The Directed Van Kampen Theorem in Lean

Henning Basold
Leiden Institute of Advanced Computer Science, Leiden University, The Netherlands

Peter Bruin
Mathematisch Instituut, Leiden University, The Netherlands

Dominique Lawson
Student, Leiden University, The Netherlands

Abstract

Directed topology augments the concept of a topological space with a notion of directed paths. This leads to a category of directed spaces, in which the morphisms are continuous maps respecting directed paths. Directed topology thereby enables an accurate representation of computation paths in concurrent systems that usually cannot be reversed.

Even though ideas from algebraic topology have analogues in directed topology, the directedness drastically changes how spaces can be characterised. For instance, while an important homotopy invariant of a topological space is its fundamental groupoid, for directed spaces this has to be replaced by the fundamental category because directed paths are not necessarily reversible.

In this paper, we present a Lean 4 formalisation of directed spaces and of a Van Kampen theorem for them, which allows the fundamental category of a directed space to be computed in terms of the fundamental categories of subspaces. Part of this formalisation is also a significant theory of directed spaces, directed homotopy theory and path coverings, which can serve as basis for future formalisations of directed topology. The formalisation in Lean can also be used in computer-assisted reasoning about the behaviour of concurrent systems that have been represented as directed spaces.

2012 ACM Subject Classification Theory of computation → Type theory; Mathematics of computing → Algebraic topology

Keywords and phrases Lean, Directed Topology, Van Kampen Theorem, Directed Homotopy Theory, Formalised Mathematics

Digital Object Identifier 10.4230/LIPIcs.ITP.2024.8

Related Version *Extended pre-print using Lean 3*: `https://arxiv.org/abs/2312.06506` [1]

Supplementary Material
Software (Lean Code): `https://github.com/Dominique-Lawson/Directed-Topology-Lean-4` [16]
archived at `swh:1:dir:479a73373a2bf508149f7d1b889b42304fe78a9e`

Funding *Peter Bruin*: Partially supported by the Dutch Research Council (NWO/OCW), as part of the Quantum Software Consortium programme (project number 024.003.037).

1 Introduction

Any topological space is equipped with a set of *paths* (continuous maps from the unit interval into the space), which is closed under composition and reversion. However, one often needs to distinguish a subset of paths following a particular *direction*, for example to model non-reversible processes. One motivation stems from models of true concurrency [9], where executions are modelled as non-reversible paths in a space. For instance, two programs A and B can be executed *sequentially* in two ways: either we first run A and then B, or vice versa, see a) of Figure 1. This choice between two sequential linearisations corresponds to semantics of labelled transition systems, but it neglects potential parallel execution. To see this, suppose that A and B have no dependency or interaction and can be run in parallel. This situation can be modelled by admitting any path in the square from the bottom left

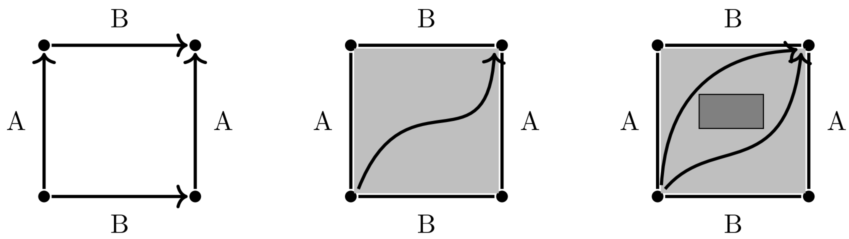

Figure 1 Possible execution paths of two programs A and B under three conditions: a) sequential (left), b) simultaneous (middle) and c) simultaneous with obstacles (right).

to the top right as a valid execution, with the intuition that going along the path tracks how far each of the processes has been run, see b) of Figure 1. The caveat is that processes can, in general, not be reversed and therefore the path may only ever go up and to the right, following the directions of the arrows. Suppose that there is a dependency between the processes, for instance they need to write to the same memory location. To prevent race conditions, we could rule out execution paths in which the processes access that memory location at the same time. This can be modelled by the space in Figure 1 c), where the darker rectangle is an obstacle that paths have to bypass. The two displayed paths in that space represent different memory access patterns: the lower path means that process B first gets access to the memory location, while the upper means that A first gets access. These two paths are essentially different because the observable behaviour of the system differs and because we cannot change the access pattern during execution. In contrast, the different paths in Figure 1 b) model executions that differ only in the relative execution speeds of A and B but are otherwise equivalent. By giving one process more execution time, we can always deform one path into another in this space. Finally, the space in Figure 1 a) has exactly two paths from the bottom left to top right, neither of which can be deformed to the other due to the absence of parallelism. This tells us that the spaces in Figure 1 all model different systems. The question is then how our intuition about relating execution paths can be made precise and how we can reason about these relations.

Directed topology and directed homotopy theory [8, 13] make the above intuition precise and enable the analysis of concurrent systems with the tools of algebraic topology. There are various ways to enforce direction in topological spaces, such as higher-dimensional automata [22, 20], spaces with a global order [10], spaces with local orders [7], streams [15], and various others [6, 11]. We will focus here on the notion of d-space [12], which represents a directed space as a topological space with a distinguished set of directed paths. It then turns out that reasoning about concurrent systems becomes reasoning about the homotopy type of d-spaces, that is, the relation between directed paths in a d-space.

An important strategy in building and analysing large systems is to prove local properties of subsystems and deduce properties of the composed system from these local properties. In algebraic topology, an important result allowing us to combine knowledge of the homotopy type of subspaces into knowledge about the whole space is the Van Kampen theorem [3]. This result expresses the fundamental group of a topological space as a pushout of fundamental groups of suitably chosen subspaces. It has been extended to d-spaces by Grandis [12]. To make the latter result applicable in larger systems, we set out in this paper to formalise the Van Kampen theorem for d-spaces in the proof assistant Lean [5], thereby enabling compositional reasoning about homotopy types of d-spaces and of concurrent systems modelled as d-spaces.

1.1 Contributions

Our main contribution is the formalisation of definitions and theorems relating to directed topology, in particular the Van Kampen Theorem. For this formalisation we used Lean 4.6.0-rc1 and we built upon the work already present in mathlib [18]. All of the formalisation can be found in the accompanying Git repository [16]. It consists of 5.6k lines of code distributed over 30 files. Throughout the article, excerpts from the formalisation are given to show the implementations of definitions and lemmas.

As directed topology has not been formalised before, our formalisation is a natural starting point for the development of a formalised directed topology. Our work has not yet been integrated into mathlib, but we plan on doing so in the near future.

1.2 Related work

There are currently no other formalisations of (parts of) directed topology. The undirected Van Kampen theorem has been formalised in Agda by Favonia and Shulman [14], and in Lean 2 by Van Doorn et al. [21]. In both cases, the formalisation uses synthetic homotopy theory in the form of univalent homotopy type theory, while our formalisation is analytic, that is, we define homotopy as concept derived from (directed) topological spaces. At the moment, mathlib does not contain a proof of the undirected Van Kampen Theorem.

1.3 Overview

In Section 2, we define the notion of directed spaces and directed maps and give a few examples. In Section 3, the definitions and some properties of directed homotopies and directed path homotopies are given. We use those to define relations on the set of directed paths between two points. In Section 4, the equivalence classes of paths under these relations are used to define the fundamental category. The Van Kampen Theorem is stated in Section 5 and we describe the connection between its proof and its formalisation in a precise manner. Finally, in Section 6 we reflect on the ideas presented in this article.

2 Directed Spaces

In this section, we will look at the basic structure of a directed space. With directed maps as morphisms, the category of directed spaces **dTop** is obtained.

2.1 Directed Spaces

A directed space is a topological space with a distinguished set of paths, whose elements are called directed paths. This set must contain all constant paths and must be closed under concatenation and monotone subparametrisation. We denote the concatenation of two paths by $\odot$.

▶ **Definition 1** (Directed space). *A directed space is a topological space X together with a subset P_X of the set of paths in X, satisfying the following three properties:*

1. *For any point $x \in X$, we have $0_x \in P_X$, where 0_x is the constant path in x.*
2. *For any two paths $\gamma_1, \gamma_2 \in P_X$ with $\gamma_1(1) = \gamma_2(0)$, we have $\gamma_1 \odot \gamma_2 \in P_X$.*
3. *For any path $\gamma \in P_X$ and any continuous, monotone map $\varphi : [0,1] \to [0,1]$, we have $\gamma \circ \varphi \in P_X$.*

The elements of P_X are called directed paths or dipaths.

We will first consider some examples of directed spaces.

▶ **Example 2** (Directed unit interval)**.** We can give the unit interval a rightward direction. This is done by taking $P_{[0,1]} = \{\varphi : [0,1] \to [0,1] \mid \varphi \text{ continuous and monotone}\}$. We will denote this directed space by I. More generally, every (pre)ordered space can be given a set of directed paths this way.

▶ **Example 3** (Product of directed spaces)**.** If (X, P_X) and (Y, P_Y) are two directed spaces, then the space $X \times Y$ with the product topology can be made into a directed space by letting $P_{X\times Y} = \{t \mapsto (\gamma_1(t), \gamma_2(t)) \mid \gamma_1 \in P_X \text{ and } \gamma_2 \in P_Y\}$. As we will see in Section 2.2, with this set of directed paths both projection maps will be examples of directed maps and $(X \times Y, P_{X\times Y})$ becomes a product in a categorical sense.

▶ **Example 4** (Induced directed space)**.** Let X be a topological space and (Y, P_Y) a directed space. Let a continuous map $f : X \to Y$ be given. If $\gamma : [0,1] \to X$ is a path in X, then $f \circ \gamma : [0,1] \to Y$ is a path in Y. We can make X into a directed space by taking $P_X = \{\gamma \in C([0,1], X) \mid f \circ \gamma \in P_Y\}$. In the special case that X is a subspace of Y and f is the inclusion map, we find that every subspace of a directed space can be given a natural directedness.

We formalised the notion of a directed space by extending the `TopologicalSpace` class. In our formalisation, we do not explicitly use a set containing paths. Rather, being a directed path is a property of a path itself, analogously to how being open is a property of a set in the `TopologicalSpace` class. Paths in topological spaces have been implemented in mathlib in the file `Topology/Connected/PathConnected.lean`. A path has type `Path x y`, where its starting point is `x` and its endpoint is `y`. The definition of a directed space can be found in `directed_space.lean`.

```
class DirectedSpace (α : Type u) extends TopologicalSpace α where
  IsDipath : ∀ {x y : α}, Path x y → Prop
  isDipath_constant : ∀ (x : α), IsDipath (Path.refl x)
  isDipath_concat : ∀ {x y z : α} {γ₁ : Path x y} {γ₂ : Path y z},
      IsDipath γ₁ → IsDipath γ₂ → IsDipath (Path.trans γ₁ γ₂)
  isDipath_reparam : ∀ {x y : α} {γ : Path x y} {t₀ t₁ : I}
      {f : Path t₀ t₁}, Monotone f → IsDipath γ →
          IsDipath (f.map (γ.continuous_toFun))
```

The term `IsDipath` determines whether a path is directed. The three other terms are exactly the three properties of a directed space. `Path.refl x` is the constant path in a point `x` and `Path.trans` is used for the concatenation of paths. The mathlib library only has support for reparametrisations of paths (meaning that the endpoints must remain the same), but we want to also allow strict subparametrisations. We do this by interpreting the subparametrisation f as a monotone path in $[0,1]$. Then the path $\gamma \circ f$ can be obtained using `Path.map`, where we interpret γ as a continuous map.

In `constructions.lean`, various instances of directed spaces can be found: topological spaces with a preorder (Example 2), products of directed spaces (Example 3) and induced directedness (Example 4).

For brevity, we introduce a notation for the set of all directed paths between x and y.

▶ **Definition 5.** *If X is a directed space and $x, y \in X$ points, we use the shorthand notation $P_X(x, y)$ for the set $\{\gamma \in P_X \mid \gamma(0) = x \text{ and } \gamma(1) = y\}$.*

This definition can also be seen as a type for our formalisation. That is exactly how to interpret the structure `Dipath`, found in `dipath.lean`:

```
variable {X : Type u} [DirectedSpace X]
structure Dipath (x y : X) extends Path x y :=
  (dipath_toPath : IsDipath toPath)
```

It extends the `path` structure and depends on two points `x` and `y` in a directed space `X`. The term `dipath_toPath` has type `IsDipath toPath`. That means that the underlying path it extends must be a directed path. Due to the axioms of a directed space, we can define `Dipath.refl` and `Dipath.trans` analogously to their path-counterparts. However, `Path.symm`, the reversal of a path, cannot be converted to a directed variant as it is not guaranteed that the reversal of a directed path is directed.

We introduce a notation for a special kind of subpath of a directed path.

▶ **Definition 6.** *Let X be a directed space and $\gamma \in P_X$ a directed path. Given integers $n > 0$ and $1 \leq i \leq n$, we will define $\gamma_{i,n} \in P_X$ to be the path from $\gamma(\frac{i-1}{n})$ to $\gamma(\frac{i}{n})$ given by $\gamma_{i,n}(t) = \gamma(\frac{i+t-1}{n})$.*

We can now say what it means for a directed path to be covered by a cover of a directed space. This definition will play a big role in proving and formalising the Van Kampen Theorem for directed spaces.

▶ **Definition 7.** *Let X be a directed space, $U \subseteq X$ a subset and $\gamma \in P_X$ a directed path. We say that γ is contained in U if $\operatorname{Im} \gamma \subseteq U$.*

▶ **Definition 8.** *Let X be a directed space and $\mathcal{U}$ a cover of X. Let $\gamma \in P_X$ be a directed path and $n > 0$ an integer. We say that γ is n-covered (by $\mathcal{U}$) if $\gamma_{i,n}$ is contained in some $U_i \in \mathcal{U}$ for each $1 \leq i \leq n$.*

In `path_cover.lean` we formalise this definition of n-covered in the special case that $\mathcal{U}$ consists of two sets X_0 and X_1 using induction:

```
variable {x y : X} (hX : X₀ ∪ X₁ = univ)

def covered (γ : Dipath x y) : Prop :=
  (range γ ⊆ X₀) ∨ (range γ ⊆ X₁)

def covered_partwise (γ : Dipath x y) (n : ℕ) : Prop := match n with
| Nat.zero => covered hX γ
| Nat.succ n =>
  covered hX (FirstPart γ (Fraction.ofPos (Nat.succ_pos n.succ))) ∧
  covered_partwise hX
    (SecondPart γ (Fraction.ofPos (Nat.succ_pos n.succ))) n
```

Here `covered` corresponds with γ being 1-covered: its image is either contained in X_0 or in X_1. We use this definition to inductively define `covered_partwise`. As it is easier to start at zero in Lean, `covered_partwise hX γ n` corresponds with γ being $(n+1)$-covered. In the case that $n = 0$, we have that `covered_partwise` simply agrees with `covered`. Otherwise, we use an induction step to define that `covered_partwise hX γ (Nat.succ n)` holds if the first part $\gamma_{1,n+2}$ is `covered` and the remainder of γ is `covered_partwise hX γ n`. Note the use of $n+2$ instead of $n+1$ due to the offset between the definitions. The remainder of `path_cover.lean` contains lemmas about conditions for being n-covered.

2.2 Directed Maps

As directed spaces are an extension of topological spaces, directed maps will be extensions of continuous maps. They will need to respect the extra directed structure. If a path in the domain space is given, a path in the codomain space can be obtained by composing the continuous map with the path. If the former is directed, so should be the latter.

▶ **Definition 9** (Directed map). *Let X and Y be two directed spaces. A directed map $f : X \to Y$ is a continuous map on the underlying topological spaces that furthermore satisfies: for any $\gamma \in P_X$, we have $f \circ \gamma \in P_Y$.*

By the construction of the product of directed spaces in Example 3, the continuous projection maps on both coordinates are directed: a directed path in the product space is a pair of directed paths and a projection returns the original directed path. Similarly, if a continuous map $f : X \to Y$ is used to induce a direction on X as in Example 4, then f becomes a directed map from X to Y, where X has the induced directedness.

In order to formalise the definition of a directed map in Lean, we define the property `Directed`, which expresses exactly that a continuous map between two directed spaces maps directed paths to directed paths. A directed map is then an extension of the `ContinuousMap` structure with a proof for being `Directed`.

```
variable {α β : Type*} [DirectedSpace α] [DirectedSpace β]
def Directed (f : C(α, β)) : Prop := ∀ {x y : α} (γ : Path x y),
  IsDipath γ → IsDipath (γ.map f.continuous_toFun)

structure DirectedMap extends ContinuousMap α β where
  protected directed_toFun : DirectedMap.Directed toContinuousMap
```

Within Lean, we use the notation $D(\alpha, \beta)$ for the type of directed maps between two spaces α and β. Directed paths are also instances of directed maps, because they map directed paths in I to monotone subparametrisation of themselves. `dipath.lean` contains definitions on how to convert the `Dipath` type to the `DirectedMap` type and the other way around. These are called `toDirectedMap` and `of_directedMap` respectively.

Directed spaces and directed maps form a category, which we will denote by **dTop**.

3 Directed Homotopies

In this section, we will look at directed homotopies and directed path homotopies. These two concepts realise the idea of deformation, while respecting the directedness of a directed space.

3.1 Homotopies

A directed homotopy is the deformation of one directed map into another.

▶ **Definition 10** (Directed homotopy). *Let X and Y be two directed spaces. A homotopy between two directed maps $f, g : X \to Y$ is a directed map $H : I \times X \to Y$ such that for all $x \in X$ we have $H(0, x) = f(x)$ and $H(1, x) = g(x)$, where the product $I \times X$ is taken between directed spaces, see Example 3.*

We say that H is a directed homotopy from f to g. This order matters, as unlike in the undirected case a directed homotopy cannot generally be reversed. In our formalisation, we adhere to the method used in defining homotopies between continuous maps in mathlib, which can be found in `Topology/Homotopy/Basic.lean`. In an analogous manner, the structure extends the `DirectedMap (I x X) Y` structure and has two extra properties.

```
structure Dihomotopy (f₀ f₁ : D(X, Y)) extends D((I × X), Y) :=
  (map_zero_left : ∀ x, toFun (0, x) = f₀.toFun x)
  (map_one_left : ∀ x, toFun (1, x) = f₁.toFun x)
```

As a directed map is always a continuous map on the underlying topological spaces, we can convert a `Dihomotopy` to a `Homotopy`. Conversely, if we are given a `Homotopy` and we know that it is directed, we can obtain a `Dihomotopy`.

If $f : X \to Y$ is a directed map, there is an identity homotopy H from f to f, given by $H(t, x) = f(x)$. Also, if G is a directed homotopy from f to g and H a directed homotopy from g to h, we obtain a directed homotopy $G \otimes H$ from f to h given by

$$(G \otimes H)(t, x) = \begin{cases} G(2t, x), & t \leq \frac{1}{2}, \\ H(2t - 1, x), & \frac{1}{2} < t. \end{cases}$$

These constructions are called `refl` and `trans` in `directed_homotopy.lean`. In both cases we coerce a `Homotopy` to a `Dihomotopy`, by supplying proofs that the obtained homotopies are directed. Here we use the existing proofs in mathlib that the constructed maps are indeed homotopies, i.e. are continuous and satisfy the two mapping properties.

3.2 Path Homotopies

▶ **Definition 11** (Directed path homotopy). *Let X be a directed space and $x, y \in X$ two points. A directed path homotopy between two directed paths $\gamma_1, \gamma_2 \in P_X(x, y)$ is a directed homotopy $H : I \times I \to X$ from γ_1 to γ_2 such that additionally for all $t \in [0, 1]$ we have $H(t, 0) = x$ and $H(t, 1) = y$.*

In other words, a path homotopy is a homotopy between two paths that keeps both endpoints fixed. Again we say that H is a directed path homotopy from γ_1 to γ_2. Between two paths γ_1 and γ_2 in I with the same endpoints exists a path homotopy under the condition that $\gamma_1(t) \leq \gamma_2(t)$ for all $t \in I$ as the following example shows.

▶ **Example 12.** Let $t_0, t_1 \in I$ be two points and $\gamma_1, \gamma_2 \in P_I(t_0, t_1)$. If $\gamma_1(t) \leq \gamma_2(t)$ for all $t \in I$, then there is a directed path homotopy H from γ_1 to γ_2 given by $H(t, s) = (1-t)\cdot\gamma_1(s)+t\cdot\gamma_2(s)$. It is continuous by continuity of paths, multiplication and addition. It can be shown that $H(a_0, b_0) \leq H(a_1, b_1)$ if $a_0 \leq a_1$ and $b_0 \leq b_1$. From this, it follows that H is directed, because a directed path in $I \times I$ is exactly a pair of monotone maps $I \to I$ by definition.

Note that H interpolates two paths γ_1 and γ_2. The formalised proof of it being a directed map can be found in the file `interpolate.lean`.

Let $x, y, z \in X$ be three points, $\beta_1, \gamma_1 \in P_X(x, y)$ and $\beta_2, \gamma_2 \in P_X(y, z)$. If there are two directed path homotopies G from β_1 to γ_1 and H from β_2 to γ_2, we can construct a directed path homotopy $G \odot H$ from $\beta_1 \odot \beta_2$ to $\gamma_1 \odot \gamma_2$ given by

$$(G \odot H)(t, s) = \begin{cases} G(t, 2s), & s \leq \frac{1}{2}, \\ H(t, 2s - 1), & \frac{1}{2} < s. \end{cases}$$

Let $x, y \in X$ be two points and $\gamma_1, \gamma_2 \in P_X(x, y)$. If there exists a path homotopy from γ_1 to γ_2, we will write $\gamma_1 \rightsquigarrow \gamma_2$. This defines a relation on the set $P_X(x, y)$, but that relation is not guaranteed to be an equivalence relation, as it is generally not symmetric. This is due to the fact that the reversal of a directed path may not be directed. In order get an equivalence relation on the set of directed paths between two points, we will take the symmetric transitive closure of this relation.

▶ **Definition 13.** *Let X be a directed space and $x, y \in X$ two points. We say that two dipaths $\gamma_1, \gamma_2 \in P_X(x, y)$ are equivalent, or $\gamma_1 \simeq \gamma_2$, if there is an integer $n \geq 0$ together with dipaths $\beta_i \in P_X(x, y)$, for each $1 \leq i \leq n$, such that*

$$\gamma_1 \rightsquigarrow \beta_1 \leftsquigarrow \ldots \rightsquigarrow \beta_n \leftsquigarrow \gamma_2.$$

This alternating sequence of arrows is also called a zigzag. As $\gamma_2 \leftsquigarrow \gamma_2$ holds for any path γ_2 by reflexivity, we can always assume that there is an odd number of paths in a zigzag between two paths γ_1 and γ_2. By taking $n = 0$, it follows that $\gamma_1 \simeq \gamma_2$ holds if $\gamma_1 \rightsquigarrow \gamma_2$. More precisely, $\simeq$ is the smallest equivalence relation on $P_X(x, y)$ such that that property holds [17, p. 129]. As $\simeq$ is an equivalence relation, we can talk about equivalence classes of paths, denoted by $[\gamma]$. An important property of these equivalence classes is that they are invariant under directed maps and path reparametrisation.

▶ **Lemma 14.** *Let X, Y be directed spaces and $x, y \in X$. Let $\gamma_1, \gamma_2 \in P_X(x, y)$ and $f : X \to Y$ directed. If $\gamma_1 \simeq \gamma_2$, then $f \circ \gamma_1 \simeq f \circ \gamma_2$.*

Proof. Let $n > 0$ odd and $\beta_i \in P_X(x, y)$ for $1 \leq i \leq n$ such that

$$\gamma_1 \rightsquigarrow \beta_1 \leftsquigarrow \beta_2 \rightsquigarrow \ldots \rightsquigarrow \beta_n \leftsquigarrow \gamma_2.$$

If $H : I \times I \to X$ is a directed path homotopy from γ_1 to β_1, then $f \circ H$ is a directed path homotopy from $f \circ \gamma_1$ to $f \circ \beta_1$. We find that $f \circ \gamma_1 \rightsquigarrow f \circ \beta_1$. Repeating this for all other arrows in the zigzag gives us

$$f \circ \gamma_1 \rightsquigarrow f \circ \beta_1 \leftsquigarrow f \circ \beta_2 \rightsquigarrow \ldots \rightsquigarrow f \circ \beta_n \leftsquigarrow f \circ \gamma_2,$$

We conclude that $f \circ \gamma_1 \simeq f \circ \gamma_2$. ◀

▶ **Lemma 15.** *Let X be a directed space and $x, y \in X$. Let $\gamma \in P_X(x, y)$ and $\varphi, \varphi' : I \to I$ continuous and monotone with $\varphi(0) = \varphi'(0) = 0$ and $\varphi(1) = \varphi'(1) = 1$. Then $\gamma \circ \varphi \simeq \gamma \circ \varphi'$.*

Proof. As γ is a directed map from I to X, it is enough by Lemma 14 to show that $\varphi \simeq \varphi'$. Let $\beta_1 = \varphi \odot 0_1$ and $\beta_2 = 0_0 \odot \varphi'$. Then, by applying Example 12 three times, we obtain the zigzag $\varphi \rightsquigarrow \beta_1 \leftsquigarrow \beta_2 \rightsquigarrow \varphi'$. This shows that $\varphi \simeq \varphi'$, completing the proof. ◀

In the next section, we will construct the fundamental category of a directed space. For that we need the following four additional equalities of equivalence classes.

▶ **Lemma 16.** *Let X be a directed space and $x, y, z, w \in X$. Let $\beta_1, \gamma_1 \in P_X(x, y)$, $\beta_2, \gamma_2 \in P_X(y, z)$ and $\gamma_3 \in P_X(z, w)$ such that $\beta_1 \simeq \gamma_1$ and $\beta_2 \simeq \gamma_2$. Then the following holds:*

1. $\beta_1 \odot \beta_2 \simeq \gamma_1 \odot \gamma_2$
2. $0_x \odot \gamma_1 \simeq \gamma_1$
3. $\gamma_1 \odot 0_y \simeq \gamma_1$
4. $(\gamma_1 \odot \gamma_2) \odot \gamma_3 \simeq \gamma_1 \odot (\gamma_2 \odot \gamma_3)$

Proof. Statements 2, 3 and 4 are direct applications of Lemma 15 as they are all reparametrisations. We will now show statement 1. Let $n, m > 0$ odd and $p_i, q_j \in P_X(x, y)$ for $1 \leq i \leq n$ and $1 \leq j \leq m$ such that

$$\beta_1 \rightsquigarrow p_1 \leftsquigarrow p_2 \rightsquigarrow \ldots \rightsquigarrow p_n \leftsquigarrow \gamma_1 \quad \text{and} \quad \beta_2 \rightsquigarrow q_1 \leftsquigarrow q_2 \rightsquigarrow \ldots \rightsquigarrow q_m \leftsquigarrow \gamma_2.$$

Let G be a directed path homotopy from β_1 to p_1 and H be the identity homotopy from β_2 to β_2. Then $G \odot H$ is a directed path homotopy from $\beta_1 \odot \beta_2$ to $p_1 \odot \beta_2$. Repeating this, we obtain a zigzag

$$\beta_1 \odot \beta_2 \rightsquigarrow p_1 \odot \beta_2 \leftsquigarrow p_2 \odot \beta_2 \rightsquigarrow \ldots \rightsquigarrow p_n \odot \beta_2 \leftsquigarrow \gamma_1 \odot \beta_2,$$

so $\beta_1 \odot \beta_2 \simeq \gamma_1 \odot \beta_2$. Analogously we obtain a zigzag

$$\gamma_1 \odot \beta_2 \rightsquigarrow \gamma_1 \odot q_1 \leftsquigarrow \gamma_1 \odot q_2 \rightsquigarrow \ldots \rightsquigarrow \gamma_1 \odot q_m \leftsquigarrow \gamma_1 \odot \gamma_2.$$

This results in $\gamma_1 \odot \beta_2 \simeq \gamma_1 \odot \gamma_2$ and combining both equivalences gives us $\beta_1 \odot \beta_2 \simeq \gamma_1 \odot \gamma_2$. ◀

The definition of a directed path homotopy and the three lemmas above have all been been formalised in `directed_path_homotopy.lean`. For the path homotopies, we followed the more general approach from `mathlib`, where we first defined directed homotopies that satisfy some property P. Thereafter we defined `DihomotopyRel` as directed homotopies that are fixed on a select subset of points. This is all defined in `directed_homotopy.lean`. A path homotopy is a homotopy that is fixed on both endpoints, that is, on $\{0, 1\} \subseteq I$, so we can define a directed path homotopy as

```
abbrev Dihomotopy (p₀ p₁ : Dipath x y) :=
  DirectedMap.DihomotopyRel p₀.toDirectedMap p₁.toDirectedMap {0, 1}
```

The construction $\odot$ is called `hcomp` and $\otimes$ is called `trans`. If $f, g \in D(I, I)$ are two directed maps with $f(t) \leq g(t)$ for all $t \in I$, the definition `Dihomotopy.reparam` constructs a homotopy from $\gamma \circ f$ to $\gamma \circ g$. This is done by composing γ and the homotopy obtained from Example 12. If H is a homotopy from γ_1 to γ_2 with $\gamma_1, \gamma_2 \in P_X(x, y)$, and $f : X \to Y$ is a directed map, then the homotopy from $f \circ \gamma_1$ to $f \circ \gamma_2$ given by $f \circ H$ is exactly what `Dihomotopy.map` entails.

Now we can formalise the relations $\rightsquigarrow$ and $\simeq$. These are called `PreDihomotopic` and `Dihomotopic` respectively.

```
    def PreDihomotopic : Prop := Nonempty (Dihomotopy p₀ p₁)
    def Dihomotopic : Prop := EqvGen PreDihomotopic p₀ p₁
```

The term `Nonempty` means exactly that there exists some directed homotopy, which corresponds with our definition of $\rightsquigarrow$. `EqvGen` gives the smallest equivalence relation generated by a relation. The lemmas `map`, `reparam` and `hcomp` in the namespace `Dihomotopic` now correspond with Lemma 14, Lemma 15 and the first point of Lemma 16 respectively.

This gives us enough tools to construct the so called fundamental category.

4 The Fundamental Category

Using the properties found in Section 3.2, we can define a category that captures the information of all paths up to directed deformation in a directed space. This is the directed version of the fundamental groupoid.

▶ **Definition 17** (Fundamental Category). *Let X be a directed space. The fundamental category of X, denoted by $\vec{\Pi}(X)$, is the category that consists of:*

- *Objects: points $x \in X$.*
- *Morphisms: $\vec{\Pi}(X)(x, y) = P_X(x, y)/\simeq$.*
- *Composition: $[\gamma_2] \circ [\gamma_1] = [\gamma_1 \odot \gamma_2]$.*
- *Identity: $id_x = [0_x]$.*

▶ Remark 18. The fact that this category is well defined follows from Lemma 16. Due to property 1, composition is well defined. Due to properties 2 and 3, the constant path behaves as an identity and property 4 gives us associativity.

Note that $\vec{\Pi}$ maps objects in **dTop** to objects in **Cat**. It turns out that it can also be defined on morphisms making it into a functor.

▶ **Definition 19.** *Let $f : X \to Y$ be a directed map. We define $\vec{\Pi}(f) : \vec{\Pi}(X) \to \vec{\Pi}(Y)$ as the functor:*

- *On objects: $\vec{\Pi}(f)(x) = f(x)$.*
- *On morphisms: $\vec{\Pi}(f)([\gamma]) = [f \circ \gamma]$.*

It is well behaved on morphisms, because of Lemma 14. It is straightforward to verify that $\vec{\Pi}(f)$ respects composition and identities.

In our formalisation, we follow the construction of the fundamental groupoid in mathlib found in `AlgebraicTopology/FundamentalGroupoid/Basic.lean` closely. Our implementation is found in `fundamental_category.lean`.

```
structure FundamentalCategory (X : Type u) where
  as : X

instance : CategoryTheory.Category (FundamentalCategory X) where
  Hom x y := Dipath.Dihomotopic.Quotient x.as y.as
  id x := ⟦Dipath.refl x.as⟧
  comp {_ _ _} := Dipath.Dihomotopic.Quotient.comp
  id_comp {x _} f := Quotient.inductionOn f fun a =>
    show ⟦(Dipath.refl x.as).trans a⟧ = ⟦a⟧ from
    Quotient.sound (EqvGen.rel _ _ ⟨Dipath.Dihomotopy.refl_trans a⟩)
  comp_id {_ y} f := /- Proof omitted -/
  assoc {_ _ _ _} f g h := /- Proof omitted -/
```

We show that `FundamentalCategory X` is an instance of a category by defining the morphisms (`hom`), identities (`id`) and composition (`comp`). The morphisms between two objects `x` and `y` are given by `Dipath.Dihomotopic.Quotient x y`. This is the quotient of `Dipath x y` under the `Dihomotopic` relation and is defined in `directed_path_homotopy.lean`. The identity on `x` is then the equivalence class (denoted by ⟦ ⟧) of the constant path in `x`. The composition of the equivalence classes of two compatible paths is defined as the equivalence class of the concatenation of the two paths in `Dipath.Dihomotopic.Quotient.comp`.

The proof that this defines a category is given by `id_comp`, `comp_id` and `assoc`. For example, `id_comp` requires us to show that the directed paths `(Dipath.refl x).trans a` and `a` are dihomotopic, corresponding to statement 2 of Lemma 16. The file also contains the definition of the $\vec{\Pi}$-functor from `dTop` to `Cat`. Analogously to the undirected mathlib implementation, we use the notation dπ for this functor.

5 The Van Kampen Theorem

In this section, we will state and prove the Van Kampen Theorem. We follow the proof of Grandis [12] and work out some of the details that were omitted there. In Section 5.2 we show how we have formalised this proof by comparing the proof to the Lean code.

5.1 The Van Kampen Theorem

Before we state and prove the theorem, we will define the notion of being covered for directed homotopies.

▶ **Definition 20.** *Let X be a directed space and $\mathcal{U}$ a cover of X. Let $H : I \times I \to X$ be a directed homotopy and $n, m > 0$ two integers. We say that H is (n, m)-covered (by $\mathcal{U}$) if for all $1 \le i \le n$ and $1 \le j \le m$ the image of $\left[\frac{i-1}{n}, \frac{i}{n}\right] \times \left[\frac{j-1}{m}, \frac{j}{m}\right] \subseteq I \times I$ under H is contained in some $U \in \mathcal{U}$.*

By the Lebesgue Number Lemma [19, p. 179], for any homotopy H and open cover $\mathcal{U}$ of X, there are $n, m > 0$ such that H is (n, m)-covered by $\mathcal{U}$.

▶ **Theorem 21** (Van Kampen Theorem)**.** *Let X be a directed space and X_1 and X_2 two open subspaces such that $X = X_1 \cup X_2$ and let $X_0 = X_1 \cap X_2$. Let $i_k : X_0 \to X_k$ and $j_k : X_k \to X$ be the inclusion maps, $k \in \{1, 2\}$. Then we obtain a pushout square in **Cat**:*

$$\begin{array}{ccc} \vec{\Pi}(X_0) & \xrightarrow{\vec{\Pi}(i_1)} & \vec{\Pi}(X_1) \\ {\scriptstyle \vec{\Pi}(i_2)}\downarrow & & \downarrow{\scriptstyle \vec{\Pi}(j_1)} \\ \vec{\Pi}(X_2) & \xrightarrow[\vec{\Pi}(j_2)]{} & \vec{\Pi}(X) \end{array}$$

Proof. As $j_1 \circ i_1 = j_2 \circ i_2$ and $\vec{\Pi}$ is a functor, the square is commutative. It remains to show it satisfies the universal property of a pushout square. Let $\mathcal{C}$ be any category and $F_1 : \vec{\Pi}(X_1) \to \mathcal{C}$ and $F_2 : \vec{\Pi}(X_2) \to \mathcal{C}$ be two functors such that $F_1 \circ \vec{\Pi}(i_1) = F_2 \circ \vec{\Pi}(i_2)$. We will explicitly construct a functor $F : \vec{\Pi}(X) \to \mathcal{C}$ such that $F \circ \vec{\Pi}(j_1) = F_1$ and $F \circ \vec{\Pi}(j_2) = F_2$. The construction will show that this functor is necessarily unique with this property.

Step 1) The objects of $\vec{\Pi}(X)$ are exactly the points of X. If an object $x \in \vec{\Pi}(X)$ is also contained in $\vec{\Pi}(X_1)$, it holds that $F(x) = F(j_1(x)) = (F \circ \vec{\Pi}(j_1))(x)$. The desired condition $F \circ \vec{\Pi}(j_1) = F_1$ then requires us to define $F(x) = F_1(x)$. A similar argument gives us that if $x \in \vec{\Pi}(X_2)$ then $F(x) = F_2(x)$. As X_1 and X_2 cover X, for all $x \in \vec{\Pi}(X)$ we have

$$F(x) = \begin{cases} F_1(x), & x \in X_1, \\ F_2(x), & x \in X_2. \end{cases}$$

By the assumption that $F_1 \circ \vec{\Pi}(i_1) = F_2 \circ \vec{\Pi}(i_2)$ this is well defined, so we know how F must behave on objects.

Step 2) Let $[\gamma] : x \to y$ be a morphism in $\vec{\Pi}(X)$. Then there is an $n > 0$ such that γ is n-covered by the open cover $\{X_1, X_2\}$, with $\gamma_{i,n}$ contained in X_{k_i}, $k_i \in \{1, 2\}$. One important thing to note is that $\gamma_{i,n}$ can be both seen as a path in X and as a path in X_{k_i} by restricting its codomain. This matters when we talk about $[\gamma_{i,n}]$, as it could be a morphism in $\vec{\Pi}(X)$ and in $\vec{\Pi}(X_{k_i})$. Within this proof will always consider it as a morphism in $\vec{\Pi}(X_{k_i})$ and write $[j_{k_i} \circ \gamma_{i,n}]$ for the morphism in $\vec{\Pi}(X)$. Note that we have $[\gamma] = [j_{k_n} \circ \gamma_{n,n}] \circ \ldots \circ [j_{k_1} \circ \gamma_{1,n}]$ in $\vec{\Pi}(X)$, as γ is equal to $\gamma_{1,n} \odot (\gamma_{2,n} \odot \ldots (\gamma_{n-1,n} \odot \gamma_{n,n}))$ up to reparametrisation. Because we want F to be a functor and thus to respect composition, we find that necessarily

$$\begin{aligned}F[\gamma] &= F([j_{k_n} \circ \gamma_{n,n}] \circ \ldots \circ [j_{k_1} \circ \gamma_{1,n}]) \\ &= F[j_{k_n} \circ \gamma_{n,n}] \circ \ldots \circ F[j_{k_1} \circ \gamma_{1,n}] \\ &= F\left(\vec{\Pi}(j_{k_n})[\gamma_{n,n}]\right) \circ \ldots \circ F\left(\vec{\Pi}(j_{k_1})[\gamma_{1,n}]\right) \\ &= (F \circ \vec{\Pi}(j_{k_n}))[\gamma_{n,n}] \circ \ldots \circ (F \circ \vec{\Pi}(j_{k_1}))[\gamma_{1,n}] \\ &= F_{k_n}[\gamma_{n,n}] \circ \ldots \circ F_{k_1}[\gamma_{1,n}].\end{aligned}$$

As multiple choices were made, we need to make sure that F is well defined this way. We do this by defining a map $F' : P_X \to \mathrm{Mor}(\mathcal{C})$, where $\mathrm{Mor}(\mathcal{C})$ is the collection of all morphisms in $\mathcal{C}$. The map is given by

$$F'(\gamma) = F_{k_n}[\gamma_{n,n}] \circ \ldots \circ F_{k_1}[\gamma_{1,n}],$$

where γ is n-covered with $\gamma_{i,n}$ contained in X_{k_i}. In the next steps, we will first show that this map is well defined. Then we show that F' respects equivalence classes. From this it follows that F is well defined, as it is simply F' descended to equivalence classes.

Step 3) We first need to make sure that F' does not depend on any choices of k_i. In the case that $\gamma_{i,n}$ is contained in both X_1 and X_2, the value of k_i can be either 1 or 2. The condition that $F_1 \circ \vec{\Pi}(i_1) = F_2 \circ \vec{\Pi}(i_2)$ assures us that both options give us the same morphism.

Step 4) The second choice we made is that of n. It is possible that γ is also m-covered for another integer $m > 0$, with $\gamma_{j,m}$ being contained in X_{p_j}. We want to show that

$$F_{k_n}[\gamma_{n,n}] \circ \ldots \circ F_{k_1}[\gamma_{1,n}] = F_{p_m}[\gamma_{m,m}] \circ \ldots \circ F_{p_1}[\gamma_{1,m}].$$

If we refine the partition of γ in n pieces into a partition of mn pieces, that partition will surely also be partwise covered. Let $l_i \in \{1, 2\}$ for all $1 \leq i \leq mn$ such that $\gamma_{i,mn}$ is contained in X_{l_i}. We now claim that for all $1 \leq i \leq n$ it holds that $F_{k_i}[\gamma_{i,n}] = F_{l_{mi}}[\gamma_{mi,mn}] \circ \ldots \circ F_{l_{m(i-1)+1}}[\gamma_{m(i-1)+1,mn}]$. As $\gamma_{m(i-1)+j,mn}$ with $1 \leq j \leq m$ is a subparametrisation of $\gamma_{i,n}$, we may assume that $l_{m(i-1)+j} = k_i$. This is because F_1 and F_2 agree on $X_1 \cap X_2$. As F_{k_i} is a functor, the claim now follows because functors respect composition and because $\gamma_{i,n}$ is exactly the concatenation of all the smaller paths up to reparametrisation. By a similar claim for $F_{p_j}[\gamma_{j,m}]$ we find:

$$\begin{aligned}F_{k_n}[\gamma_{n,n}] \circ \ldots \circ F_{k_1}[\gamma_{1,n}] &= F_{l_{mn}}[\gamma_{mn,mn}] \circ \ldots \circ F_{l_1}[\gamma_{1,mn}] \\ &= F_{p_m}[\gamma_{m,m}] \circ \ldots \circ F_{p_1}[\gamma_{1,m}].\end{aligned}$$

We conclude that the definition is independent of the value of n. This makes F' well defined.

Step 5) Before we verify that F' is independent of the choice of representative γ, we will first show that F' satisfies two properties:

$$\forall x \in \vec{\Pi}(X) : F'(0_x) = \mathrm{id}_{F(x)}. \tag{1}$$

$$\forall \gamma \in P_X(x, y), \delta \in P_X(y, z) : F'(\gamma \odot \delta) = F'(\delta) \circ F'(\gamma). \tag{2}$$

Let $x \in \vec{\Pi}(X)$ be given. If $x \in X_1$, then 0_x is already contained in X_1 and so by definition of F' we find $F'(0_x) = F_1[0_x] = \mathrm{id}_{F_1(x)} = \mathrm{id}_{F(x)}$. Otherwise it holds that $x \in X_2$, so $F'(0_x) = F_2[0_x] = \mathrm{id}_{F_2(x)} = \mathrm{id}_{F(x)}$. This proves Equation (1).

Let $\gamma \in P_X(x, y)$ and $\delta \in P_X(y, z)$ be two paths in X. We can then find an n such that both γ and δ are n-covered, with $\gamma_{i,n}$ contained in X_{k_i} and $\delta_{i,n}$ contained in X_{p_i}. Then $\gamma \odot \delta$ is $2n$-covered as it holds that

$$(\gamma \odot \delta)_{i,2n} = \begin{cases} \gamma_{i,n}, & i \leq n, \\ \delta_{i-n,n}, & i > n. \end{cases}$$

We find:

$$\begin{aligned} &F'(\delta \odot \gamma) = \\ &F_{p_n}[(\delta \odot \gamma)_{2n,2n}] \circ \ldots \circ F_{p_1}[(\delta \odot \gamma)_{n+1,2n}] \circ F_{k_n}[(\delta \odot \gamma)_{n,2n}] \circ \ldots \circ F_{k_1}[(\delta \odot \gamma)_{1,2n}] = \\ &(F_{p_n}[\delta_{n,n}] \circ \ldots \circ F_{p_1}[\delta_{1,n}]) \circ (F_{k_n}[\gamma_{n,n}] \circ \ldots \circ F_{k_1}[\gamma_{1,n}]) = F'(\delta) \circ F'(\gamma). \end{aligned}$$

This shows that Equation (2) holds.

Step 6) We will now show that F' respects equivalence classes. Then it descends to the quotient and it follows that F is well defined. If $[\gamma] = [\delta]$ with δ another path from x to y, we want that

$$F'(\gamma) = F'(\delta). \tag{3}$$

Because of the way the equivalence classes are defined, it is enough to show this for γ and δ such that $\gamma \rightsquigarrow \delta$. Let in that case a directed path homotopy H from γ to δ be given. We take $n, m > 0$ such that H is (n, m)-covered by $\{X_1, X_2\}$. Firstly assume that $n > 1$. Restricting H to the rectangle $\left[0, \frac{1}{n}\right] \times [0, 1]$ gives us a directed path homotopy H_1 from γ to the directed path η given by $\eta(t) = H\left(\frac{1}{n}, t\right)$. By restricting H to the rectangle $\left[\frac{1}{n}, 1\right] \times [0, 1]$ we get a directed path homotopy H_2 from η to δ. It is clear that H_1 is $(1, m)$-covered and that H_2 is $(n-1, m)$-covered. By applying induction on n, we can conclude that it is enough to show that Equation (3) holds for $(1, m)$-covered directed path homotopies, as we would obtain that $F'(\gamma) = F'(\eta) = F'(\delta)$.

Step 7) We will prove the case where H is $(1, m)$-covered by showing a more general statement:

Let H be any directed homotopy – not necessarily a path homotopy – from one path $\gamma \in P_X(x, y)$ to another path $\delta \in P_X(x', y')$ that is $(1, m)$-covered, $m > 0$. Let η_0 be the path given by $\eta_0(t) = H(t, 0)$ and η_1 be given by $\eta_1(t) = H(t, 1)$. Then $F'(\eta_0 \odot \delta) = F'(\gamma \odot \eta_1)$. We do this by induction on m.

In the case that $m = 1$, we have a homotopy contained in X_1 or X_2. Without loss of generality, we can assume it is contained in X_1. Let Γ_1 be the directed homotopy given by $\Gamma_1(t, s) = \eta_0(\min(t, s))$ from 0_x to η_0. Let Γ_2 be the directed homotopy given by $\Gamma_2(t, s) = \eta_1(\max(t, s))$ from η_1 to $0_{y'}$. We then can construct a directed path homotopy from $(0_x \odot \gamma) \odot \eta_1$ to $(\eta_0 \odot \delta) \odot 0_{y'}$ given by $(\Gamma_1 \odot H) \odot \Gamma_2$. It is a directed path homotopy because $\Gamma_1(t, 0) = \eta_0(\min(t, 0)) = \eta_0(0) = x$ and $\Gamma_2(t, 1) = \eta_1(\max(t, 1)) = \eta_1(1) = y'$ for all $t \in I$. As η_0, η_1 and H are all contained in X_1, this directed path homotopy will be contained in X_1 as well. We find that $[\gamma \odot \eta_1] = [\eta_0 \odot \delta]$ in $\vec{\Pi}(X_1)$. This gives us that $F'(\gamma \odot \eta_1) = F_1[\gamma \odot \eta_1] = F_1[\eta_0 \odot \delta] = F'(\eta_0 \odot \delta)$.

Let now $m > 1$ and assume the statement holds for $(1, m-1)$-covered homotopies. We can restrict H to $[0, 1] \times \left[0, \frac{m-1}{m}\right]$ to obtain a $(1, m-1)$-covered homotopy H_1, say from γ_1 to δ_1. Similarly, we can restrict H to $[0, 1] \times \left[\frac{m-1}{m}, 1\right]$ to obtain a $(1, 1)$-covered homotopy H_2, say from γ_2 to δ_2. We write η' for the path given by $\eta'(t) = H(t, \frac{m-1}{m}) = H_1(t, 1) = H_2(t, 0)$.

Note that $F'(\gamma) = F'(\gamma_2) \circ F'(\gamma_1)$ by definition, because γ_1 is $(m-1)$-covered, γ_2 is 1-covered and γ is m-covered. Similarly it holds that $F'(\delta) = F'(\delta_2) \circ F'(\delta_1)$. We find:

$$\begin{aligned}
F'(\gamma \odot \eta_1) &= F'(\eta_1) \circ F'(\gamma) && \text{(Equation (2))}\\
&= F'(\eta_1) \circ (F'(\gamma_2) \circ F'(\gamma_1))\\
&= (F'(\eta_1) \circ F'(\gamma_2)) \circ F'(\gamma_1)\\
&= (F'(\delta_2) \circ F'(\eta')) \circ F'(\gamma_1) && \text{(Case } m = 1)\\
&= F'(\delta_2) \circ (F'(\eta') \circ F'(\gamma_1))\\
&= F'(\delta_2) \circ (F'(\delta_1) \circ F'(\eta_0)) && \text{(Induction Hypothesis)}\\
&= (F'(\delta_2) \circ F'(\delta_1)) \circ F'(\eta_0)\\
&= F'(\delta) \circ F'(\eta_0)\\
&= F'(\eta_0 \odot \delta) && \text{(Equation (2)).}
\end{aligned}$$

This proves the statement. From the statement we find that Equation (3) holds:

$$F'(\delta) = F'(\delta) \circ \mathrm{id}_x = F'(\delta) \circ F'(0_x) = F'(0_x \odot \delta) =$$
$$F'(\gamma \odot 0_y) = F'(0_y) \circ F'(\gamma) = \mathrm{id}_x \circ F'(\gamma) = F'(\gamma).$$

Here, the fourth equality follows from the statement. We conclude that F is well defined.

Step 8) As we have $F[\gamma] = F'(\gamma)$, it is immediate that F is a functor by Equation (1) and Equation (2). The equalities $F \circ \vec{\Pi}(j_1) = F_1$ and $F \circ \vec{\Pi}(j_2) = F_2$ hold by construction: if γ is contained in X_1, then $\gamma_{1,1}$ is as well, so $(F \circ \vec{\Pi}(j_1))[\gamma] = F[j_1 \circ \gamma] = F'(\gamma) = F_1[\gamma_{1,1}] = F_1[\gamma]$. We conclude that the commutative square is indeed a pushout. ◀

5.2 Formalisation

In the formalisation of Theorem 21, we follow the constructive nature of its proof. It can be found in `directed_van_kampen.lean`. We have the following global variables, corresponding with the assumptions of the Van Kampen Theorem:

```
variable {X : dTopCat.{u}} {X₁ X₂ : Set X}
variable (hX : X₁ ∪ X₂ = Set.univ)
variable (X₁_open : IsOpen X₁) (X₂_open : IsOpen X₂)
```

Like in the proof, we introduce a category C and two functors $F_1 : \vec{\Pi}(X_1) \to C$ and $F_2 : \vec{\Pi}(X_2) \to C$. Using these we are going to explicitly construct a functor from $\vec{\Pi}(X)$ to C and show that it is unique. We will use that to prove that we indeed have a pushout square.

```
variable {C : CategoryTheory.Cat.{u, u}}
variable (F₁ : (dπₓ (dTopCat.of X₁) ⟶ C))
variable (F₂ : (dπₓ (dTopCat.of X₂) ⟶ C))
variable (h_comm : (dπₘ i₁) ≫ F₁ = (dπₘ i₂) ≫ F₂)
/- Here we use two abbreviations:
 i₁ = dTopCat.DirectedSubsetHom (Set.inter_subset_left X₁ X₂)
 i₂ = dTopCat.DirectedSubsetHom (Set.inter_subset_right X₁ X₂)
-/
```

The variable `h_comm` is the assumption that the two maps F_1 and F_2 out of C form a commutative square when composed with the inclusions $\vec{\Pi}(X_1) \to \vec{\Pi}(X)$ and $\vec{\Pi}(X_2) \to \vec{\Pi}(X)$. These inclusions are obtained by `DirectedSubsetHom`, defined in `dTop.lean`. This defines the inclusion morphism $X_0 \to X_1$ in **dTop** in the case that $X_0 \subseteq X_1 \subseteq X$. We start with defining the functor F on objects (**Step 1**).

```
def FunctorOnObj (x : dπₓ X) : C := Or.by_cases
  ((Set.mem_union x.as X₁ X₂).mp (Filter.mem_top.mpr hX x.as))
    (fun hx => F₁.obj ⟨x.as, hx⟩)
    (fun hx => F₂.obj ⟨x.as, hx⟩)
```

We use `Filter.mem_top.mpr hX x.as` to show that $x \in X_1 \cup X_2$. From this, we use `Set.mem_union` to obtain $x \in X_1$ or $x \in X_2$ and we can split by those cases to apply either F_1 or F_2. We abbreviate `FunctorOnObj hX F₁ F₂` to `F_obj` in our formalisation to maintain clarity. After this definition, there are two lemmas that prove for $k \in \{1, 2\}$ that $F(x) = F_k(x)$ if $x \in X_k$.

In the proof of Theorem 21, F' is first defined and it is then shown to be a valid definition. Within our Lean formalisation, we have to do these two parts in the reverse order. Once we have shown that the construction is well-defined, we can define F' in our formalisation. That is why **Step 2** will be completed later.

We use the definitions of `covered` and `covered_partwise`, shown in Section 2, to define the mapping of morphisms inductively (**Step 3**):

```
def FunctorOnHomOfCovered {γ : Dipath x y} (hγ : covered hX γ) :
    F_obj ⟨x⟩ ⟶ F_obj ⟨y⟩ :=
  Or.by_cases hγ
    (fun hγ => FunctorOnHomOfCoveredAux₁ hX h_comm hγ)
    (fun hγ => FunctorOnHomOfCoveredAux₂ hX h_comm hγ)

def FunctorOnHomOfCoveredPartwiseAux {n : ℕ} :
    ∀ (x y : X) (γ : Dipath x y) (hγ : covered_partwise hX γ n),
        F_obj ⟨x⟩ ⟶ F_obj ⟨y⟩ :=
  Nat.recOn n
    (fun _ _ _ hγ => F₀ hγ)
    (fun _ ih _ _ _ hγ => (F₀ hγ.1) ≫ (ih _ _ _ hγ.2))
```

In `FunctorOnHomOfCovered` we define what to do with a path γ that is 1-covered, that is, we map it to $F_1[\gamma]$ or $F_2[\gamma]$ depending on whether γ is contained in X_1 or X_2. It depends on `FunctorOnHomOfCoveredAux₁`, which specifies what $F_1[\gamma]$ should be, as $[\gamma]$ is a morphism in $\vec{\Pi}(X)$ and not in $\vec{\Pi}(X_1)$. We use $\mathtt{F}_0$ to abbreviate `FunctorOnHomOfCovered hX h_comm`. We can then use this base case to define `FunctorOnHomOfCoveredPartwiseAux` for an n-covered path inductively by applying $\mathtt{F}_0$ to the first covered part of γ. In the construction of `FunctorOnHomOfCoveredPartwiseAux`, the variables `x`, `y` and γ are given explicitly in order to use induction. We use this definition in order to define `FunctorOnHomOfCoveredPartwise` which uses these implicitly and we abbreviate it to $\mathtt{F}_n$ to maintain readability.

Since n is an input of the definition, we need to show that it is independent of the choice of n. The lemma `functorOnHomOfCoveredPartwise_unique` captures this (**Step 4**).

```
lemma functorOnHomOfCoveredPartwise_unique {n m : ℕ} {γ : Dipath x y}
  (hγ_n : covered_partwise hX γ n) (hγ_m : covered_partwise hX γ m) :
    Fₙ hγ_n = Fₙ hγ_m :=
  /- Proof omitted -/
```

This lemma makes use of the following lemma that shows that the image remains the same if we refine the partition of γ, so when we use an nk-covering instead of an n-covering.

```
lemma functorOnHomOfCoveredPartwise_refine {n : ℕ} (k : ℕ) :
  Π {x y : X} {γ : Dipath x y} (hγ_n : covered_partwise hX γ n),
    Fₙ hγ_n = Fₙ (covered_partwise_refine hX n k hγ_n) :=
  /- Proof omitted -/
```

Now we know that the image is independent of n, and because an $n > 0$ exists such that γ is n-covered (shown by `has_subpaths`), we can choose one such n and we obtain the following formalisation of F', completing **Step 2**. We abbreviate this map to `Fh_aux`.

```
def FunctorOnHomAux (γ : Dipath x y) : F_obj ⟨x⟩ ⟶ F_obj ⟨y⟩ :=
  Fₙ (Classical.choose_spec (has_subpaths hX X₁_open X₂_open γ))
```

Now we show that Equation (1) and Equation (2) from the proof hold (**Step 5**).

```
lemma functorOnHomAux_refl {x : X} :
  Fh_aux (Dipath.refl x) = 1 (F_obj ⟨x⟩) :=
  /- Proof omitted -/

lemma functorOnHomAux_trans {x y z : X} (γ₁ : Dipath x y)
    (γ₂ : Dipath y z) :
  Fh_aux (γ₁.trans γ₂) = Fh_aux γ₁ ≫ Fh_aux γ₂ :=
  /- Proof omitted -/
```

As shown in **Step 6**, we want to show that F' is invariant under the `Dihomotopic` relation. To do this we need to show the claim from the proof: if we have a directed homotopy H from f to g that is $(1, m)$-covered, then $F'[H(\_, 1)] \circ F'[f] = F'[g] \circ F'[H(\_, 0)]$ (**Step 7**).

```
lemma functorOnHomAux_of_homotopic_dimaps {m : ℕ} :
  Π {f g : D(I, X)} {H : DirectedMap.Dihomotopy f g}
    (_ : DirectedMap.Dihomotopy.coveredPartwise hX H 0 m),
  Fh_aux (Dipath.of_directedMap f) ≫ Fh_aux (H.eval_at_right 1) =
  Fh_aux (H.eval_at_right 0) ≫ Fh_aux (Dipath.of_directedMap g) :=
    /- Proof omitted -/
```

By using induction once again, we end up with the lemma showing us that the choice of representative does not matter.

```
variable (γ γ' : Dipath x y)
lemma functorOnHomAux_of_dihomotopic (h : γ.Dihomotopic γ') :
  Fh_aux γ = Fh_aux γ' :=
  /- Proof omitted -/
```

We can now finally define the behaviour on morphisms to obtain a functor by using the universal mapping property of quotients.

```
def FunctorOnHom {x y : dπₓ X} (γ : x ⟶ y) : F_obj x ⟶ F_obj y :=
  Quotient.liftOn γ Fh_aux
    (functorOnHomAux_of_dihomotopic hX X₁_open X₂_open h_comm)

def Functor : (dπₓ X) ⟶ C where
  obj := F_obj
  map γ := F_hom γ
  map_id x := functorOnHom_id hX X₁_open X₂_open h_comm x
  map_comp γ₁ γ₂ := functorOnHom_comp hX X₁_open X₂_open h_comm γ₁ γ₂
```

Here `F_hom` is an abbreviation for `FunctorOnHom` and the final `Functor` is abbreviated to `F`. Finally, we get to **Step 8**. The remaining lemmas show that $F \circ \vec{\Pi}(j_k) = F_k$ for $k = 1$ and $k = 2$, and that F is the unique functor with this property.

```
lemma functor_comp_left : (dπₘ j₁) ⋙ F = F₁ := /- Proof omitted -/
lemma functor_comp_right : (dπₘ j₂) ⋙ F = F₂ := /- Proof omitted -/
lemma functor_uniq (F' : (dπₓ X) ⟶ C) (h₁ : (dπₘ j₁) ⋙ F' = F₁)
    (h₂ : (dπₘ j₂) ⋙ F' = F₂) : F' = F := /- Proof omitted -/
```

The Van Kampen Theorem is stated as

```
theorem directed_van_kampen (_ : IsOpen X₁) (_ : IsOpen X₂)
    (hX : X₁ ∪ X₂ = Set.univ) :
  IsPushout (dπₘ i₁) (dπₘ i₂) (dπₘ j₁) (dπₘ j₂) :=
  /- Proof omitted -/
```

This theorem now follows easily from the lemmas above.

6 Conclusion and Further Research

In this article, we presented a formalisation of the Van Kampen Theorem in directed topology in the proof assistant Lean 4. This theorem allows one to calculate the fundamental category of a directed space using the fundamental categories of subspaces under a mild condition on the subspaces. At the moment, mathlib does not have a version of the Van Kampen Theorem for groupoids, originally proven by Brown in 1968 [2, 3]. The undirected version is a corollary of the directed version because the fundamental groupoid of a topological space can be seen as the fundamental category of a directed space, where all paths are directed. We have not formalised this implication, but it should not be hard to prove the Van Kampen Theorem for groupoids in this manner.

There are generalisations of the undirected version that allow an arbitrary open cover [4, Theorem 2.3.5]. An extension of our formalisation to allow this would be possible using the same general approach, but we have not investigated this in depth.

As a next step, it would be natural to formalise the relation between d-spaces and their homotopy theory with other models of concurrency, such as higher-dimensional automata and their languages, and to develop the homotopy theory of d-spaces further in Lean.

References

1. Henning Basold, Peter Bruin, and Dominique Lawson. The Directed Van Kampen theorem in Lean, 2023. Pre-print. arXiv:2312.06506.
2. Ronald Brown. *Elements of Modern Topology.* McGraw-Hill, 1968.
3. Ronald Brown. *Topology and Groupoids.* BookSurge Publishing, 2006.
4. Ronald Brown, Philip J. Higgins, and Rafael Sivera. *Nonabelian Algebraic Topology: Filtered spaces, crossed complexes, cubical homotopy groupoids.* European Mathematical Society, 2011. doi:10.4171/083.
5. Leonardo de Moura, Soonho Kong, Jeremy Avigad, Floris Van Doorn, and Jakob von Raumer. The Lean Theorem Prover (System Description). In Amy P. Felty and Aart Middeldorp, editors, *Automated Deduction-CADE-25: 25th International Conference on Automated Deduction*, Lecture Notes in Computer Science, pages 378–388. Springer, Springer International Publishing, 2015. doi:10.1007/978-3-319-21401-6_26.
6. Jérémy Dubut. *Directed Homotopy and Homology Theories for Geometric Models of True Concurrency.* PhD thesis, Université Paris-Saclay, 2017. URL: https://tel.archives-ouvertes.fr/tel-01590515.
7. Lisbeth Fajstrup. Dicovering Spaces. *Homology, Homotopy and Applications*, 5(2):1–17, 2003. doi:10.4310/HHA.2003.v5.n2.a1.
8. Lisbeth Fajstrup, Eric Goubault, Emmanuel Haucourt, Samuel Mimram, and Martin Raußen. *Directed Algebraic Topology and Concurrency.* Springer, 2016. doi:10.1007/978-3-319-15398-8.
9. Lisbeth Fajstrup, Eric Goubault, and Martin Raußen. Detecting Deadlocks in Concurrent Systems. In *CONCUR '98: Concurrency Theory, 9th International Conference, Nice, France, September 8-11, 1998, Proceedings*, pages 332–347, 1998. doi:10.1007/BFb0055632.

10 Lisbeth Fajstrup, Martin Raußen, and Eric Goubault. Algebraic topology and concurrency. *Theoretical Computer Science*, 357(1):241–278, 2006. Clifford Lectures and the Mathematical Foundations of Programming Semantics. doi:10.1016/j.tcs.2006.03.022.

11 Philippe Gaucher. Six Model Categories for Directed Homotopy. *Categories and General Algebraic Structures with Applications*, 15(1):145–181, 2021. doi:10.52547/cgasa.15.1.145.

12 Marco Grandis. Directed homotopy theory, I. The fundamental category. *Cahiers de topologie et géométrie différentielle catégoriques*, 44(4):281–316, 2003. URL: http://archive.numdam.org/item/CTGDC_2003__44_4_281_0/.

13 Marco Grandis. *Directed Algebraic Topology: Models of Non-Reversible Worlds*. New Mathematical Monographs. Cambridge University Press, 2009. doi:10.1017/CBO9780511657474.

14 Kuen-Bang Hou (Favonia) and Michael Shulman. The Seifert-van Kampen theorem in homotopy type theory. In *25th EACSL Annual Conference on Computer Science Logic, CSL 2016 and the 30th Workshop on Computer Science Logic*. Schloss Dagstuhl – Leibniz-Zentrum für Informatik, Dagstuhl Publishing, 2016. doi:10.4230/LIPIcs.CSL.2016.22.

15 Sanjeevi Krishnan. A Convenient Category of Locally Preordered Spaces. *Applied Categorical Structures*, 17(5):445–466, 2009. doi:10.1007/s10485-008-9140-9.

16 Dominique Lawson. GitHub - Dominique-Lawson/Directed-Topology-Lean-4. Software, version 1.1., swhId: swh:1:dir:479a73373a2bf508149f7d1b889b42304fe78a9e (visited on 2024-07-08). URL: https://github.com/Dominique-Lawson/Directed-Topology-Lean-4/tree/v1.1.

17 Tom Leinster. *Basic Category Theory*. Cambridge University Press, 2014. doi:10.1017/cbo9781107360068.

18 The mathlib Community. The Lean Mathematical Library. In *Proceedings of the 9th ACM SIGPLAN International Conference on Certified Programs and Proofs*, CPP 2020, pages 367–381, New Orleans, LA, 2020. ACM. doi:10.1145/3372885.3373824.

19 James R. Munkres. *Topology, a first course*. Prentice-Hall, 1975.

20 Vaughan R. Pratt. Modeling Concurrency with Geometry. In *Conference Record of the Eighteenth Annual ACM Symposium on Principles of Programming Languages (POPL)*, pages 311–322, 1991. doi:10.1145/99583.99625.

21 Floris van Doorn, Jakob von Raumer, and Ulrik Buchholtz. Homotopy type theory in Lean. In *Interactive Theorem Proving: 8th International Conference, ITP 2017, Brasília, Brazil, September 26–29, 2017, Proceedings 8*, pages 479–495. Springer, 2017. doi:10.1007/978-3-319-66107-0_30.

22 Rob J. van Glabbeek. On the expressiveness of higher dimensional automata. *Theoretical Computer Science*, 356(3):265–290, 2006. doi:10.1016/j.tcs.2006.02.012.