FACULTY
OF SCIENCES

FINITE TOPOLOGICAL SPACES
IN ALGEBRAIC TOPOLOGY

Jens Renders

Student number: 01700256

Supervisior: Prof. Dr. Koen Thas

Department of Mathematics: Algebra and Geometry

A dissertation submitted to Ghent University in partial fulfillment of the requirements for the degree of Master of Science in Mathematics

Academic year: 2018 - 2019

GHENT
UNIVERSITY

Acknowledgements

This thesis marks the end of five years of studying mathematics. For those five years, I owe a huge thank you to my parents. They supported my choice to study mathematics from the very beginning. I do not take this for granted, as my study results op to that point had always been subpar. Thank you for believing in me from the beginning, maybe more than I believed in myself.

I would like to thank my fellow students, as they were a part of this experience throughout. They were always available for deep and long discussions on what's right and wrong, what should and shouldn't be. Some of those discussions were useless, but some lead to useful insights, and even contributions to this work. I would especially like to thank my friend Noémie. She insisted on being mentioned in this acknowledgement, and I have to admit: she deserves it. Thank you Noémie for being there for five years, for being present at all my presentations (obligatory or not, practice or not) and for doing a truly unhealthy amount of proofreading.

For this thesis specifically, I would like to thank my promoter, professor Koen Thas, for enabling me to write it. Thank you for your useful advice and critique, and for giving me plenty of freedom for exploration.

Toelating tot bruikleen

De auteur geeft de toelating deze masterproef voor consultatie beschikbaar te stellen en delen van de masterproef te kopiëren voor persoonlijk gebruik. Elk ander gebruik valt onder de beperkingen van het auteursrecht, in het bijzonder met betrekking tot de verplichting de bron uitdrukkelijk te vermelden bij het aanhalen van resultaten uit deze masterproef.

Jens Renders
31 mei 2019

Preliminary Remarks

Typesetting a thesis inevitably provides some dilemmas, especially when writing a thesis on mathematics, containing lots of formulas that can conflict with English spelling rules. I want to take the time to clarify some of the choices that were made in this thesis.

Punctuation after formulas

All inline math is formatted as if it is an ordinary word in a sentence, like the equation $x = 3$. The period comes right after the formula. Display math (block math), however, is more problematic. Just like a poem, one might use a colon before it, and consider the sentence ended without period, like this:

Something poetic, the writer decided not to use a period

Here, a new sentence starts with a capital letter, although there was no period. Writing a period at the end of the poem might destroy the poem, and writing the period on the next line looks bad. The same holds for mathematical formulas. Many authors end display math with a period, like this:

$$f = x.$$

But this can also destroy the formula. Say for example that I have a sequence $(x_n)_{n\in\mathbb{N}}$ that I now want to regard as a function f on the natural numbers. Then I might write

$$f = x_\cdot.$$

The first dot is a subscripted `\cdot`, a placeholder for the variable, while the second dot is the period. I avoid this confusion by never writing a punctuation mark at the end of a display math formula. The display math formula is a way to separate the math from the rules of punctuation on English. However, contrary to the poem, a display math formula might still be part of a sentence. For example: the integral

$$\int x^2 \mathrm{d}x$$

is easy. But can you compute the integral

$$\int \sqrt{\tan(x)} \mathrm{d}x$$

The first sentence continued after the formula, while the second sentence ended without a period. I do acknowledge that this is not preferable for some readers, but certainly wins from the alternative for me. Therefore, I made this choice.

Line length and interline distance

When writing a simple LaTeX document, the line length and interline distance is set for you. The lines are not too long such that the reader doesn't accidentally jump to the wrong line when going to the next line, while the interline distance looks great to the eye. The downside is that this leaves huge margins when using a standard font size on A4 paper. The optimal choice would be to print on a smaller paper size, like many books do. This thesis however, is required to be printed on A4 paper, and the margins are also fixed. When using a standard font size, this leaves no control over line length. To ensure readability, this thesis uses a large interline distance of 1.4 times the point size. This is an acceptable solution to me, but not as good as the default LaTeX settings.

Contents

Chapter 1

Introduction

Point-Set topology is often used in analysis, but usually with more axioms than the ordinary definition of topology contains. One extra property that is often asked is the separation axiom T_2 (the Hausdorff property). However, if a finite topological space is T_2, or even T_1, then it is discrete. This explains for the most part why we don't see finite topological spaces popping up that often in analysis. If they do pop up they are usually discrete and not very interesting.

But what if we could model the infinite spaces of interest by finite topological spaces? What if we had an infinite space X, and we could turn it into a finite (but usually not T_2) space Y while preserving some properties? Then we could put Y in a computer and perform a computational check to infer the preserved properties of X. McCord [16] shows how we can do this with a large class of spaces of interest, for all properties that only depend on the weak homotopy type of the space. For example, the singular homology groups only depend on the weak homotopy type of a space and are thus preserved when going over to the finite model. The spaces for which he can do this are those spaces that are homeomorphic to the geometric realisation of a finite simplicial complex.

If you don't know what *weak homotopy type*, *homology group* or *simplicial complex* means, don't worry. Those concepts, and all other necessary concepts from algebraic topology, are introduced in chapter 2. This way, the thesis is readable for anyone with a basic background in point-set topology, and no background in algebraic topology. This is the same background the author had at the beginning of the research for this thesis. Sometimes, we also make use of some category theoretic language, but it is mostly just that: language. No deep results from category theory are used, and the language is just a search engine query away.

In chapter 3 we dive into the meat of the theory. Most of the fundamental theorems of finite topological spaces are covered in this chapter. We start with Alexadroff's theory that connects finite topological spaces to finite posets [1]. He gives a bijective correspondence between finite topological spaces and finite pre-ordered sets. In this correspondence, continuous maps correspond to order preserving maps, so we can interpret it as a concrete category isomorphism. The topological spaces that correspond to posets under this correspondence are precisely the T_0 spaces. Next, we discuss how homotopy theory in the finite setting can be studied nicely using the connection to order theory. This is due to Stong [21]. He gave a clean order theoretic characterisation of homotopies, and uses this to classify homotopy types up to homeomorphism. Having this strong tool set to deal with finite spaces and homotopy theory on them, we finally introduce the theory of McCord, which connects the world of finite spaces with the world of infinite spaces. This connection goes both ways. It shows how we can model many infinite spaces by finite spaces, while preserving the weak homotopy type. But it also shows that all finite spaces are weak homotopy equivalent to a finite simplicial complex (which are infinite spaces). This makes it easy to compute homological properties of finite spaces: thinking about singular homology of finite spaces can be challenging at first, but thanks to McCord, this can always be translated to simplicial homology.

In chapter 4 we deal with the computational side of finite topological spaces. We start by showing how to compute the *core* of a finite topological space, which is an object introduced in the previous chapter, that makes it easy to check if two finite spaces are homotopy equivalent. Next, we look at the computation of algebraic invariants of finite spaces that depend on its singular homology. These invariants are: the homology groups themselves, the Betti numbers and the Euler characteristic. All of those invariants can be immediately computed using McCord's theory, by converting the finite topological space to a finite simplicial complex and then computing the corresponding invariant of the simplicial complex. We show how to do this in Sage [22]. Although this is easy, it is not very efficient. The number of simplices in the simplicial complex can be exponential in function of the number of points of the finite space. Therefore, we look for alternatives in this chapter.

Chapter 2

Introduction to Algebraic Topology

2.1 Homotopy theory

Homotopy theory is a theory that can be used to describe some notion of 'shape' of a space. It can be seen as a generalisation of path-connectedness. Intuitively, one can see a that closed interval in $\mathbb{R}$ and a circle in $\mathbb{R}^2$ are not homeomorphic spaces. But what topological property is different in these spaces? All local properties usually dealt with in analytical topology are the same for these spaces. Homotopy theory will provide tools to distinguish these (and many more) spaces by describing 'how' they are connected. Standard reference texts for this section are [19] and [11].

2.1.1 Homotopies and Homotopy Groups

Definition 2.1.1. *Let X, Y be topological spaces, and let $f, g : X \to Y$ be continuous maps. A* ***homotopy*** *from f to g is a continuous map*

$$h : X \times [0,1] \to Y : (x,t) \mapsto h(x,t)$$

such that $h(\cdot, 0) = f$ and $h(\cdot, 1) = g$. We say that f and g are ***homotopic*** *if there is a homotopy between them. We denote this by $f \simeq g$. It is clear that $\simeq$ is an equivalence relation.*

As such, a homotopy can be seen as a continuous interpolation between two maps.

Paths are continuous functions with domain $[0, 1]$. Therefore, a homotopy between paths is defined by the previous definition. Usually though, we work with a slightly stronger concept of homotopies between paths:

Definition 2.1.2. *Let X be a topological space and let $f, g : [0,1] \to X$ be paths in X with the same end points ($f(0) = g(0)$ and $f(1) = g(1)$). A **path-homotopy** h from f to g is a homotopy from f to g that preserves the endpoints. That is*

$$\forall t \in [0,1] : h(0,t) = f(0) = g(0) \text{ and } h(1,t) = f(1) = g(1)$$

*We say that f and g are **path-homotopic** if there is a path-homotopy between them. It is clear that being path-homotopic is an equivalence relation.*

Path homotopies are central in the definition of the *fundamental group*. The fundamental group can be seen as the first algebraic invariant of topological spaces, the prototype of algebraic topology.

Definition 2.1.3. *Let X be a topological space and let $x_0 \in X$. The **fundamental group with basepoint** x_0 of X (denoted $\pi_1(X, x_0)$) is the group with as underlying set the set of paths that end and begin in x_0*

$$\{p : [0,1] \to X \mid p \text{ is continuous and } p(0) = p(1) = x_0\}$$

modulo path-homotopy. The group operation is given by concatenation of paths:

$$(f * g)(t) = \begin{cases} f(2t) & \text{if } t < \frac{1}{2} \\ g(2t-1) & \text{if } t \geq \frac{1}{2} \end{cases}$$

One should check that this operation is well-defined, and that this definition satisfies the axioms of a group. This is not difficult, and omitted here. We also record the following result without proof:

Proposition 2.1.4. *Let X be a path-connected topological space and let $x_0, y_0 \in X$. Then $\pi_1(X, x_0) \cong \pi_1(X, y_0)$, i.e. the fundamental group is independent of the choice of basepoint. In this case, one can just write $\pi_1(X)$ to unambiguously refer to the fundamental group of X for any basepoint up to isomorphism.*

Let's return to our example of the closed interval and the circle. Those are path-connected spaces, so they have a unique fundamental group. If the closed interval and the circle are homeomorphic, then their fundamental groups should be isomorphic. By contraposition, if their fundamental groups are not isomorphic, the spaces are not homeomorphic. So using the fundamental group, distinguishing topological spaces can sometimes be reduced to discriminating groups, which is often easier. This is one of the main motivations for the development of algebraic topology. This does require the ability to compute the fundamental group though, which can be hard. In this case it is luckily not too hard to see that the fundamental group of the closed interval is trivial, and the fundamental group of the circle is $\mathbb{Z}$. The conclusion is that the circle and the closed interval are not homeomorphic.

The fundamental group can be useful, but it has some shortcomings too. It describes the connectedness of a space in and inherently one-dimensional way (using only paths which are one-dimensional in nature). If we want to distinguish the space $[0,1]^2$ from the 2-sphere in $\mathbb{R}^3$, this becomes a problem. In both of those spaces we can contract loops to points, which implies that their fundamental groups are trivial. This is of no help to show that the spaces are not homeomorphic. We could try to create a variant of the fundamental group using '2-dimensional paths'. Of course, we would then run into problems again when trying to distinguish $[0,1]^3$ from the 3-sphere. Therefore, we define a version of the fundamental group for each $n \in \mathbb{N}$.

Definition 2.1.5. *Let X be a topological space and let $x_0 \in X$. We write $\partial[0,1]^n$ for the boundary of the unit cube in $\mathbb{R}^n$. For $n \geq 1$ we define the* ***n-th homotopy group*** *(denoted $\pi_n(X, x_0)$) as the group with underlying set*

$$\{p : [0,1]^n \to X \mid p \text{ is continuous and } \forall t \in \partial[0,1]^n : p(t) = x_0\}$$

modulo basepoint-preserving-homotopy, i.e. homotopies $h : [0,1]^n \times [0,1] \to X$ that satisfy

$$\forall t \in \partial[0,1]^n : \forall s \in [0,1] : h(t,s) = x_0$$

The group operation is given by

$$(f * g)(t_1, t_2, \cdots, t_n) = \begin{cases} f(2t_1, t_2, \cdots, t_n) & \text{if } t_1 < \frac{1}{2} \\ g(2t_1 - 1, t_2, \cdots, t_n) & \text{if } t_1 \geq \frac{1}{2} \end{cases}$$

The case for $n = 0$ is a bit special. Zero dimensional paths are just points (we don't work with a basepoint in this case), and a homotopy between them is simply a path between the points. The set of zero dimensional paths modulo homotopy is therefore just the set of path components. We denote it $\pi_0(X)$. It is not equipped with a group operation.

Again, one should check that this operation is well-defined and that the definition satisfies the group axioms. Also, notice that in the case $n = 1$ we use the same notation $\pi_1(X, x_0)$ as for the fundamental group. This is of course because in that case the definitions are identical.

2.1.2 Weak isomorphisms

We introduced homotopy and the homotopy groups as a method to distinguish topological spaces. However, not all spaces can be distinguished like this. There are topological spaces which are not isomorphic, but for example still have the same homotopy groups. An example is a single point and a closed interval. They cannot be isomorphic as they have different cardinality, but they do have the same homotopy groups (all of them are trivial). We could say that these spaces have the same 'shape' in the sense of homotopy theory, while they have a different shape in the sense of general topology.

In general topology, we express that two spaces are the same using homeomorphisms, or in other words, topological isomorphisms. In homotopy theory there are weaker versions of isomorphisms that express that two spaces are the same at homotopy level (but not necessarily at a topological level). The first such weak isomorphism is called a *homotopy equivalence*, and it is directly related to the following definition of a homeomorphism which we now state for comparison:

Definition 2.1.6. *Let X and Y be topological spaces. A continuous map $f : X \to Y$ is called a* ***homeomorphism*** *or* ***topological isomorphism*** *if there is a continuous map $g : Y \to X$ such that*

$$g \circ f = \mathrm{id}_X \text{ and } f \circ g = \mathrm{id}_Y$$

The map g is called an ***inverse*** *of f. We call X and Y* ***homeomorphic*** *or* ***topologically isomorphic*** *if there is a homeomorphism $f : X \to Y$. It is clear that being homeomorphic is an equivalence relation.*

By changing the equalities to homotopies, we obtain the definition of a *homotopy equivalence*:

Definition 2.1.7. *Let X and Y be topological spaces. A continuous map $f : X \to Y$ is called a* ***homotopy equivalence*** *if there is a continuous map $g : Y \to X$ such that*

$$g \circ f \simeq \mathrm{id}_X \text{ and } f \circ g \simeq \mathrm{id}_Y$$

The map g is called a ***homotopy inverse*** *of f. We call X and Y* ***homotopy equivalent*** *if there is a homotopy equivalence $f : X \to Y$. It is clear that being homotopy equivalent is an equivalence relation. The equivalence classes of this relation are called* ***homotopy types****.*

Definition 2.1.8. *Let X be a topological space. If X is homotopy equivalent to the one point space $\{*\}$, then X is called* ***contractible****.*

Proposition 2.1.9. *Let X be a topological space. Then X is contractible if and only if there is a homotopy between id_X and a constant map $X \to X$.*

Proof. If X is contractible, we have a homotopy equivalence

$$X \underset{g}{\overset{f}{\rightleftarrows}} \{*\}$$

Here, $g \circ f$ is constant and homotopy equivalent to id_X.

Conversely, if we have a constant map $c_a : X \to X : x \mapsto a$ which is homotopic to id_X, then the map

$$f : X \to \{*\} : x \mapsto *$$

is a homotopy equivalence with homotopy inverse

$$g : \{*\} \to X : * \mapsto a$$

Indeed, the composition $g \circ f$ is equal to $c_a \simeq \mathrm{id}_X$, and the composition $f \circ g$ is equal to $\mathrm{id}_{\{*\}}$ □

Example 2.1.10. *The space $X = [0, 1]$ is contractible. Using the following proposition (2.1.11), this shows that the homotopy groups of X are trivial.*

Proof. A homotopy between id_X and the constant map $X \to X : x \mapsto 0$ is given by

$$h : X \times [0, 1] \to X : (x, t) \mapsto xt$$

□

We record the following standard theorem without proof. It expresses how spaces that are homotopy equivalent are indeed the same, according to the homotopy groups. A proof can be found in [11] proposition 1.31 and proposition 4.1.

Proposition 2.1.11. *Let X and Y be topological spaces. Let $f : X \to Y$ be a homotopy equivalence. Then f induces group isomorphisms on the homotopy groups, i.e:*

$$\forall n \geq 1 : \forall x \in X : (\hat{f} : \pi_n(X, x) \to \pi_n(Y, f(x)) : [p] \mapsto [f \circ p]) \text{ is an isomorphism}$$

and f induces a bijection on the set of path components, i.e:

$$(\hat{f} : \pi_0(X) \to \pi_0(Y) : [p] \mapsto [f(p)]) \text{ is a bijection}$$

So in other words, a homotopy equivalence *is* an isomorphism, at the level of the homotopy groups. This is only a consequence though, not an equivalent characterisation (see example 3.4.24). Maps that satisfy this property are called *weak homotopy equivalences*.

Definition 2.1.12. *Let X and Y be topological spaces. Let $f : X \to Y$ be a continuous map. We call f a **weak homotopy equivalence** if f induces group isomorphisms on the homotopy groups, i.e.*

$$\forall n \geq 1 : \forall x \in X : (\hat{f} : \pi_n(X, x) \to \pi_n(Y, f(x)) : [p] \mapsto [f \circ p]) \text{ is an isomorphism}$$

and f induces a bijection on the set of path components, i.e:

$$(\hat{f} : \pi_0(X) \to \pi_0(Y) : [p] \mapsto [f(p)]) \text{ is a bijection}$$

This is clearly much less of a topological definition. It is also a bit harder to interpret as an equivalence relation, since there is no mention of a 'weak homotopy inverse'. It is possible that there is a weak homotopy equivalence $f : X \to Y$ but no weak homotopy equivalence $g : Y \to X$. In fact, this situation will often appear in this thesis (as illustrated by example 3.4.27). The solution to this problem is to take the symmetric and transitive closure.

Definition 2.1.13. *Let X and Y be topological spaces. We say that X and Y are* ***weak homotopy equivalent*** *if (X, Y) is in the symmetric and transitive closure of the relation*

$$\{(Z, W) \mid Z, W \text{are topological spaces and there is a weak homotopy equivalence } f : Z \to W\}$$

In other words, if there exist spaces $X_1, \cdots, X_n$ and weak homotopy equivalences

$$X \to X_1 \leftarrow X_2 \to \cdots \leftarrow X_n \to Y$$

This closure is clearly an equivalence relation. The equivalence classes of this closure are called ***weak homotopy types****.*

It is a bit artificial to take this closure, but at the level of the homotopy groups this works out nicely. If X and Y are weak homotopy equivalent, we have weak homotopy equivalences

$$X \to X_1 \leftarrow X_2 \to \cdots \leftarrow X_n \to Y$$

so we have isomorphisms

$$\pi_i(X, x) \to \pi_i(X_1, x_1) \leftarrow \pi_i(X_2, x_2) \to \cdots \leftarrow \pi_i(X_n, x_n) \to \pi_i(Y, y)$$

and contrary to weak homotopy equivalences, these isomorphisms are invertible, so we also have isomorphisms

$$\pi_i(X, x) \to \pi_i(X_1, x_1) \to \pi_i(X_2, x_2) \to \cdots \to \pi_i(X_n, x_n) \to \pi_i(Y, y)$$

so by composition we obtain an isomorphism $\pi_i(X, x) \to \pi_i(Y, y)$. It is clear that both versions of homotopy equivalences have their pro's and con's. Weak homotopy equivalences correctly represent maps that are isomorphisms on the level of homotopy groups, while homotopy equivalences are a bit too strong for that, but they work much nicer and have a clean topological definition.

To end this section, we introduce *deformation retracts*. In short, a deformation retract is a subspace that can be obtained by shrinking a topological space while maintaining the homotopy type.

Definition 2.1.14. *Let X be a topological space and let $A \subseteq X$ be a subspace of X. A continuous map $r : X \to A$ is called a* ***retraction*** *if $r|_A = \mathrm{id}_A$. The subspace A is called a* ***retract*** *if there is such a retraction.*

Definition 2.1.15. *Let X be a topological space and let $A \subseteq X$ be a subspace of X. A continuous map $r : X \to A$ is called a* ***deformation retraction*** *if it is a retraction and a homotopy equivalence with homotopy inverse the natural injection:*

$$i : A \to X : a \mapsto a$$

The subspace A is called a ***deformation retract*** *if there is such a retraction.*

2.2 Homology

Homotopy gives a useful algebraic invariant of topological spaces, but in many cases it is hard to compute. There is an alternative way to describe holes and connectivity features of a topological space, called *homology*. It is very similar to homotopy, but often easier to compute. It comes in many variations, of which the first and most simple is *simplicial homology*. A good reference textbook for beginners is [11], while a textbook that goes a bit deeper and is closer to our language is [15].

2.2.1 Simplicial complexes

The idea is to build a homeomorphic copy of a given space of interest, out of very simple building blocks. In some sense, the most basic building block in n dimensions is the n-*simplex*. This is a generalisation of a point, line, triangle and tetrahedron to n dimensions. These first four examples can be defined as the convex hull of 1, 2, 3 and 4 points respectively. This leads to the general definition of a simplex.

Definition 2.2.1. *The convex hull of $n + 1$ affinely independent points in $\mathbb{R}^n$ is called a* ***simplex*** *of dimension n or an* ***n-simplex***.

The sides of a triangle can be obtained by removing one point and looking at the convex hull of the remaining. In general, we can define the *facets* of a simplex this way:

Definition 2.2.2. *Let σ be a simplex generated by $\{v_0, \cdots, v_n\}$. The* ***facets*** *of σ are the simplices obtained as the convex hull of*

$$\{v_0, \cdots, v_{i-1}, v_{i+1}, \cdots, v_n\}$$

In other words, by removing one point from the generating set. When removing multiple points of the generating set, we speak of a ***face****.*

If we endow $\mathbb{R}^n$ with the euclidean topology, then simplices become topological (sub)spaces. A *simplicial complex* is a collection of simplices which can be regarded together as a topological space which is a little more complex.

Definition 2.2.3. *A* ***(geometric) simplicial complex*** K *is a set of simplices such that:*

- *If* $\sigma \in K$*, then the faces of* σ *are also in* K*.*
- *The intersection of two simplices* $\tau, \sigma \in K$ *is a face of both* τ *and* σ*.*

The union of the simplices in K *is called the carrier of* K*.*

A simplicial complex is determined by an easier combinatorial structure.

Definition 2.2.4. *An* ***(abstract) simplicial complex*** *is a collection* K *of non-empty subsets of a set* V*, which is closed under taking subsets, i.e.:*

$$\tau \subseteq \sigma \in K \implies \tau \in K$$

The elements of V *are called the* ***points*** *or* ***vertices*** *of the simplicial complex.*

In this more abstract version, a set $\{v_0, \cdots, v_n\}$ represents an n-simplex, but all geometric information is lost. A face of $\{v_0, \cdots, v_n\}$ is now represented by a subset of $\{v_0, \cdots, v_n\}$. The condition that K has to be closed under taking subsets corresponds to the condition that faces have to be in the simplicial complex again. The intersection of two simplices in K is a subset of both, so it is a face of both.

We can regard simplicial complexes as topological spaces. This is done by regarding its simplices as topological spaces and glueing them together along the 'face of' relation using topological constructions. The topological space we obtain by this is called the *geometric realisation* of the simplicial complex. We give this construction explicitly in the more general case of simplicial sets, further in this thesis (definition 2.2.27). Usually, the topological spaces in which we are interested do not appear as geometric realisations of simplicial complexes. However, many spaces of interest are homeomorphic to such a geometric realisation.

Example 2.2.5. *The* 2*-sphere* S^2 *is homeomorphic to a hollow tetrahedron, but also to a hollow octahedron and a hollow icosahedron. These are all examples of simplicial complexes, as they are constructed out of triangles.*

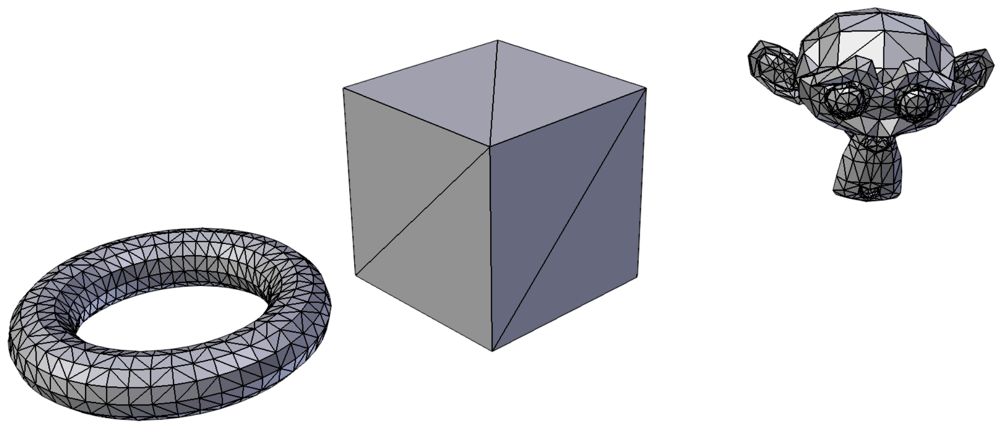


Figure 2.1: Simplicial complexes in 3D computer graphics

Example 2.2.6. *Figure 2.1 shows three simplicial complexes in the open source software Blender [3], a program for designing 3D computer graphics and animations. The first object is homeomorphic to the torus $S^1 \times S^1$, while the other two objects are homeomorphic to S^2.*

We now give a few small examples for easy experimentation.

Example 2.2.7. *A triangle is nothing but a 2-simplex, so it can be represented as the simplicial complex*

$$\{\{0, 1, 2\}, \{0, 1\}, \{1, 2\}, \{0, 2\}, \{0\}, \{1\}, \{2\}\}$$

where $V = \{0, 1, 2\}$. If we take out the 2-simplex itself, we obtain a hollow triangle

$$\{\{0, 1\}, \{1, 2\}, \{0, 2\}, \{0\}, \{1\}, \{2\}\}$$

Because we always have to include all subsets in the simplicial complex, it is redundant to write them all down. It is more efficient to just write down that the subsets need to be added, which we do with the symbol $\downarrow$. A triangle is then written as

$$\downarrow \{\{0, 1, 2\}\}$$

and a hollow triangle as

$$\downarrow \{\{0, 1\}, \{1, 2\}, \{0, 2\}\}$$

Example 2.2.8. *We can create a slightly more complicated simplicial complex by glueing together a hollow and a full triangle (figure 2.3). Here, $V = \{0, 1, 2, 3\}$ and the simplicial complex is*

$$\downarrow \{\{0, 1, 2\}, \{2, 3\}, \{1, 3\}\}$$

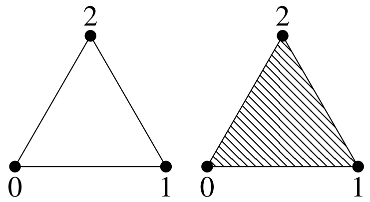


Figure 2.2: A hollow and full triangle

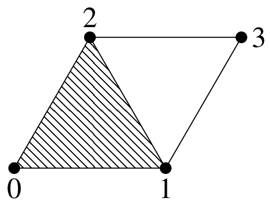


Figure 2.3: A hollow and a full triangle glued together

2.2.2 Simplicial Homology

Similar to homotopy, homology describes how a space is connected, by somehow describing what kind of holes of what dimensions are present in the space. This intuition should be kept in mind when reading the following definitions. Thanks to the simple combinatorial structure of simplicial complexes, we can give a precise definition of the following idea: a 1-dimensional hole is a loop with no points inside of it. It is easy to define loops, but some loops do not correspond to holes; they are filled with points. Those loops (that are not holes) have the property that they are the boundary of something 2-dimensional (the points inside of them). Therefore we can distinguish between loops that are the boundary of something 2-dimensional (not holes) and loops that are not the boundary of something 2-dimensional (holes). For example, in figure 2.3 the loop $0 \to 1 \to 2 \to 0$ is not a hole, as it is the boundary of the triangle $\{0, 1, 2\}$. The loop $1 \to 2 \to 3 \to 1$ however, does constitute a hole as it is not the boundary of anything. We will also be able to express this for higher dimensions.

We start by ordering the points of an abstract simplicial complex. This ordering will represent the orientation of the simplices.

Definition 2.2.9. *An* ***ordered simplicial complex*** *is an (abstract) simplicial complex with a partial order on the set of points of the simplicial complex, such that the restriction of this order to any of the simplices is a total order.*

It doesn't matter which order is chosen, but it is important that we choose one and stick to it. In the remainder of this section, K shall represent an ordered simplicial complex.

Definition 2.2.10. *The* ***group of n-chains*** *of K, denoted C_n, is the free abelian group over the n-simplices of K. This means that by* ***n-chain****, we mean a formal sum of n-simplices.*

In dimension one, it is clear that we can interpret such a formal sum as a (possibly interrupted) path by regarding the union of the 1-simplices in the sum. The signs in this sum determine in what direction we traverse the 1-simplex. We now want to express when such a path is a loop, and we do this in general for dimension n.

Definition 2.2.11. *For each n we have a group morphism*

$$d_n : C_n \to C_{n-1}$$

that sends each n-simplex to the alternating sum of its facets (and is extended linearly to all elements of C_n). We can write this down as follows: let σ be an n-simplex of K, and label the elements according to the total order on σ, i.e. $\sigma = \{v_0, \cdots, v_n\}$. We then define:

$$d_n(\sigma) = \sum_{i=0}^{n} (-1)^i \{v_0, \cdots, v_{i-1}, v_{i+1}, \cdots, v_n\}$$

This group morphism is called the ***boundary morphism*** *of C_n*

Thinking about dimension one again, we see that the boundary of a line consists of two points with different sign. We can for example call the positive point the starting point and the other point the endpoint. When adding two lines of which the endpoint of the first line is equal to the starting point of the second line, we see that this point gets cancelled in the boundary of the sum of the two lines, as the point appears twice with different sign. This is indeed wanted, as this point is not in the intuitive boundary of the union of the two lines. The same reasoning shows that if we have a loop of lines, each endpoint being equal to the starting point of another line, all points will be cancelled in the boundary so the loop gets sent to 0 by the boundary morphism. Indeed, a loop has no boundary. This leads to the following definition:

Definition 2.2.12. *The elements of the kernel of d_n are called* ***cycles*** *(the n-dimensional equivalent of loops), and the elements of the image of d_n are called* ***boundaries****.*

Intuitively we see that every boundary is a cycle, i.e. $d_{n-1} \circ d_n = 0$ or $\mathrm{im}(d_n) \subseteq \ker(d_{n-1})$ for all n. This can also be computed; see proposition 2.2.33. A chain of group morphisms

$$\cdots \xrightarrow{d_{n+1}} C_n \xrightarrow{d_n} C_{n-1} \xrightarrow{d_{n-1}} \cdots \xrightarrow{d_2} C_1 \xrightarrow{d_1} C_0 \xrightarrow{0} 0 \qquad (2.1)$$

with this property is called a *chain complex*. In case of equality $\mathrm{im}(d_n) = \ker(d_{n-1})$ for all n, a chain complex is called an *exact sequence*. Notice that in this case, this means that every cycle is a boundary, or intuitively that there are no holes. If the chain complex is not exact, there are cycles that are not boundaries, or holes, and they can be represented by the following quotient.

Definition 2.2.13. *The **simplicial homology groups** $H_n(K)$ of K are given by*

$$H_n(K) = \ker(d_n)/\operatorname{im}(d_{n+1})$$

In the following examples, all simplicial complexes are ordered by the usual order of natural numbers.

Example 2.2.14. *In this example we compare a hollow triangle with a full triangle (figure 2.2 and example 2.2.7). Observe the hollow triangle*

$$K_1 =\downarrow \{\{0,1\},\{1,2\},\{0,2\}\}$$

The element $a = \{0,1\} + \{1,2\} - \{0,2\}$ of C_1 is clearly in the kernel of d_1. It follows that $\mathbb{Z}a \cong \mathbb{Z}$ is inside the kernel, and we can check that this is the entire kernel. This image of d_2 is 0 as $C_2 = \{0\}$. The homology group of dimension one is

$$H_1(K_1) = \ker(d_1)/\operatorname{im}(d_2) = \mathbb{Z}a/0 \cong \mathbb{Z}^1$$

which means that there is one hole of dimension one. We can verify this computation in Sage [22]:

```
sage:  K1 = SimplicialComplex([[0,1],[1,2],[0,2]])
sage:  K1.homology(1)
Z
```

The output is indeed $\mathbb{Z}$.

Now observe the full triangle

$$K_2 = K_1 \cup \{\{0,1,2\}\} =\downarrow \{\{0,1,2\}\}$$

The kernel of d_1 is still $\mathbb{Z}a \cong \mathbb{Z}$, but C_2 is no longer the zero group. It now contains one generator: $b = \{0,1,2\}$. The image of this generator under the boundary morphism is

$$d_2(b) = \{1,2\} - \{0,2\} + \{0,1\} = a$$

We find $\operatorname{im}(C_2) = \mathbb{Z}a \cong \mathbb{Z}$, so

$$H_1(K_2) = \ker(d_1)/\operatorname{im}(d_2) = \mathbb{Z}a/\mathbb{Z}a \cong 0$$

so there are no more holes. Indeed, the only hole we had was filled in by the triangle $\{0,1,2\}$. We verify the result via sage:

```
sage:  K2 = SimplicialComplex([[0,1,2]])
sage:  K2.homology(1)
0
```

The output is indeed 0.

Example 2.2.15. *We now compute the one dimensional homology group of example 2.2.8, i.e. the simplicial complex*

$$K_3 =\downarrow \{\{0,1,2\},\{2,3\},\{1,3\}\}$$

Now we have two independent generators of the kernel of d_1*, namely* $a = \{0,2\}+\{2,3\}-\{1,3\}-\{0,1\}$ *and* $b = \{2,3\} - \{1,3\} + \{1,2\}$*. The image of* d_2 *is* $\mathbb{Z}c$ *with* $c = \{1,2\} - \{0,2\} + \{0,1\}$ *(the image of the only* 2*-simplex). As* $b - a = c$*, we have that* a *and* b *are equal in the quotient, so we find*

$$H_1(K_3) = \ker(d_1)/\operatorname{im}(d_2) = \frac{\mathbb{Z}a \oplus \mathbb{Z}b}{\mathbb{Z}c} \cong \mathbb{Z}^1$$

and the homology group is generated by $[a] = [b]$*. So while we have two loops that aren't boundaries, we only count one hole, as expected. In the quotient these two loops become equal, because their difference is a boundary. This is conceptually very close to homotopy, where two loops become equal if they can be continuously deformed into each other. Indeed, these two loops could be continuously deformed into each other over the simplex* $\{0,1,2\}$ *of which the boundary is the difference of the two loops. We verify the result via sage:*

```
sage:  K3 = SimplicialComplex([[0,1,2], [2,3], [1,3]])
sage:  K3.homology(1)
Z
```

The output is indeed $\mathbb{Z}$*.*

2.2.3 Simplicial Sets and Simplicial Objects

We will now introduce some generalisations of simplicial complexes. These are not fundamental to the study of finite topological spaces—everything can be done with simplicial complexes—but they will help to give cleaner definitions of important concepts like geometric realisation and singular homology. This last concept is a version of homology which is valid for any topological space (so also finite topological spaces) and it agrees with simplicial homology when the space is homeomorphic to a simplicial complex.

The first obvious generalisation is obtained by only keeping the information that is used in the chain complex (2.1). This is, for each dimension n:

- A set S_n of n-simplices. These generate the group C_n.
- The maps $\partial_i : \{v_0, \cdots, v_n\} \mapsto \{v_0, \cdots, v_{i-1}, v_{i+1}, \cdots, v_n\}$, that generate the boundary morphism d_n.

- The property $\forall i < j : \partial_i \circ \partial_j = \partial_{j-1} \circ \partial_i$, which implies that $d_{n-1} \circ d_n = 0$ (proposition 2.2.33).

We can abstract this to obtain the definition of a *semi-simplicial set* (which are called Δ-complexes in [11]).

Definition 2.2.16. *A* ***semi-simplicial set*** *S is a collection of sets $(S_n)_{n\in\mathbb{N}}$ along with maps $\partial_i : S_n \to S_{n-1}$ for $i = 0, \cdots, n$, that satisfy*

$$\partial_i \circ \partial_j = \partial_{j-1} \circ \partial_i, \text{ if } i < j$$

These maps are called ***face maps****.*

There is an equivalent high level characterisation that uses the following category:

Definition 2.2.17. *The* ***simplex category*** *is the category whose objects are the totally ordered sets*

$$[n] = \{0, 1, \cdots, n\}$$

(the abstract representation of an oriented n-simplex) and whose morphisms are the order preserving maps. We denote this category Δ. The subcategory with the same objects, but only the injective order preserving maps as morphisms is denoted Δ_+.

Proposition 2.2.18. *A semi-simplicial set induces, and is completely determined by, a contravariant functor from Δ_+ to* **Set***, i.e. a functor $\Delta_+^{op} \to$* **Set***.*

Proof. A functor $\Delta_+^{\mathrm{op}} \to$ **Set** clearly induces a collection of sets $(S_n)_{n\in\mathbb{N}}$, as such a collection is nothing but a map $\mathbb{N} \to$ **Set** and the objects of Δ_+ can be identified with the natural numbers by $n \mapsto [n]$. The images of $\mathrm{Hom}_{\Delta_+}([n-1], [n])$ through the functor determine valid face maps, as the elements of $\mathrm{Hom}_{\Delta_+}([n-1], [n])$ are of the form

$$\varepsilon_i : [n-1] \to [n] : j \mapsto \begin{cases} j & \text{if } j < i \\ i+1 & \text{if } j \geq i \end{cases}$$

with $i \in \{0, 1, \cdots, n\}$. In other words, they are natural injections up to the fact that they skip one element in the image. This implies that they satisfy

$$\varepsilon_j \circ \varepsilon_i = \varepsilon_i \circ \varepsilon_{j-1}, \text{ if } i < j$$

which in turn implies that their images under the contravariant functor satisfy the desired property. The same identification in the other direction gives a partially defined functor $\Delta^+ \to$ **Set**. Only the images of the morphisms ε_i (the elements of $\mathrm{Hom}_{\Delta_+}([n-1], [n])$ for all n) are given. However, this information completely determines the functor. This follows from the fact that every injective morphism has a unique factorisation in morphisms of the form ε_i. In other words, the ε_i generate al morphisms of Δ_+ under $\circ$. As a functor preserves $\circ$, the images of all morphisms are determined. □

In this high level characterisation, we can easily replace the category **Set** by an arbitrary category.

Definition 2.2.19. *A* ***semi-simplicial object*** *is a contravariant functor from Δ_+ to an arbitrary category $\mathcal{C}$, i.e. a functor $\Delta_+^{op} \to \mathcal{C}$.*

The concept of *simplicial sets* is obtained by regarding contravariant functors on the entire category Δ.

Definition 2.2.20. *A* ***simplicial set*** *is a contravariant functor from Δ to* **Set***, i.e. a functor $\Delta^{op} \to$* **Set**.

Simplicial sets also have a low level combinatorial description, similar to that of semi-simplicial sets. Here, we have an extra type of maps, in addition to the face maps. These new maps are called *degeneracy maps* because they send an n-simplex into the set of $(n+1)$-simplices. For example, a triangle can always be seen as a degenerate (squashed) tetrahedron.

Proposition 2.2.21. *A simplicial set S induces, and is completely determined by, a collection of sets $(S_n)_{n\in\mathbb{N}}$ along with maps $\partial_i : S_n \to S_{n-1}$ and maps $\sigma_i : S_n \to S_{n+1}$ for all $i = 0, \cdots, n$ that satisfy*

$$\partial_i \circ \partial_j = \partial_{j-1} \circ \partial_i, \text{ if } i < j$$

$$\sigma_i \circ \sigma_j = \sigma_{j+1} \circ \sigma_i, \text{ if } i \leq j$$

$$\sigma_i \circ \partial_j = \begin{cases} \sigma_{j-1} \circ \partial_i, & \text{if } i < j \\ \mathrm{id}, & \text{if } i = j \text{ or } i = j+1 \\ \sigma_j \circ \partial_{i-1}, & \text{if } i > j+1 \end{cases}$$

The maps ∂_i are called ***face maps*** *and the maps σ_i are called* ***degeneracy maps***.

Proof. The proof is analogous to the proof of proposition 2.2.18. A functor $\Delta^{\mathrm{op}} \to$ **Set** again determines a collection of sets $(S_n)_{n\in\mathbb{N}}$, as we can identify the objects of Δ with the natural numbers via $n \mapsto [n]$. The images of $\mathrm{Hom}_\Delta([n-1],[n])$ and $\mathrm{Hom}_\Delta([n+1],[n])$ through the functor determine valid face and degeneracy maps, as the elements of $\mathrm{Hom}_\Delta([n-1],[n])$ are of the form

$$\varepsilon_i : [n-1] \to [n] : j \mapsto \begin{cases} j & \text{if } j < i \\ j+1 & \text{if } j \geq i \end{cases}$$

and the elements of $\mathrm{Hom}_\Delta([n+1],[n])$ are of the form

$$\eta_i : [n+1] \to [n] : j \mapsto \begin{cases} j & \text{if } j \leq i \\ j-1 & \text{if } j > i \end{cases}$$

so they satisfy

$$\varepsilon_j \circ \varepsilon_i = \varepsilon_i \circ \varepsilon_{j-1}, \text{ if } i < j$$
$$\eta_j \circ \eta_i = \eta_i \circ \eta_{j+1}, \text{ if } i \leq j$$
$$\eta_j \circ \varepsilon_i = \begin{cases} \varepsilon_i \circ \eta_{j-1}, & \text{if } i < j \\ \text{id}, & \text{if } i = j \text{ of } i = j+1 \\ \varepsilon_{i-1} \circ \eta_j, & \text{if } i > j+1 \end{cases}$$

and their images through the contravariant functor satisfy the desired properties.

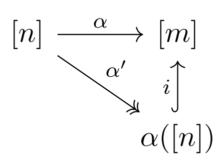
The other direction is also analogous. Every injective morphism in Δ has a unique decomposition in elements of the form ε_i and every surjective morphism has a unique decomposition in elements of the form η_i. Together with the fact that every morphism in Δ has a unique epi-mono factorisation (lemma 2.2.22), this implies that every morphism in Δ has a unique decomposition in morphisms of the above form. So if we know the face and degeneracy maps, we have completely determined the functor. □

Lemma 2.2.22 (epi-mono factorisation in Δ)**.** *Let $\alpha : [n] \to [m]$ be a morphism in the simplex category. Then*

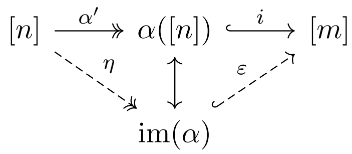
$$\alpha = \varepsilon \circ \eta : [n] \to [p] \to [m]$$

with η surjective and ε injective in a unique way. We denote $\mathrm{im}(\alpha) := [p]$.

Proof. In the supercategory of totally ordered sets, this claim is trivially true, as we can make use of the (set theoretic) image:



but $\alpha([n]) \subseteq [m]$ is usually not an object in the simplex category. To get a factorisation in Δ, we construct an element in the simplex category that is order isomorphic to $\alpha([n])$ (which is clearly unique because the cardinality completely determines the object). This then provides the desired (unique) factorisation as follows:



To construct the object $\mathrm{im}(\alpha)$ we make use of the Mostowski collapse in the next lemma. □

The Mostwoski collapse was introduced in [17] (Theorem 3). Here we present a simplified version which suffices in our case.

Lemma 2.2.23 (Mostowski collapse)**.** *Let $x \subseteq [n] \in \Delta$. Then there exists an order isomorphism $f : x \to f(x) \in \Delta$.*

Proof. We define f by the recursion

$$f(i) := \min([n] \setminus f(\{j \in x \mid j < i\}))$$

We clearly have that f is injective, because if $i \neq j$, then w.l.o.g. $j < i$ so $f(j)$ is not in the set of which $f(i)$ is the minimum. We also have that f is surjective as the codomain is $f(x)$. Since we are working with total orders, a bijective order morphism is an order isomorphism. We now check that f is indeed an order morphism:

$$\begin{aligned} k \leq l &\implies \{j \in x \mid j \leq k\} \subseteq \{j \in x \mid j \leq l\} \\ &\implies f(\{j \in x \mid j \leq k\}) \subseteq f(\{j \in x \mid j \leq l\}) \\ &\implies [n] \setminus f(\{j \in x \mid j \leq k\}) \supseteq [n] \setminus f(\{j \in x \mid j \leq l\}) \\ &\implies \min([n] \setminus f(\{j \in x \mid j \leq k\})) \leq \min([n] \setminus f(\{j \in x \mid j \leq l\})) \\ &\implies f(k) \leq f(l) \end{aligned}$$

so x is indeed order isomorphic with $f(x)$. We still need to check that $f(x) \in \Delta$, or in other words, that $f(x) = [n]$ with $n = \max(f(x))$, or equivalently

$$\forall i \in \mathbb{N} : i \in f(x) \iff i \in [n]$$

which follows from an easy induction on i. □

We can again replace the category **Set** by an arbitrary category in definition 2.2.20. This yields the concept of simplicial objects.

Definition 2.2.24. *A **simplicial object** is a contravariant functor from Δ to an arbitrary category $\mathcal{C}$, i.e. a functor $\Delta^{op} \to \mathcal{C}$.*

Just like simplicial complexes, we can interpret simplicial sets as topological spaces. The elements of the sets represent the simplices and the face maps give instructions on how these need to be glued together. The degeneracy maps tell which simplices need to be squashed down. We will now make this construction explicit. This basically involves making a lot of choices. For example, we need to choose a geometric simplex for every abstract simplex in the simplicial set. For this purpose, we introduce a standard simplex for every n. We will use the following notation

$$B_n = \left\{ \begin{array}{c} (1,0,\cdots,0), \\ (0,1,\cdots,0), \\ \vdots \\ (0,0,\cdots,1) \end{array} \right\}$$

for the canonical basis of $\mathbb{R}^n$.

Definition 2.2.25. *The* ***standard*** ***n-simplex****, denoted Δ^n, is the convex hull of B_{n+1} in $\mathbb{R}^{n+1}$. In other words*

$$\Delta^n := \{(t_0,\cdots,t_n) \mid 0 \le t_i \le 1, \sum_{i=0}^{n} t_i = 1\}$$

Remember that an element $[n] \in \Delta$ of the simplex category represents an abstract n-simplex. We can interpret it as the set of vertices of a geometric n-simplex, so it is not too far-fetched to identify them with the vertices of the standard simplex. This is done via

$$[n] \to B_{n+1} : i \mapsto \left(\begin{array}{ll} 1 & \text{if } i = j \\ 0 & \text{if } i \neq j \end{array} \right)_{j \in [n]}$$

Through this identification, the morphisms of Δ are maps on the vertices of the standard simplices. We can extend these maps linearly to the entire standard simplex, which makes the maps continuous. In category theory language, this yields a functor

$$f : \Delta \to \textbf{top}$$

that maps $[n]$ to the standard n-simplex, and acts on the morphisms as described above. This is almost a simplicial topological space, but it is not since it is a covariant functor and not a contravariant functor. We call such a functor a *cosimplicial topological space*. More generally we define:

Definition 2.2.26. *A* ***cosimplicial object*** *is a covariant functor from Δ to an arbitrary category $\mathcal{C}$, i.e. a functor $\Delta \to \mathcal{C}$.*

The functor f sends the maps ε_i to the inclusion map $\partial^i : \Delta^{n-1} \to \Delta^n$, where Δ^{n-1} is interpreted as the i-th facet of Δ^n. The maps η_i are sent to the projection $\sigma^i : \Delta^{n+1} \to \Delta^n$ on the i-th facet. Notice that we write the indices in ∂^i and σ^i on top. These maps allow us to give an explicit description of the geometric realisation of a simplicial set.

Definition 2.2.27. *Let $S : \Delta^{op} \to$ **Set** be a simplicial set. The **geometric realisation** $|S|$ of S is the topological space obtained by the following procedure. For each n we take the product*

$$S_n \times \Delta^n$$

where S_n is seen as a discrete topological space. This means that this product is a disjoint union of copies of Δ^n, one for each element of S_n. We now take the disjoint union of all those simplices together:

$$\coprod_{n \in \mathbb{N}} S_n \times \Delta^n$$

Finally, we define $|S|$ as the quotient

$$\left(\coprod_{n \in \mathbb{N}} S_n \times \Delta^n \right) / \sim$$

where $\sim$ is the smallest equivalence relation that satisfies

$$\begin{aligned} (x,t) \sim (y,s) \text{ if } \partial_i(x) = y \text{ and } \partial^i(s) = t \\ \text{or } \sigma_i(x) = y \text{ and } \sigma^i(s) = t \end{aligned}$$

Geometric realisation defines a function

$$|\cdot| : \mathcal{S}\mathbf{Set} \to \mathbf{Top}$$

from the class of simplicial sets to the class of topological spaces. This function can be extended to a functor, but then we first need to turn the class $\mathcal{S}$**Set** into a category, by introducing the morphisms.

Definition 2.2.28. *Given a category $\mathcal{C}$, we can define the **category of simplicial objects over** $\mathcal{C}$, denoted $\mathcal{SC}$. The objects of this category are the simplicial objects $\Delta^{op} \to \mathcal{C}$ and the morphisms are the natural transformations between these functors. In other words we put*

$$\mathcal{SC} := \mathcal{C}^{\Delta^{op}}$$

More explicitly, a morphism f from a simplicial object S to a simplicial object T is a collection of morphisms (in $\mathcal{C}$) $f_n : S_n \to T_n$ for every $n \in \mathbb{N}$ that commute with the face and degeneracy maps, i.e. the following diagram commutes for every n and for every i:

$$\begin{array}{ccc} S_n & \xrightarrow{\partial_i} & S_{n-1} \\ \downarrow{\scriptstyle f_n} & & \downarrow{\scriptstyle f_{n-1}} \\ T_n & \xrightarrow{\partial_i} & T_{n-1} \end{array} \qquad \begin{array}{ccc} S_n & \xrightarrow{\sigma_i} & S_{n+1} \\ \downarrow{\scriptstyle f_n} & & \downarrow{\scriptstyle f_{n+1}} \\ T_n & \xrightarrow{\sigma_i} & T_{n+1} \end{array}$$

This is sufficient because the maps ε_i and η_i generate all morphisms of Δ.

Definition 2.2.29. *The morphisms of the category $\mathcal{S}$**Set** are called **simplicial maps**.*

To extend the geometric realisation $|\cdot|$ to a functor, we need to define the action on the morphisms. This is done as follows: let $f : S \to T$ be a simplicial map. We then define

$$|f| : |S| \to |T| : [(x_n, t_n)] \mapsto [f_n(x_n), t_n]$$

with $(x_n, t_n) \in S_n^n$. This will always be a continuous map and this action turns

$$|\cdot| : \mathcal{S}\mathbf{Set} \to \mathbf{Top}$$

into a functor.

In the previous section we used free abelian groups in the definition of simplicial homology. A free abelian group is the same as a free $\mathbb{Z}$-module. There is no reason why we should only consider $\mathbb{Z}$-modules, so from now on we work with an arbitrary ring. For any ring R there is a functor

$$R[\cdot] : \mathbf{Set} \to \mathbf{R\text{-}mod} : S \mapsto \bigoplus_{x \in S} R$$

that maps a set S to the free R-module over S. When composing this functor with a simplicial set (a functor $\Delta^{\mathrm{op}} \to \mathbf{Set}$), we obtain a functor $\Delta^{\mathrm{op}} \to \mathbf{R\text{-}mod}$. This turns any simplicial set into a simplicial R-module. We can introduce simplicial homology for simplicial sets, using this simplicial R-module.

Definition 2.2.30. *Let $S : \Delta^{op} \to$ **Set** be a simplicial set, let R be a ring and take the simplicial R-module $R[S] : \Delta^{op} \to$ **R-mod** of the above discussion. This simplicial R-module gives rise to the following chain complex*

$$\cdots \xrightarrow{d_{n+1}} R[S]_n \xrightarrow{d_n} R[S]_{n-1} \xrightarrow{d_{n-1}} \cdots \xrightarrow{d_2} R[S]_1 \xrightarrow{d_1} R[S]_0 \xrightarrow{0} 0 \tag{2.2}$$

where

$$d_n = \sum_{i=0}^{n} (-1)^i \partial_i$$

*The claim that this is indeed a chain complex is verified in proposition 2.2.33. The **simplicial homology modules of** S **with coefficients in** R, denoted $H_n(S; R)$, are given by*

$$H_n(S; R) = \ker(d_n) / \operatorname{im}(d_{n+1})$$

*for each $n \in \mathbb{N}$. We often just refer to them as the **simplicial homology groups of** S **with coefficients in** R, as modules are also groups. If $R = \mathbb{Z}$, we often omit specifying R and just write:*

$$H_n(S) = H_n(S; \mathbb{Z})$$

Definition 2.2.31. *Let $S : \Delta^{op} \to$ **Set** be a simplicial set. For each $n \in \mathbb{N}$, the n-th **Betti number** is defined as the rank of $H_n(S)$, i.e. the number of linearly independent generators of the n-th homology group with integer coefficients.*

Definition 2.2.32. *Let $S : \Delta^{op} \to$ **Set** be a simplicial set and let F be a field. For each $n \in \mathbb{N}$, the n-th **Betti number with coefficients in** F is defined as the dimension of $H_n(S; F)$.*

Proposition 2.2.33. *The chains of maps* (2.1) *and* (2.2) *are chain complexes, i.e. $d_n \circ d_{n+1} = 0$ for all n.*

Proof. As said earlier, this is a consequence of the fact that

$$\partial_i \circ \partial_j = \partial_{j-1} \circ \partial_i, \text{ als } i < j$$

which holds for simplicial sets and simplicial complexes. Indeed,

$$\begin{aligned} d_n \circ d_{n+1} &= \sum_{i,j} (-1)^{i+j} \partial_i \circ \partial_j \\ &= \sum_{i \geq j} (-1)^{i+j} \partial_i \circ \partial_j + \sum_{i<j} (-1)^{i+j} \partial_i \circ \partial_j \\ &= \sum_{i \geq j} (-1)^{i+j} \partial_i \circ \partial_j + \sum_{i<j} (-1)^{i+j} \partial_{j-1} \circ \partial_i \\ &= \sum_{i \geq j} (-1)^{i+j} \partial_i \circ \partial_j - \sum_{i \leq k} (-1)^{i+k} \partial_k \circ \partial_i \\ &= 0 \end{aligned}$$

□

Example 2.2.34 (ordered simplicial complexes as simplicial sets)**.** *As discussed earlier, we can represent ordered simplicial complexes as semi-simplicial sets. The set S_n is given by the set of n-simplices and the face maps are given by*

$$\partial_i : \{v_0, \cdots, v_n\} \mapsto \{v_0, \cdots, v_{i-1}, v_{i+1}, \cdots, v_n\}$$

To complete this semi-simplicial set and obtain a simplicial set, we use the following degeneracy maps

$$\sigma_i : \{v_0, \cdots, v_n\} \mapsto (v_0, \cdots, v_i, v_i, \cdots, v_n)$$

which send an n-simplex to the degenerate $(n+1)$-simplex that repeats the i-th point. These degenerate simplices need to be added to the sets S_n.

Example 2.2.35 (singular simplicial set)**.** *Simplicial complexes only allowed us to define homology for very specific topological spaces, namely those spaces which appear as geometric realisation of a* *simplicial complex (and these are always infinite or discrete). There exists a definition of homology which is valid for every topological space (also finite ones!) and agrees with the simplicial homology if applicable, which is called singular homology [11, 13, 19]. This concept can be easily obtained using simplicial sets. Every topological space X can be regarded as a simplicial set called the singular simplicial set of the space. The set of n-simplices is the set of continuous maps from the standard n-simplex to X. The set of 1-simplices for example is the set of paths in X. The face and degeneracy maps are induced by the face and degeneracy maps of the standard simplices.*

A more precise and high level description of this simplicial set is the following. Remember that the standard simplices yield a cosimplicial topological space, i.e. a functor $\Delta \to$ **Top**. *We can compose this functor with the contravariant functor*

$$\mathrm{Hom}_{\mathbf{Top}}(\cdot, X) : \mathbf{Top} \to \mathbf{Set}$$

to obtain a contravariant functor from Δ to **Set***, i.e. a simplicial set.*

Definition 2.2.36. *Let X be a topological space. For each $n \in \mathbb{N}$, the n-th homology group of the singular simplicial set of X is called the n-th* ***singular homology group of*** *X. We sometimes just call it the n-th homology group of X and we will denote it by $H_n(X; R)$ or $H_n(X)$ if $R = \mathbb{Z}$.*

We recall two fundamental theorems of singular homology, which can be found in, for example [11] (Theorems 2.27 and 4.21 respectively). The first theorem states that singular homology is an extension of simplicial homology, i.e. for al topological spaces for which simplicial homology is defined, it agrees with singular homology:

Proposition 2.2.37. *Let X be a topological space that can be triangulated, i.e. there is a simplicial complex K such that $|K|$. Then for all $n \in \mathbb{N}$:*

$$H_n(K; R) \cong H_n(X; R)$$

In other words, the n-th simplicial homology group of X is isomorphic to the n-th singular homology group of X.

The second theorem tells us that singular homology is a coarser invariant than weak homotopy types, or that weak homotopy equivalences not only yield isomorphic homotopy groups, but also isomorphic singular homology groups.

Proposition 2.2.38. *Let $f : X \to Y$ be a weak homotopy equivalence, then f induces isomorphisms*

$$f_* : H_n(X; R) \to H_n(Y; R)$$

on all singular homology groups.

Chapter 3

Finite Topological Spaces in Algebraic Topology

3.1 Introduction to Finite Topological Spaces

Finite topological spaces are simply topological spaces of which the underlying set is finite. These spaces are usually considered highly pathological. Not only because they are finite, but also because almost all finite topological spaces are very poorly separated. More precisely, a finite topological space is T_1 if and only if it is discrete. All other finite topological spaces are less separated than T_1. In that regard, the strongest (and only) separation property of interest in this thesis will be the T_0 property. Seeing that the spaces usually of interest in analysis are T_2 (Hausdorff) or better, this explains why finite spaces are considered pathological.

In algebraic topology however, separation (like other local properties) is actually very unimportant. It is the global shape that is important here. Finite topological spaces are actually surprisingly diverse in that regard, as will be shown section 3.4. In this section, we will connect finite topological spaces to finite pre-orderd sets and posets. This can be attributed to Alexandroff [1]. More modern expositions can be found in all texts that build on it, for example [2, 14, 21, 16]

An important property of finite spaces to keep in mind is that arbitrary intersections of open sets are still open (as such an intersection is always finite). This makes life much easier. It is for example possible to define a minimal neighbourhood for each point. These will be used throughout the thesis.

Definition 3.1.1. *Let X be a a finite topological space. The* ***minimal open neighbourhood*** *of $x \in X$ is the intersection of all neighbourhoods of x. It is denoted U_x. Notice that it is indeed a neighbourhood as it* *is a finite intersection of neighbourhoods (because X is finite). It is also open because any neighbourhood contains an open neighbourhood, but U_x is minimal.*

For an infinite topological space we also write U_x for the intersection of all neighbourhoods of x, but this is in general no longer a neighbourhood.

3.1.1 The Specialisation Pre-Order

Any topological space comes naturally equipped with a certain pre-order called the *specialisation pre-order*. It can be defined in many equivalent ways.

Proposition 3.1.2. *Let X be a topological space and let $x, y \in X$. The following statements are equivalent*

1. $U_x \subseteq U_y$
2. $x \in U_y$
3. $y \in \overline{\{x\}}$
4. $\overline{\{y\}} \subseteq \overline{\{x\}}$

Proof.

(1 $\iff$ 2) 1 trivially implies 2. If 2 holds then x is an element of each neighbourhood of y, in particular x is also an element of the interior of each neighbourhood of y (as this is a neighbourhood itself). This implies that each neighbourhood of y is also a neighbourhood of x. Then we have $U_x \subseteq U_y$ as U_x is an intersection of more sets.

(2 $\iff$ 3) We show that

$$x \notin U_y \iff y \notin \overline{\{x\}}$$

$y \notin \overline{\{x\}}$ if and only if y has a neighbourhood that does not contain x if and only if $x \notin U_y$.

(3 $\iff$ 4) 4 trivially implies 3. $\overline{\{x\}}$ is closed, so if $y \in \overline{\{x\}}$, the closure of y is contained in it.

□

Definition 3.1.3. *Let $(X, \mathcal{T})$ be a topological space. The* **specialisation pre-order** *on X is given by*

$$x \leq_{\mathcal{T}} y \iff U_x \subseteq U_y$$

or any of the equivalent statements in proposition 3.1.2. We will denote it by $\leq_{\mathcal{T}}$ or by $\leq_X$ if it is clear which topology we use on X, or by $\leq$ if it is clear on which topological space we are working.

This is a pre-order and not necessarily a partial order, as exemplified by the trivial topology.

Example 3.1.4. *The specialisation order of a topological space X with the trivial topology is given by $X \times X$, i.e.*

$$\forall x, y \in X : x \leq_X y$$

Indeed, $\forall x, y \in X : U_x = X = U_y$. This is clearly not anti-symmetric.

Seeing that the specialisation pre-order depends only on the topology of a space, any order theoretic property of it, is actually a topological property. One could wonder what topological property corresponds to anti-symmetry of the specialisation pre-order. This has a rather satisfying answer.

Proposition 3.1.5. *Let X be a topological space. The specialisation order on X is anti-symmetric (equivalently, $(X, \leq_X)$ is a poset) if and only if X is T_0.*

Proof. The specialisation order on X is anti-symmetric if and only if

$$x \leq_X y \text{ and } y \leq_X x \implies x = y$$

if and only if

$$U_x = U_y \implies x = y$$

if and only if

$$x \neq y \implies U_x \neq U_y$$

Now assume X is T_0, then for every $x \neq y$ we have w.l.o.g. a neighbourhood of x that does not contain y. This implies that $y \notin U_x$ but $y \in U_y$ so $U_x \neq U_y$.
Next, assume that X is anti-symmetric. Take $x \neq y$, then we know $U_x \neq U_y$. If $x \in U_y$ and $y \in U_x$, then $U_x \subseteq U_y$ and $U_y \subseteq U_x$ according to proposition 3.1.2. This is not the case as $U_x \neq U_y$, so either $x \notin U_y$ and $y \notin U_x$. Assume w.l.o.g. that $x \notin U_y$, then there is a neighbourhood of y that does not contain x. We conclude that X is T_0. □

Up until now, we said that the specialisation pre-order is defined for any topological space (which it is), but it is only interesting for very poorly separated spaces, like finite ones. This is what the next proposition shows.

Proposition 3.1.6. *Let X be a T_1 topological space. Then the specialisation order on X is*

$$\forall x, y \in X : x \leq_X y \iff x = y$$

Proof. This follows immediately from the fact that singletons are closed in a T_1 topological space. Indeed:

$$x \leq_X y \iff y \in \overline{\{x\}} \iff y \in \{x\} \iff x = y$$

□

Equipping a space with its specialisation pre-order can be seen as a function from the class of topological spaces to the class of pre-ordered sets. As it keeps the underlying sets fixed, it automatically also acts on functions between topological spaces by sending functions

$$f : (X, \mathcal{T}) \to (Y, \mathcal{S})$$

to the same function, but between pre-ordered sets:

$$f : (X, \leq_{\mathcal{T}}) \to (Y, \leq_{\mathcal{S}})$$

Something nice happens here: continuous functions get send to order preserving functions, as the following proposition shows:

Proposition 3.1.7. *Let X and Y be topological spaces, and let $f : X \to Y$ be a continuous map. Then $f : (X, \leq_X) \to (Y, \leq_Y)$ is order preserving.*

Proof. Recall the characterisation of continuity in terms of the closure operator: the function f is continuous if and only if

$$\forall A \subseteq X : f(\overline{A}) \subseteq \overline{f(A)}$$

Take $x \leq_X y$ in X, then $y \in \overline{\{x\}}$. By continuity of f we then have

$$f(y) \in f(\overline{\{x\}}) \subseteq \overline{f(\{x\})} = \overline{\{f(x)\}}$$

or in other words $f(x) \leq_Y f(y)$. □

This means that the specialisation order actually describes a concrete functor from the category of topological spaces to the category of pre-ordered sets:

$$P : \textbf{Top} \to \textbf{PreOrd} : (X, \mathcal{T}) \mapsto (X, \leq_{\mathcal{T}})$$

3.1.2 The Order Topology

We will now see that we can also go the other way around. On any pre-ordered set, there is a natural topology called the order topology.

Definition 3.1.8. *Let $(X, \leq)$ be a pre-ordered set. The* ***down set*** *of $x \in X$ is the set*

$$\{y \in X \mid y \leq x\}$$

It is denoted $\downarrow x$.

Proposition 3.1.9. *Let $(X, \leq)$ be a pre-ordered set. The set $\{\downarrow x \mid x \in X\}$ is a basis for a topology $\mathcal{T}_{\leq}$ on X.*

Proof. We have to show that

$$\mathcal{T}_{\leq} = \left\{ \bigcup_{x \in A} \downarrow x \mid A \subseteq X \right\}$$

is a topology on X. We already have that $\mathcal{T}_{\leq}$ is closed under taking unions, by construction. By choosing $A = X$ and $A = \emptyset$ we see that X and the empty set are in $\mathcal{T}_{\leq}$. We still need to check that $\mathcal{T}_{\leq}$ is closed under taking finite intersections. Since we have that

$$\bigcup_{x \in A} \downarrow x \cap \bigcup_{y \in B} \downarrow y = \bigcup_{x \in A,\, y \in B} \downarrow x \cap \downarrow y$$

it is sufficient to check that $\downarrow x \cap \downarrow y \in \mathcal{T}_{\leq}$ for every $x, y \in X$. Therefore, it is sufficient to show that

$$\downarrow x \cap \downarrow y = \bigcup_{z \in \downarrow x \cap \downarrow y} \downarrow z$$

The inclusion $\subseteq$ is an immediate consequence of the reflexivity of $\leq$. For the inclusion $\supseteq$, let $w \in \bigcup_{z \in \downarrow x \cap \downarrow y} \downarrow z$. Then there is a $v \in \downarrow x \cap \downarrow y$ such that $w \in \downarrow v$, so $w \leq v$. Because $v \in \downarrow x$ we have that $v \leq x$, so by transitivity $w \leq x$. Therefore $w \in \downarrow x$. Analogously we find that $w \in \downarrow y$, so we can conclude that $w \in \downarrow x \cap \downarrow y$. □

Definition 3.1.10. *Let $(X, \leq)$ be a pre-ordered set. The topology $\mathcal{T}_{\leq}$ of property 3.1.9 is called the* ***order topology*** *on $(X, \leq)$.*

This construction is valid in the infinite and finite case, and notably, in the infinite case, the resulting spaces have one of the key properties of finite spaces:

Proposition 3.1.11. *Let $(X, \leq)$ be a pre-ordered set. In the space $(X, \mathcal{T}_{\leq})$ the intersection of any family of open sets is open. In particular, the set U_x is an open neighbourhood of any $x \in X$.*

Proof. The proof is completely analogous to the proof that finite intersections of open sets are open in proposition 3.1.9. □

Similarly to the situation with the specialisation pre-order, this construction can be seen as a function from the class of pro-ordered sets to the class of topological spaces. And again, it sends morphisms to morphisms (this time: order-preserving maps to continuous maps):

Proposition 3.1.12. *Let $(X, \leq_X)$ and $(Y, \leq_Y)$ be pre-ordered sets, and let $f : (X, \leq_X) \to (Y, \leq_Y)$ be an order preserving map. Then $f : (X, \mathcal{T}_X) \to (Y, \mathcal{T}_Y)$ is continuous.*

Proof. Take $G \in \mathcal{T}_Y$. We have to show that $f^{-1}(G)$ is open. We shall show this by showing that

$$\forall x \in f^{-1}(G) : U_x \subseteq f^{-1}(G)$$

so take $x \in f^{-1}(G)$. The set G is open and it contains $f(x)$ so $U_{f(x)} \subseteq G$. For any $y \in U_x$ we have $y \leq_X x$, so by order preservation $f(y) \leq_Y f(x)$. This implies $f(y) \in U_{f(x)} \subseteq G$. We can conclude that $U_x \subseteq f^{-1}(G)$ □

That means that this construction also defines a concrete functor:

$$T : \textbf{PreOrd} \to \textbf{Top} : (X, \leq) \mapsto (X, \mathcal{T}_{\leq})$$

3.1.3 Category Isomorphism Between Finite Topological Spaces and Pre-Orders

When restricting to the finite situation, the functors defined in the previous two sections work nicely together. As it turns out, they are each others inverses, yielding a concrete category isomorphism.

Definition 3.1.13. *The category* $\textbf{Top}_{\text{fin}}$ *is the category of finite topological spaces with continuous maps as morphisms.*

Definition 3.1.14. *The category* $\textbf{PreOrd}_{\text{fin}}$ *is the category of finite pre-ordered sets with order preserving maps as morphisms.*

Theorem 3.1.15. *The category* $\textbf{Top}_{\text{fin}}$ *is concretely isomorphic to the category* $\textbf{PreOrd}_{\text{fin}}$. *A concrete category isomorphism is given by*

$$P : \textbf{Top}_{\text{fin}} \to \textbf{PreOrd}_{\text{fin}} : (X, \mathcal{T}) \mapsto (X, \leq_{\mathcal{T}})$$

which equips a topological space with its specialisation pre-order. The inverse functor of this isomorphism is

$$T : \textbf{PreOrd}_{\text{fin}} \to \textbf{Top}_{\text{fin}} : (X, \leq) \mapsto (X, \mathcal{T}_{\leq})$$

which equips a pre ordered set with its order topology.

Proof. We have to show that $T \circ P = \mathrm{id}_{\mathbf{Top}_{\mathrm{fin}}}$ and $P \circ T = \mathrm{id}_{\mathbf{PreOrd}_{\mathrm{fin}}}$. So let $(X, \mathcal{T})$ be a finite topological space. Then $T(P((X, \mathcal{T}))) = (X, \mathcal{S})$ where $\mathcal{S}$ is the topology with basis $\{\downarrow x \mid x \in X\}$. A basis for $\mathcal{T}$ is $\{U_x \mid x \in X\}$. Notice that

$$\forall x \in X : \downarrow x = U_x$$

Indeed: $y \in U_x$ if and only if $y \leq_{\mathcal{T}} x$ if and only if $y \in \downarrow x$. It follows that $\mathcal{S} = \mathcal{T}$ and $T(P((X, \mathcal{T}))) = (X, \mathcal{T})$.

Vice versa, Let $(X, \leq)$ be a finite pre-ordered set. Then $P(T((X, \leq))) = (X, \preceq)$ where $\preceq$ is the specialisation pre-order of $\mathcal{T}_{\leq}$. This is by definition

$$x \preceq y \iff x \subseteq U_y$$

and because $U_z = \downarrow z$ for every $z \in X$ we have

$$x \preceq y \iff x \in \downarrow y \iff x \leq y$$

We conclude that $\leq = \preceq$ and $P(T((X, \leq))) = (X, \leq)$. □

Theorem 3.1.16. *The category of finite posets is concretely isomorphic to the category of finite topological spaces with the T_0 property*

Proof. The isomorphism is given by P of theorem 3.1.15, by restricting its domain to finite topological spaces with the T_0 property. By proposition 3.1.5, the image then restricts exactly to the category of finite posets. □

We will heavily rely upon this isomorphism in this thesis, in the sense that we will regard any finite topological space as a finite pre-ordered set and vice versa, sometimes without mentioning it, as they are truly the same thing. Likewise, any finite poset appearing in this thesis is just as much a finite T_0 space (and again, vice versa). Also categorical constructions (mostly products in this thesis) can be used without specifying which category we work in. An order theoretic product is just as much a topological product and vice versa. To clarify this, we first recall the definitions of products of topological spaces and products of pre-ordered sets.

Definition 3.1.17. *Let $(X_i)_{i \in I}$ be a family of topological spaces indexed over an arbitrary set I. The **product space** of $(X_i)_{i \in I}$ is the set $\prod_{i \in I} X_i$ equipped with the initial topology for the projections*

$$pr_k : \prod_{i \in I} X_i \to X_k : (x_i)_{i \in I} \mapsto x_k$$

for each $k \in I$. In other words, the coarsest topology on $\prod_{i\in I} X_i$ that makes all of the projections continuous, or equivalently, the topology generated by the subbasis

$$\{pr_k^{-1}(G) \mid k \in I, G \text{ open in } X_k\}$$

This topology is called the ***product topology****.*

Definition 3.1.18. *Let $(X_i)_{i\in I}$ be a family of pre-ordered sets indexed over an arbitrary set I. The* ***product pre-ordered set*** *of $(X_i)_{i\in I}$ is the set $\prod_{i\in I} X_i$ equipped with the pointwise pre-order*

$$(x_i)_{i\in I} \leq (y_i)_{i\in I} \iff \forall i \in I : x_i \leq y_i$$

This pre-order is called the ***product pre-order*** *or* ***product order****.*

Both definitions make use of the specific structures about which they talk, but actually, both definitions only depend on the morphisms corresponding to that structure (continuous maps and order morphisms respectively). This means that we can express them in a category theoretical context, where the two definitions coincide:

Definition 3.1.19. *Let $\mathcal{C}$ be a category. Let $(X_i)_{i\in I}$ be a family of objects of $\mathcal{C}$ indexed over an arbitrary set I. A* ***product object*** *of $(X_i)_{i\in I}$, denoted $\prod_{i\in I} X_i$, is an object for which there are morphisms*

$$pr_k : \prod_{i\in I} X_i \to X_k \text{ for each } k \in I$$

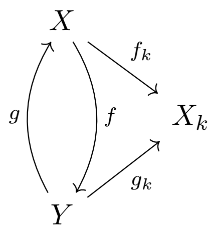
and which satisfies the following universal property: for every object $Y \in \mathcal{C}$ that also has morphisms $f_k : Y \to X_k$ for each $k \in I$, there is a unique morphism $f : Y \to \prod_{i\in I} X_i$ that makes the following diagram commute:

$$\begin{array}{ccc} \prod_{i\in I} X_i & \xrightarrow{pr_k} & X_k \\ {\scriptstyle f}\uparrow & \nearrow_{f_k} & \\ Y & & \end{array}$$

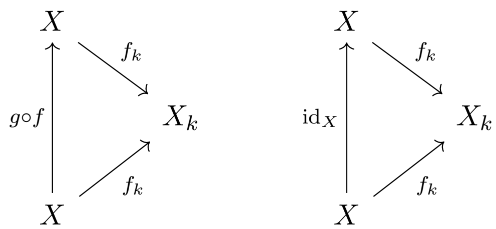
This definition is a bit different from the two previous definitions, in that it does not guarantee existence of product objects. In the category of pre-ordered sets and in the category of topological spaces, it does always exist, as the product pre-ordered set and the product space respectively satisfy the necessary properties. Notice also that we said '*a* product object' in contrast to for example '*the* product space'. This is because the definition of product objects does not guarantee uniqueness at first glance. To see that product objects are unique up to isomorphism, such that we can speak of '*the* product object' of $(X_i)_{i\in I}$, we need to do a little check.

Proposition 3.1.20. *Let $\mathcal{C}$ be a category. Let $(X_i)_{i\in I}$ be a family of objects of $\mathcal{C}$ indexed over an arbitrary set I. A product object of $(X_i)_{i\in I}$ is unique up to isomorphism.*

Proof. Let X and Y be product objects of $(X_i)_{i\in I}$. Then by the universal property of products we have a commutative diagram



for each $k \in I$. Notice that then, the following diagrams also commute:



But then $g \circ f = \mathrm{id}_X$ by the uniqueness in the universal property of products. Similarly we find that $f \circ g = \mathrm{id}_Y$, so $X \cong Y$ by the isomorphism f with inverse g. □

The conclusion is that the product space of topological spaces $(X_i)_{i\in I}$ is just the product object in the category of topological spaces, and the product pre-ordered set of pre-ordered sets $(X_i)_{i\in I}$ is just the product object in the category of pre-ordered sets. When we restrict to the finite case, we know that those categories are isomorphic. So given a collection of finite pre-ordered sets/topological spaces $(X_i)_{i\in I}$ (where I is finite to ensure the products are also finite), we find that the product space and the product pre-ordered sets are both isomorphic to the unique product object in the category of finite spaces/pre-ordered sets. So whenever we see a product of finite spaces, we can think of it as an order theoretic product and vice versa. We can also check this explicitly, and we then find that we have equality instead of mere isomorphism.

Proposition 3.1.21. *Let $(X_i)_{i=1}^n$ be a finite family of finite topological spaces. The product space $\prod_{i=1}^n X_i$ can also be seen as the set $\prod_{i=1}^n X_i$ equipped with the product order.*

Proof. We check what the specialisation order of $\prod_{i=1}^n X_i$ with the product topology is. Since this is a finite product, a basis for the product topology is given by the open boxes

$$\{G_1 \times \cdots \times G_n \mid \forall i \in \{1, \cdots, n\} : G_i \text{ open in } X_i\}$$

This implies that for any point $(x_1, \cdots, x_n) \in \prod_{i=1}^n X_i$

$$\begin{aligned} U_{(x_1,\cdots,x_n)} &= \bigcap \{G_1 \times \cdots \times G_n \mid \forall i \in \{1, \cdots, n\} : G_i \text{ open in } X_i, x_i \in G_i\} \\ &= \bigcap_{\substack{x_1 \in G_1 \\ G_1 \text{ open in } X_1}} G_1 \times \cdots \times \bigcap_{\substack{x_n \in G_n \\ G_n \text{ open in } X_n}} G_n \\ &= U_{x_1} \times \cdots \times U_{x_n} \end{aligned}$$

Therefore

$$\begin{aligned} (x_1, \cdots, x_n) \leq (y_1, \cdots, y_n) &\iff (x_1, \cdots, x_n) \in U_{(y_1,\cdots,y_n)} \\ &\iff \forall i \in \{1, \cdots, n\} : x_i \in U_{y_i} \\ &\iff \forall i \in \{1, \cdots, n\} : x_i \leq y_i \end{aligned}$$

which is precisely the definition of the product order. □

Another consequence of this category isomorphism is more practical. The fun part about the finiteness of finite spaces is that they allow for algorithmic computation. Algorithms require input, which can for example be represented using our custom `Space` class (see code listings, page 99). But according to this isomorphism, any representation of a poset on a computer is also a representation of a T_0 topological space, so this gives other options as well. For example, T_0 spaces can be represented using the `Poset` class of Sage [22].

3.2 Finite Homotopy Theory

Before we state and prove the motivating results of McCord [16] which connect finite and infinite spaces in algebraic topology, we first build up a small tool set for dealing with homotopy theory in the finite case. In particular, we will work towards a simple order-theoretic characterisation of homotopies between continuous maps between finite spaces, which will result in a classification theorem of the homotopy types of finite spaces. These results are mainly due to Stong [21] and were published right after McCord's article [16], but didactically it is useful to state them first. Apart from Stong's original article, we also used [2] and [14] as references in this section.

3.2.1 Order Theoretic Characterisation of Finite Homotopy

An important concept in homotopy theory is the fact that we can think of homotopies as paths in a certain function space. This shall be exploited in the finite case, but we first recall the general situation. Set-theoretically it is clear that a homotopy $h : X \times I \to Y$ can be regarded as a (set-theoretic) function $h' : I \to Y^X$. This is simply writing down that filling in a value $t \in I$ for the second argument, leaves a function $h(\cdot, t) : X \to Y$. In fact, this operation $h \mapsto h'$ yields a bijection between $Y^{X \times I}$ and $(Y^X)^I$ (which is aptly called the exponential law for sets). A similar situation can be found in the category of topological spaces, but it gets a bit more complicated. Given a homotopy

$$h : X \times I \to Y$$

we do get a function

$$h' : I \to C(X, Y)$$

Indeed, given an element $t \in I$, we get a continuous map $h(\cdot, t) : X \to Y$. But is h' still continuous? This of course depends on the topology we use for $C(X, Y)$. And is the operation

$$\cdot' : C(X \times I, Y) \to C(I, C(X, Y)) : h \mapsto h'$$

again a bijection? This is desired because the domain of the operation is the set of all homotopies of maps between X and Y, and the codomain is the set of all paths in the function space $C(X, Y)$. In many text books, for example [18], this situation is investigated and it is shown that for locally compact Hausdorff spaces, the above operation indeed gives a bijection. The topology used on the function space is the *compact-open topology*.

Definition 3.2.1. *Let X and Y be topological spaces. The **compact-open topology** is given by the subbasis elements*

$$S(K, G) = \{f \in C(X, Y) \mid f(K) \subseteq G\}$$

where $K \subseteq X$ is compact and $G \subseteq X$ is open.

In the finite setting, local compactness is always satisfied, but we don't want the requirement that our spaces need to be Hausdorff. In [9] it is shown that first countability is enough (which is always satisfied in the finite case). We summarise the situation in the following proposition.

Proposition 3.2.2. *Let X and Y be finite topological spaces. The set of homotopies of maps between X and Y is in bijective correspondence with the set of paths in $C(X, Y)$ with the compact-open topology. The bijection is given by*

$$\cdot' : C(X \times I, Y) \to C(I, C(X, Y)) : h \mapsto h'$$

This simplifies even further because in the finite setting, the compact-open topology is the same as the product topology.

Proposition 3.2.3. *Let X and Y be finite topological spaces. The product topology on $C(X, Y)$ (i.e. the topology induced by regarding it as a subspace of the product $Y^X = \prod_{x \in X} Y$) is equal to the compact-open topology on $C(X, Y)$. Notice that this is also the topology induced by the product order by the category isomorphism 3.1.15 (proposition 3.1.21).*

Proof. Let's start by showing that the compact-open topology is finer than the product topology. This holds in general, so we will not yet use finiteness. The product topology on $Y^X = \prod_{x \in X} Y$ is given by the initial topology for the source of the projections

$$(pr_x : Y^X \to Y : f \mapsto f(x))_{x \in X}$$

so the subspace $C(X, Y)$ is initial for the source

$$(pr_x : C(X, Y) \to Y : f \mapsto f(x))_{x \in X}$$

which means that it has a subbasis consisting of the sets

$$pr_x^{-1}(G), \quad x \in X, \quad G \text{ open in } Y$$

These subbasis elements can be rewritten as

$$\begin{aligned} pr_x^{-1}(G) &= \{f \in C(X, Y) \mid pr_x(f) \in G\} \\ &= \{f \in C(X, Y) \mid f(x) \in G\} \\ &= S(\{x\}, G) \end{aligned}$$

As singletons are compact, this shows that the subbasis elements for the product topology are also subbasis elements for the compact-open topology, so the compact-open topology is finer.

For the other inclusion, we rewrite the subbasis elements of the compact-open topology as follows:

$$\begin{aligned} S(K, G) &= \{f \in C(X, Y) \mid f(K) \subseteq G\} \\ &= \{f \in C(X, Y) \mid \forall x \in K : f(x) \in G\} \\ &= \bigcap_{x \in K} \{f \in C(X, Y) \mid f(x) \in G\} \\ &= \bigcap_{x \in K} S(\{x\}, G) = \bigcap_{x \in K} pr_x^{-1}(G) \end{aligned}$$

If K is finite, this shows that $S(K, G)$ is open for the product topology. This holds in particular in finite topological spaces. ∎

This means we can replace the compact-open topology in proposition 3.2.2 with the product topology, to get the following simple and useful result:

Proposition 3.2.4. *Let X and Y be finite topological spaces. The set of homotopies of maps between X and Y is in bijective correspondence with the set of paths in $C(X,Y)$ with the product topology (product order). The bijection is given by*

$$\cdot' : C(X \times I, Y) \to C(I, C(X,Y)) : h \mapsto h'$$

Having reduced de study of homotopies to the study of paths on the function space $C(X,Y)$, we now want to understand path-connectedness in the finite setting. The usual analytic intuition is hard to adapt to this finite case. However, Stong gives a nice order theoretic characterisation of path-connectedness which further facilitates the study of homotopies in finite spaces.

Definition 3.2.5. *Let X be a poset. Then X naturally induces a directed graph $G(X) = (V, E)$, with $V = X$ and*

$$\forall x, y \in X : (x,y) \in E \iff x < y$$

*We call $G(X)$ the **associated graph** of X, and we say that X is **(order)-connected** if $G(X)$ is a weakly connected directed graph (connected as an undirected graph).*

We can also forget about the graph and just think about order-connectedness in the following way:

Proposition 3.2.6. *Let X be a finite poset. Then X is order-connected if and only if for each pair of points $x, y \in X$ we have*

$$x = x_0 \leq x_1 \geq x_2 \leq \cdots \geq x_n = y$$

for some $x_0, \cdots, x_n \in X$

Proof. This follows immediately from the definition. Indeed, we have

$$x = x_0 \leq x_1 \geq x_2 \leq \cdots \geq x_n = y$$

for each pair $x, y \in X$ of points if and only if in the associated graph $G(X) = (V, E)$, we have edges

$$(x, x_1), (x_2, x_1), \cdots, (y, x_{n-1}) \in E$$

which connect x to y. This is equivalent to the fact that G is a connected graph, and that X is order-connected. □

Proposition 3.2.7. *Let X be a finite topological space. The following are equivalent:*

1. *X is order-connected*
2. *X is path-connected*
3. *X is connected*

Proof.

($1 \implies 2$) Assume that X is order-connected. Let $x, y \in X$ be comparable: w.l.o.g. we suppose that $x \leq y$. then

$$\alpha : [0,1] \to X : t \mapsto \alpha(t) = \begin{cases} x & \text{if } t \neq 1 \\ y & \text{if } t = 1 \end{cases}$$

is continuous. Indeed, let $U \in X$ be open, then

$$\alpha^{-1}(U) = \begin{cases} [0,1] & \text{if } x, y \in U \\ [0,1[& \text{if } x \in U, y \notin U \\ \emptyset & \text{if } x, y \notin U \end{cases}$$

The case $x \notin U, y \in U$ is impossible because if $y \in U$ then

$$x \in U_y \subseteq U$$

We also clearly have that $\alpha(0) = x$ and $\alpha(1) = y$, so α is a path from x to y.

If x and y are not comparable, then by order-connectedness of X, we have

$$x = x_0 \leq x_1 \geq x_2 \leq \cdots \geq x_n = y$$

for some $x_0, \cdots, x_n \in X$. By concatenating the paths between the consecutive points, we obtain a path between x and y.

($2 \implies 3$) This is a standard result in point-set topology.

($3 \implies 1$) Recall that a topological space is connected if and only if every continuous map to the discrete space (anti-chain order) with 2 points (we shall denote it by $[2] = \{0, 1\}$) is constant. This is a classic alternative definition of connectedness, see for example [5]. We will show that if X is not order-connected, there exists a non-constant continuous map to $[2]$. So assume X is not order-connected. Then we can find $a, b \in X$ such that they are not connected in the associated graph. Now we define

$$f : X \to [2] : x \mapsto f(x) = \begin{cases} 0 & \text{if } x \text{ is comparable to } a \\ 1 & \text{otherwise} \end{cases}$$

This map is not constant because $f(a) = 0$ and $f(b) = 1$. It is continuous though, as $x \leq y$ implies that x and y are either both comparable to a, or both not comparable to a, so $f(x) = f(y)$. In particular, f is order preserving thus continuous.

□

Corollary 3.2.8. *Let X be a finite topological space. Two points $x, y \in X$ are connected by a path if and only if*

$$x = x_0 \leq x_1 \geq x_2 \leq \cdots \geq x_n = y$$

for some $x_0, \cdots, x_n \in X$

Proof. We have that x and y are connected by a path if and only if x and y are in the same path-component, if and only if they are in the same order-connected component, if and only if

$$x = x_0 \leq x_1 \geq x_2 \leq \cdots \geq x_n = y$$

for some $x_0, \cdots, x_n \in X$ □

All of the above amounts to the following clean order-theoretic characterisation of homotopy for finite topological spaces:

Corollary 3.2.9. *Let X and Y be finite topological spaces. Two continuous maps $f, g : X \to Y$ are homotopic if and only if*

$$f = f_0 \leq f_1 \geq f_2 \leq \cdots \geq f_n = g$$

with the product order (i.e. pointwise order) for some $f_0, \cdots, f_n \in C(X, Y)$.

Proof. Note that $C(X, Y)$ is again a finite topological space. We have that $f, g : X \to Y$ are homotopic if and only if there is a path between them in $C(X, Y)$ (with the product topology/order). Now apply the previous corollary to the finite space $C(X, Y)$ □

This corollary will often be used in the particular case where f and g are comparable. Comparable maps are homotopic, but not necessarily the other way around. We shall immediately start using this fact, to make finite topological spaces smaller, without changing their homotopy type. This allows us to study the homotopy properties with fewer computations. We start by introducing two dual order-theoretic concepts.

Definition 3.2.10. *Let X be a finite poset. An order-preserving map $f : X \to X$ such that*

$$\forall x \in X : f(x) \leq x$$

*is called **deflationary** or **regressive**. Dually, an order-preserving map $f : X \to X$ such that*

$$\forall x \in X : f(x) \geq x$$

*is called **inflationary** or **progressive**.*

These maps appear for example in the Bourbaki-Witt theorem [24], an order theoretic fixed point theorem by Bourbaki [4] and Witt [23]. In the literature of finite topological spaces [21, 16, 2], they are not mentioned, but they are often implicitly used. We chose to mention them explicitly to highlight a common element in many proofs to come. As you will see, we will often rely on the following property of inflationary and deflationary maps.

Proposition 3.2.11. *Let X be a finite topological space, and $f : X \to X$ a deflationary or inflationary map. Then $f(X)$ is a deformation retract of X.*

Proof. Let f be deflationary (the inflationary case is analogous). We can factor f into a surjective function and an inclusion map, by restricting the codomain:

$$\begin{array}{ccc} X & \xrightarrow{f} & X \\ & \searrow^{\hat{f}} & \uparrow{\scriptstyle i} \\ & & f(X) \end{array}$$

Which gives rise to the desired homotopy equivalence between X and $f(X)$

$$\begin{Bmatrix} i \circ \hat{f} = f \leq \mathrm{id}_X \\ \hat{f} \circ i = f|_{f(X)} \leq \mathrm{id}_{f(X)} \end{Bmatrix} \implies \begin{Bmatrix} i \circ \hat{f} \simeq \mathrm{id}_X \\ \hat{f} \circ i \simeq \mathrm{id}_{f(X)} \end{Bmatrix}$$

□

This proposition can be used to prove the following propositions which are essential for the next section.

Proposition 3.2.12. *Let X be a finite topological space. If X has a maximum or a minimum, then X is contractible.*

Proof. Let $m \in X$ be a maximum (resp. minimum). The constant map

$$f : X \to X : x \mapsto m$$

is clearly inflationary (resp. deflationary), so X is homotopy equivalent to $f(X) = \{m\}$. □

Proposition 3.2.13. *Let X be a finite topological space. For each $x \in X$, its minimal neighbourhood U_x is contractible*

Proof. This follows immediately from the previous proposition as x is the maximum of U_x. □

Proposition 3.2.14. *Let X be a finite topological space. Then $X/\sim$ is homeomorphic to a deformation retract of X, where*

$$x \sim y \iff x \leq y \text{ and } y \leq x$$

and $X/\sim$ is T_0.

Proof. It is clear that the quotient enforces anti-symmetry, so the result is a T_0 space. Write $Y = X/\sim$. The quotient map

$$f : X \to Y : x \mapsto [x]$$

is a continuous map. For every $y \in X/\sim$, choose an element $g(y) \in y$, to define a map

$$g : Y \to X : y \mapsto g(y)$$

This map is order preserving (continuous) because $y \leq z$ in Y if and only if $a \leq b$ for some $a \in y$ and $b \in z$. This implies $g(y) \leq a \leq b \leq g(z)$. It is also clear that $f|_{g(Y)}$ and g are each other inverses. This shows that $g(Y)$ and Y are homeomorphic. Thus, it is sufficient to show that $g(Y)$ is a deformation retract of X. This follows immediately from the fact that $g \circ f$ is deflationary (and inflationary), and that $g(f(X)) = g(Y)$. □

3.2.2 Stong's Classification Theorem

If we keep using proposition 3.2.11 we obtain a so-called 'core' or 'minimal finite space'. The definition we give here is slightly different of the original definition given in [21], and later in [2] and [14]. In these sources they use a more combinatorial approach by removing one point at a time. This is easier to work with on a computer, so it shall be introduced in section 4.1. For now, the following definition is easier to work with and it is easier to get to a nice classification result of Stong.

Definition 3.2.15. *Let X be a finite topological space. There is one **trivial inflationary and deflationary map** on X, namely the identity map id_X. All other inflationary and deflationary maps are called **non-trivial**.*

Definition 3.2.16. *A finite T_0 topological space X is called a **core** or **minimal finite space** if it has no non-trivial deflationary or inflationary maps $f : X \to X$.*

As such, a core can no longer be made smaller by using proposition 3.2.11. Conversely, a T_0 space that is not a core can always be made smaller by using 3.2.11, as any non-trivial inflationary or deflationary map reduces the cardinality.

Proposition 3.2.17. *Let X be a finite T_0 topological space, and let $f : X \to X$ be a non-trivial inflationary or deflationary map. Then $f(X) \subsetneq X$.*

Proof. Assume w.l.o.g. that f is inflationary (the proof for a deflationary map is dual to this proof). Assume for a contradiction that f is surjective. As $f \neq \mathrm{id}_X$, there is $x \in X$ such that $f(x) > x$. By surjectivity, there is $x_1 < x$ such that $f(x_1) = x$. Again by surjectivity, there is $x_2 < x_1$ such that $f(x_2) = x_1$. Repeating this argument gives an infinite descending chain

$$\cdots < x_3 < x_2 < x_1 < x$$

which is impossible in a finite poset. □

Definition 3.2.18. *Let X be a finite topological space. A* ***core of the space*** *X is a subspace of X that is homotopy equivalent to X and is a core.*

The following theorems of Stong [21] show why cores are of interest.

Theorem 3.2.19. *Let X and Y be cores and $f : X \to Y$ a homotopy equivalence. Then f is a homeomorphism.*

Proof. Let $g : Y \to X$ be a homotopy inverse of f. We then have $g \circ f \simeq \mathrm{id}_X$. By 3.2.9 this means that there are maps $f_i : X \to X$ such that

$$g \circ f \leq f_1 \geq f_2 \leq \cdots \geq f_n \leq \mathrm{id}_X$$

so f_n is deflationary, but then $f_n = \mathrm{id}_X$ because X is a core. But then $f_{n-1} \geq id_X$ so f_{n-1} is inflationary. But then $f_{n-1} = \mathrm{id}_X$ etc. . . We can conclude that

$$g \circ f = f_1 = f_2 = \cdots = f_n = \mathrm{id}_X$$

The same argument on Y shows that $f \circ g = \mathrm{id}_Y$, so f is a homeomorphism with inverse g. □

Theorem 3.2.20. *Every finite topological space has a core which is unique up to homeomorphism.*

Proof. It is clear that every finite space is homotopy equivalent to a subspace that is a core. Indeed, we can first use proposition 3.2.14 to make it T_0, and then use proposition 3.2.11 until there are no non-trivial inflationary or deflationary maps any more, to arrive at a core while having never changed the homotopy type. For uniqueness, let Y and Z be two cores of a finite topological space X. They must both be homotopy equivalent to X, so also to each other. By theorem 3.2.19 they are also homeomorphic to each other. □

In the light of the previous theorem, we can unambiguously write $\mathrm{core}(X)$ for the core of a finite topological space X.

Theorem 3.2.21 (Stong's Classification Theorem)**.** *Let X and Y be finite topological space. Then X and Y are homotopy equivalent if and only if they have homeomorphic cores.*

Proof. If $f : X \to Y$ is a homotopy equivalence, then we have a homotopy equivalence

$$\mathrm{core}(X) \to X \xrightarrow{f} Y \to \mathrm{core}(Y)$$

which is a homeomorphism according to theorem 3.2.19. Conversely, if $f : \mathrm{core}(X) \to \mathrm{core}(Y)$ is a homeomorphism, then we have a homotopy equivalence

$$X \to \mathrm{core}(X) \xrightarrow{f} \mathrm{core}(Y) \to Y$$

□

3.3 The ‘Shape’ of Small Categories and the Order Complex

Simplicial complexes and simplicial sets are nice abstract ways of describing topological spaces. They are not topological spaces themselves, but they give the information needed to build a topological space, through geometric realisation. Many other abstract structures in mathematics can also be interpreted that way. These structures can be thought of as representing a shape. We will exhibit this on small categories, as they are on the one hand very general, and on the other hand, the ‘shape’ interpretation is not that far-fetched. Our reference for this section is [6] chapter XI.2. A *small category* is a category of which the class of objects and the classes of morphisms are sets. This allows us to work with it as if it were any other algebraic structure (sets equipped with algebraic operations, in this case the composition of morphisms). Specifically, we can regard all small categories together in a category **Cat**, where the objects are the small categories, and the morphisms are the functors between them. We will assign a shape to each small category by giving a functor **Cat** $\to$ **Top**. But first we will exemplify the generality of small categories:

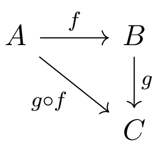
- A **group** can be thought of as a small category, with one object $*$, and a morphism $* \to *$ for each element of the group. This actually models the more general structure of a **monoid**, and the group axiom that demands that everything is invertible translates to the demand that every morphism is an isomorphism.

- A **pre-ordered set** can be thought of as a small category, where the objects are the points of the pre-ordered set, and we have a unique morphism $x \to y$ if and only if $x \leq y$. Such a category is a **poset** if and only if every isomorphism is an identity map.

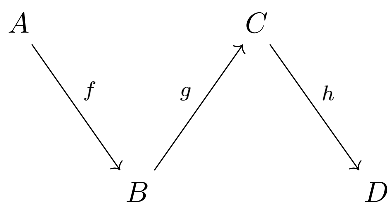
- A **vector space** becomes an abelian group by forgetting the scalar multiplication, and it induces a poset, namely the poset of subspaces ordered by inclusion. This means we can interpret a vector space as a small category in two ways (via a group or via a poset). The same can be said about R**-modules** and **rings** (with submodules and (prime)-ideals as substructures).

Thus, assigning a topological space to each small category also assigns a space to each of the above-mentioned structures. Especially pre-orders and posets will be of interest for us, because this assignment of spaces is different from the category isomorphism in 3.1.15. It will assign infinite spaces, even to finite posets. This provides somewhat of a connection between finite and infinite spaces which will be exploited in the next section.

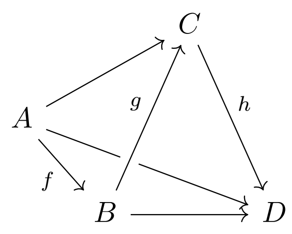
Let's now exhibit the shape of small categories. We will take a detour via simplicial sets for this, which are topological spaces after geometric realisation. Notice that small categories can be thought of as directed graphs, by forgetting the composition operation. So we can think of the objects as points or 0-simplices and the morphisms as edges or 1-simplices. A pair of two composable morphisms f and g yields a commutative triangle by adding the composition.



Or in other words, a commutative 2-simplex. Likewise, a collection of three composable morphisms



yields a commutative tetrahedron or 3-simplex, by adding all possible compositions.



For each $n \in \mathbb{N}$, a chain of n composable morphisms

$$A_0 \to A_1 \to \cdots \to A_n$$

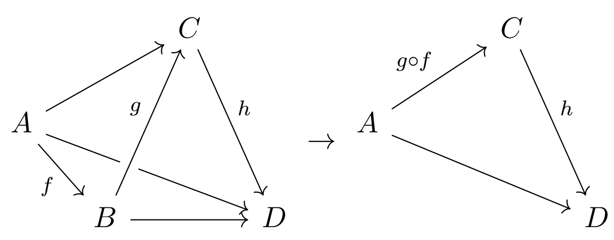
yields a commutative n-simplex like this. Thanks to the composition operation of small categories, these simplices are naturally equipped with face maps: we can delete the space A_i, $0 < i < n$, by composing the morphisms before and after A_i. This gives us a chain of $n-1$ composable morphisms, thus an $(n-1)$-simplex. For example, in the chain

$$A \xrightarrow{f} B \xrightarrow{g} C \xrightarrow{h} D$$

we can remove space B by composing f and g, giving the chain

$$A \xrightarrow{g \circ f} C \xrightarrow{h} D$$

At the level of the simplices, this looks like:



If the deleted space is A_0 or A_n, then no composition is made, but the corresponding morphism is just dropped.

We can also define degeneracy maps by repeating a space and adding the identity morphism. This completes the description of a particular simplicial set which is called the *nerve* of the small category.

Definition 3.3.1. *Let $\mathcal{C} \in$ **Cat** be a small category. The **nerve** of $\mathcal{C}$, denoted by $N(\mathcal{C})$ is the simplicial set where for each $n \in \mathbb{N}$*

- *The set of n-simplices, $N_n(\mathcal{C})$ is given by the set of chains of n composable morphisms*

$$A_0 \to A_1 \to \cdots \to A_n$$

  in $\mathcal{C}$.

- *The face maps $\partial_i : N_n(\mathcal{C}) \to N_{n-1}(\mathcal{C})$ are given by*

$$\partial_i(A_0 \to A_1 \to \cdots \xrightarrow{f} A_i \xrightarrow{g} \cdots \to A_n) =$$
$$(A_0 \to A_1 \to \cdots \to A_{i-1} \xrightarrow{g \circ f} A_{i+1} \to \cdots \to A_n)$$

  for $0 < i < n$, and

$$\partial_0(A_0 \to A_1 \to \cdots \to A_n) = (A_1 \to A_2 \to \cdots \to A_n)$$
$$\partial_n(A_0 \to A_1 \to \cdots \to A_n) = (A_0 \to A_1 \to \cdots \to A_{n-1})$$

- *The degeneracy maps $\sigma_i : N_n(\mathcal{C}) \to N_{n+1}(\mathcal{C})$ are given by*

$$\sigma_i(A_0 \to A_1 \to \cdots \to A_i \to \cdots \to A_n) =$$
$$(A_0 \to A_1 \to \cdots \to A_i \xrightarrow{1_{A_i}} A_i \to \cdots \to A_n)$$

This way, we obtain a functor

$$N : \mathbf{Cat} \to \mathcal{S}\mathbf{Set}$$

which we can compose with the geometric realisation functor to obtain a functor

$$|N| : \mathbf{Cat} \to \mathbf{Top}$$

In the specific case of finite posets, the nerve can be seen as a simplicial complex called the *order complex*.

Definition 3.3.2. *The* ***order complex*** $\mathscr{K}(X)$ *of a finite poset X is the ordered simplicial complex, of which the vertices are given by the points of X, and the simplices are the non-empty chains $x_0 < \cdots < x_n$ of X. When interpreting this ordered simplicial complex as a simplicial set, this is the same as the nerve of X regarded as a small category.*

3.4 Equivalences Between Infinite and Finite Spaces

In this section we will finally expose McCord's connection between finite simplicial complexes and finite topological spaces which was first presented in [16]. We will combine parts of May's unfinished book [14] and Barmak's book [2] to obtain an exposition that is as clear as possible. McCord's theory primarily talks about weak homotopy equivalences. Homotopy equivalences are too strong for his theorems. We start by stating the most important property that weak homotopy equivalences have, but homotopy equivalences do not have.

3.4.1 McCord's Theorem

Definition 3.4.1. *Let X be a topological space. A* ***basis-like open cover of*** *X is an open cover of X which is the basis for a topology. Note that it does not need to be a basis for the topology of X, but it has to be open for the topology of X.*

Definition 3.4.2. *Let $f : X \to Y$ be a continuous map between topological spaces. We say that f is a* ***local (weak) homotopy equivalence*** *If there is a basis open cover $\mathscr{U}$ of Y such that for all $U \in \mathscr{U}$*

$$f|_{f^{-1}(U)} : f^{-1}(U) \to U$$

is a (weak) homotopy equivalence.

Theorem 3.4.3 (McCord)**.** *A local weak homotopy equivalence is a weak homotopy equivalence*

In particular, a local homotopy equivalence is a weak homotopy equivalence, but not necessarily a homotopy equivalence. Virtually all weak homotopy equivalences which are discussed in this thesis appear as, or were derived from local (weak) homotopy equivalences, using McCord's theorem. An example of how McCord's theorem can be used to prove the weak homotopy equivalence of an infinite space with a finite space, is given by the finite version of the *suspension* of a space. Let's first introduce this concept, along with the concept of the *cone* of a space. If $X = S^1$, then $X \times [0,1]$ is a topological cylinder, built out of a lot of copies of X. If we now identify the top copy of X to a single point, we obtain a topological cone. In general, we define a cone of a space as follows:

Definition 3.4.4. *Let X be a topological space. The* ***cone*** *of X is the space*

$$C(X) = (X \times [0,1])/\sim$$

where $(x,t) \sim (y,s) \iff (x,t) = (y,s)$ or $t = s = 1$. We denote the top element $[(x,1)]$ by $+$

It is easy to see that the cone of any space is contractible to the top point by the homotopy

$$h : C(X) \times [0,1] \to C(X) : ([(x,s)],t) \mapsto [(x,(1-t)s)]$$

It is also clear that it preserves Hausdorffness. What it clearly does not preserve is finiteness, but we can easily create a finite version of this construction, which then obviously does not preserve Hausdorffness.

Definition 3.4.5. *Let X be a finite topological space. The* ***non-Hausdorff cone*** *or* ***finite cone*** *of X is the space $\mathbb{C}(X) = X \cup \{+\}$ where the open sets are given by the open sets of X with the addition of the set $X \cup \{+\}$.*

Again this makes the space $\mathbb{C}(X)$ contractible, which follows immediately from the fact that $U_+ = X \cup \{+\}$, or in other words, from the fact that $+$ is a maximum. In this sense it fulfils the same homotopical role as $C(X)$, but it is of course a completely different space, with a different underlying set.

The suspension of a space can be thought of as a double cone on the space.

Definition 3.4.6. *The suspension of a topological space X is the space*

$$S(X) = (X \times [-1, 1])/\sim$$

where $(x,t) \sim (y,s) \iff (x,t) = (y,s)$ or $t = s = -1$ or $t = s = 1$ We denote the top element $[(x,1)]$ by $+$ and the bottom element $[(x,-1)]$ by $-$

This construction has an interesting effect on the homology groups of the space, namely that it shifts them by one index. This follows from the exactness of the Mayer-Vietoris sequence (see e.g. [11] for more information on the Mayor-Vietoris sequence):

$$\begin{aligned} \cdots \to H_{n+1}(S(X)) \to H_n(A \cap B) \to H_n(A) \oplus H_n(B) \to H_n(S(X)) \\ \to H_{n-1}(A \cap B) \to \cdots \to H_0(A) \oplus H_0(B) \to H_0(S(X)) \to 0 \end{aligned}$$

for the open cover $S(X) = A \cup B$ with $A = S(X) \setminus \{+\}$ and $B = S(X) \setminus \{-\}$. Indeed, A and B are both homeomorphic to the cone $C(X)$ with the bottom copy of X removed[1], so they are both contractible and $A \cap B$ is retractible to $X \times \{0\}$ by the retraction

$$h : A \cap B \times [0,1] \to A \cap B : ([(x,s)], t) \mapsto [(x, (1-t)s)]$$

so $A \cap B$ is homotopy equivalent to X. The Mayor-Vietoris sequence then simplifies to

$$\begin{aligned} \cdots \to 0 \to H_{n+1}(S(X)) \to H_n(X) \to 0 \to H_n(S(X)) \\ \to H_{n-1}(X) \to 0 \to \cdots \to 0 \to H_0(S(X)) \to 0 \end{aligned}$$

[1]This can be seen as follows:

$$\begin{aligned} S(X) \setminus \{-\} &= (X \times]-1, 1])/\sim \\ &\cong (X \times]0,1])/\sim \\ &= C(X) \setminus (X \times \{0\}) \end{aligned}$$

where $(x,t) \sim (y,s) \iff (x,t) = (y,s)$ or $t = s = 1$ in both cases.

Because of the exactness, we have for each $n \in \mathbb{N}$

$$H_{n+1}(S(X)) \cong H_n(X)$$

Similarly to the non-Hausdorff cone, we can create a non-Hausdorff version of suspension that does preserve finiteness.

Definition 3.4.7. *Let X be a finite topological space. The* ***non-Hausdorff suspension*** *or* ***finite suspension*** *of X is the space $\mathbb{S}(X) = X \cup \{+, -\}$ where the open sets are given by the open sets of X with the addition of the sets $X \cup \{+\}$, $X \cup \{-\}$ and $X \cup \{+, -\}$.*

Notice that $X \cup \{+\}$ and $X \cup \{-\}$ are both homeomorphic to $\mathbb{C}(X)$, thus contractible. Along with the fact that their intersection is X, the exactness of the Mayor-Vietoris sequence again shows that this construction shifts the homology groups by one index. But we can say something even stronger, namely that the finite suspension is weak homotopy equivalent to the suspension, using McCord's theorem. This of course implies that it has the same effect on the homology groups as the suspension.

Proposition 3.4.8. *Let X be a finite topological space. There is a weak homotopy equivalence*

$$f : S(X) \to \mathbb{S}(X)$$

given by

$$f([(x,t)]) = \begin{cases} + & \text{if } t = 1 \\ - & \text{if } t = -1 \\ x & \text{else} \end{cases}$$

Proof. We show that f is a local homotopy equivalence using the basis-like open cover

$$\{U_+, U_-, X\} = \{X \cup \{+\}, X \cup \{-\}, X\}$$

The inverse image of U_+ is $S(X) \setminus \{-\}$ which is contractible as discussed earlier. We also have that U_+ is contractible. This implies that the constant map

$$g : U_+ \to S(X) \setminus \{-\} : a \mapsto +$$

is a homotopy inverse for $f|_{S(X)\setminus\{-\}}$. The same argument holds for U_-. The inverse image of X is $S(X) \setminus \{+, -\}$. A homotopy inverse for $f|_{f^{-1}(X)}$ is given by

$$g : X \to S(X) \setminus \{+, -\} : x \mapsto [(x, 0)]$$

Indeed, $f|_{f^{-1}(X)} \circ g = \text{id}_X$, and $g \circ f|_{f^{-1}(X)}$ is homotopic to the identity by the homotopy

$$h : S(X) \setminus \{+, -\} \times [0, 1] \to A \cap B : ([(x, s)], t) \mapsto [(x, (1 - t)s)]$$

This concludes the proof that f is a local homotopy equivalence. By McCord's theorem, f is then a weak homotopy equivalence. □

Proposition 3.4.9. *Let X and Y be finite topological spaces, and let*

$$f : X \to Y$$

be a weak homotopy equivalence. Then there is a weak homotopy equivalence

$$g : \mathbb{S}(X) \to \mathbb{S}(Y)$$

given by

$$g(x) = \begin{cases} + & \text{if } x = + \\ - & \text{if } x = - \\ f(x) & \text{else} \end{cases}$$

Proof. We will use the basis-like open cover

$$\{Y \cup \{+\}, Y \cup \{-\}, Y\}$$

We have that $g|_{g^{-1}(Y)} = g|_X = f$ is a weak homotopy equivalence by assumption and $g|_{g^{-1}(Y \cup \{+\})} = g|_{X \cup \{+\}}$ is a continuous map between contractible spaces thus trivially a weak homotopy equivalence. The same holds for $g|_{g^{-1}(Y \cup \{-\})} = g|_{X \cup \{-\}}$. By McCord's theorem, g is a weak homotopy equivalence. □

Proposition 3.4.10. *Let X be a finite topological space. There is a weak homotopy equivalence*

$$S^n(X) \to \mathbb{S}^n(X)$$

for all $n \in \mathbb{N}$

Proof. By proposition 3.4.8 we have a weak homotopy equivalence

$$f : S(X) \to \mathbb{S}(X)$$

Using this in proposition 3.4.9 we get a weak homotopy equivalence

$$g : \mathbb{S}(S(X)) \to \mathbb{S}(\mathbb{S}(X))$$

and using proposition 3.4.8 again on the space $S(X)$ we also have a weak homotopy equivalence

$$h : S(S(X)) \to \mathbb{S}(S(X))$$

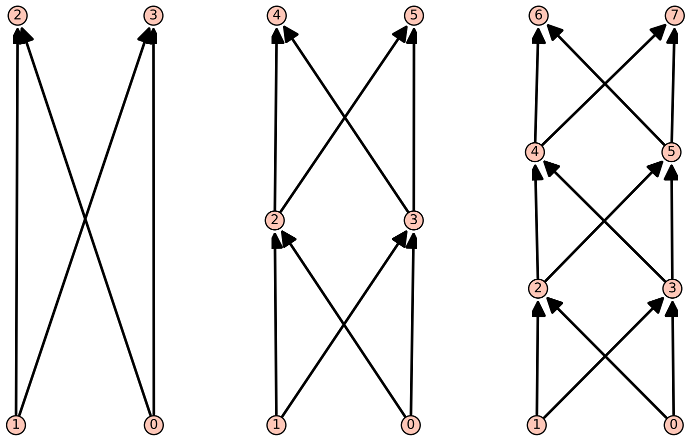


Figure 3.1: The Hasse diagram for the pseudo circle, the pseudo sphere and the pseudo 3-sphere respectively.

The composition

$$S(S(X)) \xrightarrow{h} \mathbb{S}(S(X)) \xrightarrow{g} \mathbb{S}(\mathbb{S}(X))$$

gives the desired weak homotopy equivalence for $n = 2$. Repeating this argument gives the result for all n. □

One of the main motivations for this section was to be able to represent some infinite topological spaces by finite ones. We can already give a nice example of this using finite suspensions.

Definition 3.4.11. *Let X be a (possibly infinite) topological space. A **finite model** for X is a finite topological space Y of the same weak homotopy type.*

Example 3.4.12. *The n-sphere S^n is the subspace of points at distance 1 from the origin in $\mathbb{R}^{n+1}$. A standard fact about the n-spheres is that S^n is homeomorphic to the suspension $S(S^{n-1})$ of S^{n-1}. Together with the fact that S^0 is the discrete space on two points, this gives us a finite model of each n-sphere, as proposition 3.4.10 gives a weak homotopy equivalence*

$$S^n = S^n(S^0) \to \mathbb{S}^n(S^0)$$

*Taking the finite suspension adds two points, and S^0 already has two points, so $\mathbb{S}^n(S^0)$ is a finite model of the n-sphere with $2n + 2$ points. We call this finite model the **pseudo n-sphere**. In the case $n = 1$ it is called the **pseudo circle** and in the case $n = 2$ it is called the **pseudo sphere**. The poset representations of the pseudo n-spheres for $n = 1, 2, 3$ are visualised in figure 3.1.*

3.4.2 Equivalence of Finite Spaces and Finite Simplicial Complexes

For this thesis, the most important use of McCord's theorem is the proof that each finite T_0 topological space X is weak homotopy equivalent to (the geometric realisation of) its order complex, $|\mathscr{K}(X)|$. This will require some supporting definitions and a lemma.

Definition 3.4.13. *Let X be a finite T_0 topological space, and $\alpha \in |\mathscr{K}(X)|$. Remember that $|\mathscr{K}(X)|$ is a quotient of a bunch of copies of standard simplices, of which we identified the vertices with the vertices of the abstract simplices of $\mathscr{K}(X)$. This means that α is of the form*

$$\alpha = t_0 x_0 + \cdots + t_n x_n \text{ with } \sum_{i=0}^{n} t_i = 1$$

*for some chain $x_i < \cdots < x_n$ in X. We can w.l.o.g. assume that $t_i > 0$ for all i, because we can just drop the vertices where t_i is zero, to obtain a new (smaller) chain. When dropping all such vertices, we obtain the smallest possible chain which generates α. The **support** of α is the set of vertices $\{x_0, \cdots, x_n\}$ of this smallest chain. It is denoted $\operatorname{supp}(\alpha)$.*

The following definition and lemma are a special case of the standard theory of simplicial complexes and can be found in [19] lemma 70.1 and chapter 8.72.

Definition 3.4.14. *Let X be a finite T_0 topological space. For each chain $\sigma \in \mathscr{K}(X)$ of X, we have a copy of a standard simplex in $|\mathscr{K}(X)|$, which we shall denote as $|\sigma|$ and its interior as $|\sigma|^\circ$. The **open star of a vertex** $x \in X$ is defined as*

$$\operatorname{star}(x) = \bigcup_{\sigma \ni x} |\sigma|^\circ$$

*The **open star of an open set** $V \in X$ is defined as*

$$\operatorname{star}(V) = \bigcup_{x \in V} \operatorname{star}(x)$$

This is clearly an open subset of $|\mathscr{K}(X)|$.

Lemma 3.4.15. *Let X be a finite T_0 topological space and V an open subset of X. Then we have*

$$\operatorname{star}(V) = |\mathscr{K}(X)| \setminus |\mathscr{K}(X \setminus V)|$$

and $|\mathscr{K}(V)|$ is a deformation retract of $\operatorname{star}(V)$.

We are now able to prove that the following map is a weak homotopy equivalence.

Definition 3.4.16. *Let X be a finite T_0 topological space. The **$\mathscr{K}$-McCord map** is the map*

$$\mu_X : |\mathscr{K}(X)| \to X : \alpha \mapsto \min(\operatorname{supp}(\alpha))$$

Proposition 3.4.17. *Let X be a finite T_0 topological space. The $\mathscr{K}$-McCord map is a weak homotopy equivalence.*

Proof. We will use McCord's theorem using the minimal basis $\{U_x \mid x \in X\}$ as basis-like open cover. We already know that each element of this cover is contractible, so it remains to be shown that $\mu_X^{-1}(U_x)$ is open (this implies continuity) and contractible for each x, because a continuous map between two contractible spaces is trivially a weak homotopy equivalence.

So let $x \in X$. We show that

$$\mu_X^{-1}(U_x) = |\mathscr{K}(X)| \setminus |\mathscr{K}(X \setminus U_x)| = \mathrm{star}(U_x)$$

Where the second equality is lemma 3.4.15. Indeed:

$$\begin{aligned} \alpha \in \mu_X^{-1}(U_x) &\iff \mu_X(\alpha) \in U_x \\ &\iff \min(\mathrm{supp}(\alpha)) \in U_x \\ &\iff \exists y \in \mathrm{supp}(\alpha) : y \in U_x \\ &\iff \alpha \notin |\mathscr{K}(X \setminus U_x)| \\ &\iff \alpha \in |\mathscr{K}(X)| \setminus |\mathscr{K}(X \setminus U_x)| \end{aligned}$$

As $\mathrm{star}(U_x)$ is open, μ_x is continuous. Now, thanks to lemma 3.4.15, we know that $|\mathscr{K}(U_x)|$ is a deformation retract of $\mathrm{star}(U_x)$, so it only remains to be shown that $|\mathscr{K}(U_x)|$ is contractible. This can be seen as follows. For each simplex σ in $\mathscr{K}(U_x)$ that does not contain x, $\sigma \cup \{x\}$ is again a simplex of $\mathscr{K}(U_x)$ because x is maximal thus it extends σ to a longer chain. Every point of σ can then be contracted to x through this extended simplex. To be more precise, $|\mathscr{K}(U_x)|$ is contractible via the homotopy

$$h : |\mathscr{K}(U_x)| \times [0,1] \to |\mathscr{K}(U_x)| : (\alpha = t_0x_0 + \cdots + t_nx_n, t) \mapsto (1-t)\alpha + tx$$

where the output is well-defined as it is a point of the simplex $|\sigma \cup \{x\}|$ for $\alpha \in |\sigma|$. To summarise, $\mu_X^{-1}(U_x) = \mathrm{star}(U_x)$ is open and contractible for each x, so μ_x is a weak homotopy equivalence. □

Theorem 3.4.18. *Every finite topological space is weak homotopy equivalent to the geometric realisation of a finite simplicial complex.*

Proof. Let X be a finite topological space. By proposition 3.2.14, X is homotopy equivalent to a finite T_0 topological space Y. By proposition 3.4.17, Y is weak homotopy equivalent to the geometric realisation of its order complex. Therefore, X is weak homotopy equivalent to the geometric realisation of the order complex of Y. □

Now we know how we can replace any finite topological space by a finite simplicial complex without changing its weak homotopy type. We also want to be able to change any finite simplicial complex into a finite topological space without changing its homotopy type. This will follow very easily from some standard theory about the barycentric subdivision of a simplicial complex, as can be found in, for example, [19]. The barycentric subdivision appears there as a special case of subdivisions of simplicial complexes, which all preserve the geometric realisation. In other words, subdividing a simplicial complex does not change the topological space it represents, only the way we represent it. We shall record this fact and one of the equivalent characterisations of barycentric subdivision.

Definition 3.4.19. *Any simplicial complex K can be interpreted as a poset $\mathscr{X}(K)$, by ordering the simplices by inclusion. We call this poset the* ***face poset*** *of K.*

Definition 3.4.20. *The* ***barycentric subdivision*** *of a simplicial complex K is the ordered simplicial complex $K' = \mathscr{K}(\mathscr{X}(K))$, i.e. the order complex of the face poset. This can be interpreted as a subdivision of K at the barycenters of each simplex, which is the average point of the simplex. This comes with a homeomorphism $s_K : |K'| \to |K|$ defined as follows: remember that each simplex of K' is a chain $\sigma_1 \subset \cdots \subset \sigma_n$ in the face poset of K. Every point in $|K'|$ is a convex combination of such a chain. We now define $s_K(\sigma) = b(\sigma)$ for each $\sigma \in K$ where $b(\sigma)$ is the barycenter of σ, and extend this linearly to $|K'|$.*

Using the barycentric subdivision, we get a McCord map going the other way. Indeed, a finite simplicial complex K must be weak homotopy equivalent to its face poset, as the $\mathscr{K}$-McCord map shows that the face poset is weak homotopy equivalent to the first barycentric subdivision of the complex, which is homeomorphic to the original complex. We can just compose

$$|K| \xrightarrow{s_K^{-1}} |\mathscr{K}(\mathscr{X}(K))| \xrightarrow{\mu_{\mathscr{X}(K)}} \mathscr{X}(K)$$

Definition 3.4.21. *Let K be a finite simplicial complex. The* ***$\mathscr{X}$-McCord map*** *is the map*

$$\mu_K : |K| \to \mathscr{X}(K) : \alpha \mapsto \mu_{\mathscr{X}(K)}(s_K^{-1}(\alpha))$$

Proposition 3.4.22. *Let K be a finite simplicial complex. The $\mathscr{X}$-McCord map is a weak homotopy equivalence.*

Proof. This follows immediately as it is the composition of a weak homotopy equivalence and a homeomorphism. □

So we have now proven the following theorem.

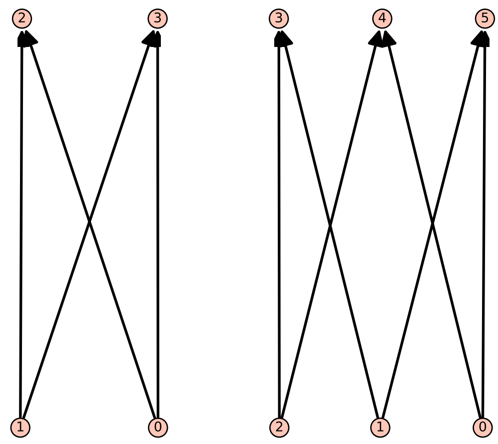


Figure 3.2: The Hasse diagram for the pseudo circle $\mathbb{S}(S^0)$ and the face poset P of a hollow triangle respectively.

Theorem 3.4.23. *Every topological space that is homeomorphic to the geometric realisation of a finite simplicial complex has a finite model.*

This theorem provides a finite model for many infinite topological spaces. For example for the circle S^1. This finite model is different from the finite model obtained by taking finite suspensions. Comparing these two finite models for the sphere yields an interesting counter example in algebraic topology. It shows that not every weak homotopy equivalence is a homotopy equivalence.

Example 3.4.24. *The circle can be topologically realised as the suspension of the 0-sphere (discrete space on 2 points), or as the geometric realisation of a hollow triangle $K = \downarrow \{\{0,1\}\{1,2\}\{0,2\}\}$ (example 2.2.7). Both versions give a finite model. The first finite model is the finite suspension of the 0-sphere $\mathbb{S}(S^0)$, and the second finite model is the face poset $P = \mathscr{X}(K)$ of a hollow triangle K (figure 3.2). The situation is summarised in the following diagram:*

$$\begin{array}{ccccc} S(S^0) & \longleftrightarrow & S^1 & \longleftrightarrow & |K| \\ \big\downarrow f & & & & \big\downarrow \mu_K \\ \mathbb{S}(S^0) & & & & P \end{array}$$

Here, f is the weak homotopy equivalence given by proposition 3.4.8 and μ_K is the $\mathscr{X}$-McCord map, which is a weak homotopy equivalence according to proposition 3.4.22. Let's assume for the sake of contradiction that these weak homotopy equivalences are homotopy equivalences. Then there is also a homotopy equivalence going the other way, and all of these maps then compose to a homotopy equivalence $\mathbb{S}(S^0) \to P$. An easy check (for example, with algorithm 1) shows that $\mathbb{S}(S^0)$ and P are cores, so this *homotopy equivalence is a homeomorphism according to Stong's theorems. This is a contradiction because $\mathbb{S}(S^0)$ has 4 points, while P has 6 points.*

McCord's theory also provides numerous counter examples that show that if we have a weak homotopy equivalence $f : X \to Y$, we do not necessarily have a weak homotopy equivalence $g : Y \to X$. This follows from a simple lemma:

Lemma 3.4.25. *Let X be a connected finite topological space, and let Y be a T_1 (possibly infinite) topological space. Then any continuous map $f : X \to Y$ is constant.*

Proof. Since X is finite and connected, $f(X)$ is also finite and connected. Since Y is T_1, its subspace $f(X)$ is also T_1. Combined, this tells us that $f(X)$ is a connected finite T_1 space. Any finite T_1 space is discrite, and a discrete connected space is a singleton. Therefore $f(X)$ is a singleton. □

Corollary 3.4.26. *Let X be a connected finite topological space, and let Y be a T_1 (possibly infinite) topological space. If there is a weak homotopy equivalence $f : X \to Y$, then all homotopy groups of X and Y are trivial.*

Proof. A weak homotopy equivalence is in particular continuous, so by the previous lemma, f is constant. Then the induced maps

$$\hat{f}_n : \pi_n(X, x) \to \pi_n(Y, f(x)) : [p] \mapsto [f \circ p]$$

are also constant for each $n \geq 1$. But they also have to be isomorphisms (by definition of a weak homotopy equivalence), which implies that $\pi_n(X, x)$ and $\pi_n(Y, f(x))$ are trivial. □

Using the previous lemma and corollary, McCord's theory gives us the building blocks to find a weak homotopy equivalence $f : Y \to X$ such that there is no weak homotopy equivalence $g : X \to Y$. Indeed, the McCord maps are weak homotopy equivalences from infinite Hausdorff (in particular T_1) spaces to finite spaces (proposition 3.4.17 and 3.4.22). This is also the case for the weak homotopy equivalence of proposition 3.4.8. In the case of connected spaces with a non-trivial homotopy group, there cannot be a weak homotopy equivalence going the other way. We give a simple concrete example.

Example 3.4.27. *Let*

$$f : S(S^0) \to \mathbb{S}(S^0)$$

be the weak homotopy equivalence of proposition 3.4.8 from the circle to the pseudo circle. Suppose for the sake of contradiction that there is also a weak homotopy equivalence

$$g : \mathbb{S}(S^0) \to S(S^0)$$

Because the pseudo circle is finite and connected, and the circle is Hausdorff (in particular T_1*), all their homotopy groups are trivial according to the previous lemma and corollary. This is of course not true, the circle's fundamental group is* $\mathbb{Z}$*.*

Chapter 4

Algorithmic Considerations

4.1 One-Point Reduction

In this section, we shall give a combinatorial way of describing the core of a finite topological space. This shall be done by introducing the idea of 'beat' points, originally introduced by Stong [21] under the name 'linear and colinear points'. These are points which can be removed from the space without changing its homotopy type and removing all of them yields the core of the space. This immediately gives rise to an algorithm for computing the core of a space. This in turn is desirable when computing homotopy invariants of finite topological spaces, as we can then do the computation with fewer points. It can also be used as a building block for an algorithm that checks if two spaces are homotopy equivalent. We can do this by first computing their cores, and then checking of this cores are homeomorphic (or isomorphic as posets/graphs).

Remember that the homotopy theory of finite spaces can be reduced of that of finite T_0 spaces by proposition 3.2.14. Therefore, this section only deals with finite T_0 spaces.

Definition 4.1.1. *Let X be a finite T_0 topological space. A point $x \in X$ is called **downbeat** if it covers exactly one point, or in other words, if its in-degree in the Hasse diagram of X is 1. Dually, a point is called **upbeat** if it is covered by exactly one point, or in other words, if its out-degree in the Hasse diagram of X is 1. A point is called **beat** if it is up beat or down beat.*

Proposition 4.1.2. *Let X be a finite T_0 topological space and let $x \in X$ be an upbeat (resp. downbeat) point. Let y be the unique point that covers x (resp. is covered by x). Then*

$$f : X \to X : a \mapsto \begin{cases} a & \text{if } a \neq x \\ y & \text{if } a = x \end{cases}$$

is inflationary (resp. deflationary). In particular, X is homotopy equivalent to the subspace $f(X) = X \setminus \{x\}$.

Proof. Suppose x is upbeat. We check that f is order preserving. Indeed, let $a < x$, then

$$f(a) = a < x < y = f(x)$$

If $x < b$ then

$$f(x) = y \leq b = f(b)$$

because otherwise $b < y$ (but y covers x so this is not the case), or b and y are incomparable but then the interval $]x, b]$ contains an element that covers x and is not equal to y. So in general, if $a \leq b$ then we are either in one of the previous cases or we trivially have that $f(a) \leq f(b)$.
It is then clear that f is inflationary as every $f(x) = y > x$ and $f(a) = a$ for all $a \neq x$. Proposition 3.2.11 now tells us that X is homotopy equivalent to $f(X) = X \setminus \{x\}$.

If x is downbeat and y is the unique point that is covered by x, the same arguments show that f is deflationary, so we obtain the same conclusion. □

Proposition 4.1.3. *Let X be a finite T_0 topological space. Then X is a core if and only if it has no beat points.*

Proof. If X still has a beat point, then removing it yields an inflationary or deflationary map (proposition 4.1.2). Conversely, suppose that X is not a core. Then w.l.o.g. there is an inflationary map $f : X \to X$ with $f \neq \mathrm{id}_X$ (the argument for an inflationary map is dual). This means the set

$$\{y \in X | f(y) \neq y\}$$

is non-empty. It is also finite so it has a maximal element x. We shall show that $f(x)$ is the unique element that covers x to conclude that x is upbeat. Indeed, suppose that there is an element $x < z < f(x)$, then by maximality of x, $f(z) = z$, but then $f(z) < f(x)$ which is in contradiction with the fact that f is order preserving. This already shows that $f(x)$ covers x. Now suppose that y also covers x. Then $x < y$ so $x < f(x) \leq f(y) = y$ which implies $y = f(x)$. □

We can immediately translate proposition 4.1.2 to the following algorithm:

Algorithm 1 Computing Cores

```
function GET_CORE(X)
    core = X.
    while X has a beat point p do
        core = core \ {p}
    end while
    return core
end function
```

This algorithm is implemented as the `core` function on the `Space` object (page 99).

4.2 The Euler Characteristic and the Möbius Function

The Euler character characteristic (see for example [11]) is a well-known characteristic of (the surface of) polyhedra. Given a polyhedron P, It is classically defined as $\chi(P) = V - E + F$, where V, E and F are the number of vertices, edges and faces of P, respectively. For a convex polyhedron, we always have $V - E + F = 2$, which is known as Euler's formula. If the polyhedron's faces are all triangles, then its surface is a simplicial complex of dimension 2. Other polyhedra can always be triangulated without changing the Euler characteristic. Take a quadrilateral face for example. By adding a diagonal, we divided it up into 2 triangles. This increases the number of edges by one, but also the number of faces, therefore keeping the Euler characteristic constant. For any face of a polyhedron, one can just keep adding diagonals until there are only triangles left. We can thus interpret the Euler characteristic of a polyhedron, as the Euler characteristic of its triangulation. In this interpretation, the Euler characteristic is a property of simplicial complexes (so far up to dimension 2). We can generalise this definition to arbitrary finite simplicial complexes.

Definition 4.2.1. *The* ***Euler characteristic of a finite simplicial complex*** K *is defined as*

$$\chi(K) = \sum_{i=0}^{\infty} (-1)^n \# K_i = \# K_0 - \# K_1 + \# K_2 - \# K_3 + \cdots$$

where K_i *is the set of all* i*-simplices, and* $\# K_i$ *is the number of elements in this set. Notice that this sum is finite because* K *is finite.*

The following equivalent characterisation of the Euler characteristic places it in the theory of homology. It shows that the Euler characteristic is invariant under (weak) homotopy equivalences, and in particular, it gives a simple proof of Euler's formula.

Proposition 4.2.2. *Let K be a finite simplicial complex. Then we have*

$$\chi(K) = \sum_{i=0}^{\infty} (-1)^n b_i = b_0 - b_1 + b_2 - b_3 + \cdots$$

where b_i is the i-th Betti number of K with coefficients in $\mathbb{Q}$, i.e. the dimension of the vector space $H_i(K; \mathbb{Q})$. Notice that this sum only has a finite number of non-zero terms because K is finite.

Proof. Recall that for each n we have a free $\mathbb{Q}$ vector space of n-chains C_n spanned by the n-simplices. In particular $\dim(C_n) = \#K_n$. We also have a boundary morphism

$$d_n : C_n \to C_{n-1}$$

By the first isomorphism theorem, we have $C_n / \ker(d_n) \cong \operatorname{im}(d_n)$, which implies

$$\#K_n = \dim(C_n) = \dim(\operatorname{im}(d_n)) + \dim(\ker(d_n))$$

By definition of homology, we have $H_n(K; \mathbb{Q}) = \ker(d_n) / \operatorname{im}(d_{n+1})$, which implies

$$b_n = \dim(H_n(K; \mathbb{Q})) = \dim(\ker(d_n)) - \dim(\operatorname{im}(d_{n+1}))$$

and this reduces the proof to a simple computation:

$$\begin{aligned}
\chi(K) - \sum_{i=0}^{\infty} (-1)^n b_i &= \sum_{i=0}^{\infty} (-1)^n \#K_i - \sum_{i=0}^{\infty} (-1)^n b_i \\
&= \sum_{i=0}^{\infty} (-1)^n (\#K_i - b_i) \\
&= \sum_{i=0}^{\infty} (-1)^n (\dim(\operatorname{im}(d_n)) + \dim(\operatorname{im}(d_{n+1}))) \\
&= \dim(\operatorname{im}(d_0)) \\
&= 0
\end{aligned}$$

where we have used that the infinite sums are actually finite because K is finite. □

Corollary 4.2.3 (Euler's formula)**.** *Any polyhedron of which the surface is homeomorphic to a* 2*-sphere has Euler characteristic* 2*. This includes all convex polyhedra.*

Proof. As discussed earlier, the Euler characteristic of a polyhedron is the same as the Euler characteristic of the simplicial complex obtained by triangulating its surface. The surface of a tetrahedron has 4 vertices, 6 edges and 4 faces, so it has Euler characteristic 4 - 6 + 4 = 2. Given any polyhedron of which (a triangulation of) the surface is homeomorphic to a 2-sphere, its surface is also homeomorphic to that of a tetrahedron. In particular, it has the same Betti numbers which implies it has the same Euler characteristic: 2. □

4.2.1 The Euler Characteristic of a Finite Topological Space

Clearly, the formula for the Euler characteristic given in proposition 4.2.2 makes sense for arbitrary topological spaces, as homology is defined for them by singular homology. What could go wrong is that the homology groups might have infinite dimension, or there might be infinitely many of them with non-zero dimension. For all topological spaces that do not have those problems, this formula defines a version of the Euler characteristic which agrees with the Euler characteristic of simplicial complexes.

Definition 4.2.4. *Let X be a topological space, such that only finitely many singular homology groups $H_n(X;\mathbb{Q})$ are non-trivial, and all homology groups are of finite dimension. Then we define the* ***Euler characteristic*** *of X as*

$$\chi(X) = \sum_{i=0}^{\infty}(-1)^n b_i = b_0 - b_1 + b_2 - b_3 + \cdots$$

where b_i is the i-th Betti number of X with coefficients in $\mathbb{Q}$, i.e. the dimension of the vector space $H_i(X;\mathbb{Q})$. Notice that this sum has only finitely many non-zero terms because of our assumptions.

In particular, this definition is valid for every finite topological space. The Euler characteristic actually provides a first and prototypical example of a (weak) homotopy invariant of finite topological spaces, that can readily be computed via the order complex using McCord's theory. In this case, it can easily be seen that this is not the most efficient way to compute it, which will be shown in the rest of this chapter. This motivates the question if other (weak) homotopy invariants like the Betti numbers and the homology groups can be computed more efficiently than by going over to the order complex. This will be investigated in the next chapter.

For completeness' sake, let's describe an algorithm for computing the Euler characteristic of a finite topological space using McCord's theory. The input is a finite T_0 topological space X which can be represented by our own `Space` class (page 99), or by a poset, for example in Sage [22]. Any simplicial complex can be represented as a set of subsets, using `frozenset` in Python, or they can be represented by the `SimplicialComplex` class of Sage.

Algorithm 2 Euler Characteristic using McCord — Implementation on page 107

function EULER(X)
  Generate the order complex K of X.
  for n **in** $\{0, \cdots, \dim(K)\}$ **do**
    Count the number of n-simplices of K and store as $\#K_n$.
  end for
  return $\sum_{i=0}^{\infty}(-1)^n \#K_i$
end function

The reason that this is a bad algorithm is because the order complex K has 2^n simplices in the worst case, where n is the number of points in X. This worst case is attained when X is a chain. In that case, every subset is a chain, so the order complex is the power set. This implies that the above algorithm has a worst case space complexity of $\Theta(2^n)$, and the same worst case time complexity.

Instead of actually computing and storing the order complex, we can count the number of chains of each length by using a dynamic programming approach. This is a variant of the graph algorithm described in [8] chapter 24.2, pages 592-595. Let's first translate this problem to a problem of directed acyclic graphs (DAG's). Recall that we can regard a poset X as a directed graph by taking its associated graph $G(X)$ (definition 3.2.5). This is indeed an acyclic graph, by the antisymmetry property of the poset. We add two nodes to this graph. A *source* node and a *sink* node, where the source node is a node that has an outgoing edge to every other node, and the sink node has an incoming edge from every other node. This is equivalent to first adding a minimum and maximum to X, and then taking the associated graph.

Definition 4.2.5. *Let X be a finite poset. The* ***augmented poset*** *$\hat{X}$ is the poset obtained by adding a maximum and a minimum to X. Explicitly, $\hat{X} = X \cup \{0, 1\}$ (we assume $0, 1 \notin X$) equipped with the following partial order:*

$$x \leq_{\hat{X}} y \iff x \leq_X y \text{ or } x = 0 \text{ or } y = 1$$

Definition 4.2.6. *Let X be a finite poset. The* ***augmented graph*** *$\hat{G}(X) = (\hat{V}, \hat{E})$ is the graph obtained by augmenting the associated graph $G(X) = (V, E)$ as follows:*

$$\hat{V} = V \cup \{\text{source}, \text{sink}\}$$
$$\hat{E} = E \cup \{(\text{source}, v) \mid v \in \hat{V} \setminus \{\text{source}\}\} \cup \{(v, \text{sink}) \mid v \in \hat{V} \setminus \{\text{sink}\}\}$$

Equivalently, $\hat{G}(X) := G(\hat{X})$.

Proposition 4.2.7. *Let X be a finite poset. There is a bijective correspondence between the chains of length n in X and the paths of length $n + 1$ from source to sink in $\hat{G}(X)$.*

Proof. Let $x_1 < x_2 < \cdots < x_n$ be a chain of length n, then

$$\text{source } \to x_1 \to x_2 \to \cdots \to x_n \to \text{ sink}$$

is a path of length $n + 1$ in $\hat{G}(X)$ and vice versa. □

Definition 4.2.8. *Let $G = (V, E)$ be a directed graph and $v \in V$. A* ***direct predecessor*** *of v is a vertex v' such that $(v', v) \in E$.*

Given a finite poset X, any path from source to a vertex v of $\hat{G}(X) = (\hat{V}, \hat{E})$ first passes trough a direct predecessor v'. If we already know how many paths of length $n-1$ there are from source to v', then we know that there are at least that many paths of length n from source to v. If we know for any direct predecessor of v how many paths there from source to that predecessor, then we can just sum them together to obtain the number of paths of length n from source to v. Writing $G^n_{v,w}$ for the number of paths of length n from v to w, we have

Proposition 4.2.9. *Let X be a finite topological space. Let $\hat{G} := \hat{G}(X) = (\hat{V}, \hat{E})$ and $v \in \hat{V}$. We have*

$$\hat{G}^n_{\text{source},v} = \sum_{(v',v)\in\hat{E}} \hat{G}^{n-1}_{\text{source},v'}$$

for all $n \in \mathbb{N} \setminus \{0\}$.

This leads to the following recursive algorithm.

Algorithm 3 Counting Paths — Implementation on page 108

```
function COUNT_PATHS_RECURSION(G = (V, E), source, v)
                                    ▷ Returns a dictionary mapping from n to G^n_{source,v}
    dict[0] = 1 if v = source else 0
    dict_{v'} = COUNT_PATHS_RECURSIVE(G, source, v') for (v', v) ∈ E
    dict[i] = ∑_{(v',v)∈Ê} dict_{v'}[i − 1] for 1 < i ≤ #V
    return dict
end function
function COUNT_PATHS(G)
    return COUNT_PATHS_RECURSION(G, source, sink)
end function
```

We should make sure that we never call the function COUNT_PATHS_RECURSION twice with the same input, as this is redundant. This can be done by storing the inputs and outputs of previous function calls, and first checking if a given input is already stored. In Python, this can be done automatically, by decorating the recursive function with `@lru_cache`. Once we have taken care of this, we know that we will at most have one call of COUNT_PATHS_RECURSION per vertex of G. Each such call runs for all direct predecessors over all path lengths (at most $\#V$). The total number of direct predecessors for all vertices together is at most $\#E$. Therefore, the worst case time complexity of this algorithm is $\Theta(\#V + \#V\#E) = \Theta(\#V\#E)$.

As a consequence, we can compute the Euler characteristic of a finite topological space on n points in $\Theta(n^3)$ time. Indeed, we can run the above algorithm on $\hat{G}(X)$ which has (worst case) $\Theta(n^2)$ edges and $n+2$ vertices. Then, all we need to do is take the alternating sum of the number of the paths for each length (which is the same as the number of chains according to proposition 4.2.7), to obtain the Euler Characteristic in $\Theta(n^3)$ time. This is much better than $\Theta(2^n)$, but we can still do better.

In the previous computation, we first calculated how many chains of each length we have (and we did this in each vertex we encountered), to then take the alternating sum in the end. But, as it turns out, we can actually already take the alternating sum in each vertex. This avoids looping over all possible lengths of chains, reducing the complexity. We need some extra notation to adequately discuss this. For a finite directed graph $G = (V, E)$, we write

$$G_{v,w} = \sum_{n=0}^{\infty}(-1)^n G^n_{v,w}$$

Proposition 4.2.10. *Let X be a finite topological space. We have*

$$\chi(X) - 1 = \hat{G}(X)_{\textit{source, sink}}$$

Proof. Writing out the first two terms explicitly gives

$$\hat{G}(X)_{\text{source, sink}} = 0 - 1 + \sum_{n=2}^{\infty}(-1)^n \hat{G}^n(X)_{\text{source, sink}}$$

According to proposition 4.2.7, $\sum_{n=2}^{\infty}(-1)^n \hat{G}^n(X)_{\text{source, sink}}$ is nothing but the alternating sum of the number of n-simplices of the order complex of X, i.e. $\chi(X)$. □

We are interested in a recursion relation for $\hat{G}(X)_{\text{source, sink}}$. Not too surprisingly, we can obtain this recursion as a consequence of proposition 4.2.9.

Proposition 4.2.11. *Let X be a finite topological space. Let $\hat{G} := \hat{G}(X) = (\hat{V}, \hat{E})$ and $v \in \hat{V} \setminus \{\text{source}\}$. We have*

$$\hat{G}_{\text{source},v} = -\sum_{(v',v)\in\hat{E}} \hat{G}_{\text{source},v'}$$

Proof. The proof is mainly plugging in proposition 4.2.9 and rewriting a bit.

$$
\begin{aligned}
\hat{G}_{\text{source},v} &= \sum_{n=0}^{\infty}(-1)^n \hat{G}^n_{\text{source},v} \\
&= \sum_{n=1}^{\infty}(-1)^n \hat{G}^n_{\text{source},v} \\
&= \sum_{n=1}^{\infty}(-1)^n \sum_{(v',v)\in\hat{E}} \hat{G}^{n-1}_{\text{source},v'} && \text{proposition 4.2.9} \\
&= \sum_{(v',v)\in\hat{E}} \sum_{n=1}^{\infty}(-1)^n \hat{G}^{n-1}_{\text{source},v'} \\
&= \sum_{(v',v)\in\hat{E}} \sum_{n=0}^{\infty}(-1)^{n+1} \hat{G}^{n}_{\text{source},v'} \\
&= -\sum_{(v',v)\in\hat{E}} \sum_{n=0}^{\infty}(-1)^{n} \hat{G}^{n}_{\text{source},v'} \\
&= -\sum_{(v',v)\in\hat{E}} \hat{G}_{\text{source},v'}
\end{aligned}
$$

□

This leads to the following recursive algorithm.

Algorithm 4 Dynamic Euler characteristic — Implementation on page 108

```
function EULER_RECURSION(G = (V, E), source, v)
                                                    ▷ Returns G_{source,v}
    if v = source then
        return 1
    else
        G_{source,v'} = EULER_RECURSION(G, source, v') for (v', v) ∈ E
        return − ∑_{(v',v)∈Ê} G_{source,v'}
    end if
end function
function EULER(X)
    return EULER_RECURSION(Ĝ(X), source, sink) + 1
end function
```

If we again take care that we never call EULER_RECURSION twice on the same input, then we have at most $\#V$ calls to it. This time, each call only needs to run over the direct predecessors of v, giving a total (worst case) time complexity of $\Theta(\#V + \#E)$. Running EULER on a poset X with n points than has a worst case time complexity of $\Theta(n + n^2) = \Theta(n^2)$. Much better than the $\Theta(n^3)$ algorithm that counts all chains.

Computing the Euler characteristic of a space X gives us some information about the Betti numbers of X. If X is T_0 and it has no chains of length 3 or more, then the Euler characteristic even determines the Betti numbers completely. To show why, we use this standard fact about homology (found in for example [11] proposition 2.7):

Proposition 4.2.12. *Let X be a topological space. The number of path-connected components of X is equal to* $\dim(H_0(X;\mathbb{Q})) = b_0$

Proposition 4.2.13. *Let X be a finite T_0 topological space with k path-connected components. If X has no chains of length 3 or more, then its Betti numbers are given by*

$$b_n = \begin{cases} k & \textit{if } n = 0 \\ k - \chi(X) & \textit{if } n = 1 \\ 0 & \textit{else} \end{cases}$$

Proof. The case $n = 0$ is just proposition 4.2.12. The case $n \geq 2$ follows immediately from the fact that, if X has no chains of length 3 or more, its order complex has no simplices of dimension 2 or more, so all of the homology groups for $n \geq 2$ are trivial. For $n = 1$ we then find

$$\begin{aligned} \chi(X) &= k - b_1(X) + 0 - 0 \cdots \\ \implies b_1(X) &= k - \chi(X) \end{aligned}$$

□

We end this subsection by showing an interesting counter example. It shows that of we have two weak homotopy equivalent finite spaces X and Y, we do not necessarily have a weak homotopy equivalence $f : X \to Y$ or $f : Y \to X$. It is based on example 1.4.17 in [2]. We use much of the theory of finite topological spaces, and we use the Euler characteristic to distinguish weak homotopy types.

Example 4.2.14. *Let X be the finite suspension of a discrete space on 3 points (figure 4.1). Notice that X is a T_0 space, thus a poset. We can also regard the space X^{op} corresponding to the opposite poset. Thanks to the theory of McCord, we know immediately that X and X^{op} are weak homotopy equivalent. Indeed, they have the same order complex K (opposite posets have the same chains) so McCord gives weak homotopy equivalences*

$$X \leftarrow |K| \to X^{op}$$

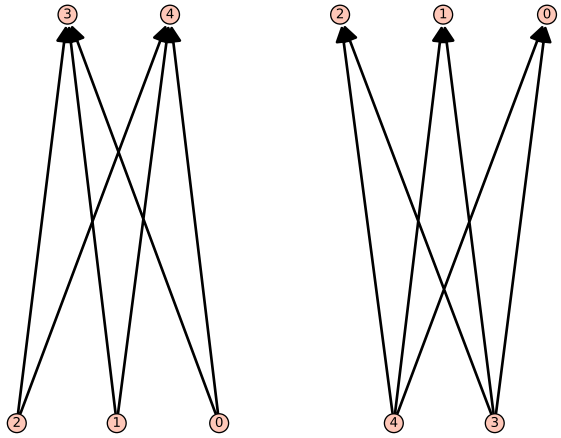


Figure 4.1: The Hasse diagram for the finite suspension of the discrete space on 3 points, and its opposite poset.

Now let's assume for the sake of contradiction that there is a weak homotopy equivalence $f : X^{op} \to X$ (the case $f : X \to X^{op}$ is completely dual to this one). This cannot be a bijection, as an order preserving bijection has to send minimal elements to minimal elements, and X has 3 minimal elements while X^{op} has only 2. This means our weak homotopy equivalence factors in an injective and surjective map, giving the commutative diagram:

$$\begin{array}{ccc} X & \xrightarrow{\quad f \quad} & X^{op} \\ & \searrow \quad \nearrow & \\ & Y = f(X) & \end{array}$$

where $Y \subsetneq X$. Putting this whole diagram through the singular homology functor (we take coefficients in $\mathbb{Q}$), we get a commutative diagram

$$\begin{array}{ccc} H_1(X;\mathbb{Q}) & \xrightarrow{\quad f_* \quad} & H_1(X^{op};\mathbb{Q}) \\ & \searrow \quad \nearrow_{g} & \\ & H_1(Y;\mathbb{Q}) & \end{array}$$

where f_* is an isomorphism, because f is a weak homotopy equivalence (proposition 2.2.38). In particular, f_* is surjective, so the linear map

$$g : H_1(Y;\mathbb{Q}) \to H_1(X^{op};\mathbb{Q})$$

is surjective which implies

$$\dim(H_1(Y;\mathbb{Q})) \geq \dim(H_1(X^{op};\mathbb{Q}))$$

or in other words

$$b_1(Y) \geq b_1(X^{op}) = b_1(X)$$

One could compute by hand the first Betti number of every strict subspace Y of X to see that this is a contradiction, but we will do this computationally, using proposition 4.2.13 in Sage [22]. Here, we use our implementation of finite topological spaces (listings page 99) and our implementation of algorithm 4 (listings page 108).

```
sage:  %runfile topology.py
sage:  %runfile euler_dynamic.sage
sage:  X = discrete(3).suspension()
sage:  subspaces = [X.subspace(A) for A in powerset(X) if A != X.points]
sage:  print "b_1(X) =", 1 - euler_of_space(X)
b_1(X) = 2
sage:  print [len(Y.components()) - euler_of_space(Y) for Y in subspaces]
[0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
0, 1, 1, 1]
```

We see that every strict subspace Y has $b_1(Y) < b_1(X)$ which is a contradiction with the surjectivity of g. We conclude that there is no weak homotopy equivalence $f : X^{op} \to X$.

4.2.2 The Möbius Function of a Poset

In the literature (most notably [20]), algorithm 4 is usually derived in a different way, by taking a detour via the Möbius function.

Definition 4.2.15. *Let X be a finite poset. The* ***incidence algebra*** $\mathcal{I}(X)$ *of X is the set*

$$\{f : X \times X \to \mathbb{R} \mid x \not\leq y \implies f(x, y) = 0\}$$

equipped with the following multiplication:

$$(f * g)(x, y) = \sum_{x \leq z \leq y} f(x, z)g(z, y)$$

and pointwise addition.

Notice that when X is totally ordered, w.l.o.g. $X = \{0, 1, \cdots, n\}$, then $\mathcal{I}(X)$ is just the algebra of real valued upper triangular $n \times n$ matrices with matrix multiplication. The incidence algebra of a general poset can thus be seen as a generalisation of upper triangular square matrices, to index sets that are not necessarily totally ordered. The identity element of this algebra is usually called δ:

$$\delta(x,y) = \begin{cases} 1 & \text{if } x = y \\ 0 & \text{if } x \neq y \end{cases}$$

and a full upper triangle filled with ones is usually called ζ;

$$\zeta(x,y) = \begin{cases} 1 & \text{if } x \leq y \\ 0 & \text{else} \end{cases}$$

Inverting an upper triangular matrix is easy to do, so it shouldn't be too hard to compute the inverse of the ζ function:

Proposition 4.2.16. *Let X be a finite poset. There is an element $\mu \in \mathcal{I}(X)$ such that $\mu * \zeta = \delta$. The function μ is given by the recursion*

$$\mu(x,y) = -\sum_{x \leq z < y} \mu(x,z)$$

for $x < y$ with base case $\mu(x,x) = 1$ (and of course $\mu(x,y) = 0$ if $x \not\leq y$).

Proof. We calculate $\mu * \zeta$. If $x < y$, we have

$$\begin{aligned}(\mu * \zeta)(x,y) &= \sum_{x \leq z \leq y} \mu(x,z)\zeta(z,y) \\ &= \sum_{x \leq z \leq y} \mu(x,z) \cdot 1 \\ &= \sum_{x \leq z \leq y} \mu(x,z) \\ &= \sum_{x \leq z < y} \mu(x,z) + \mu(x,y) \\ &= \sum_{x \leq z < y} \mu(x,z) - \sum_{x \leq z < y} \mu(x,z) \\ &= 0 = \delta(x,y)\end{aligned}$$

where we plugged in the recursion relation for $\mu(x,y)$ in the last step. For $x = y$ we get

$$(\mu * \zeta)(x,x) = \sum_{x \leq z \leq x} \mu(x,z)\zeta(z,x) = \mu(x,x)\zeta(x,x) = 1 \cdot 1 = 1 = \delta(x,x)$$

and for $x \not\leq y$, everything is zero by definition. □

Definition 4.2.17. *Let X be a finite poset. The* ***Möbius function*** *on X is the function $\mu \in \mathcal{I}(X)$ such that $\mu * \zeta = \delta$.*

This version of the Möbius function is usually studied in combinatorics. We are interested in its relation to the Euler characteristic. The following proposition of Hall [10] shows that, if we can compute the Möbius function, we can also compute the Euler characteristic. In fact, because the recursion formula for the Möbius function naturally produces a $\Theta(n^2)$ algorithm to compute it, we immediately obtain a $\Theta(n^2)$ for the Euler characteristic. Here, we work the other way around, using our $\Theta(n^2)$ algorithm for the Euler characteristic to show the theorem of Hall.

Theorem 4.2.18 (Hall)**.** *Let X be a finite poset. We have*

$$\chi(X) - 1 = \mu_{\hat{X}}(0, 1)$$

where $\mu_{\hat{X}}$ is the Möbius function on the augmented poset $\hat{X}$ of X.

Proof. In [20] this is proven by algebraic means, but we have in fact already provided a proof in a bit of a different language in proposition 4.2.11. A simple translation yields the proof in this setting. For completeness, we will write it out here. We write $\hat{G} = \hat{G}(X) = G(\hat{X})$ where we identify 0 with source and 1 with sink. We will show inductively that

$$\hat{G}_{\text{source},y} = \mu_{\hat{X}}(0, y)$$

for all y. As a base case, $\hat{G}_{\text{source},\text{source}} = 1$ as there is one path of length 0 from source to source (and no other paths). By definition we also have that $\mu_{\hat{X}}(0, 0) = 1$, so the base case is valid. Now assume for all $z < y$ that $\hat{G}_{\text{source},z} = \mu_{\hat{X}}(0, z)$, than we have

$$\begin{aligned}
\hat{G}_{\text{source},y} &= - \sum_{(z,y)\in\hat{E}} \hat{G}_{\text{source},z} \\
&= - \sum_{z<y} \mu_{\hat{X}}(0, z) \\
&= - \sum_{0\leq z<y} \mu_{\hat{X}}(0, z) \\
&= \mu_{\hat{X}}(0, y)
\end{aligned}$$

which completes the induction. Now just choose $y = 1 =$ sink to obtain

$$\chi(X) - 1 = \hat{G}_{\text{source},\text{sink}} = \mu_{\hat{X}}(0, 1)$$

□

4.3 Divide-and-Conquer Computations of a Finite Topological Space

There is some interest in using finite topological spaces in the study of big data [12]. The idea is to look for a simple topological model of more complex geometrical objects. Topological queries about the complex geometrical object can then quickly be handled by the topological model, but this is only accurate if the model is *topologically consistent* with the geometrical object. The introduced notion of topological consistency requires the comparing of Betti numbers of posets/finite topological spaces. But how to compute them is left open by that article. Similar to the situation for the Euler characteristic, McCord's theory immediately provides an algorithm for this case, given an algorithm for the Betti numbers of a finite simplicial complex (algorithm 5).

Algorithm 5 Betti numbers using McCord Implementation on page 109

function BETTI(X)
  Generate the order complex K of X.
  return the Betti numbers of K.
end function

And just like for the Euler characteristic, we want to avoid this method because computing the order complex gives an exponential blow-up. In this section, we investigate a possible approach for avoiding the order complex.

4.3.1 Divide-and-Conquer Algorithms

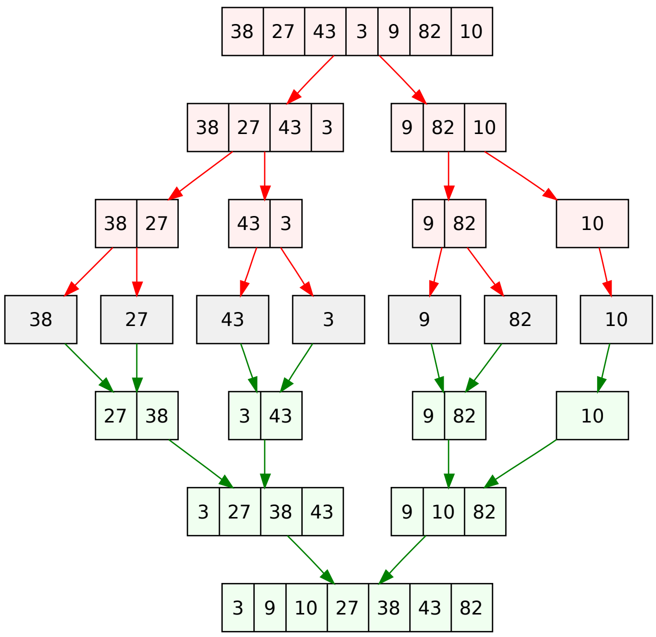


Figure 4.2: Merge sort: a divide-and-conquer sorting algorithm.

A divide-and-conquer algorithm (see for example [8]) is an algorithm that subdivides its input into two or more smaller inputs, for which it is easier to compute the desired output. If one keeps subdividing, the input might even become so small that the computation becomes trivial. The hard part of divide-and-conquer algorithms is combining the outputs to get the desired result.

Example 4.3.1 (Merge sort)**.** *Figure 4.2 shows a divide-and-conquer algorithm to sort a list of numbers. After enough divide steps, the lists are singletons and are trivially sorted. Two sorted lists can be merged together to one sorted list by iteratively selecting the biggest element of the two leftmost elements of the lists. After enough merge steps, a sorted version of the original list is obtained. A recursive implementation is given in algorithm 6.*

Algorithm 6 Merge sort

```
function SORT(list)
    if length of list = 1 then
        return list
    else
        split list in the middle into list1 and list2
        list1 ← SORT(list1)
        list2 ← SORT(list2)
        return MERGE(list1, list2)
    end if
end function
```

That double recursion might seem scary. The function calls itself twice every time it runs. This corresponds to the fact that each node in the red tree of figure 4.2 splits in two, which means that the number of nodes grows exponentially in each layer. This is not a problem tough, as each split cuts the list in half, so after about $\log_2(n)$ *splits, for a list of length* n*, the lists have length* 1*. In the end, the number of nodes in the tree will be in the order of* n*.*

In an unpublished article [7], one of the authors of [12] proposes a divide-and-conquer method for computing the Betti numbers of a finite topological space, without explicitly computing the order complex and thus avoiding the exponential blow-up that comes with it. The approach turned out to be flawed or at least incomplete, but the same approach can be used to compute the Euler characteristic of a finite topological space. We start with the divide step, which is the same in both situations. We then talk about the (incomplete) merge step for computing the Betti numbers and discuss what is still needed to make this approach work. Then we go to the merge step for computing the Euler characteristic and compare the complexity with other methods to compute it.

Given a finite T_0 topological space X, we want to subdivide it into some smaller topological spaces $X = X_1 \cup X_2$ in such a way that the homology groups of X_1 and X_2 tell us something about the homology groups of the original space. The exactness of the Mayer-Vietoris sequence

$$\begin{aligned}\cdots \to H_{n+1}(X) \;\to\; H_n(X_1 \cap X_2) \;\to\; H_n(X_1) \oplus H_n(X_2) \;\to\; H_n(X)\\ \to\; H_{n-1}(X_1 \cap X_2) \to \cdots \to H_0(X_1) \oplus H_0(X_2) \;\to\; H_0(X) \to 0\end{aligned}$$

provides such information in the case that X_1 and X_2 are open subspaces of X. In fact, to use this sequence to say something about X, we need information about X_1, X_2 and $X_1 \cap X_2$. This way of computing homology groups is not exclusive to finite spaces (in fact, we already used it on infinite spaces in section 3.4.1) but for finite spaces, we can recursively use it on X_1, X_2 and $X_1 \cap X_2$ as well. When subdividing those spaces, we end up with 9 spaces that are even smaller. If we keep subdividing them, we will at some point end up with homotopically trivial spaces because of finiteness, which can serve as a base case for the recursion. The choice of X_1 and X_2 that we will make is motivated by the desire to get the homotopically trivial spaces as fast as possible.

We know from the previously developed theory that a finite space is contractible if it has a single maximal element. So let's say that X has w maximal elements. If we subdivide X such that X_1 and X_2 each have half of the maximal elements, then after $\log_2(w)$ subdivisions, they will be contractible (notice we didn't say anything about the intersections $X_1 \cap X_2$ yet). Thus, we will subdivide as follows: let

$$X_{\max} = \{x \in X \mid \overline{\{x\}} = \{x\}\} = \{x \in X \mid x \text{ is maximal}\}$$

be the set of closed points, or equivalently maximal elements in X. Then we have the cover

$$X = \bigcup_{x \in X_{\max}} U_x$$

We split $X_{\max}$ into two disjoint parts of approximately equal cardinality:

$$X_{\max} = I \cup J, \quad I \cap J = \emptyset, \quad |I| - |J| \in \{-1, 0, 1\}$$

and we write

$$X_I = \bigcup_{x \in I} U_x, \quad X_J = \bigcup_{x \in J} U_x$$

This gives us the cover

$$X = \bigcup_{x \in X_{\max}} U_x = \bigcup_{x \in I} U_x \cup \bigcup_{x \in J} U_x = X_I \cup X_J$$

for which we can use the Mayer-Vietoris sequence. Notice that X_J and X_I have only half as many maximal elements as X, which was desired.

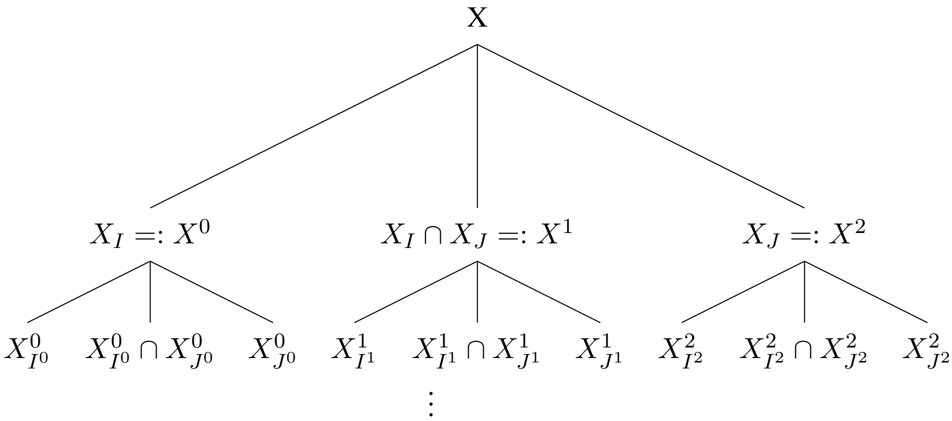
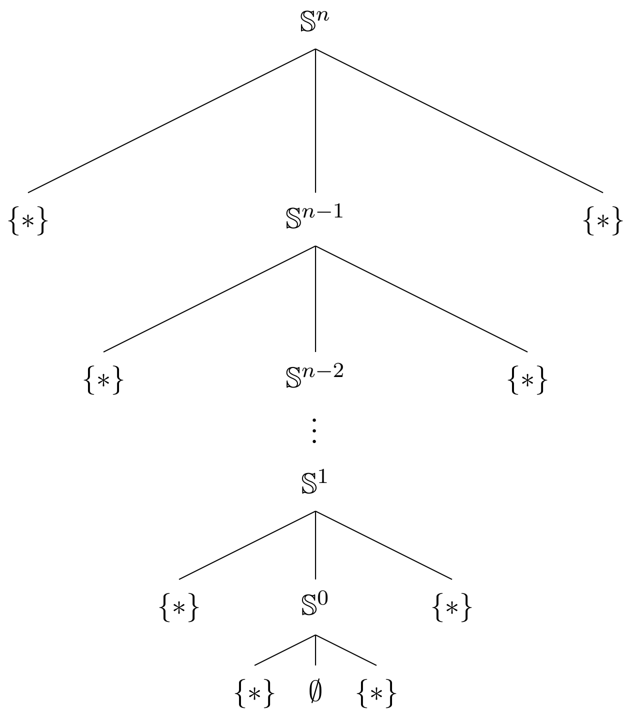


Figure 4.3: Subdividing an arbitrary finite T_0 space X

Example 4.3.2 (Pseudo spheres)**.** *Any pseudo sphere $\mathbb{S}^n$ has exactly two maximal elements, as it is obtained as a finite suspension (which always has exactly two maximal elements). This means that $\mathbb{S}^n_I$ and $\mathbb{S}^n_J$ are homotopy equivalent to a point, leading to the following subdivision tree:*



4.3.2 Betti Numbers

The idea presented in [7] is to use the above subdivision tree (figure 4.3) as a divide and conquer scheme to compute the Betti numbers (over some field F) of finite topological spaces. Indeed, the exactness of the Mayer-Vietoris sequence

$$0 \to \cdots \to H_{n+1}(X;F) \to H_n(X_I \cap X_J;F) \to H_n(X_J;F) \oplus H_n(X_J;F) \to H_n(X;F)$$
$$\to H_{n-1}(X_I \cap X_J;F) \to \cdots \to H_0(X_I;F) \oplus H_0(X_J;F) \to H_0(X;F) \to 0$$

provides partial information on how the Betti numbers of some node X the subdivision tree depend on its children X_I, X_J and $X_I \cap X_J$. The mentioned homology groups are 0 for n large enough because we work with finite topological spaces. As it is an exact sequence of F-vector spaces, it implies that the alternating sum of its dimensions is zero:

$$\begin{aligned} 0 = &+ \dim_F(H_0(X;F)) \\ &- \dim_F(H_0(X_I;F) \oplus H_0(X_J;F) \\ &\ldots \\ &+ \dim_F(H_{n-1}(X_I \cap X_J;F)) \\ &- \dim_F(H_n(X;F)) \\ &+ \dim_F(H_n(X_J;F) \oplus H_n(X_J;F)) \\ &- \dim_F(H_n(X_I \cap X_J;F)) \\ &+ \dim_F(H_{n+1}(X;F)) \\ &- \cdots \end{aligned}$$

which is in terms of the Betti numbers:

$$0 = \sum_{n=0}^{\infty}(-1)^n(b_n(X;F) - b_n(X_I;F) - b_n(X_J;F) + b_n(X_I \cap X_J;F))$$

Notice that this sum is finite if X is a finite topological space. So if we know the Betti numbers of X_I, X_J and $X_I \cap X_J$, this puts a linear relation on the Betti numbers of X.

Clearly, one linear relation is not going to be enough to determine all unknown Betti numbers of X (unless we know all Betti numbers but one). It seems that we need to extract more information out of the Mayer-Vietoris sequence to be able to determine the Betti numbers of X. If we could explicitly compute the (dimensions of the) images and kernels appearing in the Mayer-Vietoris sequence, we could split it into short exact sequences

$$0 \to \ker(\phi) \to H_n(X;F) \to \operatorname{im}(\phi) \to 0$$

$$\phi : H_n(X;F) \to H_{n-1}(X_I \cap X_J;F)$$

and compute the Betti numbers from them. One way to do this is to explicitly compute the maps ϕ in the Mayer-Vietoris sequence, using the order complex and McCord's theory, but this is of course exactly what we are trying to avoid. This leads to the divide and conquer algorithm presented in algorithm 7 of which the merge step is incomplete (or needs to use the order complex anyway).

Algorithm 7 Divide and Conquer Betti numbers

```
function BETTI(X)
    if |X_max| = 1 then                                   ▷ X is contractible
        return [1, 0, 0, ···]
    else if |X_max| = 0 then                              ▷ X is empty
        return [0, 0, 0, ···]
    else
        split X_max in half into I and J
        b_I ← BETTI(X_I)
        b_J ← BETTI(X_J)
        b_IJ ← BETTI(X_I ∩ X_J)
        return MERGE_BETTI(b_I,b_J,b_IJ)
    end if
end function
function MERGE_BETTI(a,b,c)
    Use Mayer-Vietoris sequence to find Betti numbers of node
    using the Betti numbers of its children (a,b,c)
                                ▷ This cannot be done yet without order complex
    return Betti numbers of node
end function
```

4.3.3 Euler Characteristic

The problem with algorithm 7 was the merge step. This is no longer an issue when we use this same algorithm to compute the Euler characteristic. Indeed, the Mayor-Vietoris sequence provided us the equation

$$\begin{aligned}
0 &= \sum_{n=0}^{\infty}(-1)^n(b_n(X;\mathbb{Q}) - b_n(X_I;\mathbb{Q}) - b_n(X_J;\mathbb{Q}) + b_n(X_I \cap X_J;\mathbb{Q})) \\
&= \sum_{n=0}^{\infty}(-1)^n b_n(X;\mathbb{Q}) - \sum_{n=0}^{\infty}(-1)^n b_n(X_I;\mathbb{Q}) - \sum_{n=0}^{\infty}(-1)^n b_n(X_J;\mathbb{Q}) + \sum_{n=0}^{\infty}(-1)^n b_n(X_I \cap X_J;\mathbb{Q}) \\
&= \chi(X) - \chi(X_I) - \chi(X_J) + \chi(X_I \cap X_J)
\end{aligned}$$

This shows how we can compute the Euler characteristic of any node X in the tree in figure 4.3 by using the Euler characteristic of its leaves X_I, X_J and $X_I \cap X_J$:

$$\chi(X) = \chi(X_I) + \chi(X_J) - \chi(X_I \cap X_J)$$

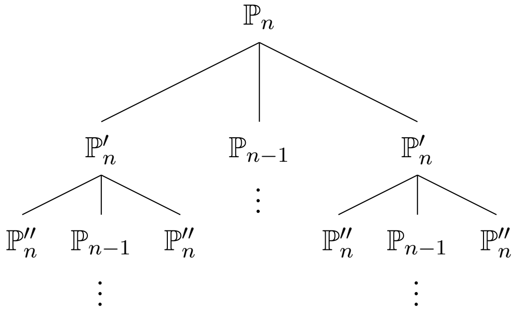
leading to algorithm 8.

Algorithm 8 Divide and Conquer Euler Characteristic — Implementation on page 107

function EULER(X)
  if $|X_{\max}|$ = 1 **then** ▷ X is contractible
    return 1
  else if $|X_{\max}|$ = 0 **then** ▷ X is empty
    return 0
  else
    split $X_{\max}$ in half into I and J
    return EULER(X_I) + EULER(X_J) - EULER($X_I \cap X_J$)
  end if
end function

How does this algorithm compare to the algorithms discussed in section 4.2? Similar to those algorithms, this algorithm has exponential worst case time complexity if we do not remember which inputs already have been computed. To execute this algorithm, we will have to call the function EULER of algorithm 8 as many times as there are nodes in the subdivision tree (figure 4.3). For pseudo spheres, the amount of nodes in this tree is linear in terms of the amount of points (example 4.3.2), but for general finite spaces it can be exponential, as is shown by the following example:

Example 4.3.3. *Let $\mathbb{P}_n$ be the poset obtained by stacking n copies of anti-chains of length 4 on top of each other (see figure 4.4). The subdivision tree of $\mathbb{P}_n$ can be computed recursively as*



Here $\mathbb{P}'_n$ is the poset $\mathbb{P}_n$ with two maximal elements removed, and $\mathbb{P}''_n$ is $\mathbb{P}_n$ with only one maximal element remaining (thus it is contractible and a leaf). If we denote the amount of nodes of $\mathbb{P}_n$ with $N(\mathbb{P}_n)$,

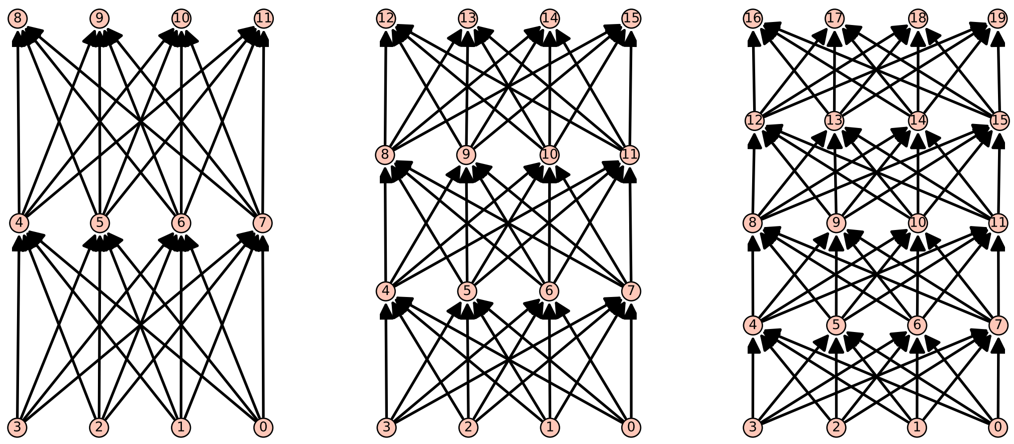


Figure 4.4: The Hasse diagram of $\mathbb{P}_3$, $\mathbb{P}_4$ and $\mathbb{P}_5$ respectively.

we see that

$$N(\mathbb{P}_n) = 3N(\mathbb{P}_{n-1}) + 7$$

Which shows that $N(\mathbb{P}_n) \in \Theta(3^n)$.

However, if we are more careful, and remember which inputs already have been computed before to avoid redundant computation, it becomes more difficult to find the time complexity of this algorithm. We are then interested in the number of (set-theoretically) distinct nodes in the subdivision tree. At the point of writing, it is still unclear to the author what the asymptotic behaviour of this number is. It might be exponential, rendering this algorithm and algorithm 7 not much more useful than using McCord's theory (algorithm 5). But for all sequences of spaces that have been tested so far, the complexity was polynomial. In other words, a counter example for the polynomiality of this algorithm has not yet been found.

Chapter 5

Conclusion and Future Research

After giving a quick introduction to the basics of algebraic topology, this thesis covered the main results that warrant finite topological spaces a place in algebraic topology. We covered:

- The connection to pre-ordered sets and posets of Alexandroff [1].
- Stong's homotopy theory for finite spaces [21].
- McCord's connection between finite spaces and finite simplicial complexes [16].

Instead of restating the results as they could be found in the literature, we tried to make some improvements here and there. In the section on Stong's theory (section 3.2), we introduced the use of inflationary and deflationary maps. Not only to highlight a common factor in many proofs using Stong's theory, but also to give an alternative definition of cores. This alternative definition makes the proof of Stong's classification theorem much easier. The idea of beat points is still useful, but this can be introduced later, as we did. In the section on McCord's theory (section 3.4), we combined the best parts of the proofs found in [2] and [14], hopefully resulting in a more readable exposition than those two sources together, and certainly much more readable than the original exposition of McCord [16].

Next, this thesis dealt with the computational aspects of finite topological spaces. We reintroduced Stong's beat points, not to define cores, but as a means to compute them. McCord's theory on the other hand, could be seen as a simple tool to compute homological invariants like homology groups, Betti numbers and the Euler characteristic. We showed how this gives simple working algorithms using Sage, but those algorithms are of exponential time complexity. As a better alternative for the Euler characteristic, we derived a $\Theta(n^2)$ algorithm using dynamic programming. The resulting algorithm is a known algorithm that is usually derived using the Möbius function and its relation to the Euler characteristic, of which we gave a complete exposition. We showed how our alternative derivation of that same algorithm can be interpreted as a proof for the relation between the Euler Characteristic and Möbius function.

Finally, we looked at a divide-and-conquer algorithm for the Betti numbers of finite spaces, which Bradley proposed in his pre-print article [7]. This article turned out to contain a couple of mistakes. We explained what is still needed to complete this algorithm, and we showed that the algorithm does work if we look at the easier problem of computing the Euler characteristic. Even then, computing the complexity of that algorithm is not easy. The divide step does not split a space into two disjoint parts like a normal divide-and-conquer method. Instead, it splits the space into two overlapping parts and its intersection. This extra space, the intersection, made it difficult to count the number of spaces encountered when executing this algorithm. We looked for a space to show that this number is exponential in the number of points of the space, but such an example was not found.

In future research, Bradley's algorithm can be further investigated in two directions. Effort can be put into completing Bradley's algorithm for Betti numbers, or in finding the complexity of Bradley's algorithm for the Euler characteristic. If the latter turns out to be exponential, then the algorithm for the Betti numbers is certainly exponential, so it might not be worth pursuing. Apart from Bradley's algorithm, research can be pointed towards other algorithms that compute homology groups or Betti numbers while avoiding McCord's exponential blow up. On the other hand, it might be possible that those algorithms do not exist, a proof of which also settles the issue.

Appendix A

Nederlandstalige Samenvatting

De studie van eindige topologische ruimten begon in 1937 met een artikel van Alexandroff [1] dat het verband legde tussen eindige topologische ruimten en orde theorie. Elke topologische ruimte (eindig of niet) bezit een natuurlijke preorde, genaamd de *specialisatie preorde*.

Definitie 3.1.3. *Stel* $(X, \mathcal{T})$ *een topologische ruimte. De* ***specialisatie preorde*** *op* X *wordt gegeven door*

$$x \leq_{\mathcal{T}} y \iff U_x \subseteq U_y$$

We noteren deze orde met $\leq_{\mathcal{T}}$ *of met* $\leq_X$ *als het duidelijk is welke topologie we gebruiken op* X*, of met* $\leq$ *als het duidelijk is op welke ruimte we werken.*

Voor Hausdorff ruimten en zelfs al voor T_1 ruimten is deze orde niet interessant. Daar vormt ze altijd een anti-keten, onafhankelijk van de topologie. Alle topologische informatie gaat dus verloren. Voor eindige topologische ruimten is deze orde wel interessant. Eindige topologische ruimten zijn bijna altijd zwakker gesepareerd dan T_1: de enige eindige T_1 ruimten zijn discreet. In dit eindige geval bevat deze orde genoeg informatie om de originele topologie er terug uit te construeren. De originele topologie is namelijk gelijk aan de *orde topologie* van de specialisatie preorde.

Definitie 3.1.10. *Stel* $(X, \leq)$ *gepreordende verzameling. De topologie op* X *voortgebracht door de basis*

$$\{\downarrow x \mid x \in X\}$$

wordt de ***ordetopologie*** *genoemd.*

Dit geeft een bijectief verband tussen eindige preordes en eindige topologische ruimten. Ook interessant is dat continue afbeeldingen door dit verband worden omgezet in orde bewarende afbeeldingen en omgekeerd. Het bijectief verband is dus een concreet categorie isomorfisme tussen de categorie van eindige topologische ruimten en eindige gepreordende verzamelingen. De T_0 ruimten komen hierbij net overeen met de posets. Hierna maken we geen onderscheid meer tussen eindige posets en eindige T_0 ruimten, of tussen eindige gepreordende verzamelingen en eindige topologische ruimten.

In 1966 maakte Stong [21] gebruik van deze orde theoretische kijk om homotopietheorie voor eindige ruimten te behandelen. Hij bekwam onder meer deze elegante karakterisatie van homotopie tussen afbeeldingen:

Gevolg 3.2.9. *Stel X en Y eindige topologische ruimten. Twee continue afbeeldingen $f, g : X \to Y$ zijn homotoop als en slechts als*

$$f = f_0 \leq f_1 \geq f_2 \leq \cdots \geq f_n = y$$

voor zekere $f_0, \cdots, f_n \in C(X, Y)$. Hier worden functies puntsgewijs geordend.

Een belangrijk speciaal geval hiervan zijn *progressieve* en *regressieve* afbeeldingen. Regressieve afbeeldingen worden in deze thesis gedefiniëerd als ordemorfismen $f : X \to X$ waarvoor geldt:

$$f \leq \mathrm{id}_X \text{ of equivalent } \forall x \in X : f(x) \leq x$$

en progressieve afbeeldingen worden duaal gedefinieerd. In de literatuur kunnen de definities soms verschillen, er wordt soms bijvoorbeeld een strikte ongelijkheid gevraagd, of de eis dat f een orde morfisme is wordt weggelaten.

Stong gebruikte zijn karakterisatie van homotopie in eindige ruimten om een mooi classificatie resultaat te tonen. Een belangrijke tool hierbij is het concept van *cores*. Wij geven in deze thesis een alternatieve definitie voor cores, anders dan die van Stong:

Definitie 3.2.16. *Een eindige T_0 topologische ruimte X wordt een* **core** *genoemd als er geen progressieve of regressieve functies $f : X \to X$ bestaan, verschillend van de identiteit.*

Het voordeel is dat onze definitie onmiddellijk tot zijn classificatie stelling leidt. Het voordeel van de definitie van Stong is dat het een meer combinatorische definitie is, dus makkelijker te berekenen. Daarom komt zijn definitie later in de thesis terug, bij de algoritmische beschouwingen (Hoofdstuk 4).

Elke eindige topologische ruimte X bezit een unieke core als deelruimte van hetzelfde homotopie type. Deze core noemen we dan de core van X. De classificatie stelling van Stong reduceert homotopie types tot homeomorfie klassen:

Stelling 3.2.21 (Classificatie stelling van Stong)**.** *Stel X en Y eindige topologische ruimten. Dan zijn X en Y homotopie equivalent als en slechts als hun cores homeomorf zijn.*

Rond dezelfde tijd bestudeerde McCord [16] eindige topologische ruimten vanuit een ander perspectief. Hij was niet geïnteresseerd in homotopie equivalenties tussen eindige topologische ruimten, maar wel in zwakke homotopie equivalenties tussen eindige en oneindige ruimten. Hij bekwam de volgende twee krachtige resultaten:

Stelling 3.4.18. *Elke eindige topologische ruimte is zwak homotopie equivalent met de meetkundige realisatie van een eindig simpliciaal complex.*

Stelling 3.4.23. *Elke topologische ruimte die homeomorf is aan de meetkundige realisatie van een eindig simpliciaal complex is zwak homotopie equivalent aan een eindige topologische ruimte.*

Deze twee stellingen bouwen een brug tussen de eindige en oneindige ruimten.

In het laatste hoofdstuk werd aandacht geschonken aan de computationele kant van de zaak. Om een algoritme te bekomen om cores te berekenen toonden we de equivalentie tussen onze definitie van cores en de originele definitie van Stong. Deze maakt gebruik van *beat* punten.

Definitie 4.1.1. *Stel X een eindige T_0 topologische ruimte. Een punt $x \in X$ wordt* ***downbeat*** *genoemd als zijn in-degree in het Hasse diagram van X gelijk is aan 1. Duaal wordt een punt $x \in X$* ***upbeat*** *genoemd als zijn out-degree in het Hasse diagram van X gelijk is aan 1. Een punt wordt* ***beat*** *genoemd als het upbeat of downbeat is.*

Een beat punt kan verwijderd worden uit de ruimte zonder het homotopie type te wijziging. Dit leidt onmiddellijk tot een algoritme om cores te berekenen, gebruik maken van Stongs definitie van cores:

Propositie 4.1.3. *Stel X een eindige T_0 topologische ruimte. Dan is X een core als en slechts als X geen beat punten heeft.*

De rest van de thesis wordt besteed aan het onderzoeken van algoritmen die bekende algebraïsche invarianten van eindige topologische ruimten berekenen (homologiegroepen, Betti-getallen, Euler-karakteristiek). De theorie van McCord geeft in al deze gevallen meteen een algoritme: bereken deze invarianten op het zwak homotopie equivalent eindig simpliciaal complex. Dit is jammer genoeg exponentieel, dus wordt er gezocht naar betere algoritmen.

Voor de Euler-karakteristiek bekomt men eenvoudig weg een kwadratisch algoritme via de Möbius functie en een stelling van Hall [10, 20]. Wij gebruiken in deze thesis een andere aanpak: We gebruiken variaties van dynamische algoritmen om paden te tellen in een DAG [8] om ook een kwadratisch algoritme voor de Euler-karakteristiek te bekomen. Dit algoritme blijkt gelijk te zijn aan het algoritme via de Möbius functie, en we bewijzen op die manier toevallig de stelling van Hall.

Voor de Betti-getallen stelde Bradley in 2018 [7] een verdeel-en-heers algoritme voor in een pre-print artikel. Dit algoritme zou de exponentiële blow-up van McCords stelling vermijden. Het algoritme werd in deze thesis onder de loep genomen. Jammer genoeg is het algoritme onvolledig en is de complexiteitsberekening van Bradley fout. Wanneer we terug gaan naar de eenvoudigere situatie van de Euler-karakteristiek leidt dit wel tot een volledig algoritme. Het algoritme verschilt van het voorgaande algoritme voor de Euler-karakteristiek en is in die zin wel interessant. Ook de complexiteit van dit algoritme is interessant aangezien het veel zou zeggen over de complexiteit van een vervollediging van Bradley's algoritme. Het blijkt jammer genoeg niet zo eenvoudig te zijn om deze complexiteit te berekenen. De verdeel stap van het algoritme deelt een ruimte namelijke niet op in twee disjuncte delen (zoals gebruikelijk bij een verdeel-en-heers algoritme), maar in twee overlappende delen en hun doorsnede. Er werd gezocht naar een voorbeeld dat toont dat het algoritme exponentieel is, maar dit werd niet gevonden.

Bibliography

[1] Pavel Alexandroff. Diskrete räume. *Matematičeskij sbornik*, 2(3):501–519, 1937.

[2] Jonathan A Barmak. *Algebraic topology of finite topological spaces and applications*, volume 2032. Springer, 2011.

[3] Blender Online Community. *Blender - a 3D modelling and rendering package*. Blender Foundation, Blender Institute, Amsterdam, 2017.

[4] Nicolas Bourbaki. Sur le theoreme de zorn. *Archiv der Mathematik*, 2(6):434–437, 1949.

[5] Nicolas Bourbaki. *General Topology: Chapters 1–4*, volume 18. Springer Science & Business Media, 2013.

[6] Aldridge Knight Bousfield and Daniel Marinus Kan. *Homotopy limits, completions and localizations*, volume 304. Springer Science & Business Media, 1972.

[7] Patrick Erik Bradley. A divide-and-conquer method for computing the betti numbers of finite topological spaces. *arXiv preprint arXiv:1806.01144*, 2018.

[8] Thomas H Cormen, Charles E Leiserson, Ronald L Rivest, and Clifford Stein. *Introduction to algorithms*. MIT press, 2009.

[9] Ralph H Fox. On topologies for function spaces. *Bulletin of the American Mathematical Society*, 51(6):429–432, 1945.

[10] P Hall. The eulerian functions of a group. *The Quarterly Journal of Mathematics*, 7:134–151, 1936.

[11] Allen Hatcher. *Algebraic Topology*. Cambridge University Press, 2002.

[12] Markus W Jahn, Patrick E Bradley, Mulhim Al Doori, and Martin Breunig. Topologically consistent models for efficient big geo-spatio-temporal data distribution. *ISPRS Annals of Photogrammetry, Remote Sensing & Spatial Information Sciences*, 4, 2017.

[13] John Lee. *Introduction to topological manifolds*, volume 940. Springer Science & Business Media, 2010.

[14] J. P. May. Finite spaces and larger contexts. Online, 2019.

[15] J Peter May. *Simplicial objects in algebraic topology*, volume 11. University of Chicago Press, 1992.

[16] Michael C McCord et al. Singular homology groups and homotopy groups of finite topological spaces. *Duke Mathematical Journal*, 33(3):465–474, 1966.

[17] Andrzej Mostowski. An undecidable arithmetical statement. In *Studies in Logic and the Foundations of Mathematics*, volume 93, pages 531–552. Elsevier, 1979.

[18] James R Munkres. *Topology*. Prentice Hall, 2000.

[19] James R Munkres. *Elements of algebraic topology*. CRC Press, 2018.

[20] Gian-Carlo Rota. On the foundations of combinatorial theory i. theory of möbius functions. *Probability theory and related fields*, 2(4):340–368, 1964.

[21] Robert E Stong. Finite topological spaces. *Transactions of the American Mathematical Society*, 123(2):325–340, 1966.

[22] The Sage Developers. *SageMath, the Sage Mathematics Software System (Version 8.6)*, 2019. `https://www.sagemath.org`.

[23] Ernst Witt. Beweisstudien zum satz von m. zorn. herrn erhard. schmidt zum 75. geburtstag gewidmet. *Mathematische Nachrichten*, 4(1-6):434–438, 1950.

[24] Mohamad Bakri Zubir. A gentle and friendly guide to the bourbaki-witt theorem and the zorn's lemma. *Discovering Mathematics (Menemui Matematik)*, 39(1):8–21, 2017.

Code Listings

Python Code

topology.py

This file contains the `Space` class which represents a finite topological space in Python 2.7. This class can represent T_0 spaces and non T_0 spaces. It can check if the space is T_0, and if not, it can make it T_0 without changing the homotopy type. If the space is T_0, an alternative is to represent it with the `Poset` class of Sage [22]. This file also contains a `Map` class to represent maps between topological spaces. This can be used to experiment with function spaces. Finally, it contains some basic examples of finite topological spaces, like the trivial and discrete space on n points, and the pseudo n-spheres.

```
import itertools as it

class Space:
    """
    An object of this class represents a finite topological space.

    Args:
        base: a dict mapping each point of the space to its
            smallest neighbourhood (set)

    Atrributes:
        base: a dict mapping each point of the space to its
            smallest neighbourhood (set)
        points: the points of the space (dict_keys)
        top_base: topological base of the space (dict_values)
    """

    def __init__(self, base):

        # The point set and for each point its smallest neighbourhood
        self.base = base
        self.points = base.viewkeys()
        self.top_base = base.viewvalues()

    def __repr__(self):
        return "Space(" + repr(self.base) + ")"

    def __len__(self):
        return len(self.points)

    def __contains__(self, item):
```

```
        return item in self.points

    def __iter__(self):
        return iter(self.points)

    def is_base_union(self, G):
        """
        Checks if a set can be written as union of
        basis elements
        """
        U = set()
        U.update(*[self.base[point] for point in G])
        return G == U

    def validate(self):
        """
        Checks if the base generates a valid topology
        """
        axiom1 = all(p in self.base[p] for p in self.points)
        axiom2 = self.is_base_union(self.points)
        pairs = it.combinations(self.base.values(), 2)
        axiom3 = all(self.is_base_union(A & B) for A, B in pairs)
        return axiom1 and axiom2 and axiom3

    # interior functions
    def in_interior(self, a, G, complementation = False):
        if complementation:
            return not self.in_closure(a, self.points - G)
        else:
            return self.base[a] <= G

    def is_open(self, G, complementation = False):
        if complementation:
            return self.is_closed(self.points - G)
        else:
            return all(self.in_interior(point, G) for point in G)

    def interior(self, G, complementation = False):
        if complementation:
            return self.points - self.closure(self.points - G)
        else:
            int_G = set()
            for point in G:
                if self.in_interior(point, G):
                    int_G.add(point)
            return int_G

    def topology(self):
        return list(filter(self.is_open, powerset(self.points)))

    # closure functions
    def in_closure(self, a, G, complementation = False):
        if complementation:
            return not self.in_interior(a, self.points - G)
        else:
            return not self.base[a].isdisjoint(G)

    def is_closed(self, G, complementation = False):
        if complementation:
            return self.is_open(self.points - G)
```

```
        else:
            return not any([self.in_closure(p, G) for p in self.points - G])

    def closure(self, G, complementation = False):
        if complementation:
            return self.points - self.interior(self.points - G)
        else:
            closure_G = G.copy()
            for point in self.points:
                if self.in_closure(point, G):
                    closure_G.add(point)
            return closure_G

    def closed_sets(self):
        return list(filter(self.is_closed, powerset(self.points)))

    def subspace(self, subset):
        base = {p: self.base[p] & subset for p in subset}
        return Space(base)

    def T0(self):
        """
        checks if the space is T0
        """
        pairs = it.combinations(self.points, 2)
        base = self.base
        return all(a not in base[b] or b not in base[a] for a, b in pairs)

    def make_T0(self):
        """
        returns the kolmogorov quotient: a T0 space with the same
        homotopy type as self
        """

        # indecernability relation
        R = lambda x, y: x in self.base[y] and y in self.base[x]

        subset = []
        points = set(self.points)
        while len(points) > 0:
            x = next(iter(points))
            subset.append(x)
            equivalence_class = [x]
            for y in points:
                if R(x,y):
                    equivalence_class.append(y)
            points -= set(equivalence_class)
        return self.subspace(set(subset))

    def component(self, a):
        """
        returns the connected component of a
        """
        comp = self.base[a].copy()
        for _ in range(len(self.points)):
            for open in self.top_base:
                if not comp.isdisjoint(open):
                    comp.update(open)
        return comp
```

```
    def components(self):
        """
        returns the list of connected components of self
        """
        comps = []
        points = set(self.points)
        while len(points) > 0:
            x = next(iter(points))
            comp = self.component(x)
            points -= comp
            comps.append(comp)
        return comps


    def is_connected(self):
        """
        checks if the space is connected
        """
        some_point = next(iter(self.points))
        return self.component(some_point) == self.points

    def endomorphisms(self):
        """
        returns all continuos maps from the space to itself
        """
        functionspace = it.product(self.points, repeat=len(self.points))
        index = {p: i for i, p in enumerate(self.points)}
        endos = []
        for tup in functionspace:
            function = {p: tup[index[p]] for p in self.points}
            f = Map(self, self, mapping = function)
            if f.is_continuous():
                endos.append(f)
        return endos

    def cone(self, a = None):
        """
        returns the non-Hausdorff cone of the space
        this requires a new point a
        """
        if a == None:
            a = 0
            while a in self.points:
                a += 1
        if a in self.points:
            raise ValueError("Point already in space")
        else:
            base = self.base.copy()
            base[a] = self.points.union({a})
            return Space(base)


    def suspension(self, a = None, b = None):
        """
        returns the non-Hausdorff suspension of the space
        this required two new points: a and b
        """
        if a == None:
            a = 0
            while a in self.points:
```

```
                a += 1
        if b == None:
            b = 0
            while b in self.points or b == a:
                b += 1
        if a in self.points:
            raise ValueError("Point already in space")
        elif b in self.points:
            raise ValueError("Point already in space")
        elif b == a:
            raise ValueError("The given points are equal")
        else:
            base = self.base.copy()
            base[a] = self.points | {a}
            base[b] = self.points | {b}
            return Space(base)

    def product(self, *others):
        points = set(it.product(self.points,
                                *[other.points for other in others]))
        base = {}
        for point in points:
            V0 = self.base[point[0]]
            Vi = [others[i-1].base[point[i]] for i in range(1, len(point))]
            V = set(it.product(V0, *Vi))
            base[point] = V
        return Space(base)

    def coproduct(self, *others):
        spaces = [self] + list(others)
        base = dict()
        for i, other in enumerate(spaces):
            indexed = lambda V: {(p, i) for p in V}
            base.update({(p, i): indexed(other.base[p]) for p in other.points})
        return Space(base)

    def __mul__(self, other):
        return self.product(other)

    def __add__(self, other):
        return self.coproduct(other)

    def simplify(self):
        """
        Returns a homeomorphic copy of the space, with
        the names of the points changed to the first
        n integers.
        """
        index = {point: i for i, point in enumerate(self.points)}
        indexed_neigh = lambda V: {index[point] for point in V}
        base = {index[p]: indexed_neigh(self.base[p]) for p in self.points}
        return Space(base)

    def is_downbeat(self, a, check_T0=True):
        """
        Checks if a point is downbeat, this requires a T0 space.
        For non-T0 spaces, use make_T0 first.
        """
        if check_T0:
            if not self.T0():
```

```
                raise ValueError("Space is not T0, use make_T0() first")

        U = self.subspace(self.base[a] - {a})

        # downbeat iff U has exactly one closed point
        return len([p for p in U.points if U.is_closed({p})]) == 1

    def is_upbeat(self, a, check_T0=True):
        """
        Checks if a point is upbeat, this requires a T0 space.
        For non-T0 spaces, use make_T0 first.
        """
        if check_T0:
            if not self.T0():
                raise ValueError("Space is not T0, use make_T0() first")

        F = self.subspace(self.closure({a}) - {a})

        # upbeat iff F has exactly one open point
        return len([p for p in F.points if len(F.base[p]) == 1]) == 1

    def is_beat(self, a, check_T0=True):
        """
        Checks if a point is beat, this requires a T0 space.
        For non-T0 spaces, use make_T0 first.
        """
        if check_T0:
            if not self.T0():
                raise ValueError("Space is not T0, use make_T0() first")

        return self.is_downbeat(a, check_T0=False)\
               or self.is_upbeat(a, check_T0=False)

    def beat_points(self, check_T0=True):
        """
        returns the set of beat points, this requires a T0 space.
        For non-T0 spaces, use make_T0 first.
        """
        if check_T0:
            if not self.T0():
                raise ValueError("Space is not T0, use make_T0() first")

        return {p for p in self.points if self.is_beat(p, check_T0=False)}

    def beat_points_generator(self, check_T0=True):
        """
        same as beat_points, but returns a generator.
        This requires a T0 space. For non-T0 spaces, use make_T0 first.
        """
        if check_T0:
            if not self.T0():
                raise ValueError("Space is not T0, use make_T0() first")

        return (p for p in self.points if self.is_beat(p, check_T0=False))

    def core(self, check_T0=True):
        """
        returns the core of a space, this requires a T0 space.
        For non-T0 spaces, use make_T0 first.
        """
```

```
        if check_T0:
            if not self.T0():
                raise ValueError("Space is not T0, use make_T0() first")

        core = self

        while True:
            beat_points = core.beat_points_generator(check_T0=False)
            try:
                beat_point = next(beat_points)
            except StopIteration:
                break
            core = core.subspace(core.points - {beat_point})
        return core

    def is_contractible(self, check_T0=True):
        """
        checks if a space is contractible, this requires a T0 space.
        For non-T0 spaces, use make_T0 first.
        """
        if check_T0:
            if not self.T0():
                raise ValueError("Space is not T0, use make_T0() first")

        return len(self.core(check_T0=False)) == 1

class Map:
    """
    Objects of this class represent maps between two finite
    topological spaces.

    Atrributes:
        domain: Space object representing the domain
        codomain: Space object representing the codomain
        formula: function that maps points from domain to codomain
        mapping: dict that maps points from domain to codomain
    """

    def __init__(self, domain, codomain, formula = None, mapping = None):
        if mapping == None:
            self.formula = formula
            self.mapping = None
        else:
            self.mapping = mapping
            self.formula = lambda x: mapping[x]

        self.domain = domain
        self.codomain = codomain

    def __call__(self, input):
        if type(input) is not set and input in self.domain.points:
            pointInput = True
        elif type(input) is set and input <= self.domain.points:
            pointInput = False
        else:
            raise ValueError("Input not in domain")

        if pointInput:
            output = self.formula(input)
            if output in self.codomain.points:
```

```
                return output
            else:
                raise ValueError("Output not in codomain")
        else:
            output = {self.formula(x) for x in input}
            if output <= self.codomain.points:
                return output
            else:
                raise ValueError("Output not in codomain")

    def update_mapping(self):
        self.mapping = {x : self.formula(x) for x in self.domain.points}

    def is_continuous_in(self, point):
        image = self(point)
        W = self.domain.base[point]
        V = self.codomain.base[image]
        return self(W) <= V

    def is_continuous(self):
        return all(self.is_continuous_in(x) for x in self.domain.points)


# list of all subsets
def powerset(iterable):
    s = list(iterable)
    return map(set,it.chain.from_iterable(it.combinations(s, r)
                                          for r in range(len(s)+1)))


# n point spaces
def trivial(n):
    X = set(range(n))
    base = {p: X for p in X}
    return Space(base)


def discrete(n):
    X = set(range(n))
    base = {p: {p} for p in X}
    return Space(base)


def chain(n):
    X = set(range(n))
    base = {p: set(range(p + 1)) for p in X}
    return Space(base)


# pseudo n-sphere (2n+2 points)
def sphere(n):
    if n == 0:
        return discrete(2)
    else:
        return sphere(n-1).suspension()
```

divide_and_conquer.py

Implementation of algorithm 8.

```
from functools32 import lru_cache

def euler_dc(X, make_T0=True):
    if make_T0:
        Y = X.make_T0()
    else:
        Y = X

    @lru_cache(maxsize=None)
    def euler_recursion(subset):
        Z = Y.subspace(subset)

        # Get the maximal points (the closed points)
        Z_max = [p for p in Z.points if Z.is_closed({p})]

        if len(Z_max) == 1:
            return 1
        elif len(Z_max) == 0:
            return 0
        else:

            # subdivde
            middle = len(Z_max)//2
            I = Z_max[:middle]
            J = Z_max[middle:]
            Z_I = frozenset(set.union(*[X.base[p] for p in I]))
            Z_J = frozenset(set.union(*[X.base[p] for p in J]))

            # inclusion-exclusion principle
            return euler_recursion(Z_I)\
                   + euler_recursion(Z_J)\
                   - euler_recursion(Z_I & Z_J)

    return euler_recursion(frozenset(Y.points))
```

Sage Code

The following implementations were made using Sage [22].

euler_mccord.sage

Implementation of algorithm 2.

```
def euler_of_poset(X):
    K = X.order_complex()
    faces = K.faces()
    return sum((-1)^(n) * len(faces[n]) for n in [0..K.dimension()])

def euler_of_space(X):
    X_T0 = X.make_T0()
```

```
    P = Poset((X_T0.points, lambda y, z: y in X_T0.base[z]))
    return euler_of_poset(P)
```

count_paths.sage

Implementation of algorithm 3.

```
from functools32 import lru_cache
from itertools import product

def count_paths(G):
    """
    Returns a dictionary: for each n < len(G) the number of
    paths from 'top' to 'bottom' of length n.
    """

    # lru_cache makes sure the function doesn't get called
    # twice with the same input
    @lru_cache(maxsize=None)
    def count_paths_recursion(a, b):
        count_dict = dict()
        count_dict[0] = 1 if a == b else 0

        prev_count_dicts = []
        for edge in G.incoming_edges([b]):
            prev_count_dicts.append(count_paths_recursion(a, edge[0]))

        for i in range(1, len(G)+1):
            count_dict[i] = sum(d[i-1] for d in prev_count_dicts)

        return count_dict
    return count_paths_recursion('bottom', 'top')

def to_graph(X):
    """
    Returns the associated graph of a poset
    """
    return DiGraph([list(X), [(x,y) for x, y in product(X,X)
                    if X.is_less_than(x,y)]])

def euler_by_counting_paths(X):
    """
    Calculates the Euler Characteristic of Poset X by first
    counting the number of chains of each length and
    then taking the alternatning sum
    """
    G = to_graph(X.with_bounds())
    paths = count_paths(G)
    return sum((-1)^(n) * paths[n] for n in [2..len(G)])
```

euler_dynamic.sage

Implementation of algorithm 4.

```
from functools32 import lru_cache
```

```
def euler_of_poset(X):
    """
    Returns the euler characteristic of a poset X
    """

    # add minimal element 'bottom' and maximal element 'top'
    Xhat = X.with_bounds()

    # lru_cache makes sure the function doesn't get called
    # twice with the same input
    @lru_cache(maxsize=None)
    def euler_recursion(a, b):
        if a == b:
            return 1
        else:
            predecessors = [a] + Xhat.open_interval(a, b)
            return -sum(euler_recursion(a,c) for c in predecessors)

    return euler_recursion('bottom', 'top') + 1

def euler_of_space(X):
    X_T0 = X.make_T0()
    P = Poset((X_T0.points, lambda y, z: y in X_T0.base[z]))
    return euler_of_poset(P)
```

betti_mccord.sage

Implementation of algorithm 5.

```
def betti_of_poset(X):
    K = X.order_complex()
    return K.betti()

def betti_of_space(X):
    X_T0 = X.make_T0()
    P = Poset((X_T0.points, lambda y, z: y in X_T0.base[z]))
    return betti_of_poset(P)
```