fundamental group of the cylinder, $C$, and that of the Möbius band, $K^2$. Next, we will get the fundamental group of two much more complex surfaces, the Torus $T^2$ and the real projective plane $R^2$. 
2 Computational Paths

Before we enter in details of computational paths, let’s recall what motivated the introduction of computational paths to type theory. In type theory, our types are interpreted using the so-called Brower-Heyting-Kolmogorov Interpretation. That way, a semantic interpretation of types are not given by truth-values, but by the concept of proof as a primitive notion. Thus, we have [7]:

**A proof of the proposition:**

- $A \land B$  
  - a proof of $A$ and a proof of $B$
- $A \lor B$  
  - a proof of $A$ or a proof of $B$
- $A \rightarrow B$  
  - a function that turns a proof of $A$ into a proof of $B$
- $\forall x^D.P(x)$  
  - a function that turns an element $a$ into a proof of $P(a)$
- $\exists x^D.P(x)$  
  - an element $a$ (witness) and a proof of $P(a)$

Also, based on the Curry-Howard functional interpretation of logical connectives, one have [7]:

**A proof of the proposition:**

- $A \land B$  
  - $(p, q)$ where $p$ is a proof of $A$ and $q$ is a proof of $B$
- $A \lor B$  
  - $i(p)$ where $p$ is a proof of $A$ or $j(q)$ where $q$ is a proof of $B$  
  - (‘$i$’ and ‘$j$’ abbreviate ‘into the left/right disjunct’)
- $A \rightarrow B$  
  - $\lambda x.b(x)$ where $b(p)$ is a proof of $B$ provided $p$ is a proof of $A$
- $\forall x^A.B(x)$  
  - $Ax.f(x)$ where $f(a)$ is a proof of $B(a)$ provided $a$ is an arbitrary individual chosen from the domain $A$
- $\exists x^A.B(x)$  
  - $\varepsilon x.(f(x), a)$ where $a$ is a witness from the domain $A$, $f(a)$ is a proof of $B(a)$

If one looks closely, there is one interpretation missing in the BHK-Interpretation. What constitutes a proof of $t_1 = t_2$? In other words, what is a proof of an equality statement? We answer this by proposing that an equality between those two terms should be a sequence of rewrites starting from $t_1$ and ending at $t_2$. Thus, we would have [7]:

**A proof of the proposition:**

- $t_1 = t_2$  
  - ? 
  - (Perhaps a sequence of rewrites starting from $t_1$ and ending in $t_2$?)

We call computational path the sequence of rewrites between these terms.
### 2.1 Formal Definition

Since computational path is a generic term, it is important to emphasize the fact that we are using the term computational path in the sense defined by [5]. A computational path is based on the idea that it is possible to formally define when two computational objects $a, b : A$ are equal. These two objects are equal if one can reach $b$ from $a$ applying a sequence of axioms or rules. This sequence of operations forms a path. Since it is between two computational objects, it is said that this path is a computational one. Also, an application of an axiom or a rule transforms (or rewrite) an term in another. For that reason, a computational path is also known as a sequence of rewrites. Nevertheless, before we define formally a computational path, we can take a look at one famous equality theory, the $\lambda\beta\eta - equality$ [9]:

**Definition 1.** The $\lambda\beta\eta - equality$ is composed by the following axioms:

1. $(\alpha)$ $\lambda x. M = \lambda y. M[y/x]$ if $y \notin FV(M)$;
2. $(\beta)$ $(\lambda x. M)N = M[N/x]$;
3. $(\rho)$ $M = M$;
4. $(\eta)$ $(\lambda x. Mx) = M$ $(x \notin FV(M))$.

And the following rules of inference:

- $(\mu)$ $\frac{M = M'}{NM = NM'}$
- $(\tau)$ $\frac{M = N}{M = P}$
- $(\nu)$ $\frac{M = M'}{MN = M'N}$
- $(\sigma)$ $\frac{M = N}{N = M}$
- $(\xi)$ $\frac{M = M'}{\lambda x. M = \lambda x. M'}$

**Definition 2.** [7] $P$ is $\beta$-equal or $\beta$-convertible to $Q$ (notation $P =_{\beta} Q$) iff $Q$ is obtained from $P$ by a finite (perhaps empty) series of $\beta$-contractions and reversed $\beta$-contractions and changes of bound variables. That is, $P =_{\beta} Q$ iff there exist $P_0, \ldots, P_n$ $(n \geq 0)$ such that $P_0 \equiv P$, $P_n \equiv Q$, $(\forall i \leq n - 1)(P_i \triangleright_{1,\beta} P_{i+1} \lor P_{i+1} \triangleright_{1,\beta} P_i \lor P_i \equiv_{\alpha} P_{i+1})$.

(Note that equality has an existential force, which will show in the proof rules for the identity type.)

The same happens with $\lambda\beta\eta$-equality:

**Definition 3.** $(\lambda\beta\eta - equality)$ [7] The equality-relation determined by the theory $\lambda\beta\eta$ is called $=_{\beta\eta}$; that is, we define

$$M =_{\beta\eta} N \iff \lambda\beta\eta \vdash M = N.$$
Example 1. Take the term \( M \equiv (\lambda x. (\lambda y. yx)(\lambda w. zw))v \). Then, it is \( \beta\eta \)-equal to \( N \equiv vz \) because of the sequence:
\[
(\lambda x. (\lambda y. yx)(\lambda w. zw))v, \quad (\lambda x. (\lambda y. yx)z)v, \quad (\lambda y. yv)z, \quad vz
\]
which starts from \( M \) and ends with \( N \), and each member of the sequence is obtained via 1-step \( \beta \)- or \( \eta \)-contraction of a previous term in the sequence. To take this sequence into a path, one has to apply transitivity twice, as we do in the example below.

Example 2. The term \( M \equiv (\lambda x. (\lambda y. yx)(\lambda w. zw))v \) is \( \beta\eta \)-equal to \( N \equiv vz \) because of the sequence:
\[
(\lambda x. (\lambda y. yx)(\lambda w. zw))v, \quad (\lambda x. (\lambda y. yx)z)v, \quad (\lambda y. yv)z, \quad vz
\]
Now, taking this sequence into a path leads us to the following:
The first is equal to the second based on the grounds:
\( \eta((\lambda x. (\lambda y. yx)(\lambda w. zw))v, (\lambda x. (\lambda y. yx)z)v) \)
The second is equal to the third based on the grounds:
\( \beta((\lambda x. (\lambda y. yx)z)v, (\lambda y. yv)z) \)
Thus, the first one is equal to the fourth one based on the grounds:
\( \tau((\lambda x. (\lambda y. yx)(\lambda w. zw))v, (\lambda x. (\lambda y. yx)z)v, \beta((\lambda x. (\lambda y. yx)z)v, (\lambda y. yv)z)), \beta((\lambda y. yv)z, vz)) \).

The aforementioned theory establishes the equality between two \( \lambda \)-terms. Since we are working with computational objects as terms of a type, we need to translate the \( \lambda\beta\eta \)-equality to a suitable equality theory based on Martin L"of’s type theory. We obtain:

\textbf{Definition 4.} The equality theory of Martin L"of’s type theory has the following basic proof rules for the \( \Pi \)-type:

\[
\begin{align*}
(\beta) & \quad \frac{N : A}{M : B} \quad \frac{[x : A]}{\lambda x. M}N = M[N/x : B[N/x]} \\
(\rho) & \quad \frac{M : A}{M = M : A} \\
(\sigma) & \quad \frac{M = N : A}{N = M : A} \\
(\tau) & \quad \frac{M = N : A \quad N = P : A}{M = P : A} \\
(\eta) & \quad \frac{M : (\Pi x : A)B}{(\lambda x. Mx) = M : (\Pi x : A)B} \quad (x \notin \text{FV}(M))
\end{align*}
\]
We are finally able to formally define computational paths:

**Definition 5.** Let \( a \) and \( b \) be elements of a type \( A \). Then, a computational path \( s \) from \( a \) to \( b \) is a composition of rewrites (each rewrite is an application of the inference rules of the equality theory of type theory or is a change of bound variables). We denote that by \( a =_s b \).

As we have seen in example 2, composition of rewrites are applications of the rule \( \tau \). Since change of bound variables is possible, each term is considered up to \( \alpha \)-equivalence.

### 2.2 Equality Equations

One can use the aforementioned axioms to show that computational paths establishes the three fundamental equations of equality: the reflexivity, symmetry and transitivity:

\[
\begin{align*}
\frac{a = t : A}{a =_{\tau(t,a)} c : A} & \quad \text{transitivity} \\
\frac{a : A}{a =_{\rho} a : A} & \quad \text{reflexivity} \\
\frac{a =_t b : A}{b =_{\sigma(t)} a : A} & \quad \text{symmetry}
\end{align*}
\]

### 2.3 Identity Type

We have said that it is possible to formulate the identity type using computational paths. As we have seen, the best way to define any formal entity of type theory is by a set of natural deductions rules. Thus, we define our path-based approach as the following set of rules:

- **Formation and Introduction rule** [7][12]:

\[
\frac{A \text{ type} \quad a : A \quad b : A}{Id_A(a,b) \text{ type} \quad Id - F} \\
\frac{a =_s b : A \quad s(a,b) : Id_A(a,b)}{Id - I}
\]

One can notice that our formation rule is exactly equal to the traditional identity type. From terms \( a, b : A \), one can form that is inhabited only if there is a proof of equality between those terms, i.e., \( Id_A(a,b) \).

The difference starts with the introduction rule. In our approach, one can notice that we do not use a reflexive constructor \( r \). In other words, the reflexive path is not the main building block of our identity type. Instead, if
we have a computational path \( a =_s b : A \), we introduce \( s(a, b) \) as a term of the identity type. That way, one should see \( s(a, b) \) as a sequence of rewrites and substitutions (i.e., a computational path) which would have started from \( a \) and arrived at \( b \).

- Elimination rule \([7,12]\):

\[
\begin{align*}
[a =_g b : A] \\
m : \text{Id}_A(a,b) & \quad h(g) : C \\
\text{REW}R(m, \hat{g}.h(g)) : C \\
\text{Id} - E
\end{align*}
\]

Let’s recall the notation being used. First, one should see \( h(g) \) as a functional expression \( h \) which depends on \( g \). Also, one should notice the the use of ‘\( \hat{\cdot} \)’ in \( \hat{g} \). One should see ‘\( \hat{\cdot} \)’ as an abstractor that binds the occurrences of the variable \( g \) introduced in the local assumption \([a =_g b : A]\) as a kind of Skolem-type constant denoting the reason why \( a \) was assumed to be equal to \( b \).

We also introduce the constructor \( \text{REW}R \). In a sense, it is similar to the constructor \( J \) of the traditional approach, since both arise from the elimination rule of the identity type. The behavior of \( \text{REW}R \) is simple. If from a computational path \( g \) that establishes the equality between \( a \) and \( b \) one can construct \( h(g) : C \), then if we also have this equality established by a term \( m : \text{Id}_A(a,b) \), we can put together all this information in \( \text{REW}R \) to construct \( C \), eliminating the type \( \text{Id}_A(a,b) \) in the process. The idea is that we can substitute \( g \) for \( m \) in \( \hat{g}.h(g) \), resulting in \( h(m/g) : C \). This behavior is established next by the reduction rule.

- Reduction rule \([7,12]\):

\[
\begin{align*}
[a =_g b : A] \\
\begin{array}{c}
\text{REW}R(m, \hat{g}.h(g)) : C \\
\text{Id} - E
\end{array} & \quad \begin{array}{c}
[a =_g b : A] \\
h(g) : C \\
\text{REW}R(m, \hat{g}.h(g)) : C \\
\text{Id} - E \\
\end{array} & \quad \begin{array}{c}
[a =_m b : A] \\
h(m/g) : C
\end{array}
\end{align*}
\]

- Induction rule:

\[
\begin{align*}
e : \text{Id}_A(a,b) & \quad [a =_t b : A] \\
\text{REW}R(e, \hat{t}.t(a,b)) : \text{Id}_A(a,b) & \quad \text{Id} - I \\
\text{REW}R(e, \hat{t}.t(a,b)) : \text{Id}_A(a,b) & \quad \text{Id} - E \\
\end{align*}
\]

Our introduction and elimination rules reassert the concept of equality as an existential force. In the introduction rule, we encapsulate the idea that a witness of a identity type \( \text{Id}_A(a,b) \) only exists if there exist a computational path establishing the equality of \( a \) and \( b \). Also, one can notice that our elimination rule is similar to the elimination rule of the existential quantifier.
The objective of this subsection is to show how to use in practice the rules that we have just defined. The idea is to show construction of terms of some important types. The constructions that we have chosen to build are the reflexive, transitive and symmetric type of the identity type. Those were not random choices. The main reason is the fact that reflexive, transitive and symmetric types are essential to the process of building a groupoid model for the identity type \[10\]. As we shall see, these constructions come naturally from simple computational paths constructed by the application of axioms of the equality of type theory.

Before we start the constructions, we think that it is essential to understand how to use the eliminations rules. The process of building a term of some type is a matter of finding the right reason. In the case of \(J\), the reason is the correct \(x, y : A\) and \(z : \text{Id}_A(a, b)\) that generates the adequate \(C(x, y, z)\). In our approach, the reason is the correct path \(a =_g b\) that generates the adequate \(g(a, b) : \text{Id}(a, b)\).

### Reflexivity

One could find strange the fact that we need to prove the reflexivity. Nevertheless, just remember that our approach is not based on the idea that reflexivity is the base of the identity type. As usual in type theory, a proof of something comes down to a construction of a term of a type. In this case, we need to construct a term of type \(\Pi(a : A) \Pi \text{Id}_A(a, a)\). The reason is extremely simple: from a term \(a : A\), we obtain the computational path \(a =_\rho a : A\) \[12\]:

\[
\begin{array}{c}
[a : A] \\
\hline
a =_\rho a : A \\
\hline
\rho(a, a) : \text{Id}_A(a, a) \\
\hline
\lambda a, \rho(a, a) : \Pi_{(a : A)} \text{Id}_A(a, a) \\
\hline
\Pi - I
\end{array}
\]

### Symmetry

The second proposed construction is the symmetry. Our objective is to obtain a term of type \(\Pi_{(a : A)} \Pi_{(b : A)} (\text{Id}_A(a, b) \rightarrow \text{Id}_A(b, a))\).

We construct a proof using computational paths. As expected, we need to find a suitable reason. Starting from \(a =_t b\), we could look at the axioms of \textit{definition 4.1} to plan our next step. One of those axioms makes the symmetry clear: the \(\sigma\) axiom. If we apply \(\sigma\), we will obtain \(b =_{\sigma(t)} a\). From this, we can then infer that \(\text{Id}_A\) is inhabited by \((\sigma(t))(b, a)\). Now, it is just a matter of applying the elimination \[12\]:

\[
\begin{array}{c}
[a : A] \\
\hline
a =_t b : A \\
\hline
\sigma(a, b) : \text{Id}_A(a, b) \\
\hline
\lambda a, \sigma(a, b) : \Pi_{(a : A)} \text{Id}_A(a, a) \\
\hline
\Pi - I
\end{array}
\]
Transitivity. The third and last construction will be the transitivity. Our objective is to obtain a term of type $\Pi(a : A)\Pi(b : A)\Pi(c : A)(Id_A(a, b) \to Id_A(b, c) \to Id_A(a, c))$.

To build our path-based construction, the first step, as expected, is to find the reason. Since we are trying to construct the transitivity, it is natural to think that we should start with paths $a =_t b$ and $b =_u c$ and then, from these paths, we should conclude that there is a path $z$ that establishes that $a =_z c$. To obtain $z$, we could try to apply the axioms of definition 4.1. Looking at the axioms, one is exactly what we want: the axiom $\tau$. If we apply $\tau$ to $a =_t b$ and $b =_u c$, we will obtain a new path $\tau(t, u)$ such that $a =_{\tau(t, u)} c$. Using that construction as the reason, we obtain the following term [12]:
As one can see, each step is just straightforward applications of introduction, elimination rules and abstractions. The only idea behind this construction is just the simple fact that the axiom $\tau$ guarantees the transitivity of paths.

3 A Term Rewriting System for Paths

As we have just shown, a computational path establishes when two terms of the same type are equal. From the theory of computational paths, an interesting case arises. Suppose we have a path $s$ that establishes that $a =_s b : A$ and a path $t$ that establishes that $a =_t b : A$. Consider that $s$ and $t$ are formed by distinct compositions of rewrites. Is it possible to conclude that there are cases that $s$ and $t$ should be considered equivalent? The answer is yes. Consider the following example:

Example 3. Consider the path $a =_t b : A$. By the symmetry property, we obtain $b =_{\sigma(t)} a : A$. What if we apply the property again on the path $\sigma(t)$? We would obtain a path $a =_{\sigma(\sigma(t))} b : A$. Since we applied symmetry twice in succession, we obtained a path that is equivalent to the initial path $t$. For that reason, we conclude the act of applying symmetry twice in succession is a redundancy. We say that the path $\sigma(\sigma(t))$ can be reduced to the path $t$.

As one could see in the aforementioned example, different paths should be considered equal if one is just a redundant form of the other. The example that we have just seen is just a straightforward and simple case. Since the equality theory has a total of 7 axioms, the possibility of combinations that could generate redundancies are rather high. Fortunately, most possible redundancies were thoroughly mapped by [2]. In that work, a system that establishes redundancies and creates rules that solve them was proposed. This system, known as $LND_{EQ}$-TRS, originally mapped a total of 39 rules. For each rule, there is a proof tree that constructs it. We included all rules in appendix B. To illustrate those rules, take the case of example 2. We have the following [7]:

$$
\frac{x =_t y : A}{y =_{\sigma(t)} x : A} \quad \triangleright_{ss} \quad x =_t y : A
$$

It is important to notice that we assign a label to every rule. In the previous case, we assigned the label $ss$.

Definition 6. (rw-rule [12]) An rw-rule is any of the rules defined in $LND_{EQ}$-TRS.

Definition 7. (rw-contraction [12]) Let $s$ and $t$ be computational paths. We say that $s \triangleright_{1rw} t$ (read as: $s$ rw-contracts to $t$) iff we can obtain $t$ from $s$ by an application of only one rw-rule. If $s$ can be reduced to $t$ by finite number of rw-contractions, then we say that $s \triangleright_{rw} t$ (read as $s$ rw-reduces to $t$).
Definition 8. (rw-equality [1]) Let $s$ and $t$ be computational paths. We say that $s =_{rw} t$ (read as: $s$ is $rw$-equal to $t$) iff $t$ can be obtained from $s$ by a finite (perhaps empty) series of $rw$-contractions and reversed $rw$-contractions. In other words, $s =_{rw} t$ iff there exists a sequence $R_0, \ldots, R_n$, with $n \geq 0$, such that

\[
(\forall i \leq n - 1)(R_i \triangleright_{1rw} R_{i+1} \text{ or } R_{i+1} \triangleright_{1rw} R_i) \quad R_0 \equiv s, \quad R_n \equiv t
\]

Proposition 1. $rw$-equality is transitive, symmetric and reflexive.

Proof. Comes directly from the fact that $rw$-equality is the transitive, reflexive and symmetric closure of $rw$.

The above proposition is rather important, since sometimes we want to work with paths up to $rw$-equality. For example, we can take a path $s$ and use it as a representative of an equivalence class, denoting this by $[s]_{rw}$.

We’d like to mention that $LND_{EQ}$-TRS is terminating and confluent. The proof of this can be found in [2,4,3,6]. One should refer to [7,6] for a more complete and detailed explanation of the rules of $LND_{EQ}$-TRS.

4 Fundamental Group of surfaces obtained by means of Computational Paths

The objective of this section is to show that it is possible to use computational paths to obtain the fundamental group of the some surfaces, and this is one of the main results of homotopy theory. We avoid again the use of the heavy and rather complicated machinery of the code-encode-decode approach. In what follows we will get the fundamental group of some surfaces.

4.1 Fundamental Group of Circle $S^1$

Definition 9 (The circle $S^1$). The circle is the type generated by:

- A point - base : $S^1$
- A computational path - base $\tau_{loop}$ : $S^1$.

The first thing one should notice is that this definition does not use only the points of the type $S^1$, but also a computational path $loop$ between those points. That is why it is called a higher inductive type [13]. Our approach differs from the one developed in the HTT book on the fact that we do not need to simulate the path-space between those points, since computational paths do exist in the syntax of the theory. Thus, if one starts with a path $base \tau_{loop}$ : $S^1$, one can naturally obtain additional paths applying the path-axioms $\rho$, $\tau$ and $\sigma$. Thus, one has a path $\sigma(loop) = loop^{-1}$, $\tau(loop, loop)$, etc. In Martin-Löf’s type theory,
the existence of those additional paths comes from establishing that the paths should be freely generated by the constructors [13]. In our approach, we do not have to appeal to this kind of argument, since all paths come naturally from direct applications of the axioms and the inference rules which define the theory of equality.

With that in mind, one can define the fundamental group of a circle. In homotopy theory, the fundamental group is the one formed by all equivalence classes up to homotopy of paths (loops) starting from a point \( a \) and also ending at \( a \). Since the we use computational paths as the syntactic counterpart of homotopic paths in type theory, we use it to propose the following definition:

**Definition 10 \((\Pi_1(A, a) \text{ structure})\).** \(\Pi_1(A, a)\) is a structure defined as follows:

\[
\Pi_1(A, a) = \{[\text{loop}]_{rw} \mid a = a : A\}
\]

We use this structure to define the fundamental group of a circle. We also need to show that it is indeed a group.

**Proposition 2.** \((\Pi_1(S, a), \circ)\) is a group.

**Proof.** The first thing to define is the group operation \(\circ\). Given any \( a = a : S^1 \) and \( a = a : S^1 \), we define \( r \circ s \) as \( \tau(s, r) \). Thus, we now need to check the group conditions:

- **Closure:** Given \( a = a : S^1 \) and \( a = a : S^1 \), \( r \circ s \) must be a member of the group. Indeed, \( r \circ s = \tau(s, r) \) is a computational path \( a = a : S^1 \).

- **Inverse:** Every member of the group must have an inverse. Indeed, if we have a path \( r \), we can apply \( \sigma(r) \). We claim that \( \sigma(r) \) is the inverse of \( r \), since we have:

\[
\sigma(r) \circ r = \tau(r, \sigma(r)) = r \\
\rho \circ \sigma(r) = \tau(\sigma(r), r) = \rho
\]

Since we are working up to \( rw \)-equality, the equalities hold strictly.

- **Identity:** We use the path \( a = a : S^1 \) as the identity. Indeed, we have:

\[
r \circ \rho = \tau(\rho, r) = r \\
\rho \circ r = \tau(r, \rho) = r
\]

- **Associativity:** Given any members of the group \( a = a : S^1 \), \( a = a \) and \( a = a \), we want that \( r \circ (s \circ t) = (r \circ s) \circ t \):

\[
r \circ (s \circ t) = \tau(\tau(t, s), r) = \tau(t, \tau(s, r)) = (r \circ s) \circ t
\]
All conditions have been satisfied. \((\Pi_1(S, a), \circ)\) is a group.

Thus, \((\Pi_1(S, a), \circ)\) is indeed a group. We call this group the fundamental group of \(S^1\). Therefore, the objective of this section is to show that \(\Pi_1(S, a) \simeq \mathbb{Z}\).

Before we start to develop this proof, the following lemma will prove to be useful:

**Lemma 1.** All paths generated by a path \(a = a\) are \(rw\)-equal to a path \(\text{loop}^n\), for \(n \in \mathbb{Z}\).

We have said that from a \(\text{loop}\), one can freely generate different paths by applying composition \(\tau\) and the symmetry \(\sigma\). Thus, one can, for example, obtain something as \(\text{loop} \circ \text{loop} \circ \text{loop}^{-1} \circ \text{loop} \ldots\). Our objective with this lemma is to show that, in fact, this path can be reduced to a path of the form \(\text{loop}^n\), for \(n \in \mathbb{Z}\).

**Proof.** The idea is to proceed by induction. We start from a base \(\rho\). For the base case, it is trivially true, since we define it to be equal to \(\text{loop}^0\). From \(\rho\), one can construct more complex paths by composing with \(\text{loop}\) or \(\sigma(\text{loop})\) on each step. We have the following induction steps:

- A path of the form \(\rho\) concatenated with \(\text{loop}\): We have \(\rho \circ \text{loop} = \tau(\text{loop}, \rho) = \text{loop} = \text{loop}^1\):

- A path of the form \(\rho\) concatenated with \(\sigma(\text{loop})\): We have \(\rho \circ \sigma(\text{loop}) = \tau(\sigma(\text{loop}), \rho) = \sigma(\text{loop}) = \text{loop}^{-1}\)

- A path of the form \(\text{loop}^n\) concatenated with \(\text{loop}\): We have \(\text{loop}^n \circ \text{loop} = \text{loop}^{n+1}\).

- A path of the form \(\text{loop}^n\) concatenated with \(\sigma(\text{loop})\): We have \(\text{loop}^n \circ \sigma(\text{loop}) = \text{loop}^{n-1} \circ (\text{loop} \circ (\sigma(\text{loop}) \circ \text{loop})) = \text{loop}^{n-1} \circ (\tau(\sigma(\text{loop}), \text{loop})) = \text{loop}^{n-1} \circ \rho = \tau(\rho, \text{loop}^{n-1}) = \text{loop}^{n-1}\)

- A path of the form \(\text{loop}^{-n}\) concatenated with \(\text{loop}\): We have \(\text{loop}^{-n} = \text{loop}^{-(n-1)} \circ \text{loop}^{-1} = \text{loop}^{-(n-1)} \circ \sigma(\text{loop})\). Thus, we have \((\text{loop}^{-(n-1)} \circ \sigma(\text{loop}) \circ \text{loop}) = \text{loop}^{-(n-1)} \circ (\sigma(\text{loop}) \circ \text{loop}) = \text{loop}^{-(n-1)} \circ \tau(\text{loop}, \sigma(\text{loop})) = \text{loop}^{-n} \circ \rho = \tau(\rho, \text{loop}^{-n}) = \text{loop}^{-n}\).

Thus, every path is of the form \(\text{loop}^n\), with \(n \in \mathbb{Z}\).

This lemma shows that every path of the fundamental group can be represented by a path of the form \(\text{loop}^n\), with \(n \in \mathbb{Z}\).
Theorem 1. $\Pi_1(S,a) \simeq \mathbb{Z}$

To prove this theorem, one could use the approach proposed in [13], defining a pair of encode and decode functions. Nevertheless, since our computational paths are already part of the syntax, one does not need to rely on this kind of approach to simulate a path-space. We can work directly with the concept of path.

Proof. The proof is done by establishing a function from $\Pi_1(S,a)$ to $\mathbb{Z}$ and then an inverse from $\mathbb{Z}$ to $\Pi_1(S,a)$. Since we have access to the previous lemma, this task is not too difficult. The main idea is that the $n$ in $\text{loop}^n$ means the amount of times one goes around the circle, while the sign gives the direction (clockwise or anti-clockwise). In other words, it is the winding number. Since we have shown that every path of the fundamental group is of the form $\text{loop}^n$, with $n \in \mathbb{Z}$, then we just need to translate $\text{loop}^n$ to an integer $n$ and an integer $n$ to a path $\text{loop}^n$. We define two functions, $\text{toInteger} : \Pi_1(S,a) \rightarrow \mathbb{Z}$ and $\text{toPath} : \mathbb{Z} \rightarrow \Pi_1(S,a)$:

- $\text{toInteger}$: To define this function, we use two functions defined in $\mathbb{Z}$: the successor function $\text{succ}$ and the predecessor function $\text{pred}$. We define $\text{toInteger}$ as follows. Of course, we use directly the fact that every path of $\Pi_1(S,a)$ is of the form $\text{loop}^n$ with $n \in \mathbb{Z}$:

$$
\text{toInteger} : \begin{cases} 
\text{toInteger}(\text{loop}^n \equiv \rho) = 0 & n = 0 \\
\text{toInteger}(\text{loop}^n) = \text{succ}(\text{toInteger}(\text{loop}^{n-1})) & n > 0 \\
\text{toInteger}(\text{loop}^n) = \text{pred}(\text{toInteger}(\text{loop}^{n+1})) & n < 0
\end{cases}
$$

- $\text{toPath}$: We just need to transform an integer $n$ into a path $\text{loop}^n$:

$$
\text{toPath} : \begin{cases} 
\text{toPath}(n) = \rho & n = 0 \\
\text{toPath}(n) = \text{toPath}(n - 1) \circ \text{loop} & n > 0 \\
\text{toPath}(n) = \text{toPath}(n + 1) \circ \sigma(\text{loop}) & n < 0
\end{cases}
$$

That they are inverses is a straightforward check. Therefore, we have $\Pi_1(S,a) \simeq \mathbb{Z}$.

The technique used to obtain the fundamental group of the circle will provide us with the means to obtain the fundamental group of the cylinder and the Möbius Band. In these cases, all proofs used in the circle can be used, thus we will focus on knowing what kinds of loops are in our interest and on getting the bijections. In the case of the torus and projective real plane, we will need more advanced analysis and tests.

4.2 Fundamental Group of Cylinder $C$

We are interested in getting the fundamental group of a cylindrical surface $C$. To do this, we need to choose $\text{loops}_{x_0}$ which are of interest to our study. All the
loops_{x_0} which do not go, at least, one full turn in the cylinder are homotopic to the point x_0, as shown in figure 1. When we write loops we refer to loops_{x_0}.

Thus, loops that spin around the cylinder, such as the curve α in figure 2, cannot deform continuously to the point and therefore they are the loops of our interest.

Similarly to the case of Π_1(S^1, x_0), we can prove that all paths generated by α are rw-equal to a path loop^n for a n ∈ Z and prove that Π_1(C, x_0) is a group. Furthermore, using the maps toPath and toInteger, we define the bijection between Z and Π_1(C, x_0).

**Theorem 2.** Π_1(C, x_0) ≃ Z.

**Proof.** Proving this is equivalent to finding a bijection between spaces. Consider the map:

\[ \text{toPath}: \mathbb{Z} \rightarrow \Pi_1(C, x_0) \]

\[ n \rightarrow \text{toPath}(n). \]

Defined by:

\[
\begin{align*}
\text{toPath}(n) &= \begin{cases} 
\text{toPath}(0) = \rho \\
\text{toPath}(n) = \text{toPath}(n - 1) \circ \text{loop}^1 & n > 0 \\
\text{toPath}(n) = \text{toPath}(n + 1) \circ \sigma(\text{loop}^1) & n < 0.
\end{cases}
\end{align*}
\]
Now, consider the map:

\[
\text{toInteger} : \Pi_1(C, x_0) \longrightarrow \mathbb{Z} \\
\text{loop}^n \longrightarrow n.
\]

Defined by:

\[
\text{toInteger}(n) = \begin{cases} 
\text{toInteger}(\text{loop}^0 = \rho) = 0 \\
\text{toInteger}(\text{loop}^n) = \text{succ}(\text{toInteger}(\text{loop}^{n-1})) & n > 0 \\
\text{toInteger}(\text{loop}^n) = \text{pred}(\text{toInteger}(\text{loop}^{n+1})) & n < 0
\end{cases}
\]

That way, we have the desired isomorphism.

4.3 Fundamental Group of the Möbius Band - \(\Pi_1(\mathbb{K}^2, x_0)\)

In this case, we will again disregard all loops which are homotopic to the constant \(x_0\). Thus, the loops of our interest are those that spin the surface in a fixed direction, such as the loops denoted by \(\alpha\) in figure 3.

![Fig. 3. \(\alpha \in \Pi_1(\mathbb{K}^2, x_0)\)](image)

**Proposition 3.** \(\Pi_1(\mathbb{K}^2, x_0) \simeq \mathbb{Z}\).

**Proof.** Similarly to the case of \(\Pi_1(S^1, x_0)\), we can prove that all paths generated by \(\alpha\) are \(rw\)-equal to a path \(\text{loop}^n\) for a \(n \in \mathbb{Z}\) and prove that \(\Pi_1(\mathbb{K}^2, x_0)\) is a group. Furthermore, using the maps \(\text{toPath}\) and \(\text{toInteger}\), we have the bijection between \(\mathbb{Z}\) and \(\Pi_1(\mathbb{K}^2, x_0)\). That way, we have the desired isomorphism.

4.4 Fundamental Group of the Torus - \(\Pi_1(\mathbb{T}^2, x_0)\)

Consider \(\mathbb{T}^2\) as the surface known as Torus and the point \(x_0 \in \mathbb{T}^2\). We will prove using computational paths that the fundamental group of the torus is isomorphic to \(\mathbb{Z} \times \mathbb{Z}\).
Before we proceed, we need to look at some instances of loop\(_{x_0}\) in \(T^2\). Consider the figure below:

In figure 4 above, we have an example of a loop\(_{x_0}\) that is not particularly interesting because it continuously deforms to the constant path \(\rho\). Thus, these types of loops will be discarded in our study.

Therefore, we will be interested in working with loops that are not homotopic to the base point \(x_0\), like loops \(\alpha\) and \(\beta\). These loops will be the generators of \(T^2\) as shown in figure 5.

**Definition 11 (vertical loop).** We define as vertical loop the path (loop) that passes through the inner part of \(T^2\) in the vertical direction. In figure 5, this loop is denoted by \(\alpha\).

**Definition 12 (horizontal loop).** We define as horizontal loop the path (loop) that passes the inner part of \(T^2\) in the horizontal direction. In figure 5, this loop is denoted by \(\beta\).

Note that this two loops are not of the type \(\rho\) (homotopic to constant \(x_0\)). Furthermore, they generate \(T^2\). In what follows, we define and denote by: \(\alpha^n = \text{loop}^n_v\) the path composed by \(n\) vertical loops and by \(\beta^m = \text{loop}^m_h\) the path composed by \(m\) horizontal loops.

We now give the formal definition of the torus in homotopy type theory:

**Definition 13.** The torus \(T^2\) is generated by:
(i) Two paths $\alpha$ and $\beta$ such that: $x_0^\alpha = x_0$ and $x_0^\beta = x_0$.

(ii) One path $co$ that establishes $\beta \alpha = \alpha \beta$, i.e., a term $co : \text{Id}(\beta \alpha, \alpha \beta)$.

Given a point $x_0$, we can slice the Torus and represent it as a rectangle whose laterals are the loops $\alpha$ and $\beta$, how show in figure 6.

![Fig. 6. Sliced Torus with oriented paths $\alpha$ and $\beta$](image)

Consider the following path in the figure: $\tau(\beta, \alpha, \sigma(\beta), \sigma(\alpha)) = \beta * \alpha * \beta^{-1} * \alpha^{-1}$:

**Proposition 4.** The aforementioned path is rw-equal to the reflexive path.

**Proof.** Indeed,

$$\beta * \alpha * \beta^{-1} * \alpha^{-1} = \beta * \beta^{-1} * \alpha * \alpha^{-1}$$

$$= \tau(\sigma(\beta), \beta, \sigma(\alpha))$$

$$= \tau(\rho, \rho)$$

$$= trr \rho$$

and thus, $\tau(\beta, \alpha, \sigma(\beta), \sigma(\alpha)) = \beta * \alpha * \beta^{-1} * \alpha^{-1} = \rho$

**Lemma 2.** All paths in $T^2$ are rw-equal to $\beta^n \alpha^m$, with $m, n \in \mathbb{Z}$ and $\beta^0, \alpha^0 = \rho$.

**Proof.** Consider the following cases:

(i) Base case: $\beta^0 \alpha^0 = \rho$.

(ii) $\rho \circ \alpha = \tau(\alpha, \rho)_trr = \alpha = \beta^0 \alpha^1$.

(iii) $\rho \circ \beta = \tau(\beta, \rho)_trr = \beta = \beta^1 \alpha^0$.

(iv) $\rho \circ \alpha^{-1} = \tau(\sigma(\alpha), \rho)_trr = \sigma(\alpha) = \beta^0 \alpha^{-1}$.

(v) $\rho \circ \beta^{-1} = \tau(\sigma(\beta), \rho)_trr = \sigma(\beta) = \beta^{-1} \alpha^0$.

Assuming, by the induction hypothesis, that every path is rw-equal $\beta^n \alpha^m$, we have:
(1) \( \rho \circ \beta^n \alpha^m = \tau(\beta^n \alpha^m, \rho) = \beta^n \alpha^m \).

(2) \( \alpha \circ \beta^n \alpha^m = \alpha \circ \alpha^m \beta^n = \alpha^{m+1} \beta^n = \beta^n \alpha^{m+1} \).

(3) \( \beta \circ \beta^n \alpha^m = \beta^{n+1} \alpha^m = \beta^n \alpha^{m+1} \).

(4) \( \beta^{-1} \circ \beta^n \alpha^m = (\beta^{-1} \circ (\beta \circ \beta^{-1})) \alpha^m = ((\beta^{-1} \circ \beta) \circ \beta^{-1}) \alpha^m = (\rho \circ \beta^{-1}) \alpha^m = \beta^{-1} \alpha^m \).

(4) \( \alpha^{-1} \circ \beta^n \alpha^m = \alpha^{-1} \circ \alpha^m \beta^n = (\alpha^{-1} \circ (\alpha \circ \alpha^{-1})) \beta^n = ((\alpha^{-1} \circ \alpha) \circ \alpha^{-1}) \beta^n = (\rho \circ \alpha^{-1}) \beta^n = \alpha^{-1} \beta^n = \beta^n \alpha^{-1} \).

So all paths in \( T^2 \) are rw-equal to \( \beta^n \alpha^m \).

**Proposition 5.** \( (\Pi_1(T^2)) \) is a group.

**Proof.**

\( (+) \): Sum

\[
\begin{array}{c@{\,}c@{\,}c@{\,}c@{\,}c@{\,}c@{\,}c@{\,}c@{\,}c@{\,}c@{\,}c@{\,}c@{\,}c@{\,}c@{\,}c@{\,}c@{\,}c@{\,}c@{\,}c@{\,}c@{\,}c@{\,}c@{\,}c@{\,}c@{\,}c@{\,}c@{\,}c@{\,}c@{\,}c@{\,}c@{\,}c@{\,}c@{\,}c@{\,}c@{\,}c@{\,}c@{\,}c@{\,}c@{\,}c@{\,}c@{\,}c@{\,}c@{\,}c@{\,}c@{\,}c@{\,}c@{\,}c@{\,}c@{\,}c@{\,}c@{\,}c@{\,}c@{\,}c@{\,}c@{\,}c@{\,}c@{\,}c@{\,}c@{\,}c@{\,}c@{\,}c@{\,}c@{\,}c@{\,}c@{\,}c@{\,}c@{\,}c@{\,}c@{\,}c@{\,}c@{\,}c@{\,}c@{\,}c@{\,}c@{\,}c@{\,}c@{\,}c@{\,}c@{\,}c@{\,}c@{\,}c@{\,}c@{\,}c@{\,}c@{\,}c@{\,}c@{\,}c@{\,}c@{\,}c@{\,}c@{\,}c@{\,}c@{\,}c@{\,}c@{\,}c@{\,}c@{\,}c@{\,}c@{\,}c@{\,}c@{\,}c@{\,}c@{\,}c@{\,}c@{\,}c@{\,}c@{\,}c@{\,}c@{\,}c@{\,}c@{\,}c@{\,}c@{\,}c@{\,}c@{\,}c@{\,}c@{\,}c@{\,}c@{\,}c@{\,}c@{\,}c@{\,}c@{\,}c@{\,}c@{\,}c@{\,}c@{\,}c@{\,}c@{\,}c@{\,}c@{\,}c@{\,}c@{\,}c@{\,}c@{\,}c@{\,}c@{\,}c@{\,}c@{\,}c@{\,}c@{\,}c@{\,}c@{\,}c@{\,}c@{\,}c@{\,}c@{\,}c@{\,}c@{\,}c@{\,}c@{\,}c@{\,}c@{\,}c@{\,}c@{\,}c@{\,}c@{\,}c@{\,}c@{\,}c@{\,}c@{\,}c@{\,}c@{\,}c@{\,}c@{\,}c@{\,}c@{\,}c@{\,}c@{\,}c@{\,}c@{\,}c@{\,}c@{\,}c@{\,}c@{\,}c@{\,}c@{\,}c@{\,}c@{\,}c@{\,}c@{\,}c@{\,}c@{\,}c@{\,}c@{\,}c@{\,}c@{\,}c@{\,}c@{\,}c@{\,}c@{\,}c@{\,}c@{\,}c@{\,}c@{\,}c@{\,}c@{\,}c@{\,}c@{\,}c@{\,}c@{\,}c@{\,}c@{\,}c@{\,}c@{\,}c@{\,}c@{\,}c@{\,}c@{\,}c@{\,}c@{\,}c@{\,}c@{\,}c@{\,}c@{\,}c@{\,}c@{\,}c@{\,}c@{\,}c@{\,}c@{\,}c@{\,}c@{\,}c@{\,}c@{\,}c@{\,}c@{\,}c@{\,}c@{\,}c@{\,}c@{\,}c@{\,}c@{\,}c@{\,}c@{\,}c@{\,}c@{\,}c@{\,}c@{\,}c@{\,}c@{\,}c@{\,}c@{\,}c@{\,}c@{\,}c@{\,}c@{\,}c@{\,}c@{\,}c@{\,}c@{\,}c@{\,}c@{\,}c@{\,}c@{\,}c@{\,}c@{\,}c@{\,}c@{\,}c@{\,}c@{\,}c@{
\[
\begin{align*}
&\quad \frac{x_0 = x_0}{x_0} \quad \frac{x_0 = x_0}{x_0} \\
\quad \frac{x_0 = x_0}{\tau(\sigma(\beta^n)\sigma(\alpha^m), \beta^n \alpha^m)}
\end{align*}
\]

But,

\[
\tau(\sigma(\beta^n)\sigma(\alpha^m), \beta^n \alpha^m) = (\beta^n \alpha^m) \circ (\sigma(\beta^n)\sigma(\alpha^m))
\]

\[
= \beta^n \alpha^m \sigma(\beta^n)\sigma(\alpha^m)
\]

\[
= \beta^n \sigma(\beta^n)\alpha^m \sigma(\alpha^m)
\]

\[
= \tau(x_0).
\]

(ε): Identity

\[
\begin{align*}
&\quad \frac{x_0 = x_0}{x_0} \quad \frac{x_0 = x_0}{\rho x_0} \\
\quad \frac{x_0 = x_0}{\tau(\rho x_0, \beta^n \alpha^m)}
\end{align*}
\]

But,

\[
\tau(\beta^n \alpha^m, \rho x_0) = (\rho x_0) \circ (\beta^n \alpha^m)
\]

\[
= \rho x_0 \beta^n \alpha^m
\]

\[
= \beta^n \alpha^m
\]

and so

\[
\tau(\beta^n \alpha^m, \rho x_0) = \beta^n \alpha^m.
\]

On the other hand, we have:

\[
\begin{align*}
&\quad \frac{x_0 = x_0}{\rho x_0} \quad \frac{x_0 = x_0}{\beta^n \alpha^m} \\
\quad \frac{x_0 = x_0}{\tau(\rho x_0, \beta^n \alpha^m)}
\end{align*}
\]

But,

\[
\tau(\rho x_0, \beta^n \alpha^m) = (\beta^n \alpha^m) \circ (\rho x_0)
\]

\[
= \beta^n \alpha^m \rho x_0
\]

\[
= \beta^n \alpha^m
\]

and so

\[
\tau(\rho x_0, \beta^n \alpha^m) = \beta^n \alpha^m.
\]
(°): Associativity

\[
\begin{align*}
x_0 &= \beta^n \alpha^m x_0 & x_0 &= \beta^i \alpha^j \quad & x_0 &= \beta^r \alpha^s \quad & x_0 &= \tau(\beta^n \alpha^m, \beta^i \alpha^j) \quad & x_0 &= \tau(\tau(\beta^n \alpha^m, \beta^i \alpha^j), \beta^r \alpha^s) \\
\end{align*}
\]

But,

\[
\begin{align*}
\tau \left( \tau \left( \beta^n \alpha^m, \beta^i \alpha^j \right), \beta^r \alpha^s \right) &= \left( \beta^r \alpha^s \right) \circ \tau(\beta^n \alpha^m, \beta^i \alpha^j) \\
&= \left( \beta^r \alpha^s \right) \circ \left( \beta^i \alpha^j \circ \beta^n \alpha^m \right) \\
&= \left( \beta^r \alpha^s \circ \beta^i \alpha^j \beta^n \alpha^m \right) \\
&= \beta^r \alpha^s \beta^i \alpha^j \beta^n \alpha^m.
\end{align*}
\]

On the other hand, we have:

\[
\begin{align*}
x_0 &= x_0 & x_0 &= x_0 \quad & x_0 &= \beta^t \alpha^u \quad & x_0 &= \tau(\beta^t \alpha^u, \beta^r \alpha^s) \\
\end{align*}
\]

But,

\[
\begin{align*}
\tau \left( \beta^n \alpha^m, \tau \left( \beta^i \alpha^j, \beta^r \alpha^s \right) \right) &= \tau(\beta^n \alpha^m, \beta^r \alpha^s) \circ (\beta^n \alpha^m) \\
&= \left( \beta^r \alpha^s \circ \beta^i \alpha^j \right) \circ (\beta^n \alpha^m) \\
&= \left( \beta^r \alpha^s \beta^i \alpha^j \right) \circ (\beta^n \alpha^m) \\
&= \beta^r \alpha^s \beta^i \alpha^j \beta^n \alpha^m.
\end{align*}
\]

Therefore, it follows that \((\Pi_1(T^2, x_0), \circ)\) is a group.

**Theorem 3.** \(\Pi_1(T^2, x_0) \simeq \mathbb{Z} \times \mathbb{Z}\).

**Proof.** To prove this we need to find a bijection between spaces.
Consider the map:

\[
toPath^2 : \mathbb{Z} \times \mathbb{Z} \rightarrow \Pi_1(T^2, x_0) \\
(n, m) \rightarrow toPath^2(n, m).
\]
Defined by:

\[
\text{toPath}^2(n, m) =
\begin{cases}
\text{toPath}^2(0, 0) = \rho \\
\text{toPath}^2(n, 0) = \text{toPath}^2(n - 1, 0) \circ \text{loop}_v^1 & n > 0 \\
\text{toPath}^2(n, 0) = \text{toPath}^2(n + 1, 0) \circ \sigma(\text{loop}_h^1) & n < 0 \\
\text{toPath}^2(n, m) = \text{toPath}^2(n, m - 1) \circ (\text{loop}_h^1) & m > 0 \\
\text{toPath}^2(n, m) = \text{toPath}^2(n, m + 1) \circ \sigma(\text{loop}_h^1) & m < 0
\end{cases}
\]

Now, consider:

\[
\text{toInteger}^2 : \Pi_1(\mathbb{T}^2, x_0) \rightarrow \mathbb{Z} \times \mathbb{Z}
\]

\[
\text{loop}_v^n \circ \text{loop}_h^m \rightarrow (n, m).
\]

Defined by:

\[
\text{toInteger}^2(n, m) =
\begin{cases}
\text{toInteger}^2(\rho) = (0, 0) \\
\text{toInteger}^2(\text{loop}_v^n) = \text{toInteger}^2(\text{loop}_v^{n-1}) + (1, 0) & n > 0 \\
\text{toInteger}^2(\text{loop}_v^n) = \text{toInteger}^2(\text{loop}_v^{n+1}) + (-1, 0) & n < 0 \\
\text{toInteger}^2(\text{loop}_v^n \circ \text{loop}_h^m) = \text{toInteger}^2(\text{loop}_v^n \circ \text{loop}_h^{m-1}) + (0, 1) & m > 0 \\
\text{toInteger}^2(\text{loop}_v^n \circ \text{loop}_h^m) = \text{toInteger}^2(\text{loop}_v^n \circ \text{loop}_h^{m+1}) + (0, -1) & m < 0
\end{cases}
\]

Therefore, we defined two injective maps:

\[
\text{toInteger}^2 : \Pi_1(\mathbb{T}^2, x_0) \rightarrow \mathbb{Z} \times \mathbb{Z}
\]

and

\[
\text{toPath}^2 : \mathbb{Z} \times \mathbb{Z} \rightarrow \Pi_1(\mathbb{T}^2, x_0)
\]

That way, we have that \( \Pi_1(\mathbb{T}^2, x_0) \) is isomorphic to \( \mathbb{Z} \times \mathbb{Z} \)

### 4.5 Fundamental Group of the Real Projective Plane - \( \Pi_1(\mathbb{P}^2, x_0) \)

The real projective plane, denoted by \( \mathbb{P}^2 \), is by definition the set of all straight lines that pass through the origin of space \( \mathbb{R}^3 \). Each of these lines is a point in the projective plane. In \( \mathbb{P}^2 \), the points \( (x, y, z) \) and \( (\tilde{x}, \tilde{y}, \tilde{z}) \) are in \( \mathbb{R}^3 - \{0, 0, 0\} \) and only if, if they are on the same line, that is, there is a constant of proportionality between them.

We can visualize \( \mathbb{P}^2 \) taking the sphere \( S^2 \) of radius 1. at points in the sphere where \( z \neq 0 \) we can denote by \([x, y, z] \) the equivalents points \((x, y, z)\) and \((\tilde{x}, \tilde{y}, \tilde{z})\). Therefore, we can represent the projective plane as the upper hemisphere of the sphere together with the set of all pairs of antipodal points located on the curve \( z = 0 \) in the sphere.

Let’s then map it on the unit disk through the following map \([x, y, z] \rightarrow (x, y, 0)\), as follows in the figure 8:

We denote by \( \alpha \) any loop that connects the identified antipodal points, so we can consider \( \alpha \) as a loop (as follows in the figure 9) and any other loop that connects the identified antipodal points is homotopic to \( \alpha \).
Since we can represent the real projective plane $\mathbb{P}^2$ for a disk $D_1$, we can define $\mathbb{P}^2$ homotopically as follows:

**Definition 14.** The real projective plane $\mathbb{P}^2$ is defined (inductively) by:

(i) The points $Q : \mathbb{P}^2$, such that $Q \in \text{Int}D_1$, where $\text{Int}D_1$ is the set of all points in the interior of the disk.

(ii) The pairs of antipodal points $P, P' : \mathbb{P}^2$, such that $P, P' \in \partial D_1$, where $\partial D_1$ is the set of pairs of antipodal points.

(iii) A path $\alpha : P = P'$.

(iv) A path $\text{cicl}$ that establishes $\alpha \circ \alpha = \rho$, i.e., $\text{cicl} : \text{Id}_{\mathbb{P}^2}(\alpha \circ \alpha, \rho)$.

**Lemma 3.** All paths in $\mathbb{P}^2$ generated by $\rho$ or $\alpha$ are rw-equal to $\rho$ or $\alpha$.

**Proof.** Consider the following base cases, $\rho$ and $\alpha$:
Base case $\rho$:
(i) $\rho \circ \rho = \tau(\rho, \rho) = \rho$.
(ii) $\rho \circ \alpha = \tau(\alpha, \rho) = \alpha$.
(iii) $\alpha \circ \rho = \tau(\rho, \alpha) = \alpha$.

Base case $\alpha$:
(i) $\rho \circ \alpha = \tau(\alpha, \rho) = \alpha$.
(ii) $\alpha \circ \rho = \tau(\rho, \alpha) = \alpha$.
(iii) $\alpha \circ \alpha = \tau(\alpha, \alpha) = \alpha$.

Inductive case: Assuming true for $n$, we have:

If $\text{loop}^n = \rho$, we have two possibilities for $n+1$:
(i) $\text{loop}^{n+1} = \text{loop}^n \circ \alpha = \rho \circ \rho = \tau(\rho, \rho) = \rho$.
(ii) $\text{loop}^{n+1} = \text{loop}^n \circ \alpha = \rho \circ \alpha = \tau(\alpha, \rho) = \alpha$.

If $\text{loop}^n = \alpha$, we have two possibilities for $n+1$:
(i) $\text{loop}^{n+1} = \text{loop}^n \circ \alpha = \alpha \circ \rho = \tau(\rho, \alpha) = \alpha$.
(ii) $\text{loop}^{n+1} = \text{loop}^n \circ \alpha = \alpha \circ \alpha = \tau(\alpha, \alpha) = \alpha$.

Thus, all paths in $P_2$ generated by $\rho$ or $\alpha$ are $rw$-equal to either $\alpha$ or $\rho$.

Since we have $\alpha \circ \alpha = \tau(\alpha, \alpha) = \rho$, the term $\text{cicl}$ gives us one important result:
$\alpha = \sigma(\alpha)$.

**Proposition 6.** ($\Pi_1(P_2), \circ$) is a group.

**Proof.**

$(\pm)$: Sum

\[
\begin{array}{c|c}
\text{P}_\alpha & \text{P}_\alpha \\
\hline
\text{P} = \text{P} & \text{P} = \text{P} \\
\text{P} = \tau(\alpha, \alpha) & \text{P} = \text{P} \\
\end{array}
\]

But,
$\alpha \circ \alpha = \tau(\alpha, \alpha) \in \Pi_1(P_2)$.

$(\sigma)$: Inverse

\[
\begin{array}{c|c}
\text{P}_\alpha & \text{P}_\alpha \\
\hline
\text{P} = \text{P} & \text{P} = \text{P} \\
\text{P} = \tau(\alpha, \alpha) & \text{P} = \text{P} \\
\end{array}
\]

But,
$\sigma(\alpha) \circ \alpha = \tau(\alpha, \alpha) \in \Pi_1(P_2)$.

On the other hand, we have:
\[ P = \sigma(\alpha) \quad P = P \quad P = \sigma(\alpha) \]

But,

\[ \alpha \circ \sigma(\alpha) = \tau(\sigma(\alpha), \alpha) = \rho \in \Pi_1(\mathbb{P}^2). \]

(\epsilon): Identity

\[ P = P \quad P = P \quad P = \rho \]

But,

\[ P = P \quad P = P \quad P = \tau(\rho, \alpha) \]

But,

\[ \rho \circ \alpha = \tau(\alpha, \rho) = \alpha \in \Pi_1(\mathbb{P}^2). \]

On the other hand, we have:

\[ P = P \quad P = \rho \quad P = \alpha \]

But,

\[ P = P \quad P = \tau(\rho, \alpha) \]

But,

\[ \alpha \circ \rho = \tau(\rho, \alpha) \]

(\circ): Associativity

\[ P = P \quad P = P \quad P = P \]

But,

\[ P = P \quad P = P \quad P = P \]

On the other hand, we have:
But,

\[
\tau(\alpha, \tau(\alpha, \alpha)) = \tau(\alpha, \alpha) \circ \alpha
\]

\[
= \rho \circ \alpha
\]

\[
= \tau(\alpha, \rho)
\]

\[
= \tau(\alpha, \rho)
\]

How \( \tau(\alpha, \alpha, \alpha) = \tau(\alpha, \tau(\alpha, \alpha)) \), follow that associativity is valid and therefore \((\Pi_1(\mathbb{P}^2), \circ)\) is a group generated by \(\rho\) and \(\alpha\).

**Theorem 4.** \(\Pi_1(\mathbb{P}^2) \simeq \mathbb{Z}_2\).

**Proof.** As before, proving this is equivalent to finding a bijection between spaces.

First, consider the map \(\text{toPath}_{\mathbb{Z}_2} : \mathbb{Z}_2 \rightarrow \Pi_1(\mathbb{P}^2)\) defined by:

\[
\text{toPath}_{\mathbb{Z}_2} = \begin{cases} 
\text{toPath}(0) = \rho \\
\text{toPath}(1) = \alpha = \text{loop}^1
\end{cases}
\]

Now, consider the map \(\text{toInteger} : \Pi_1(\mathbb{P}^2) \rightarrow \mathbb{Z}_2\) defined by:

\[
\text{toInteger} = \begin{cases} 
\text{toInteger}(\text{loop}^0 = \rho) = 0 \\
\text{toInteger}(\text{loop}^1 = \alpha) = 1
\end{cases}
\]

Thus, the isomorphism holds.

**5 Conclusion**

In this work, our main objective has been the calculation of the fundamental groups of many surfaces using homotopy type theory. We have seen that it is possible to do these calculations by means of an entity known as computational paths. The main advantage of this approach is that we have avoided the use of more complex techniques, code-encode-decode one. As a consequence, our calculations proved to be straightforward and simple. Using computational paths as our main tool, we have calculated the fundamental group of the circle, cylinder, Möbius band, torus and projective plane. Therefore, we have shown that it is possible to use the theory of computational paths to obtain central results of algebraic topology and homotopy type theory.
References

1. Steve Awodey. Type theory and homotopy. In P. Dybjer, Sten Lindström, Erik Palmgren, and G. Sundholm, editors, Epistemology versus Ontology, volume 27 of Logic, Epistemology, and the Unity of Science, pages 183–201. Springer Netherlands, 2012.

2. A. G. de Oliveira. Proof transformations for labelled natural deduction via term rewriting. 1995. Master’s thesis, Depto. de Informática, Universidade Federal de Pernambuco, Recife, Brazil, April 1995.

3. A. G. de Oliveira and R. J. G. B. de Queiroz. A normalization procedure for the equational fragment of labelled natural deduction. Logic Journal of IGPL, 7(2):173–215, 1999.

4. R. J. G. B. de Queiroz and A. G. de Oliveira. Term rewriting systems with labelled deductive systems. In Proceedings of Brazilian Symposium on Artificial Intelligence (SBIA’94), pages 59–72, 1994.

5. R. J. G. B. de Queiroz and A. G. de Oliveira. Natural deduction for equality: The missing entity. In Luiz Carlos Pereira, Edward Haesler, and Valeria de Paiva, editors, Advances in Natural Deduction - A Celebration of Dag Prawitz’s Work, pages 63–91. Springer, 2014.

6. R. J. G. B. de Queiroz, A. G. de Oliveira, and D. M. Gabbay. The Functional Interpretation of Logical Deduction. World Scientific, 2011.

7. R. J. G. B. de Queiroz, A. G. de Oliveira, and A. F. Ramos. Propositional equality, identity types, and direct computational paths. South American Journal of Logic, 2(2):245–296, 2016. Special Issue A Festschrift for Francisco Miraglia, M. E. Coniglio, H. L. Mariano and V. C. Lopes (Guest Editors).

8. R. J. G. B. de Queiroz and D. M. Gabbay. Equality in labelled deductive systems and the functional interpretation of propositional equality. In Proceedings of the 9th Amsterdam Colloquium, pages 547–565. ILLC/Department of Philosophy, University of Amsterdam, 1994.

9. J. Roger Hindley and Jonathan P. Seldin. Lambda-calculus and combinators: an introduction. Cambridge University Press, 2008.

10. Martin Hofmann and Thomas Streicher. The groupoid model refutes uniqueness of identity proofs. In Logic in Computer Science, 1994, LICS’94. Proceedings., Symposium on, pages 208–212. IEEE, 1994.

11. Philippe Le Chenadec. On the logic of unification. Journal of Symbolic computation, 8(1):141–199, 1989.

12. Arthur F. Ramos, Ruy J. G. B. De Queiroz, and Anjolina G. De Oliveira. On the identity type as the type of computational paths. Logic Journal of the IGPL, 25(4):562–584, 2017.

13. The Univalent Foundations Program. Homotopy Type Theory: Univalent Foundations of Mathematics. https://homotopytypetheory.org/book Institute for Advanced Study, 2013.

14. V. Voevodsky. Univalent foundations and set theory, 2014. Univalent Foundations and Set Theory, Lecture at IAS, Princeton, New Jersey, Mar 2014.
A Subterm Substitution

In Equational Logic, the sub-term substitution is given by the following inference rule [4]:

\[
\frac{s = t}{s\theta = t\theta}
\]

One problem is that such rule does not respect the sub-formula property. To deal with that, [11] proposes two inference rules:

\[
\frac{M = N \quad C[N] = O}{C[M] = O} \quad IL \quad \frac{M = C[N] \quad N = O}{M = C[O]} \quad IR
\]

where M, N and O are terms.

As proposed in [7], we can define similar rules using computational paths, as follows:

\[
\frac{x = r \quad C[y] : A \quad y = s \quad u : A'}{x = _{\text{sub}_k(r,s)} \quad C[u] : A} \quad \frac{x = r \quad w : A' \quad C[w] = s \quad u : A}{C[x] = _{\text{sub}_k(r,s)} \quad u : A}
\]

where C is the context in which the sub-term detached by ']' appears and A' could be a sub-domain of A, equal to A or disjoint to A.

In the rule above, C[u] should be understood as the result of replacing every occurrence of y by u in C.
B List of Rewrite Rules

We present all rewrite rules of $LND_{EQ-TRS}$. They are as follows (All have been taken from [7]):

1. $\sigma(\rho) \triangleright_{sr} \rho$
2. $\sigma(\sigma(r)) \triangleright_{ss} r$
3. $\tau(C[r], C[\sigma(r)]) \triangleright_{tr} C[\rho]$
4. $\tau(C[\sigma(r)], C[r]) \triangleright_{str} C[\rho]$
5. $\tau(C[r], C[\rho]) \triangleright_{tr} C[r]$
6. $\tau(C[\rho], C[r]) \triangleright_{ut} C[r]$
7. $\text{sub}_2(C[r], C[\rho]) \triangleright_{slr} C[r]$
8. $\text{sub}_2(C[\rho], C[r]) \triangleright_{arr} C[r]$
9. $\text{sub}_2(\text{sub}_2(s, C[r]), C[\sigma(r)]) \triangleright_{sls} s$
10. $\text{sub}_2(\text{sub}_2(s, C[\sigma(r)]), C[r]) \triangleright_{slss} s$
11. $\text{sub}_2(C[s], \text{sub}_2(C[\sigma(s)], r)) \triangleright_{srs} r$
12. $\text{sub}_2(C[\sigma(s)], \text{sub}_2(C[s], r)) \triangleright_{srr} r$
13. $\mu_1(\xi_1(r)) \triangleright_{mx21} r$
14. $\mu_1(\xi_1, (r, s)) \triangleright_{mx22} r$
15. $\mu_2(\xi_1(s), (r, s)) \triangleright_{mx21} s$
16. $\mu_2(\xi_2(s)) \triangleright_{mx22} s$
17. $\mu_1(\xi_1(r), s, u) \triangleright_{mx3} s$
18. $\mu_4(\xi_3(s), r, u) \triangleright_{mx3u} u$
19. $\mu_4(\xi_4(r)) \triangleright_{mxz} r$
20. $\mu_5(\xi_5(r), s) \triangleright_{mxr} s$
21. $\xi(\mu_1(r), \mu_2(r)) \triangleright_{mx} r$
22. $\mu_4(\xi_5, \xi_5(s)) \triangleright_{mx5} t$
23. $\xi(\nu(r)) \triangleright_{xmr} r$
24. $\mu(s, \xi_2(s)) \triangleright_{mx1} s$
25. $\sigma(\tau(r, s)) \triangleright_{sxs} \tau(\sigma(s), \sigma(r))$
26. $\sigma(\text{sub}_2(r, s)) \triangleright_{sbs} \text{sub}_2(\sigma(s), \sigma(r))$
27. $\sigma(\text{sub}_2(r, s)) \triangleright_{sbr} \text{sub}_2(\sigma(s), \sigma(r))$
28. $\sigma(\xi(r)) \triangleright_{xs} \xi(\sigma(r))$
29. $\sigma(\xi(s, r)) \triangleright_{sxs} \xi(\sigma(s), \sigma(r))$
30. $\sigma(\mu(r)) \triangleright_{sm} \mu(\sigma(r))$
31. $\sigma(\mu(s, r)) \triangleright_{sms} \mu(\sigma(s), \sigma(r))$
32. $\sigma(\mu(r, u, v)) \triangleright_{sms} \mu(\sigma(r), \sigma(u), \sigma(v))$
33. $\tau(r, \text{sub}_2(s, r)) \triangleright_{tsb} \text{sub}_2(r, s)$
34. $\tau(r, \text{sub}_2(s, r)) \triangleright_{tsbr} \text{sub}_2(r, s)$
35. $\tau(\text{sub}_2(r, s), t) \triangleright_{tsbr} \tau(r, \text{sub}_2(s, t))$
36. $\tau(\text{sub}_2(s, t), u) \triangleright_{tsbr} \text{sub}_2(s, t, u)$
37. $\tau(\tau(r, t, s), u) \triangleright_{ut} \tau(t, \tau(r, s))$
38. $\tau(C[u], \tau(C[\sigma(u)], v)) \triangleright_{ts} v$
39. $\tau(C[\sigma(u)], \tau(C[u], v)) \triangleright_{tul} u$