Thése de Doctorat **en Informatique**

Embedded Graphs: Crossings and Decompositions

Niloufar Fuladi

Membres de jury:

Éric COLIN de VERDIÈRE	Membre invité	CNRS, Université Gustave Eiffel
Federica FANONI	Examinatrice	CNRS, Université Paris Est Créteil
Cyril GAVOILLE	Rapporteur	Université de Bordeaux
Alfredo HUBARD	Co-encadrant de thèse	Université Gustave Eiffel
Francis LAZARUS	Examinateur	CNRS, Université Grenoble Alpes
Arnaud de MESMAY	Directeur de thèse	CNRS, Université Gustave Eiffel
Jorge RAMÍREZ-ALFONSÍN	Rapporteur	Université de Montpellier
Ana RECHTMAN	Examinatrice	Université Grenoble Alpes

Université Gustave Eiffel
Laboratoire d'Informatique Gaspard-Monge (LIGM)
École Doctorale 532
Mathématiques et Sciences et Technologies de l'Information et de la Communication (Mstic)

to Women,
to Life,
to Freedom.

Abstract

In this thesis we investigate some topological problems on surfaces and on graphs embedded on them from a computational viewpoint. Specifically, we focus on decompositions of surfaces and crossing numbers of graphs. The main contributions of this thesis can be divided into three parts.

The first problem we tackle concerns finding short *canonical decompositions* for a surface: given a cross-metric surface, that is, a surface endowed with a discrete metric in the form of an embedded graph, we are interested in cutting it to a disk along a system of loops with a specific shape such that the length of the cut is short. We provide the first algorithm that computes a non-orientable canonical decomposition of length $O(gn)$ where g is the genus of the surface and n is the number of edges of the graph embedded on it. This decomposition confirms a special case of a conjecture of Negami on the joint crossing number of two embeddable graphs. This conjecture posits that for any cross-metric surface, short decomposition of any shape exists. We also provide a correction for an argument of Negami bounding the joint crossing number of two non-orientable graph embeddings. Furthermore, we provide a generalization of $O(g)$-universal shortest path metrics to non-orientable surfaces.

The second problem that we address concerns two instances of crossing numbers of graphs: the degenerate crossing number and the genus crossing number. Based on the observation that both of these crossing numbers can be interpreted in terms of certain embeddings in a non-orientable surface, Mohar conjectured that under certain conditions on graph embeddings these concepts coincide. We provide the first counterexample to one of Mohar's conjectures which concerns two graphs given by fixed embeddings. Conversely, we prove a structure theorem that shows that the conjecture holds for the majority of the two-vertex graph embeddings.

The third problem is motivated by a question on the existence of a universal shortest path metric. We investigate the minimal size of a family of curves on a surface that realizes all types of pants decompositions of the surface, i.e., for any pants decomposition of the surface, there exists a homeomorphism sending it to a subset of the curves in this family.

Keywords: Surface, Embedded graph, Computational topology, Crossing number, Pants decomposition

Résumé

Dans cette thèse, nous étudions des problèmes topologiques sur les surfaces et sur les graphes qui y sont plongés d'un point de vue algorithmique. En particulier, nous étudions des problèmes autour de décompositions de surfaces et de nombres de croisements de graphes. Les principales contributions de cette thèse peuvent être divisées en trois parties.

Le premier problème auquel nous nous attaquons consiste à trouver des décompositions canoniques courtes pour une surface : étant donnée une surface avec une métrique de croisements, c'est-à-dire une surface dotée d'une métrique discrète sous la forme d'un graphe plongé ; nous souhaitons la couper en un disque le long d'un système de boucles ayant une forme spécifique et telle que les boucles soient courtes. Nous fournissons le premier algorithme qui calcule une décomposition canonique non orientable de longueur $O(gn)$ où g est le genre de la surface et n est le nombre d'arêtes du graphe plongé. Cette décomposition confirme un cas particulier d'une conjecture de Negami sur le nombre de croisements conjoints de deux graphes plongeables. Cette conjecture postule que pour toute surface avec une métrique de croisements, une décomposition courte de n'importe quelle forme existe. Nous corrigeons également un argument de Negami limitant le nombre de croisements conjoints de deux graphes plongés non orientables. De plus, nous généralisons la métrique universelle de plus courts chemins $O(g)$ aux surfaces non orientables.

Le deuxième problème que nous abordons concerne deux exemples de nombres de croisements de graphes : le nombre de croisements dégénérés et le nombre de croisements de genre. À partir de l'observation que ces deux nombres de croisements peuvent être interprétés en termes de certains plongements sur une surface non orientable, Mohar a conjecturé que sous certaines conditions sur les plongements de graphes, ces concepts coïncident. Nous fournissons le premier contre-exemple à l'une des conjectures de Mohar qui concerne deux graphes donnés par des plongements fixes. Inversement, nous prouvons un théorème de structure qui montre que la conjecture tient pour la majorité des plongements de graphes à 2 sommets.

Le troisième problème est motivé par une question sur l'existence d'une métrique universelle de plus courts chemins. Nous étudions la taille minimale d'une famille de courbes sur une surface qui réalise tous les types de décompositions en pantalons de la surface, c'est-à-dire que pour toute décomposition en pantalons de la surface, il existe un homéomorphisme qui l'envoie sur un sous-ensemble de courbes de cette famille.

Mots-clé: Surface, Graphe plongé, Topologie Algorithmique, Nombre de croisement, Décomposition en pantalon

Contents

Acknowledgements

Firstly, I wish to express my heartfelt appreciation for the guidance, relentless support, and patience of my dear supervisors, Alfredo Hubard and Arnaud de Mesmay. Arnaud and Alfredo, your mentorship and expertise have been indispensable in helping me navigate challenges and overcome the obstacles that stood before me. I am profoundly grateful for your words of encouragement during moments of self-doubt, your patience when I stumbled, and your unwavering support through the highs and lows of this journey. I specially appreciate all the good cop- bad cop moments where your constructive feedback propelled my growth. Thank you for never losing faith in me and for gracing me with the honor of being your student (especially as your first). I could not have hoped for better supervisors and I know that I have been extremely lucky. You have been a source of inspiration for me and I wish I can be like you some day. Thanks "dudes".

I am deeply thankful to Cyril Gavoille and Jorge Ramírez-Alfonsín for accepting and dedicating their time to review this thesis and demonstrating a genuine interest in my work. I also extend my gratitude to the other members of my thesis committee— Federica Fanoni, Francis Lazarus, and Ana Rechtman—for their invaluable presence during the defense and their insightful questions and feedback. A special acknowledgment goes to Éric Colin de Verdière for accepting to be the formal supervisor at the beginning of my thesis, for attending the defense as a guest and for his support throughout my thesis.

I am extremely grateful to Hugo Parlier for our collaboration, our fruitful discussions, his genuine concern for my progress, and his unfaltering support. I would like to express my gratitude to Gelasio Salazar and Vincent Delecroix for their invitations to San Luis Potosí and Bordeaux, respectively, and for generously sharing their insights with me. Additionally, I would like to express my sincere gratitude to Marcus Schaefer, for always supporting me from the very first stages of my thesis while we have never met.

I feel fortunate to have Corentin and Jean as my academic brothers, whose friendship has made these years unforgettable. I am grateful for the laughter and the enjoyable journey we have shared. Corentin, thank you for our friendship that formed in a blink of an eye, for your companionship, moral support, and our countless gossip sessions. Jean, thank you for all the exciting memories that we made together despite the language barrier that was hard to overcome at first. My heartfelt thanks also extend to Loïc, and to my

office mates Aaron, Julien and Zéphyr, for creating a welcoming and friendly environment and to dear Corinne for all the nice chats and the time that we spent together.

I owe a great debt of gratitude to the Bateni family, whose support has been indispensable in reaching this point. Also I am thankful to my friend Mehdi, who helped me with the very first figures in this thesis.

Special thanks are due to my family and friends who have contributed to my upbringing and to reaching this milestone. To my parents, Fatemeh and Ali who always believed in me and did all they could to provide everything I needed to pursue my academic goals. I am grateful to Artemis and Mehdi for considering me as their child and always supporting me. To Shiva, for her heartwarming presence in my life and to my friends—Jalal, Samad, Hedieh, Siavash, and Amal—for the cherished memories. To Meliné, for the moral support and the enriching discussions which helped me understand that I am not alone.

Finally, I have to thank Arya, my best friend, for his enduring patience with me, for his unconditional love, his crazy ways for motivating me and protecting me from myself. Arya, thank you for being the amazing friend that you are, and here is to an even brighter and more exhilarating journey ahead!

Chapter 1

Introduction

Consider a strip of paper, twist it by half, and then connect the ends together. The resulting object is called a *Möbius band*. One intriguing characteristic of this band is its inherent one-sidedness: imagine a cockroach walking on it. After going once around the band, the cockroach ends up in the same point that it started but on the other side of the paper, see Figure 1.1. Similarly, the concepts of right and left are not defined globally on the Möbius band: what was at the right of the cockroach at a point on the band, after going once around it is now on its left. This absence of distinction between right and left characterizes such a surface as *non-orientable*.

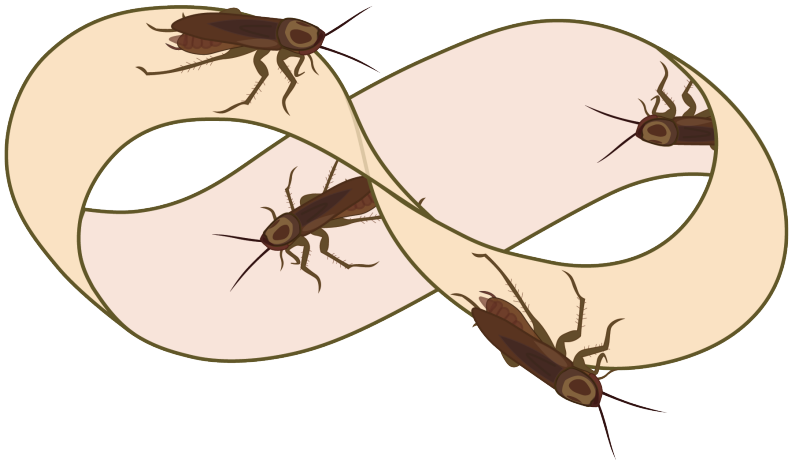

Figure 1.1: The figure depicts a Möbius band and the movement of the cockroach shows that this surface has only one side.[1]

In this thesis, we study algorithms and computational features of graphs embedded on surfaces with a special focus for non-orientable ones. Before defining all these notions and delving into the mathematical aspects of this thesis, let us first introduce certain

[1]This drawing is inspired by a graphic work by Maurits Cornelis Escher.

phenomena in human anatomy and DNA that exhibit some interesting non-orientable behaviour as well.

A phenomenon in human anatomy. Let us first explain how the visual system of humans work. When we look at an object, the lens in each of our eyes invert the image that is formed on the retina. The optic nerve fibers then transmit this image to the brain, where the retinal images from both eyes are fused together to create a congruent image that the brain perceives.

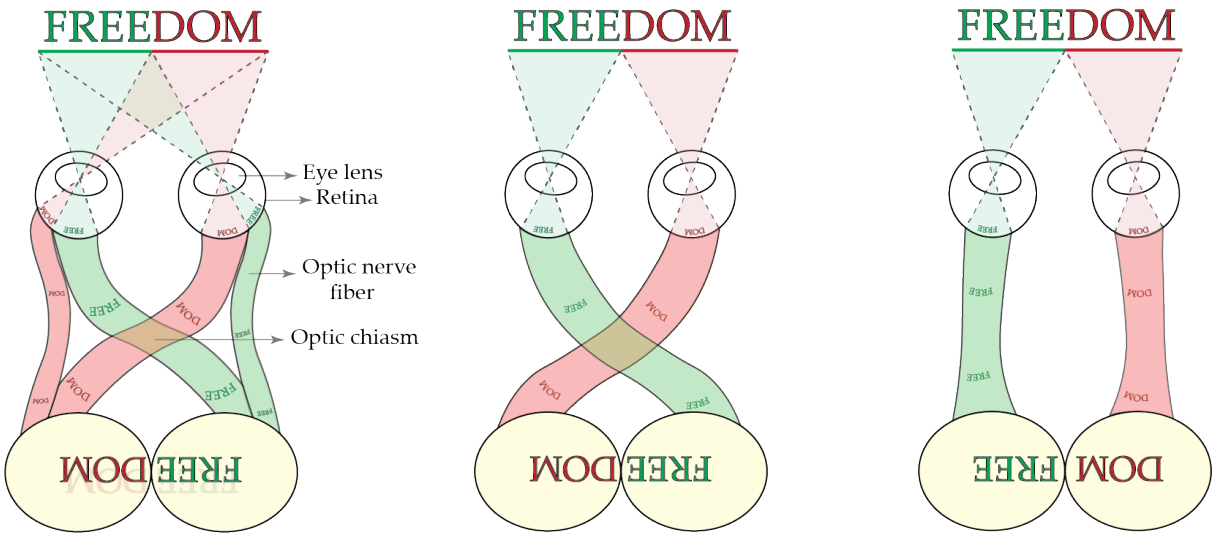

Figure 1.2: The left picture depicts how our optical system processes the information. The middle picture depicts an explanation for the crossing in the nerve fibers. The right picture shows the singularity that happens if the optical system was not crossed as it is. Note that in the middle and the right picture, for simplicity, it is assumed that the visual fields do not overlap.[2]

An interesting phenomenon in human anatomy is *decussation*, that is, the crossing of nerve fibers in the body. In the human visual system, the decussation of the optic nerve fibers happens in the optic chiasm. These nerve fibers cross over each other inside the chiasm, which reverses their mapping. This decussation is necessary to restore image continuity after fusing both retinal images in the brain, see the middle picture in Figure 1.2 (the crossing of optic nerve fibers is partial in the human visual system to deal with overlapping visual fields and restore a congruent image. The left picture in Figure 1.2 provides a more accurate depiction of the visual system, taking into account this partial crossing). If no crossing occurred, the brain would perceive a singularity in the image: for example the brain perceives "DOMFREE" instead of the word "FREEDOM", see the right picture in Figure 1.2.

[2]The ideas of the pictures in Figures 1.2 and 1.3 has been taken from [73] and [39], respectively.

The decussation of nerve fibers and map reversing of data also occur in many sites in the brainstem and spinal cord. For instance, the decussation of the pyramids involves the crossing over of motor pathways in the brainstem (see Figure 1.3), resulting in the left side of the brain controlling the right side of the body, and vice versa. The reason for this phenomenon is not yet completely understood, but a possible topological explanation is that without this crossing, it would result in a geometric singularity that would mix up information about left/right and up/down orientation in the brain (see [70]). We refer the reader to [73] for more details on these crossing phenomena in the human anatomy.

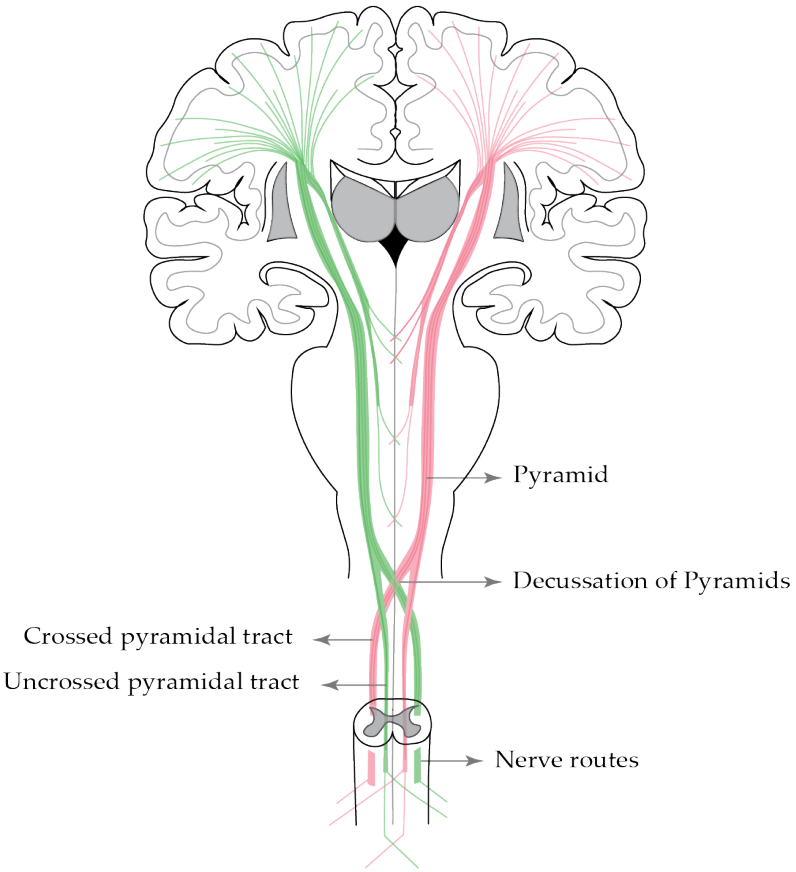

Figure 1.3: The picture depicts the nerve system in the brain and the decussation of the pyramids where the nerves cross.

A phenomenon in biology. At the end of the 1930s, after making extensive efforts to collect samples of fruit fly[3] genomes from various regions in the American West, Dobzhansky and Sturtevant [27] made a remarkable discovery. They found that genomes of flies from different regions differed by one or two blocks of genes in the sequence, and within each of these blocks, the linear order of genes was reversed. This discovery paved the way for the recognition of similar operations on genomes (such as block interchange, fusion,

[3]Drosophila

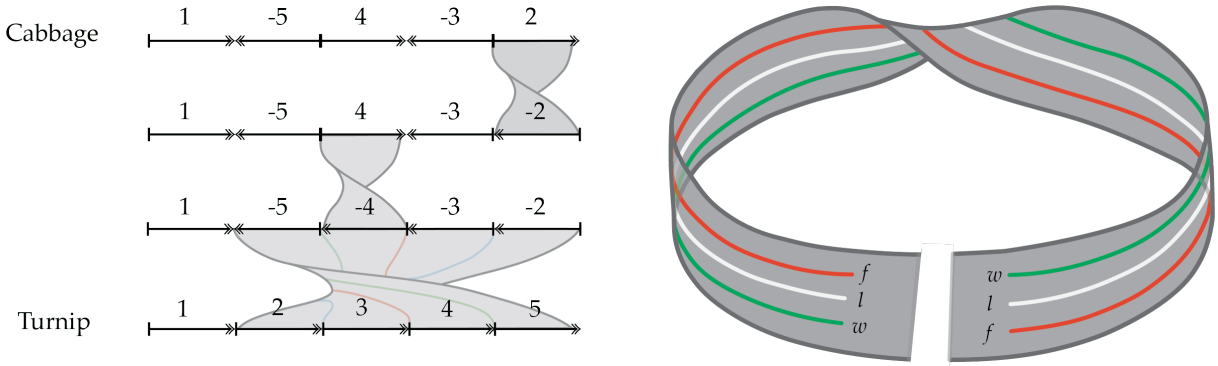

Figure 1.4: Left: the top (resp. bottom) sequence shows the gene sequence in the DNA of cabbage (resp. turnip). The figure shows that 3 reversals are enough to transform cabbage to turnip. Right: the picture depicts the reversing of the sequence of strands w, l, f when they go around the Möbius band.

fission, etc.) throughout the 20th century, leading to a new paradigm in the study of evolutionary changes in the DNA of living organisms: *genome rearrangements*.

By the 1980s, enough experimental data had been analyzed for Palmer and Herbon [62] to propose that the evolutionary distance between two species can be approximated by the number of reversals required to transform one gene sequence into another. At this point, the problem could be abstracted and handed over to a computer scientist: develop an algorithm that can reconstruct, within a reasonable amount of time, the minimum number of rearrangements needed to transition from one genome to another. Specifically, for unichromosomal genomes, this corresponds to finding the smallest number of reversals necessary to transform one genome into another. This algorithmic problem turned out to be central to this thesis. Figure 1.4 illustrates a minimal series of reversals between the genes in cabbage and turnip, where each reversal can be visualized as a flip or a twist in the gene sequence.

The crossing of nerve fibers at the midline of the human body and the reversals of gene sequences in the DNA share a common thread: both involve a change in left-right orientation in a symmetric object; the characterizing property in a Möbius band.

The decussation in nerve fibers and the reversals of gene sequences can be formalized by the trajectory of strands around a Möbius band as depicted in the right picture in Figure 1.4. In addition to their intriguing non-orientable behavior, the genome rearrangement algorithms derived from computational biology have proven to be invaluable in our study within this thesis.

The theme of this thesis

Any space that locally looks like the plane is called a *surface*. A surface is called *non-orientable* if it contains a Möbius band. This type of surface inherits the absence of a consistent notion of left and right from the Möbius band. Otherwise the surface is called *orientable*. Most surfaces that we deal with in our everyday life, like the Earth, are orientable. In fact, the consistency that helps us choose directions and read maps comes from the orientability of the Earth.

From the topological point of view, two surfaces are equivalent if we can transform one into the other through continuous deformation, without resorting to cutting or gluing. More accurately, two surfaces X and Y are equivalent if there exists a map $h : X \to Y$ such that h is bijective and both h and its inverse are continuous. The map h is called a *homeomorphism* and X and Y are called *homeomorphic*. For example consider a cube and a sphere. Although these surfaces may appear quite distinct in the three-dimensional space, they are actually homeomorphic surfaces in topology. On the other hand, a cylinder and a Möbius band are not homeomorphic and there is no way to transform one to the other without cutting and gluing once. This can also be seen from the fact that the Möbius band has only one boundary component while the cylinder has two.

In this thesis, we study structures such as graphs and curves that reside on surfaces. These graphs and curves endow our topological objects with a discrete structure. This structure provides a way to encode the complex topological data of the surface, allowing us to study them from an algorithmic perspective. To explain how a graph captures the topological structure of the surface, let us start by looking at an example. Imagine that we have a sphere that we want to cut to obtain a disk. One cut along any path is enough to achieve this, see Figure 1.5. However, not all surfaces are as simple as a sphere. Consider the surface of a donut (or a torus). We need at least two cuts to obtain a disk from the torus, see Figure 1.5. This observation implies that these two surfaces are topologically different. If we view the cuts on the surface as a graph, it can provide valuable insights into the structure of the surface. This can be formalized through the Euler characteristic.

Let G be a graph that is drawn on a surface S without self-intersections; such a drawing is called an *embedding* of G on S. An embedding of G on S is called *cellular*, if all the connected components of the complement of G, known as the *faces* of the embedding, are homeomorphic to disks. Let v, e, and f denote the number of vertices, edges, and faces in the cellular embedding of G on S, then $v - e + f$ is called the *Euler characteristic* of the embedding. Notably, this quantity is independent of the graph, i.e., any cellular graph embedding on this surface has the same Euler characteristic. Therefore we also call this quantity the Euler characteristic of the surface, denoted by $\chi(S)$. Furthermore, the Euler characteristic is a *topological invariant* of the surface: a property that is preserved under homeomorphisms. Showing that two surfaces have different Euler characteristics

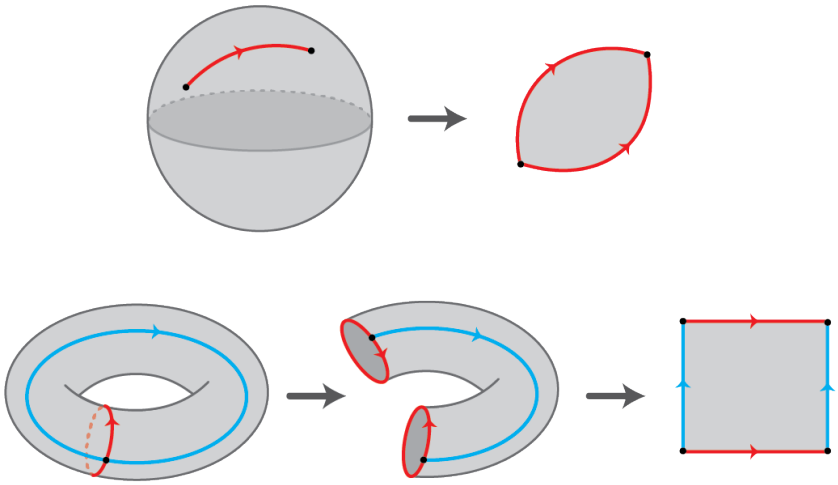

Figure 1.5: The minimum cuts to obtain a disk from a sphere and a torus.

imply that the surfaces are not homeomorphic. To see an example, let us compute the Euler characteristic of the sphere and torus by computing the Euler characteristic of the graph of our cut depicted in Figure 1.5. On the sphere, the graph has two vertices, one edge and one face, therefore, the Euler characteristic is equal to 2. On the other hand, the graph embedding on the torus has one vertex, two edges and one face which means that the Euler characteristic of the torus is equal to 0. This gives us a quantitative way to distinguish between these two surfaces.

Classification of surfaces. A fundamental theorem in topology, the classification theorem of surfaces, states that surfaces can be completely classified using two topological invariants: the orientability and the Euler characteristic.

From this classification, it follows that every surface can be obtained from a punctured sphere to which we glue Möbius bands (also called *cross-caps*) or *handles* (the torus with one puncture) along their boundary. The surface is orientable if it is obtained by attaching handles; the number of handles in this case is the *genus* of the surface. A surface that is obtained by only attaching Möbius bands is non-orientable and the genus of the surface is the number of Möbius bands that we glued to the sphere. In this sense, the number of punctures and handles or Möbius bands determine the Euler characteristic of the surface; namely, for a surface S, $\chi(S) = 2 - g - h$ where g is the genus of S and h is the number of punctures.

Now imagine you glue both a Möbius band and a handle to a sphere. Where does this surface reside in this classification? This surface is non-orientable because it contains a Möbius band and it can be shown that its Euler characteristic is -1 which is the same as the surface S that we obtain by gluing 3 cross-caps to a sphere ($\chi(S) = 2 - 3 = -1$), see Figure 1.6. As can be seen in this figure, visualising the homeomorphism between

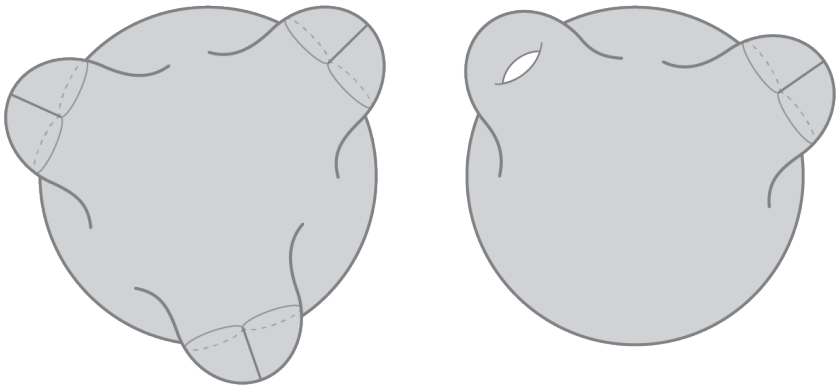

Figure 1.6: Two depictions of a non-orientable surface of genus 3.

these two surfaces is quite hard due to the fact that non-orientable surfaces without boundary cannot be embedded in the three dimensional space (see the discussion after Corollary 3.46 in [45]). Here, computational topology provides us with a framework to formalize and analyze this problem and allows us to compute such homeomorphisms.

Computational topology of surfaces. The field of computational topology provides computational methods and algorithms to address topological problems and analyze topological structures. Topological problems that are undecidable in general often admit efficient solutions when restricted to surfaces. Many basic problems in the computational topology of surfaces are obtained by taking a fundamental topological invariant and wondering how to compute it efficiently. Here we present a sampler of problems in computational topology of surfaces. We refer the reader to the surveys [20], [19] and [32] for more problems in this field.

To get started talking about algorithms, we need a discrete formulation of surfaces. Here, we consider surfaces that are obtained by gluing polygonal disks; we call such a surface a *combinatorial surface*. Equivalently, these polygonal disks can be looked upon as faces of a graph embedded on the surface, as we already introduced above.

1. **Testing homeomorphism.** Perhaps the most basic and natural question regarding topological surfaces is to check whether two surfaces are the same. This appeals to the classification of surfaces which we introduced above: two surfaces are homeomorphic if and only if their Euler characteristic and orientability match. Given two combinatorial surfaces, we can compute and compare their Euler characteristics as explained above. If the Euler characteristics of the surfaces are equal, then we proceed to test their orientability. This can be done by choosing an orientation for each face on the surface so that the orientations given to two faces that share an edge, induces opposite orientation on their common edge. If we can choose an orientation for all the faces on the surface that abides by this rule, then the sur-

face is orientable, otherwise it is non-orientable. If the orientability and the Euler characteristic of both surfaces match then they are homeomorphic.

2. **Computing the shortest cut graph.** As we explained above, any surface can be cut into a disk. The graph of this cut is called a *cut graph*. Looking for cut graphs on a surface is crucial for some applications as it simplifies the topology of the surface and makes it easier to work with. It is natural to turn this into an optimization problem and wonder what is the best way to do this cutting. For such optimization problems to be meaningful, the surface needs to be combined with a metric. The graph embedded on a combinatorial surface endows a discrete metric to the surface. In the model describing this metric, a curve is allowed to only use the edges of this graph and its length is the number of edges that it uses (throughout this thesis, we predominantly utilize an equivalent model referred to as the cross-metric model which we introduce in detail in the preliminaries).

   The problem of computing the shortest cut graph on a surface has been much studied recently [18, 17, 33] from various algorithmic angles (complexity, approximation and parameterized algorithms).

Now let us go back to the problem of visualizing a homeomorphism between the two representations of the non-orientable surface of genus 3, depicted in Figure 1.6. A common approach to obtain a homeomorphism between two surfaces is to cut them into a disk, put the disks in correspondence and extend the homeomorphism between the two disks to a homeomorphism between the two surfaces by gluing them back. For this approach to work, both cut graphs need to have the same shape. We can cut the surfaces depicted in Figure 1.6 along a one-vertex graph with 3 edges a, b and c such that their ends appear around the vertex with the order $aabbcc$. This way we obtain a hexagon in which each side corresponds to a copy of an edge a, b and c with the order $aabbcc$, see Figure 1.7. This graph gives a combinatorial representation of a non-orientable surface of genus 3 in which each edge of the graph encodes one of the cross-caps, see the leftmost picture in Figure 1.7. Cutting the graph along such a one-vertex graph is called a *canonical decomposition* of the surface. Let $aabbcc$ denote the hexagon we obtain by cutting one of the surfaces and $a'a'b'b'c'c'$ denote the other one. We can define a map between these hexagons such that a,b and c are mapped to a', b' and c', respectively. Gluing back the surfaces along the edges of the decomposition, we retrieve our surfaces and the map, defines a homeomorphism between the two.

Here again, it is natural to ask for efficient ways to canonically decompose a surface: given a combinatorial surface, can we always compute a canonical decomposition that is relatively short with respect to the metric of the surface? There exists an efficient algorithm that computes a short canonical decomposition of orientable surfaces [53]. For non-

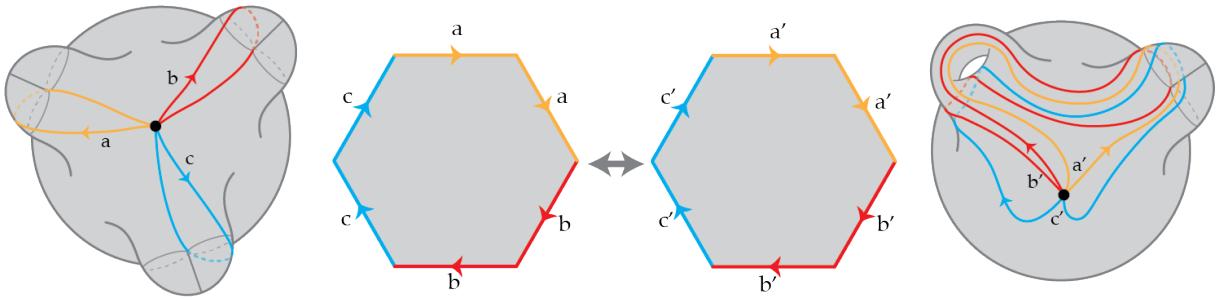

Figure 1.7: The homeomorphism between two non-orientable surfaces of genus 3 and the canonical decompositions of the surfaces.

orientable surfaces, there is an algorithms that computes a canonical decomposition [52], but this is not as short as in the orientable case.

Computing short canonical decompositions of non-orientable surfaces is one of the main results of this thesis. This problem illustrates perfectly the essence of our work, the computational approach in the study of a topological problem on surfaces and the interplay between topology and computation. Furthermore, this problem helps to illuminate the underlying topology in the genome rearrangement problem discussed above which in turn leads us to establish a fruitful connection with computational biology.

1.1 Specifics of the thesis and contributions

The predominant theme of this work is on studying decompositions of surfaces in which we strive to identify efficient ways to decompose a surface into pieces with simpler topology. We provide short decompositions of non-orientable surfaces and efficient algorithms to compute them. This investigation is closely intertwined with the second theme of this thesis, which centers around crossing numbers in graphs. Our work makes significant contributions to the following three topics which are summarized in Figure 1.8.

1) Decompositions of surfaces and joint crossing numbers

Decomposing a surface to simplify its topology is an important tool for studying the structure of the surface and has both practical and theoretical applications. As we mentioned before, in most parts of this thesis, our surfaces come with a geometric structure, that is given by a graph embedded on them. These graphs endow the surface with a notion of discrete metric. We use two ways to model this metric; the combinatorial model, which we briefly described above, and the cross-metric model (we refer the reader to Section 3.5.1 for more detailed introduction on these two models). Informally in the cross-metric model,

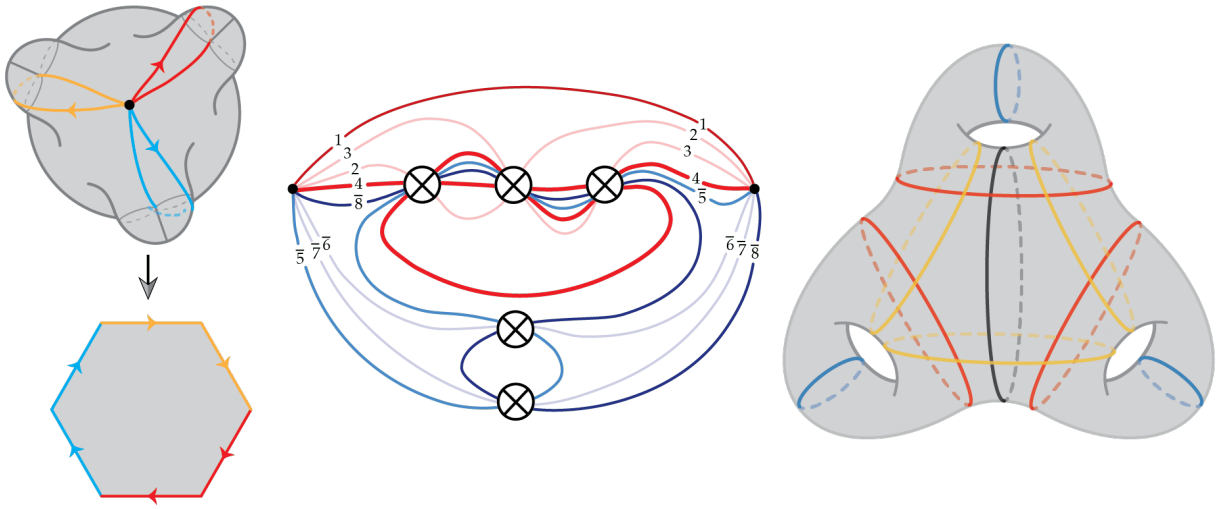

Figure 1.8: Left: can we always find a short canonical decomposition of a non-orientable surface? **Middle:** can we always find a planar drawing for a graph embedding such that $dcr(G) = gcr(G)$? In this picture $gcr(G) = 5$ but the edge 4 is not simple. Can we find a drawing for this graph with five crossing points $\otimes$ such that every curve is simple? **Right:** How many curves are needed to realize all pants decompositions of a surface? The figure shows 10 curves that realize all pants decomposition of the orientable surface of genus 3.

the length of the decomposition is given by the number of times that the graph of the metric is crossed by the graph of the decomposition where the crossings can only happen on the edges. On the other hand, in the combinatorial model, the graph of the decomposition lives on the edges of the embedded graph and the length of the decomposition is the number of edges of the graph that it uses (with multiplicity).

For applicational purposes, it makes sense to try to find a decomposition of a surface that has a short length. For example, an archetypal problem is to find a short canonical decomposition that canonically cuts the surface into a disk. The best known bounds for the lengths of such canonical decompositions of a surface of genus g is $O(g|E(G)|)$ where G is the graph of the metric and $|E(G)|$ is the number of edges in G and this is known for only few decompositions. Notably, this bound only exists for decompositions of orientable surfaces and no decomposition that achieves this bound in terms of minimizing length was known for non-orientable ones, see the left picture in Figure 1.8. One of the goals of this thesis is to provide a systematic understanding of these decompositions and provide algorithms to compute them.

From decompositions to joint crossing numbers of graphs. The problem of finding efficient decompositions of the surface is dually equivalent to investigating the best way to embed simultaneously two graphs on the same surface. More precisely, we consider the *joint crossing number* of a pair of embeddings: for two graphs G_1 and G_2 embedded

separately on a closed surface S, we are interested in minimizing the number of crossing points between $h(G_1)$ and G_2 over all the homeomorphisms $h : S \to S$ with the constraint that edges are only allowed to cross transversely. The following is a conjecture of Negami, which has been open for over 20 years.

Conjecture 1. (Negami's conjecture) *There exists a universal constant C such that for any pair of graphs G_1 and G_2 embedded on a surface S, the joint crossing number is at most $C|E(G_1)||E(G_2)|$.*

Negami's conjecture implies that efficient decompositions of any shapes exist, in the sense that each edge in the graph of the decomposition has length at most $O(|E(G)|)$ where G is the graph of the metric on the surface. Despite subsequent discoveries [3, 47, 66], this conjecture is still open. The best known bound for the joint crossing number of graphs is $O(g|E(G_1)||E(G_2)|)$, where g is the genus of the surface S (see [59]). At the level of decompositions, this readily implies that for any discrete surface and any decomposition that cuts the surface into a disk, we can decompose the surface such that the length of the decomposition is $O(g^2|E(G)|)$ where G is the graph of the metric; this bound is quite big for practical purposes.

Universal shortest path metrics. In an attempt to make progress on Negami's conjecture, Hubard, Kaluža, de Mesmay and Tancer [50] asked the following more geometric question.

Question 1. *Given a surface S of genus g, does there exist a Riemannian metric on S such that any simple graph embeddable on S can be embedded so that the edges are shortest paths on S?*

The existence of such a metric would imply Negami's conjecture: this is because any two graphs can be embedded such that their edges are shortest paths. In a Riemannian metric two shortest paths cross at most once which implies that any pair of edges of the graphs would cross at most once. (see Section 4.1 for a more detailed explanation).

Although Question 1 is not answered, a relaxation of it is answered positively in [50]. In this article, a universal Riemannian metric on an orientable surface of genus g was provided such that every graph embeddable on the surface can be embedded so that each edge is a concatenation of $O(g)$ shortest paths. It was also shown that the existence of this metric implies the bound $O(g|E(G_1)||E(G_2)|)$ for the joint crossing number of graphs proved in [59]. Nonetheless, the case of non-orientable surfaces was left as an open problem in this paper.

Contributions

In this thesis, we mainly focus on decompositions of non-orientable surfaces, as they are relatively unexplored compared to orientable ones.

- We provide some of the first short decompositions of non-orientable surfaces and polynomial algorithms to compute them by devising a new technique. See Theorems C, D and Section 6.3. In the proof of Theorem D, we leverage results from computational biology.
- We use this new technique to provide an alternative way to compute a short orientable canonical decomposition. See Theorem F.
- We provide a lower bound for canonical decompositions using a counting argument. See Theorem H.
- We correct the proof of Negami for the bound $O(g|E(G_1)||E(G_2)|)$ for joint crossing numbers of any two graphs G_1 and G_2, in the non-orientable case. See Theorem A.
- We generalize the construction of the universal metric introduced in [50] to the non-orientable setting. See Theorem B.

2) Degenerate crossing number vs. genus crossing number

We then shift our focus from decompositions toward crossing numbers for graphs. Crossing numbers are an important and popular tool in graph drawing and visualization (see [68] for a survey) where minimizing the crossing number is important in reducing the complexity of the drawings. A classical question in the study of crossing numbers is whether two different instances are equal. In this thesis we are interested in the following two crossing numbers. The *degenerate crossing number* of G, denoted by $dcr(G)$, first introduced by Pach and Tóth [61], is the smallest number of crossings among drawings of G in the plane such that the edges are simple arcs, the crossings are transversal and a crossing of multiple edges in a point is counted as one. If we allow edges to self-intersect, this defines the *genus crossing number* of G, denoted by $gcr(G)$, that was first introduced by Mohar [57]. Interestingly, these two crossing numbers find a translation in the realm of graphs embedded on non-orientable surfaces: the degenerate crossing number of G is equal to the smallest number of cross-caps needed on a surface to embed G so that each edge of G passes through each cross-cap at most once. On the other hand, the genus crossing number of G, is equal to the smallest number of cross-caps needed on a surface to embed G; here, an edge is allowed to enter the cross-caps more than once. It is evident

from the definition that $gcr(G) \leq dcr(G)$. We have the following conjecture of Mohar regarding these two crossing numbers.

Conjecture 2. ([57, Conjecture 3.1]) *For every simple graph G, $dcr(G) = gcr(G)$.*

Mohar also made a stronger conjecture that this also holds for any loopless graph embedding. Denote by $S^2 \setminus g\bigotimes$, the sphere minus g tiny disks, and by $(S^2 \setminus g\bigotimes)/\sim$ the space obtained by quotienting the boundary of each disk with the antipodal map. We call each $\bigotimes$ a cross-cap. Topologically, this amounts to gluing a Möbius band on each missing disk, thus yielding the non-orientable surface of genus g, denoted by N_g. We say that an embedded multi-graph $\phi\colon G \to N_g$ *admits a cross-cap drawing* $\phi'\colon G \to S^2$, if there is a homeomorphism $f\colon N_g \to (S^2 \setminus g\bigotimes)/\sim$ such that $f(\phi(G)) = \phi'(G)$. When an edge of the graph intersects the boundary of a cross-cap, we usually say that the edge *enters* the cross-cap. The equality of degenerate crossing number and genus crossing number of a graph G implies that the graph admits a cross-cap drawing with $dcr(G) = gcr(G)$ cross-caps in which each edge enters each cross-cap at most once. A *pseudo-triangulation* is a cellularly embedded multi-graph $\phi\colon G \to S$ in which each face has degree three. The following is the stronger conjecture of Mohar regarding pseudo-triangulations.

Conjecture 3. ([57, Conjecture 3.4]) *For any positive integer g, every loopless pseudo-triangulation of N_g admits a cross-cap drawing with g cross-caps in which each edge enters each cross-cap at most once.*

See the middle picture in Figure 1.8 for an illustration of a cross-cap drawing and a question to which this conjecture boils down.

Contributions

- We provide a loopless two-vertex graph embedding on a surface of genus 5 that represents a counterexample to Conjecture 3. See Theorem I.
- We prove a structure theorem that almost completely classifies the loopless two-vertex graph embeddings for which Conjecture 3 holds. This shows that most loopless two-vertex graph embeddings satisfy this conjecture. The proof makes direct use of an algorithm in computational biology. See Theorem J.

3) Enumerating curves and arcs on surfaces

Our results in the last part of this thesis have a rather different flavour compared to the previous ones as we do not care about the lengths of the curves and crossing numbers and contrary to the main focus of this work, we concentrate only on orientable surfaces. Our

motivation behind studying this problem is to make progress toward answering Question 1 on the existence of a universal shortest path metric.

We move our attention towards pants decompositions of orientable surfaces. A *pants decomposition* is an arrangement of closed curves on a surface that cuts it into pairs of pants (spheres with three holes). These pairs of pants provide fundamental building blocks in the study of surfaces. We investigate the minimal size of a universal family of curves that realizes all pants decompositions of the surface: a family of simple closed curves Γ on a surface *realizes all types of pants decompositions* if for any pants decomposition of the surface, there exists a homeomorphism sending it to a subset of the curves in Γ. We are interested in estimating the size of such a family and we provide upper and lower bounds for different instances of the problem, see the right picture in Figure 1.8. Especially in the case of surfaces without boundaries, further improvements might provide a negate answer to Question 1 on the existence of a shortest path metric on surfaces (see Section 8.1 for further details on this connection).

Contributions

- We provide an exponential upper bound and a superlinear lower bound on the minimal size of a family of curves that realizes all types of pants decompositions in the case of surfaces without punctures. See Theorem K.
- We provide upper and lower bounds in the case of surfaces with punctures which we can consider labelled or unlabelled. See Theorem L.
- We also investigate a similar concept of universality for triangulations of polygons, where we provide bounds which are tight up to logarithmic factors. See Theorems M and N.

1.2 Organization

In Chapter 3, we begin by introducing the various notions we will be using throughout this thesis. Our work uses tools from different fields of mathematics such as geometry, topology and topological graph theory. We also explain the related topic of genome rearrangements in bio-informatics. While we strive to provide references for most of the notions introduced within this chapter, it is worth noting that some concepts (such as box drawings and relations to genome rearrangements) are presented and formalized here for the first time. Nevertheless, we have diligently cited the sources from which the ideas have been derived.

Our results appear in Sections 4, 5, 6, 7 and 8 and except Sections 5 and 6 that are

closely related, the rest of the chapters can be read independently. Note that our main theorems are labelled by alphabetic letters throughout the thesis.

Chapter 4 concentrates on two results related to the joint crossing numbers. Both are generalizing results on orientable surfaces to the non-orientable setting. First, we provide a correction for an argument of Negami on the joint crossing number of graphs that has a minor issue in the case of non-orientable surfaces. Later, we provide a universal metric for non-orientable surfaces that allows every graph embeddable on the surface to be embedded such that every edge is a concatenation of $O(g)$ shortest paths. This proof involves generalizing an octagonal decomposition known for orientable surface to the non-orientable setting. Namely, Theorems A, B and C can be found in this chapter. The first result is contained in the preliminary version of [A] that is published in the Proceedings of the 38th Symposium on Computational Geometry. Both results appear in the journal version [A] that has been published in Discrete & Computational Geometry.

In Chapters 5 and 6, we initiate a thorough study of short topological decompositions on non-orientable surfaces. Chapter 5 contains our result on computing a canonical decomposition of non-orientable surfaces, in which we introduce a new approach. Namely Theorem D can be found in this chapter. This is the main result in [A] which appears in both the conference and the journal version. Chapter 6 contains unpublished results on other shapes of decompositions of both orientable and non-orientable surfaces. The results in this chapter mostly come from using our new approach to find other short decompositions. Also we provide a lower bound for canonical decompositions in this chapter. Namely, Theorems F and H can be found in this chapter.

Chapter 7 contains our results on a conjecture of Mohar about genus crossing numbers and degenerate crossing numbers. Namely, Theorems J and I can be found in this chapter. The results in this chapter appear in [C] which has appeared in the Proceedings of the 31st International Symposium on Graph Drawing and Network Visualization.

Chapter 8 contains our result on estimation of the size of the family of curves that realizes all types of pants decompositions. Namely, Theorems K, L, M and N can be found in this chapter. Results in this chapter appear in [B] which has been accepted in Israel Journal of Mathematics.

Finally, to conclude in Chapter 9, we summarize our main results and the remaining open problems and explain some future avenues of research arising from our work.

Chapitre 2

Introduction en Français

Prenez une bande de papier, faites lui une torsion d'un demi tour, puis collez les deux extrémités ensemble. L'objet qui en résulte est appelé *ruban de Möbius*. Une caractéristique intrigante de ce ruban est qu'il n'a qu'un seul côté : imaginez un cafard marchant sur le ruban. Après avoir fait une fois le tour du ruban, le cafard se retrouve au même endroit qu'au début, mais de l'autre côté du papier, voir la Figure 2.1. De même, les concepts de droite et de gauche ne sont pas définis sur l'ensemble du ruban de Möbius : ce qui était à la "droite" du cafard en un point de la bande, après en avoir fait le tour une fois, se trouve maintenant à son "gauche". Cette absence de distinction entre la droite et la gauche caractérise une telle surface comme étant *non orientable*.

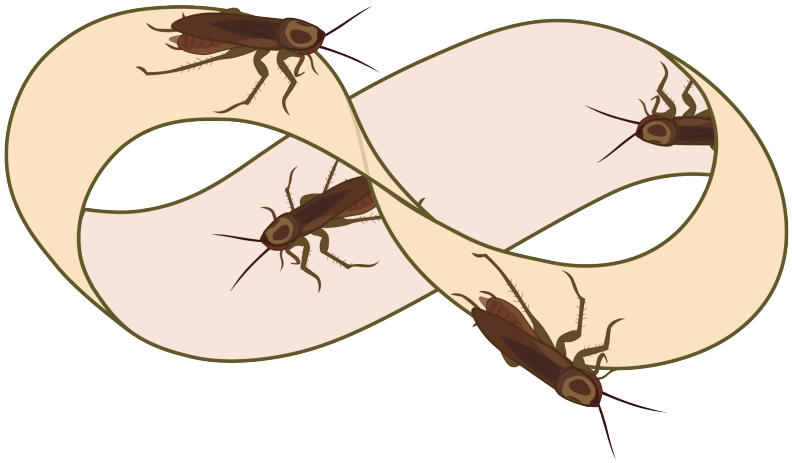

Figure 2.1 : Le cafard est capable de parcourir le ruban de Möbius en entier, ceci montre que cette surface n'a qu'un seul côté.

Dans cette thèse, nous étudions les algorithmes et le potentiel informatique issus des plongements de graphes sur des surfaces, plus particulièrement sur les surfaces non orientables. Avant de définir toutes ces notions et d'aborder les aspects mathématiques de cette thèse, commençons par présenter certains phénomènes d'anatomie humaine et de l'ADN

qui présentent un aspect non orientable intéressant.

Phénomène d'anatomie humaine. Expliquons d'abord comment fonctionne le système visuel humain. Lorsque nous regardons un objet, dans chacun de nos yeux, le cristallin inverse l'image de l'objet en la formant sur la rétine. Les fibres du nerf optique transmettent ensuite ces deux images rétiniennes au cerveau, qui les fusionne pour créer une image congruente qui est l'image perçue.

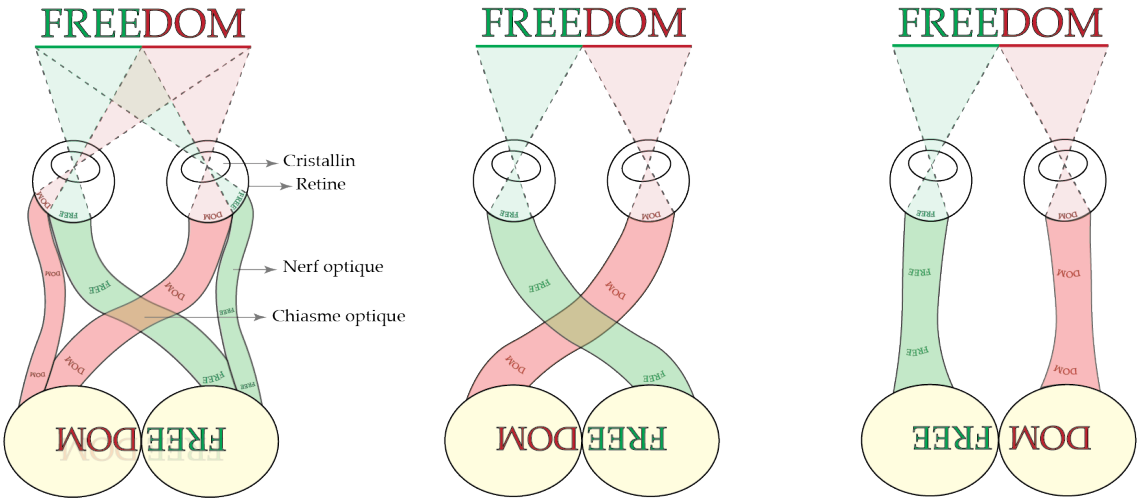

Figure 2.2 : L'image de gauche schématise le traitement de l'information par notre système optique. L'image du milieu explique le croisement des fibres nerveuses. L'image de droite montre la singularité qui se produirait si le système optique n'avait pas été croisé comme il l'est. Notez que dans les images du milieu et de droite, pour des raisons de simplicité, on suppose que les champs visuels ne se chevauchent pas.[1]

Un phénomène intéressant de l'anatomie humaine est la *décussation*, c'est-à-dire le croisement des fibres nerveuses dans le corps. Dans le système visuel humain, la décussation des fibres nerveuses optiques se produit dans le chiasme optique. Ces fibres nerveuses se croisent à l'intérieur du chiasme, ce qui inverse la position des images transmises. Cette décussation est nécessaire pour rétablir la continuité de l'image après la fusion des deux images rétiniennes dans le cerveau, voir l'image du milieu dans la Figure 2.2 (le croisement des fibres nerveuses optiques est partiel dans le système visuel humain pour rétablir une image congruente et gérer le chevauchement des champs de visions). L'image de gauche de la Figure 2.2 donne une représentation plus précise du système visuel, en tenant compte de ce croisement partiel). S'il n'y avait pas de croisement, le cerveau percevrait une singularité dans l'image : par exemple, il percevrait "DOMFREE" au lieu du mot "FREEDOM", voir l'image de droite dans la Figure 2.2.

La décussation des fibres nerveuses et l'inversion des données se produisent également

[1]Les idées des images des Figures 2.2 et 2.3 ont été tirées de [73] et [39], respectivement.

dans de nombreux sites du tronc cérébral et de la moelle épinière. Par exemple, la décussation des pyramides dans le tronc cérébral témoigne du croisement des fonctions motrices (voir Figure 2.3) : le côté gauche du cerveau contrôle le côté droit du corps, et vice versa. La raison de ce phénomène n'est pas encore totalement comprise, mais une explication topologique possible est que sans ce croisement, il en résulterait une singularité géométrique qui provoquerait le mélange des informations sur l'orientation gauche/droite et haut/bas dans le cerveau (voir [70]). Nous renvoyons le lecteur à [73] pour plus de détails sur ces phénomènes de croisements dans l'anatomie humaine.

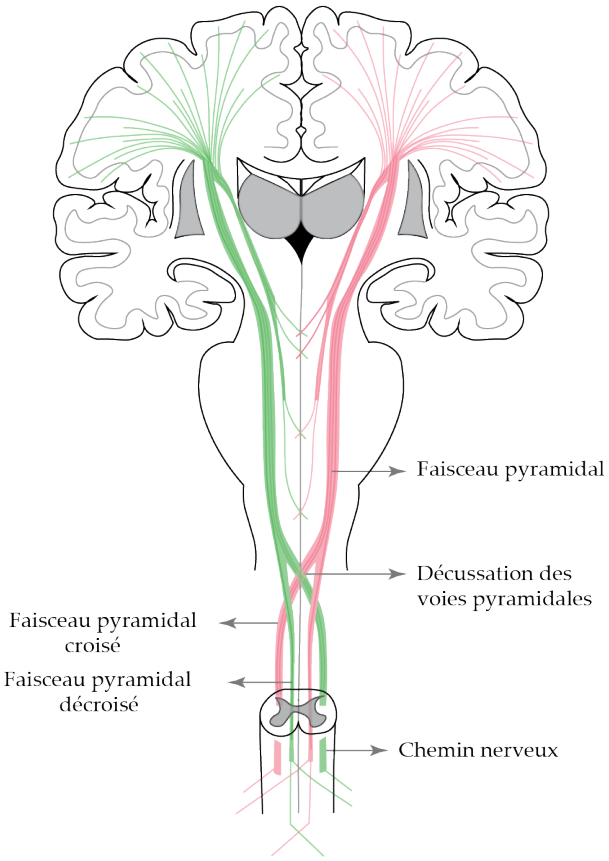

Figure 2.3 : L'image représente le système nerveux du cerveau et la décussation des pyramides où les nerfs moteurs se croisent.

Un phénomène biologique. À la fin des années 1930, après avoir déployé des efforts considérables pour collecter des échantillons de génomes de drosophiles provenant de diverses régions de l'Ouest américain, Dobzhansky et Sturtevant [27] ont fait une découverte remarquable. Ils ont constaté que le génome de mouches provenant de différentes régions différaient uniquement d'un ou deux blocs de gènes dans la séquence ADN, et qu'à l'intérieur de chacun de ces blocs, l'ordre linéaire des gènes était inversé. Cette découverte a ouvert la voie à la reconnaissance d'autres opérations similaires sur les génomes (telles que

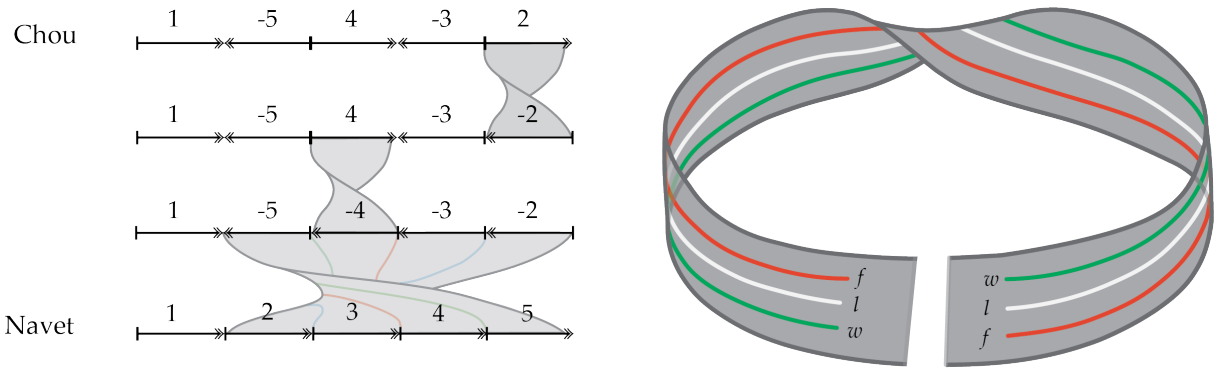

Figure 2.4 : À gauche : la séquence du haut (respectivement du bas) représente une séquence de gènes dans l'ADN du chou (respectivement du navet). Trois inversions suffisent pour la transformer en une séquence de gène de navet. À droite : l'image illustre l'inversion de la séquence w, l, f lorsqu'elle fait le tour du ruban de Möbius.

l'échange de blocs, la fusion, la séparation, etc.) tout au long du XXe siècle. Ceci a conduit à un nouveau paradigme dans l'étude, par l'ADN, de l'évolution : les *réarrangements du génome*.

Dans les années 1980, Palmer et Herbon ont analysé suffisamment de données expérimentales pour proposer l'idée que la distance évolutive entre deux espèces puisse être estimée par le nombre d'inversions nécessaires pour transformer une séquence de gènes en une autre. À ce stade, le problème peut être abstrait et confié à un informaticien : développer un algorithme capable de reconstruire, en un temps raisonnable, le nombre minimum de réarrangements nécessaires pour passer d'une séquence à l'autre. Plus précisément, pour les génomes unichromosomiques, cela correspond à trouver le plus petit nombre d'inversions nécessaires pour transformer un génome en un autre. Ce problème algorithmique s'est avéré être au cœur de cette thèse. La Figure 2.4 illustre une suite minimale d'inversions entre les séquences ADN du chou et du navet, où chaque inversion peut être visualisée comme un retournement ou une torsion dans la séquence du gène.

Le croisement des fibres nerveuses sur la ligne médiane du corps humain et les inversions de séquences génétiques dans l'ADN ont un point commun : tous deux impliquent un changement d'orientation gauche-droite dans un objet symétrique, ce qui est la propriété caractéristique du ruban de Möbius.

La décussation des fibres nerveuses et les inversions des séquences génétiques peuvent être formalisées par la trajectoire des brins autour d'une bande de Möbius, comme le montre l'image de droite de la Figure 2.4. Outre leur intrigant comportement non orientable, les algorithmes de réarrangement du génome dérivés de la bio-informatique se sont révélés inestimables dans le cadre de cette thèse.

Le thème de cette thèse

Tout espace qui ressemble localement au plan est appelé *surface*. Une surface est dite *non orientable* si elle contient un ruban de Möbius. Ce type de surface hérite de l'absence de notion cohérente de gauche et de droite. Dans le cas contraire, la surface est dite *orientable*. La plupart des surfaces que nous côtoyons dans notre vie quotidienne, comme celle de la Terre, sont orientables. En fait, la cohérence qui nous aide à choisir des directions et à lire des cartes provient de l'orientabilité de la surface de la Terre.

Du point de vue topologique, deux surfaces sont équivalentes si l'on peut transformer l'une en l'autre par une déformation continue, sans avoir recours à un découpage ou à un collage. Plus précisément, deux surfaces X et Y sont équivalentes s'il existe une application $h : X \to Y$ telle que h est bijective et que h et son inverse sont toutes deux continues. L'application h est appelée un *homéomorphisme* et X et Y sont appelés *homéomorphes*. Par exemple, considérons un cube et une sphère. Bien que ces surfaces semblent très distinctes dans l'espace tridimensionnel, il s'agit en fait de surfaces homéomorphes en topologie. En revanche, un cylindre et un ruban de Möbius ne sont pas homéomorphes et il n'y a aucun moyen de transformer l'un en l'autre sans couper et coller une fois. Ceci est également visible dans le fait que le ruban de Möbius n'a qu'un seul bord alors que le cylindre en a deux.

Dans cette thèse, nous étudions des structures telles que les graphes et les courbes sur des surfaces. Ces graphes et ces courbes donnent à nos objets topologiques une structure discrète qui permet d'encoder des données topologiques complexes de la surface afin de les étudier d'un point de vue algorithmique. Pour expliquer comment un graphe capture la structure topologique de la surface, commençons par un exemple. Imaginons que nous ayons une sphère que nous voulons couper pour obtenir un disque. Une coupe le long d'un chemin qui ne s'auto-intersecte pas suffit pour y parvenir, voir la Figure 2.5. Cependant, toutes les surfaces ne sont pas aussi simples qu'une sphère. Considérons la surface d'un donut (appelée tore). Il faut au moins deux coupes le long de tels chemins pour obtenir un disque à partir du tore, voir la Figure 2.5. Cette observation implique que ces deux surfaces sont topologiquement différentes. En définissant un graphe à partir de ces coupes de la surface, nous pouvons obtenir des informations précieuses sur sa structure. Ceci peut être formalisé par la caractéristique d'Euler.

Soit G un graphe dessiné sur une surface S sans auto-intersections ; un tel dessin est appelé un *plongement* de G sur S. Un plongement de G sur S est appelé *cellulaire* si toutes les composantes connexes du complément de G, appelées *faces* du plongement, sont homéomorphes à des disques. Soit v, e et f le nombre de sommets, d'arêtes et de faces dans le plongement cellulaire de G sur S. Alors $v - e + f$ est appelé la *caractéristique d'Euler* du plongement. Notamment, cette quantité est indépendante du graphe, c'est-à-dire que tout graphe cellulairement plongé sur cette surface a la même caractéristique d'Euler.

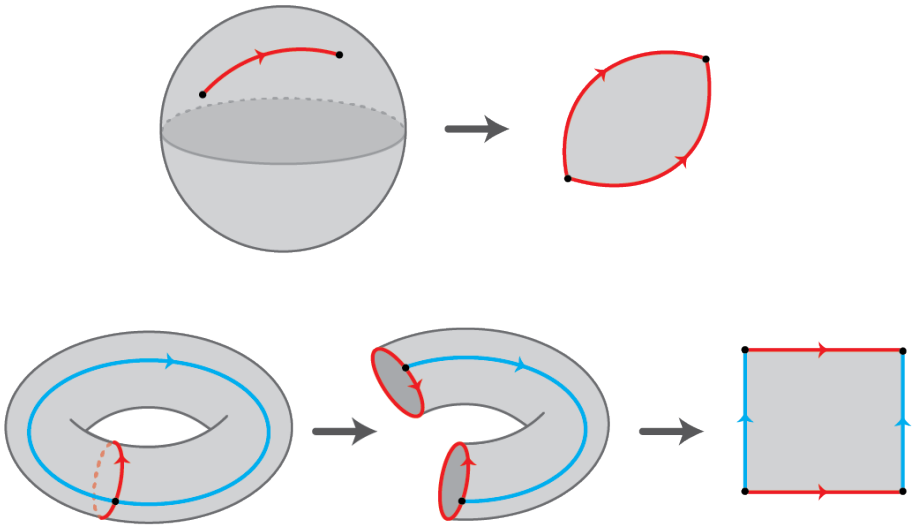

Figure 2.5 : Des coupes minimales le long de chemins sans auto-intersection pour obtenir un disque à partir d'une sphère et d'un tore.

Par conséquent, nous appelons également cette quantité la caractéristique d'Euler de la surface, dénotée par $\chi(S)$. De plus, la caractéristique d'Euler est un *invariant topologique* de la surface : c'est une propriété qui est préservée par les homéomorphismes. Montrer que deux surfaces ont des caractéristiques d'Euler différentes implique donc qu'elles ne sont pas homéomorphes. Par exemple, calculons la caractéristique d'Euler de la sphère et du tore en calculant la caractéristique d'Euler des graphes de découpe représentées dans la Figure 2.5. Sur la sphère, le graphe a deux sommets, une arête et une face, la caractéristique d'Euler est donc égale à 2. En revanche, le graphe plongé sur le tore a un sommet, deux arêtes et une face, ce qui signifie que la caractéristique d'Euler du tore est égale à 0. Cela nous donne un moyen quantitatif de distinguer ces deux surfaces.

Classification des surfaces. Le théorème de classification des surfaces est un résultat fondamental de topologie qui permet de déterminer entièrement les surfaces à partir de deux invariants topologiques : l'orientabilité et la caractéristique d'Euler.

Par corollaire, toute surface peut être obtenue à partir d'une sphère épointée à laquelle ont été collées des surfaces à bords simples le long de leurs bords : des rubans de Möbius, alors appelés *cross-caps*, ou des tores épointés une fois, alors appelés *anses*. La surface est orientable si elle est obtenue par le recollement d'anses uniquement, leur nomnbre est alors appelé *genre* de la surface. Similairement si la surface est obtenue par recollement de ruban de Möbius, elle est non orientable et son genre est le nombre de rubans de Möbius attachés à la sphère. Le nombre de bords, et celui de surfaces attachés, cross-cap et anses, déterminent alors la caractéristique d'Euler de la surface. En effet, pour une surface S de genre g et avec h bords, nous avons $\chi(S) = 2 - g - h$.

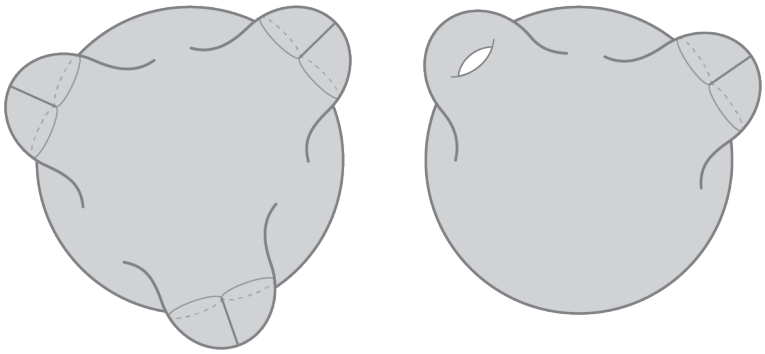

Figure 2.6 : Deux représentations d'une surface non orientable de genre 3.

Comment classer maintenant une surface construite en collant à la fois un ruban de Möbius et une anse à une sphère ? Cette surface est non orientable car elle contient un ruban de Möbius. De plus il peut être montré que sa caractéristique d'Euler est -1, ce qui est aussi le cas d'une sphère à laquelle nous avons ajouté 3 cross-caps ($\chi(S) = 2-3 = -1$), voir Figure 2.6. Sur cette image, nous pouvons voir qu'il est difficile de visualiser un homéomorphisme entre ces deux surfaces, en particulier car les surfaces non orientables et sans bords ne peuvent être plongées dans des espaces de dimension 3 (voir la discussion après le Corollaire 3.46 dans [45]). La topologie algorithmique formalise et analyse ce genre de problèmes et nous permet de calculer de tels homéomorphismes.

Topologie algorithmique des surfaces. La topologie algorithmique dans son ensemble nous donne des outils et algorithmes pour traiter et analyser des problèmes et structures topologiques. Plusieurs problèmes topologiques indécidables dans le cas général admettent des solutions efficaces lorsqu'ils sont restreints aux surfaces. Nombre de problèmes élémentaires de topologie algorithmique sont obtenus en essayant de calculer efficacement des invariants topologiques fondamentaux. Nous présentons dans cette section un échantillons de problèmes de topologie algorithmique. Nous revoyons le lecteur à [20], [19] et [32] pour d'autres problèmes du domaine.

Pour définir proprement nos algorithmes, nous avons d'abord besoin d'un codage discret des surfaces. Ici, nous considérons des surfaces obtenues en collant des disques polygonaux sur leurs bords. De telles surfaces sont appelées *surfaces combinatoires*. De manière équivalente, ces disques polygonaux peuvent être considérés comme les faces d'un graphe cellulairement plongé sur la surface, comme défini plus haut.

1. **Tester l'homéomorphisme.** La question la plus élémentaire et naturelle sur les surfaces est de vérifier si deux surfaces sont les mêmes, si elles sont homéomorphes. Ceci renvoie au théorème de classification des surfaces que nous avons présentés plus haut : deux surfaces sont les mêmes si et seulement si elles ont la même orientabilité et si la même caractéristique d'Euler. Comme expliqué plus haut, il est

possible de calculer, et donc de comparer, les caractéristique d'Euler des surfaces combinatoires. Si les caractéristiques d'Euler sont égales, il suffit ensuite de tester leur orientabilité. Ceci peut être fait en tentant de choisir une orientation sur chaque face, de telle sorte que sur chaque arête, les deux orientations induites par les orientations des faces adjacentes soient opposées. Si une telle orientation existe alors la surface est orientable, elle est non orientable sinon.

2. **Calcul du plus petit graphe de découpe** Comme expliqué précédemment, toute surface peut être découpée le long d'un graphe pour en obtenir un disque. Un tel graphe est appelé *graphe de découpe* (*cut graph*). Le calcul de graphes de découpe est essentiel pour plusieurs problèmes topologiques car ils simplifient la topologie de la surface et la rendent plus simple à manipuler. Il est ensuite naturel d'en faire un problème d'optimisation et de se demander quel est le meilleur moyen de faire cette découpe. Pour qu'un tel problème d'optimisation soit bien défini, la surface doit être munie d'une métrique. Un moyen de procéder est de considérer un graphe cellulairement plongé sur la surface qui induit alors une métrique discrète. Dans ce modèle, les courbes utilisent uniquement les arêtes du graphes et leurs longueur est le nombre d'arêtes utilisées (dans cette thèse, nous utiliserons principalement un modèle équivalent appelé modèle de métrique de croisements (*cross-metric surface* qui sera défini en détail dans les préliminaires).

   Le calcul du plus petit graphe de découpe a été très étudié récemment [18, 17, 33] sous différents aspects algorithmiques (complexité, approximation et complexité paramétré).

Revenons maintenant au problème de la visualisation d'un homéomorphisme entre les deux représentations de la surface non orientable de genre 3, présenté en Figure 2.6. Une approche classique pour obtenir un homéomorphisme entre deux surfaces est de d'abord les découper en disques, puis de faire correspondre les disques par un homéomorphisme, et enfin d'étendre cet homéomorphisme aux surfaces en recollant les disques sur eux-mêmes. Pour que cette approche fonctionne, les deux graphes de découpe doivent avoir la même forme. Il est possible de découper la surface de la Figure 2.6 le long d'un graphe à un sommet et trois arêtes a, b et c pour que leurs extrémités apparaissent dans l'ordre $aabbcc$ autour du sommet. De cette façon, nous obtenons un hexagone pour lequel chaque côté correspond à une copie d'une des arêtes a, b ou c et ceci avec l'ordre $aabbcc$ autour de l'hexagone (voir Figure 2.7). Ce graphe donne une représentation combinatoire d'une surface non orientable de genre 3 dans laquelle chaque arête encode l'un des cross-caps, voir l'image de gauche sur la Figure 2.7. Le produit d'une telle découpe, le long d'un graphe cellulairement plongé à un sommet, est appelé *décomposition canonique* de la surface. Appelons $aabbcc$ l'hexagone obtenu en coupant la première surface, et $a'a'b'b'c'c'$

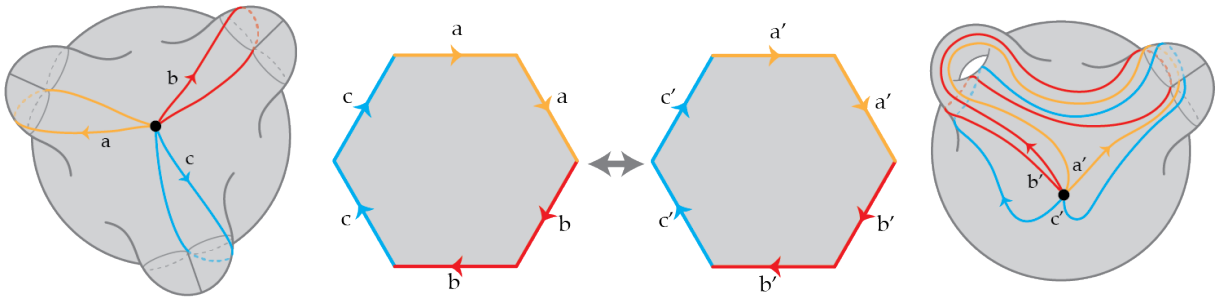

Figure 2.7 : L'homéomorphisme entre deux surfaces non orientables de genre 3 et les décompositions canoniques des surfaces.

celui obtenu en coupant l'autre. Nous pouvons définir un homeomorphisme entre ces hexagones tel que a, b et c soient envoyés sur a', b' et c' respectivement. En recollant les hexagones le long de leurs bords, en accord avec les arêtes du graphe de découpe, nous récupérons les surfaces initiales et un homeorphisme entre elles.

Ici encore, il est naturel de demander des façons efficaces de décomposer canoniquement une surface : étant donnée une surface combinatoire, est-il toujours possible de calculer une décomposition canonique relativement courte vis à vis de la métrique de la surface ? Il existe un algorithme efficace qui calcule une décomposition canonique courte des surfaces orientables [53]. Pour les surfaces non orientables, il existe un algorithme qui calcule une décomposition canonique [52], mais elle n'est pas aussi courte que dans le cas orientable.

Le calcul de décompositions courtes des surfaces non orientable est l'un des résultats principaux de cette thèse. Ce problème illustre à la perfection l'essence de notre travail : l'approche d'un problème topologique sur les surfaces sous l'angle de calculabilité ainsi qu'un dialogue entre topologie et informatique. De plus, ce problème aide à mettre en lumière la topologie sous-jacente au problème du réarrangement de génomes présenté plus haut, qui à son tour établit une connexion féconde avec la bio-informatique.

2.1 Contributions de la thèse

Le thème prédominant de ce travail est l'étude de décompositions de surfaces et plus particulièrement la recherche de moyen efficace de décomposer une surface en des morceaux à topologie plus simple. Nous établissons le premier résultat de décomposition efficace pour des surfaces non orientables ainsi qu'un algorithme pour les calculer. Cette étude est étroitement liée au deuxième thème de cette thèse, qui est centré sur les nombres de croisements dans les graphes. Notre travail apporte des contributions significatives aux trois sujets suivants qui sont résumés en Figure 2.8.

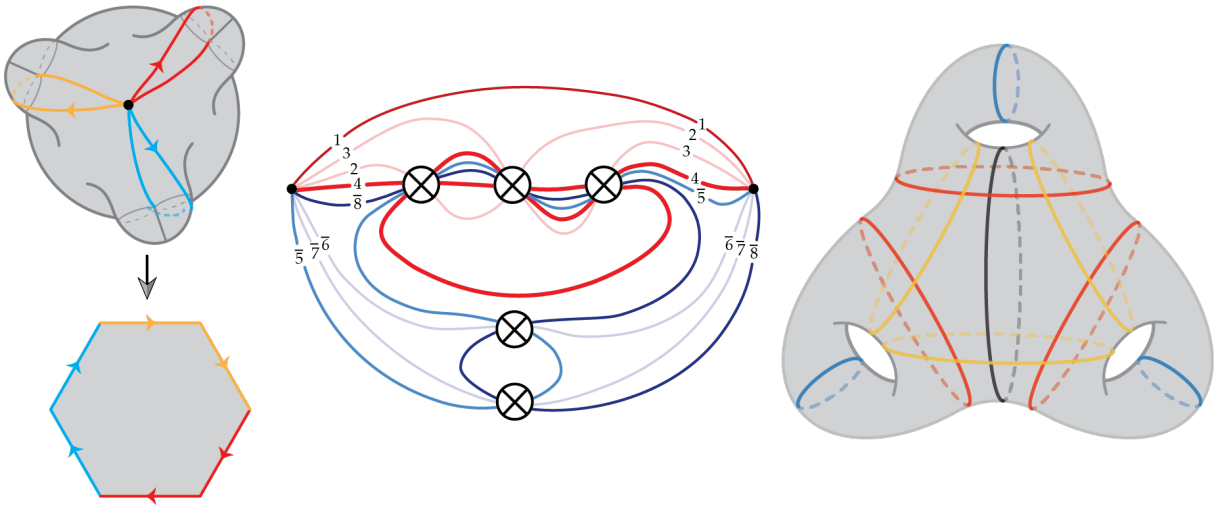

Figure 2.8 : À gauche : Peut-on toujours trouver une décomposition canonique courte pour une surface non orientable ?
Au milieu : Peut-on toujours trouver un dessin planaire pour un graphe plongé tel que $dcr(G) = gcr(G)$? Dans cette image $gcr(G) = 5$ mais l'arête 4 n'est pas simple. Est-il possible de trouver un dessin pour ce graphe avec cinq croisement $\otimes$ tel qu'aucune courbe ne soit compliquée ?
À droite : Combien de courbes sont nécessaires pour contenir toutes les décompositions en pantalon d'une surface ? L'image présente 10 courbes qui réalisent toutes les décompositions en pantalon de la surface orientable de genre 3.

1) Décompositions de surfaces et nombres de croisements joints

La décomposition d'une surface pour simplifier sa topologie est une méthode majeure dans l'étude de sa structure qui a des applications à la fois théoriques et pratiques. Comme mentionné plus haut, dans la plupart de ette thèse, nos surfaces seront associées à une structure géométrique donnée par un graphe plongé sur elles. Ces graphes leurs confèrent ainsi une métrique discrète. Il y a plusieurs façon de modéliser cette métrique : le modèle combinatoire, qui a déjà été abordé, et le modèle de métrique de croisements (nous renvoyons le lecteur à la Section 3.5.1 pour une définition plus détaillée de ces deux modèles). Intuitivement, dans le modèle de métrique de croisements, la longueur d'une décomposition est donnée par le nombre de fois que le graphe dont est issu la métrique est croisé par le graphe de décomposition, sachant que de tels croisements ne peuvent se produire que sur les arêtes. Dans l'autre modèle, le modèle combinatoire, le graphe de décomposition vit sur les arêtes du graphe dont est issue la métrique, et la longueur de la décomposition est le nombre d'arêtes utilisées par le graphe, compté avec multiplicité.

Pour des applications pratiques, il est naturel d'essayer de trouver une décomposition d'une surface qui soit courte. Par exemple, un problème classique consiste en la recherche d'une décomposition canonique courte, qui découpe la surface en un disque. La meilleure borne connue, pour des cas particuliers, sur la longueur de telles décompositions d'une

surface de genre g est $O(g|E(G)|)$ où G est le graphe de la métrique et $|E(G)|$ son nombre d'arêtes. Cependant, avant notre notre travail, aucune borne de ce type n'était connue pour les décompositions de surfaces non orientables, voir l'image de gauche dans la Figure 2.8. Un des objectifs de cette thèse est l'analyse et l'étude systématiques des décompositions efficaces de surfaces, ainsi que la conception d'algorithmes pour les calculer.

Des décompositions aux nombres de croisements joints de graphes. Le problème de la recherche de décompositions efficaces de la surface est dualement équivalent à l'étude de la meilleure façon de plonger simultanément deux graphes sur la même surface. Plus précisément, nous considérons le *nombre de croisements joints* d'une paire de graphes plongés : pour deux graphes séparément plongés G_1, G_2 sur une surface fermée S, nous cherchons à minimiser le nombre d'intersections entre $h(G_1)$ et G_2 sur l'ensemble des homéomorphismes $h : S \to S$ avec la contrainte additionnelle que toutes les intersetions doivent être transverses et sur les arêtes. La conjecture qui suit est une conjecture de Negami, ouverte depuis plus de 20 ans :

Conjecture 4. (Conjecture de Negami) *Il existe une constante universelle universelle C telle que pour toue paire de graphes G_1 et G_2 plongés sur une surface S, leurs nombre de croisements joints est au plus $C|E(G_1)||E(G_2)|$.*

La conjecture de Negami implique qu'il existe des décompositions efficaces de n'importe quelle forme : chaque arête du graphe de décomposition a une longueur maximale de $O(|E(G)|)$ où G est le graphe de la métrique de la surface. Malgré des découvertes ultérieures [3, 47, 66], cette conjecture reste ouverte. La meilleure borne connue pour le nombre de croisements joints de graphes est $O(g|E(G_1)||E(G_2)|)$, où g est le genre de la surface S (voir [59]). Au niveau des décompositions, cela implique directement que pour toute surface discrète et tout graphe qui découpe la surface en un disque, il existe une décomposition de longueur $O(g^2|E(G)|)$ où G est le graphe de la métrique. Cette limite est assez élevée pour des applications pratiques.

Métriques universelles de plus courts chemins. Pour tenter de faire des progrès sur la conjecture de Negami, Hubard, Kaluža, de Mesmay et Tancer [50] ont posé la question, plus géométrique, suivante :

Question 2. *Étant donné une surface S de genre g, existe-t-il une métrique riemannienne sur S telle que tout graphe simple plongeable sur S puisse être plongé tel que les arêtes soient des plus courts chemins sur S ?*

L'existence d'une telle métrique impliquerait la conjecture de Negami, en effet, deux graphes quelconques pourraient être plongés de telle sorte que leurs arêtes soient des plus

courts chemins. Dans une métrique riemannienne, deux plus courts chemins se croisent au plus une fois, ce qui implique que toute paire d'arêtes des graphes se croiseraient au plus une fois (Voir Section 4.1 pour une explication plus détaillée).

Bien que la question 2 reste ouverte, une relaxation de celle-ci est prouvée dans [50]. Dans cet article, il est exhibé une métrique universelle Rimanienne sur une surface orientable de genre g telle que chaque arête est une concaténation de $O(g)$ plus courts chemins. Il ets également connu que cette métrique implique la borne $O(g|E(G_1)||E(G_2)|)$ pour le nombre de croisements joints de graphes prouvée dans [59]. Néanmoins, le cas des surfaces non orientables reste un problème ouvert dans cet article.

Contributions
Dans cette thèse, nous nous concentrons principalement sur les décompositions de surfaces non orientables, étant relativement peu explorées par rapport aux surfaces orientables.

- Nous fournissons les premières décompositions courtes pour les surfaces non orientables et des algorithmes polynomiaux pour les calculer en nous appuyant sur une nouvelle technique. Voir les Théorèmes C et D et la Section 6.3. Dans la preuve du théorème D, nous exploitons des résultats de bio-informatique.
- Nous utilisons cette nouvelle technique pour fournir une nouvelle méthode de calcul de décompositions canoniques des surfaces orientables. Voir le théorème F.
- Nous fournissons une borne inférieure pour les décompositions canoniques en utilisant un argument de comptage. Voir le théorème H.
- Nous corrigeons la preuve de Negami de la borne $O(g|E(G_1)||E(G_2)|)$ pour les nombres de croisements joints de deux graphes quelconques G_1 et G_2 dans le cas non orientable. Voir le théorème A.
- Nous généralisons la construction de la métrique universelle introduite dans [50] au cas non orientable. Voir le théorème B.

2) Nombre de croisements dégénérés vs. nombre de croisements de genre

Nous passons ensuite des décompositions aux nombres de croisements pour les graphes. Les nombres de croisements sont un outil important et populaire dans le dessin et la visualisation de graphes (voir [68] pour un panorama) où la minimisation du nombre de croisement est la principale manière de réduire la complexité des dessins. Une question

classique dans l'étude des nombres de croisements est de savoir si deux nombres de croisements différents peuvent coïncider sur de larges classes de graphes. Dans cette thèse, nous nous intéressons aux deux nombres de croisements suivants. Le *nombre de croisements dégénérés* de G, noté $dcr(G)$ (*degenerate crossing number*), défini pour la première fois par Pach et Tóth [61], est le plus petit nombre de croisements parmi les dessins simples de G dans le plan, dans lesquels les croisements sont transversaux et le croisement de plusieurs arêtes en un point est compté comme un seul. Si nous autorisons les auto-intersections d'arêtes de G, cela définit le nombre de croisements de genre de G, $gcr(G)$ qui a été défini pour la première fois par Mohar [57]. Il est intéressant de noter que ces deux nombres de croisements se traduisent dans notre contexte de graphes plongés sur des surfaces non orientables : le nombre de croisements dégénérés de G est égal au plus petit nombre de cross-caps nécessaires sur une surface pour y plonger G de telle sorte que chaque arête de G passe par chaque cross-cap au plus une fois. De plus, le nombre de croisements de genre de G est égal au plus petit nombre de cross-caps nécessaires sur une surface pour y plonger G, mais ici une arête est autorisée à entrer dans chaque cross-cap plus d'une fois. Il est évident, de par leurs définitions que $gcr(G) \leq dcr(G)$. La conjecture suivante, de Mohar, concerne ces deux nombres de croisements :

Conjecture 5. ([57, Conjecture 3.1]) *Pour tout graphe simple G, $dcr(G) = gcr(G)$.*

Mohar a également formulé une conjecture plus forte selon laquelle cette affirmation tient aussi pour tout plongement de graphe sans boucle. Notons $S^2 \setminus g\bigotimes$, la sphère privée de g petits disques, et par $(S^2 \setminus g\bigotimes)/\sim$ l'espace obtenu en quotientant la frontière de chaque disque avec une carte antipodale. Nous appelons chaque $\bigotimes$ une cross-cap. Topologiquement, ceci revient à coller un ruban de Möbius, le long de son bord, sur chaque disque manquant ; ce qui donne la surface non orientable de genre g, que nous notons N_g. Nous disons qu'un multigraphe plongé $\phi\colon G \to N_g$ *admet un dessin en cross-caps* $\phi'\colon G \to S^2$, s'il existe un homéomorphisme $f\colon N_g \to (S^2 \setminus g\bigotimes)/\sim$ tel que $f(\phi(G)) = \phi'(G)$. Lorsqu'une arête du graphe intersecte une cross-cap, on dit que l'arête *entre* dans la cross-cap. Si un graphe G a un nombre de croisements dégénérés égal au nombre de croisements de genre alors il existe un dessin en cross-caps de G avec $dcr(G) = gcr(G)$ cross-caps dans lesquelles chaque arête entre au plus une fois. Une *pseudo-triangulation* est un multi-graphe cellulairement plongé $\phi\colon G \to S$ dans lequel chaque face a degré trois. La conjecture suivante est la plus forte conjecture de Mohar concernant les pseudo-triangulations :

Conjecture 6. ([57, Conjecture 3.4]) *Pour tout entier strictement positif g, il existe une pseudo-triangulation sans boucle de N_g qui admet un dessin en cross-caps avec g cross-caps dans lesquelles chaque arête entre au plus une fois dans chacune d'entre elles.*

L'image centrale de la Figure 2.8 illustre le dessin d'une cross-cap et la question à laquelle cette conjecture se résume.

Contributions

- Nous fournissons un plongement d'un graphe à 2 sommets sans boucle sur une surface de genre 5 qui est un contre-exemple à la Conjecture 6. Voir le Théorème I.
- Nous prouvons un théorème de structure qui classifie presque complètement les plongements de graphes à 2 sommets sans boucle pour lesquels la conjecture 6 se vérifie. Cela montre que la plupart des plongements de graphes à 2 sommets sans boucle vérifient cette conjecture. La preuve utilise directement un algorithme issu de la bio-informatique. Voir Théorème J.

3) Dénombrer les courbes et les arcs sur les surfaces

Les résultats de la dernière partie de cette thèse ont une saveur assez différente par rapport au reste car nous oublions les longueurs des courbes et des nombres de croisements pour nous concentrer, contrairement aux résultats précédents, uniquement sur les surfaces orientables. Notre motivation pour étudier ce problème est de progresser vers une réponse à la Question 2 sur l'existence d'une métrique universelle de plus courts chemins.

Nous nous intéressons maintenant aux décompositions en pantalons de surfaces orientables. Une *décomposition en pantalons* est un ensemble de courbes fermées sur une surface qui la découpe en pantalons (sphères à trois trous). Les pantalons sont des blocs fondamentaux dans l'étude des surfaces. Nous étudions la taille minimale d'une famille universelle de courbes qui réalise toutes les décompositions en pantalons d'une surface : une famille de courbes fermées simples Γ sur une surface *réalise tous les types de décompositions en pantalons* si pour toute décomposition en pantalons de la surface, il existe un homéomorphisme l'envoyant sur un sous-ensemble des courbes de Γ. Nous nous intéressons à l'estimation de la taille d'une telle famille et nous fournissons un encadrement de sa taille pour différentes instances du problème, voir l'image de droite dans la Figure 2.8. En particulier, dans le cas des surfaces fermées, des progrès pourraient répondre négativement à la question 2 sur l'existence d'une métrique de plus courts chemins sur ces surfaces (voir la section 8.1 pour plus de détails sur cette implication).

Contributions

- Nous fournissons une borne supérieure exponentielle et une borne inférieure superlinéaire sur la taille minimale d'une famille de courbes qui réalise tous les types de décompositions en pantalons dans le cas de surfaces sans bords. Voir le théorème K.
- Nous fournissons des bornes supérieures et inférieures dans le cas de surfaces épointées avec des perforations que nous pouvons considérer comme étiquetées ou non. Voir le Théorème L.
- Nous étudions également un concept similaire d'universalité pour les triangulations de polygones, où nous fournissons des bornes optimales à un facteur logarithmique près. Voir les théorèmes M et N.

2.2 Organisation

Dans le chapitre 3, nous commençons par définir les différentes notions que nous utiliserons tout au long de cette thèse. Notre travail utilise des outils provenant de différents domaines des mathématiques tels que la géométrie, la topologie et la théorie topologique des graphes. Nous expliquons également le sujet connexe des réarrangements du génome en bio-informatique. Bien que nous nous efforcions de fournir des références pour la plupart des notions abordées dans ce chapitre, il convient de noter que certains concepts (tels que les dessins à boîte et les relations avec les réarrangements du génome) sont présentés et formalisés ici pour la première fois. Néanmoins, nous avons diligemment cité les sources à partir desquelles les idées ont été dérivées.

Nos résultats apparaissent dans les sections 4, 5, 6, 7 et 8 et, à l'exception des sections 5 et 6 qui sont étroitement liées, le reste des chapitres peut être lu indépendamment. Notez que nos principaux théorèmes sont désignés par des lettres alphabétiques tout au long de la thèse.

Le chapitre 4 se concentre sur deux résultats en rapport aux nombres de croisements joints. Ces deux résultats généralisent des résultats sur les surfaces orientables aux contexte non orientable. Tout d'abord, nous corrigeons un argument de Negami sur le nombre de croisements joints qui a un problème mineur dans le cas des surfaces non orientables. Ensuite, nous fournissons une métrique universelle pour les surfaces non orientables qui permet à chaque graphe plongeable sur la surface d'être plongé tel que chaque arête est une concaténation de $O(g)$ plus courts chemins. Cette preuve repose sur la généralisation d'une décomposition octogonale connue pour les surfaces orientables au cadre

non orientable. Les théorèmes A, B et C se trouvent dans ce chapitre. Le premier résultat est contenu dans la version préliminaire de [A] qui est publiée dans les Proceedings of the 38th Symposium on Computational Geometry. Les deux résultats apparaissent dans une version journal [A] qui a été publiée dans le journal of Discrete & Computational Geometry.

Dans les chapitres 5 et 6, nous entamons une étude approfondie des décompositions topologiques courtes sur les surfaces non orientables. Le chapitre 5 expose notre résultat sur le calcul de décompositions canoniques pour les surfaces non orientables, dans lequel nous introduisons une nouvelle approche. Le théorème D se trouve dans ce chapitre. C'est le résultat principal de [A] qui apparaît à la fois dans la version conférence et la version journal de l'article. Le chapitre 6 contient des résultats non publiés sur d'autres formes de décompositions pour des surfaces orientables et non orientables. Les résultats de ce chapitre proviennent principalement de l'utilisation de notre nouvelle approche pour trouver d'autres décompositions courtes. Nous y fournissons également une borne inférieure pour les décompositions canoniques et on y trouve les théorèmes F et H.

Le chapitre 7 expose notre résultat sur la conjecture de Mohar concernant les nombres de croisements de genre et les nombres de croisements dégénérés. Les théorèmes J et I s'y trouvent. Les résultats de ce chapitre apparaissent dans [C] qui est paru dans les Proceedings of the 31st International Symposium on Graph Drawing and Network Visualization.

Le chapitre 8 contient notre résultat sur l'estimation de la taille de la famille de courbes qui réalise tous les types de décompositions en pantalons. Les théorèmes K, L, M et N s'y trouvent. Les résultats de ce chapitre apparaissent dans [B] qui a été accepté dans Israel Journal of Mathematics.

Enfin, dans le chapitre 9, nous résumons nos principaux résultats et problèmes ouverts, et nous exposons quelques pistes de recherche découlant de notre travail.

Chapter 3

Preliminaries

This chapter serves as an introduction to the key concepts that will be utilized throughout the thesis. The presentation will primarily be formal and precise, so the reader may choose to skip this section initially and return to it later as needed to fully comprehend the necessary concepts. We have included references in each section to enable the reader to access further information.

3.1 Topological surfaces

The primary objects and mathematical tools we employ are drawn from the field of topology. This chapter offers an introduction to the essential concepts in this field that are required for this work. However, a more detailed comprehension might be necessary to fully grasp these concepts. For an in-depth explanation, we recommend consulting standard textbooks such as Hatcher's [45] or Stillwell's [72].

A *surface* S is a topological Hausdorff space that locally looks like the plane, i.e., each point of the surface has a neighborhood homeomorphic to either the plane or the closed half-plane. The points without a neighborhood homeomorphic to the plane comprise the *boundary* of S. See Figure 3.1 for examples of surfaces. A surface is called *orientable* if it does not contain a subspace homeomorphic to a Möbius band; otherwise, it is called *non-orientable* (the two rightmost pictures in Figure 3.1 are non-orientable). A compact surface without boundary is called a *closed surface*.

Throughout this thesis, we denote orientable surfaces and non-orientable surfaces by M and N respectively, and by S whenever orientability is not of importance.

As briefly mentioned in the introduction, surfaces can be obtained by removing disks from the sphere and gluing *handles* and *cross-caps* to it. A handle is obtained by removing a small disk and gluing a punctured torus along its boundary to the boundary circle of the resulting hole (see the left picture in Figure 3.2) and a cross-cap is obtained by removing a

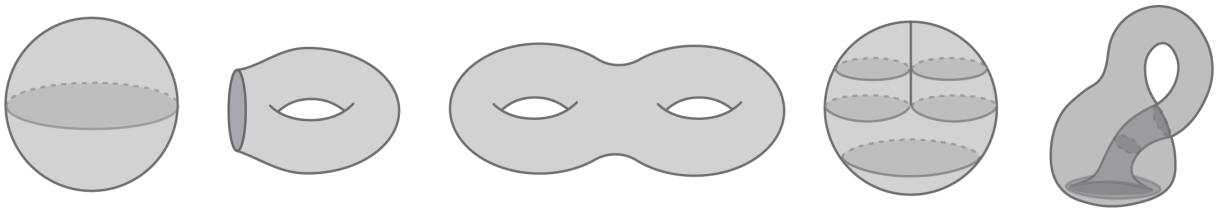

Figure 3.1: Examples of surfaces: from left to right: sphere, punctured torus (the torus with one boundary component), double torus, Projective plane and Klein bottle.

small disk from the sphere and gluing a Möbius band along its boundary to the boundary circle of the resulting hole (see the right picture in Figure 3.2). If we only attach handles to a sphere, the surface that we obtain is orientable but as soon as we attach a cross-cap, we obtain a non-orientable surface.

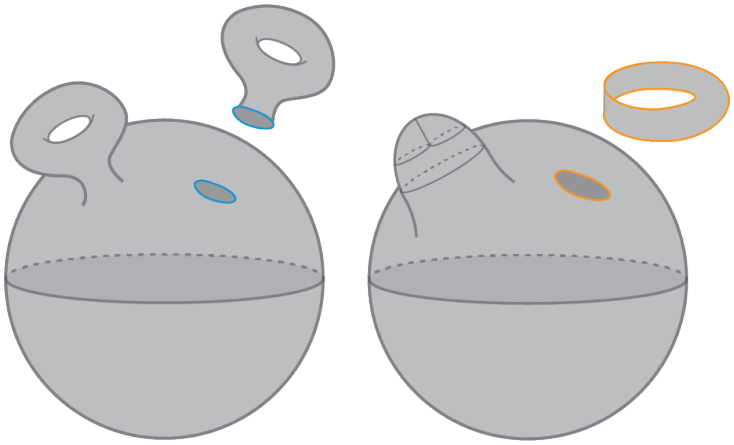

Figure 3.2: Gluing Möbius band and handles to punctured spheres to obtain higher genus surfaces. At left, the surface that we obtain is non-orientable and at right, it is orientable.

The following theorem states a fundamental topological classification result about surfaces. We refer the reader to [72, Section 1.3] for a proof of this theorem.

Theorem 3.1.1 (Classification of surfaces)**.** *Every connected surface is homeomorphic to either*

- *the orientable surface of genus g and b boundaries, which is obtained by attaching g handles to a sphere and removing b open disks from it.*
- *the non-orientable surface of genus g with b boundaries, which is obtained by attaching g cross-caps to a sphere, and removing b open disks from it.*

Therefore, a topological surface is completely recognized by its genus g, number of boundaries and its orientability. If the surface is orientable, we call g its *orientable genus* and if it is non-orientable, we call g its *non-orientable genus*. We sometimes simply use

genus to refer to the orientable genus (resp. non-orientable genus) of an orientable surface (resp. non-orientable surface).

The following is directly implied by the classification of surfaces.

Corollary 3.1.2. *A handle counts as two cross-caps for non-orientable surfaces: a surface that is obtained by attaching k handles and $l \geq 1$ cross-caps is homeomorphic to a non-orientable surface of non-orientable genus $2k + l$.*

We denote by $N_{g,b}$ (resp. $M_{g,b}$) the non-orientable (resp. orientable) surface of genus g with b boundaries. As an example, the surfaces in Figure 3.1 from left to right, are $M_{0,0}, M_{1,1}, M_{2,0}$, $N_{1,0}$ and $N_{2,0}$, respectively. The *Euler genus* of a surface S, denoted by $eg(S)$, is twice the orientable genus for orientable surfaces and equal to the non-orientable genus for non-orientable ones. Another way to define the *Euler characteristic* of a surface S with $h(S)$ boundary components is $\chi(S) = 2 - eg(S) - h(S)$.

3.2 Structures on surfaces

In this thesis, our main objects of study are graphs and curves that live on surfaces.

3.2.1 Graphs embedded on surfaces

For a thorough study of graphs embedded on surfaces and their properties, we refer the reader to the book of Mohar and Thomassen [58]. Here, we only recall the basic notions that we use in this thesis.

An embedding of a graph G on a surface S is a continuous injective map $\phi : G \to S$. Therefore ϕ maps the vertices of G to distinct points on S and an edge $e = vw$ of G, that connects the vertex v to w, to a simple curve connecting $\phi(v)$ to $\phi(w)$. Throughout this thesis, we generally identify a graph with its embedding. For an embedding of G on a surface S, each connected component of the complement of the image of G is called a *face* of the embedding. A face can be recognized by a cyclic sequence of its incident edges and vertices in which an edge might appear twice and a vertex might appear multiple times. We denote by $V(G), E(G)$ and $F(G)$ the vertices, edges and faces of G respectively and we use $|.|$ to refer to the cardinality of these sets.

An embedding is *cellular* if every face of the embedding is homeomorphic to an open disk. The *degree* of a face in a cellular embedding is the number of sides in the polygonal disk corresponding to the face that we obtain by cutting the surface along the embedding. A cellular embedding of a graph where every face has degree exactly three is called a *triangulation*. In case that the graph of the triangulation is a multigraph (loops and multiple edges between the same pair of vertices are allowed), it is called a *pseudo-triangulation*.

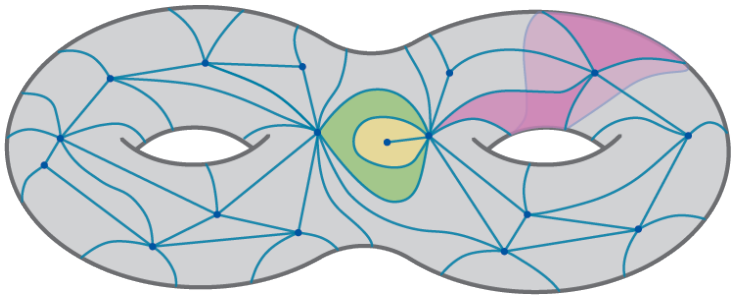

Figure 3.3: An example of a pseudo-triangulation of the double torus. The three colored triangles provide examples of triangles that are not incident to exactly three distinct vertices or three distinct edges.

Note that a triangle in a pseudo-triangulation is not necessarily incident to three distinct vertices and edges, see Figure 3.3 for examples of such triangles in a pseudo triangulation of the double torus.

An embedding of a graph G on a surface S is called an *orientable embedding* if each cycle in G is mapped to a two-sided closed curve on S; otherwise it is called a *non-orientable* embedding. All graph embeddings on an orientable surface are orientable.

Two embedded graphs are *equivalent*, if there is a homeomorphism of the surface that maps one to the other one.

3.2.1.1 Genus of a graph

A graph is called *planar* if it can be embedded on $\mathbb{R}^2$; otherwise it is *non-planar*. The *non-orientable genus of a graph* G, $\tilde{g}(G)$, is the smallest possible genus of a non-orientable surface on which it can be embedded. Similarly the *orientable genus of a graph* G, $g(G)$, is the smallest possible genus of an orientable surface on which it can be embedded.

Lemma 3.2.1. *Let G be a non-planar graph. Then $\tilde{g}(G) \leq 2g(G) + 1$.*

Proof. Consider an embedding of the graph G on an orientable surface M of genus $g(G)$. We know that $eg(M) = 2g(G)$. Choose a face of this embedding. Cut a disk inside this disk and attach a Möbius band. By doing so we obtain a non-orientable surface N of non-orientable genus $g(N) = 2g(G) + 1$ and an embedding of G on N. This implies that $\tilde{g}(G) \leq g(N) = 2g(G) + 1$. This finishes the proof. □

This lemma shows that the non-orientable genus of a surface is bounded by its orientable genus but there is no inequality in the other direction to bound the orientable genus of a graph by its non-orientable one. Indeed, there are examples of graphs for which the non-orientable genus is considerably less than their orientable genus (see [4] or [58, Theorem 5.8.1]).

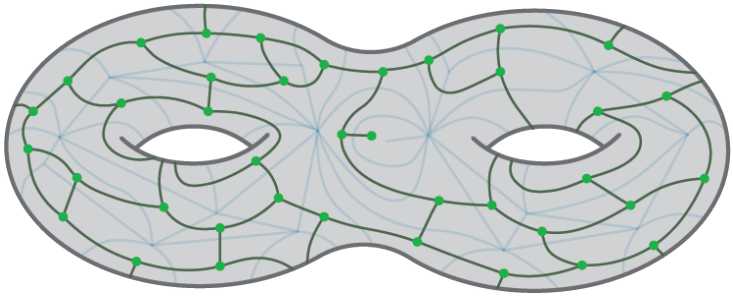

Figure 3.4: The dual graph to the pseudo-triangulation of the double torus depicted in Figure 3.3.

Recall from the introduction that for a graph G cellularly embedded on a surface S, *Euler's formula* states that $|V(G)| - |E(G)| + |F(G)| = \chi(S)$. When the graph is not cellularly embedded, we have the inequality: $|V(G)| - |E(G)| + |F(G)| \geq \chi(S)$.

3.2.1.2 Duality

For a graph G cellularly embedded on a surface without boundary, the *dual graph* G^* is defined as follows: to each face f of G we associate a vertex f^*. To each edge e in G that is incident to faces f_1 and f_2 (possibly $f_1 = f_2$), we associate an edge e^* that connects the vertices f_1^* and f_2^* in G^*, see Figure 3.4. Note that in this construction, two vertices can be connected to each other with multiple edges if their corresponding faces have more than one common edge. It is easy to see that $G^{**} = G$.

In Section 3.5.1.3, we introduce a generalization of this duality to the case of surfaces with boundaries.

3.2.2 Curves on surfaces

A *path* from x to y on a surface, is a continuous map $\theta : [0,1] \to S$ where $\theta(0) = x$ and $\theta(1) = y$. Let $\theta : [0,1] \to S$ and $\theta' : [0,1] \to S$ be two paths on the surface, such that $\theta(1) = \theta'(0)$. The *concatenation* of θ and θ', denoted by $\theta.\theta'$, is a path $\theta'' : [0,1] \to S$ such that $\theta''(t) = \theta(2t)$ for $0 \leq t \leq \frac{1}{2}$ and $\theta''(t) = \theta'(2t-1)$ for $\frac{1}{2} \leq t \leq 1$. If a path has the same endpoints $x = \theta(0) = \theta(1)$, it is called a *loop* and x is called the *basepoint* of the loop. A path with two ends on the boundary components of the surface is called an *arc*. A *closed curve* or a *cycle* on a surface S is a continuous map $\theta : S^1 \to S$ where S^1 is the unit circle. A *curve* is a path or a closed curve. Curves are called *simple* if the maps are injective, i.e., the curves do not self-intersect. A *constant cycle* is a constant map $\theta : S^1 \to S$.

A curve on a surface is called *two-sided* if a small closed neighborhood of it is homeomorphic to the annulus. On the other hand, it is called *one-sided* if it has a closed neighborhood homeomorphic to the Möbius band, see Figure 3.5. Consequently, every

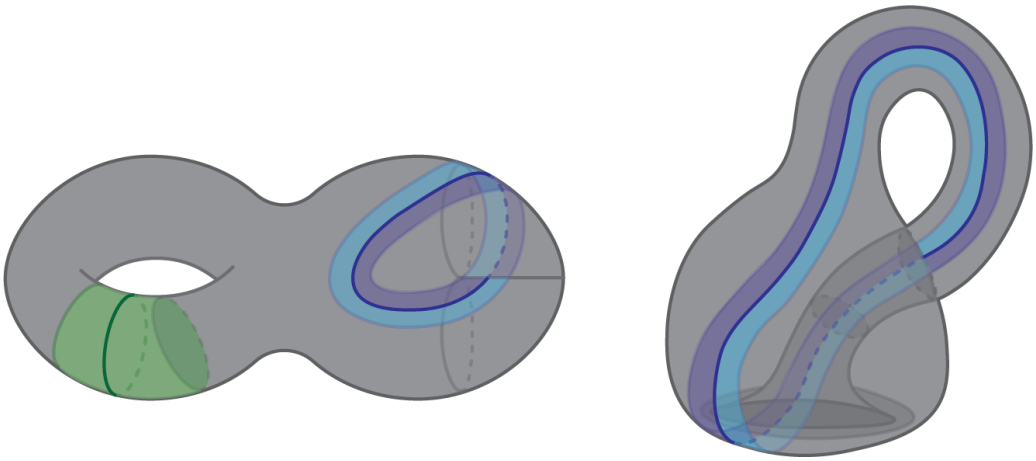

Figure 3.5: Left: the surface $N_{3,0}$ with a two-sided curve (in green) and a one-sided one (in blue), right: Klein bottle, $N_{2,0}$, and a one-sided curve. The green (resp. blue) area shows the neighborhood of the two-sided green curve (resp. one-sided blue curve/s) which is homeomorphic to the annulus (resp. Möbius band).

closed curve on an orientable surface is two-sided.

Note that notions of sidedness are only defined for closed curves. We extend it for arcs that have both ends on the same boundary component. Such an arc is two-sided (resp. one-sided), if the closed curve obtained by connecting the two ends of the arc along one of the boundary segments between its endpoints is two-sided (resp. one-sided).

3.2.2.1 Cutting a surface along a curve

Given a simple curve ν on a surface S, we say that S_ν is a surface with boundary (or boundaries) that has been obtained by *cutting* S along ν if S_ν is equipped with a homeomorphism h on the boundary (or the boundaries) such that the quotient space $S_\nu/(h(x) \sim x)$ is a surface homeomorphic to S and the image of the boundary (boundaries) of S_ν under this quotient map is ν.

Let ν be a closed simple curve on S. If ν is one-sided, S_ν has one boundary component and the corresponding homeomorphism h on the boundary is the one that maps antipodal points to each other, as depicted in the top right picture in Figure 3.6. In the case where ν is two-sided, S_ν has two boundary components and the corresponding homeomorphism h on the boundaries is the one that maps one boundary to the other. Figure 3.6 provides three different examples for this case (see the top left, bottom left and bottom right pictures).

A curve ν on a surface S is called *non-separating* if S_ν is connected; otherwise ν is separating and by cutting along it we obtain two connected components. In this case, we denote the two components by S^1_ν and S^2_ν. Note that a separating curve is always two-sided.

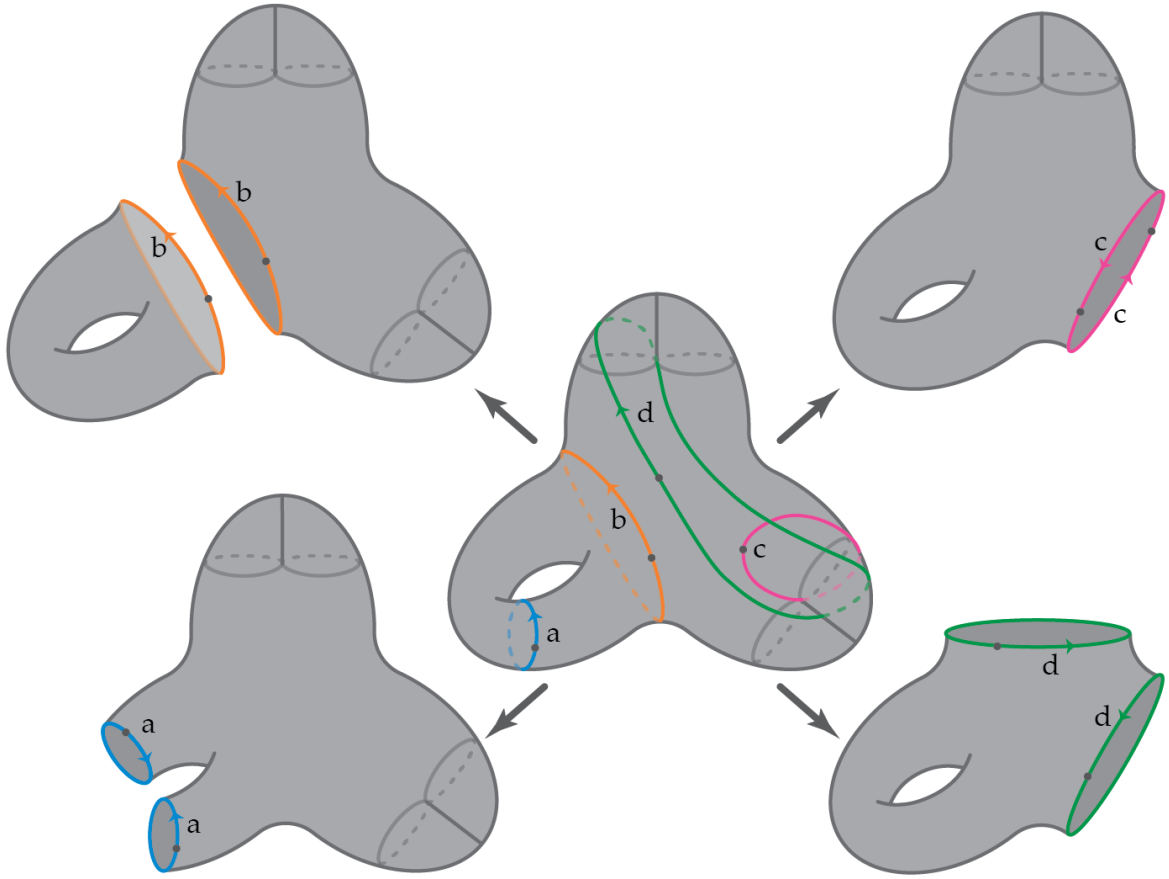

Figure 3.6: Middle: the non-orientable surface $N_{4,0}$; the surfaces on the corners are the ones that we obtain after cutting the middle surface along each of the curves a,b,c and d.

An arc is separating if it has both ends on the same boundary component and the closed curve obtained by connecting the two ends of the arc along one of the boundary segments is separating. Cutting along an arc with endpoints on different boundary components reduces the number of boundary components but it does not reduce the genus (we will show this in Lemma 3.2.3). Furthermore such a curve cannot be separating.

Lemma 3.2.2. *Let ν be a simple curve on a surface S (possibly with boundary components) and let S_ν be the surface that we obtain by cutting S along ν (in case ν is separating, we denote the components by S^1_ν and S^2_ν). Depending on the type of ν, the Euler characteristic of the resulting surface is described by the following table.*

	ν is a cycle	*ν is an arc*
ν is separating	$\chi(S) = \chi(S^1_\nu) + \chi(S^2_\nu)$	$\chi(S) = \chi(S^1_\nu) + \chi(S^2_\nu) - 1$
ν is non-separating	$\chi(S) = \chi(S_\nu)$	$\chi(S) = \chi(S_\nu) - 1$

Proof. Let us assume that the surface is triangulated such that ν is divided into k sub-edges and these sub-edges belong to the triangulation; denote this triangulation by T.

We prove the claim by computing the Euler characteristic of this triangulation before and after cutting the surface along ν. We know that after cutting ν, we get two copies of ν so the edges and vertices of ν are doubled.

- First let us assume that ν is a separating cycle. The triangulation T induces a triangulation T^1 and T^2 for the two components S^1_ν and S^2_ν after the cut. The total number of faces before and after cutting remains the same, $F(T^1)+F(T^2)=F(T)$. Also note that since ν is a closed curve, the number of vertices and edges of ν are equal. Therefore we have: $\chi(S^1_\nu)+\chi(S^2_\nu) = (|V(T^1)|-|E(T^1)|+|F(T^1)|)+(|V(T^2)|-|E(T^2)|+|F(T^2)|) = (|V(T)|+k)-(|E(T)|+k)+|F(T)| = |V(T)|-|E(T)|+|F(T)| = \chi(S)$.

- Let us assume that ν is a non-separating cycle. This case is quite similar to the previous one with the difference that here, after cutting along ν, we obtain a surface with only one component. Let us denote the triangulation induced by T on S_ν after cutting along ν by T'. We have $\chi(S_\nu) = |V(T')| - |E(T')| + |F(T')| = (|V(T)| + k) - (|E(T)| + k) + |F(T)| = |V(T)| - |E(T)| + |F(T)| = \chi(S)$.

Note that in the case that ν is an arc, the number of vertices on ν is one more than the sub-edges of ν.

- In the case where ν is separating, we denote by T^1 and T^2 the induced triangulations on ν^1 and ν^2. We have $\chi(S^1_\nu)+\chi(S^2_\nu) = (|V(T^1)|-|E(T^1)|+|F(T^1)|)+(|V(T^2)|-|E(T^2)|+|F(T^2)|) = (|V(T)|+(k+1))-(|E(T)|+k)+|F(T)| = |V(T)|-|E(T)|+|F(T)|+1 = \chi(S)+1$.

- In the case where ν is non-separating, denote by T' the triangulation induced on S_ν by T after cutting S along ν. We have $\chi(S_\nu) = |V(T')| - |E(T')| + |F(T')| = (|V(T)|+k+1)-(|E(T)|+k)+|F(T)| = |V(T)|-|E(T)|+|F(T)|+1 = \chi(S)+1$.

This finishes the proof. □

Lemma 3.2.2 and the Euler formula together, prove the following lemma that relates the Euler genus of surfaces before and after cutting along a curve.

Lemma 3.2.3. *Let ν be a simple curve on a surface S and S_ν be the surface that we obtain by cutting S along ν (in case ν is separating, we denote the components by S^1_ν and S^2_ν). Depending on the type of ν, the Euler genus of the resulting surface is described by the following table.*

	ν is a cycle or an arc with endpoints on the same boundary component
ν is separating	$eg(S) = eg(S_\nu^1) + eg(S_\nu^2)$
ν is two-sided non-separating	$eg(S) = eg(S_\nu) + 2$
ν is one-sided	$eg(S) = eg(S_\nu) + 1$

In the case where ν has endpoints on different boundary components, $eg(S) = eg(S_\nu)$.

Proof. An arc with endpoints on the same boundary can be seen as a closed curve if it is concatenated with one of the segments of the boundary. Therefore we only prove the theorem for closed curves as the proofs for these arcs are similar. Let us denote the number of boundary components in S by $h(S)$.

- If ν is a separating curve, by cutting along ν, we get two surfaces S_ν^1 and S_ν^2 such that each of them has one boundary component that is obtained by the cut and corresponds to a copy of ν, see the top left picture in Figure 3.6. Therefore we have $h(S) = h(S_\nu^1) + h(S_\nu^2) - 2$. Since ν is a separating closed curve, by Lemma 3.2.2 we have $2-eg(S)-h(S) = \chi(S) = \chi(S_\nu^1)+\chi(S_\nu^2) = (2-eg(S_\nu^1)-h(S_\nu^1))+(2-eg(S_\nu^2)-h(S_\nu^2)) = 2-(eg(S_\nu^1)+eg(S_\nu^2))-h(S)$ which implies that $eg(S) = eg(S_\nu^1) + eg(S_\nu^2)$.

- If ν is two-sided and non-separating, then S_ν has one connected component and two additional boundary components compared to S such that each of them corresponds to a copy of the curve ν after cutting, see bottom left and bottom right pictures in Figure 3.6. Since ν is non-separating, by Lemma 3.2.2, we know that $2 - eg(S) - h(S) = \chi(S) = \chi(S_\nu) = 2-eg(S_\nu)-(h(S)+2)$ which implies that $eg(S) = eg(S_\nu)+2$.

- If ν is a one-sided curve, then S_ν has one additional boundary component compared to S that corresponds to both copies of the curve ν after cutting, see the top right picture in Figure 3.6. Since ν is one-sided, it is non-separating and therefore by Lemma 3.2.2, we have $2 - eg(S) - h(S) = \chi(S) = \chi(S_\nu) = 2 - eg(S_\nu) - (h(S) + 1)$ which implies that $eg(S) = eg(S_\nu) + 1$.

- Let ν be an arc with endpoints on different boundary components of S denoted by h_1 and h_2 . In this case, ν is non-separating and by cutting along it, we reduce the number of boundary components in S by 1: cutting along ν merges h_1 and h_2 to a single boundary component. By Lemma 3.2.2, we have $2 - eg(S) - h(S) = \chi(S) = \chi(S_\nu) - 1 = 2 - eg(S_\nu) - h(S_\nu) - 1 = 2 - eg(S_\nu) - h(S)$ which implies that $eg(S) = eg(S_\nu)$ and concludes the proof.

This finishes the proof of the lemma. □

3.2.2.2 Types of curves on surfaces

We can classify simple curves up to homeomorphisms of the surface. We say that two curves ν and ν' on a surface S have the same *type* or belong to the same *homeomorphism class* of curves if there exists a homeomorphism of the surface $h : S \to S$ for which $h(\nu) = \nu'$. Two arcs with endpoints on different boundary components are always of the same type.

Types of curves on orientable surfaces

The following lemmas can be used to classify the different types of curves on an orientable surface.

Lemma 3.2.4. *Two simple non-separating closed curves ν and ν' on an orientable surface M have the same type.*

Proof. Let M_ν and $M_{\nu'}$ be the orientable surfaces that we obtain by cutting M along the curves ν and ν', respectively. We know that M_ν (resp. $M_{\nu'}$) have two boundary components h_1 and h_2 (resp. h'_1 and h'_2) that correspond to the copies of ν (resp. ν'). By Lemma 3.2.2, $\chi(M_\nu) = \chi(M_{\nu'})$. The classification theorem of surfaces implies that there exists a homeomorphism $h : M_\nu \to M_{\nu'}$ that maps h_1 to h'_1 and h_2 to h'_2. Gluing back M_ν and $M_{\nu'}$ along the boundaries, the map h induces a self homeomorphism on M that maps ν to ν'. This implies that ν and ν' have the same type. □

Lemma 3.2.5. *Two separating simple curves ν and ν' on an orientable surface M of genus g are of the same type if and only if there exists $0 \leq i \leq \lfloor \frac{g}{2} \rfloor$ such that both ν and ν' separate M into two components of genus i and $g - i$.*

Proof. Let us first assume that both ν and ν' separate M into components of genus i and $g - i$ for $0 \leq i \leq \lfloor \frac{g}{2} \rfloor$. The idea of the proof is the same as in the proof of Lemma 3.2.4: we first cut M along ν and ν' and using the classification of surfaces, we obtain homeomorphisms between the components of M_ν and $M_{\nu'}$; this is possible since M_ν and $M_{\nu'}$ have components of the same genus. This homeomorphism induces a self homeomorphism of M that maps ν to ν'.
Now let us assume that ν and ν' are of the same type. Then there exists a homeomorphism $h : M \to M$ for which $h(\nu) = \nu'$. Let $g(M^1_\nu) \leq g(M^2_\nu)$ and $g(M^1_{\nu'}) \leq g(M^2_{\nu'})$. Then h induces a homeomorphism between M^1_ν and $M^1_{\nu'}$ and a homeomorphism between M^2_ν and $M^2_{\nu'}$ which implies that $g(M^1_\nu) = g(M^1_{\nu'})$ and $g(M^2_\nu) = g(M^2_{\nu'})$. This finishes the proof. □

We say that the homeomorphism class of ν is (M_i, M_j)-separating if ν is a separating curve that cuts the surface into components of orientable genus i and j such that $i \leq j$.

Remark 3.2.6. *By the same technique as in the proof of Lemmas 3.2.4 and 3.2.5, we can see that no non-separating curve has the same type as a separating curve and that the list is exhaustive: a simple closed curve on an orientable surface of genus g is either non-separating or is of type (M_i, M_{g-i})-separating for some $0 \leq i \leq \lfloor \frac{g}{2} \rfloor$.*

Types of curves on non-orientable surfaces

On a non-orientable surface, the types of simple closed curves are more diverse than in an orientable one.

Let N be a non-orientable surface and γ be a curve on N. The curve γ is called *orienting* if by cutting along it we obtain an orientable surface. A curve that is not orienting is called *non-orienting*.

Remark 3.2.7. *An orienting curve cannot be separating. To see this, assume that γ is orienting and when we cut along it we obtain two orientable connected components. Each connected component contains a copy of γ as a boundary. Re-gluing along γ corresponds to attaching two orientable surfaces which gives us an orientable surface. This is a contradiction.*

The sidedness of an orienting curve depends on the genus of the surface. The following lemma analyzes the different cases where the genus is odd or even (see [54, Lemma 5.3]).

Lemma 3.2.8 ([54, Lemma 5.3])**.** *Let N be a non-orientable surface of genus g with h boundary components and let γ be an orienting closed curve. Let g_γ be the (orientable) genus and h_γ be the number of boundary components in N after cutting along γ.*

- *If g is odd, then γ is one-sided, $g_\gamma = \frac{g-1}{2}$, and $h_\gamma = h + 1$.*
- *If g is even, then γ is two-sided, $g_\gamma = \frac{g-2}{2}$, and $h_\gamma = h + 2$.*

Proof. The proofs for the cases of odd genus and even genus are quite similar, therefore, we only prove the case where g is odd. We know by definition that the non-orientable genus of N, g, is equal to $eg(N)$.

Let us denote by N_γ the surface that we obtain after cutting N along γ. Since γ is orienting, N_γ, is orientable. We know that the Euler genus of an orientable surface is twice its orientable genus, therefore $eg(N_\gamma) = 2g(N_\gamma)$ must be an even number. On the other hand, since γ is a non-separating closed curve we know by Lemma 3.2.3 that $eg(N_\gamma)$ is 1 or 2 less than $eg(N) = g$ depending on the sidedness of γ.

If g is odd, for $eg(N_\gamma)$ to be even, γ should be one-sided. Therefore, $g_\gamma = \frac{g-1}{2}$ and $h_\gamma = h + 1$. The proof for the case where g is even is similar. □

The following table classifies all different types of simple closed curves on a non-orientable surface depending on the parity of the non-orientable genus g of the surface and the sidedness of the curves.

	two-sided	one-sided
g is odd	separating non-separating	orienting one-sided non-orienting
g is even	non-separating non-orienting separating orienting	one-sided (non-orienting)

Note that this table does not distinguish different types of separating curves. As in the orientable case, the type of a separating curve is determined by the topology of the components that we obtain after cutting along the curve.

Let us denote the homeomorphism class of a separating curve on a non-orientable surface that cuts it into two non-orientable components of genus i and j for $0 < i \leq j$ by (N_i, N_j)-separating. Note that a separating curve on a non-orientable surface might cut the surface into one orientable component and one non-orientable component. In this case we denote the type of the curves by (M_i, N_j)-separating where i is the orientable genus of the orientable component and j is the non-orientable genus of the non-orientable one. A separating curve on a non-orientable surface cannot cut the surface into two orientable components since otherwise it would be orienting and by Remark 3.2.7, this is not possible.

Remark 3.2.9. *The proof that the list in this table is exhaustive is almost identical to the proof of the orientable case (see the proofs of Lemmas 3.2.4, 3.2.5 and Remark 3.2.6).*

3.2.2.3 Homotopy of curves

The classification of simple closed curves up to homeomorphism defined in the previous section is rather coarse. In this section, we define the concept of homotopy that provides a finer classification of simple curves, see Figure 3.7. This notion formalizes the intuitive idea of a continuous transformation between curves.

Two paths p and q with the same endpoints x and y on a surface S are *homotopic* if there is a continuous map $H : [0,1] \times [0,1] \to S$ such that $H(0,\cdot) = p$, $H(1,\cdot) = q$, $H(\cdot,0) = x$, and $H(\cdot,1) = y$.

This notion can be extended to loops with basepoints $x \in S$ (paths for which $x = y$). Note that in this case $H(\cdot,0) = H(\cdot,1) = x$ and each curve $H(t,\cdot)$ for $t \in [0,1]$ is a loop with basepoint x. We say that a curve p *bounds a disk* on the surface, if p is the boundary of a disk on the surface.

Lemma 3.2.10. *Let p and q be two disjoint simple curves with the same endpoints on a surface S such that $p.q$ is two-sided and bounds a disk. Then p and q are homotopic.*

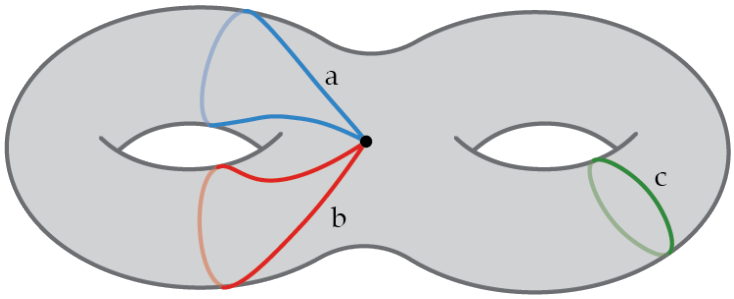

Figure 3.7: All the curves in this figure have the same homeomorphism type but none of them are homotopic. On the other hand, a and b are freely homotopic.

Proof. We can define the homotopy between the curves p and q on S to be the natural one on the disk that they bound. □

We consider loops on the surface S as maps $\theta : S^1 \to S$. Two loops ν and ν' are *freely homotopic* if there is a continuous map $H : [0,1] \times S^1 \to S$ such that $H(0,\cdot) = \nu$, $H(1,\cdot) = \nu'$. This notion of homotopy is coarser than the homotopy with fixed basepoints in the sense that two curves might be freely homotopic but not homotopic (with fixed basepoints), see Figure 3.7 for an example.

These homotopy relations partition the set of all curves on a surface into homotopy classes, where each class contains all curves that are homotopic to each other and are not homotopic to any curve outside the class.

A cycle is *contractible* if it is homotopic to a constant cycle. For simple curves, this homotopy class of curves coincides with the homeomorphism class of separating curves that cut a disk from the surface, i.e., a curve in this homotopy class separates a surface of genus g into two surfaces of genus 0 and genus g.

Lemma 3.2.11. *A simple closed curve c on a surface S is contractible if and only if it bounds a disk on S.*

Proof. For the forward implication, we refer the reader to [31, Theorem 1]. The reverse implication is straightforward as we can take the natural homotopy between c and a constant cycle based on a point on c, on this disk. □

An arc with endpoints on the same boundary component is contractible if it is homotopic to a path on the boundary.

We say that two homotopy classes of curves on a surface are *disjoint* if there exists a curve in each class such that the two curves are disjoint.

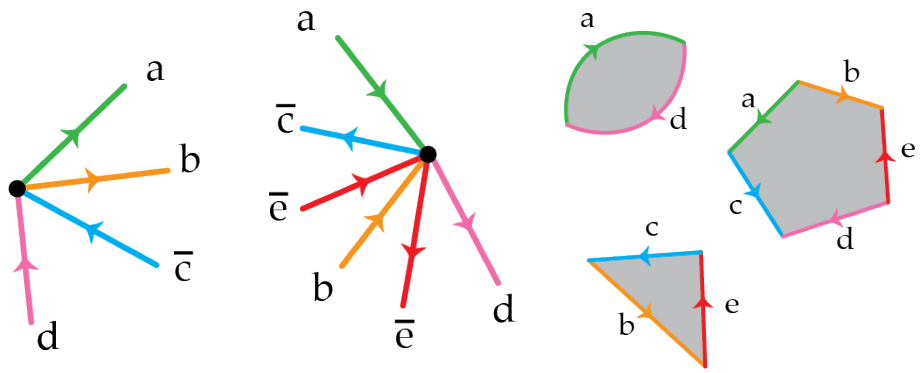

Figure 3.8: An embedding scheme with two vertices and its three faces.

3.3 A combinatorial model for graph embeddings

In this section, we define a notion in topological graph theory that is used to describe and encode embedding of graphs on surfaces.

Consider an embedding for a graph G and let $v \in V(G)$. We denote by ρ_v the cyclic permutation of edges incident to v that we visit when we rotate clockwise around v in this embedding. We denote by $\rho = \{\rho_v, v \in V(G)\}$ the *rotation system* of this embedding. Given a permutation π, we denote by $\overline{\pi}$ the permutation in which the order of elements in π is reversed.

In the case of graphs embedded on orientable surfaces, the embedding is completely encoded by the rotation system. On the other hand, to encode a graph embedding on a non-orientable surface, we need additional information on the edges, a function $\lambda : E(G) \to \{+1, -1\}$ which assigns a *signature* to each edge in G. These signatures encode the sidedness of the curves in the embedding.

Definition 3.3.1 (Embedding scheme)**.** *An* embedding scheme *consists of a triple* (G, ρ, λ) *in which* G *is a graph,* ρ *is the rotation system and* λ *is a function that assigns signatures to the edges of* G*.*

We use an overline over the label of an edge (for example $\overline{e}$) when it has signature -1, otherwise it has signature $+1$.

An embedding scheme completely encodes a cellular embedding up to homeomorphism. Throughout this thesis, whenever no confusion arises, we denote an embedding scheme by the label of the graph, e.g., G.

A closed curve in an embedding scheme is one-sided (resp. two-sided) if and only if the signatures of its edges multiply to -1 (resp. $+1$). Consequently, if for every cycle in the graph, the signatures of all the edges multiply to $+1$, we call it an *orientable scheme*; otherwise it is called a *non-orientable scheme*.

Recognizing the embedding from the embedding scheme. From the embedding scheme (G, ρ, λ), we can compute the faces of the embedding purely combinatorially as follows: direct the edges of the graph. Start at a vertex v, and choose an orientation $s \in \{-1, +1\}$. Walk on an arbitrary edge e with endpoints v and u, consider the vertex incident to u that is cyclically closest to v in the direction $s\lambda(e)$ ($+1$ for clockwise direction and -1 for counter clockwise). Update s to be $s\lambda(e)$ and then iterate this process until this walk cycles, passing again on the edge e from v to u when the orientation s matches the initial orientation at the start of the walk. This cycle corresponds to a face of the embedding. Choose a new edge and continue this process until every edge is met exactly twice. Figure 3.8 depicts a two-vertex embedding scheme with its faces. If we paste a convex polygon on this graph cycle we obtain a cellular embedding of the graph.

If (G, ρ, λ) is a non-orientable embedding scheme, we obtain a non-orientable surface. We denote the Euler genus and the non-orientable genus of this surface by $eg(G, \rho, \lambda)$ and $\tilde{g}(G, \rho, \lambda)$, respectively. We know that in this case $eg(G, \rho, \lambda) = \tilde{g}(G, \rho, \lambda)$. On the other hand, if (G, ρ, λ) is an orientable scheme, we obtain an orientable surface. We denote the Euler genus and the orientable genus of this surface by $eg(G, \rho, \lambda)$ and $g(G, \rho, \lambda)$ respectively. We know that in this case $eg(G, \rho, \lambda) = 2g(G, \rho, \lambda)$.

We sometimes consider non-cellular embeddings of graphs on a non-orientable surface such that one face has non-orientable genus 1. An embedding scheme does not completely encode such an embedding as it cannot distinguish the face with non-zero genus. Yet, in this thesis, the information encoded by embedding schemes suffices for our purposes in dealing with such embeddings. In all the cases in which we use this unconventional treatment for embeddings, all the closed curves in the embedding are two-sided (or equivalently, as we will explain in the following, the embedding schemes are orientable). To be more precise, let ϕ be a non-cellular embedding of a graph G on N_g in which only one face has genus 1 and all the closed curves in G are two-sided. We say that an orientable embedding scheme (G, ρ, λ) *realizes* ϕ if ρ respects the permutation of edges around vertices in ϕ and the signatures of the edges in all of the closed curves in G multiply to $+1$. In this case, g is the minimum genus of a non-orientable surface embedding realized by (G, ρ, λ). Throughout the thesis, we denote g by $\tilde{g}(G, \rho, \lambda)$ and call the *non-orientable genus of the orientable embedding scheme* (G, ρ, λ).

Lemma 3.3.2. *Let (G, ρ, λ) be an orientable embedding scheme with $g := g(G, \rho, \lambda) \neq 0$. Then $\tilde{g}(G, \rho, \lambda) = 2g + 1$. In particular, the non-orientable genus of an orientable scheme with non-zero orientable genus is odd.*

To prove this lemma, we need the following lemma from [69] which we state here without proof.

Lemma 3.3.3 ([69, Lemma 6])**.** *Let (G, ρ, λ) be an orientable embedding scheme. Then either $\tilde{g}(G, \rho, \lambda) = 0$, or $\tilde{g}(G, \rho, \lambda) \geq 3$ and is odd.*

Proof of Lemma 3.3.2. The embedding scheme has Euler genus $2g$, hence at least $2g$ cross-caps are required on a non-orientable surface for (G, ρ, λ) to realize an embedding on it. By Lemma 3.3.3, $\tilde{g}(G, \rho, \lambda)$ is odd so at least $2g + 1$ cross-caps are required, i.e., $\tilde{g}(G, \rho, \lambda) \geq 2g + 1$. To see that this number of cross-caps always suffices, we use the same approach as in the proof of Lemma 3.2.1. We begin with an embedding of G on an orientable surface M of orientable genus g and add a cross-cap in one of the faces of G on M; this gives us an embedding of G on a non-orientable surface with one cross-cap and g handles that is homeomorphic to a closed non-orientable surface of non-orientable genus $2g + 1$. □

Equivalent embedding schemes. Given an embedding scheme (G, ρ, λ), a *flip* at a vertex v yields another embedding scheme of the same graph, in which we reverse the order of the edges incident to v and invert the signature of those edges incident to v that are not loops. We say that two embedding schemes (G, ρ, λ) and (G, ρ', λ') are *equivalent* if one can transform one to the other by a sequence of flips. We state the following lemma from [58] without proof.

Lemma 3.3.4 ([58, Theorem 3.3.1])**.** *Two embedding schemes are equivalent if and only if they induce the same embedding up to homeomorphism.*

Intuitively, flipping a vertex corresponds to dragging it through a cross-cap on the surface. This justifies that equivalence classes of embedding schemes and embedded graphs can be considered as being two representations of the same object and we will switch freely between the two points of views.

3.3.1 Contracting a tree in an embedding scheme

As in many similar works, the first step in some of our results is to contract a spanning tree of the underlying graph, reducing the problems to the setting of one-vertex graphs embedded on a non-orientable surface. These embeddings are described by a one-vertex embedding scheme which are much easier to deal with; in Section 5.2 we describe these embedding schemes with more detail.

Note that in this thesis, the graphs that we embed on our surfaces are connected. Let (G, ρ, λ) be an embedding scheme in which G is connected and let T be a spanning tree in G. Here we describe the one-vertex embedding scheme that we obtain by contracting T.

We contract the edges of T one by one. Let $e = vw$ be an edge in T and let G_e be the embedding scheme that we obtain by contracting e; after the contraction, v and w are

merged into a single vertex z. If e is two sided, the cyclic permutation of edges around the vertex z is given by merging ρ_v and ρ_w as follows: If e has signature $+1$ we merge ρ_v and ρ_w from the the place where e appears in each and remove e, see Figure 3.9. If e has signature -1, we first flip one of v and w and then do as in the previous case. We continue this merging process until we have one vertex left.

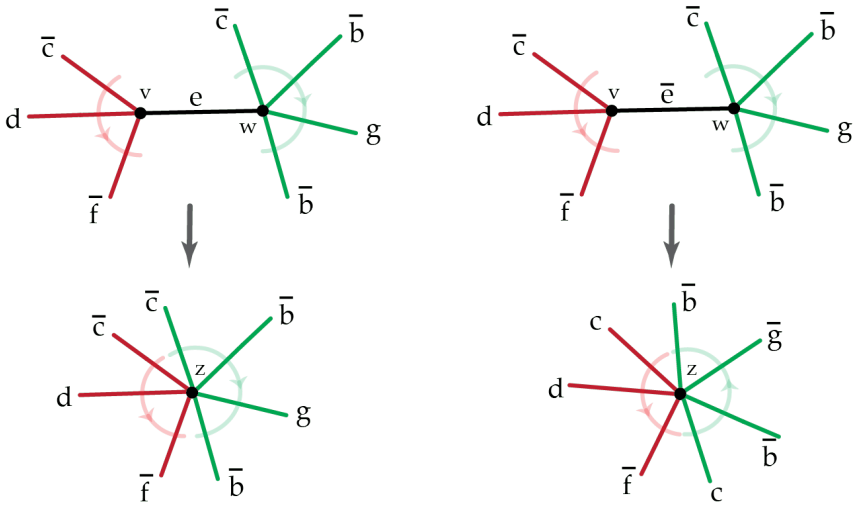

Figure 3.9: Contracting an edge e in the embedding scheme at the top reduces the number of vertices by 1 and yields the embedding scheme at the bottom. At right e has signature -1 and at left it has signature $+1$.

3.3.2 Contracting a boundary

In this thesis, by *contracting a boundary* we mean to replace it by a point. Let S be a surface with boundary and G be a graph embedded on it. Let $v_1, \ldots, v_k$ be the vertices of G that are on the boundary h of S, and let $\rho_1, \ldots, \rho_k$ be the cyclic permutation of edges around them respectively. Here we describe the contraction of the boundary h and its effect on G: replace the boundary with a vertex, denoted as v, which is formed by merging $v_1, \ldots, v_k$. The cyclic permutation of edges at this vertex corresponds to the merging of $\rho_1, \ldots, \rho_k$ as we see them while rotating around the boundary h. Figure 3.10 illustrates this process.

3.4 A geometric model for graph embeddings on surfaces

In this section, we describe geometric models for representing embeddings of graphs on surfaces: the *cross-cap model* for non-orientable surfaces in which we localize the cross-caps on the sphere and the *box model* for orientable surfaces in which we localize the handles on the sphere. We also provide discussions about certain crossing numbers of

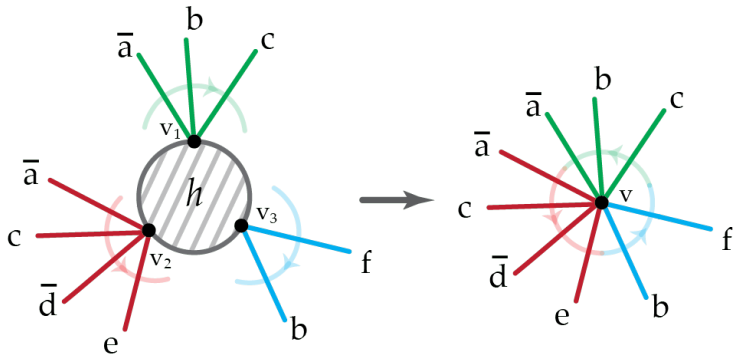

Figure 3.10: Contracting a boundary.

graphs on the plane that are closely related to these models.

3.4.1 The cross-cap model for non-orientable surfaces

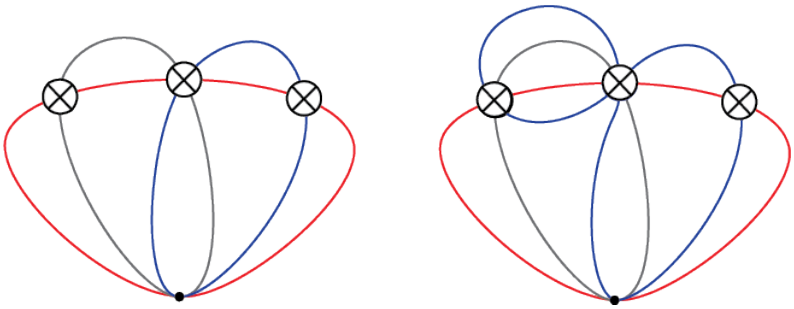

Figure 3.11: Two cross-cap drawings realizing the same embedding scheme.

Following Schaefer and Štefankovič [69], we will use a model with localized cross-caps to represent graph embeddings on non-orientable surfaces as a drawing on the sphere. By removing a point from the sphere, we obtain a planar drawing with geometrically drawn cross-caps; we consider the cross-caps as additional vertices on the edges of the graph which we depict using ⊗ and also call cross-caps. Two edges of the graph that meet at a cross-cap, leave it with a reversed order. Considering the cross-caps as new vertices and the edges of the drawing as the sub-edges in the initial graph, we obtain our planar drawing which we call a *cross-cap drawing*, see Figure 3.11 for examples of cross-cap drawings of a one-vertex graph.

Obtaining the non-orientable embedding from a cross-cap drawing. Let D be a cross-cap drawing of a graph G on the sphere with g cross-caps which we now think of as small disks⊗. We quotient the boundary of each ⊗ by the antipodal map of the small disk

(this pastes the end of each edge that passes through the cross-cap to its other end that exits the cross-cap). This is possible since the edges passing through a cross-cap exit it in the reversed order. We obtain a non-orientable graph embedding of genus g.

Topologically, this process can be seen as attaching a Möbius band on the boundary of each $\otimes$ and extending the edges inside the Möbius band so as to get a valid graph embedding. Figure 3.12 depicts how to extend the edges inside the Möbius band.

Given a cross-cap drawing D with g cross-caps for a graph G, let $\phi(G)$ be the aforementioned embedding of G on N_g. We say that an embedding ϕ' for a graph G *admits* the cross-cap drawing D, if there is a homeomorphism $f : N_g \to N_g$ such that $f(\phi'(G)) = \phi(G)$.

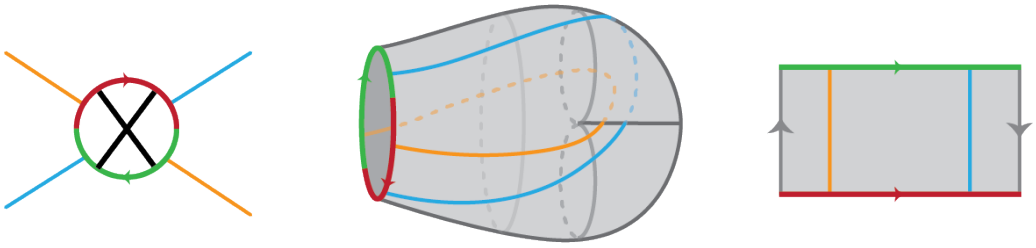

Figure 3.12: The process of obtaining a non-orientable embedding from a cross-cap drawing by attaching the Möbius band on the boundary of $\otimes$. In the middle and right picture, we see the Möbius band with its boundary identified by red and green segments which correspond to the colors in the boundary of $\otimes$ depicted at left.

Obtaining a cross-cap drawing from a non-orientable embedding. A *planarizing system of disjoint one-sided curves* on a non-orientable surface, abbreviated *PD1S*, is a system of g disjoint one-sided curves such that by cutting along them, we obtain a sphere with g holes (this was first introduced by Mohar [57]). By cutting along such a system, from any graph embedded on a non-orientable surface, we obtain a sphere with g boundary components and therefore, a planar representation for the graph. Replacing each boundary by a $\otimes$, we obtain a cross-cap drawing of the graph.

We say that a cross-cap drawing *realizes* an embedding scheme (G, ρ, λ) or the embedding of G if it is a cross-cap drawing of (G, ρ, λ) or of an equivalent scheme (under flips), that respects the cyclic permutations and signatures on the edges, i.e., if an edge has signature $+1$ (resp. -1) then it enters an even (resp. odd) number of cross-caps.

Remark 3.4.1. *The correspondence between cross-cap drawings and embeddings is not one to one. While a cross-cap drawing uniquely describes an embedded graph, the converse is not true. For example different PD1S systems give different cross-cap drawings for the same embedding (or embedding scheme), see Figure 3.11.*

3.4.1.1 Recognizing types of curves in a cross-cap drawing

In this section, we provide tools to recognize the type of curves in a given cross-cap drawing. Notice that a closed curve in a cross-cap drawing is one-sided (resp. two sided) if and only if it enters an odd (resp. even) number of cross-caps (where this number is counted with multiplicity). Considering the neighborhood of a closed curve as a band, each cross-cap that the curve enters, introduces a half twist in the band, see Figure 3.13. We know by the classification of surfaces (Theorem 3.1.1) that a band with an odd (resp. even) number of half-twists is homeomorphic to the Möbius band (resp. the annulus). This implies that a curve enters an odd (resp. even) number of cross-caps, if and only if it has a neighborhood homeomorphic to the Möbius band (resp. the annulus) and therefore is one-sided (resp. two-sided).

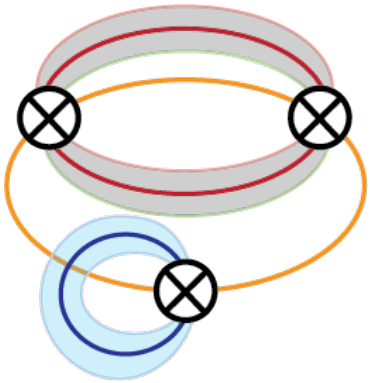

Figure 3.13: In the given cross-cap drawing, the blue and yellow curve are one-sided and the red curve is two-sided. The gray area depicts a neighborhood of the red curve that is homeomorphic to the annulus and the blue area is a neighborhood of the blue curve, homeomorphic to the Möbius band.

The following lemma helps us recognize orienting and separating curves in a cross-cap drawing. Another proof can also be found in [69, Lemma 3 and 4].

Lemma 3.4.2. *Let G be an embedding scheme. In any cross-cap drawing of G,*

1. *a closed curve is separating if and only if it passes through each cross-cap an even number of times.*

2. *a closed curve is orienting if and only if it passes through each cross-cap an odd number of times.*

Proof. Actually this statement does not depend on the whole embedding scheme, only on the loops. It is enough to show that:

- A cross-cap drawing of a separating closed curve passes through each cross-cap an even number of times.

- A cross-cap drawing of an orienting closed curve passes through each cross-cap an odd number of times.

Notice that an edge entering a cross-cap k times, divides the region around the cross-cap into $2k$ parts locally. We call these parts the *wedges* around that cross-cap. We number the wedges from 1 to $2k$ as going around the cross-cap. Two wedges $i > j$ are called *opposite* if $i - j = k$. A curve entering the cross-cap from a wedge, exits the cross-cap in its opposite wedge.

1. First, let us show that if the curve γ is separating, then it enters each cross-cap an even number of times. Observe that if γ is one-sided it cannot be separating, moreover γ can separate the surface into at most two connected components, one for each side of γ and every time we cross γ we should change connected components. Now consider a cross-cap drawing of γ, and assume, in order to reach a contradiction, that some cross cap is crossed an odd number of times by γ. Choose any wedge around that cross cap and notice that if we go around the cross-cap with a curve γ', to reach the opposite wedge defined by γ, then γ' and γ intersect an odd number of times, so they should be in different connected components of the complement of γ. This is a contradiction since opposite wedges are clearly in the same connected component.

   Now, let us show that if a closed curve γ enters each cross-cap an even number of times, it is separating. Let us consider the dual graph to the cross-cap drawing of the curve γ (we consider the cross-cap drawing as a planar drawing with cross-caps treated as vertices). Since γ enters each cross-cap an even number of times, then the dual face corresponding to each cross-cap has faces of degree divisible by four. In particular, it is bipartite and admits a two coloring of vertices such that no two adjacent vertices have the same color and two vertices corresponding to opposite wedges around a cross-cap get the same color, see Figure 3.14. Such a coloring for the vertices of the dual graph thus induces a partition of the faces of the embedding of γ in two regions. These two regions are two connected components obtained by cutting along γ, thus γ is separating.

2. Let us first show that if γ is orienting then it enters each cross-cap an odd number of times. Consider a closed curve γ' that enters each cross-cap once. We first show that any such curve is orienting. The curve γ' divides the plane into two regions, see Figure 3.15. Any curve starting in one region has to cross an even number of cross-caps to end up in the same region. This implies that any closed curve on this surface that does not cross γ' is two-sided, which implies that γ' is orienting.

   Let γ be an orienting curve, and denote by $\alpha_1, \ldots, \alpha_g$ the PD1S underlying the cross-cap drawing. By the classification of surfaces, γ is unique up to homeomorphism,

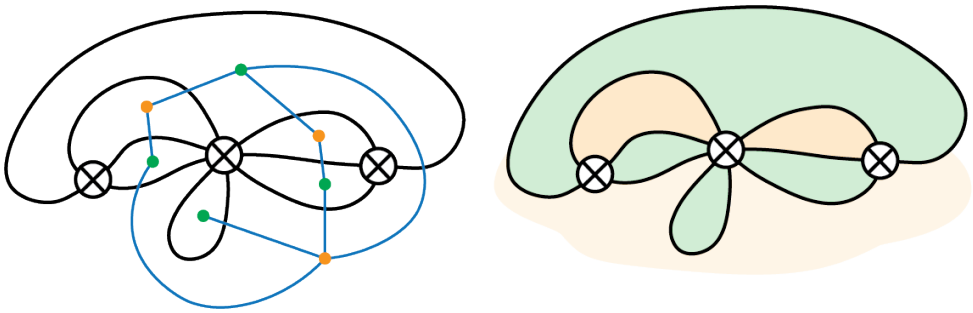

Figure 3.14: The cross-cap drawing of a separating curve and the coloring for its dual graph.

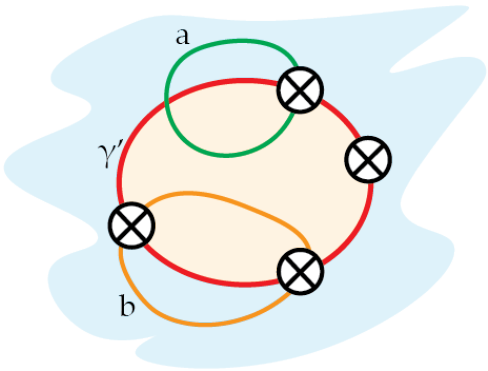

Figure 3.15: The cross-cap drawing of an orienting curve γ' that enters each cross-cap exactly once. The two-sided curve b does not cross γ' but the one-sided curve a crosses it once.

i.e., there exists a homeomorphism of the surface that maps γ to γ'. This gives us (another) cross-cap drawing where γ goes exactly once through every cross-cap. The image of a curve α_i under this homeomorphism is a simple closed curve on the same surface, which in this new cross-cap drawing intersects γ and the new cross-caps. Furthermore, it intersects the new cross-caps an odd number of times since it is one-sided. Now it is immediate that in this new representation, γ partitions the plane into two regions, and any curve crossing the cross-caps an odd number of times must cross γ an odd number of times. The curves $\alpha_1, \ldots, \alpha_g$ crossing γ an odd number of times, correspond to γ entering each cross-cap an odd number of times. This concludes the proof of the forward implication.

Now we prove that any closed curve that crosses each cross-cap an odd number of times, is orienting. Let us again consider the dual graph of the cross-cap drawing as in the separating case in the first part of this proof. Since each cross-cap is crossed an odd number of times, each cross-cap is adjacent to an even number of half-edges that is not divisible by four. Therefore the dual graph has faces of even degree not divisible by four. In particular, it is bipartite, and the vertices admit a two coloring in which no adjacent vertices have the same color. In any such coloring, the colors for two opposite wedges around each cross-cap is different which implies that a curve

entering a cross-cap, goes to a region with the opposite color. Therefore a closed curve that does not cross γ, has to enter an even number of cross-caps to end up in the colored region from which it started. This shows that on the surface that we obtain after cutting along γ, all simple closed curves are two-sided. Hence γ is orienting.

□

Remark 3.4.3. *Lemma 3.4.2 answers to a question asked in [60, page 134] regarding the recognition of an orienting loop in a cross-cap drawing. Particularly, it confirms the hunch that is discussed after the question.*

3.4.1.2 Genus crossing number and degenerate crossing number

One way to interpret a cross-cap drawing is to consider it in relation to the crossing numbers in planar drawings of graphs. Here we introduce two crossing numbers that are related to non-orientable embeddings of graphs.

As in the introduction, the *genus crossing number* of a graph G, denoted by $gcr(G)$, is the minimum number of self-intersections of G over all its planar drawings (where intersections are transverse and multiple edges crossing in one point are counted as one crossing, see the crossing of edges in the left picture in Figure 3.16). If we add the restriction that the edges of G are simple, i.e., do not self-intersect, this introduces the *degenerate crossing number* of G, denoted by $dcr(G)$.

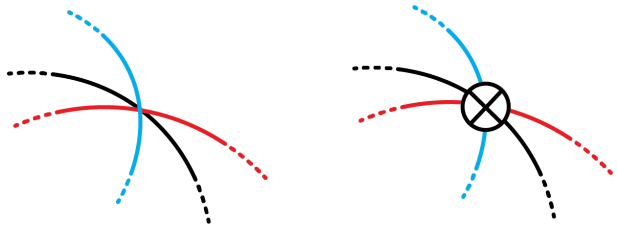

Figure 3.16: From cross-cap drawings to crossing numbers and vice versa.

Mohar proved the following lemma that reveals a nice relation between these crossing numbers in planar drawings and graphs embedded on surfaces.

Lemma 3.4.4 ([57, Theorem 2.1])**.** *For any non-planar graph G, $gcr(G) = \tilde{g}(G)$.*

Proof. Consider an embedding of G on a non-orientable surface of genus $\tilde{g}(G)$. There exists a cross-cap drawing with $\tilde{g}(G)$ cross-caps that realizes this embedding of G. Looking at a cross-cap as a crossing point for edges entering it, this cross-cap drawing provides a planar drawing of G with $\tilde{g}(G)$ crossings. This implies that $gcr(G) \leq \tilde{g}(G)$.

For the other side of the inequality, assume that a planar drawing of G with $gcr(G)$ crossings is given. Replace each crossing by a $\bigotimes$. This gives us a cross-cap drawing with $gcr(G)$ cross-caps that in turn realizes an embedding of G on a non-orientable surface of genus $gcr(G)$. This implies that $\tilde{g}(G) \leq gcr(G)$. Figure 3.16 illustrates the process in this proof. □

The condition of edges being simple in the definition of degenerate crossing number translates into restricting the edges in a cross-cap drawing to pass through each cross-cap at most once.

3.4.2 The box model for orientable surfaces

In this section, we extend the notion of cross-cap localization to a localization for handles. The ideas come from [1] on the bundled crossing number which we explain in the next section.

In this model, a handle is replaced by a hexagonal box ⬡ on the sphere such that the sides of this box correspond to each other two by two with the order $abc\bar{a}\bar{b}\bar{c}$ meaning that a box transfers an edge entering from one side to its corresponding side without crossing any other edge that enters the box. A *box drawing* of an embedding scheme G of orientable genus $g(G)$, is a planar drawing of G with $g(G)$ hexagonal boxes in which boxes are considered as additional vertices and the edges in the planar drawing are sub-edges in G.

Obtaining the orientable embedding from a box drawing. Given a box drawing with g boxes on the sphere, we paste the sides of the hexagon along the word $abc\bar{a}\bar{b}\bar{c}$. This gives us an orientable surface of genus g.

Topologically, this corresponds to attaching a handle on the boundary of each hexagonal box and extending the edges passing through the box without making them cross. The following lemma implies that this is always possible and justifies the hexagonal representation of the boxes.

Lemma 3.4.5. *On an orientable surface of genus* 1 *with* 1 *boundary component (a handle) there exists at most three simple non-homotopic non-contractible disjoint arcs.*

Proof. The middle picture in Figure 3.17 illustrates three arcs w, l and f on a handle that are pairwise non-homotopic. The right picture represents the handle: by gluing the polygon along the oriented black and gray side, we retrieve the handle. We can see in this representation that any arc that we draw on this surface that is disjoint from w, l and f is homotopic to one of them. □

The three arcs in the lemma subdivide the boundary of a handle (the punctured torus) to 6 segments that can be put to correspondence with the 6 sides of the hexagonal box. By doing so, we obtain a map from each box on the sphere (viewed as a boundary component) to the boundary of the handle. Edges that pass through the box, entering it from the side a (resp. b or c) can be extended along the arc l (resp w or f) inside the handle without crossing each other, see Figure 3.17.

Given a box drawing D with g boxes for a graph G, let $\phi(G)$ be the embedding of G on M_g that we obtain as described. We say that an embedding ϕ' of a graph G *admits* the box drawing D, if there is a homeomorphism $f : M_g \to M_g$ such that $f(\phi'(G)) = \phi(G)$.

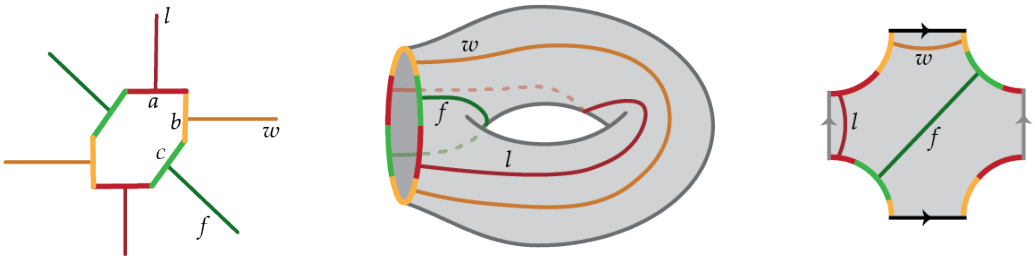

Figure 3.17: The correspondence between a box and the boundary of a handle.

Obtaining a box drawing from an orientable embedding. An orientable surface of genus g can be modeled using a $4g$-gon in the plane with sides labelled by $a_1b_1\overline{a}_2\overline{b}_2 \dots a_gb_g\overline{a}_g\overline{b}_g$ such that the sides a_1 and $\overline{a}_1$ are identified in opposite direction[1]. See the left picture in Figure 3.18, which depicts an embedding of a graph with three vertices on an orientable surface of genus 2. A graph embedding on an orientable surface of genus g in this model can be turned into a box drawing as follows: each 4 sides $a_ib_i\overline{a}_i\overline{b}_i$ of the $4g$-gon can be treated as 4 sides a,b,$\overline{a}$ and $\overline{b}$ in one hexagonal box $abc\overline{a}\overline{b}\overline{c}$. An edge in the embedding can be extended as depicted in the right picture in Figure 3.18 to obtain a box drawing.

Remark 3.4.6. *This suggests that a quadrilateral box suffices to model orientable embeddings of graphs which is actually how it is dealt with in [1]. We adopt hexagonal boxes as they give us nicer planar drawings (this is justified in Lemma 3.4.5). Note that an edge entering a box $abc\overline{a}\overline{b}\overline{c}$ from the the side c, can be drawn in a quadrilateral box $ab\overline{a}\overline{b}$ by entering the box twice, once from a and once from b.*

We say that a box drawing *realizes* an orientable embedding of G on a surface of genus g if it has g boxes and respects the cyclic permutations of edges around vertices of G in the embedding.

[1]This is called a canonical polygonal scheme which we introduce later in this chapter.

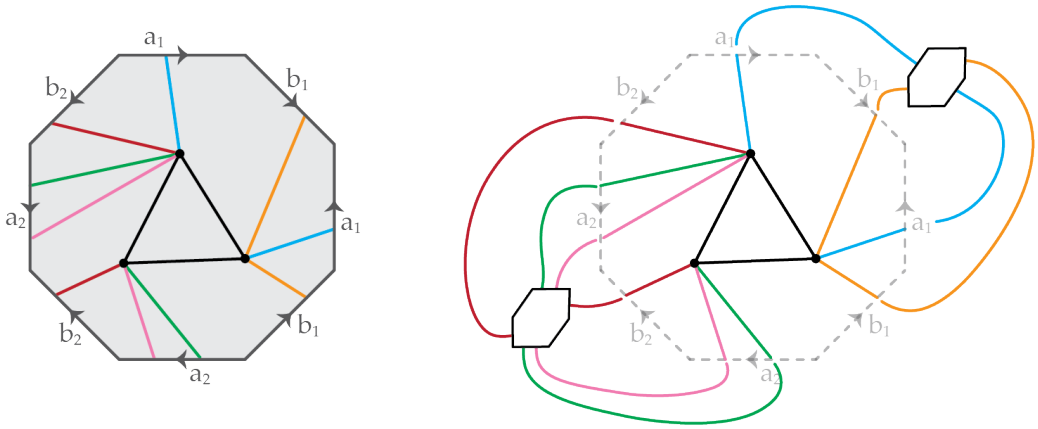

Figure 3.18: From an embedding to a box drawing.

3.4.2.1 Bundled crossing number

A *bundled crossing* of a graph drawing refers to a collection of intersections that occur between two sets of parallel edges, see Figure 3.19. The *bundled crossing number* of a graph is the minimum number of bundled crossings in all planar drawing of the graph such that multiple crossings and self-intersections of edges are allowed. We denote the bundled crossing number of a graph G by $bc(G)$. The following lemma connects this crossing number with the orientable genus of graph.

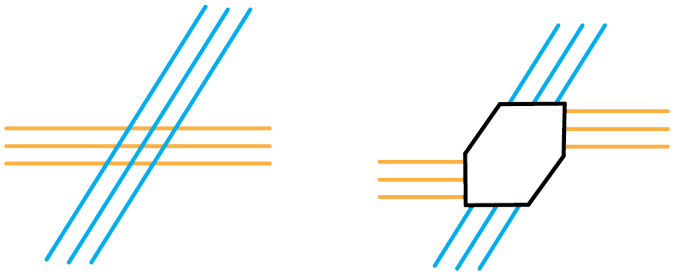

Figure 3.19: From box drawings to bundled crossing number and vice versa.

Lemma 3.4.7 ([1, Theorem 1])**.** *For every graph G, $bc(G) = g(G)$.*

Proof. Consider an embedding of G on an orientable surface of genus $g(G)$. There exists a box drawing with $g(G)$ boxes that realizes this embedding of G such that each box has a side that is not used by any edge of G (see Figure 3.18 and Remark 3.4.6). Looking at each box as a crossing point for edges entering it as depicted in Figure 3.19, this box drawing provides a planar drawing of G with $g(G)$ crossings. This implies that $bc(G) \leq g(G)$.

For the other side of the inequality, assume that a planar drawing of G with $bc(G)$ bundled crossings is given. Replace each crossing by a ▱. This gives us a box drawing with $bc(G)$ boxes that in turn realizes an embedding of G on an orientable surface of

genus $bc(G)$. This implies that $g(G) \leq bc(G)$. Figure 3.19 illustrates the process in the proof. □

3.5 Metrics on surfaces

In this section, we look at our surfaces through a geometric lens and introduce continuous and discrete metrics on our surfaces.

3.5.1 Discrete Metrics

As we mentioned in the introduction, a graph that is cellularly embedded on a surface S defines a discrete metric on S. In this section, we introduce two models that are used to describe this metric and show that these two models are equivalent up to the duality of the graph embeddings. To obtain a more comprehensive understanding of these models, we refer the reader to the work of Éric Colin de Verdière on algorithms for graphs on surfaces [19] and the work of Francis Lazarus on combinatorial graphs and surfaces [52].

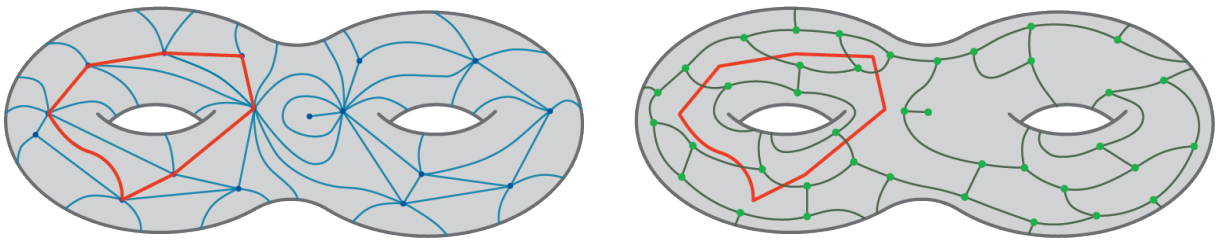

Figure 3.20: A curve (in red) depicted in both the combinatorial model (left) and the cross-metric model (right). The figure also shows the equivalence of the two models.

3.5.1.1 Combinatorial Surface

The first model is called the *combinatorial model*. A *combinatorial surface* is a surface S together with a weighted graph G which is cellularly embedded on S. In the case that S has boundary, G is embedded so that the boundary is the union of some edges of G.

The curves we consider in this model are the walks on the graph G and the length of a curve C is the the sum of the weights of the edges of G traversed by C. We emphasize that in this model, a curve might traverse an edge multiple times even if it is a simple curve. The red curve in the left picture in Figure 3.20 shows a curve in this model where the discrete metric on the surface is given by the triangulation.

3.5.1.2 Cross-metric Surface

A *cross-metric surface* is a surface S together with a weighted graph G which is cellularly embedded on S. In the case that S has boundary, G is embedded so that the boundary is the union of some edges of G. The curves that we consider in this model are those that cross G transversely and away from the vertices, and the length of a curve C is the the sum of the weights of the edges of G that C crosses. The red curve in the right picture in Figure 3.20 shows a curve in this model.

3.5.1.3 Equivalence by duality

As we mentioned these two models are equivalent up to duality of the graph embeddings. To any combinatorial surface S without boundary and its graph G, we can associate by duality a cross-metric surface by considering S with the graph G^*, where G^* is the dual graph of G and each dual edge e^* has the same weight as e. This transformation preserves the lengths of the curves, see Figure 3.20.

We can extend this transformation to the case of surfaces with boundaries. To do so, we extend our definition for dual graphs for embedded graphs on a closed surface in Section 3.2.1.2 to the case where the surface has boundary and the boundary (or boundaries) is the union of some edges of the embedded graph. We associate to each face of G, and each edge on the boundary a vertex, such that in the latter the vertex lies on the interior of the edge. We associate to each boundary edge e, an edge e^* that connects the vertex associated to e and the vertex associated to its incident face in G. We also associate an edge to each vertex in G on the boundary, connecting the associated vertices to its incident boundary edges; these edges build the boundary edges of G^*. Finally, we associate an edge to each non-boundary edge in G, such that it connects the vertices associated to its incident faces. Each dual edge e^* has the same weight as e and each e^* on the boundary has infinite weight.

In both models, we refer to the cellularly embedded graph G on the surface S, as the *primal* graph on S. Although these models are essentially the same up to duality, depending on the use, each has its advantages and drawbacks. In this thesis, we utilize both models, although the cross-metric model is employed far more than the combinatorial one due to its greater suitability for our purposes. This follows from the fact that in the combinatorial model, the places where two curves cross is not encoded properly while the cross-metric model is based on the crossings between the curves on the surface.

Multiplicity. In the cross-metric model, the *multiplicity* of a curve, respectively of a system of curves, at some edge e of G is the number of times e is crossed by the curve, respectively the sum of all the intersections of e with the curves of the system. The

multiplicity of a curve (or a system of curves) is the maximal multiplicity of the curve (curves) at any edge e of G.

3.5.2 Continuous Metrics

We use some notions from Riemannian geometry which we introduce briefly in this section. For more background we refer the reader to the book of do Carmo and Francis [26].

A Riemannian metric introduces smooth distances between points on the surface. Specifically, at each point x on a surface S, a Riemannian metric involves an inner product g_x on the tangent space $T_x(S)$ that varies smoothly as x moves on S.

This inner product induces a length on the smooth curves on the surface in which the length of a path p on S is $\int_0^1 \sqrt{g_x(p'(t), p'(t))}dt$. The distance between two points x and y is the infimum length among all smooth paths from x to y. A *shortest path* between x and y is a path that realizes the distance between x and y. Two shortest paths in a Riemannian metric cross at most once.

Informally, a *geodesic* on a Riemannian surface is a path or a closed curve that is locally minimal. Note that since the minimality condition is considered only locally in this definition, a geodesic may not be a shortest path.

From an extrinsic point of view, a Riemannian metric can be seen as a metric structure that is induced from an embedding of the surface in $\mathbb{R}^d$. This is obtained by the Nash embedding theorems [42] which state that every smooth Riemannian manifold can be smoothly isometrically embedded into some Euclidean space. Although this result unifies the intrinsic and extrinsic points of view of Riemannian geometry, and it provides a better intuition, it is not easier to work with compared to the intrinsic definition.

3.6 Decompositions of surfaces

Informally, a *decomposition* of a surface is any shape of cut that dissects the surface to a topologically simpler piece or pieces. The decompositions that we deal with are either given by cellularly embedded graphs (such as canonical decompositions or an octagonal decomposition) or by a set of simple disjoint closed curves (such as a pants decomposition). This section aims to consolidate all the decompositions encountered by the reader throughout this thesis. We consider all of our decompositions in the cross-metric setting.

Short decompositions. We say that a decomposition of a cross-metric or combinatorial surface is *short*, if each edge in the graph of the decomposition has constant multiplicity. All the decompositions that we work with, are comprised of $O(g)$ edges, which implies

that a short decomposition has total length $O(g|E(G)|)$ in which G is the primal graph on the surface.

3.6.1 Decompositions along systems of loops

In this thesis, we extensively study the decomposition of surfaces along a system of loops, that is, a one-vertex embedding scheme such that by cutting along the loops we obtain a topological disk.

Specifying a system of loops and a cyclic order of the edges around the basepoint is the same as specifying a polygon and data indicating how to glue the edges so as to recover the surface: such a polygon is called a *polygonal scheme*. See Figures 3.21 and 3.22 for examples of polygonal schemes.

Orientable canonical decomposition. For the orientable surface M_g, we define an *orientable canonical system of loops* to be a family of two-sided loops with cyclic order $a_1b_1a_2b_2 \dots a_gb_ga_gb_g$ around the basepoint. Cutting M along this family yields a topological disk. Decomposing an orientable surface along such system of loops is called a *canonical decomposition* of the orientable surface. This system of loops corresponds to a decomposition of the surface with associated polygonal scheme $a_1b_1\overline{a}_1\overline{b}_1 \dots a_gb_g\overline{a}_g\overline{b}_g$ which is called the *orientable canonical polygonal scheme*. Figure 3.21 depicts such a decomposition.

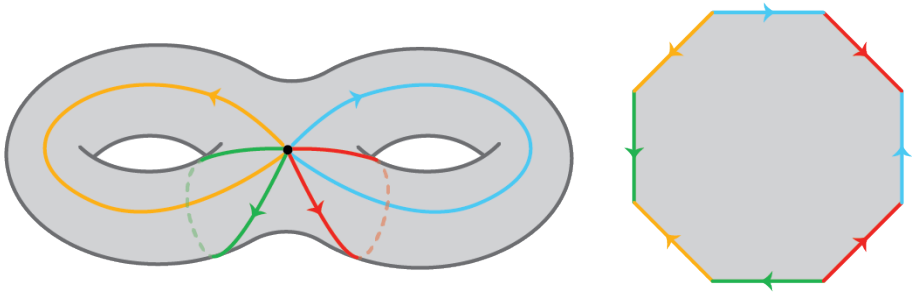

Figure 3.21: Canonical decomposition of an orientable surface of genus two.

Non-orientable canonical decomposition. For the non-orientable surface N_g, a *non-orientable canonical system of loops* is a family of one-sided loops with cyclic order $a_1a_1a_2a_2 \dots a_ga_g$ around the basepoint. Cutting N along this family of loops yields a topological disk. This system of loops corresponds to a decomposition of the surface with associated polygonal scheme $a_1a_1a_2a_2 \dots a_ga_g$ which is called the *non-orientable canonical polygonal scheme*. Figure 3.22 depicts such a canonical decomposition.

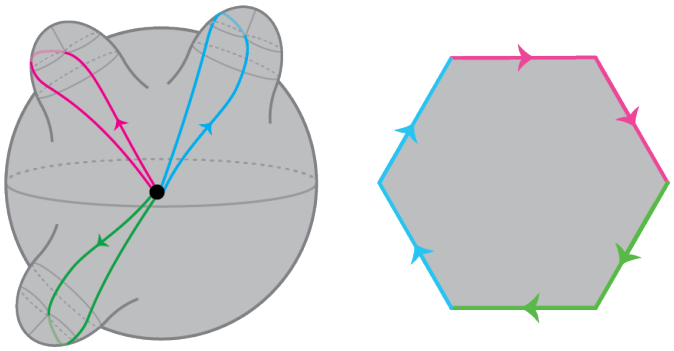

Figure 3.22: Canonical decomposition of a non-orientable surface of genus three.

Other decompositions along systems of loops. Given a non-orientable surface of genus g, we also study the decompositions of this surface along a system of loops with associated polygonal schema $a_1a_1a_2a_2\cdots a_la_lb_1c_1\bar{b}_1\bar{c}_1\ldots b_kc_k\bar{b}_k\bar{c}_k$ where $l \neq 0$ and $g = l + 2k$ (l and g have the same parity).

3.6.2 Octagonal and hexagonal decomposition

Another decomposition of surfaces that we are interested in, is one that decomposes the surface into sets of disks, in this case, sets of octagons and hexagons.

An *octagonal decomposition* of an orientable surface is an arrangement of closed curves (not necessarily disjoint) that decomposes the surface into octagonal faces and the overlayed graph of this decomposition is the one depicted in the left picture in Figure 3.23. Every vertex in this graph has degree four. An orientable surface of genus g can be decomposed to $2g-2$ octagons.

A *hexagonal decomposition* of an orientable surface is an arrangement of closed curves that decomposes the surface into hexagonal faces such that the overlayed graph of this system of curves is the one depicted in the right picture in Figure 3.23. Every vertex in this graph has degree four. An orientable surface of genus g can be decomposed to $4g-4$ hexagons.

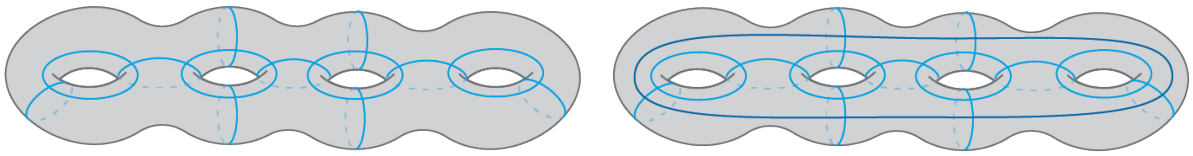

Figure 3.23: Left: an octagonal decomposition of M_4, right: a hexagonal decomposition of M_4.

A non-orientable surface can be decomposed into $2g-4$ hexagons when g is even and into $2g-6$ hexagons and 4 pentagons when g is odd. Figure 4.7 shows such a decomposition

of a non-orientable surface. In Chapter 4, we will generalize these decompositions to the setting of non-orientable surfaces.

3.6.3 Pants decomposition

A sphere with three boundary components is called a *pairs of pants*. A *pants decomposition* of a surface is a family of disjoint closed curves such that cutting the surface along all of the them gives a disjoint union of pairs of pants. Pairs of pants are the simplest possible surfaces that one can obtain after cutting along simple closed curves, and thus constitute fundamental building blocks in the study of surfaces.

The Euler characteristic of a pairs of pants is equal to -1. This implies that a surface $S_{g,b}$ can be decomposed to exactly $2g + b - 2 = -\chi(S)$ pairs of pants. Any pants decomposition of this surface consists of $3g - 3 + b$ disjoint closed curves. Two pants decompositions are of the same *type* if there exists a homeomorphism of the surfaces that maps one to the other. The homeomorphism type of a pants decomposition is entirely determined by its *intersection graph*: a graph with vertices corresponding to each pairs of pants in the surface in which two vertices are connected if the corresponding pairs of pants have a common boundary curve. Since each pairs of pants has three boundary components, the intersection graph of a pants decomposition of a closed surface is a trivalent graph. Similarly, the intersection graph of a pants decomposition of a punctured sphere is a trivalent tree, that is a tree in which each non-leaf edge has degree three. Figure 3.24 depicts a pants decomposition of a closed surface and a punctured sphere together with their corresponding intersection graphs.

The following lemma gives an estimate for the number of different types of pants decompositions of an orientable surface of genus g that is obtained by computing the number of different trivalent graphs. By $f(n) \approx g(n)$, we mean that f and g are equal up to an exponential factor in n.

Lemma 3.6.1. *The number of trivalent graphs with $2n$ vertices is $\approx n^n$.*

Proof. For the proof see [9] or the proof of [41, Lemma 1]. □

3.7 Decomposing a non-orientable surface along an orienting curve

Orientable surfaces and graphs embedded on them are better studied than non-orientable ones. One of the main techniques that we use in this thesis to deal with non-orientable surfaces is to cut the surface along an orienting curve to turn it to an orientable surface. This allows us to generalize some results on orientable surfaces to the non-orientable

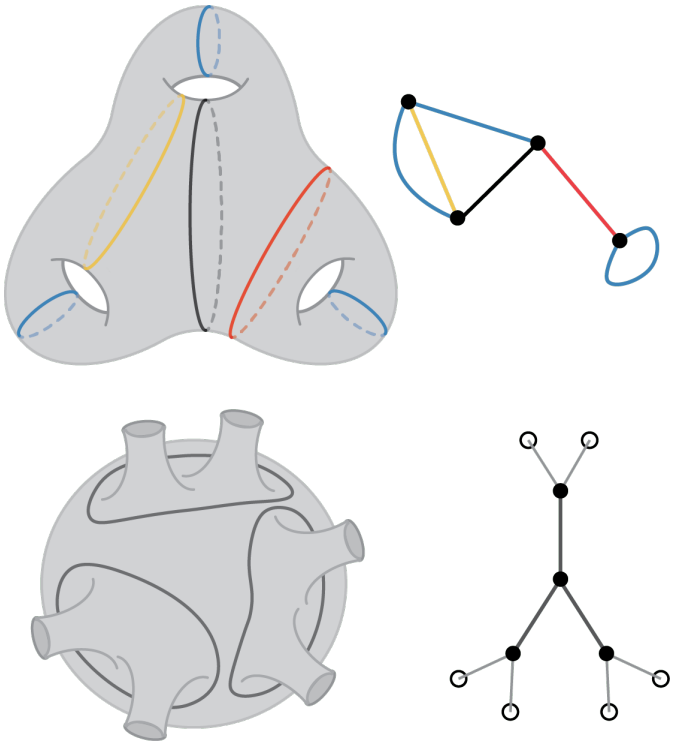

Figure 3.24: Pants decompositions and their intersection graphs.

setting. For this approach to work in our problems, we need to ensure that we can find an orienting curve that crosses our primal graph not too many times. The following lemma is a restatement of Proposition 5.5 in [54]. We provide a sketch of its proof, explaining how to extend it to arbitrary embedded graphs and how to modify it to get orienting arcs in the presence of boundaries.

Lemma 3.7.1. *Let N be a non-orientable surface of genus g without boundary and G be a graph embedded on N. Then there exists an orienting curve of multiplicity at most 2.*

Sketch of the proof. We begin by adding edges to the embedded graph G in order to get cellular faces. We assign a local orientation to each face, that is a cyclic order to the vertices of each face along its boundary. Two adjacent faces are said to have *incoherent* orientations if they induce the same orientation on the edge they have in common. Cutting the surface along all incoherent edges gives us an orientable surface. For any vertex v, there is an even number of edges with incoherent orientations adjacent to v. We pair these edges around each vertex, shift them slightly so that they cross the original graph only transversely and we add segments to join two paired edges. We can modify our local orientation slightly to show that cutting along our new system of edges gives us an orientable surface; see Figure 3.25 which is almost identical to one in the proof of [54, Proposition 5.5].

[2]The picture is redrawn from a figure in the proof of Proposition 5.5 in [54]

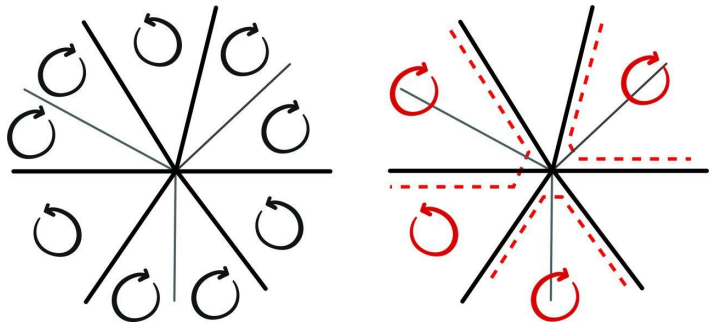

Figure 3.25: The thick lines in the left picture, depict the edges adjacent to a vertex that inherit the same orientation from different faces. The dashed red lines in the right picture are shortened and shifted copies of these edges joined according to the pairing. The arrows indicate the refinement of the local orientation.[2]

This collection of edges forms a disjoint union of closed curves (possibly a single curve). These curves can only cross an edge of the graph near the vertices and therefore the whole system of curves crosses each edge of G at most twice. If we have one curve, then this curve is the desired orienting curve. In the case that we have more than one curve, we can find short paths (paths that do not intersect edges that are already crossed by the curves) that join a pair of these curves at each step. By slightly changing the orientations around these paths, we can merge these curves and paths to a single curve. For the complete proof see [54]. □

Remark 3.7.2. *If N is a non-orientable surface with a boundary component, by a little modification of the building process in the proof of Lemma 3.7.1, we can get an orienting arc instead of an orienting cycle. If the surface has boundary, the local orientation around each boundary is similar to that of a vertex. We choose one boundary component. We shift incoherent edges around this boundary and join all these edges by pairs except for one pair of edges. This way we get an arc and a system of closed curves and we can proceed as explained in the proof to join these components and get one orienting arc.*

3.8 Algorithms and genome rearrangements

In this section, we introduce two computational problems in biology, more accurately in genome rearrangements, and a mathematical model to study them. For more background, we refer the reader to the books [36] and [25]. After introducing the notions and the mathematical model, we explain how this problem can find a topological interpretation and relate to the topics of interest in this thesis. In particular, our results in Chapters 5 and 7 take advantage of this connection.

DNA is a double-stranded molecule consisting of long sequences of nucleotides. Genes

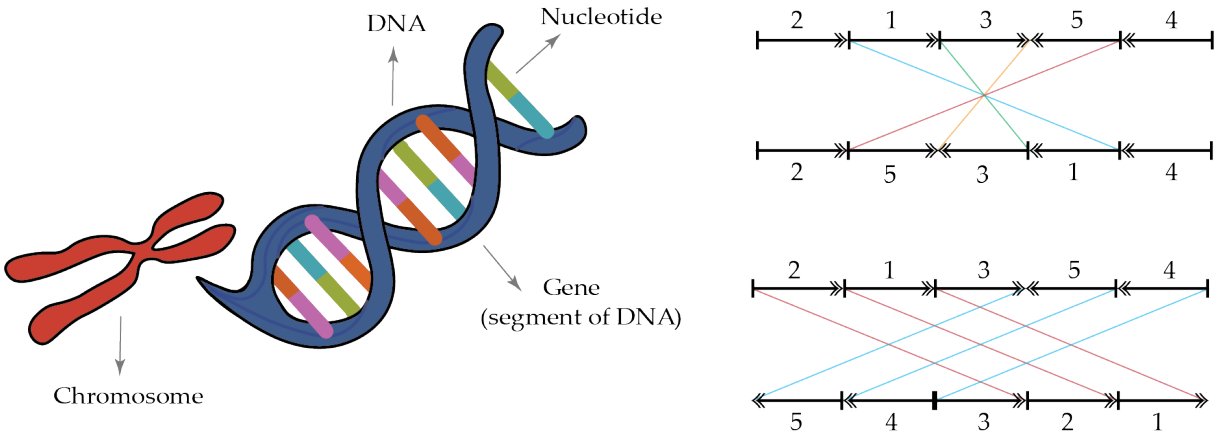

Figure 3.26: Left: the structure of DNA, right: a reversal (top) and a block interchange (bottom) between two gene sequences.

are specific segments of DNA that carry the information required to construct proteins within a cell. Molecular evolution in DNA is responsible for the diversity observed in living organisms. This evolution can occur at the level of individual nucleotides, involving *substitutions* or *insertions*, as well as at a larger scale involving gene sequences and their rearrangements such as *reversals*, *transpositions*, and *block interchange*. The focus of interest in this section lies on the latter type of mutations, referred to as *genome rearrangements*. Detecting efficient rearrangement scenarios between two genomics is the goal of genome rearrangement problems.

These problems can be formulated in combinatorial optimization by representing the relative positions of genes in different genomes as permutations and looking at the mutations as operations to sort these permutations. In general, the problems in genome rearrangements are formulated as follows: given two genomes and some evolutionary events, what is the shortest set of (minimum number of) events that transforms one genome to the other? the length of an optimal solution is called the *distance* between the two genomes.

The two instances of these sorting problems that we are interested in are: 1) computing the signed reversal distance in which the genomes are given as signed permutations and the events are signed reversals, see the top right picture in Figure 3.26, 2) computing block interchange distance in which the genomes are given as permutations and the events are exchanging placements of two sub-permutations, see bottom right picture in Figure 3.26.

In Sections 3.8.1 and 3.8.3, we describe the signed reversal distance and the block interchange distance and introduce an efficient algorithm to compute each of them. After each section, we give an alternative topological formulation of the sorting problem which we believe provides deeper insights into these combinatorial sorting problems. The topological relation was first noticed by Huang and Reidys in [49], in which they construct a bijection between permutations and a particular equivalence classes of fatgraphs.

Although our approach is rather similar, it is independent of their formulation.

3.8.1 Signed reversal distance

One of the most biologically relevant distances in the study of genome rearrangements is the signed reversal distance which is also one of the few that can be calculated efficiently. In this problem, a one chromosome genome is encoded by a *signed permutation* $\pi = (\pi_1, \ldots, \pi_n)$ (i.e., permutations of integers in which each element has a sign). We denote an element i that is negative by $\overline{i}$. For an element $\overline{\pi}_j$, the overline negates the sign of π_j. For example $\theta = (\overline{3}, 2, \overline{1})$ is a signed permutation on $\{1, 2, 3\}$ in which 1 and 3 has negative sign and 2 is positive. An *interval* (π_i, π_j) is the sub-permutation consisting of the elements between π_i and π_j. For example the interval $(6, 2)$ in the permutation $(\overline{1}, 6, \overline{5}, 3, 2, \overline{4})$ is the sub-permutation $(6, \overline{5}, 3, 2)$. The *reversal* on the interval (π_i, π_j) acts on π by reversing the order of the elements $\pi_i, \ldots, \pi_j$ as well as their signs, it maps

$$(\pi_1, \pi_2, \ldots, \pi_{i-1}, \underline{\pi_i, \pi_{i+1} \ldots, \pi_{j-1}, \pi_j}, \pi_{j+1} \ldots \pi_n),$$

to

$$(\pi_1, \pi_2, \ldots, \pi_{i-1}, \underline{\overline{\pi}_j, \overline{\pi}_{j-1} \ldots, \overline{\pi}_{i+1}, \overline{\pi}_i}, \pi_{j+1} \ldots \pi_n).$$

The *reversal distance* between signed permutations π and π', denoted by $d(\pi, \pi')$ is the minimum number of reversals needed to transform π to π'. The problem of computing the signed reversal distance between two permutations π and π' is equivalent to computing the signed reversal distance between $\pi\pi'^{-1}$ (π^{-1} is the inverse of π in the group of signed permutations) and the identity permutation id (every element is positive in the identity permutation). In the following, we sometimes refer to the problem of transforming π to id as the problem of *sorting* π by reversals.

The celebrated algorithm of Hannenhalli and Pevzner [44] (see also [6, 7]) computes in polynomial time the reversal distance between two signed permutations. Below, we briefly describe this algorithm.

The Hannenhalli-Pevzner algorithm

Hannenhalli and Pevzner investigated the problem of sorting permutation with reversals by first extending a standard permutation π on $\{1, \ldots, n\}$ by adding a 0 and $n + 1$ to the beginning and the end of the permutation; we call such a permutation an *extended* permutation on $\{0, \ldots, n + 1\}$. For example, the extended permutation of our example θ above is the permutation $\theta' = (0, \overline{3}, 2, \overline{1}, 4)$. In this section, we recall their results for these permutations; all the results are stated without proofs in this section.

In sorting a permutation π by reversals, an obstacle is to deal with *breakpoints* in π; the places in the permutation that consecutive elements are not consecutive numbers. For

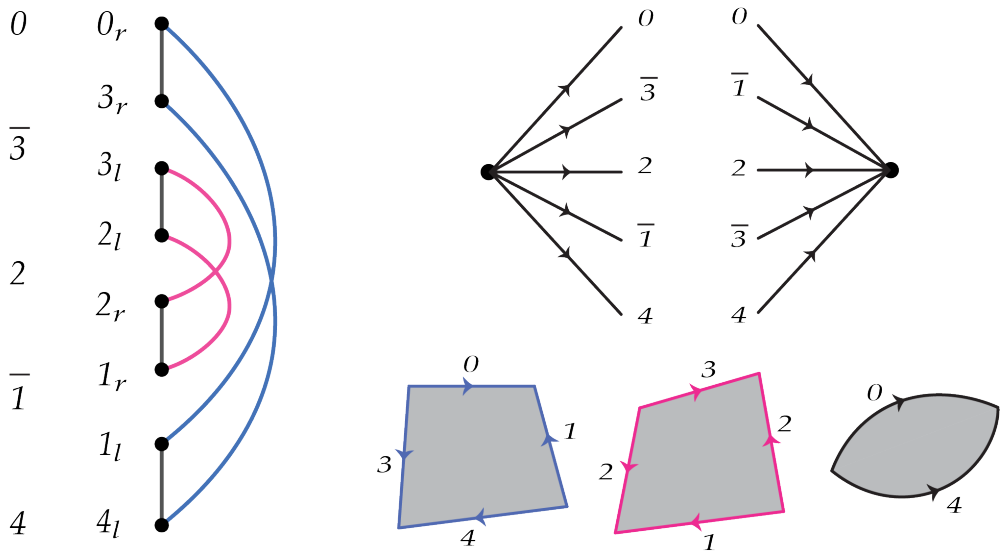

Figure 3.27: Left: the breakpoint graph associated to the signed permutation $\theta' = (0, \overline{3}, 2, \overline{1}, 4)$ and its cycle decomposition, the gray edges are the reality edges and the colored edges are the desire edges, right: the corresponding two-vertex embedding scheme and its faces. We can see the correspondence between the cycles in the breakpoint graph and the faces of the embedding scheme (except for the face $(0, 4)$). If we considered θ' as a cyclic permutation, $(0, 4)$ would also correspond to a cycle.

example in θ', every two consecutive elements (such as 0 and $\overline{3}$) introduce a breakpoint. We can see that the identity permutation has no breakpoint, and therefore sorting a permutation by reversals corresponds to eliminating all of its breakpoints. This suggests that the reversal distance can be estimated by the number of breakpoints in the permutation. However this estimation is not very accurate.

Here we describe another parameter that captures the necessary data from the breakpoints as well as the combinatorial dependencies between them and estimates the reversal distance with much greater accuracy: the number of cycles in the *breakpoint graph* of the permutation.

Breakpoint graph. The *breakpoint graph* of an extended signed permutation π on $\{0, \ldots, n+1\}$ is the graph G_π that has two vertices i_l and i_r associated to each element i for $1 \leq i \leq n$ (l and r are chosen to refer to left and right of an element i, respectively), plus two vertices denoted 0_r and $(n+1)_l$ associated to the added elements 0 and $n+1$. The edges of G_π are comprised of:

- The desire edges: $(i_r, i+1_l)$ for $0 \leq i \leq n$.
- The reality edges for $0 \leq i \leq n$:
  (π_{i_r}, π_{i+1_l}) if π_i and π_{i+1} are positive,

(π_{i_l}, π_{i+1_l}) if π_i is negative and π_{i+1} is positive,
(π_{i_r}, π_{i+1_r}) if π_i is positive and π_{i+1} is negative,
and (π_{i_l}, π_{i+1_r}) if π_i and π_{i+1} are negative.

In this graph, every vertex has degree exactly 2 and therefore, the edges of the graph form a set of cycles (see Figure 3.27 for the breakpoint graph of the permutation θ'). Let us denote by $c(\pi)$ the number of cycles in G_π.

Among the permutations on $\{0, \ldots, n+1\}$ elements, the identity permutation is the only one with $n+1$ cycles and no breakpoint. We can check that a reversal changes the number of cycles by at most 1 (see [7, Proposition 3]). Therefore, in sorting a permutation π, at least $n+1-c(\pi)$ reversals are needed, i.e., $d(\pi, id) \geq n+1-c(\pi)$. A reversal is making progress toward sorting a permutation if it increases the parameter $c(\pi)$; we call such a reversal *optimal*.

Given a signed permutation π, we call a pair of consecutive integers i and $i+1$ *reversible* if they have opposite signs in π (this is called an *oriented pair* in [7]). For a given reversible pair there exists a reversal σ such that after applying them on π, i and $i+1$ become consecutive with matching signs, either $i, i+1$ or $\overline{i+1}, \overline{i}$. For example in the permutation $(1, 3, 6, \overline{4}, \overline{2}, \overline{5}, 7)$, 3 and $\overline{4}$ are a reversible pair, and applying a reversal on $(6, \overline{4})$ transforms the permutation to $(1, 3, 4, \overline{6}, \overline{2}, \overline{5}, 7)$ and makes 3 and 4 consecutive and positive. Such reversals seem like a good way to make progress in sorting a permutation and actually the starting idea of the Hannenhalli-Pevzner algorithm is to identify these reversals (actually we need to check an extra condition for these reversals as we explain in what follows.)

The approach of finding reversible pairs in the permutation fails as soon as we obtain a signed permutation in which all elements have the same sign. This leads to the definition of *blocks* (*hurdles* in [44]) which introduces a second parameter that together with the number of cycles in the breakpoint graph determines the signed reversal distance. Here, we introduce the concept of blocks that is similar to the notion of unoriented components in [7].

Blocks. A *positive block* in a signed permutation is an interval $I = (\pi_i, \ldots, \pi_j)$ where all the elements are positive, $\pi_i < \pi_j$, and the elements that are contained in I are all the integers in $[\pi_i, \pi_j]$. A *negative block* in a signed permutation is an interval $I = (\overline{\pi}_i, \ldots \overline{\pi}_j)$ where all the elements are negative, $\pi_i > \pi_j$, and the elements that are contained in I are all the integers in $[\pi_j, \pi_i]$. A block is *non-trivial* if it is not already sorted, i.e., it is not equal to $(\pi_i, \pi_i + 1, \ldots \pi_j - 1, \pi_j)$ or to $(\overline{\pi}_i, \overline{\pi_i - 1}, \ldots, \overline{\pi_j + 1}, \overline{\pi}_j)$. In both cases, we call π_i and π_j the *frames* of the block. For example in the signed permutation $(0, 2, 1, 3, \overline{7}, \overline{5}, \overline{6}, \overline{4}, 8)$, $(0, 3)$ bounds a positive block and $(\overline{7}, \overline{4})$ bounds a negative block.

As there are no reversible pairs in a block, it costs us extra reversals to deal with them. It turns out that applying a reversal on an interval in a block destroys the block

without changing the number of cycles in the breakpoint graph (this is called *cutting hurdles* in [7]). To optimize the number of reversals that we need to destroy the blocks, a *merging* process is defined in which two blocks are dealt with only one reversal. The optimal number of reversals, $t(\pi)$, needed to demolish all the blocks in a permutation π, is computed using a single PQ-tree. The full description of these processes will not be needed and is rather out of the scope of this thesis, therefore we refer the reader to [7] for more details.

As blocks add to the number of reversals needed to sort a permutation, it is a natural idea to avoid creating one during sorting. When we apply a reversal on a reversible pair, there is a chance that it creates a new block in the permutation. For example applying a reversal on the reversible pair 5 and $\overline{4}$ (namely, applying a reversal on $(5, \overline{1})$), in the permutation $(0, 2, 5, \overline{3}, \overline{1}, \overline{4}, 6)$ which contains no block, transforms it to $(0, 2, 1, 3, \overline{5}, \overline{4}, 6)$ in which $(0, 3)$ is a positive block. Given a reversible pair $(i, i+1)$ in the signed permutation π, let σ be a reversal that makes i and $i+1$ consecutive. The *score* of $(i, i+1)$ (or of σ) is the number of reversible pairs in $\pi \cdot \sigma$. A reversal that has the maximal score among all reversible pairs is *safe* in the sense that it does not create new blocks in the permutation, see [6, Theorem 10] (we also provide a proof for this, but for cyclic permutations, see Lemma 3.8.5). For example the reversal applied on the reversible pair $0, \overline{1}$ in the example above, i.e., the reversal on the interval $(2, \overline{1})$, has the maximal score 4 and is safe while the reversal on $\overline{4}$ and 5 has score 2. The following lemma states that a safe reversal is optimal.

Lemma 3.8.1 ([6, Proposition 1])**.** *If $i, i+1$ are a reversible pair with maximal score in π and σ is a reversal that makes them consecutive and with same sign in $\pi \cdot \sigma$, then $d(\pi \cdot \sigma, id) = d(\pi, id) - 1$.*

Now we are ready to sketch the algorithm of Hannenhalli and Pevzner, although we do not describe the step that deals with the blocks.

The sketch of the Hannenhalli-Pevzner algorithm:

- **Step 1: If there exists a reversible pair.** Apply a reversal on a reversible pair with maximal score.
- **Step 2: If there are no reversible pairs.** If the permutation is the identity, we are done. Otherwise, there exists a block in the permutation. Demolish the blocks using a cutting or a merging process (the details of this step depend on the PQ-tree). Then recurse.

The reversals applied on the permutation π on $\{0, \ldots, n+1\}$ in Step 1 of the algorithm, increase the number of cycles and reduces the distance to the identity permutation. As

we explained above, at most $n + 1 - c(\pi)$ reversals are applied in Step 1 to reach the maximal number of cycles in a permutation (which belongs to the identity permutation). As we mentioned, the optimal number of reversal that we need in Step 2 of the algorithm is denoted by $t(\pi)$. This gives us the following theorem that we state here without proof.

Theorem 3.8.2 ([7, Theorem 2])**.** *Let π be an extended signed permutation on the set $\{0, \ldots, n+1\}$. Then $d(\pi, id) = n + 1 - c(\pi) + t(\pi)$.*

Remark 3.8.3. *Note that for a permutation without any blocks, Step 1 of the algorithm is sufficient to sort the permutation. Therefore, for an extended permutation π that has no block, $d(\pi) = n + 1 - c(\pi)$ (see [7, Corollary 1]). One of our results in Chapter 7, extensively uses the first step of this algorithm.*

3.8.2 Topology of the signed reversal distance and relation to cross-cap drawings

As we mentioned above, the problem of sorting signed permutations by reversals is one of the few that can be calculated efficiently among the genome rearrangement problems. In this section, we investigate the topology behind the signed reversal distance which we believe can explain this efficiency.

The algorithm of Hannenhalli-Pevzner, primarily deals with sorting standard signed permutations that are extended by adding 0 and $n + 1$ at the beginning and the end of the permutation, while in this section, we consider the more general case of sorting cyclic signed permutations. A cyclic permutation describes a loopless 2-vertex embedding scheme: consider a two-vertex embedding scheme with edges labelled by the elements in π such that the cyclic permutation of edges around one vertex is π and around the other one is $\overline{id}$. The signature of edges in the embedding scheme is determined by the sign of the elements in π, i.e., if an element i has sign $+$ (resp. $-$) in the permutation π, the edge i in the corresponding embedding scheme has signature $+1$ (resp. -1). In the following, we use the same letter π to denote the two-vertex embedding scheme associated to the permutation π and in turn we use the same notations that we used for embedding schemes, for π, notations such as $\tilde{g}(\pi)$ and $eg(\pi)$.

Our sorting problem now becomes computing the minimum number of reversals, $d(\pi, id)$, needed to transform a cyclic permutation π to id. For a signed permutation π notions of breakpoint, reversible pair, blocks and optimal and safe reversals can all be defined similarly to the case of standard permutations. The definition of the breakpoint graph for a cyclic permutation $\pi = (\pi_1, \ldots, \pi_n)$ is quite similar to the extended standard case except that we do not have 0_r and $(n+1)_l$ and i is considered modulo n. It is easy

to check that the cycle decomposition of this graph for a cyclic permutation π coincides with the faces in the embedding scheme corresponding to π, see Figure 3.27.

Let X be a sorting scenario that transforms the cyclic permutation π to the identity permutation by $d(\pi, id)$ reversals. The scenario X for sorting π can be employed to obtain a cross-cap drawing for π (probably with more cross-caps than $\tilde{g}(\pi)$) as follows: trace the edges in π under the actions of reversals, such that if a reversal is applied on an interval (i, j), all the edges in G that belong to this interval go through a cross-cap, see Figure 3.28. If a and b are two reversals in X such that a happens before b, orienting the edges from the vertex with cyclic permutation π to the other vertex, no edge in π that enters the cross-caps associated to a and b, enters b before a. We call a cross-cap drawing that admits such an order on the cross-caps a *monotone* cross-cap drawing.

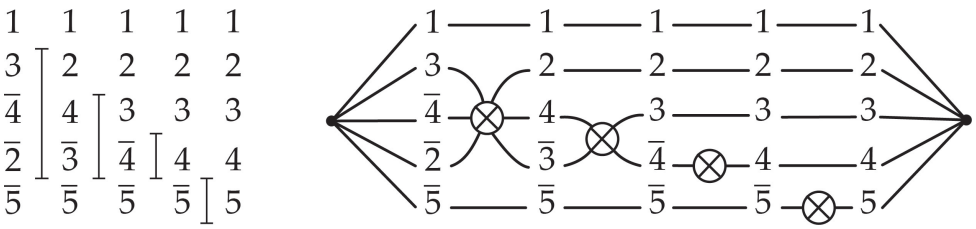

Figure 3.28: From sorting permutations with reversals to monotone cross-cap drawings.

The Bafna-Pevzner inequality from Euler's formula

Let π be a signed cyclic permutation, which as explained above, we think of as a two-vertex embedding scheme, which requires $\tilde{g}(\pi)$ cross-caps to be drawn. We can compute its number of faces, which we denote by $f(\pi)$ and the number of elements in the permutation corresponds to the number of edges in the scheme, which we denote by $e(\pi)$. Then Euler's formula reads $2-eg(\pi) = 2-e(\pi)+f(\pi)$ which simplifies to $eg(\pi) = e(\pi)-f(\pi)$. Since the reversals (or cross-caps) in a sorting scenario are monotone and by Lemma 3.3.2, we thus have $d(\pi, id) \geq \tilde{g}(\pi) \geq e(\pi) - f(\pi)$ (the first inequality from left is due to monontonicity of a cross-cap drawing obtained from a sorting scenario and the other is implied from Lemma 3.3.2).

A very similar inequality was first discovered by Bafna and Pevzner [5, Theorem 2] without reference to embeddings. Figure 3.29 shows an example where the inequalities are strict: the embedding scheme has Euler genus two, non-orientable genus three and one can show that the signed permutation requires four reversals to be sorted. However, as pictured on the right, it does admit a cross-cap drawing with three cross-caps in which each edge enters each cross-cap at most once. Necessarily, in that example, the cross-caps can not be interpreted as reversals: this is apparent here as they do not occur in a monotone order.

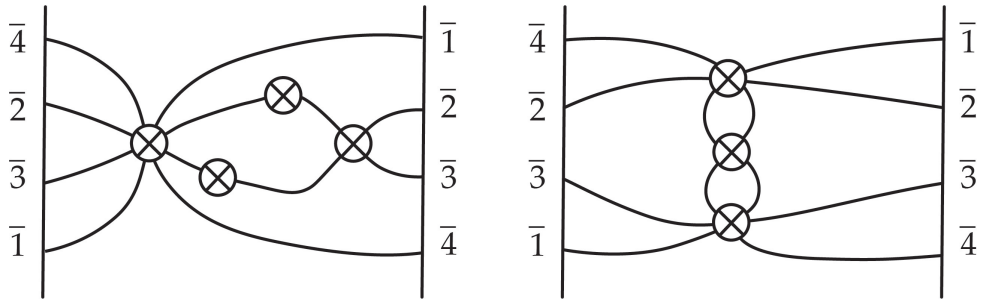

Figure 3.29: The embedding scheme depicted in this picture has non-orientable genus 3 but requires four reversals (left). However, it admits a cross-cap drawing with three cross-caps such that each edge enters each cross-cap at most once (right).

Two edges in a 2-vertex embedding scheme or a permutation π are *parallel*[3] if either they are both positive and appear consecutively in increasing order in π (e.g., $(i, i+1)$) or they are both negative and they appear in the reverse order (e.g., $(\overline{i+1}, \overline{i})$). Under this definition, given a reversible pair $(i, i+1)$, there are two reversals σ and σ' that make i and $i+1$ parallel. For example, in the signed cyclic permutation $(1, \overline{3}, 2, 4)$, two reversals on $(\overline{3}, 2)$ and $(4, 1)$ make 3 and 4 parallel. These two reversals are equivalent in the sense that the two permutations (embedding schemes) that they yield are flipped versions of each other, i.e., $\pi \cdot \sigma = \overline{\pi \cdot \sigma'}$ (recall that $\overline{\pi}$ is a permutation obtained by reversing the elements in the permutation π) and the signatures of all the edges are reversed. The following lemma is equivalent to Lemma 3.8.1 from the point of view of graph embeddings. We provide a proof for this version.

Lemma 3.8.4. *If $i, i+1$ are a reversible pair in π and σ is a reversal that turns them into parallel curves in $\pi \cdot \sigma$, then $eg(\pi \cdot \sigma) = eg(\pi) - 1$.*

Proof. Since $e(\pi) = e(\pi \cdot \sigma)$, we need to show that $f(\pi \cdot \sigma) = f(\pi) + 1$. The edges i and $i+1$ appear in some face a in π. We claim that the faces of $\pi \cdot \sigma$ are the same as the faces of π except that a is subdivided into the bigon $(i, i+1)$ and another face which contains the edges of a minus i and $i+1$. Indeed, every other face is not disrupted by the reversal. This finishes the proof. □

As mentioned in Remark 3.8.3, in the case where π is a standard permutation that does not contain a block, applying safe reversals sorts the permutation (it is ensured that no block emerges during the algorithm). Here, we generalize this result to the case where the permutation is cyclic. The following lemma provides an almost identical result to that of [6, Theorem 10] but for embedding schemes.

[3]We show in Chapter 7 that this definition coincides with the the curves being homotopic in a two-vertex embedding scheme.

Lemma 3.8.5. *Let π be a non-orientable signed permutation without non-trivial blocks, $(i, i+1)$ be a reversible pair of maximal score, and σ be a reversal that makes i and $i+1$ parallel. If $\pi \cdot \sigma$ is not the identity, it is non-orientable and has no non-trivial blocks.*

We postpone the proof of this lemma to the following section. Here, we use this lemma to prove that in the absence of blocks, $eg(\pi) = d(\pi, id)$. This theorem is used extensively to obtain our results in Chapter 7.

Theorem 3.8.6. *If a signed permutation π is non-orientable and has no non-trivial blocks then $d(\pi, id) = eg(\pi)$, and the Hannenhalli-Pevzner algorithm gives a sequence of reversals of this optimal length.*

Proof of Theorem 3.8.6. By the previous lemma, if π is non-orientable has no non-trivial block and σ is a reversal of maximum score, then $\pi \cdot \sigma$ is also non-orientable and also has no non-trivial block. By induction, $d(\pi \cdot \sigma, id) = e(\pi \cdot \sigma) - f(\pi \cdot \sigma) = eg(\pi \cdot \sigma)$. Therefore, by Lemma 3.8.4, $eg(\pi) \leq d(\pi, id)$, and $d(\pi, id) \leq d(\pi \cdot \sigma, id) + 1 = eg(\pi \cdot \sigma) + 1 = e(\pi) - f(\pi) = eg(\pi)$. □

Proof of Lemma 3.8.5

In this section, we provide a proof for Lemma 3.8.5. Two intervals (i, j) and (k, l) are called *interleaving* in a cyclic permutation, if we either have $\pi^{-1}(i) < \pi^{-1}(k) < \pi^{-1}(i+1)$ or $\pi^{-1}(i) < \pi^{-1}(l) < \pi^{-1}(i+1)$. The following is similar to the definition of overlapping graph and the proof of Theorem 1 in [6]. We also take advantage of this definition and a technique very similar to that of the proof of Lemma 3.8.5 in our main result in Chapter 5.

Interleaving graph. Given a permutation π on $\{1, \ldots, n\}$, let π^* be the permutation of size $2n$ on elements $\{i^l, i^r \text{ for } 1 \leq i \leq n\}$ in which $\pi^*_{2k-1} = \pi^l_i$ and $\pi^*_{2k} = \pi^r_i$ when π_i is positive and $\pi^*_{2k-1} = \pi^r_i$ and $\pi^*_{2k} = \pi^l_i$ when π_i is negative. We build the *interleaving* graph I_π as follows:

- The graph has n vertices labelled by $(i, i+1)$ for i modulo n. A vertex $(i, i+1)$ is called *reversible* if $(i, i+1)$ is a reversible pair, otherwise it is called *non-reversible*. We denote a reversible pair by a white vertex and a non-reversible pair by a black one.

- Two vertices $(i, i+1)$ and $(j, j+1)$ are connected if the intervals $(i^r, i+1^l)$ and $(j^r, j+1^l)$ are interleaving in π^*.

See Figure 3.30 for an example.

A connected component in I_π is *non-trivial* if it has more than one vertex and it is called *orientable* if it only contains non-reversible vertices.

Remark 3.8.7. *If $(i, i+1)$ are parallel in π then the vertex $(i, i+1)$ is an isolated vertex. A non-trivial block and the vertices in I_π associated to the pair of edges that belong to the block correspond to a non-trivial orientable connected component in I_π. Note that the reverse is not true and an orientable connected component does not always correspond to a non-trivial block in the permutation.*

Lemma 3.8.8. *Let π be a signed permutation for which I_π has a non-trivial orientable connected component U. Then either U corresponds to a non-trivial block or π is orientable.*

Proof. We say that an element i belongs to U if $(i, i+1) \in U$ or $(i-1, i) \in U$. By orientability, all the elements belonging to U have the same signature, let us first assume that it is positive. Let us furthermore assume, for the sake of contradiction, that π is not orientable and that U does not correspond to a block. Since π is not orientable, not every element has positive signature, so there is at least one element that does not belong to U. Let a be an element belonging to U such that $a-1$ does not belong to U. Without loss of generality, we can fix an origin to the signed permutation π at $a-1$. Now let b be an element that belongs to U and such that for any element j that belongs to U, a, j and b appear in this cyclic order in π. We claim that either $a-2=b$, or $a-2$ does not appear between a and b in π. Indeed, if $a-2$ appears between a and b, then $(a-2^r, a-1^l)$ would interleave with one pair on a path between $(a, a+1)$ and $(b, b\pm 1)$ in U, and thus $a-1$ would belong to U. Therefore, either $a-2=b$ or $a-2$ does not belong to U. Inductively, none of the elements that are not in $[a, b]$ lie between $[a, b]$ in π. Similarly, all the elements within $[a, b]$ lie between a and b in π, as otherwise the smallest one that does not, call it k, does not belong to U, yet $(k-1^r, k^l)$ interleaves with a pair on a path between $(a, a+1)$ and $(b, b\pm 1)$ in U, contradicting the fact that k is not in U. We conclude that a and b are the frames of a block, which is non-trivial since there is at least one pair in U. This is a contradiction. The case where all the signatures in U are negative follows from the fact that flipping π does not change its interleaving graph nor the orientability of its components. □

When we apply a reversal on a reversible pair $(i, i+1)$, the effect on I_π is to complement the subgraph induced by the vertex $(i, i+1)$ and its neighbors in I_π. Also if a vertex in this subgraph was reversible, it gets non-reversible and vice versa. Indeed, let $(j, j+1)$ be a reversible vertex connected to $(i, i+1)$. This means that the intervals $(j^r, j+1^l)$ and $(i^r, i+1^l)$ are interleaving. Without loss of generality let us assume that j^r is the element that belongs to the interval $(i^r, i+1^l)$ and it is positive. Reversing the elements between $(i, \pi_{\pi^{-1}(i+1)-1})$ makes i and $i+1$ parallel and isolates $(i, i+1)$. Also it makes j negative and therefore the pair $(j, j+1)$ is not reversible anymore. Similarly it can be seen that if $(j, j+1)$ and $(k, k+1)$ are two neighbors of $(i, i+1)$ in I_π and the intervals $(j^r, j+1^l)$ and $(k^r, k+1^l)$ interleave, after applying the reversal they stop being interleaved and therefore

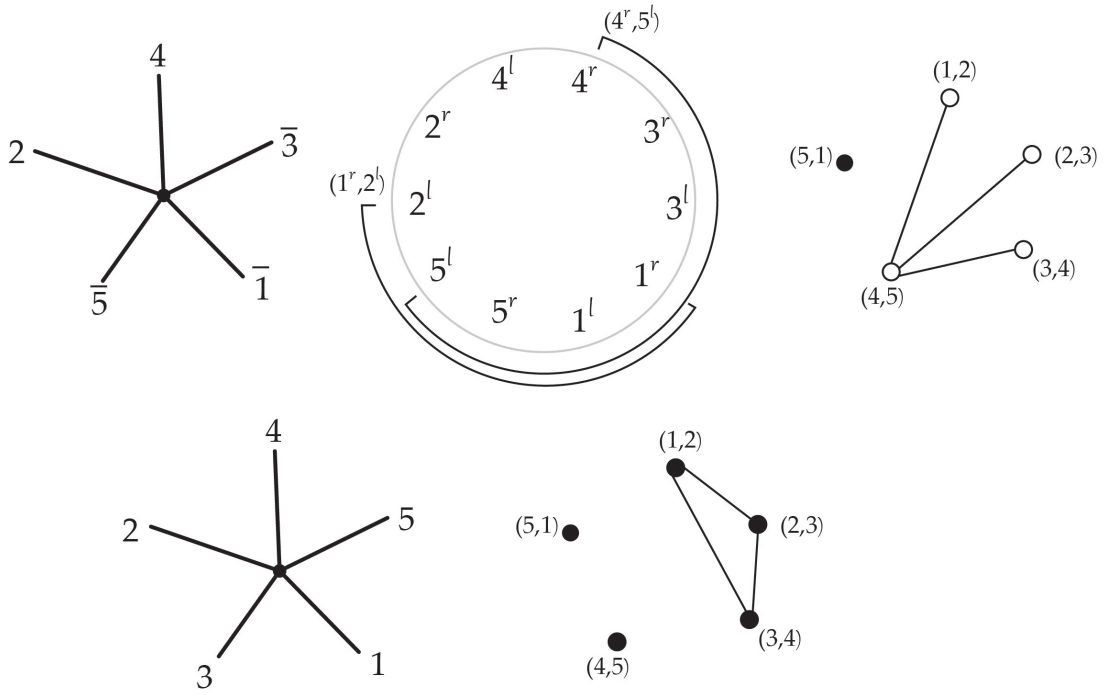

Figure 3.30: Figure depicts a permutation π, its associated doubled permutation π^* and its interleaving graph I_π. At the bottom we can see the effect of reversing elements $1, 3, 5$ on the interleaving graph; this reversal makes 4 and 5 parallel.

vertices $(j, j+1)$ and $(k, k+1)$ are not connected anymore in the interleaving graph. This explains the complementing of the induced subgraph. Figure 3.30 depicts the effect of applying the reversal on the pair $(4, 5)$.

Proof of Lemma 3.8.5. To prove the lemma, we first show that the number of non-trivial orientable connected components in $I_{\pi\cdot\sigma}$ cannot be more than I_π. Let us assume that this is the case and that by applying σ we create such a component C. In this case, we claim that the score of any pair $(j, j+1)$ for which the corresponding vertex is in C, is higher than the score of the pair $(i, i+1)$.

Denote by $\#^+(i, i+1)$ (resp. $\#^-(i, i+1)$) the number of reversible (resp. non-reversible) pairs of edges adjacent to $(i, i+1)$ in I_π. If π has k reversible pairs of edges, then the score$(i, i+1) = k - \#^+(i, i+1) + \#^-(i, i+1)$.
Before applying the reversal, every non-reversible vertex that is connected to $(i, i+1)$ has to be connected to $(j, j+1)$. This is because otherwise after applying the reversal this vertex will be a reversible vertex connected to $(j, j+1)$ and therefore belonging to C which is not possible. This implies that $\#^-(i, i+1) \leq \#^-(j, j+1)$.
Before applying the reversal, every reversible vertex $(t, t+1)$ that is connected to $(j, j+1)$ has to be connected to $(i, i+1)$. This is because if $(t, t+1)$ is not connected to $(i, i+1)$, after applying the reversal, this vertex remains connected to $(j, j+1)$ without changing

its reversibility. This means that $(t, t+1)$ is a reversible pair that belongs to C which is not possible. This implies that $\#^+(j, j+1) \leq \#^+(i, i+1)$. The equality does not happen since the component C has more than one vertex and a vertex in C to which $(j, j+1)$ is connected after the reversal, is a reversible vertex that used to be connected to $(i, i+1)$ but not to $(j, j+1)$. Therefore $\#^+(j, j+1) < \#^+(i, i+1)$.

We have that $\text{score}(i, i+1) = k - \#^+(i, i+1) + \#^-(i, i+1) < k - \#^+(j, j+1) + \#^-(j, j+1) = \text{score}(j, j+1)$ which contradicts our assumption. This finishes the proof of the claim.

Now, the assumptions of Lemma 3.8.5 imply that there are no non-trivial orientable connected components in I_π. Thus there are also none in $I_{\pi\cdot\sigma}$. A non-trivial block or the entire scheme being orientable but not the identity would induce such a non-trivial orientable connected component. This concludes the proof. □

3.8.3 Block interchange distance

Another distance that can be calculated efficiently among the genome rearrangement problems is the block interchange distance. In this problem, a one chromosome genome is encoded by an unsigned permutation $\pi = (\pi_1, \ldots, \pi_n)$. As we defined above, an *interval* (π_i, π_j) is the sub-permutation consisting of the elements between π_i and π_j. A block interchange acts on two disjoint intervals (not necessarily contiguous) (π_i, π_j) and (π_k, π_l) by replacing their elements: it maps

$$(\pi_1, \ldots, \pi_{i-1}, \underline{\pi_i, \ldots, \pi_j}, \pi_{j+1} \ldots, \pi_{k-1}, \underline{\pi_k, \ldots, \pi_l}, \pi_{l+1}, \ldots \pi_n),$$

to

$$(\pi_1, \ldots, \pi_{i-1}, \underline{\pi_k, \ldots, \pi_l}, \pi_{j+1} \ldots, \pi_{k-1}, \underline{\pi_i, \ldots, \pi_j}, \pi_{l+1}, \ldots \pi_n).$$

The *block interchange distance* between permutations π and π', denoted by $d_b(\pi, \pi')$ is the minimum number of block interchanges needed to transform π to π'. The problem of computing the block interchange distance between two permutations π and π' is equivalent to computing the block interchange distance between $\pi\pi'^{-1}$ and the identity permutation id. In the following, we sometimes refer to the problem of transforming π to id as the problem of *sorting* π by block interchange.

Although the block interchange distance and its topological interpretation are not used in the thesis but they are worth presenting as they provide a simple orientable counterpart to Sections 3.8.1 and 3.8.2.

The Christie algorithm

Similar to the Hannenhalli-Pevzner algorithm for computing the signed reversal distance, Christie [16] also deals with extended permutations on $\{0, \ldots, n+1\}$. In sorting a permu-

tation by block interchange, an obstacle is to deal with breakpoints in the permutation, as in the case of sorting by reversals. Although in this case, breakpoints provide sufficient data to completely describe the block interchange distance.

The same notion of breakpoint graph that was introduced in Section 3.8.1, is introduced to capture the necessary data in the permutation and study the effect of a block interchange in the number of breakpoints[4] (every edge is assumed to have sign +). Here again, we enumerate the number of cycles in this graph and a winning strategy is to apply block interchanges that increase the number of cycles to $n+1$ that is the number of cycles in the identity permutation id on $\{0, \ldots, n+1\}$.

Christie proves the following lemma.

Lemma 3.8.9 ([16, Lemma 1])**.** *If π is not the identity permutation, there always exist a block interchange that removes at least two breakpoints from π.*

The lemma is proved by introducing a *minimal* block interchange. Consider a permutation π that is not the identity. There exist two elements $x < y$ that appear in the wrong order in π, i.e., $\pi = (1, \ldots, y, \ldots, x, \ldots, n)$. Let x be the smallest such value and y be the largest one to the left of x. By this choice, we know that $x-1$ must appear before y in π and $y+1$ must appear to the right of x. Let us denote by z the element in π that appears after x, i.e., $z = \pi_{\pi^{-1}(x-1)+1}$ and let w be the element in π that appears before $y+1$, i.e., $w = \pi_{\pi^{-1}(y+1)-1}$. We introduce the minimal block interchange in π to be the one that replaces the interval (z, y) with (x, w) and vice versa. This block interchange, maps $\pi = (1, \ldots, x-1, \underline{z, \ldots, y}, \ldots, \underline{x, \ldots, w}, y+1, \ldots, n)$ to $\pi = (1, \ldots, x-1, \underline{x, \ldots, w}, \ldots, \underline{z, \ldots, y}, y+1, \ldots, n)$.

A minimal block interchange increases the number of cycles in the breakpoint graph by 2 (see [16, Lemma 2]).

Although a block interchange might decrease the number of breakpoints by 4, no block interchange can increase the number of cycles in the breakpoint graph by more than 2 (see [16, Lemma 3]). Using this observation, Christie proves that applying minimal block interchanges repeatedly gives an optimal bound.

Theorem 3.8.10 ([16, Theorem 4])**.** *Let π be a permutation on $\{0, \ldots, n+1\}$. Then $d_b(\pi, id) = \frac{1}{2}(n+1-c(\pi))$.*

3.8.4 Topology of the block interchange distance and relation to box drawings

As in the case of sorting permutation with reversals, in this section, we investigate the topology behind the block interchange distance. Here again, we consider this sorting

[4] This is called *cycle graph* in [16]).

problem for the case of cyclic permutations. We reduce the problem in the cyclic setting to the setting of extended standard permutations.

We associate to an unsigned cyclic permutation π, an orientable embedding scheme: consider a two-vertex orientable embedding scheme with edges labelled by elements in π such that the cyclic permutation of edges around one vertex is π and around the other one is $\overline{id}$. In the following, we use the same letter π to denote the two-vertex embedding scheme associated to the permutation π and in turn we use the same notations that we used for embedding schemes, for π; for example $g(\pi)$ refers to the orientable genus of the embedding scheme that corresponds to the permutation π.

Similar to the case of reversal distance, the cycle decomposition of the breakpoint graph of a cyclic permutation corresponds to the faces of the corresponding embedding scheme. Let X be a sorting scenario with block interchange that transforms π to the identity permutation by $d_b(\pi, id)$ block interchanges. The scenario X for sorting π can be employed to obtain a box drawing for G as follows: let ρ be a block interchange that replaces the interval (i, j) with (k, l) and vice versa. Trace the edges in π under the actions of block interchange by drawing a box with sides labeled by $abc\overline{a}\overline{b}\overline{c}$ such that the elements in (i, j), (j, k) and (k, l) enter the box from sides a, b and c, respectively, see Figure 3.31. If m and n are two block interchanges in X such that m happens before n, orienting the edges from the vertex with cyclic permutation π to the other vertex, no edge in π that enters both boxes associated to m and n, enters n before m. We call a box drawing that admits such an order on the boxes, a *monotone* box drawing.

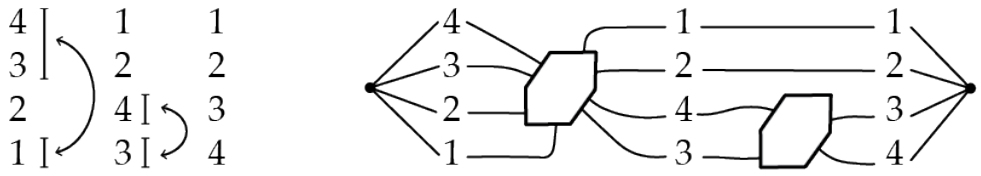

Figure 3.31: From sorting permutations with block interchange to monotone box drawings. Note that this sorting scenario is not optimal if the permutation $(4, 3, 2, 1)$ is viewed as a cyclic permutation (see Figure 3.32 for an optimal scenario), but it is optimal if it is a standard permutation.

The block interchange distance and Euler's formula. Let π be an unsigned permutation, which as explained above, we think of as a two-vertex orientable embedding scheme, which requires $g(\pi)$ boxes to be drawn. By Euler's formula $2-2g(\pi) = 2-eg(\pi) = 2-e(\pi)+f(\pi)$ which simplifies to $g(\pi) = \frac{1}{2}(e(\pi) - f(\pi))$. The monotonicity of a box drawing obtained from a sorting scenario implies that $d_b(\pi, id) \geq g(\pi)$.

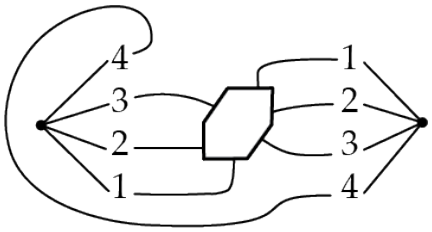

Figure 3.32: A scenario for sorting the cyclic permutation $(4, 3, 2, 1)$. The corresponding embedding scheme has orientable genus 1.

We can adopt Christie's algorithm in the cyclic setting to show that the equality holds in that setting. Let π be a cyclic permutation on $\{1, \ldots, n\}$. Let x and y be the elements appearing in π, after and before n, respectively. Consider the interval (x, y) as an standard permutation. Extend this by adding 0 and n and denote the extended standard permutation by π'. The permutation π' is an extended permutation on $\{0, \ldots, n\}$ and can be sorted by Christie's algorithm using $\frac{1}{2}(n - c(\pi'))$ block interchanges. Now, one can check that $c(\pi') = f(\pi)$ which implies that this permutation can be sorted using $\frac{1}{2}(n - c(\pi')) = \frac{1}{2}(n - f(\pi)) = g(\pi)$ block interchanges. This sorting scenario for π' induces a sorting scenario for π (such that n does not enter any box in a box drawing corresponding to this scenario). This gives us the opposite inequality $d(\pi, id) \leq g(\pi)$. Figure 3.32 depicts such a scenario for sorting the cyclic permutation $(4, 3, 2, 1)$.

Chapter 4

Joint crossing numbers of graphs and Negami's conjecture

Summary. Negami proved that any two graphs G_1 and G_2 embedded on a surface S have joint crossing number $O(g|E(G_1)||E(G_2)|)$. In this chapter, we first provide an example to demonstrate why his proof technique for non-orientable surfaces fails. Then we provide a correction for this proof.
A *k-universal shortest path metric* is a metric on a fixed surface such that each graph embeddable on this surface can be embedded so that each edge is a concatenation of at most k shortest paths. In the second part of this chapter we provide an $O(g)$-universal shortest path metric for non-orientable surfaces of genus $g \geq 3$. This generalizes a similar result for orientable surfaces. The proof involves generalizing an algorithm that provides a short octagonal decomposition of orientable surfaces to the non-orientable setting.

The results in this chapter were obtained with Alfredo Hubard and Arnaud de Mesmay. Both results appear in [A] which has been published in Discrete & Computational Geometry. The first result also appears in a preliminary version that is published in the Proceedings of the 38th Symposium on Computational Geometry [A].

4.1 Introduction

The *joint crossing number* of two graphs G_1 and G_2 embedded on a surface S is the minimum number of crossings between $h(G_1)$ and G_2 over all homeomorphisms $h : S \to S$ with the constraint that edges are only allowed to cross transversely. This crossing number was initially introduced by Negami [59]. He made the following conjecture, which is still open.

Conjecture 1. (Negami's conjecture) *There exists a universal constant C such that for any pair of graphs G_1 and G_2 embedded on a surface S, the joint crossing number is at most $C|E(G_1)||E(G_2)|$.*

This conjecture has been investigated further [3, 47, 66] and variants of this problem have appeared in various works with applications as diverse as finding explicit bounds for graph minors [38] or designing an algorithm for the embeddability of simplicial complexes into $\mathbb{R}^3$ [54].

Despite all these studies, the conjecture has been open for more than 20 years. The best known bound is due to Negami himself who proved that any two graphs G_1 and G_2 embeddable on a surface (orientable or non-orientable) of genus g have joint crossing number $O(g|E(G_1)||E(G_2)|)$. Although the statement is correct, his approach in dealing with non-orientable surfaces suffers from a small flaw. In the first contribution of this chapter, we provide a correction for the non-orientable case using an alternative technique.

A geometric approach toward a better understanding of Negami's conjecture has been devised by Hubard, Kaluža, de Mesmay and Tancer in [50]. They investigate a problem that is an adequate generalization of the celebrated Fàry's theorem for embedding planar graphs [51]. Fàry's theorem states that every planar graph can be embedded on the plane with straight lines. They formalize this generalization by asking the following question.

Question 1. *Given a surface S of genus g, does there exist a Riemannian metric on S such that any simple graph embeddable on S can be embedded so that the edges are shortest paths on S?*

Other generalized instances of Fàry's theorem have been provided. For example, in [68], an instance of this question is answered positively in which the edges of the graphs are restricted to be geodesics instead of shortest paths. Also, Question 1 has a positive answer if instead of a universal metric we merely asked for a metric for each graph, see [71]. But we are interested in both universality of the metric and the restriction on edges being shortest paths, since a positive answer to this instance of the problem implies Negami's conjecture. Indeed, if there existed a universal Riemannian metric on each surface so that all the graphs embeddable on the surface could be embedded with shortest paths for this metric, then for such an embedding for any two graphs, pairs of edges would cross at most once since shortest paths cross at most once.

Hubard, Kaluža, de Mesmay and Tancer provided such a metric for the sphere, the torus, the projective plane and the Klein bottle but the problem in its generality is still open. They introduced a relaxation of the problem. A *k-universal shortest path metric* is a metric on a fixed surface such that each graph embeddable on this surface can be embedded so that each edge is a concatenation of at most k shortest paths. They provided

an $O(g)$-universal shortest path metric for any orientable surface of genus $g > 1$. They also showed that this relaxation reproves Negami's result in the orientable case.

The main technical tool in this proof is an algorithm that computes a short octagonal decomposition of orientable surfaces provided by Colin de Verdière and Erickson [21]. In this chapter, we provide a proof for the existence of an $O(g)$-universal shortest path metric for non-orientable surfaces by first generalizing this octagonal decomposition for non-orientable surfaces. This, to the best of our knowledge, provides the first short decomposition of non-orientable cross-metric surfaces (of length $O(g|E(G)|)$ where G is the primal graph on the surface). Later in Chapters 5 and 6, we provide other decompositions of short length for non-orientable surfaces.

4.1.1 Our results

Negami claimed in [59] that if we have two graphs embeddable on a closed surface, we can reembed them simultaneously such that their edges cross few times. The Betti number of a connected graph G is $\beta(G) = |E(G)| - |V(G)| + 1$.

Negami's claim ([59, Theorem 1])**.** *Let G_1 and G_2 be two connected graphs embeddable on a closed surface of genus g, orientable or non-orientable. We can embed them simultaneously such that they intersect transversely in their edges at most $4g\beta(G_1)\beta(G_2)$ times.*

We first show that Negami's proof for this claim has a subtle flaw in the case of non-orientable surfaces. We exhibit a specific counterexample to the proof technique. Then we provide an alternative proof based on a different technique.

Theorem A. *Let S be a non-orientable surface of genus $g \geq 1$ and G_1 and G_2 be two graphs embedded on S. Then there exists a homeomorphism h such that any edge of $h(G_1)$ crosses each edge of G_2 at most $O(g)$ times. In particular, the total number of crossings between $h(G_1)$ and G_2 is $O(g|E(G_1)||E(G_2)|)$.*

In order to prove this theorem we take advantage of a technique in [54] to compute a short orienting curve.

Another application of cutting along an orienting curve is the following theorem, generalizing results on universal shortest path metrics obtained in [50] to non-orientable surfaces.

Theorem B. *For $g \geq 3$, there exists an $O(g)$-universal shortest path metric on the non-orientable surface N_g.*

The proof of Theorem B relies, after cutting along an orienting loop, on techniques fairly identical to those in [50]. The main technical tool in the proof provided in [50] is a hexagonal decomposition that is directly obtained from the octagonal decomposition of

orientable surfaces provided in [21]. We provide an analogue of these decompositions for non-orientable surfaces.

Theorem C. *Let N be a non-orientable cross-metric surface, with genus $g \geq 3$ and no boundary. We can decompose N into $2g-4$ hexagons when g is even and into $2g-6$ hexagons and 4 pentagons when g is odd such that the multiplicity of each curve in the decomposition is $O(1)$ except for one closed curve which has multiplicity $O(g)$. Furthermore, the graph of the decomposition is the graph shown in Figure 4.7 and its dual graph is the one shown in Figure 4.8.*

Outline. We prove Theorem A in Section 4.2 and Theorems B and C in Section 4.3.

4.1.2 Main ideas and proof techniques

One of the techniques we use in this chapter is contracting a spanning tree of the embedded graphs as explained in Section 3.3.1. This simplifies the objects we deal with without making it lose its important topological features.

In proving both of the results in this chapter, we also use an orienting curve (as explained in Section 3.7) to first reduce the problems to the orientable case where similar results are known. By Lemma 3.7.1, given a graph embedded on a non-orientable surface, one can compute an orienting curve that crosses each edge of the graph at most a constant number of times. With this tool at hand, we provide a corrected proof of Theorem A with slightly worse constants. Also, this lemma enables us to generalize the octagonal decomposition in [21] to the non-orientable setting by first cutting the non-orientable surface along an orienting curve.

4.2 Correcting Negami's proof

In this section we show that in the non-orientable case, Negami's proof for his claim (see Section 4.1.1) is incorrect, but the statement remains correct.

In his argument, he reduces the proof of his claim to two simple lemmas regarding the number of crossings between sets of arcs on a punctures surface. Lemma 4.2.1 is his exact reduction of the claim in the orientable case and is correct (see [59, Lemma 3]). In the case of non-orientable surfaces, he claims a similar constant to the orientable case (see [59, Lemma 4]) but his proof technique is not correct. Here we prove Lemma 4.2.2 that is analogous to his claim in [59, Lemma 4] but with a slightly worse constant. An arc is called an *essential proper* arc if it does not cut off a disk from the surface.

Lemma 4.2.1. *For two orientable surfaces M_i of genus $g \geq 1$, with one boundary component and β_i disjoint essential proper arcs ($i = 1, 2$) where $\beta_i \leq \beta(G_i)$, there exist an*

orientable surface M of genus g with one boundary component and homeomorphic embeddings of M_1 and M_2 in M so that the images of the arcs in M_1 and M_2 intersect at most $4(g-1)\beta_1\beta_2$ times.

Lemma 4.2.2. *For two non-orientable surfaces N_i of genus $g \geq 1$ with one boundary component and β_i disjoint essential proper arcs ($i = 1, 2$), there exist a non-orientable surface N of genus g with one boundary component and homeomorphic embeddings of N_1 and N_2 in N so that the images of the arcs in N_1 and N_2 intersect at most $18(g-1)\beta_1\beta_2$ times when g is odd and $72(g-2)\beta_1\beta_2$ times when it is even.*

In the proof of Lemma 4 in [59] (that is analogous to Lemma 4.2.2), Negami uses induction on the genus of the non-orientable surface. Assuming that the claim is true for genus $g-1$, to prove it for genus g, he claims that there is an essential proper arc α that runs along the center line of a Möbius band (a one-sided arc). This arc can be either included in the system of arcs or be disjoint from it. The idea is then to cut along α to get a non-orientable surface of genus $g-1$ to use the induction hypothesis. The problem lies in the fact that such an arc might be orienting. Cutting along an orienting arc leaves us with an orientable surface and this interferes with the induction. This is illustrated in the following lemma.

Lemma 4.2.3. *Consider the non-orientable surface of genus 3 with one boundary component and embedded essential arcs shown in Figure 4.1. Any one-sided arc disjoint from the embedded arcs is orienting.*

Proof. Figure 4.1 depicts a non-orientable surface of genus 3 (obtained by identifying the boundary edges according to their letters and orientations) and one boundary component, and a family of essential arcs on it (consisting of the boundary edges a, b and c and the blue arcs). The arc c is a one-sided orienting arc and a and b are two non-homotopic two-sided arcs. The blue arcs are two-sided arcs embedded on the surface. A one-sided arc disjoint from the system of arcs in this polygonal schema, must have one end on the segment of the boundary component between two copies of c and the other end on one of the two other segments adjacent to c. Such an arc is orienting and by cutting along it, we obtain an orientable surface of Euler genus 2 with one boundary. □

A Correction. To prove Theorem A, we provide a different proof of Lemma 4.2.2. The idea of the proof is to cut the surface along an orienting curve that does not cross the graph embedded on the surface too many times (this curve exists by Lemma 3.7.1). By cutting along such a curve, we obtain an orientable surface and we can use Lemma 4.2.1. Let us first introduce some additional terminology that is tailored to surfaces with boundaries.

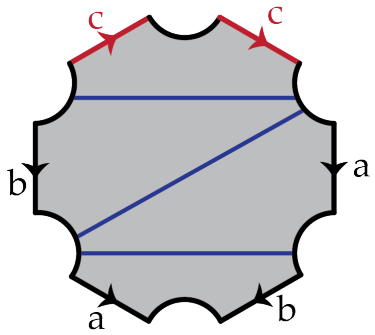

Figure 4.1: A non-orientable surface of genus 3 with embedded system of arcs. The dents in the picture indicate the segments of the boundary component.

Let M be an orientable surface with boundaries that are given with some orientations. We choose an orientation for M, i.e., a consistent choice of clockwise and counter-clockwise for simple contractible curves (see, e.g., Hatcher [45, Section 3.3] for a formal definition); such an orientation induces a (possibly different) orientation for each boundary, which we call its natural orientation. We say that the orientations of the boundaries are mutually *compatible* if they either all match the natural orientation or are all oriented oppositely to the natural orientation. Figure 4.2 shows a surface with non-compatible boundary orientations.

Lemma 4.2.4. *Let N be a non-orientable surface of even genus and M be the orientable surface obtained from cutting N along an orienting curve. The orientations on the two boundaries of M induced by the orienting curve are compatible.*

Proof. If the orientations on the boundaries in M are not compatible, identifying these boundaries corresponds to adding a handle to the surface which in turn implies that the surface we obtain is orientable. This contradicts the fact that N was non-orientable. Figure 4.2 illustrates this. □

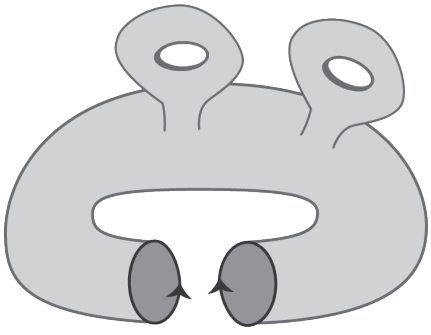

Figure 4.2: An orientable surface with non-compatible boundary orientations

Recall that for an arc with both ends on the same boundary component, we say that the arc is two-sided (resp. one-sided) if the closed curve obtained by connecting the two ends of the arc along one of the boundary segments is two-sided (resp. one-sided). An orienting closed curve γ is either two-sided or one-sided depending on the genus, as described in Lemma 3.2.8. In the same spirit of this lemma, one can characterize orienting arcs instead of orienting closed curves, the only difference is that when g is odd, $h_\gamma = h$ and when g is even, $h_\gamma = h + 1$. Figure 4.3 shows the boundary/s that appear after cutting along a one-sided and a two sided arc.

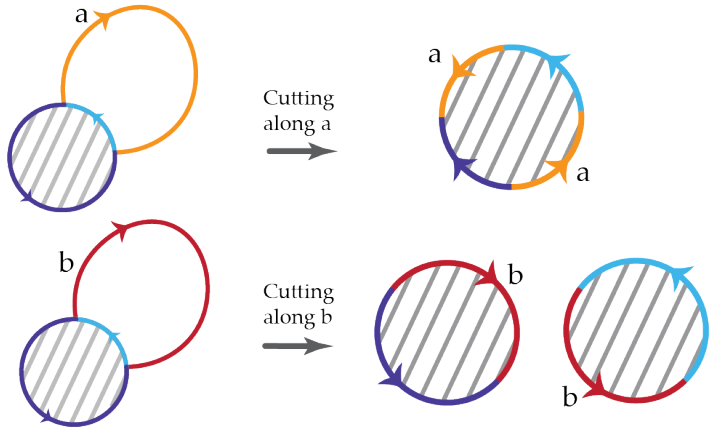

Figure 4.3: The arc a is one-sided and b is two-sided. The segments of the boundary divided by the arcs is depicted in light and dark blue. The figure shows the effect of cutting along a and b on the number of boundaries.

Proof of Lemma 4.2.2. For $i = 1, 2$, let N_i be a non-orientable surface of genus g and with one boundary component, with Γ_i, a system of β_i disjoint essential proper arcs. We distinguish the cases where g is odd from even.

g is odd. Let γ_i be an orienting arc in N_i which has $c_i \leq 2\beta_i$ intersections with Γ_i, whose existence is guaranteed by Lemma 3.7.1 and Remark 3.7.2. Cut N_i along γ_i. Let M_i be the resulting surface. From Lemma 3.2.8 we know that M_i is an orientable surface of orientable genus $\frac{g-1}{2}$ and by the discussion before this proof, we know that it has one boundary component. Each arc in Γ_i is cut into at most 3 arcs by γ_i. We denote by Γ_i' the system of disjoint essential arcs in M_i and thus we have $|\Gamma_i'| \leq 3\beta_i$. By Lemma 4.2.1, we know that there exist a surface M' with genus $\frac{g-1}{2}$ and one boundary component and homeomorphisms ϕ_1 and ϕ_2 which map M_i to M' such that $\phi_1(\Gamma_1')$ and $\phi_2(\Gamma_2')$ have at most $4^{\frac{g-3}{2}}|\Gamma_1'||\Gamma_2'| \leq 36^{\frac{g-3}{2}}\beta_1\beta_2$ crossings.

We modify ϕ_2 in a small neighborhood of the boundary so that the copies of γ_2 are aligned with those of γ_1. This will allow us to glue back the surface along γ_1 (or γ_2). In order to do so, we slide out the ends of the arcs in Γ_2' which are not on γ_2, containing

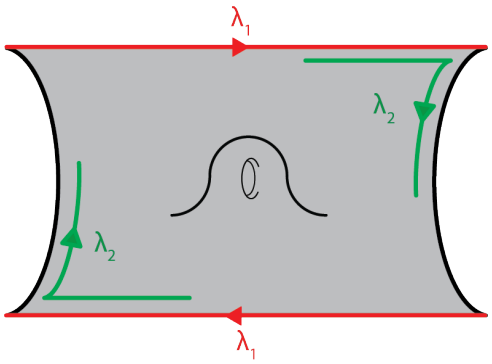

Figure 4.4: The non-orientable surface cut along an orienting curve. The handle depicted in the middle of the polygon is a representation for all the handles on the surface.

$2\beta_2$ ends, into the two segments of the boundary of N_1. Each one of the ends we are sliding might intersect each end of the arcs in Γ_1' which lies in the two segments of the primary boundary component which contains $2\beta_1$ ends. This modification introduces at most $4\beta_1\beta_2$ intersections. Each copy of γ_i contains c_i ends of the arcs Γ_i'. Next we align copies of γ_1 with γ_2 and this introduces at most c_1c_2 new intersections on each copy. Therefore, we have $4\beta_1\beta_2 + 2c_1c_2 \leq 12\beta_1\beta_2$ new intersections.

After this modification, we are able to glue back M' along γ_1 (or γ_2) to get back to a non-orientable surface N with genus g and one boundary component. Now ϕ_1 and ϕ_2 introduce two homeomorphisms from N_1 and N_2 into N such that $\phi_1(\Gamma_1)$ and $\phi_2(\Gamma_2)$ intersect at most $12\beta_1\beta_2 + 36\frac{g-3}{2}\beta_1\beta_2 \leq 18(g-1)\beta_1\beta_2$ times.

g is even. Let γ_i be an orienting arc in N_i which has $c_i \leq 2\beta_i$ intersections with Γ_i whose existence is guaranteed by Lemma 3.7.1 and Remark 3.7.2. Let M_i be the surface obtained by cutting N_i along γ_i. Each arc in Γ_i is cut into at most 3 arcs by γ_i. We denote by Γ_i' the system of disjoint essential arcs in M_i and we have $|\Gamma_i'| \leq 3\beta_i$. By Lemma 3.2.8, we know that M_i is an orientable surface of genus $\frac{g-2}{2}$ and by the discussion before this proof, we know that it has two boundary components. Also we know that it has $3\beta_i$ disjoint essential arcs. Fixing an orientation for γ_i will let us see how the two boundary components were pasted and according to Lemma 4.2.4, we know that these orientations are compatible (see Figure 4.5).

We claim that there is a curve ν_i with an end on each of the boundary components and with at most $3\beta_i$ intersections with the system of arcs in M_i. We know that cutting along an arc increases the Euler characteristic of the surface by 1 (Lemma 3.2.2) and we choose ν_i such that it reduces the number of boundary components by 1. From the relation $\chi = 2 - 2g - h$ where h is the number of boundary components, we will see that after the cut, the genus is unchanged. Therefore, by cutting along ν_i, we get the orientable surface M_i' with genus $\frac{g-2}{2}$ and one boundary component with $6\beta_i$ disjoint essential arcs.

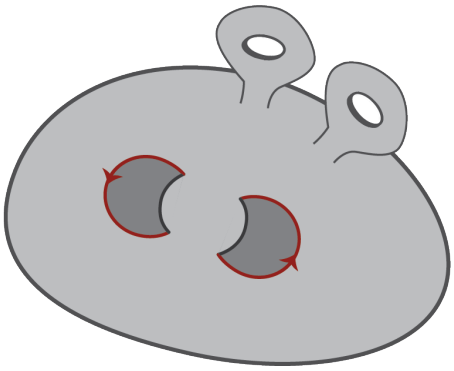

Figure 4.5: The surface M_i, the red arcs are the copies of γ_i.

We choose ν_i such that its ends lie on the segments of the boundary components which belonged to the primary boundary component of N_i and so that it connects these two boundaries. The problem reduces to finding such an arc so that it has at most $3\beta_i$ intersections with the system of arcs in M_i. Consider the graph H such that the vertices are the endpoints of the arcs and the edges are the arcs Γ_i' which we denote by E_1 together with the segments on the boundary components denoted by E_2. We choose a face ρ which has one edge in E_2 such that a part of this edge lies on the segment of the primary boundary component. We choose the face ρ' analogously to ρ but on the other boundary component. The shortest path between vertices associated to ρ and ρ' in the dual graph of H induces a curve with ends on ρ and ρ' that passes through the inner faces at most once. We connect the ends of this curve to the edges on the boundary components in both ρ and ρ'. This gives us an arc ν_i that intersects each edge of E_1 at most once. Thus ν_i is the desired arc.

Again, by Lemma 4.2.1, we know that there exist ϕ_1 and ϕ_2 mapping M_1' and M_2' to M' with $4\frac{g-4}{2} \cdot 6\beta_1 \cdot 6\beta_2$ intersections on the arcs (see Figure 4.6). Similar to the case where the genus was odd, we can modify ϕ_1 and ϕ_2 by introducing at most $4 \cdot 6\beta_1 \cdot 6\beta_2$ intersections such that we can align copies of γ_1 with γ_2 and ν_1 with ν_2 and glue back M' to obtain N and at the end we have at most $72(g-2)\beta_1\beta_2$ intersections. This finishes the proof. □

Proof of Theorem A. Once we are equipped with Lemma 4.2.2, the proof of Theorem A follows the same strategy as that of Negami [59]. For each $i = 1, 2$, we contract a spanning tree in G_i, reducing it to a one-vertex embedding scheme, and remove contractible arcs, yielding an embedding scheme E_i. We puncture the surface at the single vertex of E_i, yielding a non-orientable surface N_i with one boundary component and $O(|E(G_i)|)$ disjoint essential arcs. We are now in the situation to apply Lemma 4.2.2, which gives us a pair of homeomorphisms sending N_1 and N_2 to N such that the number of crossings between the arcs is $O(g|E(G_1)||E(G_2)|)$. We glue back a disk on the puncture and connect all the

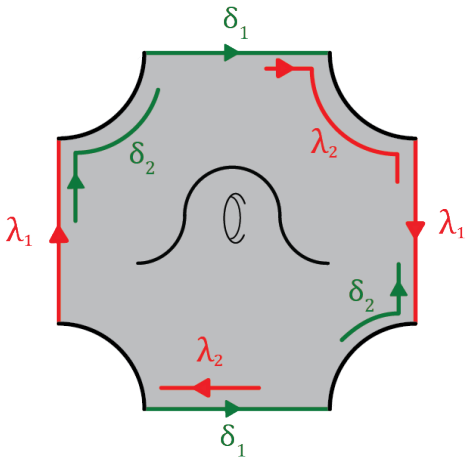

Figure 4.6: The orientable surface M' with one boundary component. The handle depicted in the middle of the polygon is a representation for all the handles on M'.

incoming arcs of E_i to a basepoint b_i, so that the two basepoints b_1 and b_2 are slightly off. This induces an additional $O(|E(G_1)||E(G_2)|)$ crossings. Finally, we add back the contractible arcs and uncontract the spanning trees in small neighborhoods of b_1 and b_2, which does not change the number of crossings, concluding the proof.

□

4.3 An $O(g)$-universal shortest path metric for non-orientable surfaces

In [50], the authors initiate a study on universal shortest path metrics in an attempt to reduce Negami's conjecture (Conjecture 1) to a problem in metric geometry. They provide an $O(g)$-universal Riemannian metric of constant curvature -1 for any orientable surface of genus $g > 1$ ([50, Theorem 4]). Furthermore, they provide an alternative proof for Negami's bound on the joint crossing numbers of graphs embeddable on orientable surfaces (Theorem 4.2.1) that is derived from their proof techniques as a corollary of their theorem ([50, Corollary 20]).

In this section, we provide a generalization of their result for non-orientable surfaces that was left as an open problem in [50]. At the end we explain how Negami's bound on the joint crossing number of two graphs (for non-orientable surfaces) can be deduced from our theorem. In the following, when we refer to a metric we mean a Riemannian metric.

We prove the following theorem.

Theorem B. *For $g \geq 3$, there exists an $O(g)$-universal shortest path metric on the non-orientable surface N_g.*

The construction in the orientable case is based on a decomposition of orientable genus g surfaces into hexagons. Given a graph embedded on a surface, it is first proved that there exists a *hexagonal decomposition* such that each edge of the graph is cut $O(g)$ times by the graph of the decomposition. Each hexagon is endowed with a hyperbolic metric and the following theorem is applied on each hexagon.

Theorem 4.3.1 ([50, Theorem 18], see also [24])**.** *Let G be a graph embedded as a triangulation in a hyperbolic hexagon H endowed with the metric of an equilateral right-angled hyperbolic hexagon. If there are no edges between two non-adjacent vertices on the boundary of H in G, then G can be embedded with geodesics, with the vertices on the boundary of H in the same positions as in the initial embedding.*

A convex hyperbolic hexagon can be isometrically embedded in the hyperbolic plane, and therefore exactly one geodesic connects any pair of points. Thus in this setting, geodesics are shortest paths and this theorem allows us to re-embed the part of G restricted to each face with shortest paths. The metric that we obtain by pasting these hyperbolic hexagons is an $O(g)$-shortest path metric.

It turns out that we can generalize the hexagonal decomposition for non-orientable surfaces, and apply the same strategy as in the orientable case.

Theorem C. *Let N be a non-orientable cross-metric surface, with genus $g \geq 3$ and no boundary. We can decompose N into $2g-4$ hexagons when g is even and into $2g-6$ hexagons and 4 pentagons when g is odd such that the multiplicity of each curve in the decomposition is $O(1)$ except for one closed curve which has multiplicity $O(g)$. Furthermore, the graph of the decomposition is the graph shown in Figure 4.7 and its dual graph is the one shown in Figure 4.8.*

To prove the theorem, we use the following lemma.

Lemma 4.3.2. *Let M be an orientable cross-metric surface with orientable genus $g \geq 1$, and two (resp. one) boundary component(s). We can decompose M into $2g$ octagons (resp. 2 hexagons and $2g-2$ octagons) such that each edge of the decomposition has multiplicity $O(1)$.*

The proof of this lemma is completely analogous to that of the octagonal decomposition in Colin de Verdière and Erickson [21, Theorem 3.1], and thus we only sketch it. The first step in their proof is to cut the surface along a shortest non-separating curve, yielding a surface with two boundaries. Their approach applies equally well if one starts with a surface with two boundaries, as it suffices to skip that first step. Their second step is to

connect the two boundaries with an essential arc so as to have a single boundary. Thus, we can deal with the case of a surface with a single boundary by skipping these first two steps.

Then, the octagonal decomposition is obtained, after cutting along a maximum sequence of non-separating curves (*unzipping*), by going backwards (*zipping*) and adding all the curves one by one (see [21, Section 3 and Figure 3.2(a)]). In the case of two boundaries, we can go back in an identical fashion to obtain an octagonal decomposition. In the case of a single boundary, this backwards step ends one step earlier (since one additional step was skipped at the start), hence we obtain two hexagons instead of octagons. In that case, the Euler characteristic is odd ($= 2 - 2g - 1$) and thus no decomposition made exclusively of octagons can exist.

Proof of Theorem C. Denote by G the primal graph of the surface N. The first curve in the decomposition is the orienting curve λ with multiplicity 2 that exists according to Lemma 3.7.1. Cutting along λ, we get an orientable surface with boundary which we denote by M and we denote by G' the graph we obtain from G by cutting along λ. Each edge of G is cut into at most three sub-edges. Using Lemma 3.2.8, we know that the orienting curve is two-sided (resp. one-sided), if the genus of the surface is even (resp. odd). Therefore, if g is odd, M has orientable genus $\frac{g-1}{2}$ and one boundary component and if g is even, M has orientable genus $\frac{g-2}{2}$ and two boundary components.

By Lemma 4.3.2, we can find a decomposition of M into octagons when g is even, and to both octagons and hexagons when g is odd, such that each curve in the decomposition crosses each edge of G' a constant number of times, see Figure 4.7. Since each edge of G is cut into at most three edges in G', we know that the multiplicity of each curve is constant with respect to G.

We add an arc ρ that follows closely the sub-paths of the curves in the decomposition obtained by Lemma 4.3.2 as depicted in Figure 4.7. The multiplicity of this curve is $O(g)$ and divides each octagon into two hexagons (and each hexagon to two pentagons). The segments of this curve that belong to each face have constant multiplicity in G' and therefore in G.

There are two arcs that are intersecting the copies of the orienting curve. At the end, we slide the ends of these curves very close to the boundary so that all of them build a closed curve after gluing the surface back along λ. Since the orienting loop has multiplicity 2, this adds at most 2 to the multiplicity of these arcs. This finishes the proof. □

Theorem 4.3.1 is also valid for hyperbolic pentagons with the same proof. As in the orientable case, we apply Theorem 4.3.1 to each polygon of the decomposition provided by Theorem C to obtain the theorem. We rely on the following proposition, showing that this yields an $O(g)$-shortest path embedding.

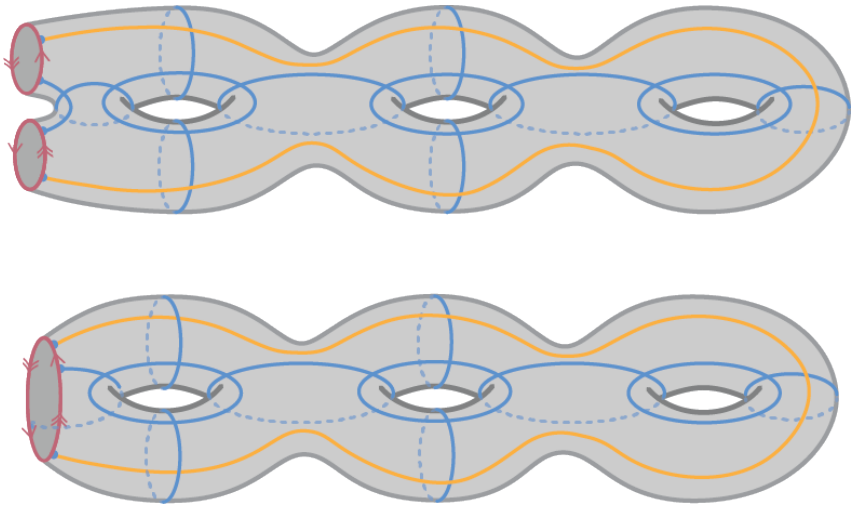

Figure 4.7: Our decomposition of non-orientable surfaces with even genus 8 (top picture) and odd genus 7 (bottom picture). The pink curve is the orienting curve and the orientations shows how they are pasting together. We can see that the boundaries in the even case have compatible orientations according to Lemma 4.2.4. The yellow curve is the arc ρ added as the last step in the proof of Theorem C.

Proposition 4.3.3. *For every face F in the decomposition, every path between $x, y \in F$ that is a shortest path in F is also a shortest path in N.*

Coupled with Theorem C, this proposition immediately implies Theorem B. Its proof makes strong use of the symmetries of the decomposition, which we first introduce.

Figure 4.8 depicts the graphs dual to the decompositions output by Theorem 4.3.1. It depicts two involutions σ_1 and σ_2 which, since all the hexagons are isometric, and in the case of odd genus, the 4 pentagons are isometric, induce isometric involutions of the whole surface. If we take the square vertices to correspond to the top polygons in the decomposition and the star vertices to be the ones in the bottom, then the involution σ_1 maps the top polygons to the bottom polygons and vice versa and it is identity on the edges that are in common between top and bottom polygons. In particular, σ_1 maps the faces a and d to each other and b and c to each other. Similarly, σ_2 maps neighboring polygons at the top (and the bottom) on each other, in particular it maps a and b to each other and d and c to each other. We can cut N into four planar quadrants which we denote by Q_i, $1 \leq i \leq 4$. Each of these quadrants are linear concatenation of hexagons and pentagons. Take Q_1 to be the pictured quadrant in Figure 4.9. We can see that each of the quadrants Q_2, Q_3 and Q_4 can be obtained by applying one of the σ_1, σ_2 and $\sigma_1\sigma_2$ to Q_1.

Proof of Proposition 4.3.3. Take two points x and y in a face F in Q_1 and let θ be a shortest path between them. The path θ may leave the face F, but we will show that

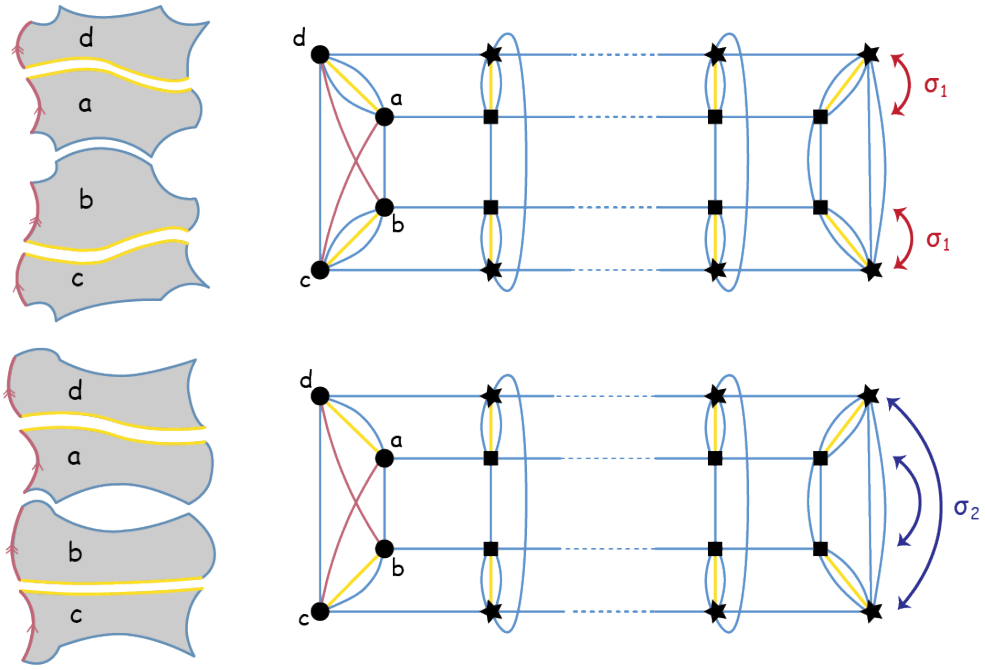

Figure 4.8: Right: the dual graph of the decompositions in the cases of even genus (top picture) and odd genus (bottom picture). The circular vertices correspond to the four polygons that are adjacent to the orienting curve and the pink edges are the dual edges to the segments of the orienting curve. Left: the faces a, b, c and d are the faces adjacent to the orienting curve. Note that these four faces are pentagons (resp. hexagons) in the case where the genus is odd (resp. even).

in that case there is another shortest path between x and y that remains inside F. If θ leaves Q_1, we reflect the parts of the path that leave Q_1, using one of the maps σ_1, σ_2 or $\sigma_1\sigma_2$ back in Q_1. We need to check that the reflected parts together with the part of θ that is inside Q_1 define a path. The only troublesome case here is when our path leaves Q_1 by crossing the orienting curve γ. In this case, we can see that the path enters c. We use $\sigma_1\sigma_2$ to reflect the sub-path in c back to a. A closer look at $\sigma_1\sigma_2$ shows that it is identity on γ, therefore the sub-path in c gets reflected to a in a way that defines a new path in Q_1. We call the new path θ'. Since the maps are all isometric, θ' has the same length as θ and therefore it is a shortest path between x and y which remains in Q_1.

We show that θ' must be contained in F. Let us assume that θ' leaves F. Since the dual graph of each Q_1 is a line, there is a face F' in Q_1 that θ' enters and leaves once. We denote by α, the sub-path of θ' inside F'. The endpoints of α are both on the same edge of this face. α could be shortcut by following this edge instead of going inside the face F'; see Figure 4.10.

This implies that θ' cannot leave F and it is the shortest path between x and y that we were looking for. This finishes the proof. □

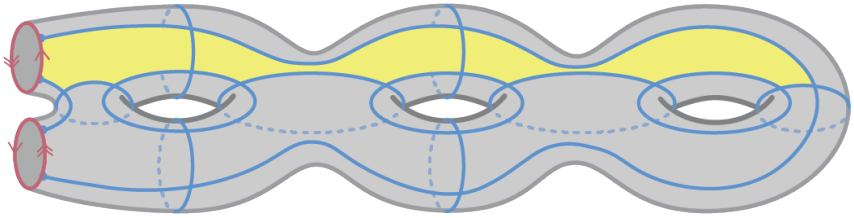

Figure 4.9: The surface N cut along the orienting curve γ and the yellow area shows the quadrant Q_1.

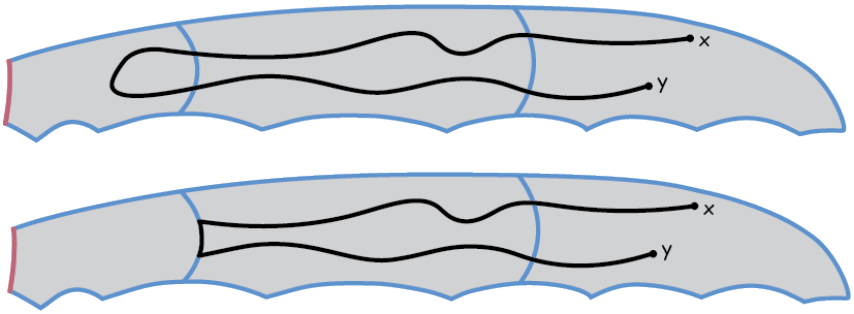

Figure 4.10: The figure shows the shortest path θ' in Q_1 and how it can be shortcut.

As for the orientable case, our proof techniques also provide an alternative proof of Negami's bound (which we corrected in Theorem A) on joint crossing numbers on non-orientable surfaces. If two graphs are simultaneously embedded with our $O(g)$-universal shortest path embeddings, then each edge is cut into $O(g)$ shortest paths, but each hexagon/pentagon contains at most $O(1)$ of those. Since two shortest paths cross at most once, it follows that each pair of edges crosses at most $O(1)$ times in each hexagon/pentagon, and thus $O(g)$ times in total.

Chapter 5

Short Non-orientable Canonical Decomposition

Summary. In this chapter we provide a polynomial-time algorithm that for any non-orientable cross-metric surface computes a canonical decomposition so that any loop in the canonical system intersects any edge of the primal graph in at most 30 points.

The result in this chapter was obtained with Alfredo Hubard and Arnaud de Mesmay and is the main result appearing in [A] which has been published in Discrete & Computational Geometry. This result also appears in a preliminary version that is published in the Proceedings of the 38th Symposium on Computational Geometry.

5.1 Introduction

Decomposing a surface along a graph or a curve is a standard way to simplify its topology. The classification of surfaces and classical tools to compute both homology groups and fundamental groups typically rely on such topological decompositions, which are also important in meshing and 3D-modeling (see [48]). Surfaces often come with extra structure which can be modeled by an embedded graph (see the cross-metric model and the combinatorial model in Section 3.5.1). Decomposing such a surface efficiently corresponds to finding another cellularly embedded graph that has few (transverse) intersections[1] with the original graph (see for example [53] or [34]). Such decompositions also appear in

[1]Throughout this chapter and Chapter 6, we use the cross-metric model in which we decompose surface-embedded graphs by cutting them along embedded graphs which are transverse to the primal graph, and count the number of intersections. This is equivalent to the primal setting studied in, e.g., Lazarus, Pocchiola, Vegter and Verroust [53] via graph duality.

algorithm design: often, to generalize results on planar graphs to graphs embedded on surfaces, it is enough to find a decomposition that cuts open the surface into a disk, then solve the resulting planar instance and stitch back the solution, see, e.g., [33, 53, 19].

In many applications, it is important that the graph along which we cut is canonical in some sense. For example, as we mentioned in the introduction, to obtain a homeomorphism between two surfaces, we can cut them into disks and extend the homeomorphism between the disks to a homeomorphism between the surfaces. However, this only works if the cut graphs have the same combinatorial structure. A seminal result on topological decompositions was pionereed by Lazarus, Pocchiola, Vegter and Verroust [53] (see also [52]) who designed an algorithm that finds, for any graph G embedded on a closed orientable surface S a canonical decomposition H such that no edge of H intersects any edge of G more than a constant number of times. Recall that a canonical system of loops for an orientable surface is a one-vertex and one-face embedded graph in which the cyclic order of the edges around the vertex is $a_1b_1a_1b_1 \dots a_gb_ga_gb_g$.

On the other hand, before our work, it was unknown whether cross-metric non-orientable surfaces could admit similar short non-orientable canonical decompositions. Recall that a non-orientable canonical decomposition is a system of one-sided loops with the cyclic order $a_1a_1a_2a_2 \dots a_ga_g$ around the vertex. The best known upper bound for the length of this decomposition was $O(g^2|E(G_2)|)$ (see [52]), which matches the bound proved by Negami (see Negami's claim or Theorem A in Section 4.1.1). Furthermore, this upper bound can be achieved by the same technique as in the proof for computing a short orientable canonical decomposition [52, Theorem 4.3.9].

In this chapter, we provide an algorithm using a new approach that computes a short canonical decomposition of a non-orientable cross-metric surface of total length $O(g|E(G)|)$ which matches the best known upper bound for the length of an orientable canonical decomposition.

Here, it is worth reminding the reader of the relation between finding short decompositions and Negami's conjecture. From the perspective of topological decompositions, Negami's conjecture (Conjecture 1) posits that short decompositions of any fixed shape exist, in the sense that one can always decompose an embedded graph along a chosen topological decomposition (modeled by a second, cellularly embedded graph), in such a way that each edge of the decomposition crosses each edge of the graph $O(1)$ times.[2] The orientable canonical system of loops provided in [53], the octagonal decomposition in [21] and the (almost) hexagonal decomposition of non-orientable surfaces provided by Theorem C in Chapter 4 all provide examples with bounds matching those of Negami's conjecture. The algorithm we introduce shows that the non-orientable canonical decom-

[2]This statement is slightly stronger than Conjecture 1 since it enforces a control on the number of crossings between each pair of edges instead of the total number of crossings, but it is equally open.

position, which is arguably the most natural way to decompose a non-orientable surface, can also be computed with such upper bounds.

5.1.1 Our results

The main result of this chapter is the following theorem.

Theorem D. *There exists a polynomial time algorithm that given a non-orientable cross-metric surface N, computes a canonical decomposition such that each loop in the decomposition intersects any edge of the graph in at most* 30 *points.*

In order to prove Theorem D, we heavily rely on an algorithm introduced by Schaefer and Štefankovič in [69] for drawing graphs on non-orientable surfaces. Compared to that algorithm, our approach enforces more structure on this construction by imposing specific orders in the way we deal with the loops in the induction. In Section 5.3, we first explain this algorithm and then in Section 5.4, we introduce and explain our modifications, ultimately leading to a proof of this theorem in Section 5.5.

5.1.2 Main ideas and proof techniques

One of the techniques we use to prove Theorem D, is to contract a spanning tree of the underlying graph as explained in Section 3.3.1. This reduces the problem to the setting in which the primal graph embedded on the cross-metric surface is a one-vertex graph and the embedding is encoded completely by a one-vertex embedding scheme. Such an embedding scheme will be the basic object with which we work in this chapter and we provide lemmas and tools to deal with these embedding scheme in Section 5.2.

As in the last chapter, we rely on computing an orienting curve by the Lemma 3.7.1. This lemma allows us, at the cost of slightly increasing the constants in our results, to assume that our embedding schemes always have an orienting curve. The existence of this orienting curve will significantly simplify our work to prove Theorem D.

For the proof of Theorem D, we first point out that the techniques used to prove the orientable version in [53] do not readily apply, as they rely on a fine control of the cut-and-pasting operations used in the proof of the classification of surfaces, and in the non-orientable case there is an additional step in these operations which incurs an overhead of $O(g)$ in the number of crossings of the resulting curves (see [52, Theorem 4.3.9]). Instead, our proof of Theorem D builds on important recent work of Schaefer and Štefankovič [69]. The foundational idea behind this work, which takes its roots in an article of Mohar [57] on the degenerate crossing number, is that they use a geometric model (introduced in detail in Section 3.4.1) to represent a graph embedded on a non-orientable surface of genus g as a planar drawing (namely, a cross-cap drawing). We recall that in this model cross-caps are

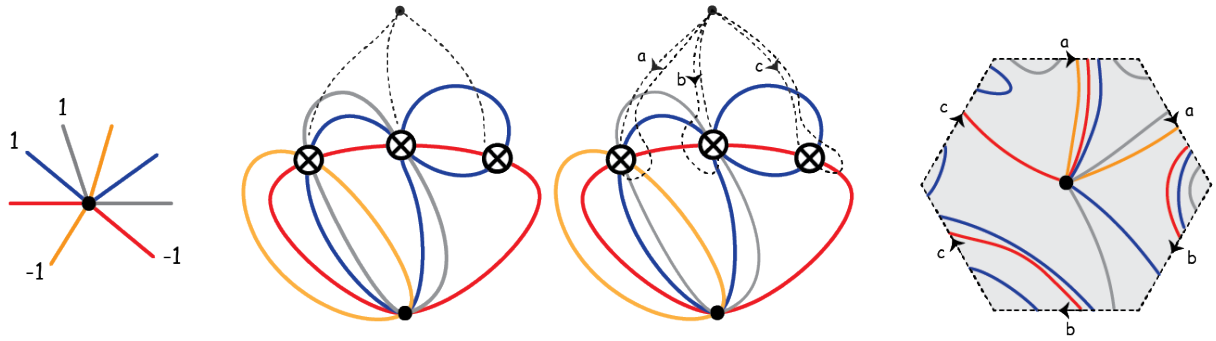

Figure 5.1: From left to right: 1) The combinatorial information of a one-vertex graph. 2) A cross-cap drawing of this graph, with cross-caps connected to a basepoint. 3) A joint drawing of the graph and a canonical system of loops. 4) A different representation: decomposing the graph with a canonical system of loops.

points on the sphere where multiple edges are allowed to cross in a way that reverses the permutation, as illustrated in the second picture of Figure 5.1. Using an intricate argument inducting on the loops of an embedding scheme, Schaefer and Štefankovič showed that any graph embedded on a non-orientable surface can be represented by such a cross-cap drawing so that each edge uses each cross-cap at most twice. Our main technical contribution is to upgrade their construction so that the cross-caps can be connected to each other so as to yield a non-orientable canonical system of loops (Lemma 5.1.1), so that each loop intersects each edge of the one-vertex graph in at most 30 points (see Figure 5.1).

The complexity of the drawings provided by the proof of Schaefer and Štefankovič increases too fast to directly obtain a good bound by just connecting the cross-caps. Therefore, we modify their algorithm. First, by the aforementioned techniques, we can assume that we always have an orienting loop, which simplifies some of the steps and provides additional structure to the inductive argument. But more importantly, we show that one can impose a certain order in which we choose the one-sided loops, as well as the separating loops, in the inductive argument of Schaefer and Štefankovič so as to obtain a finer control on the resulting drawing.

The order in which we choose loops comes from a seemingly unrelated problem in computational biology, and more precisely genome rearrangements. Here we remind the reader some key concepts in this topic that are relevant to our purposes in this chapter and refer the reader to Section 3.8.1 and 3.8.2 for a more detailed introduction. Given a permutation w on a set of distinct letters with signatures (a bit assigned to each letter), a signed reversal consists in choosing a subword in w, and reversing it as well as the signatures of all its letters. The *signed reversal distance* between two signed permutations is the minimum number of signed reversals needed to go from one permutation to the other

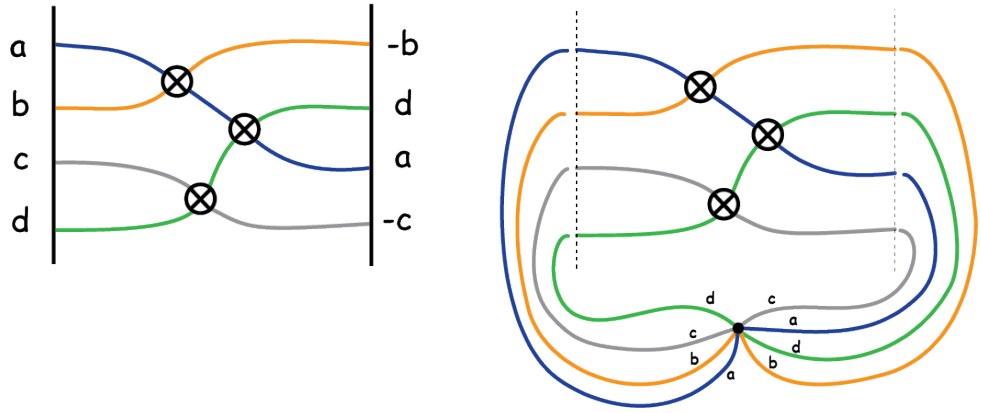

Figure 5.2: Left: a pictorial representation of three signed reversals bringing the signed permutation on the left to the signed permutation on the right. Right: Attaching the two permutations to a common basepoint yields a one-vertex graph with an embedding scheme, and the signed reversals provide a cross-cap drawing of that embedding scheme where each loop enters each cross-cap at most once.

one. This distance, and in particular algorithms to compute it has been intensively studied in the computational biology literature due to its relevance for phylogenetic reconstruction (see for example [46]). A cornerstone of the theory is the breakthrough of Hannenhalli and Pevzner [44] who provided an algorithm to compute the signed reversal distance between two signed permutations in polynomial time (see also the reformulation by Bergeron [6]). Now, as we illustrate in Figure 5.2, there is a very strong similarity between computing the signed reversal distance between two permutations and embedding a one-vertex graph built from these two permutations with a minimum number of cross-caps (see [11, 49]). Surprisingly, the algorithms of Hannenhalli and Pevzner on one side and of Schaefer and Štefankovič on the other also have similarities. The proof of Theorem D leverages the literature on the signed reversal distance problem, and in particular the structure we impose on the cross-caps drawings of non-orientable graphs is inspired on ideas from the aforementioned genome rearrangements algorithms.

The following lemma underpins our strategy to prove Theorem D: in order to find a non-orientable canonical system of loops, first find a cross-cap drawing and connect the cross-caps to a root using short paths (see Figure 5.1).

Lemma 5.1.1. *Let H be a cross-cap drawing for a graph of non-orientable genus g and let b be a point in one face of the drawing. Let $\{p_i\}$ be a family of paths in the dual graph to this drawing from each cross-cap to b. Introduce a loop c_i by starting from b, passing along the path p_i, entering the corresponding cross-cap, going around the cross-cap and passing along p_i to return to b. The system of loops $\{c_i\}$ is a non-orientable canonical system of loops.*

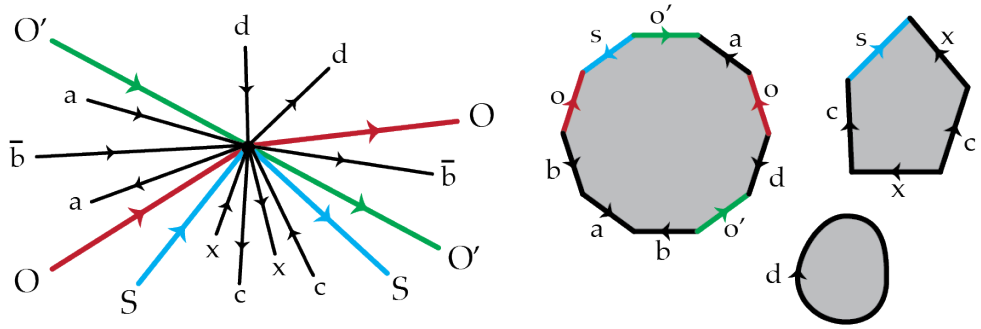

Figure 5.3: Left: a one-vertex embedding scheme; right: the three faces corresponding to the embedding scheme at left.

Proof. It is easy to check that each c_i has consecutive ends around b. Each curve c_i is homotopic to a concatenation of a curve in a planarizing system of disjoint one-sided curves (defined in Section 3.4.1) and two copies of a path p_i, therefore it is one-sided. Cutting along these system of curves corresponds to cutting along a planarizing system of disjoint one-sided loops which gives us a sphere with g boundary components and then cutting along the paths $\{p_i\}$ which connect these boundary components and cut them to a single boundary. Therefore, cutting along $\{c_i\}$, cuts the surface into a disk, and we obtain a non-orientable canonical system of loops. This is illustrated in Figure 5.1. □

5.2 Preliminaries

In this chapter, we deal extensively with one-vertex embedding schemes. In the following, we study these embedding schemes and provide some useful lemmas to work with them.

One-vertex embedding schemes

In a one-vertex embedding scheme, each edge of the graph is a closed curve in the embedding. This makes it easier to learn about the combinatorial features of different types of curves. Specifically, in a one-vertex embedding scheme, the signatures of the edges correspond to their sidedness in the embedding.

Given a one-vertex embedding scheme, a loop l divides the half-edges around the vertex into two parts; each part is called a *wedge* of l. When a loop m has exactly one end in each wedge of l, we say that the ends of m *alternate* with those of l; otherwise, both ends of m are in one wedge of l and we say that the ends of l *enclose* the ends of m and vice versa. For example, in Figure 5.3, the loops c and x are alternating and s and x enclose each other.

Cutting along a curve in a one vertex embedding scheme

Let G be a one-vertex embedding scheme with vertex v and let e be an edge in G. Denote the wedges of e in G by ω_1 and ω_2 and let $\rho_v|_{\omega_i}$ denote the sub-permutation in ρ_v consisting of the edges in the wedge ω_i for $i = 1, 2$. Denote by G_e the embedding scheme that we obtain by cutting along e and contracting the boundary component/s of the surface that we obtain (as introduced in Section 3.3.2), see the rightmost pictures in Figures 5.4 and 5.5. In the following lemma, we use the notation that we introduced in this paragraph.

Lemma 5.2.1. *Let e be a one-sided loop in a one-vertex embedding scheme G with vertex v. Then G_e is a one-vertex embedding scheme with the rotation system $\rho_v|_{\omega_1}\overline{\rho_v|_{\omega_2}}$ around its vertex. Also the signature of each edge that interleaves with e is reversed in G_e.*

Proof. We know that cutting along a one-sided closed curve, reduces the genus by 1 (see Lemma 3.2.3) and increases the number of boundary components by 1. To prove the claim for the rotation system around the vertex and the signatures of the edges, we refer the reader to Figure 5.4. We can see that by pasting the two copies of e that form the boundary in the middle picture, we retrieve the initial rotation system at the left. □

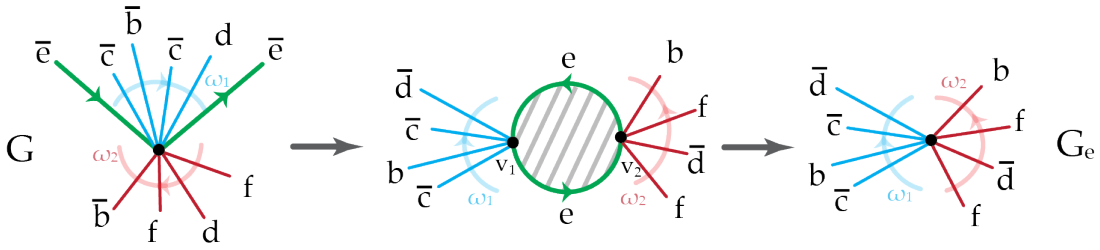

Figure 5.4: Cutting along a one-sided loop and contracting the emerging boundary component.

Lemma 5.2.2. *Let e be a two-sided loop in a one-vertex embedding scheme G with vertex v. Then G_e is a two-vertex embedding scheme. If e is not orienting, then the cyclic permutation of edges around the vertices are given by $\rho_v|_{\omega_1}$ and $\rho_v|_{\omega_2}$ and edges have the same signatures (up to flips). In the case where e is orienting, then G_e is an orientable embedding scheme with cyclic permutations $\rho_v|_{\omega_1}$ and $\overline{\rho_v|_{\omega_2}}$ around the vertices and the signature of every edge that alternates with e is reversed.*

Proof. We know that by cutting along the two-sided closed curve e, we obtain two boundary components that correspond to the copies of e. The proof of the claim for the cyclic permutation of edges around the vertex and the signatures of the edges, is the same as in Lemma 5.2.1 and we refer the reader to Figure 5.5 for an illustration.

In the case where e is orienting, the same description works, but the scheme that we obtain may have non-loop edges of signature -1, which is undesirable for an orientable embedding scheme. This is resolved by applying one flip to one of the vertices, which has the effect of reversing once the permutations and the signature of every edge that alternates with e. This concludes the proof. □

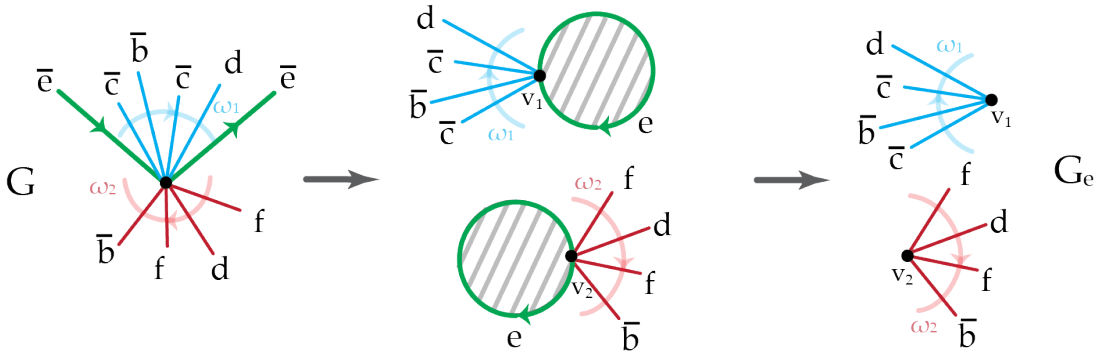

Figure 5.5: Cutting along a two-sided loop and contracting the two emerging boundary components.

Recognizing types of curves in a one-vertex embedding scheme

The following lemma states how to recognize contractible, separating non-contractible and orienting curves in a one-vertex embedding scheme.

Lemma 5.2.3. *Given a one-vertex embedding scheme G,*

1. *A loop d is contractible if and only if it has signature $+1$, it has a wedge ω containing only edges with signature $+1$ and no pair of edges in ω and d interleaves, see the loop d in Figure 5.3.*

2. *A loop is separating if and only if it has signature $+1$ and its ends enclose the ends of any other loop in the scheme, see the loop s in Figure 5.3.*

3. *If G is a non-orientable scheme, a loop in G is orienting if and only if its ends enclose the ends of any two-sided loop and alternate with the ends of any one-sided loop in the embedding scheme, see the loop o in Figure 5.3.*

Proof. 1. Let d be a contractible curve in G. Then d separates a disk D from the surface (Lemma 3.2.11). One side of d bounds D and consequently, one wedge of d belongs to D (see Lemma 5.2.2). Denote this wedge by ω. This implies that no edge in G can interleave with d. If ω is empty, we conclude. Otherwise, any edge that appears in ω can be embedded on D and therefore in the plane. Since two

interleaving edges, or an edge that has signature -1 cannot be embedded on the plane, ω does not contain such an edge. This concludes the forward implication.

For the other direction, note that in one wedge of d no two loops are interleaving and no loop has signature -1. This implies that one of the edges has signature $+1$ and has an empty wedge. Thus it bounds a disk. Removing that edge and recursing, shows that d also bounds a disk and therefore is contractible (see Lemma 3.2.11).

2. Let s be a separating loop in G. If we cut the surface along s, we obtain two connected components. Assume that there is a loop e that interleaves with s. This means that after cutting, each end of e appears in one of the connected components (see Lemma 5.2.2) which is in contradiction with s separating the surface.

   For the other direction, let us assume that s encloses all the loops in G. This means that s divides the scheme into two sub-schemes such that no edge has exactly one end in each one of them. This implies that s is separating. This concludes.

3. First, let us assume that the loop o is orienting. Let G_o denote the surface and the embedding scheme that we obtain by cutting along o and contracting the boundary or boundaries. Then we know that G_o is an orientable embedding scheme.

   Let us first assume that the non-orientable genus $\tilde{g}$ of G is odd. Then by Lemma 3.2.8, o is one-sided and G_o is a one-vertex embedding scheme by Lemma 5.2.1. Since G_o is orientable, all the closed curves in G_o have to be two-sided. Each edge in G_o is a closed curve and therefore all the edges must have signature $+1$. The signatures of edges in G_o are obtained by reversing the signature of every edge that interleaves with o (see Lemma 5.2.1). This implies that the edge o used to interleave with all the loops with signature -1 in G and enclose all the loops with signature $+1$. This finishes the proof of the case where the genus is odd.

   Now let us assume that $\tilde{g}$ is even. Note that any orienting curve in this case is two-sided (Lemma 3.2.8) and therefore o has signature $+1$. Then G_o has two vertices by Lemma 5.2.2. Similar to the previous case, for G_o to be an orientable scheme, all the closed curves in it have to be two-sided. The closed curves in G_o are either loops at a vertex or the concatenation of two non-loop edges. This implies that all loops in G_o have signature $+1$. Since all such loops are formed by the loops with both ends in one wedge of o in G, this implies that all edges that are enclosed by o have signature $+1$. For a closed curve that is obtained by concatenating non-loop edges to be two-sided, we need all such edges to have the same signature. But since in this case o has signature $+1$, it is not possible for all the non-loop edges to have signature $+1$ in G: this implies that G is originally an orientable scheme which is a contradiction. Then all non-loop edges have signature -1 in G. These edges

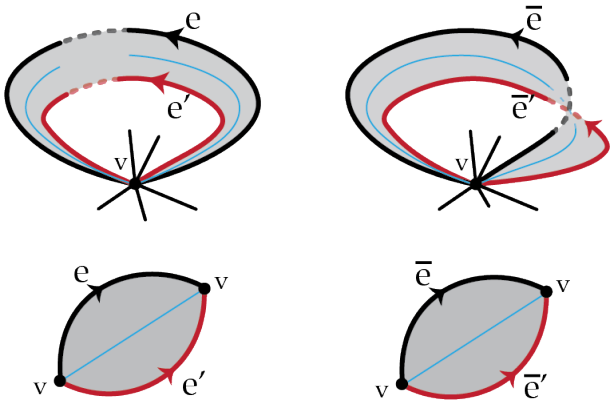

Figure 5.6: Homotopic edges in one-vertex embedding schemes.

correspond to the loops in G that interleave with o. Therefore, all the loops that are interleaving with o have signature -1. This finishes the proof of the forward implication.

For the other direction, we first cut G along o. Depending on the parity of the genus, we get one of the embedding schemes described by Lemmas 5.2.1 and 5.2.2. The proof follows by the observation that all closed curves in both cases become two-sided which implies that G_o is orientable and that o has been orienting.

□

Remark 5.2.4. *An embedding scheme might contain multiple orienting curves, see for example the loops o and o' in Figure 5.3.*

Recognizing homotopic edges in a one-vertex embedding scheme

The following lemma helps us recognize homotopic edges in a one-vertex embedding scheme.

Lemma 5.2.5. *In a one-vertex embedding scheme (G, ρ, λ), two loops e and e' are homotopic if and only if both have the same signature and each end of e appears right next to an end of e' in ρ and*

- *if e and e' have signature $+1$, then the ends of e encloses the end of e' and,*
- *if e and e' have signature -1, then e and e' interleave in ρ.*

Proof. To prove this lemma, it is enough to show that $e.e'$ bounds a disk on the surface (see Lemma 3.2.10). Figure 5.6 shows the disk whose boundary is comprised of these two edges in both cases. This finishes the proof. □

5.3 The Schaefer-Štefankovič algorithm

Throughout this section, we will be working with a one-vertex graph G endowed with an embedding scheme (ρ, λ), which for simplicity we denote by G. Schaefer and Štefankovič proved the following theorem.

Theorem 5.3.1 ([69, Lemma 9])**.** *If G is a one-vertex non-orientable (resp. orientable) scheme (ρ, λ), then it admits a cross-cap drawing with $eg(G)$ (resp. $eg(G)+1$) cross-caps in which every edge passes through every cross-cap at most twice.*

The proof is by induction on the number of loops and is readily algorithmic, computing an actual cross-cap drawing with the required bound on the number of intersections with the cross-caps. Before explaining the algorithm, let us introduce the different moves and techniques we use to deal with different types of loops.

First, let us introduce the following terminology for an embedding scheme around a vertex v. By *flipping* a wedge of a one-sided loop e in a one-vertex scheme, we mean reversing the order of the edges in the wedge and changing the signature of the loops that have exactly one end in the wedge. We call the empty wedge between two consecutive half-edges around v a *root wedge*. A face incident to v in a cross-cap drawing may correspond to more than one root wedge. We refer to such a face as a *root face*.

Contractible loop move. Let c be a contractible loop with consecutive ends in the embedding scheme G. Remove c. The new embedding scheme can be drawn using the same number of cross-caps. Having a drawing for the new embedding scheme, we can draw the loop c without passing through any of the cross-caps.

Gluing move. Let s be a non-contractible separating loop in the embedding scheme G. We divide the embedding scheme to G_1 and G_2 by cutting along s and splitting the vertex into two vertices (the embedding schemes of G_1 and G_2 are induced by the embedding scheme of G). Denote by F_o^1 and F_o^2 the root wedges in G_1 and G_2 in which s formerly was placed. Let H_1 and H_2 be drawings for G_1 and G_2 respectively. We glue the drawings by identifying the root wedges F_o^1 and F_o^2 to get the drawing H' for $G \setminus \{s\}$.

Note that removing s does not change the genus (Lemma 3.2.3) and we have $eg(G) = eg(G_1) + eg(G_2)$. If G_1 and G_2 are both non-orientable, then H' can be extended to a cross-cap drawing for G by adding s without using any of the cross-caps; see Figure 5.7. When at least one of G_1 or G_2 is orientable, say G_2, H' uses one extra cross-cap (G_2 needs $eg(G_2) + 1$ cross-caps to be drawn). In order to get a drawing with minimum number of cross-caps, we need to eliminate one cross-cap from the drawing. To deal with this case, we need the following lemma and the dragging move which allows us to reduce one cross-cap from the drawing.

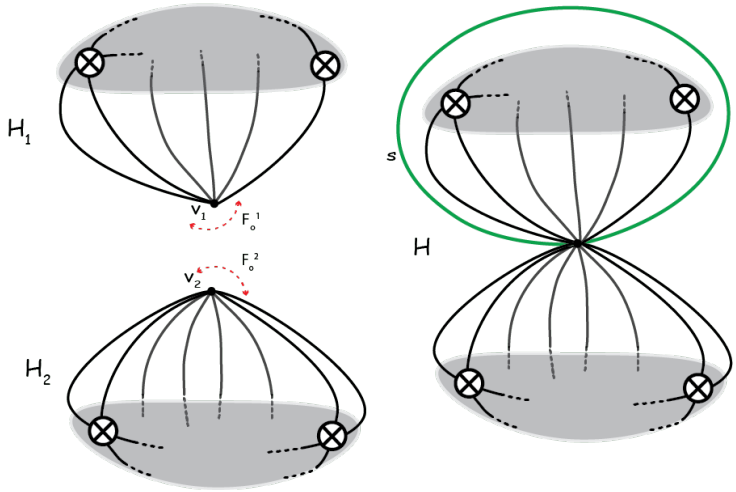

Figure 5.7: The gluing move on two cross-cap drawings when G_1 and G_2 are both non-orientable.

Lemma 5.3.2. *Let G be a one-vertex orientable embedding scheme. Adding a one-sided loop o with consecutive ends to the embedding scheme (anywhere in a rotation around the vertex) increases the Euler genus by 1. Thus, the new embedding scheme needs as many cross-caps as G to embed. Furthermore, the loop o is orienting.*

Proof. Adding a one-sided curve with consecutive ends to an embedding scheme does not change the number of faces. By the Euler formula we know that the Euler genus of the new embedding scheme is increased by 1. By Lemma 3.3.2, G needs $eg(G)+1$ cross-caps to embed. Therefore, the new embedding scheme needs as many cross-caps as G to embed.

The loop o is the only one-sided loop in the embedding scheme and every other loop has both ends in the same wedge of o. Lemma 5.2.3 implies that o is orienting. □

Dragging move. Let us assume that G_2 is orientable. By Lemma 5.3.2, we can add a one-sided loop o with consecutive ends in the root wedge F_o^2 without increasing the number of cross-caps that we need to draw G_2. The loop o is orienting and the new embedding scheme needs $eg(G_2)+1$ cross-caps to be drawn. Having a drawing for the new embedding scheme, we can draw the loop s in the drawing for $G_2+\{o\}$ as follows: we start the loop from one of the root wedges between o and another loop of G_2, we draw s by following o through all the cross-caps, except that after coming out of the last cross-cap, we go back to the first one entered, and traverse all of the cross-caps again. At the end, we follow o back to the vertex; see Figure 5.8. We denote this drawing of $G_2+\{o\}+\{s\}$ by H_2'.

By gluing H_1 to H_2', we get a drawing H' for $G+\{o\}+\{s\}$ but the drawing is not using the minimum number of cross-caps. We eliminate one of the cross-caps in H' as follows.

Let i_1 be the rightmost half-edge in G_1 that follows immediately the separating loop

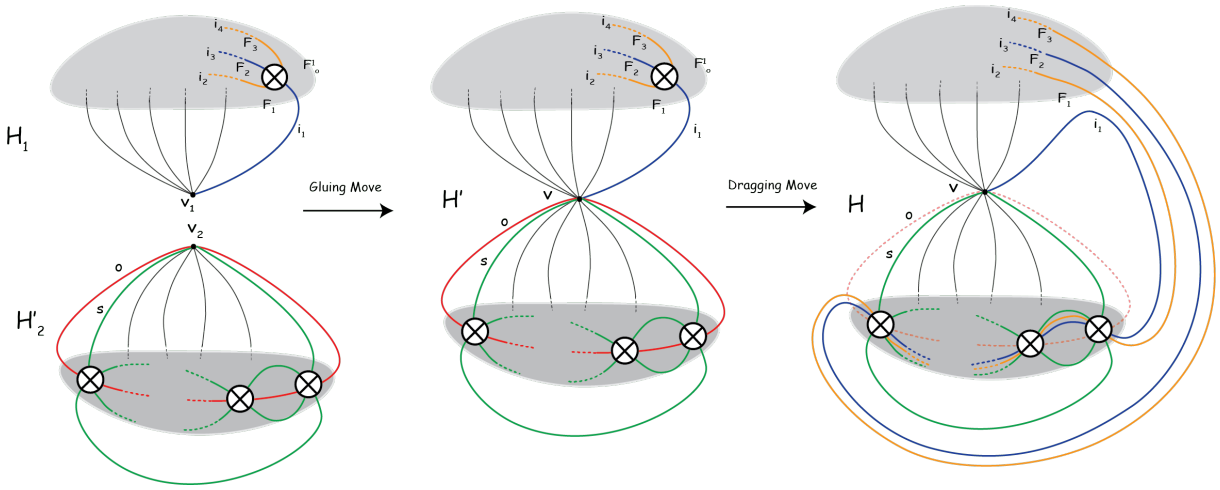

Figure 5.8: Left: the gluing move. Right: the dragging move when G_2 is orientable: the top right cross-cap is removed and the corresponding curves are dragged through the bottom component.

in G. Denote by $\mathfrak{c}$ the first cross-cap that i_1 passes through. Let us assume that there are $2k$ half-edges passing through $\mathfrak{c}$. Let us denote by $(i_1, F_1, \ldots, i_{2k}, F_o^1)$ the alternating sequence of half-edges and faces adjacent to $\mathfrak{c}$ in the cross-cap drawing by moving clockwise around it. Now, we disconnect the edges that enter $\mathfrak{c}$ and remove the cross-cap $\mathfrak{c}$. We drag $i_1, \ldots, i_k$ through all the cross-caps in G_2 along the loop o. After exiting the last cross-cap in G_2, we remove o and we attach the half-edges to their other ends $(i_{k+1}, \ldots, i_{2k})$. Since G_2 uses an odd number of cross-caps (Lemma 3.3.2), the half-edges will have the correct orientability and order to get attached to their other ends; see Figure 5.8. If only one of G_1 and G_2 is orientable, the drawing we get uses $eg(G)$ cross-caps and if both are orientable, we get a drawing with $eg(G) + 1$ cross-cap which is the minimum number of cross-caps needed to draw the embedding scheme in this case.

One-sided loop move. Let r be a one-sided loop in the embedding scheme G. We remove r and flip one of its wedges. One can check that the new embedding scheme G' has Euler genus $eg(G) - 1$. Let us assume that H' is a drawing for G'. We add r to this drawing by adding a cross-cap near the vertex and the flipped wedge and dragging r and every edge in the flipped wedge in it; see Figure 5.9. Note that flipping different wedges of r leads to two different cross-cap drawings. This freedom in choosing the wedge is important for us and we use this later in this chapter.

If r is not orienting, the drawing we get at the end uses $eg(G)$ cross-caps. But if r is orienting, then G' is orientable and any drawing for G' needs an extra cross-cap (Lemma 3.3.2). This means that if we apply a one-sided loop move to an orienting loop, the drawing we get does not use the minimum number of cross-caps (the embedding is not cellular).

We use the following move to deal with orienting loops.

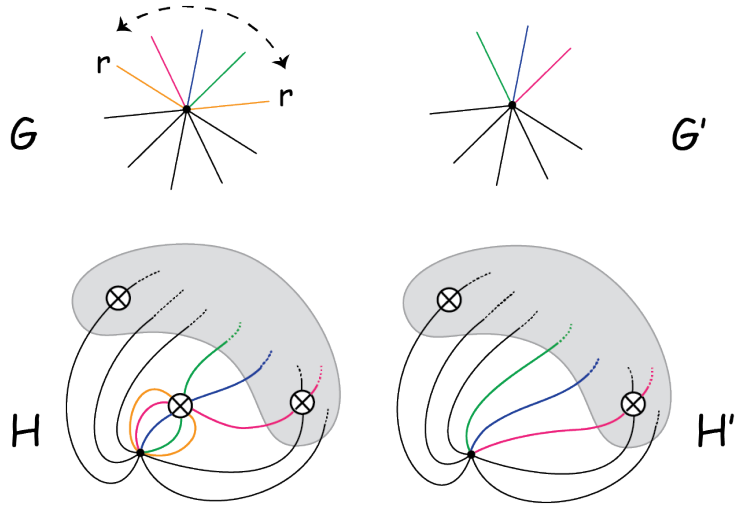

Figure 5.9: The one-sided loop move on the loop r.

Concatenation move. Let o be an orienting loop in the embedding scheme G such that one of its ends is immediately followed by an end of a two-sided non-separating loop t in the cyclic permutation around the vertex. By Lemma 5.2.3, since t is non-separating, the concatenation of o and t which we denote by o', is not orienting. Denote by G' the embedding scheme in which we replace o by o' (we need $eg(G)$ cross-caps to draw both G and G'). If H' is a drawing for G', one can obtain from H' a drawing for G by replacing the drawing o' by its concatenation with t. Depending on the wedge of o' that we choose to flip, we slide o' along t in the drawing:

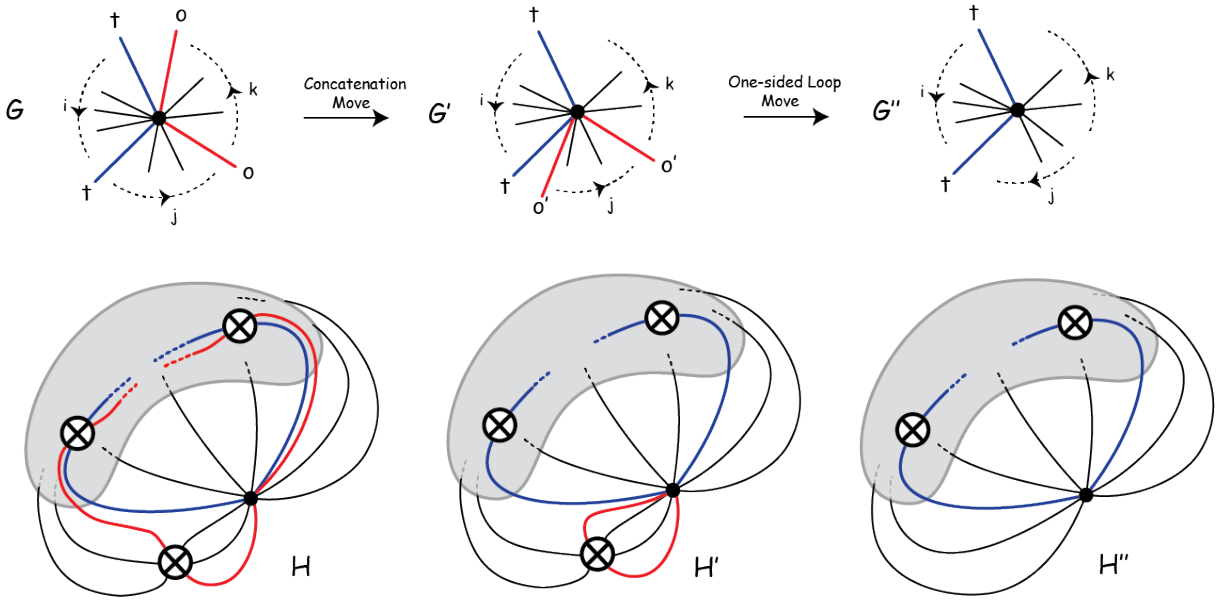

Figure 5.10: The concatenation move on the loops o and t, when in applying the one-sided loop move we flip the wedge of o' that does not encompass the ends of the loop t.

If we flipped the wedge that does not encompass the loop t, we detach the end of o' next to t and slide it along t and we attach it to the vertex. This way, it ends up where the end of o was placed originally; see Figure 5.10.

If we flipped the wedge that encompasses the loop t, we do as follows: the loop o' passes through only one cross-cap. We draw o next to the end of o' that is not slid along t, but instead of following o' into the cross-cap, we follow t. We can do this because the loop o' is next to the loop t in the rotation around this cross-cap; see Figure 5.11.

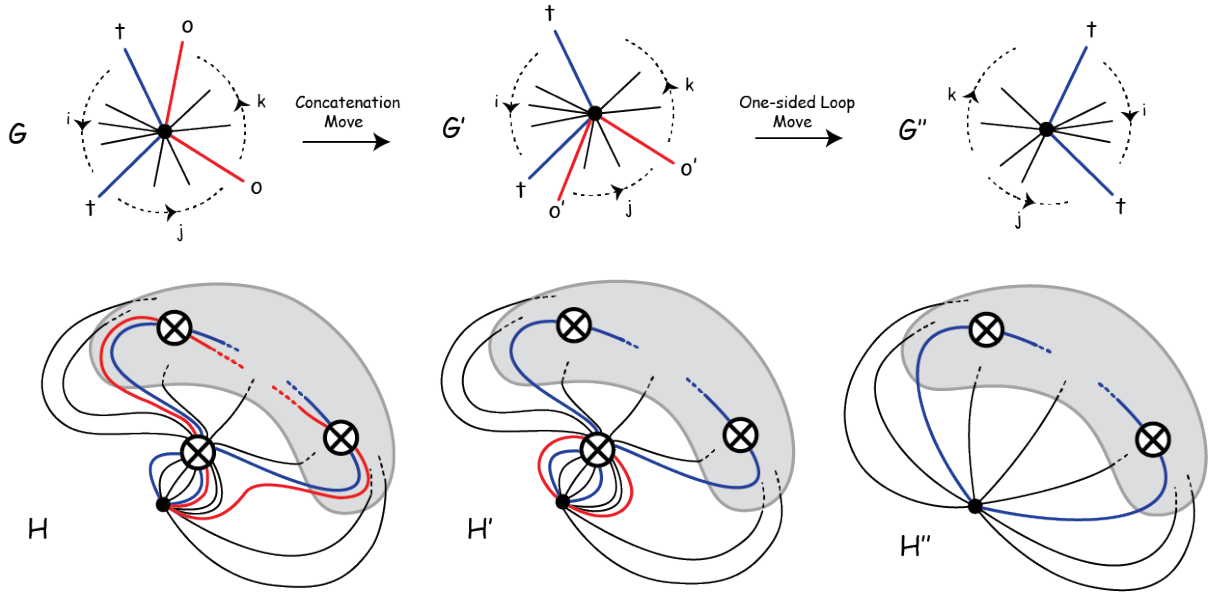

Figure 5.11: The concatenation move on the loops o and t, when in applying the one-sided loop move we flip the wedge of o' that encompasses the ends of the loop t.

Exchange move. Let G be an embedding scheme that has only two-sided loops and no separating loop. We exchange two consecutive half-edges and change the signatures of the corresponding edges. One can prove that the new embedding scheme can be drawn using the same number of cross-caps as the initial embedding scheme. Having a drawing of the new embedding scheme, we obtain a drawing of the original embedding scheme by adding a cross-cap near the reversed half-edges. This drawing uses $eg(G) + 1$ cross-caps and this is the minimum number of cross-caps that we need to draw an orientable scheme (Lemma 3.3.2).

Each of these moves provides a way to draw a loop assuming that some simpler one-vertex graph without that loop has already been drawn. Therefore, we can use the moves in an inductive algorithm as follows. The input is an embedding scheme G.

The Schaefer-Štefankovič algorithm:

- **Step 1: If there exists a contractible loop.** We recurse on the embedding scheme without the loop and apply the contractible loop move.
- **Step 2. If there exists a separating (non-contractible) loop s.** We divide the embedding scheme into two sub-schemes on each side of s. We have the following cases:
  - **Step 2.1: Both sub-schemes are non-orientable.** We recurse on the two sub-schemes and apply the gluing move.
  - **Step 2.2: At least one of the sub-schemes is orientable.** We recurse on the two sub-schemes and apply the gluing move followed by the dragging move.
- **Step 3.1: If there is a one-sided non-orienting loop r.** We recurse on the embedding scheme without the loop r and the flipped wedge, and apply the one-sided loop move to the loop r.
- **Step 3.2: If all one-sided loops are orienting.** If there exists no two-sided loop, by Lemma 5.2.3, all pairs of loops are interleaving and we can draw the embedding scheme with one cross-cap so that each loop enters it once. If there exists a two-sided loop in the embedding scheme, we can find a place in the embedding scheme where an orienting loop o is followed immediately by a two-sided loop t. We recurse on the embedding scheme H' described in the concatenation move, and apply the concatenation move to the curve o.
- **Step 3.3: If every loop in the embedding scheme is two-sided.** We apply the exchange move to two consecutive half-edges and recurse on the new embedding scheme.

The proof of Schaefer and Štefankovič for Theorem 5.3.1 proceeds by analyzing this algorithm and proving that (1) the resulting cross-cap drawing has the correct number of cross-caps and (2) each loop passes through every cross-cap at most twice.

Remark 5.3.3. *By Lemma 3.4.2, in a drawing that is obtained by this algorithm, every orienting loop passes through each cross-cap exactly once and if a separating loop enters a cross-cap, it passes through that cross-cap exactly twice.*

There is some leeway in this algorithm: while the steps have to be applied in this specific order, in each step a loop of the given type is chosen arbitrarily. Our modification

of the algorithm follows the exact same blueprint but enforces specific orders in which we choose separating and one-sided non-orienting loops. These specific orders provide more structure to the resulting drawing, make it lend itself more to connecting the cross-caps, in order to form the desired non-orientable canonical system of loops of low multiplicity.

5.4 Our modification to the Schaefer-Štefankovič algorithm

Preprocessing. Our algorithm first starts with some preprocessing.

Lemma 5.4.1. *Given a graph G embedded on a non-orientable surface N, there exists a one-vertex scheme $\hat{G}$ such that $\hat{G}$ has an orienting loop, and if $\hat{G}$ has a non-orientable canonical system of loops such that each loop crosses each edge of $\hat{G}$ at most k times, then G has a non-orientable canonical system of loops such that each loop crosses each edge of G at most $3k$ times.*

Proof. By Lemma 3.7.1, there exists an orienting curve γ embedded on the surface N that has multiplicity two. Denote by G' the overlay of G and γ, that is, the graph obtained by superimposing G and γ and adding dummy vertices of degree four at the crossings. Each edge in G is divided into at most three sub-edges in G'. If γ crosses G, n times totally, then it corresponds to n edges in G'. We choose a spanning tree T in G' that contains $n-1$ of these edges. Without loss of generality, we can assume that all the signatures of the edges of T are $+1$ (this is because one can apply local changes to the embedding scheme without changing its homeomorphism class, see, e.g., [58, Section 4.1]). We contract the edges of T into a single point to get a one-vertex graph $\hat{G}$ and merge the cyclic permutations around the vertices in the embedding scheme to obtain an embedding scheme for the one-vertex graph $\hat{G}$. The non-contracted sub-edge of γ is an orienting loop in $\hat{G}$.

Let us assume that $\hat{G}$ has a non-orientable canonical system of loops such that each loop crosses each edge of $\hat{G}$ at most k times. We can uncontract the spanning tree T close to the vertex so that the uncontracted edges do not cross the canonical system of loops. This gives us a canonical system of loops for G'. To get a drawing for G, we remove the orienting curve γ. Since any edge of G is formed by at most three edges of G', the canonical system of loops for G' crosses each edge of G at most $3k$ times □

Remark 5.4.2. *The curve γ divides each edge of G into at most three sub-edges. Therefore, each edge in G is obtained as the concatenation of at most 3 edges of $\hat{G}$.*

This lemma lets us assume, at the cost of a slightly bigger multiplicity, that (1) the input graph is a one-vertex graph endowed with an embedding scheme, and (2) the embedding scheme that we work with contains an orienting loop. This orienting loop will in turn allow us to avoid some steps in the algorithm.

For the second prepossessing move we need a definition that is inspired by a similar notion from the literature on sorting signed permutations by reversals [44] (see Section 3.8.2 in the Preliminaries for a description). Given an embedding scheme G, the *interleaving graph* I_G has as vertex set the set of loops of G, and two vertices are connected if their corresponding loops have interleaving ends, see Figure 5.13 for an example. When we talk about the sidedness of a vertex, we mean the sidedness of the loop it is associated to. A connected component in the interleaving graph is called *non-orientable* if it has a one-sided vertex, and *orientable* otherwise. We call a component with only one-vertex a *trivial* component, and *non-trivial* otherwise. Separating loops (contractible or non-contractible) correspond to isolated two-sided vertices, i.e., trivial orientable components, in the interleaving graph.

Our second preprocessing step aims at subdividing G into sub-schemes G_i such that each I_{G_i} has only one non-trivial component. In order to do this, we *saturate* the embedding scheme with auxiliary separating loops, i.e., we add a separating loop for any non-trivial component that is not divided from the rest of the graph by some separating loops.

Remark 5.4.3. *If $\bar{G}$ is a saturated extension of G, then $eg(G) = eg(\bar{G})$ by Lemma 3.2.3. Thus a minimum genus drawing of $\bar{G}$ contains a minimum genus drawing of G.*

Given a non-orientable scheme G saturated with separating loops and any cellular embedding of G on a surface N, cutting G along the separating loops yields subsurfaces N_i of N, each containing (possibly empty) components of G, which we denote by G_i (see Figure 5.12). The *component tree* of G has a vertex for every such subgraph G_i, and two vertices are connected if their corresponding components are separated by a separating loop. See Figure 5.12 for an example of a component tree. We shall quickly see that in the context of our algorithm, there will actually be exactly one non-orientable component. We root the tree at the vertex corresponding to the non-orientable component.

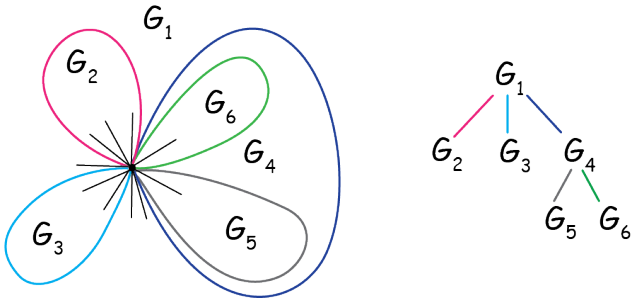

Figure 5.12: A saturated one-vertex scheme (left) in which the drawn loops are the separating loops and its component tree (right). The component G_4 is an empty sub-scheme.

In a saturated embedding scheme, each separating loop relates two subgraphs G_i that exist on its different sides. If we have k non-trivial components or empty subgraphs in

total in the interleaving graph, in order to separate all of them from each other, we need exactly $k-1$ separating loops.

We are now ready to describe our modified algorithm in which we force the presence of an orienting curve and we put more structure on the order we deal with loops at each step.

The modified algorithm:
Pre-processing steps:

- **Step A.** If there is no orienting loop, we add an orienting loop and contract a spanning tree using Lemma 5.4.1.
- **Step B.** If G is not saturated by separating curves, we saturate it.

Main loop:
Throughout the main loop of our algorithm, if we have homotopic loops we remove all except one. We draw this loop then re-introduce the other loops parallel to it.

- **Step 1: If there is a contractible loop.** We recurse on the embedding scheme without the loop and apply the contractible loop move.
- **Step 2: If there exists a separating (non-contractible) loop.** We pick a separating loop that separates a non-root leaf from the component tree, recurse on the sub-schemes and apply a gluing and a dragging move.
- **Step 3.1: If there exists a one-sided non-orienting loop.** We pick a one-sided non-orienting loop such that the embedding scheme G' that we obtain when removing it and flipping its wedge maximizes the number of one-sided loops. We recurse on G' and apply the one-sided loop move to this loop.
- **Step 3.2.a: If all one-sided loops are orienting and there are two-sided loops.** We pick an orienting loop adjacent to a two-sided loop, recurse on the drawing H' described in the concatenation move and apply this move on these loops.
- **Step 3.2.b: If all one-sided loops are orienting and there are no two-sided loops.** In this case one cross-cap is sufficient to draw all the loops.

Post-processing steps:

- **Step B'.** Erase the extra separating loops added in step B.
- **Step A'.** Uncontract the spanning tree and remove the added loop in Step A.

The numbering in this algorithm comes from the Schaefer-Štefankovič algorithm. A main difference with our modified version is that it is not clear at first sight that we cover all cases as we do not have the steps 2.1 and 3.3 in the Schaefer-Štefankovič algorithm. Yet we do cover all the cases: this follows from the presence of an orienting loop and proving it will be the main purpose of Sections 5.4.1 and 5.4.2. After that we will be ready to prove in Lemma 5.4.12 that our algorithm terminates and outputs a cross-cap drawing where a loop does not enter each cross-cap more than 6 times.

5.4.1 Completeness of the case analysis

The following lemma explains why we do not need to consider the case in which all loops are two-sided.

Lemma 5.4.4. *An embedding scheme with an orienting loop is non-orientable.*

Proof. Let us assume that G is an orientable scheme, hence the orienting loop o is two-sided. By Lemma 3.3.2, G needs $eg(G)+1$ cross-caps to embed and we know that $eg(G)$ is twice the orientable genus of G. Therefore, G needs an odd number of cross-caps to embed and the loop o passes through each of them an odd number of times (Lemma 3.4.2). Thus, o is one-sided which is a contradiction. □

Lemma 5.4.4 guarantees that we do not need to consider the case where all loops are two-sided when the algorithm starts, but this case might a priori still happen during recursive calls to the algorithm. Fortunately, this will actually not be the case, as we will prove in Corollary 5.4.10 that there is always an orienting loop in each of the recursive calls.

The following lemma explains why there is only one case that can happen in step 2 of our algorithm.

Lemma 5.4.5. *Let G be an embedding scheme with an orienting loop o and a non-contractible separating loop s. The loop s separates the graph into an orientable and a non-orientable subgraph.*

Proof. By Lemma 5.4.4, G is non-orientable and therefore it has at least one one-sided loop. Let us assume that s separates the embedding scheme into G_1 which contains the loop o, and G_2. We show that G_1 is non-orientable and G_2 is orientable.

By Lemma 5.2.3, any one-sided loop has exactly one end in the wedge of the orienting loop in G_1 and has to have both ends in the same side of the separating loop, therefore no one-sided loop can exist in G_2 so G_1 is non-orientable and G_2 is orientable. The case where the only one-sided loop is the orienting loop is trivial.

□

5.4.2 The order on the one-sided non-orienting loops

In this section, we explain an order on one-sided loops when we apply the one-sided loop move in step 3.1 of the algorithm. The reason for imposing this restriction is to avoid creating separating loops in the induction that did not exist in the embedding scheme initially.

Lemma 5.4.6. *If there exists an orienting loop o (two-sided or one-sided) in the embedding scheme G, the connected component that has the vertex o is the only non-orientable component in I_G.*

Proof. By Lemma 5.2.3, the ends of every one-sided loop interleave with the ends of the orienting loop. Therefore, the vertex o is connected to every one-sided vertex in I_G and this finishes the proof; see Figure 5.13. □

The following lemma is analogous to a similar result in signed reversal distance theory [6, Fact 2][3].

Lemma 5.4.7. *Applying a one-sided loop move to a loop r corresponds to removing the vertex r in the interleaving graph and complementing the subgraph induced by its neighbors.*

Proof. In a one-sided loop move, we remove a one-sided loop r and flip one of its wedges. When we flip, we change the signature of each loop that has exactly one end in each wedge of r. Thus, we are changing the sidedness of everything that was connected to the vertex r in the interleaving graph. Also, due to the flip, every two vertices adjacent to r, that were connected to each other before the flip, are now disconnected and vice versa. The situation of the loops that are not interleaving with r in the embedding scheme is unchanged; see Figure 5.13. □

Such a move can change the number of connected components in the interleaving graph, and might increase the number of orientable components, as is the case when we apply the one-sided loop move to o in Figure 5.13.

Lemma 5.4.8. *Let G be an embedding scheme with orienting loop o and a one-sided non-orienting loop r. Let G' be the graph we obtain after removing r and flipping its wedge. The loop o is orienting in G'.*

Proof. We need to show that after removing r, the loop o is connected to all the one-sided vertices in the interleaving graph and it is not connected to any two-sided vertex (Lemma 5.2.3). We know that at the start o is connected to r. Any one-sided loop that r used to be adjacent to is now two-sided and since o used to be connected to these loops,

[3]The exact same property has been used in the proof of Lemma 3.8.5.

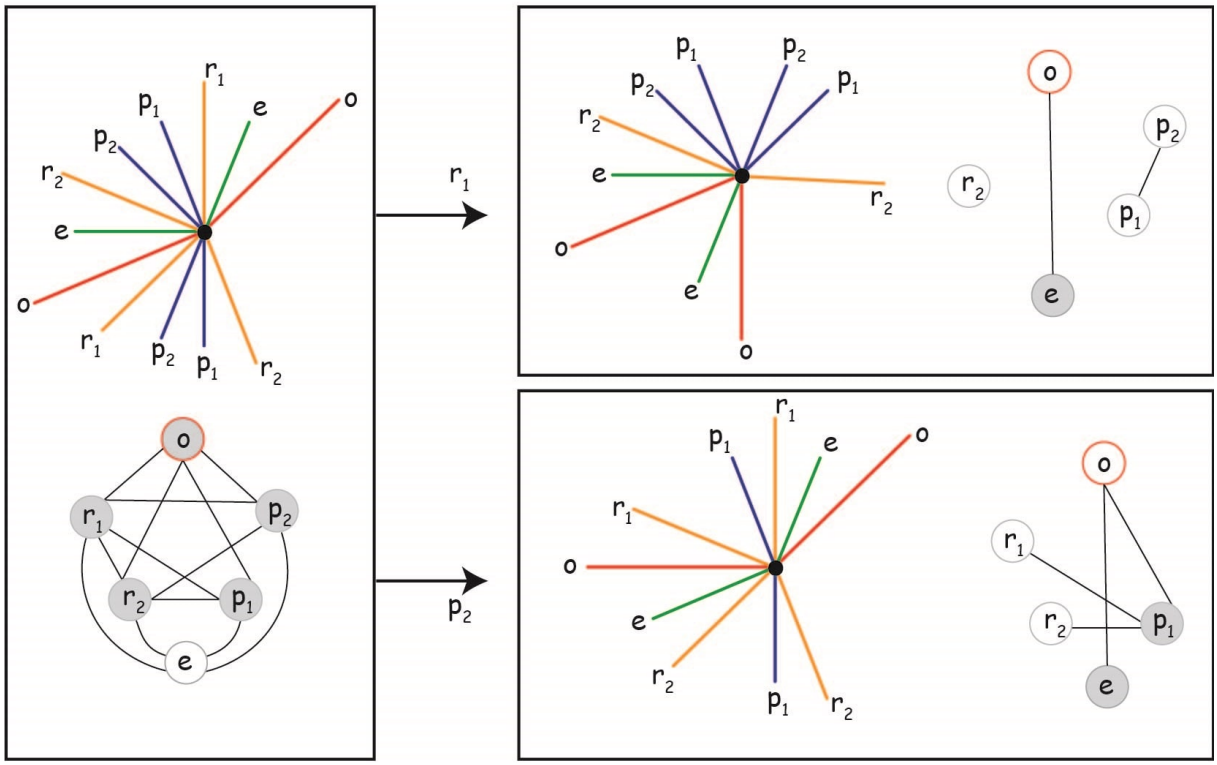

Figure 5.13: Each box represents an embedding scheme with its interleaving graph. In all these embedding schemes the loop o is orienting; right: the impact of applying the one-sided loop to r_1 (top) and p_2 (bottom).

by complementing the subgraph induced by the vertices adjacent to r (Lemma 5.4.7), o is no longer connected to them. Similarly, we can see that if r used to be connected to two-sided loops, after flipping the wedge, they become one-sided and since the loop o was not connected to any two-sided vertex before, now it gets connected to them. The situation for the loops to which o was connected and r was not, remains unchanged. □

Lemma 5.4.9. *If G is an embedding scheme that contains only orienting and non-separating two-sided loops, and has at least one of each, then there exists an orienting loop o followed by a non-separating two-sided loop t. Denote by o' a loop homotopic to the concatenation of o and t. If G' is the embedding scheme obtained by replacing o by o', and G'' is the embedding scheme obtained by applying a one-sided loop move to o' in G', then t is orienting in G''.*

Proof. Since there exist only orienting and two-sided non separating loops, and there is at least one of each, there exists an orienting loop o and a non-separating two-sided loop t such that an end of o is consecutive to an end of t in the embedding scheme.

First we show that t and o belong to different components in I_G. Since o is orienting and there is no one-sided non-orienting loop in the embedding scheme, the component

of o is a complete graph with only orienting loops. Therefore, t does not belong to this component and everything in the component of t is two-sided. Replacing o by o' corresponds to replacing the vertex o' with o in I_G and connecting it to the neighbors of t, since o' now interleaves with every loop that t interleaves with. Now applying the one-sided loop move to o' makes every neighbor of t one-sided and all the orienting vertices (that were formerly adjacent to o) two-sided. Therefore, the only one-sided loops in the new embedding scheme are the neighbors of t and t is not adjacent to any two-sided vertex since everything in its component used to be two-sided. By Lemma 5.2.3, t is orienting in G''. □

Since the contractible loop move clearly preserves orienting loops, Lemmas 5.4.5 (the dragging move first adds an orienting loop to the orientable part), 5.4.8 and 5.4.9 imply the following corollary.

Corollary 5.4.10. *Let G be a one-vertex scheme with an orienting loop. Let G' be the graph on which the modified algorithm recurses when applying one of the following moves:*

- *A contractible loop move.*
- *A one-sided loop move on a one-sided non-orienting loop.*
- *The concatenation move on an orienting loop.*

Then G' has an orienting loop. Likewise, when the modified algorithm applies a gluing and dragging move to a separating loop s, the two subgraphs G_1 and G_2 on which it recurses have an orienting loop.

The next lemma explains our choice of rule in Step 3.1:

Lemma 5.4.11. *Let G be a one-vertex scheme with an orienting loop and no non-contractible separating loop such that I_G has only one non-trivial component. The embedding scheme G can be drawn by applying a sequence of contractible loop moves, one-sided loop moves and concatenation moves. Specifically, we do not use the gluing move, the dragging move and the exchange move.*

This ensures that in Step 3.1 no non-contractible separating loop is created during the process, hence we can avoid increasing the number of orientable components. This proof mirrors results in the signed reversal distance theory (see Bergeron [6, Theorem 1] or Lemma 3.8.5)) in which similar claims are proved in the context of applying reversals to permutations.

Proof. In order to have a non-contractible separating loop, it is necessary to have either two non-trivial components, or an orientable non-trivial component and a trivial non-orientable component (an isolated one-sided vertex).

Since there exists an orienting loop in the embedding scheme and the ends of the orienting loop interleave with those of every one-sided loop, all one-sided loops belong to the same component in the interleaving graph and thus there exists only one non-orientable component. By Corollary 5.4.10, in applying these moves, there is always an orienting loop in every step. This, together with Lemma 5.4.6, implies that the number of non-orientable components remains 1 through the algorithm. Therefore, it suffices to prove that we can draw G such that in each step we do not increase the number of non-trivial orientable components.

The non-trivial component is non-orientable. If there exists no one-sided non-orienting loop, then every loop in the embedding scheme is orienting or contractible. In this case, the non-orientable genus is one and the result is trivial. Let us assume that there exists at least a one-sided non-orienting loop. We claim that if we choose the one-sided non-orienting loop such that flipping its wedge maximizes the number of one-sided loops[4], then we do not increase the number of orientable components. In Figure 5.13, it is shown that applying a one-sided loop move to p_2 maximizes the number of one-sided loops but applying it to r_1, increases the number of orientable components and turns r_2 into a separating loop for the new embedding scheme.

Let us assume that applying a one-sided loop move to the one-sided loop r maximizes the number of one-sided loops and increases the number of non-trivial orientable components. Let i be a vertex that was adjacent to r, belonging to a component U that got disconnected when complementing the subgraph induced by the neighbors of r. The vertex i was one-sided and we claim that taking i instead of r would have created more one-sided vertices which is a contradiction.
Denote by $\#_1(r)$ (resp. $\#_2(r)$) the number of one-sided (resp. two-sided) loops adjacent to r. Removing a one-sided loop is equivalent to removing the vertex in the interleaving graph and complementing the subgraph induced by its adjacent vertices. Therefore, removing r increases the number of one-sided loops in the embedding scheme by $\#_2(r) - \#_1(r)$. All two-sided vertices adjacent to r are one-sided after removing r and therefore they are not in U, meaning that they were formerly connected to i, so we have $\#_2(i) \geq \#_2(r)$.
Similarly, r has to be connected to every one-sided vertices that were formerly connected to i, so we have $\#_1(r) \geq \#_1(i)$.
If $\#_2(i) = \#_2(r)$ and $\#_1(r) = \#_1(i)$, then they have the same neighbors and removing r will isolate i which contradicts the fact that the connected component U is not trivial. Therefore, applying a one-sided loop move to the loop i creates more one-sided loops than applying a one-sided loop move to the loop r, which is a contradiction. Thus, removing r cannot add to the number of non-trivial orientable components.

[4]This is analogous to choosing a reversible pair with maximal score in sorting a permutation by reversals, explained in Section 3.8.1

The non-trivial component is orientable. In this case, there exists only one one-sided loop o which is orienting. We replace o with its concatenation with one of the two-sided loops that is immediately next to it in the cyclic permutation around the vertex. Denote this two-sided loop by t and the concatenated loop by o'. This corresponds to replacing the vertex o by o' that is connected to all the neighbors of t in the interleaving graph and therefore reduces the number of components by 1. Applying the one-sided loop move to o', isolates t and makes it orienting for the new embedding scheme (Lemma 5.4.9). The resulting graph falls in the last case where the non-trivial component is non-orientable and therefore we can draw it by applying only contractible loop moves and one-sided loop moves. This completes the proof. □

5.4.3 Correctness of the modified algorithm

Lemma 5.4.12. *Let G be a graph cellularly embedded on a non-orientable surface. If G has an orienting loop, applying the modified algorithm, we obtain a cross-cap drawing of G with $eg(G)$ cross-caps such that each loop of G enters each cross-cap at most twice. Otherwise, we obtain a cross-cap drawing of G with $eg(G)$ cross-caps such that each loop of G enters each cross-cap at most 6 times.*

Proof. By Lemma 5.4.1, Step A in the algorithm reduces the graph G to a one-vertex scheme $\hat{G}$ that has an orienting loop such that a drawing $\hat{G}$ leads to a drawing for G. Let $\bar{G}$ be the embedding scheme that we obtain after step B on $\hat{G}$. This step does not change the Euler genus of the embedding scheme.

Thus, by Remark 5.4.2, it is sufficient to prove that there is a cross-cap drawing for $\bar{G}$ with $eg(G) = eg(\bar{G})$ cross-caps in which each edge passes through each cross-cap at most twice. We prove this claim for any one-vertex scheme G with an orienting loop.

In order to prove this claim, we follow the recursive steps of the main loop of the modified algorithm, using induction on $eg(G) + |E(G)|$.

Step 1. We apply the contractible loop move to every contractible loop. By the induction hypothesis, we can obtain a drawing with $eg(G)$ cross-caps for the resulting embedding scheme in which every loop passes through each cross-cap at most two times and the contractible loop does not use any of them.

Step 2. If there exists separating (non-contractible) loops, we deal with them in the prescribed order. Take the separating loop s and divide the embedding scheme. By Lemma 5.4.5, we know that one of these subgraphs is orientable and the other one is non-orientable, without loss of generality let us assume that G_1 is non-orientable. We apply a combination of the gluing move and the dragging move to these subgraphs: we add an auxiliary orienting loop o to G_2. By the induction hypothesis, there are cross-cap

drawings H_1 with $eg(G_1)$ cross-caps and a drawing H_2 with $eg(G_2)+1$ cross-caps for G_1 and $G_2+\{o\}$ so that each edge of G_1 and G_2 passes through each cross-cap at most twice. Let H_2' be the drawing for $G_2+\{o\}+\{s\}$ that we obtain as described in the dragging move. By Remark 5.3.3 and by the induction hypothesis, we know that in H_2, the loop o passes through each cross-cap exactly once. The loop s follows o twice and therefore it passes through each cross-cap in H_2 exactly twice.

The gluing move does not interact with the number of entrances for any loop. In the dragging move, every loop that is being dragged from H_1 to H_2 follows the auxiliary orienting loop o in G_2. Since each edge in H_1 passes through each cross-cap at most twice, at most two of its sub-edges are being dragged along o and therefore they pass through the cross-caps in H_2 at most twice.

Since our graph is non-orientable, and we have only dealt with two-sided loops so far, not all of the edges can be orientable at this point.

Step 3.1. If G has one-sided non-orienting loops, we apply a one-sided loop move to the one that respects our prescribed order; we denote this loop by r. Since r is non-orienting, the embedding scheme G' obtained after removing r and flipping its wedge is still non-orientable and by the induction hypothesis, there is a drawing H' with $eg(G)-1$ cross-caps for G' in which every loop passes through each cross-cap at most twice. After drawing r and the new cross-cap, we can see that each loop in the flipped wedge passes through this cross-cap at most twice (the exact value depends on the number of its ends within the flipped wedge).

Step 3.2.a We can find an orienting loop o that is followed immediately by a two-sided loop t. We apply the concatenation move to o and replace it with the non-orienting loop o' (denote by G', the embedding scheme we obtain at this point; we have $eg(G')=eg(G)$). We apply the one-sided loop move to o'. By the induction hypothesis, there is a drawing H' for G' with $eg(G)$ cross-caps in which each loop passes through each cross-cap at most twice. Since we first apply a one-sided loop move to o', we can see that o' uses only one cross-cap. Depending on our choice in flipping a wedge of o', the loop t uses this cross-cap either twice or not at all (the loop t uses every cross-cap except this cross-cap exactly once, since after removing o', t gets orienting, see Lemma 5.4.9). One can check that at the end for both choices of wedge, the loop o passes through each cross-cap exactly once.

Step 3.2.b If all one-sided loops in the embedding scheme are orienting and there is no two-sided loop in the embedding scheme, all of the orienting loops in the embedding scheme are homotopic and we can draw them using one cross-cap with all of the loops passing though the cross-cap exactly once.

By Corollary 5.4.10, the graph has an orienting loop at each step of the algorithm and therefore by Lemma 5.4.5, we never have a graph in which every loop is two-sided

throughout the algorithm. This completes the proof of the claim and we conclude. □

This proof is independent of the orders we defined for the loops in Step 2 and Step 3.1. These orders are shown to be useful in the next section.

In addition to the fact that our proof does not cover all the steps that happen in the original case (the case that there might not exist an orienting loop in the embedding scheme), another difference between this proof and the proof of the Schaefer and Štefankovič algorithm is in Step 3.2. In this Step, the original algorithm flips the wedge of o' that does not encompass the loop t. We prove the step for both choices of wedge because we favour the freedom to choose a wedge that we want to flip for our further purposes.

5.5 The non-orientable canonical system of loops

The modified algorithm that we described in the previous section provides us with a cross-cap drawing of any embedded graph G where each edge of the graph enters each cross-cap at most six times, as per Lemma 5.4.12. Furthermore, our algorithm has the following key advantage compared to the algorithm of Schaefer and Štefankovič: due to the order in which we choose the loops in Steps 2 and 3.1, we know that dragging moves and the other moves do not intermingle during the recursive calls of the algorithm. When the algorithm draws an embedding scheme with a single non-trivial component, it only relies on contractible, one-sided and concatenation moves. Second, due to the order in which we choose the loops in Steps 2 and 3.1, we know that whenever a dragging move is applied, the orientable sub-scheme on which we recurse has only one non-trivial component. In this section, we leverage these two key advantages to find a non-orientable canonical system of loops of small multiplicity.

5.5.1 The dual graph of the cross-cap drawing

In this rather tedious but straightforward section, we first investigate the effect of every move involved in the modified algorithm on the dual graph of the cross-cap drawing (viewed as a planar graph). Every edge e in a cross-cap drawing H, corresponds to an edge e^* and every face F corresponds to a vertex F^* in the dual graph. The vertex v corresponds to the face v^* and the cross-caps correspond to the other faces in the dual graph. See Figure 5.14 for an example of a cross-cap drawing and its dual. In this section, cross-caps are denoted by $\mathfrak{u}$, $\mathfrak{b}$, $\mathfrak{c}$, $\mathfrak{k}$ and $\mathfrak{g}$ and the letter F is reserved for faces in a cross-cap drawing.

By Remark 5.3.3, in the drawing obtained via the modified algorithm, the orienting loop passes through each cross-cap exactly once. Thus if G is drawn using k cross-caps, an orienting loop in G, corresponds to a set of $k+1$ edges in the dual graph. Furthermore,

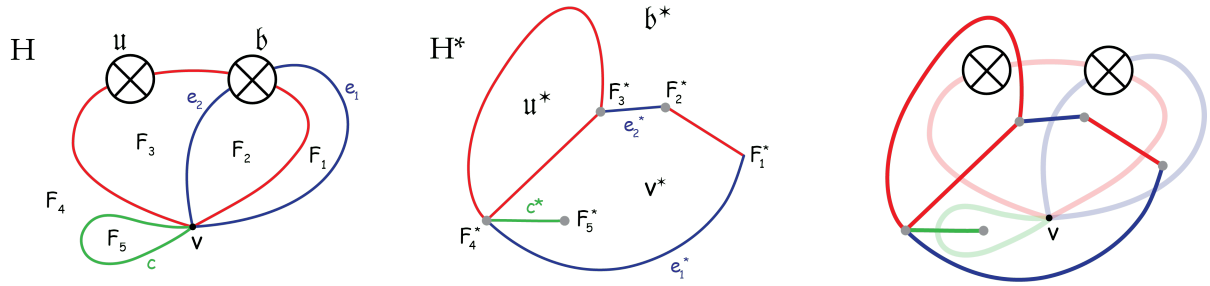

Figure 5.14: H is a cross-cap drawing for an embedding scheme G and H^* is the dual graph to H. The rightmost picture shows the overlay of H and H^*. The red loop in H is orienting and the red edges in H^* correspond to segments of this orienting loop. The faces $\mathfrak{b}^*, \mathfrak{u}^*$ and v^* in the dual, each has exactly two edges corresponding to the orienting loop.

there are exactly 2 dual edges corresponding to the orienting loop in each face of the dual, see Figure 5.14.

The effect of a contractible loop move is as follows:

Lemma 5.5.1. *Drawing a contractible loop in a face F of the cross-cap drawing corresponds to adding a vertex with degree one attached to the vertex F^*.*

Proof. The proof follows directly from the definition of graph duality. Figure 5.14 depicts the edge c^*, dual to the contractible loop c. □

We can see that applying a contractible loop move does not change the situation of any of the dual edges corresponding to the orienting loop.

Let us assume there is a loop s that separates G into G_1 and G_2, where G_1 is non-orientable and G_2 is orientable. We glue the drawings H_1 and H_2 for G_1 and G_2 and we apply a dragging move to this case. The following lemma explains the effect of this move on the dual graph. In this lemma, we use the notation introduced in the description of the dragging move. We denote the vertex associated to the root face F_o^i in H_i by F_o^{i*} for $i \in \{1, 2\}$. We use the notation $(i_1, F_1, \ldots, i_{2K}, F_o^1)$ for the sequence of edges and faces around the eliminated cross-cap in the dragging move, and finally we denote by o the auxiliary orienting loop drawn in H_2.

Lemma 5.5.2. *Let s be a loop that separates the embedding scheme G into the non-orientable subgraph G_1 and the orientable subgraph G_2. In this case, the gluing move, the dragging move and drawing back the separating loop corresponds to:*

- *splitting F_o^{1*} into two vertices F_o^{11*} and F_o^{12*} such that F_o^{11*} inherits only i_1^* and F_o^{12*} inherits the rest of the edges incident to F_o^{1*},*
- *removing the two dual edges corresponding to the loop o incident to the vertex F_o^{2*} in v_2^* (o_1^* and o_2^* in Figure 5.15),*

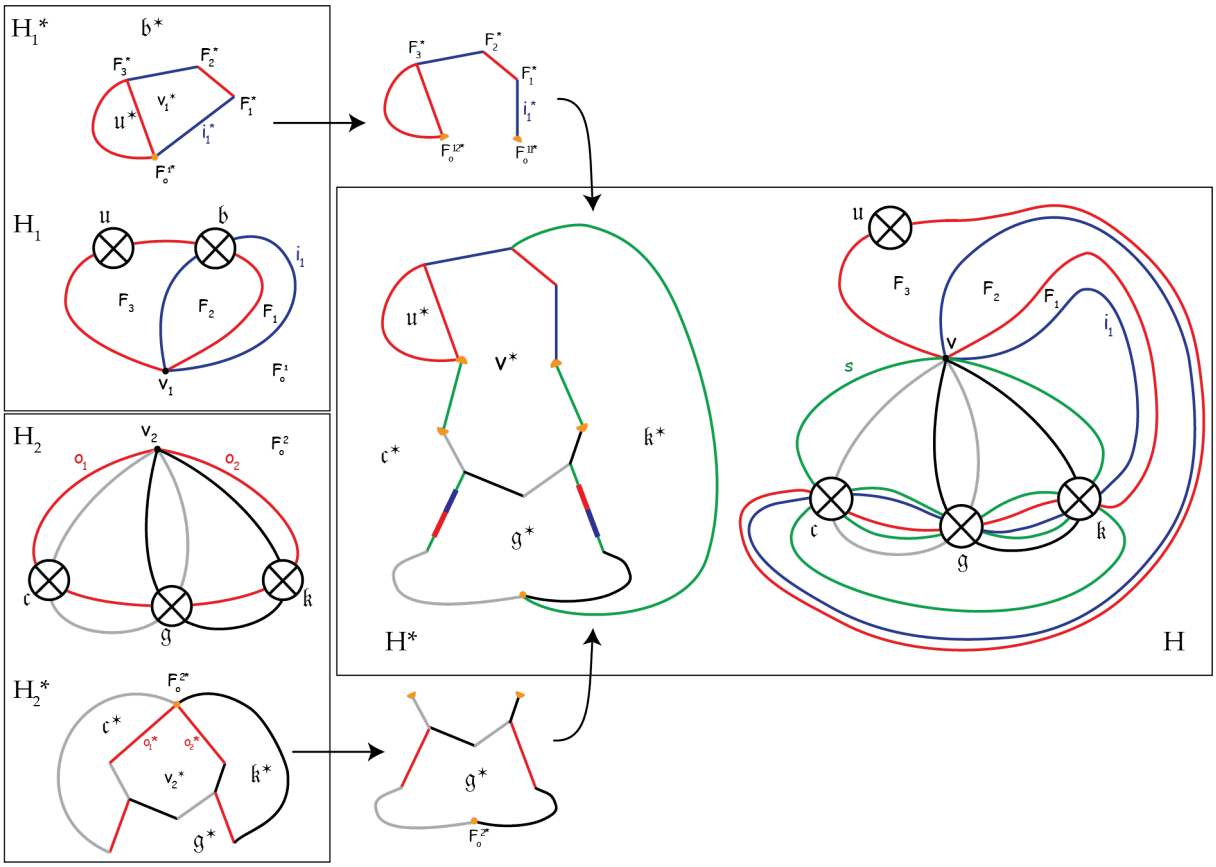

Figure 5.15: The impact of the gluing move and dragging move on the dual graph.

- *connecting* F_o^{11*} *and* F_o^{12*} *to the adjacent vertices to* o *in the correct order by adding an edge for each one (this edges correspond to the segments of the separating loop that are attached to the vertex),*
- *connecting* F_o^{2*} *to* F_k^* *by adding an edge,*
- *replacing the dual edges corresponding to the segments of* o *in* H_2 *by* $k+2$ *edges.*

The operations performed in Lemma 5.5.2 are pictured in Figure 5.15.

Proof. Splitting F_o^{1*} and connecting it to two vertices formerly incident to F_o^{2*} corresponds to the gluing move between the drawings. We can see in Figure 5.15 that these steps merge the faces v_1^* and v_2^* to the face v^*. The edges that we add to connect these vertices correspond to the sub-edges of the separating loop s that are incident to the vertex. On the other hand, connecting F_o^{2*} to F_k^* and replacing the dual edges in H_2 by a path corresponds to the dragging move; see Figure 5.15.

□

Remark 5.5.3. *Let us denote by* H *the drawing of* G *that we obtain by applying the three moves in Lemma 5.5.2. The only modification done on the subgraph induced by the vertices*

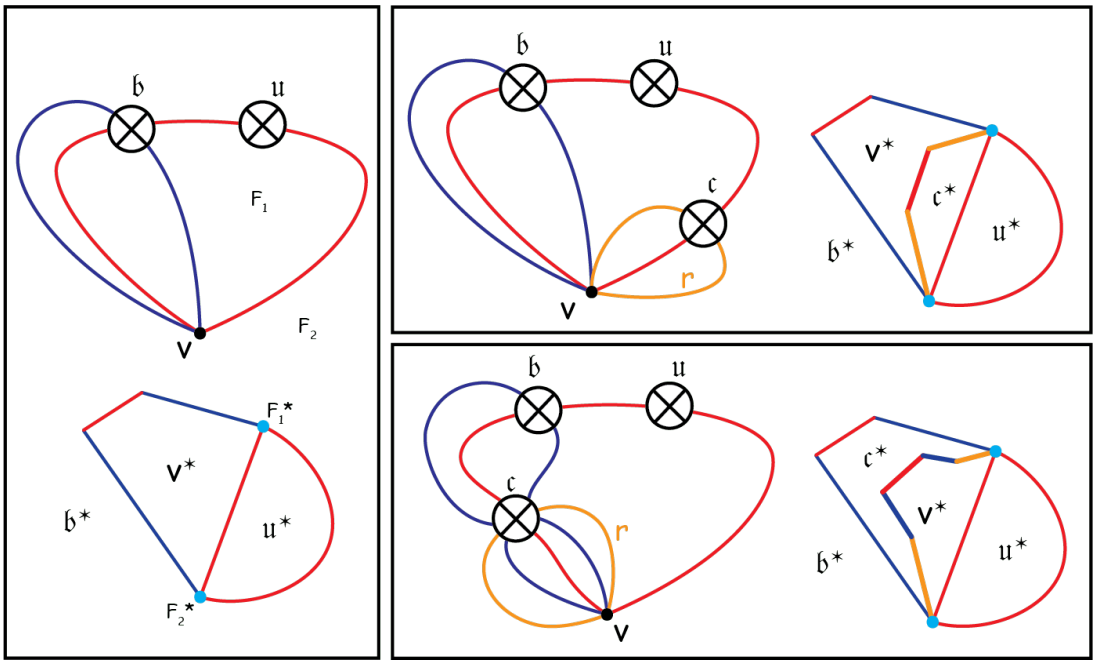

Figure 5.16: The impact of the one-sided loop move on the dual graph. Left: the drawing for the embedding scheme before adding the one-sided loop r and its dual; right: the drawings and the dual graphs after drawing r and the impact of flipping different wedges of r is depicted.

that come from H_1^* in H^* is that we split the vertex F_o^{1*} to F_o^{11*} and F_o^{12*}. Both F_o^{11} and F_o^{12} are root faces in G.

The modifications in the subgraph induced by the vertices that come from H_2^* is that we replace some edges by paths and we disconnect the two root faces adjacent to F_o^{2*} by removing the incident edges. Also F_o^{2*} gets connected to a face in H_1^* and it is not necessarily a root face anymore.

The effect of a one-sided loop move is as follows:

Lemma 5.5.4. *Adding a one-sided non-orienting loop r with ends in root faces F_1 and F_2 in the drawing (F_1 and F_2 are possibly identical), corresponds to subdividing the face v^* into two faces and adding a path of length $k+2$ from F_1^* to F_2^* where k is the length of one of the paths from F_1^* to F_2^* in the face v^*.*

Proof. Figure 5.16 depicts that the addition of the new cross-cap in the one-sided loop move corresponds to adding a duplicate of the set of edges and vertices between F_1^* and F_2^* in the face v^*. Choosing between the two different sequences from F_1^* to F_2^* in the face v^* corresponds to choosing different wedges of r to flip. □

Finally, we analyze the effect of the concatenation move. Let G be an embedding scheme with no separating loop in which every one-sided loop is orienting and it has at least one two-sided loop. Let o' be the one-sided non-orienting loop obtained by concatenating

the orienting loop o and the two-sided loop t which has an end immediately next to an end of o in the cyclic permutation around the vertex (step 3.2 of the algorithm). Denote by G' the embedding scheme in which o is replaced by o' and let H' be a drawing for G'. After applying a one-sided loop move to o', t gets orienting (by Lemma 5.4.9) and it goes through $eg(G) - 1$ cross-caps. The loop o' is the last loop that is drawn in the algorithm and therefore it passes through only one cross-cap. Let us denote this cross-cap by $\mathfrak{c}$. We denote by t_i, $1 \leq i \leq eg(G)$, the dual edges to sub-edges of t and by o'_1 and o'_2 the sub-edges of o', where t_1 corresponds to the sub-edge next to o'_1.

Depending on the wedge of o' that we choose to reverse, we proceed with concatenating o' with t to get back the loop o as discussed in the description of the concatenation move. The following lemmas describe the effect of applying the concatenation move on the dual graph of the cross-cap drawing.

Lemmas 5.5.5 and 5.5.6 explain the effect of the concatenation move in the dual graph:

Lemma 5.5.5. *When we reverse the wedge of o' that does not encompass t, concatenating the loop o' along t corresponds to:*

- *subdividing the edge adjacent to t_1^* and $o_1'^*$ that is in the cyclic rotation of $\mathfrak{c}^*$*
- *contracting the edge $o_1'^*$*
- *subdividing the edges t_i^* for $i \geq 2$*

The added edges together with $o_2'^*$, correspond to the segments of o in H.

Proof. By sliding o' along t, we remove o'_1, see Figure 5.17. Removing an edge corresponds to contracting its dual edge. Following t into the cross-caps adds parallel edges in the cross-cap drawing that corresponds to subdividing the dual edges t_i^*. □

Lemma 5.5.6. *When we reverse the wedge of o' that encompasses the ends of t, concatenating the loop o' with t corresponds to:*

- *subdividing every t_i^* except for $i = 1, 2$*
- *subdividing the edge adjacent to t_2^* and $o_1'^*$ in the face v^**
- *contracting the edge $o_1'^*$ and $o_2'^*$*

The added edges together with $o_2'^*$, correspond to the segments of o in H.

Proof. The proof is similar to the proof of Lemma 5.5.5. □

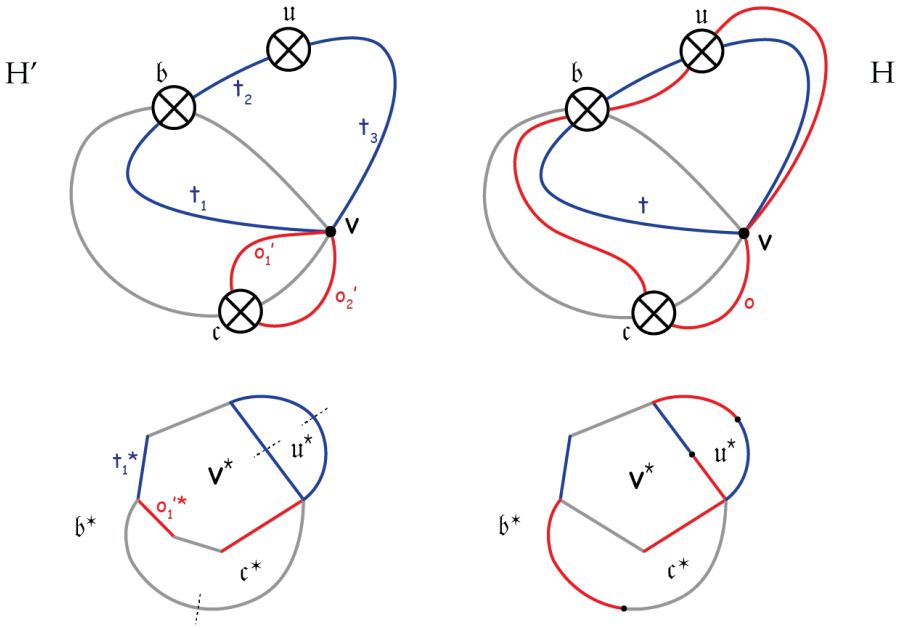

Figure 5.17: The impact of the concatenation move in the dual graph. The bottom graphs are dual to the top cross-cap drawings.

5.5.2 Short paths from each cross-cap to the vertex

Recall that a root face in a cross-cap drawing of a one-vertex graph is a face of the drawing (seen as a planar graph) adjacent to the vertex. The aim of this section is to show that there is a cross-cap drawing output by the modified algorithm, in which the cross-caps are not too far from the vertex (at distance $O(|E(G)|)$). To show this, we find paths in the dual graph of the cross-cap drawing from a vertex adjacent to the face dual to each cross-cap to the vertex corresponding to a root face.

We first show this claim for an embedding scheme with an orienting loop that has exactly one non-trivial component in its interleaving graph. Additionally, we claim that we can find a cross-cap drawing which allows us to force the paths to arrive in the same vertex in the dual graph that is chosen arbitrarily. Furthermore, we show that we can find these paths such that they do not use the dual edges corresponding to the orienting loop.

To prove this we use the modified algorithm, but with an additional specification: as we mentioned before, different choices in flipping a wedge when doing a one-sided loop move or a concatenation move yield different cross-cap drawings. The modified algorithm that we described gives us the freedom to choose the wedge whenever a one-sided loop move is applied. Here we use this freedom to build our desired paths.

Lemma 5.5.7. *For any one-vertex scheme G with an orienting loop o that has exactly one non-trivial component in its interleaving graph I_G, for any choice of root wedge ω,*

there is a choice of wedges in the modified algorithm which outputs a cross-cap drawing H with $eg(G)$ cross-caps such that for every cross-cap $\mathfrak{c}$, there is a dual path $p_\mathfrak{c}$ from a face adjacent to $\mathfrak{c}$ to the root face corresponding to ω with multiplicity two, which does not cross the orienting loop.

Proof. We prove the result by induction on $eg(G) + |E(G)|$, following the recursive steps of the modified algorithm. By Lemma 5.4.11, we know that the modified algorithm draws such an embedding scheme using only contractible loop moves, one-sided loop moves and concatenation moves. In this proof, we show that these moves can be applied in each step such that they do not increase the multiplicity of the paths that we obtain by the induction hypothesis. Crucially, when applying a one-sided loop move or a concatenation move, this relies on choosing a correct wedge to flip in each step.

We fix an arbitrary root wedge ω around the vertex v. When we remove a loop that affects ω, we update ω to be the wedge that encompasses the former fixed root wedge ω. Similarly, when we re-introduce the edges in the drawing, we subdivide ω and we choose the sub-wedge that is consistent with the first choice of ω.

Contractible loop move. Denote by G' the embedding scheme we have after removing a contractible loop c. By the induction hypothesis, there exists a drawing H' and a system of paths $\{p_\mathfrak{c}\}$ from every cross-cap to the wedge ω in H'^* with multiplicity two such that they do no use the dual edges of the orienting loop. When re-introducing c, if c does not sub-divide the wedge ω, then it does not affect the paths $p_\mathfrak{c}$. In the case that we need to update ω to be the empty wedge between the ends of c itself, by Lemma 5.5.1, we can see that in this case, the paths need to use the edge c^* in the dual once. Thus, the multiplicity of the paths remains two. The paths still do not use the dual edges of the orienting loop.

One-sided loop move on a one-sided non-orienting loop r. Denote by G' the graph we have after applying a one-sided loop move to r. By the induction hypothesis, there exists a drawing H' and a system of paths $\{p_\mathfrak{c}\}$ from every cross-cap to the wedge ω in H'^* with multiplicity at most two such that they do not use the dual edges of the orienting loop. In this case, we flip the wedge of r that does not contain ω. By this choice, we can see that the situation of the vertex dual to the face ω does not change, since by Lemma 5.5.4, drawing a one-sided loop corresponds to adding a path to the dual graph that does not separate ω from v^*. Therefore, this move does not affect any $p_\mathfrak{c}$.

We know that the orienting loop interleaves with r. For the new cross-cap $\mathfrak{c}_1$, we define the path $p_{\mathfrak{c}_1}$ as follows. We choose a face adjacent to $\mathfrak{c}$ that is a root face, and such that it is both in the flipped wedge of r and in the same wedge of the orienting loop as ω; see Figure 5.18. We introduce the path $p_{\mathfrak{c}_1}$ to be the sequence of dual edges and vertices around v between ω and this root face such that it does not use the dual edges corresponding to the orienting loop. Since each edge has exactly two ends around the vertex, the path $p_{\mathfrak{c}_1}$ uses each edge at most twice in the dual graph and its multiplicity is

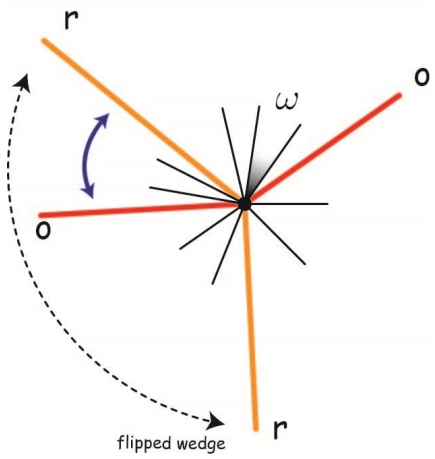

Figure 5.18: The choice of the adjacent root face for the new cross-cap: the loop o is orienting and we applied the one-sided loop move on the loop r. The dotted arrow shows the wedge of r that we flipped and the blue arrow shows the adjacent root wedges to the last cross-cap that we can choose from.

at most two. By construction, all these paths avoid using the dual edges of the orienting loop.

Concatenation move on an orienting loop o and the two-sided loop t. We denote by o' the concatenation of o and t and by G' the graph in which we replaced o by o'. The modified algorithm applies the one-sided loop move to o', and here again we choose to flip the wedge of o' that does not encompass ω. We denote the new graph after removing o' by G''. By the induction hypothesis there exists a drawing H'' for G'' and a suitable system of dual paths $\{p_c\}$ in H''^* from a face adjacent to each cross-cap to the root wedge ω with multiplicity two. These paths do not use the dual edges corresponding to the loop t since after removing o', t is orienting for the new embedding scheme (this follows from Lemma 5.4.9). Similar to the case before in which the algorithm applies the one-sided loop move, we can see that after drawing o' and adding a cross-cap, we can re-introduce the paths $\{p_c\}$ with the same multiplicity so that they do not use the two dual edges corresponding to o'. Now, the modified algorithm slides o' along t to get a drawing for the initial embedding scheme. By Lemmas 5.5.5 and 5.5.6 (depending on the situation of ω with respect to t and o'), we know that sliding o' along t corresponds to sub-dividing the dual edges corresponding to t and since the paths $\{p_i\}$ do not use the dual edges of t, they do not use the dual edges of o either. For the last added cross-cap, we take a root face adjacent to it in the same wedge that ω is placed and introduce a path by going around the vertex. As before, we know that this path has multiplicity at most two and does not use the dual edges of the orienting loop. This finishes the proof. □

Using Lemma 5.5.7, we prove the claim for the more general case in which the embed-

ding scheme can have more than one non-trivial component. Here, we do not need the paths to arrive in the same root face.

Lemma 5.5.8. *For any saturated one-vertex scheme G with an orienting loop o, there is a choice of wedges in the modified algorithm which outputs a cross-cap drawing H with $eg(G)$ cross-caps such that there is a path from every cross-cap to a root face (not necessarily fixed) with multiplicity at most two.*

Proof. The proof is by induction on the number of separating loops. When there is no separating loop, the graph has only one non-trivial component and it is non-orientable. In this case, the result follows by Lemma 5.5.7.

Let s_l be the separating loop chosen by the algorithm during Step 2, separating two sub-scheme G_l and $G \backslash G_l$ on which it recurses. Since s_l separates a leaf from the component tree, one of these sub-schemes, say $G \setminus G_l$, is non-orientable and has an orienting loop. Therefore, by the induction hypothesis, there is a drawing H' for $G \setminus G_l$ with $eg(G \setminus G_l)$ cross-caps such that there is a path with multiplicity two from every cross-cap to a root wedge.

Now, G_l is made of exactly one non-trivial component due to our way of choosing s_l. Let ω be a root wedge of G_l different from F_o, the face where the ends of s_l used to exist. We apply Lemma 5.5.7 to obtain a cross-cap drawing H_l of $G_l + \{o\}$ and a system of dual paths $\{p_\mathfrak{c}\}$ with multiplicity at most two from a face adjacent to every cross-cap to ω, such that none of them use the dual edges corresponding to the orienting loop o. Now, the algorithm glues H_l to H' and proceeds with dragging the loops from H' to H_l. We denote the resulting drawing by H.

By Remark 5.5.3, we know that the paths connecting cross-caps to root wedges in H' can be re-introduced in H, since dual edges and vertices corresponding to the edges and faces in H' are not changed in H except the vertex that is split into two vertices. Since both of these vertices are root faces in the new embedding scheme, this does not interfere with the multiplicity of these paths and each of them arrives at one of these vertices (recall that we do not require all the paths to arrive at the same root wedge). By the choice of ω and the fact that none of the paths in $\{p_\mathfrak{c}\}$ use the dual edges corresponding to the orienting loop, none of the paths visit the vertex F_o^* (F_o and ω are in different wedges of the orienting loop o). Since the paths $\{p_\mathfrak{c}\}$ do not use o, we can choose the incident face to each cross-cap so that replacing the dual edges of o by a sequence of edges (as explained in Lemma 5.5.2), does not impact the multiplicity of the paths $\{p_\mathfrak{c}\}$ from each cross-cap to ω. This finishes the proof. □

It is immediate from the proofs of Lemmas 5.5.7 and 5.5.8 that the choice of wedges in the modified algorithm in these lemmas is computable in polynomial-time. We call the

modified algorithm with the choice of wedges of these lemmas the *refined algorithm*. We finally have all the tools to prove our main result, which we recall for convenience.

Theorem D. *There exists a polynomial time algorithm that given a non-orientable cross-metric surface N, computes a canonical decomposition such that each loop in the decomposition intersects any edge of the graph in at most* 30 *points.*

In the case that the primal graph in N contains an orienting cycle, our proof yields a better bound: each loop in the system intersects any edge of the graph in at most 10 points.

Proof. Let G denote the embedding scheme of the primal graph in N. Applying the algorithm to G, we obtain the saturated one-vertex scheme $\bar{G}$ that has an orienting loop after the preprocessing steps. By Lemma 5.4.1, to prove the theorem, it is sufficient to show that there exists a canonical system of loops for a drawing of $\bar{G}$ such that each loop in the system has multiplicity 10.

The one-vertex scheme $\bar{G}$ has an orienting loop and therefore by Lemma 5.5.8, the refined algorithm outputs a cross-cap drawing $\bar{H}$ with $eg(N)$ cross-caps such that there are paths $\{p_j\}$ with multiplicity two from a face incident to each cross-cap (denote this face by b_j for each j) to a root face (denote this face by a_j for each j) in this cross-cap drawing.

Fix a root face F in the drawing $\bar{H}$. For each path p_j, build a loop ν_j by going from F to a_j, by going around the vertex in shortest way possible. By doing so, so far the loop has multiplicity at most two. Follow p_j to b_j: this adds at most two to the multiplicity since p_j has multiplicity two. Go into the cross-cap and come back to b_j by going around it (this adds at most two to the multiplicity, since every edge passes through each cross-cap at most twice by Lemma 5.4.12) Finally, follow p_j back to a_j and go back to F from the same path (these two last steps add at most 4 to the multiplicity). Therefore, each ν_j has multiplicity 10. By Lemma 5.1.1, we know that the system of loops we obtain, is a non-orientable canonical system of loops. This finishes the proof. □

Chapter 6

More on decompositions of surfaces

Summary. In this chapter, we first prove that every orientable embedding scheme admits a box drawing in which each edge of the graph enters each box at most twice. Then we show that the same approach used to provide a short canonical decomposition of non-orientable cross-metric surfaces in Chapter 5 can be adopted to provide an alternative polynomial-time algorithm that computes a short canonical decomposition of an orientable cross-metric surface. Furthermore, we show that the techniques for orientable surfaces and non-orientable surfaces can be joined to provide short decompositions along other systems of loops for non-orientable surfaces. Finally, we provide a lower bound for canonical decompositions of both orientable and non-orientable surfaces using a counting argument.

The results in this chapter were obtained with Alfredo Hubard and Arnaud de Mesmay. This chapter serves as a follow-up to the previous chapter and is highly dependent on the concepts and definitions established there.

6.1 Introduction

In the previous chapter, we devised a new technique for computing a short canonical decomposition of non-orientable cross-metric surfaces. Our technique used the geometric model introduced in Section 3.4.1 (cross-cap drawings), to get a planar drawing of the primal graph on the surface. Schaefer and Štefankovič [69] provided an algorithm that computes a cross-cap drawing for a given graph embedding in which each edge of the graph enters each cross-cap at most twice. Using a modification of this algorithm, we controlled the diameter of this drawing so that each cross-cap $\bigotimes$ is not far from a fixed basepoint. This allowed us to build a canonical system of loops by dragging a system of planarizing

disjoint one-sided curves to the basepoint such that each loop in the decomposition has constant multiplicity.

Following up on our analogous geometric model to that of cross-cap drawings for orientable surfaces in Section 3.4.2 (box drawings), we provide an analogous result to that of Schaefer and Štefankovičc (see Theorem 5.3.1 or more accurately [69, Theorem 10]) but for graphs embedded on orientable surfaces.

Theorem E. *Let G be a (connected) orientable embedding scheme with orientable genus $g(G)$. Then G admits a box drawing with $g(G)$ boxes such that each edge of G enters each box at most twice.*

Note that, given an orientable cross-metric surface, any orientable canonical decomposition of it in which each edge has low multiplicity [53], leads to a box drawing in which each edge enters each box a constant number of times, see Figure 3.18 and also [69, Theorem 13].

Combining Theorem E with the approach that we described above, allows us to provide an alternative proof for the existence of a short canonical decomposition of orientable surfaces of total length $O(g|E(G)|)$, where G is the primal graph on the surface (with a slightly worse constant compared to the one provided in [53], see also [12]).

Theorem F. *There exists a polynomial time algorithm that given an orientable cross-metric surface M with primal graph G, computes a canonical decomposition such that each loop in the decomposition intersects any edge of G in at most 6 points.*

Furthermore, we show that by merging these drawing algorithms, we can obtain a planar drawing with a combination of cross-caps and boxes such that each edge enters each box or cross-cap at most 6 times. This theorem generalizes Theorem 5.3.1.

Theorem G. *Let G be a non-orientable embedding scheme with non-orientable genus $\tilde{g}(G)$. For every l and k with $l+2k = \tilde{g}(G)$ (l and $\tilde{g}(G)$ have the same parity), G admits a drawing with $l \geq 1$ cross-caps and k boxes such that every edge passes through each cross-cap and each box at most 6 times.*

We explain how this theorem would lead to polynomial algorithms that provide short decompositions along other systems of loops for non-orientable surfaces.

Finally, using a counting argument we prove a lower bound for the total length of canonical decompositions (both orientable and non-orientable). The proof technique is inspired by Colin de Verdière, Hubard and de Mesmay [22] who prove a similar bound for combinatorial maps. Their strategy is itself derived from [41] by Guth, Parlier and Young, who prove a similar bound for pants decompositions.

Theorem H. *For any $\epsilon > 0$, the following holds with probability tending to one as g tends to ∞: an orientable (resp. non-orientable) one-vertex one-face scheme with orientable genus g (resp. non-orientable genus g), chosen uniformly at random has no orientable (resp. non-orientable) canonical decomposition of length at most $g^{\frac{5}{4}-\epsilon}$ (resp. $g^{\frac{3}{2}-\epsilon}$).*

This theorem implies that for almost all one-vertex one-face cross-metric surfaces the length of a canonical decomposition is at least superlinear.

Outline. Theorems E and F are proved in Section 6.2 and Theorems G and H are proved in Sections 6.3 and 6.4, respectively.

6.2 Canonical decomposition of orientable surfaces

A canonical decomposition of an orientable surface can be computed such that each loop in the system of loops has multiplicity at most 4 (see [53, Theorem 1]). The proof uses a cut and pasting technique and works on the combinatorial model. In this section, we provide an alternative way to compute a short orientable canonical system of loops such a decomposition with slightly higher multiplicity.

Theorem F. *There exists a polynomial time algorithm that given an orientable cross-metric surface M with primal graph G, computes a canonical decomposition such that each loop in the decomposition intersects any edge of G in at most 6 points.*

Our proof uses the cross-metric model. Recall that by $g(G)$ we refer to the orientable genus of the graph (or the embedding scheme) G.

6.2.1 Box drawing with low multiplicity

To prove Theorem F, we first introduce an algorithm that provides a box drawing for an orientable embedding scheme such that each edge passes through each box a constant number of times.

Theorem E. *Let G be a (connected) orientable embedding scheme with orientable genus $g(G)$. Then G admits a box drawing with $g(G)$ boxes such that each edge of G enters each box at most twice.*

As a pre-processing step in our algorithm, we first contract a spanning tree in the given embedding scheme G. This way, we reduce the graph to a one-vertex graph G_0 with the rotation system ρ such that, $g(G_0, \rho) = g(G)$. An embedding for (G_0, ρ) can be turned back into an embedding of G by uncontracting edges close to the vertex so that these edges do not go inside any of the boxes.

We need the following two moves on a one-vertex embedding scheme. The contractible loop move is similar to the contractible loop move in the Schaefer-Štefankovič algorithm. We recall this move here.

Contractible loop move. Let c be a contractible loop with consecutive ends in the embedding scheme G. Remove c. The new embedding scheme can be drawn using the same number of boxes. Having a drawing for the new embedding scheme, we can draw the loop c without passing through any of the boxes.

Interleaving loops move. Let e and f be two interleaving loops in the embedding scheme G. These loops divide the rotation system into four wedges which we number from 1 to 4 as in the left picture in Figure 6.1. Remove e and f and change the order between wedges 1 and 3 (See the middle picture in Figure 6.1). The new embedding scheme can be drawn using $g(G) - 1$ boxes. Given a box drawing H for the new scheme, we add a box near the vertex and next to the reordered edges; let us label the sides of the box by the sequence *abcabc*. We disconnect the edges in wedges 1, 2 and 3 from the vertex and pass them through the sides a, b and c, respectively and then connect them again to the vertex. This retrieves the original order of edges in 1, 2 and 3 in G. Finally, we add e and f back passing them through two different sides of the box (see the right picture in Figure 6.1). This gives a box drawing for G.

These moves provide a way to draw one or two loops assuming that some simpler one-vertex graph without that loop has already been drawn. Therefore, we can use the moves in an inductive algorithm as follows. The input is an orientable embedding scheme G.

The box drawing algorithm:
Pre-processing step:

- **Step A.** Contract a spanning tree to obtain a one-vertex scheme.

Main loop:

- **Step 1: If there is a contractible loop.** Apply the contractible loop move and recurse.
- **Step 2: If there exists a pair of interleaving loops.** Pick two interleaving loops e and f. Apply the interleaving loops move on e and f and recurse.

Post-processing step:

- **Step A'.** Uncontract the spanning tree near the vertex such that the edges in the spanning tree do not enter any of the boxes.

We now prove Theorem E by showing that the box drawing output by this algorithm has the correct number of boxes and each loop passes through every box at most twice.

Proof of Theorem E. Since contracting a spanning tree does not change the genus of the graph, Step A in the algorithm reduces the graph G to a one-vertex scheme G_0 such that a drawing for this graph leads to a drawing for G.

Thus, it is sufficient to prove that there is a cross-cap drawing for G_0 with $g(G_0) = g(G)$ boxes in which each loop passes through each box at most twice. We prove this claim for any orientable one-vertex scheme G.

In order to prove this claim, we follow the recursive steps of the main loop of the box drawing algorithm, using induction on $|E(G)|$.

Step 1: Contractible loop move. We apply the contractible loop move on a loop c. Let G' denote the new embedding scheme. We know that $g(G') = g(G)$ since removing a contractible loop does not change the genus. By induction, we obtain a drawing for G' in which each loop enters each box at most twice. We draw c such that it does not enter any of the boxes.

Step 2: Interleaving loops move. If $g(G) \neq 0$ then there exists such pair of loops in the graph. This is because if there are no loops with interleaving ends then all the loops are contractible and can be drawn in the plane. Let e and f be the loops in G on which the interleaving loops move has been applied. Let G' denote the new scheme we obtain after applying the interleaving loop move. We can see that $g(G') = g(G) - 1$. This is because the process of removing e and f and reordering the wedges is equivalent to cutting along e and f and contracting the emerging boundary component. Since e and f are alternating, none of them can be separating (see Lemma 5.2.3) and therefore cutting along f reduces the genus by 1 and produces two boundary components on the surface. Since the ends of e were alternating with those of f's, the edge e becomes an arc connecting these two boundaries. Therefore, cutting along e will reduce the boundary components by 1 and preserves the genus. We can check further to see that after contracting this boundary (as explained in Section 3.3.2, we get the rotation system in G'.

Since $|E(G')| < |E(G)|$, G' has a box drawing H' with $g(G) - 1$ boxes such that each loop in G' enters each box at most twice. Let H denote the box drawing we obtain for G after adding the last box near the wedges 1, 2 and 3 and passing the edges in these wedges through this box. Since the wedges 1, 2 and 3 may contain both ends of a loop, each edge in these wedges enter the new box at most twice and edges e and f enter this box exactly once. Note that in applying this move, the situation between the edges and the other boxes remains intact. This finishes the proof. □

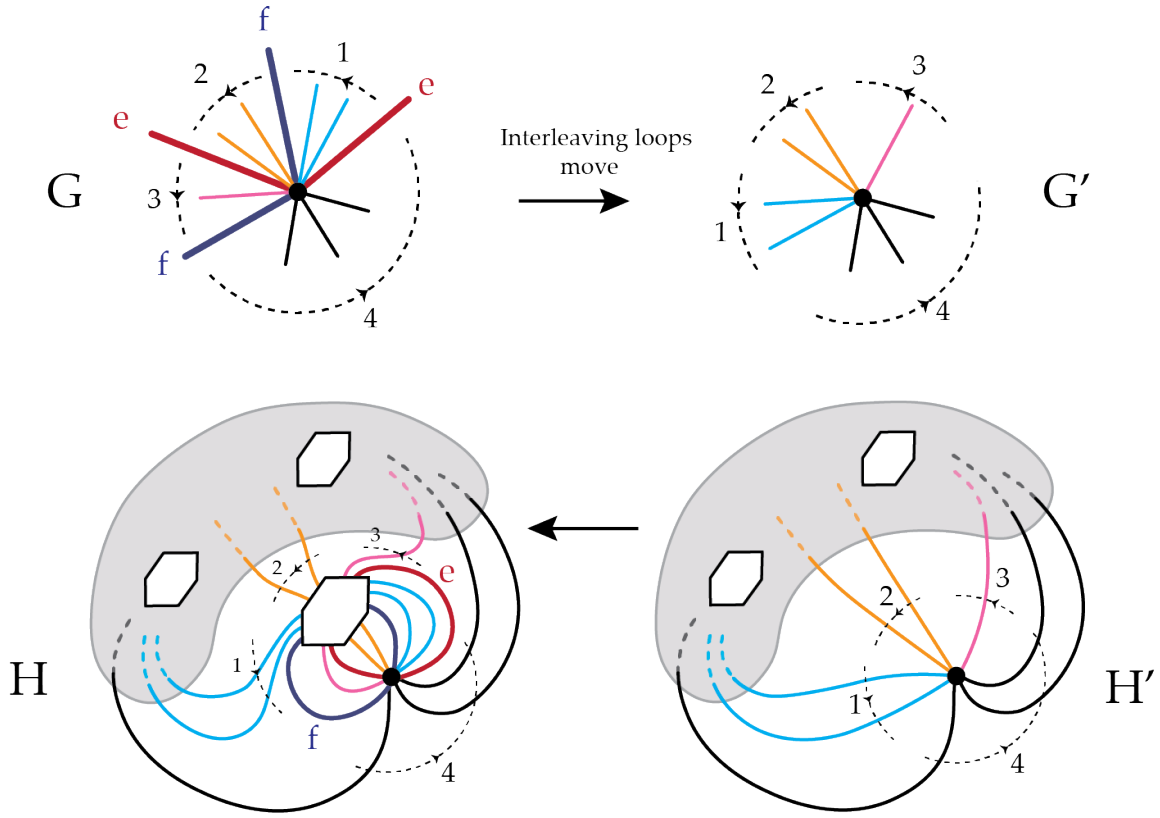

Figure 6.1: The process in case 2.

6.2.2 Reproving the $O(gn)$ bound for orientable canonical system of loops

In this section, we prove Theorem F using the Box drawing algorithm and Theorem E. The idea of the proof is similar to the non-orientable case (Section 5.5). We show that the complexity of the drawing obtained from the box drawing algorithm can be controlled so that the boxes are not too far from a basepoint b. We achieve this by choosing the wedge whose edges do not enter the new box whenever we apply an interleaving loops move (the wedge 4 in our previous notation).

Lemma 6.2.1. *For any orientable one-vertex scheme G, there is a choice of wedges in the box drawing algorithm which outputs a box drawing H with $g(G)$ boxes such that there is a path from every box to a point b on the surface with multiplicity at most two.*

Given such system of paths from every box to a basepoint b, we obtain a canonical system of loops by the following lemma.

Lemma 6.2.2. *Let H be a box drawing for a graph of orientable genus g and let b be a point in one face of the drawing. Let $\{p_i\}$ be a family of paths in the dual graph to this drawing from each box to b. Introduce a loop c_i by starting from b, passing along the*

path p_i, entering the corresponding box from one of the sides, going around the box and passing along p_i to return to b. Introduce d_i similarly but make d_i enter another side of the box such that it does not cross c_i. The system of loops we obtain by all c_i and d_is is an orientable canonical system of loops.

Proof. It is easy to check that for each i the ends of c_i and d_i interleave around b. Also for each $i \neq j$, the ends of c_i and d_i encloses the ends of c_j and d_j. Cutting along $\{c_i\}$, cuts the surface into a disk, and therefore we obtain an orientable canonical system of loops. This can be seen in Figure 6.2. □

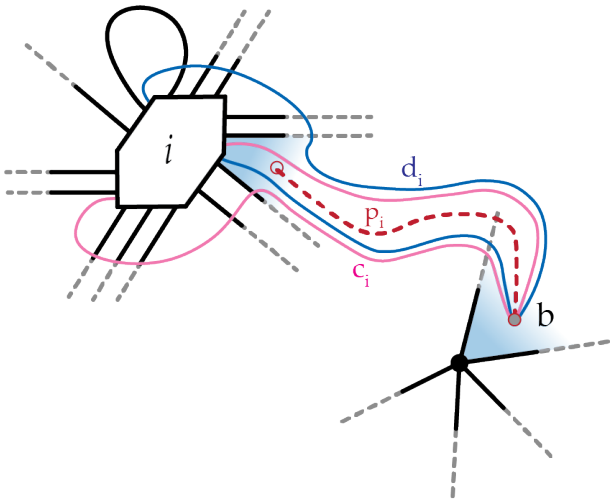

Figure 6.2: The process of building c_i and d_i using the path p_i for the box i.

Given Lemma 6.2.2, to prove Theorem F, it is sufficient to prove Lemma 6.2.1. In ordert to prove Lemma 6.2.1, we first investigate the effect of applying the moves in the box drawing algorithm in the dual graph of the box drawing (viewed as a planar graph).

The dual graph of the box drawing. Given a one-vertex embedding scheme G with vertex v and a box drawing H, recall that we call the empty wedge between two consecutive half-edges around v a *root wedge*. A face incident to v in a cross-cap drawing may correspond to more than one root wedge. We refer to such a face as a *root face*. Every edge e in a box drawing H, corresponds to an edge e^* and every face F corresponds to a vertex F^* in the dual graph. The vertex v corresponds to the face v^* and the boxes correspond to the other faces in the dual graph.

The effect of applying a contractible loop move in the dual graph of the drawing is similar to the one showed in the previous section. Therefore for the sake of redundancy, we do not state it again here and we refer the reader to Lemma 5.5.1. The following lemma states the effect of an interleaving loops move in the dual graph of the drawing.

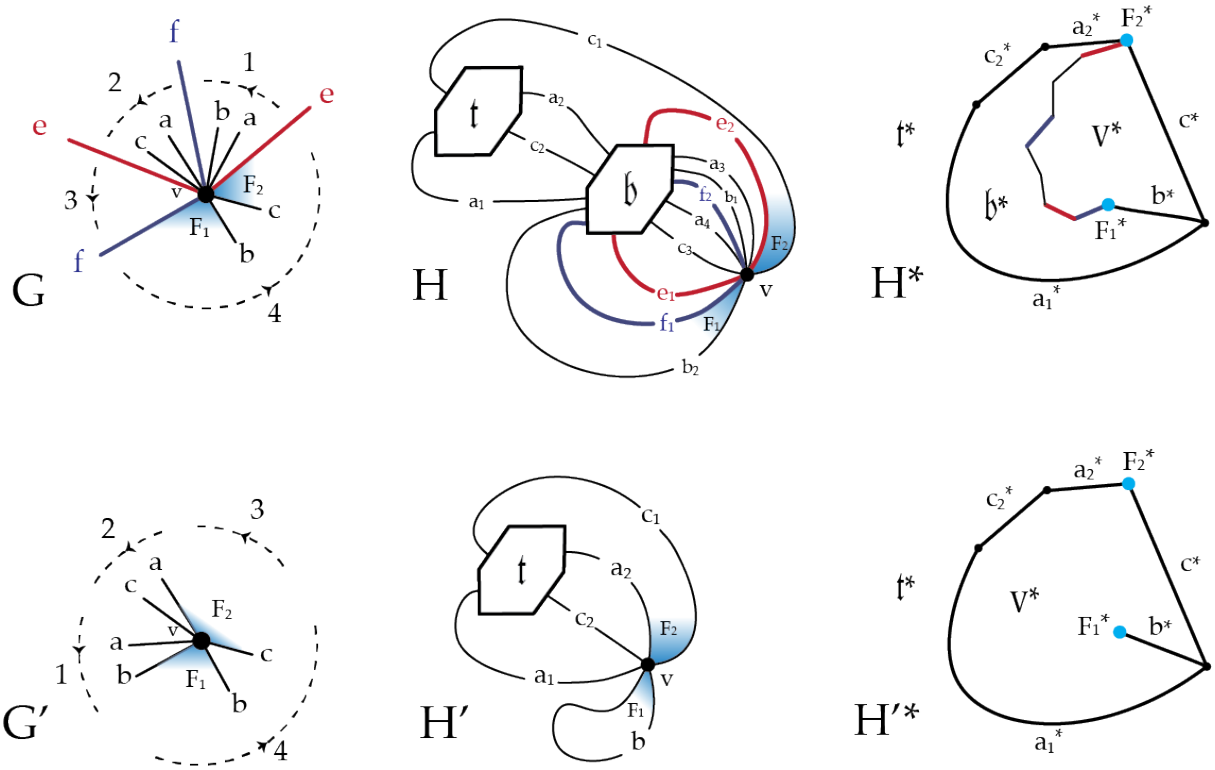

Figure 6.3: The interleaving loops move in the dual graph. The scheme G' is the scheme we obtain by applying a interleaving loops move on edges e and f in G. H and H' are box drawings for G and G' and the graphs $H*$ and $H'*$ depict the dual graph these box drawings, respectively.

Lemma 6.2.3. *Let G be a one-vertex orientable embedding scheme, and e and f two interleaving loops in G. Number the four wedges between the ends of e and f from 1 to 4. Let G' be the scheme we obtain after removing the loops e and f and reordering the two wedges 1 and 3 as explained in the interleaving loops move. Let H' be a box drawing for G' and denote by F_1 (resp. F_2) the root faces formed by an edge in wedge 4 and an edge in wedge 1 (resp. an edge in wedge 3). Drawing e and f, adding the last box and dragging the half-edges in the wedges 1, 2 and 3, corresponds to subdividing the face v^* into two faces and adding a path of length $k+4$ from F_1^* to F_2^* where k is the number of half edges in the wedges 1, 2 and 3.*

Proof. Figure 6.3 depicts that the addition of the new box in the interleaving loops move corresponds to adding a duplicate of the set of edges and vertices between F_1^* and F_2^* in the face v^* and drawing the edges e and f corresponds to 4 additional dual edges (2 of them correspond to two segments of edge e and the other 2 correspond to the segments of the edge f) in the dual graph. □

Now we are ready to prove Lemma 6.2.1 which implies Theorem F.

Proof of Lemma 6.2.1. Let b be a point on one of the faces in G. Without loss of generality

let ω be a root wedge corresponding to this face. Denote by H the planar box drawing in which each box is considered as a vertex, and edges of H are the sub-edges in G.

Claim: G admits a box drawing with $g(G)$ boxes such that for each box in H, there exists a path with multiplicity at most 2 from ω to a face incident to that box.

We prove the claim by induction on $|E(G)|$ (the proof is similar to the proof of Lemma 5.5.7).We show that the contractible loop move and the interleaving loops move can be applied in each step such that they do not increase the multiplicity of the paths that we obtain by the induction hypothesis. Specially in the case of the interleaving loops move, we achieve this by carefully labeling the wedges in the interleaving move: we always label the wedges so that the edges of the root wedge ω do not enter a new box.

- Denote by G' the embedding scheme we obtain after removing a contractible loop c. By the induction hypothesis, there exists a drawing H' for G' and a system of paths $\{p_{\mathfrak{c}}\}$ from every box to the wedge ω in H'^* with multiplicity two. When re-introducing c, if c does not sub-divide the wedge ω, then it does not affect the paths $p_{\mathfrak{c}}$. In the case that we need to update ω to be the empty wedge between the ends of c itself, by Lemma 5.5.1, we can see that in this case, the paths need to use the edge $c*$ in the dual once. Thus, the multiplicity of the paths remains two.

- Let e and f be the interleaving loops in G on which the interleaving loops move is applied. Number the four wedges formed by the ends of e and f from 1 to 4 such that the wedge 4 is the one that contains the root wedge ω. Denote by G' the embedding scheme we obtain after removing e and f and replacing the wedges 1 and 3. By the induction hypothesis, there exists a drawing H' for G' and a system of paths $\{p_{\mathfrak{c}}\}$ from every box to the wedge ω in H'^* with multiplicity two. When re-introducing e and f, and dragging the edges in the wedges $1, 2$ and 3 into the new box, we do not change the situation of the paths $P_{\mathfrak{c}}$. This is because our choice for wedge 4 implies that the situation of the vertex dual to ω does not change since by Lemma 6.2.3, completing the drawing when applying an interleaving loops move, corresponds to adding a path to the dual graph, that does not separate ω from $v*$. Therefore, the paths $P_{\mathfrak{c}}$ can be reintroduced using the same dual edges and vertices as before.

  Now we introduce a path from the new cross-cap $\mathfrak{c}_1$ to ω. We choose the face $F_{\mathfrak{c}_1}$ adjacent to $\mathfrak{c}_1$ in the dual graph that is a root face. We introduce the path $p_{\mathfrak{c}_1}$ to be the sequence of dual edges and vertices around v between ω and $F_{\mathfrak{c}_1}$. Since each edge has exactly two ends around the vertex, the path $p_{\mathfrak{c}_1}$ uses each edge at most twice in the dual graph and its multiplicity is at most two.

This finishes the proof of the claim.

□

We now are ready to prove Theorem F.

Proof of Theorem F. Let G denote the embedding scheme corresponding to the graph of the metric in M. Choose a point b on M and proceed with the box drawing algorithm. After the pre-processing step, we obtain a one-vertex embedding scheme with b as a root face. Using Lemma 6.2.1, we obtain a box drawing for G such that each edge of G enters each box at most twice and there exists paths $\{p_{\mathfrak{c}}\}$ from each box to b with multiplicity at most two.

We build the canonical system of loops using Lemma 6.2.2 and the paths $\{p_{\mathfrak{c}}\}$. Each of the loops constructed in this lemma, is obtained by following a path $p_{\mathfrak{c}}$ twice and entering a box and going around it once. Since each box is entered by each edge at most twice, then going around the boxes add 2 to the multiplicity of the loops. This concludes that the multiplicity of each loop is at most 6. We conclude. □

6.3 Non-orientable embeddings with a combination of boxes and cross-caps

In this section, we provide an algorithm that provides a planar drawing with both cross-caps and boxes. Combining the techniques and moves in the box drawing algorithm in Section 6.2.1 and our modified algorithm in Section 5.4, we prove the following theorem.

Theorem G. *Let G be a non-orientable embedding scheme with non-orientable genus $\tilde{g}(G)$. For every l and k with $l+2k = \tilde{g}(G)$ (l and $\tilde{g}(G)$ have the same parity), G admits a drawing with $l \geq 1$ cross-caps and k boxes such that every edge passes through each cross-cap and each box at most 6 times.*

To prove this theorem, we use the technique in the modified algorithm to reduce our embedding scheme to one that has only one vertex and contains an orienting loop (see Step A in the modified algorithm in Section 5.4). Then we apply the modified algorithm until we have drawn $l-1$ cross-caps. We know that the scheme still contains an orienting loop which allows us to draw the last cross-cap in our budget by applying a one-sided loop move on the orienting loop (Lemma 3.3.2 provides that the orienting loop at this point is indeed one-sided). After this, we have an orientable scheme and we apply the box drawing algorithm in Section 6.2.1 to finish the drawing of the initial scheme.

We describe the algorithm in the following. We use the moves on loops that we introduced in Sections 5.3 and 6.2.

The (l, k) combination algorithm:
Pre-processing steps:

- **Step A.** If there is no orienting loop, we add an orienting loop and contract a spanning tree using Lemma 5.4.1.

Main loop:

- **Step 1: If there exists a separating (non-contractible) loop and $l > 1$.** We pick a separating loop s that divides the embedding scheme to a non-orientable sub-scheme G_1 and an orientable sub-scheme G_2.
  - If $\tilde{g}(G_1) > l$: recurse on G_1 to obtain a drawing with l cross-caps and $\frac{\tilde{g}(G)-l}{2}$ boxes and apply the box drawing algorithm on G_2. Apply the gluing move on G_1 and G_2.
  - If $\tilde{g}(G_1) \leq l$: apply the modified algorithm on G_1 and recurse on G_2 to obtain a drawing with $l - \tilde{g}(G_1) + 1$ cross-caps and k boxes. Then apply the gluing and dragging move (introduced in Section 5.3) to remove the extra cross-cap from the drawing.
- **Step 2: If there is no separating loop and $l > 1$.** Apply the modified algorithm until we reduce the scheme to a scheme of genus $2k + 1$. Recurse to obtain a drawing with one cross-cap and k boxes.
- **Step 3: If $l = 1$.** Apply a one-sided loop move on the orienting loop. Then apply the box drawing algorithm on the resulting orientable scheme.

Post-processing steps:

- **Step A'.** Uncontract the spanning tree and remove the orienting loop added in step A.

Now we are ready to prove that this algorithm provides a drawing with the desired number of cross-caps and boxes such that each edge enters each cross-cap and each box at most twice.

Proof of Theorem G. Let G be an embedding scheme with non-orientable genus $\tilde{g}(G)$. Applying Step A, we obtain a one-vertex scheme G' with an orienting loop that also has non-orientable genus $\tilde{g}(G)$. We know that each edge in G is subdivided to at most three loops in G'. A drawing for G' can be extended to a drawing for G without changing the number of times a cross-cap or a box is entered by the loops in G' (sub-edges in G).

Therefore, it is sufficient to show that G' can be drawn using l cross-caps and k boxes such that each loop in G' enters each box and each cross-cap at most twice.

We know by Lemma 5.4.4 that G' is a non-orientable scheme and contains at least one one-sided loop. Also by Corollary 5.4.10 we know that in the main loop in the modified algorithm, at each step, our embedding scheme contains an orienting loop.

We show that the algorithm gives a drawing with a correct number of cross-caps and boxes. The proof is by induction on $\tilde{g}(G') = \tilde{g}(G)$ and we show that $k = \frac{\tilde{g}(G)-l}{2}$.

Step 1: Since G' has an orienting loop, by Lemma 5.4.5 we know that a separating loop s divides G' to an orientable scheme G_1 and a non-orientable schemes G_2. Therefore Step 1 deals with all the cases where there exists a separating loop. Let us assume that G_1 is non-orientable and G_2 is orientable. We know that $\tilde{g}(G) = \tilde{g}(G_1) + \tilde{g}(G_2) - 1$ since G_2 needs an extra cross-cap to embed (see Lemma 3.3.2).

- In case $\tilde{g}(G_1) > l$, by the induction hypothesis we obtain a drawing for G_1 with l cross-caps and $\frac{\tilde{g}(G_1)-l}{2}$ boxes and the box drawing algorithm gives a drawing for G_2 with $\frac{eg(G_2)}{2} = \frac{\tilde{g}(G_2)-1}{2}$ boxes. By attaching these drawings we get a drawing with l cross-caps and $\frac{\tilde{g}(G_1)-l}{2} + \frac{\tilde{g}(G_2)-1}{2} = \frac{\tilde{g}(G_1)+\tilde{g}(G_2)-1-l}{2} = \frac{\tilde{g}(G)-l}{2}$ boxes that is the correct number of boxes.

- In case $\tilde{g}(G_1) \leq l$, by the modified algorithm we obtain a drawing for G_1 with $\tilde{g}(G_1)$ cross-caps and by the induction hypothesis we obtain a drawing for G_2 with $l - \tilde{g}(G_1) + 1$ cross-caps and k boxes. By attaching these drawings we get a drawing with $\tilde{g}(G_1) + (l - \tilde{g}(G_1) + 1) = l + 1$ cross-caps and k boxes. By applying the dragging move, we remove one cross-cap from the drawing of G_1 by dragging edges in G_1 through all the cross-caps in the drawing for G_2. This gives us the right combination of boxes and cross-caps.

Step 2 and 3: In Step 3, note that when $l = 1$ the non-orientable genus is odd and therefore an orienting loop in the scheme is one-sided. We obtain a drawing for the graph with one cross-cap and $\frac{\tilde{g}(G)-1}{2}$ boxes by applying a one-sided loop move on an orienting loop o. This is because applying a one-sided loop move on o is equivalent to cutting along o and contracting the emerging boundary component (as explained in Section 3.3.2); let us denote the scheme we obtain by $\hat{G}$. First, cutting along a one-sided loop reduces the non-orientable genus by 1. Second, we know that cutting along o changes the signature of the loops that interleave with o in G. Since o is orienting, all one-sided loops interleave with o and no two-sided loop interleaves with o. This implies that after applying this move, all loops are two-sided.

In Step 2, we apply the modified algorithm until we obtain an embedding scheme of non-orientable genus $2k + 1$ (this is done by drawing $\tilde{g}(G) - 2k - 1$ cross-caps). By

induction, and by using Step 3 we get a drawing with 1 cross-cap and k boxes for this embedding scheme. This gives a drawing $(\tilde{g}(G) - 2k - 1) + 1 = l$ cross-caps and k boxes.

Note that in the drawing we obtain by this algorithm, the orienting loop does not enter any of the boxes. The number of times each loop enters a cross-cap or a box is at most twice. This can be seen easily as we basically only merged the steps in the modified algorithm and the box drawing algorithm. This concludes.

□

Remark 6.3.1. *By using the same approach as the one we used to compute a short canonical decompositions of orientable surfaces (in Section 6.2) and for non-orientable surfaces (in Section 5.5), we can provide a polynomial time algorithm that computes other types of short decompositions of non-orientable surfaces. By controlling the complexity of a drawing obtained by a (l, k) combination algorithm, we obtain a drawing in which the boxes and cross-caps are not far from a basepoint. This way we can build a decomposition whose corresponding polygonal schema is $a_1a_1 \dots a_la_lb_1c_1\overline{b}_1\overline{c}_1 \dots b_kc_k\overline{b}_k\overline{c}_k$. The proof is completely similar to the ones in Sections 5.5 and 6.2.*

6.4 A lower bound for canonical decompositions

In the previous sections of this chapter, we proved that canonical decompositions of both orientable and non-orientable cross-metric surfaces exist such that the total length of the decomposition is $O(g|E(G)|)$ where G is the primal graph on the surface. We can actually see that this bound is asymptotically tight: consider an orientable cross-metric surface with primal graph G in which half of the handles are at distance $O(|E(G)|)$ from the other half. In any canonical decomposition of this surface, at least half of the loops have length $\Omega(|E(G)|)$. However, we do not know if the bound $O(g|E(G)|)$ is tight if we restrict ourselves to cross-metric surfaces in which the primal graph has one vertex and its embedding has only one face. In particular, the example described above does not belong to this family of cross-metric surfaces: for two handles to be relatively far from each other in a cross-metric surface, the embedding of the primal graph needs to have $\Omega(|E(G)|)$ faces.

In the case of cross-metric surfaces with one-vertex one-face primal graphs, the upper bound $O(g|E(G)|)$ for the total length of canonical decompositions becomes $O(g^2)$ (by Euler formula, $|E(G)| = g$ in this case). Can we do any better for these graphs? (This question was asked by Lazarus in [52, page 143]).

Although we do not answer this question, we prove a superlinear lower bound for the length of canonical decompositions of these cross-metric surfaces. In this section, $f(n) \lessapprox g(n)$ means that $f(n)$ is less than $g(n)$ up to terms that are only exponential in n.

We prove the following probabilistic theorem. The proof is inspired by a theorem in [22] in which a similar bound for combinatorial maps is provided.

Theorem H. *For any $\epsilon > 0$, the following holds with probability tending to one as g tends to ∞: an orientable (resp. non-orientable) one-vertex one-face scheme with orientable genus g (resp. non-orientable genus g), chosen uniformly at random has no orientable (resp. non-orientable) canonical decomposition of length at most $g^{\frac{5}{4}-\epsilon}$ (resp. $g^{\frac{3}{2}-\epsilon}$).*

To prove the theorem, we need the following two lemmas.

Lemma 6.4.1. *There exist at most $O((\frac{L}{g}+1)^{4g}) \times 2^{O(g)}$ (resp. $O((\frac{L}{g}+1)^{8g}) \times 2^{O(g)}$) one-vertex one-face non-orientable (resp. orientable) schemes on a surface of non-orientable genus g (resp. orientable genus g) such that there exists a canonical system of loops with which they cross at most L times.*

Proof. The proof for orientable and non-orientable surfaces is the same but gives different constants. Here, we assume that the surface is non-orientable. Given a one-vertex one-face scheme G with genus g (G has g edges), let C be a non-orientable canonical decomposition embedded simultaneously with G on a surface N of non-orientable genus g such that it crosses the edges of G in at most L points. We denote by v the vertex of G. Cut N along C to obtain a polygon and let H denote the graph overlaid with this polygon (a $2g$-gon); the edges of this graph are comprised of sub-edges in G and C, see Figure 6.4.

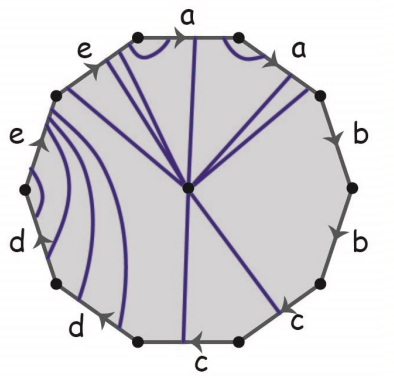

Figure 6.4: The graph H: the edges on the boundary are the copies of loops in the non-orientable canonical system of loops C, and the blue edges are the sub-edges of G after cutting along C.

A sub-edge of G is either an edge connecting the vertex v to a vertex on the boundary of the polygon (a crossing point between G and C) or an edge with both endpoints on the boundary. We denote by H_1 the graph induced by the edges incident to v and by H_2 the graph induced by the edges connecting two vertices on the boundary; we call the edges in H_2 by *chords* (see Figure 6.5). We call two chords with endpoints on the same sides of the polygon *parallel*.

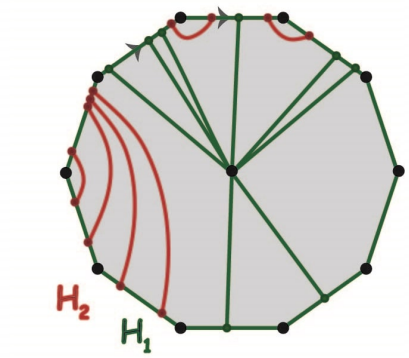

Figure 6.5: On the boundary of the polygon, the black vertices are copies of the vertices in C and the red and green vertices are copies of the points of intersections between C and G.

Without loss of generality, we can assume that the drawings of G and C have a minimum number of crossings. Then we can show that there are no chords with two endpoints on the same side of the polygon. This is because otherwise C can be modified such that the number of crossings between G and C is reduced by two which is in contradiction with C having minimum crossing with G.

Now, after analysing a simultaneous embedding of a one-vertex one-face graph and a canonical system of loops, we estimate the number of different one-vertex one-face schemes G that we can obtain by combining all possible graphs H_1 and H_2. Note that here we cannot control if the schemes we obtain are one-face schemes or not, but all we need is an upper bound, so we do not care if we are counting some schemes more than once or we are not getting a valid drawing.

Estimating all different possible graphs H_1: Since G is a one-vertex one-face embedding scheme, by the Euler formula we know that G has g loops, therefore the degree of v is $2g$. Let us consider a wheel with $2g$ spokes to be the middle vertex and its adjacent edges drawn on a disk, see Figure 6.6, left. We first add the copies of the vertices of C on the

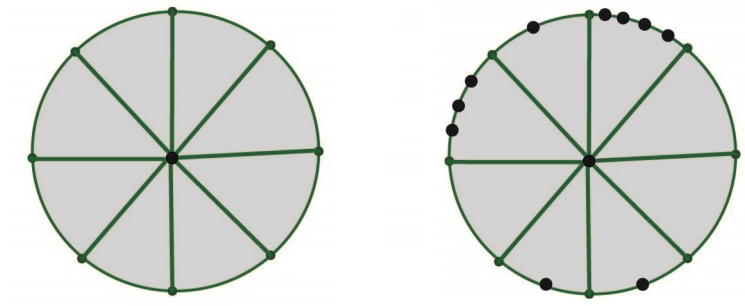

Figure 6.6: Estimating different choices for H_1.

boundary of the polygon. We need to count in how many ways we can put the $2g$ copies of the vertices of C between the $2g$ spokes, possibly with repetitions, see Figure 6.6, right. This amounts to $\binom{4g}{2g} = 2^{O(g)}$.

Estimating all different possible system of chords: Each system of chords is a subset of a triangulation of the $2g$-gon and we know there are at most 2^{2g} different triangulations.

Next, having a drawing of H_1 on a $2g$-gon, we need to combine the chords in this drawing.

We estimate the number of possible ways to combine a graph H_2 consisting of chords with a fixed H_1: at first let us forget about parallel chords in H_2. Fix a type for a system of non-parallel non-crossing chords corresponding to a triangulation. We know that in such system of chords on a polygon with $2g$ sides, there can exist at most $4g-3$ different non-crossing chords: If we associate a point to each side of the polygon, this corresponds to the number of edges for triangulating a set of $2g$ convex points, Figure 6.7 shows this correspondence.

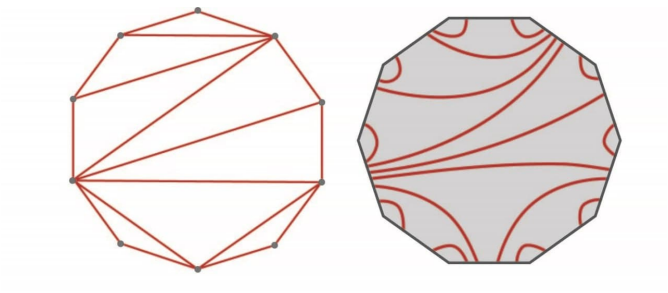

Figure 6.7: A system of non-parallel non-crossing chords for a $2g$-gon

We count the number of different distribution of all the chords on a fixed system. Note that the number of chords is equal to the number of crossings between G and C and therefore is at most L. Letting $1 \leq i \leq 4g-3$ denote different types of chords in a system, let $0 \leq x_i$, denote the number of parallel chords of type i. The number of possibilities for these distributions equals $\{(x_1, \ldots, x_{4g-3}) | x_i \geq 0, \sum_i x_i \leq L\}$, which is,

$$\binom{L+4g-3}{4g-3} \leq \left(\frac{e(L+4g-3)}{4g-3}\right)^{4g-3} = O((\frac{L}{g}+1)^{4g}) \times 2^{O(g)}.$$

Multiplying all the possible choices for H_1 and H_2, we obtain the claimed bound $O((\frac{L}{g}+1)^{4g}) \times 2^{O(g)}$ for the number of one-vertex one-face schemes for which there exists a canonical decomposition of length at most L. □

Remark 6.4.2. *The same proof works for orientable surfaces with the difference that by cutting along the canonical orientable system of loops we obtain a $4g$-gon where g is the orientable genus of the surface and all the quantities change accordingly. In this case we obtain an upper bound of $O((\frac{L}{g}+1)^{8g}) \times 2^{O(g)}$.*

The following lemma estimates the number of one-vertex one-face embedding schemes in both the orientable and the non-orientable setting.

Lemma 6.4.3. *The number of orientable one-vertex one-face schemes of orientable genus g, for g large enough is at most*

$$\frac{1}{12^g g!\sqrt{\pi}}(2g)^{3g-\frac{3}{2}}4^{2g},$$

and the number of non-orientable one-vertex one-face schemes of non-orientable genus g, for g large enough is at most

$$\frac{c_g}{6^g\sqrt{\pi}}(g)^{3g-\frac{3}{2}}4^g,$$

where $c_g = 3\cdot 2^{3g-2}\frac{g!}{(2g)!}\sum_{l=0}^{g-1}\binom{2l}{l}16^{-l}$.

The orientable case is derived from [15, Corollary 4] and the non-orientable case is derived from [8, Theorem 11]. Note that in both of these results, the number of all unicellular maps (all one-face embedding schemes) with n edges is estimated. Since for a one-face one-vertex embedding scheme $n = g$, it is enough to put $n = g$ into the formulas to achieve the number of all one-face one-vertex embedding schemes.

Remark 6.4.4. *Note that the bounds provided for the number of one-vertex one-face scheme are $\lessapprox g^{2g}$ in both cases of orientable and non-orientable schemes.*

Lemmas 6.4.1 and 6.4.3 together imply the orientable instance of Theorem H.

Proof of Theorem H. By Lemma 6.4.1, the number of orientable one-face one-vertex embedding scheme such that there exists a canonical decomposition of length at most L is $O((\frac{L}{g}+1)^{8g})\times 2^{O(g)}$. Then for g large enough and $L = g^{\frac{5}{4}-\epsilon}$, the number of one-face one-vertex embedding schemes with length at most L is

$$\lessapprox (g^{(\frac{1}{4}-\epsilon)}+1)^{8g}\times 2^{O(g)} \lessapprox g^{2g-\epsilon'}.$$

By Remark 6.4.4, the total number of orientable one-face one-vertex embedding schemes is $\lessapprox g^{2g}$. This implies that for large g, most orientable one-face one-vertex embedding schemes have no canonical decomposition of length less than L.

The proof for non-orientable surfaces is similar to the orientable one. By Lemma 6.4.1, the number of non-orientable one-face one-vertex embedding scheme for which there exists a canonical decomposition of length at most L is $O((\frac{L}{g}+1)^{4g}\times 2^{O(g)}$. Then for g large enough and $L = g^{\frac{3}{2}-\epsilon}$, the number of one-face one-vertex embedding schemes with length at most L is

$$\lessapprox (g^{(\frac{1}{2}-\epsilon)}+1)^{4g}\times 2^{O(g)} \lessapprox g^{2g-\epsilon'}.$$

By Remark 6.4.4, the total number of orientable one-face one-vertex embedding schemes is $\lessapprox g^{2g}$. This implies that for large g, most orientable one-face one-vertex embedding schemes have no canonical decomposition of length less than L. □

Chapter 7

Degenerate Crossing Number vs. Genus Crossing Number

Summary. In this chapter, we prove a structure theorem that almost completely classifies the loopless two-vertex embedding schemes for which the degenerate crossing number equals the non-orientable genus. In particular, we provide a counterexample to Mohar's stronger conjecture, but show that in the vast majority of the two-vertex cases, the conjecture does hold.

The results in this chapter were obtained with Alfredo Hubard and Arnaud de Mesmay and appear in [C]. These results appear in the Proceedings of the 31st Symposium on Graph Drawing and Network Visualization.

7.1 Introduction

Recall that the *degenerate crossing number* of G, $dcr(G)$, is the smallest number of crossings among simple drawings of G in the plane such that the crossings are transversal and crossing of multiple edges in a point is counted as one. If we allow self-crossings for edges in G, this defines the *genus crossing number* of G, $gcr(G)$. In this chapter, we investigate the conjectures of Mohar on the equality of the degenerate crossing number and the genus crossing number of graphs (Conjectures 2 and 3).

Conjecture 2. ([57, Conjecture 3.1]) *For every simple graph G, $dcr(G) = gcr(G)$.*

We described in the preliminaries how these crossing numbers can be interpreted in terms of cross-cap drawings (see Lemma 3.4.4 and the discussions in Section 3.4.1.2).

Conjecture 3. ([57, Conjecture 3.4]) *For any positive integer g, every loopless pseudo-triangulation of N_g admits a cross-cap drawing with g cross-caps in which each edge enters each cross-cap at most once.*

We say that a cross-cap drawing of graph G is *perfect* if there are $\tilde{g}(G) = gcr(G)$ cross-caps and every edge intersects each cross-cap at most once. Then with this definition, Conjecture 2 and 3 can be restated as follows.

Conjecture 2. *Every simple graph admits a perfect cross-cap drawing.*

Conjecture 3. *For any positive integer g, every loopless pseudo-triangulation of N_g admits a perfect cross-cap drawing.*

An even stronger conjecture was hinted at in [57, Paragraph following Conjecture 3.4], suggesting that one could possibly remove the loopless assumption if one forbids separating loops. This strengthening was disproved by Schaefer and Štefankovič [69, Theorem 7].

In addition to their motivation from crossing number theory, these conjectures would also shed light on the difficult task of visualizing high genus embedded graphs, providing an alternate approach to that of Duncan, Goodrich and Kobourov [29], who rely on canonical polygonal schemes [53].

7.1.1 Our results

A big step towards both these conjectures was achieved by Schaefer and Štefankovič, who proved [69, Theorem 10] that any multi-graph embedded on a non-orientable surface of genus g admits a cross-cap drawing with g cross-caps, in which each edge enters each cross-cap at most twice. This theorem applies in particular to one-vertex embedding schemes (see Theorem 5.3.1), and thus suggests a natural approach towards proving Conjectures 2 and 3. First contract a spanning tree to obtain a one-vertex graph and apply this theorem. Then, edges might enter cross-caps twice, but since the initial graph is loopless, one could hope to uncontract some edges so as to spread these two cross-caps on two edges, thus obtaining a perfect cross-cap drawing. Our first result shows that this approach cannot work, as some loopless two-vertex schemes do not admit perfect cross-cap drawings.

Theorem I. *A loopless two-vertex embedding scheme that consists of exactly one non-trivial positive block and one non-trivial negative block admits no perfect cross-cap drawing.*

We refer to Figure 7.1 for an example that should provide an intuitive idea of the notion of blocks, and to Section 7.2 for the precise definition. As a corollary, we obtain a counterexample to Conjecture 3:

Corollary 7.1.1. *There exist a loopless pseudo-triangulation G that admits no perfect cross-cap drawing.*

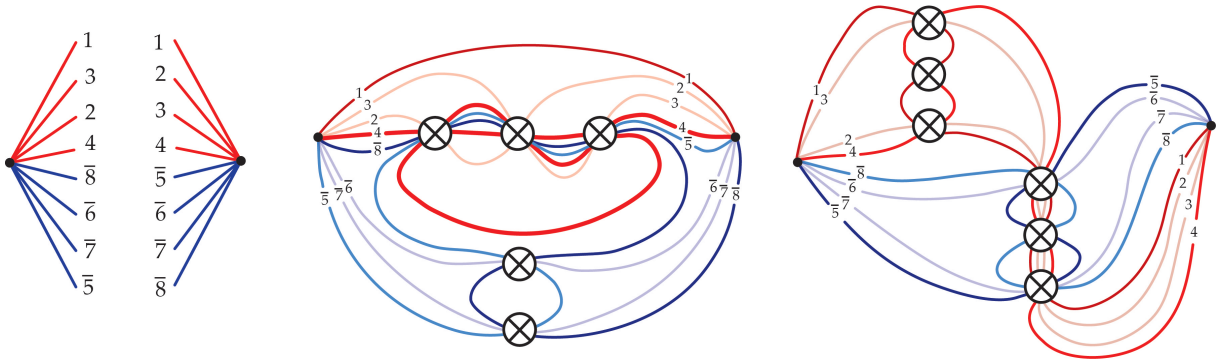

Figure 7.1: Left: a loopless two-vertex scheme made of a *positive block*, in red, consisting of only positive edges, and a *negative block*, in blue, consisting of negative edges. Middle: a cross-cap drawing showing that it has non-orientable genus 5. The bold red edge enters three cross-caps twice. Right: A cross-cap drawing where each edge enters each cross-cap at most once requires 6 cross-caps.

Our second contribution and main theorem is a converse to Theorem I.

Theorem J. *For any embedding G of a loopless two-vertex graph on N_g, at least one of the following is true.*

1. *G admits a perfect cross-cap drawing with g cross-caps,*
2. *or the reduced graph of G is one of the two schemes pictured in Figure 7.2,*

Figure 7.2: The only two exceptions to perfect cross-cap drawings.

We refer to Section 7.2 for the definition of blocks and reduced graphs. Essentially, Theorem J shows that apart from two narrow families of exceptional cases, all the loopless two-vertex embeddings do satisfy Conjecture 3. As an illustration, Figure 7.3 shows that while the example in Figure 7.1 does not admit a perfect cross-cap drawing, surprisingly, it does after *adding* two edges to it. It directly follows from Theorem J that under standard random models, any loopless two-vertex embedding scheme admits a perfect cross-cap drawing asymptotically almost surely.

7.1.2 Techniques and connections to signed reversal distance

Our focus on the two-vertex case in Theorem J is further motivated by a connection (introduced in Section 3.8.2) to computational genomics.

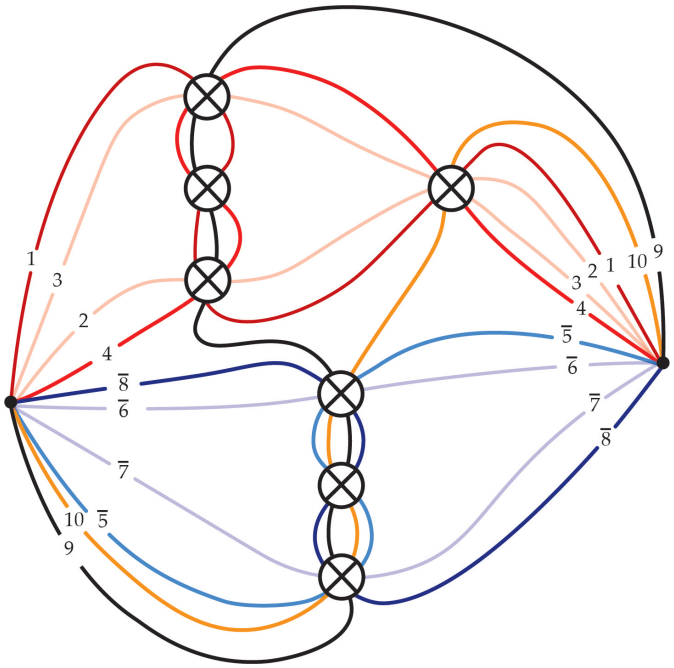

Figure 7.3: A perfect cross-cap drawing of Figure 7.1 with two additional edges.

For the ease of reading, we recall the following notations and concepts. In a two-vertex embedding scheme, each vertex is adjacent to all of the edges. This gives two-vertex embedding schemes a particularly simple form. Without loss of generality, one can number the edges so that the cyclic permutation around one of the vertices is the identity. Then the data of the embedding scheme just consists of the cyclic permutation around the other vertex, and the signature of the edges, and thus this amounts to a *signed cyclic permutation*: a cyclic permutation where each number is additionally endowed with a $+$ or $-$ sign. We also use an overline notation $\overline{i}$ to denote negative signs. Therefore, in what follows we freely identify a signed permutation and a two-vertex embedding scheme. Particularly, everything defined for a two-vertex embedding scheme is also defined for signed cyclic permutations and vice versa. For example, for a cyclic permutation π, $eg(\pi)$ refers to the Euler genus of the corresponding two-vertex embedding scheme to π.

If we trace each element of a signed permutation under the action of reversals, we obtain a cross-cap drawing of its corresponding embedding scheme, where each reversal corresponds to a cross-cap and each edge is an x-monotone curve, and in particular no edge enters twice the same cross-cap (see Figure 3.28 for an illustration and Section 3.8.2 for more detail).

This easily implies the inequalities: $\tilde{g}(\pi) \leq dcr(\pi) \leq d(\pi, id)$ where $d(\pi, id)$ is the reversal distance between π and the identity. It turns out that the inequality $\tilde{g}(\pi) \leq d(\pi, id)$ is central to the reversal distance theory, and the cases of equality are well understood (see Theorem 3.8.6). As we explained in Section 3.8.2, these arguments are natural from

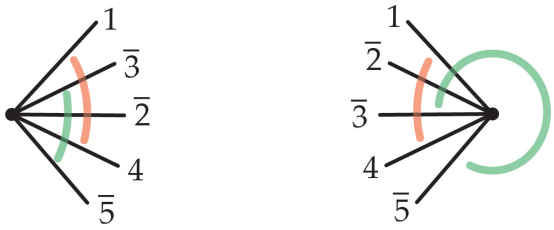

Figure 7.4: The wedges $\omega_{1,4}$ and $\omega_{3,5}$ depicted in orange and green, respectively.

the point of view of embedding schemes. Conversely, Theorem J can be reinterpreted in the setting of signed permutations as providing an extension of the Hannenhalli-Pevzner theory.

The proof of Theorem J consists of two steps which can readily be made algorithmic: we first *reduce* a signed permutation π to a simpler one $\pi|$ for which we can prove that $\tilde{g}(\pi|) = d(\pi|, id)$ (Lemma 7.4.1), then we devise a technique to *blow up* (Lemma 7.4.3) the cross-cap drawing of the reduced signed permutation $\pi|$, yielding a perfect cross-cap drawing of the original signed permutation.

7.2 Preliminaries

In this section, we introduce and recall some of the notions that we need for proving our results.

7.2.1 Loopless two-vertex embedding schemes

In this chapter, in particular, we deal with loopless two-vertex graphs with embedding schemes. Here, we provide lemmas and tools to deal with two-vertex embedding schemes that is similar to that provided in Section 5.2 for one-vertex embedding schemes.

For two edges a and b in a two-vertex embedding scheme, we denote by $a \cdot b$ the concatenation of a and b which we will interpret as a cycle. Denoting the vertices of the scheme by v_1 and v_2, and the cyclic permutation of edges around them by ρ_{v_1} and ρ_{v_2}, respectively, we define a *wedge* between a and b, $\omega_{a,b}$:

- If both a and b are negative, then $\omega_{a,b}$ contains all the half-edges in the interval (a, b) in both ρ_{v_1} and ρ_{v_2}.
- If at least one of them is positive, then $\omega_{a,b}$ contains all the half-edges in the interval (a, b) in ρ_{v_1} and (b, a) in ρ_{v_2}.

We say that a wedge *encloses* an edge if it contains both its half-edges or none of them. For example $w_{1,4}$ in Figure 7.4 encloses all the edges of the graph.

Recognizing types of curves in a two-vertex embedding scheme

The following lemmas allow us to recognize orienting and separating closed curves in loopless two-vertex embedding schemes.

Lemma 7.2.1. *For two edges a and b in a non-orientable loopless two-vertex embedding scheme, the cycle $a.b$ is orienting,*

- *if at least one of a and b is positive and $\omega_{a,b}$ contains exactly one end of all negative edges and encloses all the positive edges, or*
- *if both a and b are negative and their wedge contains exactly one end of all positive edges and encloses all the negative edges.*

Proof. We prove the lemma by contracting the edge a in order to obtain a one-vertex embedding scheme G'. Note that the topological type of the cycle formed by the edges a and b is the same as the loop b in G'. Denote the vertices of G by v_1 and v_2. When we contract a positive edge e in G, we obtain an embedding scheme G' with a single vertex w such that the cyclic permutation of the edges around w after contraction is $\rho_w = \rho_{v_1}\rho_{v_2}$ where the edge e has been removed from both cyclic permutations and the notation means that they have been concatenated at e (see the left picture in Figure 3.9). On the other hand, when we contract a negative edge e in G, $\rho_w = \rho_{v_1}\overline{\rho_{v_2}}$, where the edge e has been removed from both permutations, the signature of all the edges are reversed (the scheme does not have any loop) and the notation means that they have been concatenated at e (see the right picture in Figure 3.9). The ends of the loop b subdivide the half-edges around w into two sets. We can see that the half-edges in $\omega_{a,b}$ in G correspond to one of the sets of half-edges divided by b in ρ_w.

By Lemma 5.2.3, a loop o in a one-vertex non-orientable embedding scheme is orienting if and only if its ends *alternate* with the ends of all negative loops in the cyclic permutation around the vertex and *enclose* the ends of any positive loop; i.e., the ends of o does not alternate with the ends of any positive loop.

To prove the first case, without loss of generality we can assume that a is positive. We contract the edge a in G. Since the wedge $\omega_{a,b}$ in G contains exactly one end of each negative edge, the ends of the loop b alternate with the ends of negative loops in ρ_w. Similarly, since $\omega_{a,b}$ encloses all the positive edges, the ends of b enclose the ends of any positive loop in ρ_w. Therefore, b is orienting in G' and this implies that the cycle formed by a and b is orienting in G. For the proof of the second case in 1 we proceed identically by contracting the negative edge a. This finishes the proof. □

Lemma 7.2.2. *For two edges a and b in a loopless two-vertex embedding scheme, the cycle $a.b$ is separating if a and b have the same signature and $\omega_{a,b}$ encloses all the edges.*

Proof. We contract a. The loop b has positive signature in G' and since $\omega_{a,b}$ contains both ends of any edge inside it then the ends of b separate the ends of the other loops in G', i.e., the ends of no loop alternates with those of b. Such a loop is separating the surface, and therefore a and b form a separating cycle in G. □

See Figure 7.4 for an example of a separating (1.4) and an orienting cycle (3.5) in a two-vertex scheme.

Recognizing homotopic edges in a two-vertex embedding scheme

The following lemma helps in recognizing homotopic edges in a two-vertex embedding scheme.

Lemma 7.2.3. *Let (G, ρ, λ) be a loopless two-vertex embedding scheme with vertices v_1 and v_2. Two edges e and e' are homotopic if and only if they have the same signature, their ends are consecutive in ρ_{v_1} and ρ_{v_2} and,*

- *if they both have signature $+1$, they appear as ee' around one of the vertices and as $e'e$ around the other one and,*
- *if they both have signature -1, they appear as ee' around both vertices.*

Proof. To prove this lemma, it is enough to show that $e.e'$ bounds a disk on the surface (see Lemma 3.2.10). Figure 7.5 shows the disk whose boundary is comprised of these two edges in both cases. This finishes the proof. □

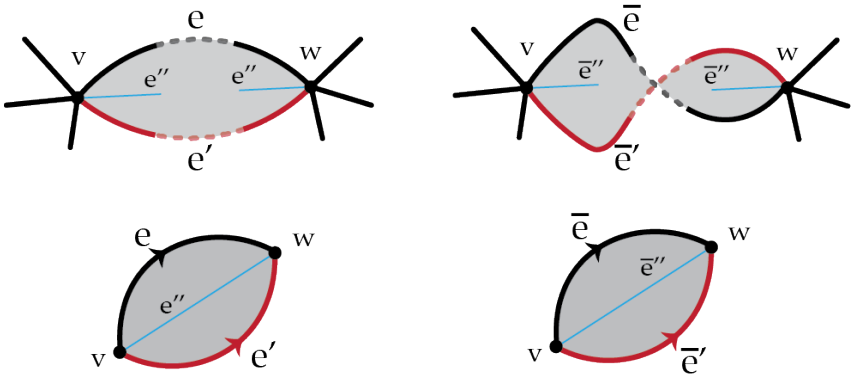

Figure 7.5: Homotopic edges in two-vertex embedding schemes.

Some notions from genome rearrangements

Here, we recall some of the notions in the signed reversal distance theory (see Sections 3.8.1 and 3.8.2) that we directly use in this chapter.

In a signed cyclic permutation π, a pair of consecutive integers i and $i+1$ is called a *reversible* pair if they have opposite signs in π. For a given reversible pair there exist two reversals σ, σ' such that i and $i+1$ become homotopic[1] in $\pi \cdot \sigma$ and in $\pi \cdot \sigma'$. Such reversals clearly make progress in sorting a permutation.

A crux in sorting a permutation appears when we are given with a permutation that contains *blocks*. We recall the definition of blocks that were introduced in Section 3.8.1. A *positive block* in a signed permutation is an interval $I = (\pi_i, \dots, \pi_j)$ where all the elements are positive, $\pi_i < \pi_j$, and all the integers in $[\pi_i, \pi_j]$ are contained in I. A *negative block* in a signed permutation is an interval $I = (\overline{\pi}_i, \dots \overline{\pi}_j)$ where all the elements are negative, $\pi_i > \pi_j$, all the integers in $[\pi_j, \pi_i]$ are contained in I. A block is *non-trivial* if it is not already sorted, i.e., it is not equal to $(\pi_i, \pi_i + 1, \dots \pi_j - 1, \pi_j)$ or to $(\overline{\pi}_i, \overline{\pi_i - 1}, \dots, \overline{\pi_j + 1}, \overline{\pi}_j)$. In both cases, we call π_i and π_j the frames of the block. A block is called *minimal* if it does not contain any block except itself.

We say that a signed permutation is *reduced* if it has no blocks. Given a signed permutation π, its reduced permutation $\pi|$ is a permutation in which we replace every minimal block with a single element of the same sign, and we iterate this process until we arrive at a reduced permutation.

7.2.2 Reversal distance and monotone cross-cap drawings

Here we recall the relation between reversal distance in genome rearrangements and cross-cap drawings, which we explained with more detail in Section 3.8.2. Signed permutations model genomes with a single chromosome in computational biology where they come endowed with the reversal distance. The reversal distance $d(\pi, id)$ is the smallest number d such that there exists a sequence $\{\pi = \pi^1, \pi^2, \dots \pi^d = id\}$ such that (π^i) and (π^{i+1}) differ by a signed reversal. We call such a (not necessarily minimizing) sequence, a *path* of signed permutations.

To any path of signed permutations $\{\pi^1, \pi^2, \dots \pi^d\}$ we can associate a cross-cap drawing. We place a source vertex at $(-1, n/2)$ and a terminal vertex at $(d+1, n/2)$. Edges will be x-monotone piece-wise linear curves between these two vertices, and at each crossing between two such curves we introduce a cross-cap. The edge j emanates from $(-1, n/2)$ to $(0, \pi^1_j)$, for each $k \leq d$ it passes through $(k, \pi^j_k) \in \mathbb{R}^2$, and finally it connects (d, π^d_j) to the terminal vertex at $(d+1, n/2)$. In the remaining of this chapter, we often forget about the vertices $(-1, n/2)$ and $(d+1, n/2)$ in our illustrations as they play no role, and we always assume that π^d is the identity (see Figure 3.28 in Chapter 3).

In Section 7.4, we use the algorithm of Hannenhalli and Pevzner for signed permuta-

[1]Note that two edges being homotopic in a two-vertex embedding scheme coincides with the notion of edges being *parallel* defined in Section 3.8.2.

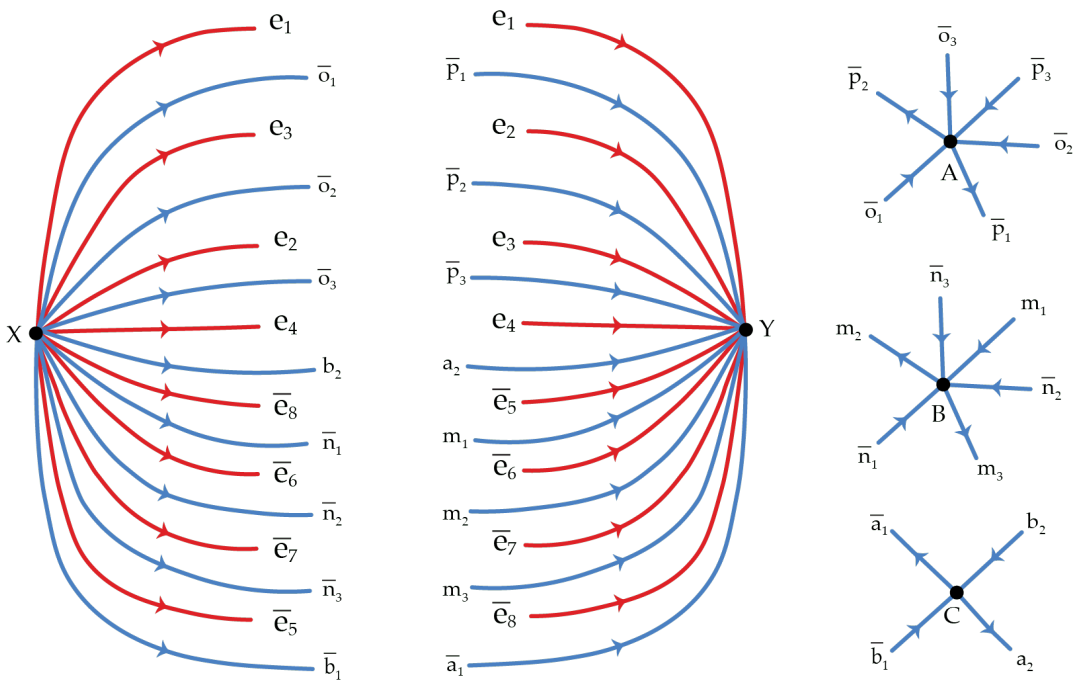

Figure 7.6: A pseudo-triangulation of N_5 admitting no perfect cross-cap drawing.

tions without blocks which gives us monotone cross-cap drawings for the corresponding embedding schemes. We recall the following theorem.

Theorem 3.8.6. *If a signed permutation π is non-orientable and has no non-trivial blocks then $d(\pi, id) = eg(\pi)$, and the Hannenhalli-Pevzner algorithm gives a sequence of reversals of this optimal length.*

7.3 The counterexample

In this section, we provide a family of two-vertex embedding schemes that do not admit a perfect cross-cap drawing. Then we provide an explicit pseudo-triangulation of N_5 (depicted in Figure 7.6), disproving Conjecture 3.

Remark 7.3.1. *If an embedding scheme G has one positive and one negative block, then so does its flipped version, therefore, we do not need to account for the possible flip in the proof of Theorem I.*

In order to prove Theorem I, we rely on Remark 7.3.1 and Lemma 7.3.2. From here till the end of this chapter, by $f(G)$ and $e(G)$ we refer to the number of faces and edges of the embedding scheme G, respectively.

Lemma 7.3.2. *Let G be an embedding scheme that consists of a non-trivial positive block A and a non-trivial negative block B, then $\tilde{g}(G) = \tilde{g}(A) + \tilde{g}(B) - 1$.*

Proof. Assume that the positive block has edges labelled $A = \{e_1, \dots e_k\}$ and the edges of the negative block are $B = \{\overline{e}_{k+1}, \overline{e}_{k+2}, \dots \overline{e}_{k+l}\}$. Notice that $f(G) = f(A) + f(B) - 1$, indeed the face e_1, e_k and the face $\overline{e}_{k+1}, \overline{e}_{k+l}$ are the outer faces of A and B, and they merge to become the face $e_1, \overline{e}_{k+1}, e_k, \overline{e}_{k+l}$. Hence by Euler's formula $eg(G) = eg(A) + eg(B) + 1$. We know that $\tilde{g}(A) = eg(A)+1$ and $\tilde{g}(B) = eg(B)+1$ (Lemma 3.3.2). On the other hand, a cycle in G that contains one edge from A and one edge from B is one-sided, therefore G is a non-orientable, hence $eg(G) = \tilde{g}(G)$. All in all we can conclude

$$\tilde{g}(G) = eg(G) = eg(A) + eg(B) + 1 = \tilde{g}(A) - 1 + \tilde{g}(B) - 1 + 1 = \tilde{g}(A) + \tilde{g}(B) - 1$$

as claimed. □

We now have all the tools to prove Theorem I; we refer to Figure 7.1 for an example to help follow the proof.

Proof of Theorem I. Let G be a concatenation of a positive block A with frames a_1 and a_2 and a negative block B with frames b_1 and b_2. Let us assume that ϕ is a perfect cross-cap drawing of G. From Lemma 7.2.1 we derive that $a_1 \cdot b_1$ and $a_1 \cdot b_2$ are orienting curves, hence by Lemma 3.4.2 each of them enters each cross-cap once. Lemma 7.2.2 implies that $b_1 \cdot b_2$ is separating. Therefore by Lemma 3.4.2, b_1 and b_2 enter the same cross-caps and do not enter any cross-cap that a_1 enters. Similarly, $a_1 \cdot a_2$ is separating and hence they enter the same cross-caps and no cross-cap that b_1 and b_2 enter. Then A is drawn with $\tilde{g}(A)$ cross-caps and B is drawn with $\tilde{g}(B)$ cross-caps that are disjoint from the cross-caps that A entered. But by Lemma 7.3.2 the non-orientable genus of G is $\tilde{g}(A) + \tilde{g}(B) - 1$. Therefore, there are not enough cross-caps available to draw both A and B. This concludes. □

Corollary 7.1.1 follows at once as we can always add edges and vertices to a scheme to triangulate it without adding loops nor changing its genus, and any perfect cross-cap drawing of the triangulation restricts to a cross-cap drawing of the scheme. We provide in Figure 7.6 an example of such a pseudo-triangulation.

7.4 Perfect drawings for most two-vertex graph embeddings

We say that a cross-cap drawing is *fantastic* if it is perfect and every edge enters at least one cross-cap.

Lemma 7.4.1. *Every reduced loopless two-vertex graph embedding scheme that is different from* (1), $(1, \overline{2})$ *or* $(1, \overline{3}, \overline{4}, 2)$ *admits a fantastic cross-cap drawing.*

The proof is based on an induction and an exhaustive analysis of all the loopless two-vertex embedding schemes of genus 2 and 3, as pictured in Figure 7.7. We first prove the following lemma.

Lemma 7.4.2. *Let G be an orientable scheme of non-orientable genus $\tilde{g}(G)$ that has no non-trivial block. We can add an edge to this embedding scheme such that we obtain a non-orientable scheme of genus $\tilde{g}(G)$ without any block.*

Proof. Let us assume that the signatures of the edges are positive. Choose an edge f. Add a negative edge e consecutive to f (at both vertices), such that in the cyclic permutation of one vertex we see ef and in the other vertex we see fe. Let us call the new embedding scheme by G'. The scheme G' is non-orientable. Since G is orientable, we know that $\tilde{g}(G) = eg(G) + 1$ (Lemma 3.3.2). Since $e(G') = e(G) + 1$, it is enough to show that $f(G') = f(G)$. By looking at the face cycles of both schemes, we can see that all face cycles are intact except a face $(i, f, j, \dots)$ in G that is turned to a face cycle $(i, e, f, e, j, \dots)$ (i and j are two adjacent edges to f in the cyclic permutation of the vertices). Since the edge e has an opposite signature compared to the edges of G, e cannot create a block in G'. The proof for the case where all the edges of G are negative can be obtained by flipping. This finishes the proof. □

Proof of Lemma 7.4.1. Let π be the signed permutation associated to the embedding scheme. If the embedding scheme is orientable, we first add one edge without changing its genus to make it non-orientable while keeping it reduced (this is possible by Lemma 7.4.2). Now, since it is reduced and non-orientable, by Theorem 3.8.6, the Hannenhalli-Pevzner algorithm provides a path in the reversal graph $\{\pi = \pi^1, \pi^2, \dots \pi^g = id\}$, where g is the non-orientable genus of π. We distinguish cases depending on the value of g.

If $g \geq 3$, we consider the sub-path $\{\pi^1, \pi^2, \dots, \pi^k\}$, with $g - k = 3$ and realize this sub-path as a cross-cap drawing ϕ as described in Section 7.2. Notice that if there exists a fantastic cross-cap drawing ϕ' for π^k, we can concatenate ϕ with ϕ' to obtain a fantastic cross-cap drawing for π. Now π^k is a reduced signed permutation of non-orientable genus 3. The proof then proceeds via an exhaustive case analysis. Without loss of generality, we can assume that

- There is no non-trivial block in π^k since that is preserved by the Hannenhalli-Pevzner algorithm.
- There are no homotopic edges since, for any collection of homotopic edges, one can remove all but one and add them in the end identical to the remaining one.
- There is at least one edge of signature -1 and one edge of signature $+1$ in π^k. Otherwise, π^k is orientable, which is impossible since by Theorem 3.8.6, the Hannenhalli-Pevzner algorithm preserves non-orientability.

- π^k is maximal while preserving these three properties. Indeed, otherwise, we can add edges, draw the resulting scheme and remove these superfluous edges at the end.

With these simplifying assumptions at our disposal, we can exhaustively enumerate all the genus 3 embedding schemes matching these assumptions. The numbers are small enough that this can be done by hand, we ran a computer search for safety. One obtains that all the maximal schemes only have faces of degree 4, and thus have six edges: for all reduced schemes with some faces of degree higher than four, one can always add an edge within a face while keeping the fact that it is reduced. Then, there are exactly eight loopless two-vertex embedding schemes matching our assumptions, their signed permutations are: $(1, \overline{6}, 5, \overline{4}, 3, \overline{2})$, $(1, \overline{6}, \overline{3}, 5, 4, \overline{2})$, $(1, \overline{3}, \overline{6}, 5, \overline{4}, 2)$, $(1, 5, \overline{3}, \overline{6}, 4, 2)$, $(1, \overline{4}, 6, 3, \overline{5}, 2)$, $(1, \overline{6}, 3, \overline{4}, 5, \overline{2})$, $(1, \overline{4}, 6, \overline{2}, 5, \overline{3})$, and $(1, \overline{4}, \overline{6}, \overline{2}, 5, 3)$. Fantastic drawings of each of them are provided in Figure 7.7.

If $g = 2$, there are exactly three reduced schemes: $(1, \overline{3}, \overline{4}, 2)$, $(1, \overline{2}, 3, \overline{4})$ and $(1, 3, \overline{2})$. Fantastic drawings of the second and third case (or rather its flipped version) are provided in Figure 7.7.

If $g = 1$, there is a single reduced scheme: $(1, \overline{2})$.

If $g = 0$, there is a single reduced scheme: (1). □

In order to prove Theorem J, our strategy is to first look for a fantastic cross-cap drawing for the reduced graph of an embedding scheme. Then, to obtain a drawing for the initial graph, we need to bring back the blocks that we replaced and extend the drawing to the edges of the block. This is achieved via the following blowing-up operation.

Blowing up a cross-cap. Let ϕ be a fantastic drawing for G. By the definition of fantastic drawing, any reduced edge enters at least one cross-cap. Let e be a reduced edge that corresponded to a block X with frames a and b and let $\mathfrak{c}$ be a cross-cap in ϕ that e enters. By Lemma 3.3.2, X needs an odd number of cross-caps to be drawn, exactly $eg(X) + 1$. We replace $\mathfrak{c}$ by $eg(X) + 1$ cross-caps and we draw the frames of X, a and b as follows. We draw a following e thoroughly. To draw b we follow e except that we make b enter the new $eg(X) + 1$ cross-caps in the reversed order that e enters. All the other edges outside of X that were entering e are now drawn in the same way as a or b depending on how they were crossing e at $\mathfrak{c}$. Finally we remove e (see Figure 7.8).

Repeatedly blowing up a drawing of a reduced permutation $\pi|$ yields a drawing for all edges of π except for the edges inside the blocks. The following lemma shows that these edges can be added in this cross-cap drawing.

Lemma 7.4.3. *Let π be a signed permutation on n elements such that all the elements form a non-trivial minimal negative block. The associated embedding scheme admits a*

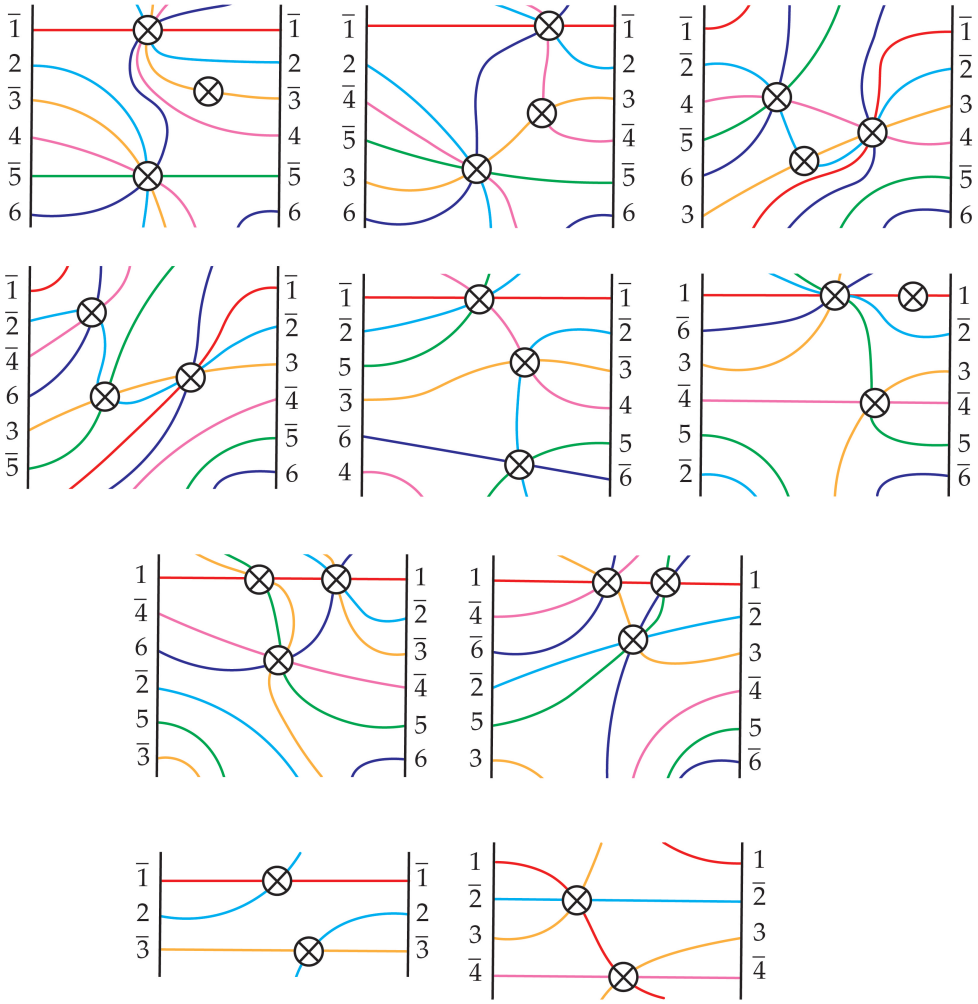

Figure 7.7: Fantastic drawings of genus-2 and genus-3 embedding schemes (these are cylindrical drawings: the top is identified to the bottom).

perfect cross-cap drawing in which the frames of π*, i.e., elements* 1 *and* n*, enter all the cross-caps but in opposite order.*

Proof. Let us denote by $\tilde{g}(\pi) = eg(\pi) + 1$ the non-orientable genus of the associated embedding scheme. We know that $\pi_1 = \overline{n}$ and $\pi_n = \overline{1}$. Let us define π' from π by replacing $\overline{1}$ by $n+1$. The following lemma is proved using the Hannenhalli-Pevzner algorithm and Theorem 3.8.6.

Lemma 7.4.4. *The optimal number of reversals to go from* π' *to the permutation* $(2, 3, \ldots, n+1)$ *is* $\tilde{g}(\pi') = \tilde{g}(\pi)$*. There exists a sequence of such reversals such that no reversal is applied on the element* $n+1$*.*

Proof. Note that the associated embedding scheme to π' is a non-orientable scheme and therefore $\tilde{g}(\pi') = eg(\pi')$. The number of edges in π and π' are equal therefore to show

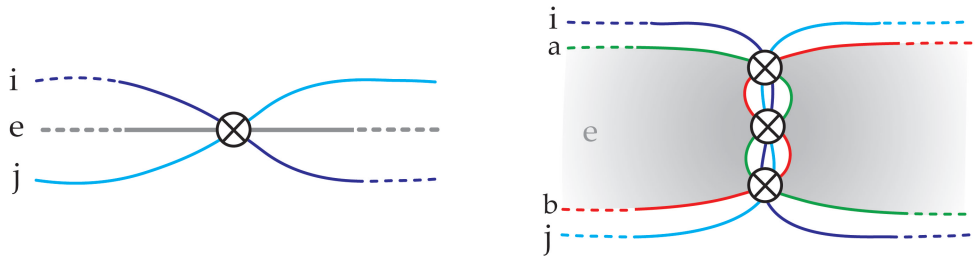

Figure 7.8: Blowing up a cross-cap to draw the frames of a block.

that $eg(\pi') = \tilde{g}(\pi) = eg(\pi) + 1$, it is enough to show that $f(\pi') = f(\pi) - 1$. Let $\pi = (\overline{n}, \dots, \overline{2}, \dots, \overline{j}, \overline{1})$. Then $\pi' = (\overline{n}, \dots, \overline{2}, \dots, \overline{j}, n+1)$. The embedding scheme π has a face $f_1 = (0, n)$ and $f_2 = (2, n, j, \dots)$. Replacing -1 with $n+1$, the faces f_1 and f_2 merge to a single face $f = (2, n+1, n, n+1, j, \dots)$ in π' (see Figure 7.9). The other faces in π' are the same as the faces in π other than f_1 and f_2. This implies that π' has one face less than π.

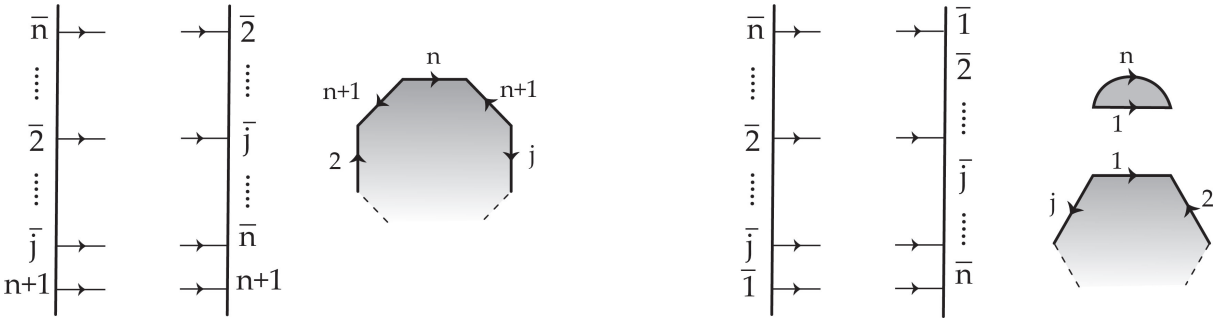

Figure 7.9: Faces of π' (left) and π (right).

Furthermore, π' does not contain any block and therefore it is reduced. It is non-orientable by construction. Having a reversible pair $(i, i+1)$, there are two reversals that make i and $i+1$ homotopic. By running the Hannenhalli-Pevzner algorithm on π' and choosing the reversal at each step that does not reverse $n+1$ we obtain a desired sequence of reversals. This finishes the proof of Lemma 7.4.4. ■

Now, the sequence of reversals of Lemma 7.4.4 gives us a cross-cap drawing for π. By Lemma 7.2.1, the edges n and $n+1$ form an orienting cycle, and thus together they have to enter all the $g(\pi')$ cross-caps exactly once. We know that the edge $n+1$ does not enter any cross-cap in this drawing which implies that n enters all the cross-caps exactly once. We can obtain a cross-cap drawing for π from this drawing for π' by drawing 1 entering the cross-caps that n enters with the opposite order as depicted in Figure 7.10. This finishes the proof of Lemma 7.4.3. □

We are now finally ready to prove Theorem J.

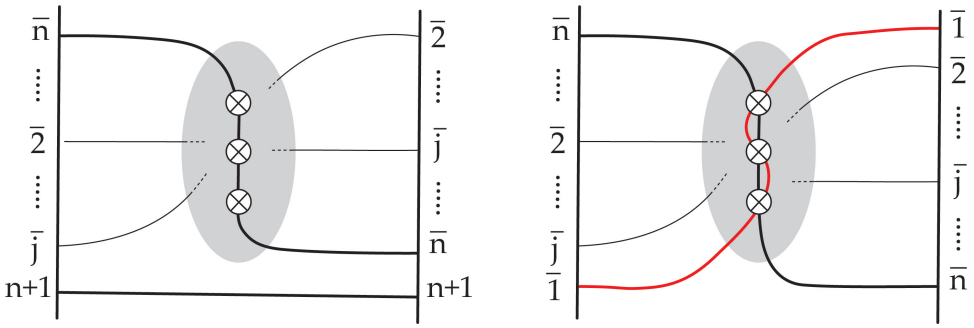

Figure 7.10: From a cross-cap drawing of π' to a cross-cap drawing of π.

Proof of Theorem J. Let G denote an embedding scheme for a loopless two-vertex graph of non-orientable genus g. We denote by G' the reduced scheme, i.e., the scheme obtained after recursively replacing minimal blocks with a curve of the corresponding sidedness. If G' is exactly one of the two graphs depicted in Figure 7.2, the conclusion of the theorem holds. If G' consists of a single edge, up to flipping G we can assume that this edge has negative signature. Then this edge can be drawn in the natural way with a single cross-cap. In all the other cases, by Lemma 7.4.1, G' admits a fantastic cross-cap drawing.

In order to finish the proof, we explain how to inductively put back minimal blocks of G in the place of the corresponding reduced edge in G'. If A is such a minimal block and is negative and we denote by e the corresponding reduced edge, e is one-sided and thus goes through at least one cross-cap. We blow up this cross-cap, replacing it by exactly the odd number of cross-caps required to draw A. By Lemma 7.4.3, since A is minimal, it is reduced and thus it can be drawn using these blown-up cross-caps, in such a way that the frames of the block enter the blown-up cross-caps in opposite orders. This process is pictured in Figure 7.10. If A is a positive minimal block and e is the corresponding reduced edge, since the drawing of G' is fantastic, we know that e enters at least two cross-caps. We first make the entirety of A enter the first of these cross-caps. Thus, there remains to draw the flipped version of A, which is now a negative block. This is achieved as before by blowing-up the second cross-cap, and appealing to Lemma 7.4.3 to draw the negative block within the space bordered by the two frames.

This process shows how to obtain a perfect cross-cap drawing of G from a fantastic cross-cap drawing of the reduced graph G', concluding the proof. □

Chapter 8

Universal families of arcs and curves on surfaces

Summary. The main goal of this chapter is to investigate the minimal size of families of curves on surfaces with the following property: a family of simple closed curves Γ on a surface *realizes all types of pants decompositions* if for any pants decomposition of the surface, there exists a homeomorphism sending it to a subset of the curves in Γ. We provide bounds for different instances of this problem. Also we investigate a similar concept of universality for triangulations of polygons.

The results in this chapter were obtained with Arnaud de Mesmay and Hugo Parlier. They appear in [B], which has been accepted in Israel Journal of Mathematics.

8.1 Introduction

Recall that we call a sphere with 3 boundary components a *pairs of pants* and a *pants decomposition* of a surface is an arrangement of simple closed curves on it that cuts it into pairs of pants. Unlike the rest of the chapters in which we focused mostly on non-orientable surfaces, here our surfaces are orientable, and hence are determined by their genus g and number of punctures n. Since the Euler characteristic of a pairs of pants is -1, a surface can be decomposed into pants if its Euler characteristic is negative. Therefore, we require that the Euler characteristic $(= 2 - 2g - n)$ of our surfaces be negative.

In this chapter, we only consider pants decompositions up to homeomorphisms: two pants decomposition of a surface have the same *type* or belong to the same *homeomorphism class* if there exist a homeomorphism of the surface that maps one to the other. The type of a pants decomposition is entirely determined by the trivalent graph encoding

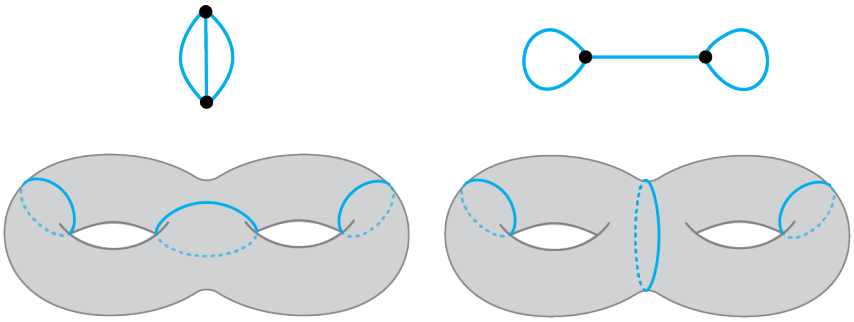

Figure 8.1: The two homeomorphism classes of pants decomposition on a closed genus two surface, and their associated intersection graphs. The family of four curves drawn in the two pictures are enough to build both pants decompositions.

the adjacencies of the different pants that it is made of, i.e., its intersection graph (see Section 3.6.3). For example, there are two types of pants decompositions in genus 2, which correspond to the two trivalent graph on two vertices (see Figure 8.1).

One of the main objects of study in this chapter are families of curves which realize all topological types of pants decompositions. A set of curves Γ on a surface S is said to be a universal family (for pants decompositions) if for any type of pants decomposition of the surface, there exists a homeomorphism sending it to a subset of the curves in Γ: we say that Γ *realizes all types of pants decompositions* of the surface. For example, in the genus 2 case, there is a universal family of size 4, pictured in Figure 8.1.

8.1.1 Our results

For surfaces of genus g, there are $g^{\Theta(g)}$ homeomorphism classes of pants decompositions (see Lemma 3.6.1), and thus, by taking an arbitrary pants decomposition in each homeomorphism class and the $3g-3$ curves it is made of, there is a trivial upper bound of $g^{O(g)}$ on the minimal size of a family of curves that realizes all types of pants decompositions. Our first result is to improve on this trivial bound to bring it to a singly-exponential dependency.

Theorem K. *Let M be a closed orientable surface of genus g, and Γ be a minimal size universal family for pants decompositions. Then*

$$|\Gamma| \leq 3^{2g-1} \text{ and } |\Gamma| = \Omega(g^{4/3-\varepsilon})$$

for any $\varepsilon > 0$.

The tantalizing gap between the exponential upper bound and the polynomial lower bound is the main open problem that we would like to advertise with this chapter.

The upper bound in Theorem K follows from an upper bound for the same problem on spheres with punctures. There, one can distinguish between the cases of labelled or unlabelled punctures, which radically changes the bounds: in the first case we consider homeomorphisms keeping the punctures fixed, while in the second case the punctures are allowed to be permuted. Our results are as follows.

Theorem L.
- *Let $S_n \coloneqq M_{0,n}$ be a sphere with n labelled punctures, and Γ be a family of curves with minimal size that realizes all types of pants decompositions. Then*

$$2^{n-1} - n - 1 \leq |\Gamma| \leq 3^{n-1}.$$

- *Let $S_n \coloneqq M_{0,n}$ be a sphere with n unlabelled punctures, and Γ be a family of curves with minimal size that realizes all types of pants decompositions. Then*

$$|\Gamma| = O(n^2) \text{ and } |\Gamma| = \Omega(n \log n)$$

In Section 8.6, we provide universal families for surfaces of small genus with labelled punctures which suggest that the upper bound in Theorem K can be improved upon, but the approach seems unlikely to provide a subexponential bound.

Perhaps surprisingly (at least to us), the bound in the second item of Theorem L can be improved if instead of asking for families that realize all types of pants decompositions, we merely ask for a family of curves realizing all the types of pairs of pants $P \subseteq S_n$ (and not the whole decomposition). The homeomorphism class of such a pair of pants is entirely determined by the triplet (k_1, k_2, k_3) counting the number of punctures in the three components it separates, where $k_1 + k_2 + k_3 = n$. There are $O(n^2)$ such triplets, but we provide a random construction that achieves a better bound:

Theorem M. *There exists a family of simple closed curves of size $O(n^{4/3} \log^{2/3} n)$ on the sphere with n punctures S_n that realizes all types of pants.*

Finally, one can phrase similar universality conditions for families of edges connecting punctures. For example, in the planar case, given a polygon π_n with n unlabelled vertices, one can look for families of edges realizing all the homeomorphism classes of triangulations of π_n, or merely all the homeomorphism classes of triangles in π_n. In that case, the situation is constrained enough that we can provide upper bounds and lower bounds which are almost tight.

Theorem N. *In a polygon with n vertices, the minimal size of a family of edges E realizing all triangulations satisfies $|E| = O(n^2)$ and $|E| = \Omega(n^{2-\varepsilon})$ for any $\varepsilon > 0$. The minimal size of a family of edges E realizing all types of triangles satisfies $|E| = O(n^{4/3} \log^{2/3} n)$ and $|E| = \Omega(n^{4/3})$.*

One can similarly investigate families of edges realizing all one-vertex (or several-vertex) triangulations of surfaces. The techniques that we use for pants decompositions apply equally well for that setting. For the sake of avoiding redundancies, we do not include the corresponding results in this work.

A Motivation. An important motivation for the work in this chapter is related to the question on the existence of a universal shortest path metric, mentioned in the introduction of this work:

Question 1. *Given a surface S of genus g, does there exist a Riemannian metric on S such that any simple graph embeddable on S can be embedded so that the edges are shortest paths on S?*

The connection between Question 1 and the problems studied in this chapter can be easily explained as follows. Given a pants decomposition of a surface of genus g, it is easy (see for example [50, Section 5]) to subdivide each curve a constant number of times and connect the resulting vertices with $O(g)$ edges so that we obtain a connected embedded graph. Furthermore, one can ensure that this graph has the property that if it is embedded so that the edges are shortest paths, then the original pants decomposition can be realized so that each curve consists of a constant number of shortest paths. Therefore, if the answer to Question 1 is affirmative, it means that there exists a metric on S so that *any* pants decomposition can be realized so that each curve consists of a constant number of shortest paths. Furthermore, by a standard cut and paste argument, shortest paths pairwise cross at most once. Therefore, an affirmative answer to Question 1 would imply that there exists a family of curves realizing all types of pants decompositions on S so that any pair of curves crosses a constant number of times. But then, such a family of curves would need to have polynomial size [64, 40] in g. Thus if we could prove that any family realizing all types of pants decompositions must have superpolynomial size, this would provide a negative answer to Question 1. Our result in Theorem K provides partial results in this direction.

There are further motivations for the study of universal families of curves coming from Teichmüller theory. Since those involve concepts that we have not defined in this thesis, we refer the interested reader to the introduction of our article [B].

Finally, let us mention that there are various related problems in combinatorics and graph theory where one investigates a given class of combinatorial objects and aims at finding a universal family that contains the entire class in some way. Such investigations date back at least to Rado [65]. Depending on the precise notions considered, the size of this universal object might or might not be exponentially larger than the size of the objects in the class. A basic example of this is a *k-universal permutation*, or *k-superpattern* on n symbols, which is a permutation containing all the possible patterns on k symbols

as subsequences. The smallest known k-superpatterns have size quadratic in k, but the best possible constant is still unknown, see for example Miller [55]. In contrast, one might consider *superpermutations*, which are strings on n letters where all the permutations of size n appear as substrings: one can easily prove that such superpermutations necessarily have a size exponentially large in n. Many universality notions for graphs have been investigated, a famous problem being the optimal size of a universal graphs containing all graphs of a fixed size as an induced subgraph, see Alon [2] and the references therein. Perhaps closest to this work is a series of recent papers [10, 28, 35, 37] on universal graphs that contain all planar or bounded genus graphs of a fixed size as a subgraph, an induced subgraph or a minor: such universal graphs only have polynomial size, but the problems that we study in this chapter do not seem to be amenable to these techniques.

Outline. This chapter is structured as follows. After a short preliminary section, we treat the case of unlabelled spheres and polygons in Section 8.3. In Section 8.4, we investigate spheres with labelled punctures. Results on the asymptotic growth in terms of genus for closed surfaces of universal families are in Section 8.5. In the final section, we show how to adapt the results for punctured spheres to surfaces of small genus. While this last part is technical, with somewhat incremental progress, it illustrates how adding genus increases the complexity of the problem, shedding light on the size of the gap in Theorem K.

8.2 Preliminaries and notations

Recall that we denote by $M_{g,n}$ an orientable surface of genus g with n punctures (where n can be zero). In this chapter, we usually denote a sphere with n punctures by S_n. Furthermore, all of our surfaces have negative Euler characteristic, i.e., $2g + n > 2$. A *pair of pants* (or just *pants*) is a topological surface homeomorphic to a sphere with 3 punctures, and a *pants decomposition* is a set of simple and disjoint curves cutting a surface into a family of pairs of pants.

In this chapter, we will be working with different surfaces and some of their substructures (e.g., triangulations, pairs of pants and pants decompositions). We denote the n-sided polygon with π_n, which we consider up to homeomorphism which are allowed to rotate the punctures. A *triangle* on π_n is the homeomorphism class of a triangle with endpoints on distinct vertices of π_n. A *triangulation* of π_n is a maximal set of interior disjoint triangles in π_n.

We consider two kinds of homeomorphisms for surfaces: homeomorphisms that fix punctures pointwise or globally (i.e., that can permute them). We will refer to the first case as the *labelled* case and the second case as the *unlabelled* case.

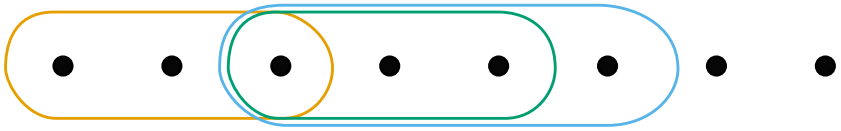

Figure 8.2: A sphere with eight punctures with a choice of curves $\gamma_{1,4}$, $\gamma_{3,6}$ and $\gamma_{3,7}$.

8.3 Unlabelled punctures

8.3.1 Unlabelled punctures: Realizing pants and triangles

In this section, we only consider planar surfaces, i.e., the polygon π_n and the surface S_n. We number the vertices/punctures (in consecutive order for the polygon) from 1 to n. We are working in the unlabelled setting, where the punctures are indistinguishable. For any pair of pants $P \subseteq S_n$, the subsurface $S_n \setminus P$ has at most three components, each containing some punctures of S_n and a component of the boundary ∂P. Therefore, the *topological type* of a pair of pants $P \subseteq S_n$ is the triple (k_1, k_2, k_3) such that the three components of $S_n \setminus P$ have k_1, k_2 and k_3 punctures (we adopt the convention to not count the boundary ∂P in k_1, k_2 and k_3 so that $k_1 + k_2 + k_3 = n$). Without loss of generality, we always assume that $k_1 \leq k_2 \leq k_3$. We say that a family of simple closed curves Γ *realizes all types of pants* if for any topological type, there exist three disjoint curves in Γ bounding a pair of pants that realizes that type.

8.3.1.1 Upper bounds

Theorem M. *There exists a family of simple closed curves of size $O(n^{4/3}\log^{2/3} n)$ on the sphere with n punctures S_n that realizes all types of pants.*

Proof. In this proof, we work with arithmetic modulo n and the interval notations are also modulo n: for example $[|n-2, 2|] = \{n-2, n-1, n, 1, 2\}$.

The proof is based on a random construction. Let us denote by $\gamma_{i,j}$ a simple closed curve separating all the punctures in $[|i, j-1|]$ from the others, and such that $\gamma_{i,j}$ and $\gamma_{k,\ell}$ are disjoint whenever $[|i, j-1|]$ and $[|k, \ell-1|]$ are disjoint, see Figure 8.2. If $i = j$ we take a simple contractible curve disjoint from all the others.

We set $p = c\log^{1/3} n/n^{1/3}$ for a constant c to be determined later, and define a random set $S \subseteq [|1, n|]$ by putting each integer in $[|1, n|]$ in S with probability p. Then by Chernoff bounds [56, Theorem 4.4], the set S has size at most $2cn^{2/3}\log^{1/3} n$ with probability at least $1 - e^{-cn^{2/3}\log^{1/3} n/3} > 1/2$ for big enough c. We define $\Gamma := \{\gamma_{i,j} \mid (i,j) \in S^2\}$. Now Γ has size at most $4c^2n^{4/3}\log^{2/3} n$ with probability at least $1/2$.

We now show that there is a nonzero probability that Γ realizes all types of pants. Let (k_1, k_2, k_3) be a topological type (thus we have $k_1 + k_2 + k_3 = n$). For i, j, k three integers in $[|1, n|]$, then the three curves $\gamma_{i,j}$, $\gamma_{j,k}$ and $\gamma_{k,i}$ bound a pair of pants $P_{i,j,k}$, which is of

type (k_1, k_2, k_3) if and only if $j - i = k_1$ and $k - j = k_2$. So for $i \in [|1, n|]$, we denote by $X_i^{k_1,k_2,k_3}$ the random variable indicating the event that $\{i, i + k_1, i + k_1 + k_2\} \subset S$, which happens with probability at least p^3 (note that if $k_1 = 0$ or $k_2 = 0$, the probability is higher). Then the probability that (k_1, k_2, k_3) is not realized is equal to

$$P(\sum_i X_i^{k_1,k_2,k_3} = 0) \leq (1 - p^3)^n \leq e^{-p^3 n} \leq \frac{1}{4n^2}$$

for big enough c. Note that this probability does not depend on (k_1, k_2, k_3), and therefore by the union bound there is a probability at least $1/4$ that all types of pants are realized. Since $1/4 + 1/2 < 1$, with nonzero probability we have the correct bound on the size of Γ and it realizes all types of pants, concluding the proof. □

Similarly to the problem of realizing all types of pants, we can look for families of edges realizing triangles in a polygon π_n, where the goal is to realize all types of triangles T. A type of triangle is determined by a triple (k_1, k_2, k_3) such that the three components of $\pi_n \setminus T$ contain respectively k_1, k_2 and k_3 vertices of the polygon π_n. Now, the exact same proof provides the following, which mirrors Theorem M.

Lemma 8.3.1. *There exists a family of edges on π_n of size $O(n^{4/3} \log^{2/3} n)$ realizing all types of triangles.*

8.3.1.2 Lower bounds

The bound obtained in Lemma 8.3.1 is sharp up to logarithmic factors.

Lemma 8.3.2. *For a polygon π_n, any family realizing all types of triangles has size at least $\Omega(n^{4/3})$.*

Proof. The vertices of the polygon π_n and the edges in a family E realizing all types of triangles form a graph. It is known (see for example Rivin [67]) that any graph with $|E|$ edges has $O(|E|^{3/2})$ triangles (i.e., cycles of length 3). Therefore, in order to realize $\Theta(n^2)$ types of triangles, the graph must have at least $\Omega(n^{4/3})$ edges. □

Remark 8.3.3. *While we believe that the bound for types of pants in Theorem M to be roughly sharp, as in Lemma 8.3.1, the same argument does not apply. The only lower bound we know for the number of curves needed to realize all topological types of pants on S_n with unlabelled punctures is the trivial lower bound of $\Omega(n)$.*

8.3.2 Unlabelled punctures: Realizing pants decompositions and triangulations

We now turn our attention to pants decompositions and triangulations. The *topological type* of a pants decomposition or triangulation is its homeomorphism type, and in the

unlabelled case, it is completely determined by the trivalent tree encoding the adjacencies of the pairs of pants/triangles (see the bottom picture in Figure 3.24 for an example). As before, we say that a family of curves Γ *realizes all types of pants decompositions* if for any topological type of pants decomposition, there exists a subset of $3g-3$ curves in Γ inducing that topological type. Therefore, it is trivial to realize all types of pants decompositions in S_n using $O(n^2)$ curves, respectively edges: one can simply number the punctures from 1 to n arbitrarily and for any pair (i,j), take a curve running around the punctures from i to j, as pictured in Figure 8.2. Likewise, one can realize all types of triangulations in π_n using $O(n^2)$ edges. For polygons, we can prove an almost matching lower bound.

Lemma 8.3.4. *For any $\varepsilon > 0$, any family of edges realizing all triangulations of π_n has $\Omega(n^{2-\varepsilon})$ edges.*

Proof. The proof follows the same idea as the proof of Lemma 8.3.2. Any family of edges realizing all triangulations defines a graph with $|E|$ edges. Since the family of edges realizes all types of triangulations, in particular it realizes all types of triangles, 4-cycles, or more generally ℓ-cycles. The type of an ℓ-cycle C is determined by the tuple $(k_1,\dots,k_\ell)$ of vertices in each of the connected components of $\pi_n \setminus C$, and thus there are $\Theta(n^{\ell-1})$ of them. Now, any graph with $|E|$ edges has at most $O(|E|^{\ell/2})$ ℓ-cycles, and thus $|E| = \Omega(n^{\frac{2\ell-1}{\ell}})$. The result follows by taking ℓ arbitrarily big. □

Lemmas 8.3.1, 8.3.2 and 8.3.4 prove Theorem N.

Here again, this proof technique fails for pants decompositions. The best lower bound we can provide is the following.

Lemma 8.3.5. *Any family of curves realizing all types of pants decompositions of S_n has size at least $\displaystyle\sum_{i=2}^{\lfloor \frac{n}{2} \rfloor} \lfloor \tfrac{n}{i} \rfloor = \Omega(n \log n)$.*

Proof. Let T be a trivalent tree with n leaves associated to a pants decomposition P of S_n, in which each internal vertex corresponds to a pair of pants and each edge corresponds to a closed curve in P. For each curve γ in P, we say that γ **bounds** $i \leq \lfloor \frac{n}{2} \rfloor$ punctures if its corresponding edge in T separates a sub-tree that has exactly i leaves. We prove the following claim.

Claim. For each $i \leq \lfloor \frac{n}{2} \rfloor$, there exists a trivalent tree T_i with n leaves such that its corresponding pants decomposition contains $\lfloor \frac{n}{i} \rfloor$ curves, each of which bounds i punctures.

Assuming the claim for now, for each $i \leq \lfloor \frac{n}{2} \rfloor$, the existence of such a tree implies that to realize the pants decomposition corresponding to this tree, $\lfloor \frac{n}{i} \rfloor$ closed curves that

bound i punctures are needed. This implies that at least $\sum_{i=2}^{\lfloor \frac{n}{2} \rfloor} \lfloor \frac{n}{i} \rfloor$ closed curves are needed to realize T_is for $2 \leq i \leq \lfloor \frac{n}{2} \rfloor$ and proves the lemma.

Proof of the claim. In a rooted tree, we say that a vertex is internal if it is not the root and not a leaf. Let B_i be any trivalent tree with $\lceil \frac{n}{i} \rceil$ leaves and let L_i be a rooted tree with i leaves in which the root has degree 2 and the internal vertices have degree 3. Let R_i be a rooted tree with $n - i\lfloor \frac{n}{i} \rfloor$ leaves in which the root has degree 2 and the internal vertices all have degree 3. We define a tree T_i as follows: in the case where i divides n, we paste a copy of L_i on each leaf of B_i by identifying the root in L_i with a leaf in B_i. If i does not divide n, we paste a copy of L_i on every leaf of B_i except one, and we paste R_i on the last leaf. We can see that T_i is a trivalent tree with n leaves. Each edge in T_i that corresponds to an edge that used to connect a leaf in B_i separates i leaves in T_i and therefore its corresponding curve in the pants decomposition bounds i punctures. We refer to Figure 8.3 for an illustration. □

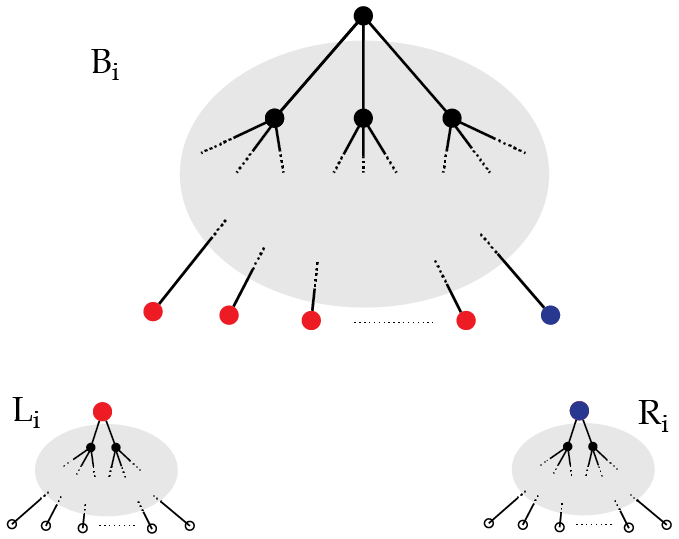

Figure 8.3: The case where i does not divide n. B_i has $\lceil \frac{n}{i} \rceil$ leaves, L_i has i leaves and R_i has $n - i \cdot \lfloor \frac{n}{i} \rfloor$ leaves.

The construction at the beginning of Section 8.3.2 and Lemma 8.3.5 proves the second item of the Theorem L.

8.4 Labelled punctures

In this section, we turn our attention towards families of curves realizing all types of pants decompositions for spheres with labelled punctures. The dual graph to a pants decompo-

sition P of S_n with labelled punctures is a trivalent tree T with n labelled leaves. We call an edge in a tree T an internal edge if it is not adjacent to a leaf. The following lemma provides a family of curves of exponential size realizing all types of pants decompositions in this setting.

Lemma 8.4.1. *Let S_n be the sphere with n labelled punctures. There exists a family of simple closed curves of size less than 3^{n-1} that realizes all types of pants decompositions of S_n up to labelled homeomorphism.*

Proof. Let P be a pants decomposition of S_n and T be the tree dual to P. Each internal edge of the tree separates the leaves into two parts and it corresponds to a simple closed curve on S_n that separates the corresponding punctures. Here, we deal with a bigger set of trees than those that are dual to pants decompositions. Denote by $\mathcal{T}_n$ the set of all trees with n leaves and with every internal vertex having degree at most 3. In this proof, we deal with free homotopy classes of curves. Two homotopy classes of curves are said to be disjoint if there exists a representative in each class such that these curves do not intersect each other.

We remove a point from the sphere and consider the punctures on the plane, which we number arbitrarily from 1 to n and consider lined up from 1 to n. For a subset $S \subset [n]$, and for each map $f : [n] \setminus S \to \{above, below\}$, we define a homotopy class as follows: the curve γ_S^f encompasses the punctures in S, and for each puncture b not in S but within $[\min S, \max S]$, it goes above, respectively below, b, depending on the value of $f(b)$. These homotopy classes are pictured in Figure 8.4, for $S = \{2, 6\}$ and the four possible maps f choosing above or below for the punctures labelled 3, 4 and 5. Note that for punctures not in S and not between the smallest and the largest element of S, there is no choice of "above" or "below" to be made, hence no need to define f. Then we denote by Γ_S the union of all the curves γ_S^f for all the possible maps f.

Let $T \in \mathcal{T}_n$ and fix a vertex v to be the root of T. We say that a set Φ_T of homotopy classes of curves recognizes the tree T with respect to v if it satisfies the following properties. The homotopy classes in Φ_T are pairwise disjoint. If e is an internal edge in T that separates a sub-tree that does not contain the vertex v and that has S as leaves, there exists a homotopy class of curves in Φ_T that encompasses the punctures in S (intuitively, a virtual leaf attached to the root should correspond to the point at infinity). We say that a set of homotopy classes of curves recognizes a tree T if for any vertex v in T, it recognizes T with respect to v. Then, in order to realize the pants decomposition corresponding to T, it is enough to choose a curve from each homotopy class of curves in Φ_T such that these curves are pairwise disjoint.

Let $\Lambda_{[n]} = \bigcup_{S \subset [n], 1 < |S| \leq n-2} \Gamma_S$. We first prove that each tree $T \in \mathcal{T}_n$ can be recognized by a set of homotopy classes in $\Lambda_{[n]}$ and then we show that families of curves can be chosen

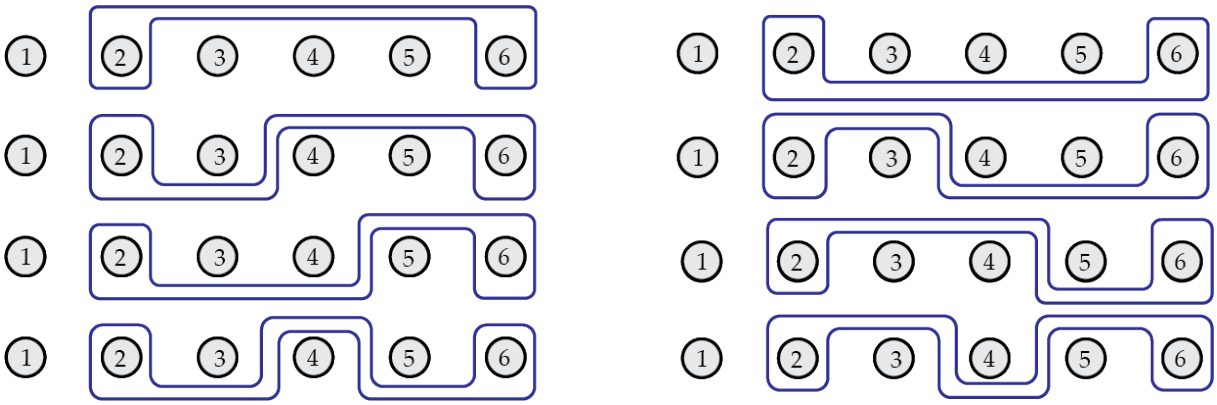

Figure 8.4: Eight possible choices for a curve that separates $\{2, 6\}$ from $\{1, 3, 4, 5\}$

from each homotopy class in $\Lambda_{[n]}$ such that they realize all types of pants decompositions of S_n.

We prove the following claim.

Claim. The set of homotopy classes $\Lambda_{[n]}$ is enough to recognize all trees in $\mathcal{T}_n$.

Proof of the claim: The proof of the claim is by induction on n. Pick a tree $T \in \mathcal{T}_n$ with a root v. Let S_1, S_2 and S_3 be the set of leaves that belong to each branch of v. We need to recognize the edges adjacent to v that are internal edges in the tree. If all the adjacent edges of v are internal edges, then they correspond to three curves that separate the punctures in S_1, S_2 and S_3 from each other. For S_1 (resp. for S_2) choose the homotopy class of curves γ_1 (resp. γ_2) in Γ_{S_1} (resp. Γ_{S_2}) that go above (resp. below) the punctures in $[n] \setminus S_1$ (resp. $[n] \setminus S_2$). For S_3, choose the homotopy class of curves γ_3 that go below the punctures in S_1 and above the punctures in S_2, see Figure 8.5. The choice of above and below guarantees that these three homotopy classes are pairwise disjoint.

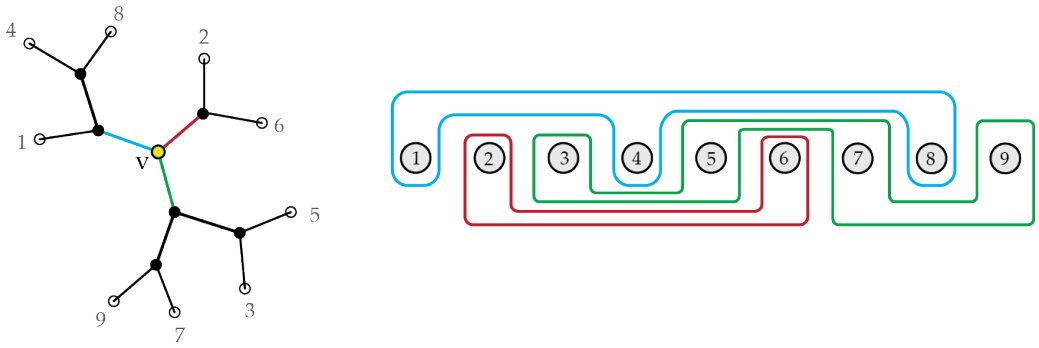

Figure 8.5: The curves realizing the edges adjacent to the root. Here $S_1 = \{1, 4, 8\}$, $S_2 = \{2, 6\}$ and $S_3 = \{3, 5, 7, 9\}$.

We denote by $T \setminus \{v\}$ the tree T where we remove v and the three adjacent edges. Let T_i denote the sub-tree in $T \setminus \{v\}$ that corresponds to S_i. Let v_i be the vertex in T_i that used to be connected to v and let it be the root of T_i; v_i has degree 2. The number of

leaves in T_i for $1 \leq i \leq 3$ is less than n and therefore, by the induction hypothesis, we can recognize the edges in T_i by a set Φ_{T_i} of pairwise disjoint homotopy classes of curves in $\Lambda_{[S_i]}$ with respect to v_i. We can consider these homotopy classes as belonging to $\Lambda_{[n]}$ by making the homotopy classes in Φ_{T_i} go above or below the punctures in $S_i \setminus S$ in the same way that γ_i does. By the way we chose the homotopy classes, they are pairwise disjoint and therefore they recognize the tree T. This finishes the proof of the claim. ■

Now, to realize the pants decomposition corresponding to a tree $T \in \mathcal{T}_n$, it suffices to choose a curve from each homotopy class in Φ_T, such that these curves are pairwise disjoint. Note that by construction, these homotopy classes are pairwise disjoint. Thus, in order to realize all the types of pants decompositions, it suffices to pick a closed curve for each homotopy class in $\Lambda_{[n]}$ so that the resulting family is in minimal position (one way to do that is for example to fix an arbitrary hyperbolic metric on S_n and pick geodesic representatives for each representative).

We conclude the proof by upper bounding the size of the set $\Lambda_{[n]}$. Recall that for a set $S \subset [n]$, if $i \notin [\min(S), \max(S)]$ then the curves in Γ_S do not need to go above or below the puncture i. Thus we have the following bound.

$$\begin{aligned}
|\Lambda_{[n]}| = \sum_{S \subset [n], 1<|S|<n-1} |\Gamma_S| &= \sum_{i=2}^{n-2} \sum_{k=0}^{n-i} (n-k-i+1) \binom{k+i-2}{i-2} 2^k \\
&\leq \sum_{k=0}^{n-2} \sum_{i=k}^{n-2} (i-k+1) \binom{n+k-i-2}{k} 2^k \\
&= \sum_{k=0}^{n-2} 2^k \left(\sum_{j=1}^{n-k-1} j \binom{n-j-1}{k} \right) \\
&= \sum_{k=0}^{n-2} 2^k \binom{n}{k+2} \\
&= \frac{1}{4}(3^n - 2n - 1)
\end{aligned}$$

Note that in the second equation, for a subset S of size i, k counts the number of punctures not in S that belong to $[\min(S), \max(S)]$ and 2^k counts the choices of going below or above these k punctures. The term $(n-k-i+1)$ counts the number of cases for $[\min(S), \max(S)]$ in this case, i.e., the number of cases where $\max(S) - \min(S) = i + k - 1$. □

The following lemma provides an easy exponential lower bound for this problem.

Lemma 8.4.2. *Any family of curves realizing all types of pants decompositions of S_n with labelled punctures has size at least $2^{n-1} - n - 1$.*

Proof. We know that for any subset S of the punctures such that $2 \leq |S| \leq n-2$, there exists a closed curve in some pants decomposition that separates the punctures in S from those in $[n] \setminus S$. Therefore we need at least $|\{S, [n] \setminus S, 2 \leq |S| \leq n-2\}|$ closed curves to realize all types of pants decompositions of S_n with labelled punctures, which is equal to $2^{n-1} - n - 1$. □

Bridging the gap between the 3^{n-1} upper bound and the 2^{n-1} lower bound seems to be an interesting open problem as well.

Lemmas 8.4.1 and 8.4.2 prove the first item of the Theorem L.

Remark 8.4.3. *All the bounds in this section apply equally well to the problem of realizing all types of* pants *(instead of types of pants decompositions), and both the upper and lower bounds are also the best ones we know for that problem.*

8.5 Surfaces without punctures

An easy reduction to the case with labelled punctures, in conjunction with Lemma 8.4.1, directly yields the following bound.

Lemma 8.5.1. *On the surface M_g, there exists a family of curves of size at most 3^{2g-1} realizing all types of pants decompositions.*

Proof. The family consists of taking g simple disjoint closed curves such that cutting along them yields a sphere with $2g$ punctures. Then we use the family of curves given by Lemma 8.4.1. This works because any pants decomposition contains g simple disjoint closed curves such that cutting along them yields a sphere with $2g$ punctures. So we can realize these curves using our cutting curves. After cutting along these g curves, the resulting pants decomposition is one on the sphere with $2g$ punctures, which can be labelled arbitrarily, and thus can be realized using a subset of the curves given by Lemma 8.4.1. The labeling will in particular preserve the matching stipulating how to glue back the punctures, and thus our curves will indeed realize the targeted pants decomposition.

Since the curves in Lemma 8.4.1 are of size at most $3^{2g-1} - g$ (we refer to the proof of that lemma which provides a bound that is slightly sharper than in the statement of the lemma), the resulting family has size at most 3^{2g-1}. □

The technique in this proof can readily be adapted to provide a singly-exponentially sized family of curves realizing all types of pants decompositions on surfaces of genus g with n punctures.

The reduction in the proof of Lemma 8.5.1 is clearly wasteful, in that we use a family of curves tailored to realize the pants decompositions for labelled punctures, but actually it

would suffice to consider only pants decompositions up to homeomorphism which preserves the matching induced by the cutting curves. Due to this inefficiency, the exponential lower bound of Lemma 8.4.2 does not translate to this setting, and we can only prove the following lower bound obtained with a counting argument.

Lemma 8.5.2. *For any $\varepsilon > 0$, any family of simple closed curves realizing all types of pants decompositions on M_g has size $\Omega(g^{4/3-\varepsilon})$.*

Proof. The homeomorphism class of a pants decomposition is determined by the trivalent graph encoding the adjacencies of the pants. A pants decomposition has $2g-2$ curves, and there are, up to terms that are only exponential in g, g^g such trivalent graphs on $2g-2$ vertices, and thus that many pants decompositions (see Lemma 3.6.1). Therefore, since any pants decomposition consists of $3g-3$ curves, any family of simple closed curves Γ realizing them all must satisfy the counting lower bound given by $\binom{|\Gamma|}{3g-3} \gtrsim g^g$, where we use the $\gtrsim$ notation to hide terms that are only exponential in g. Therefore,

$$\left(\frac{|\Gamma|e}{3g-3}\right)^{3g-3} \geq \binom{|\Gamma|}{3g-3} \gtrsim g^g$$

and thus $|\Gamma| = \Omega(g^{4/3-\varepsilon})$ for any $\varepsilon > 0$. □

Lemmas 8.5.1 and 8.5.2 prove Theorem K.

8.6 Small genus cases and labelled punctures: a small improvement

In this last section, we showcase that the approach of cutting along g curves to planarize and then using the bound of labelled sphere is wasteful, as one can do better in the small genus cases when $g = 1$ and $g = 2$. While the proofs are somewhat ad hoc and will not generalize, we believe that this provides an interesting hint that the bound given in Lemma 8.5.1 is not optimal.

First, let us consider the question of realizing all types of pants decompositions on $M_{1,n-2}$ when the punctures are labelled. Note that by cutting along a non-separating curve in $M_{1,n-2}$, we obtain a sphere with n punctures and therefore by Lemma 8.4.1, a family of size at most 3^{n-1} is enough to realize all types of pants decompositions in this case. We improve this bound in the following lemma.

Lemma 8.6.1. *There exists a family of simple closed curves of size less than 3^{n-2} that realizes all types of pants decompositions of $M_{1,n-2}$ up to labelled homeomorphisms.*

Proof. The dual graph to a pants decomposition of $M_{1,n-2}$, is a graph that has only one cycle and in which all vertices have degree three, except $n-2$ vertices of degree one, which in our case, correspond to the labelled punctures and are labelled. Recall that we call an edge that is not adjacent to a labelled vertex an internal edge. Denote the set of all such graphs by $\mathcal{G}$. Note that the edges that belong to the cycle in G are non-separating and correspond to curves that are non-separating, but any two such edges together separate G into two sub-trees and therefore correspond to curves that are separating together. Recall that two free homotopy classes are disjoint if there exists a curve in each of them such that these curves do not intersect.

Let θ be a simple closed curve that is non-separating on $M_{1,n-2}$, as in Figure 8.6. Denote by M' the surface obtained by cutting along θ: this is a sphere with n punctures, among which $n-2$ are our labelled punctures. As in the proof of Lemma 8.4.1, we think of these n punctures as being lined up, and we consider, for each S a subset of the labelled punctures and for each map $f : [n-2]\setminus S \to \{above, below\}$, a curve γ_S^f which encompasses the punctures in S and goes above or below the other punctures as specified by S. Gluing back the surface M' along θ, these curves together define a family of free homotopy classes Γ_S on $M_{1,n-2}$.

Note that by construction, θ is disjoint from γ_S^f for any $S \subset [n-2]$ and choice of f. Now, for each curve γ_S^f where f always maps to *above*, we want to define an alternate curve that additionally goes around the handle following θ. Formally: using a path going above all of the punctures, we base all these curves at a point b located somewhere on θ and consider the curves obtained by concatenating the curves γ_S^f and θ. This yields a second family of free homotopy classes which we denote by Θ_S. We refer to Figure 8.7 for the depiction of a curve γ_S^f in red and its corresponding curve in Θ_S in blue.

We say that a set Φ_G of free homotopy classes of curves recognizes a graph $G \in \mathcal{G}$ if it satisfies the following properties.

- The homotopy classes in Φ_G are pairwise disjoint.
- For each internal edge e in G that is separating a tree that contains the subset S of the labelled vertices, there exists a homotopy class of curves in Φ_G that cuts the surface into a sphere containing the punctures in S and a genus one surface containing the punctures in $[n-2] \setminus S$.
- Let e and e' be a pair of edges that separate a tree from G containing the labelled vertices in S. Then there exist two homotopy classes of non-separating curves in Φ_G corresponding to e and e' that together separate the surface into a sphere that contains the punctures in S and a sphere containing the punctures in $[n-2] \setminus S$.

In order to realize the pants decomposition corresponding to G, it is enough to choose

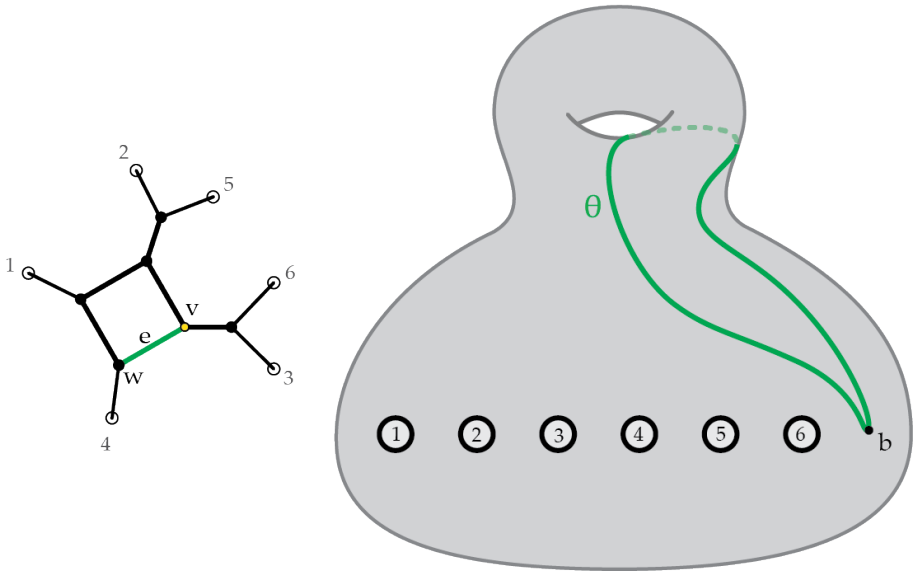

Figure 8.6: We realize the edge e by the green curve θ depicted at right.

a curve from each homotopy class of curves in Φ_G such that these curves are pairwise disjoint.

Let $\Lambda = \bigcup_{S\subset[n-2]}(\Gamma_S \cup \Theta_S)$. As in the proof of Lemma 8.4.1, we first prove that the set of homotopy classes in Λ are enough to recognize every graph in $\mathcal{G}$.

Claim. The set of homotopy classes Λ recognizes all graphs in $\mathcal{G}$.

For every graph $G \in \mathcal{G}$, we proceed as follows. We fix a vertex v in the cycle and let $e = vw$ be one of its adjacent edges that belongs to the cycle. The edge e is non-separating and we realize this edge for every graph $G \in \mathcal{G}$ by the curve $\Theta_\emptyset = \theta$, see Figure 8.6.

Let T be the tree that we obtain by removing e from G and let v be the root of T. Note that T is almost trivalent; only the vertices incident to e (v and w) are of degree two. We recognize the edges in T by the homotopy classes in $\bigcup_{S\subset[n-2]} \Gamma_S$ as in the proof of Lemma 8.4.1 and with the following additional considerations:

- In the proof of Lemma 8.4.1, during the induction step where one removes a vertex to obtain three subtrees, an arbitrary choice was made to select that the curves in one subtree were going above the other punctures, the curve in another subtree were going below and the curves in the last subtree were between. Here, this choice is no longer arbitrary, and we enforce the fact that for each edge in the cycle of G except e, the choice is made so that the corresponding curve goes *above* the remaining punctures. This is pictured in Figure 8.8.

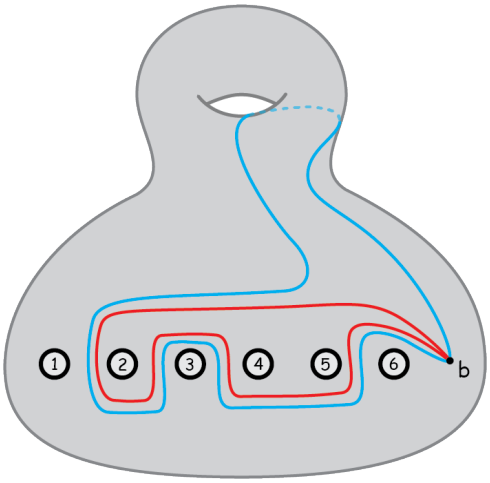

Figure 8.7: The red curve separates the surface to a sphere that has $\{2,4,5\}$ as punctures and a genus one surface with $\{1,6\}$ as punctures; this curve belongs to $\Gamma_{\{2,4,5\}}$. The blue curve is non-separating and is the concatenation of the red curve with θ, this curve belongs to $\Theta_{\{2,4,5\}}$.

- If w is adjacent to a labeled vertex i, the edge iw is not internal and does not need to be recognized. We recognize the other edge adjacent to w by a homotopy class in $\Gamma_{\{i\}}$ (note that in the proof of Lemma 8.4.1 we did not need the homotopy classes of curves that separated a single puncture). Otherwise both edges adjacent to w are recognized with the same homotopy class as the case where these two edges are replaced by one edge.

Now, denote by ϕ_r the homotopy class of curves associated to the edge r in T. We recognize the edges in G as follows. If r is an edge in G that does not belong to the cycle in G, we recognize it by ϕ_r; otherwise, we recognize it by the corresponding homotopy class in Θ_S that is the concatenation of ϕ_r and θ. Denote these curves together with $\Theta_\emptyset = \theta$ by Φ_G. We still need to prove that these homotopy classes are disjoint and that they indeed recognize the edges in G.

By construction, the curves ϕ_r are pairwise disjoint, and so are their counterparts in Θ_S. The remaining disjointnesses follows from the first additional consideration above: the curves in ϕ_r when r belongs to the cycle all go above the punctures that they did not encompass, and thus they can be safely concatenated with θ while still being disjoint from all the other curves in Φ_G.

Finally, recognizing the edges not in the cycle follows the same argument as in the case without genus. For pairs of edges in the cycle, the union of the corresponding pair of curves forms a homology cycle that separates the corresponding punctures as desired. This finishes the proof of the claim. ■

To conclude, it now suffices to pick a curve in each homotopy class of Λ, such that they are altogether in minimal position. For each G, the curves originating from Φ_G

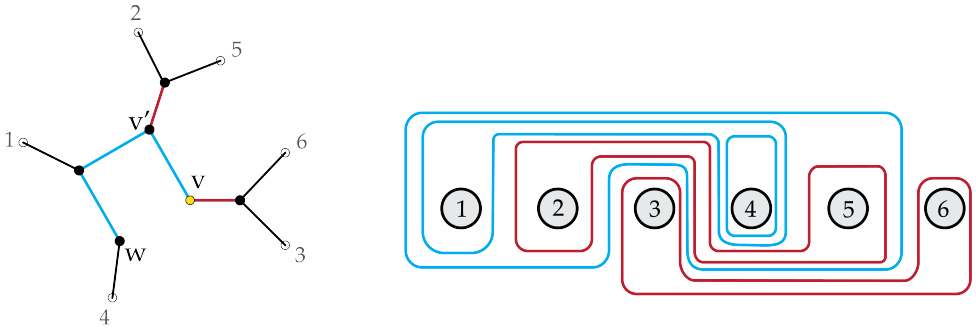

Figure 8.8: The edges in the cycle of the graph are recognized by curves that go above the remaining punctures in (the planarization of) the surface.

are pairwise disjoint and therefore realize the pants decomposition corresponding to G. Therefore, the total family of curves realizes all types of pants decompositions.

Finally, we provide an upper bound the size of the family Λ. The family Θ_S consists of only one curve for each choice of S, and thus, reusing the bound from Lemma 8.4.1, we obtain

$$|\Lambda| = \sum_{S \subset [n]} (|\Gamma_S| + |\Theta_S|) \leq 2^{n-2} + \frac{1}{4}(3^{n-2} - 2n + 3) < 3^{n-2}.$$

□

We next look at the case of genus 2 with $n-4$ punctures, where we can do even better than in the previous case.

Lemma 8.6.2. *There exists a family of simple closed curves of size at most 3^{n-3} that realizes all types of pants decompositions of $M_{2,n-4}$ up to labelled homeomorphisms.*

Proof. The dual graph to a pants decomposition of $M_{2,n-4}$ is a graph with cyclomatic number two (i.e., there are two edges so that removing them yields a tree) in which all vertices have degree three except $n-4$ vertices of degree one; these vertices correspond to the labelled punctures and are labelled. Let us denote all such graphs with $\mathcal{G}$. These graphs have either two disjoint cycles or two cycles that share at least one or more edges, see Figure 8.10. Note that in this case the edges that belong to the cycles are non-separating and therefore correspond to non-separating closed curves on the surface. Also, if the cycles are disjoint, the edges that belong to the path connecting the cycles are separating the graph into two sub-graphs with one cycle each and therefore correspond to separating closed curves that cut the surface into two genus one surfaces.

Fix a basepoint b which we intuitively consider next to the puncture $n-4$, as depicted in Figure 8.11. We consider a family of simple closed curves $\theta_1, \theta_2, \theta_3, \psi, \omega$ based at b as pictured in Figure 8.11: the curves θ_i are non-separating and pairwise disjoint, the curve

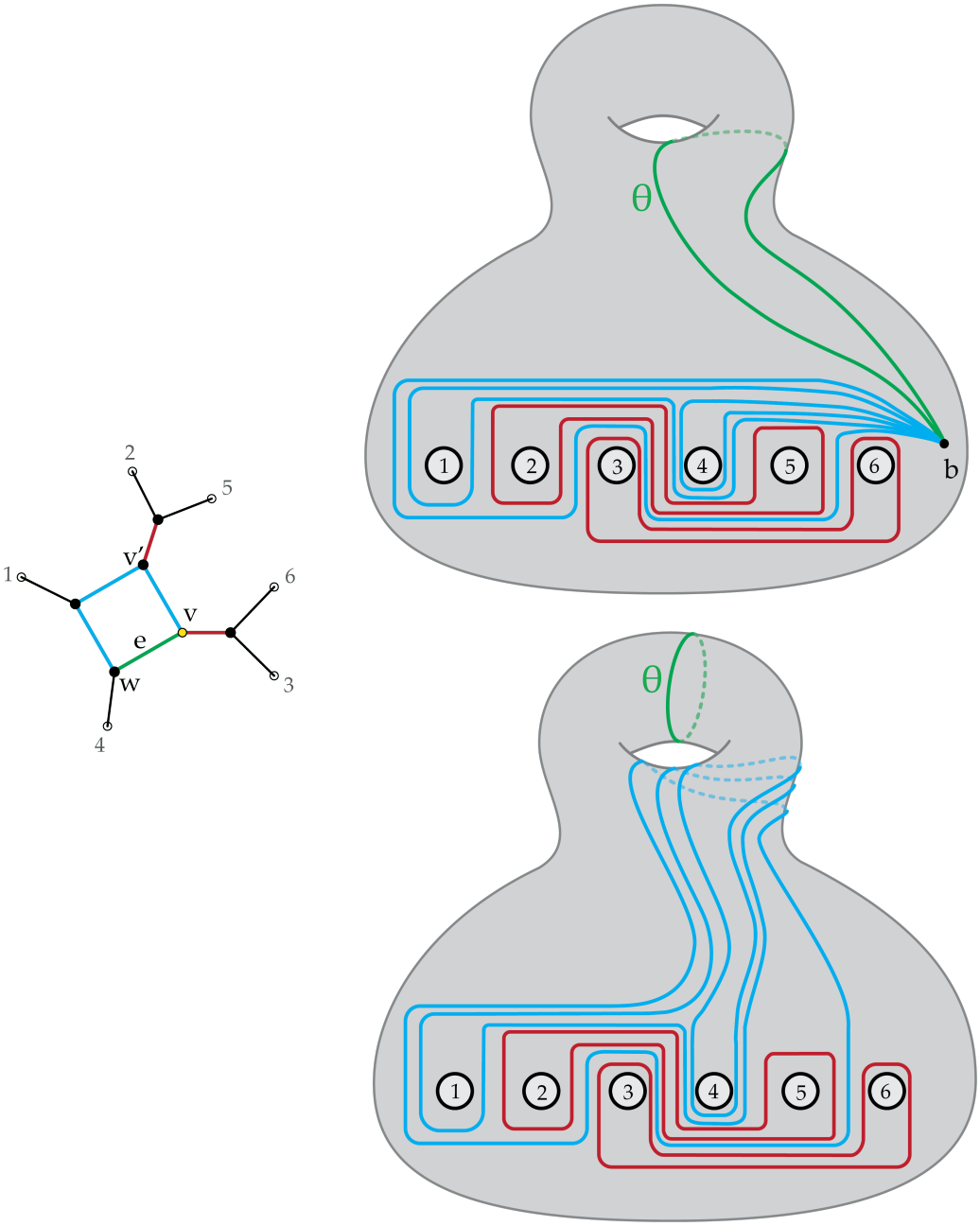

Figure 8.9: Realizing G (at left) from a realization for T. The blue curves correspond to the edges that belong to the cycle in G and the red ones correspond to those that do not. The homotopy classes of the curves in the right bottom drawing recognizes the graph G.

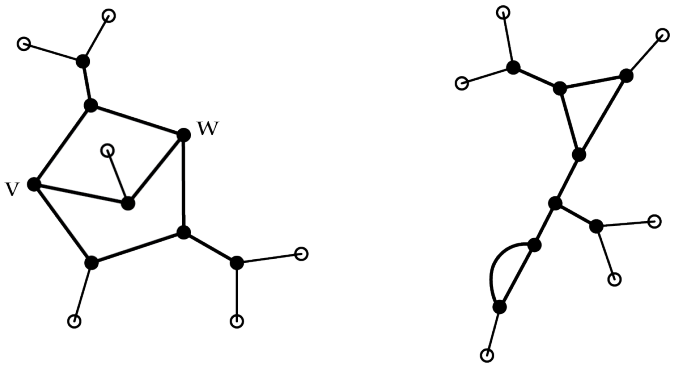

Figure 8.10: Left: the cycles are not disjoint; we can see that there are two vertices with all adjacent edges non-separating. Right: two disjoint cycles.

ψ is also non-separating and only crosses θ_1 once, and the curve ω is separating the handle formed by θ_1 and ψ from the rest of the surface.

Denote by M' the surface we obtain by cutting along a curve in θ_1 and a curve in θ_2. The surface M' is a sphere with n punctures among which $n-4$ are labelled. For a subset S of the labelled punctures, let Γ'_S be the homotopy classes of separating curves that encompass the punctures in S and remain either above or below the rest of the punctures as defined in the proof of Lemma 8.4.1. Let Γ_S be the homotopy classes of closed curves on $M_{2,n-4}$ that we obtain from Γ'_S by gluing back the surface along θ_1 and θ_2. Note that by construction, θ_i for $1 \leq i \leq 3$ and ω are disjoint from Γ_S for $S \subset [n-4]$.

Define Ω_S to be the concatenation of curves in Γ_S with ω; these curves are separating. Finally, define Θ^i_S to be the concatenation of curves in Γ_S with the θ_i for $i = 1, 2, 3$; these curves are not separating.

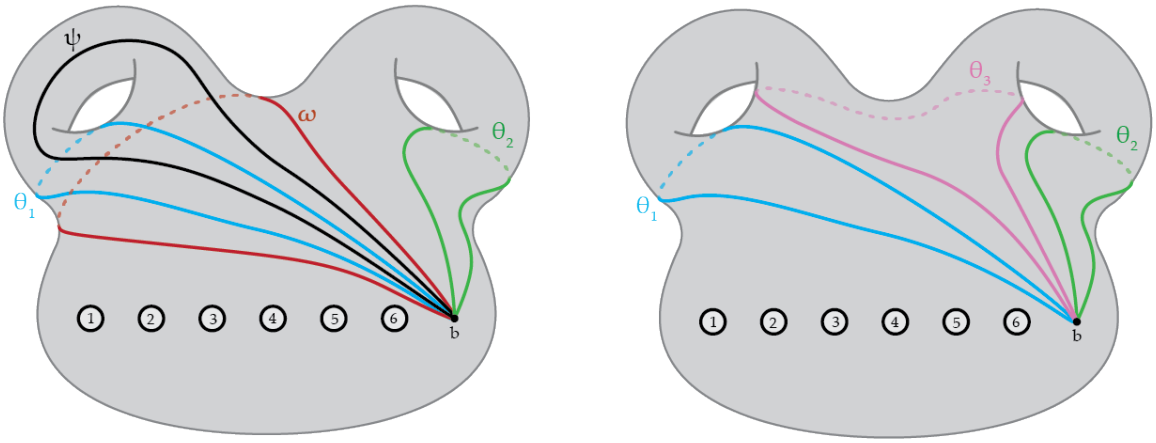

Figure 8.11: Homotopy classes of the non-separating curves $\theta_1, \theta_2, \theta_3$ and ψ and the separating curve ω.

In the proof of Lemma 8.6.1, we introduced a notion for a set of curves recognizing

a graph. Here we provide a more intricate variant of this notion in the genus 2 setting. We say that a set Φ_G of based homotopy classes of curves recognizes a graph $G \in \mathcal{G}$, if it satisfies the following properties.

- The free homotopy classes in Φ_G are pairwise disjoint.
- For each internal edge e in G that is separating a tree that contains the subset S of the labelled vertices, there exists a homotopy class of curves in Φ_G that cuts the surface into a sphere containing the punctures in S and a genus two surface containing the punctures in $[n-4] \setminus S$.
- Let e and e' be a pair of edges that separate a tree from G containing the labelled vertices in S. Then there exist two homotopy classes of non-separating curves in Φ_G corresponding to e and e' that together separate the surface into a sphere that contains the punctures in S and a genus one surface containing the punctures in $[n-4] \setminus S$.
- If an edge separates the graph into two sub-graphs with one cycle such that one contains the vertices in S and the other contains the vertices in $[n-4] \setminus S$, then there exists a homotopy class in Φ_G that separates the surface into two surfaces of genus one such that one contains the punctures in S and the other contains the punctures in $[n-4] \setminus S$.

In order to realize the pants decomposition corresponding to G, it is enough to choose a curve from each homotopy class of curves in Φ_G such that these curves are pairwise disjoint.

Let $\Lambda = \bigcup_{S \subset [n-4]} (\Gamma_S \cup \Theta^1_S \cup \Theta^2_S \cup \Theta^3_S \cup \Omega_S)$. We first prove that the homotopy classes in Λ are enough to recognize all graphs in $\mathcal{G}$. Then we show that we can choose a curve from each homotopy class in Λ such that these curves realize all types of pants decompositions of $M_{2,n-4}$.

Claim. The set of homotopy classes Λ is enough to recognize all graphs in $\mathcal{G}$.

We consider two different cases for the proof of this claim.

The case where the cycles in G are not disjoint. In this case, there exist exactly two vertices v and w such that all their adjacent edges are non-separating and belong to the cycles, see left picture in Figure 8.10. By removing one of these vertices from G, let us say w, we obtain a tree; denote this tree by T. Consider the vertex v to be the root of T and proceed as in Lemma 8.4.1 to recognize the edges of T with homotopy classes in $\bigcup_{S \subset [n-4]} \Gamma_S$. Number the branches of T at v by 1,2 and 3 (there might only exist 2 branches or even one). As in Lemma 8.6.1, we have the power to choose the homotopy classes corresponding to the edges that belong to the cycles and are in branch 1 to go above everything else. Likewise, we ensure that the cycles corresponding to branch 2 go

below everything else, and finally that the edges corresponding to branch 3 go inbetween the cycles of branches 1 and 2. Let Φ_T be the set of homotopy classes of curves that recognize T. We denote the curves in Φ_T that recognize the edge r in T by ϕ_r.

In order to recognize G, we proceed as follows. If r is an edge in G that does not belong to any cycle in G, we recognize it by ϕ_r; otherwise we recognize it by a homotopy class of curves that is the concatenation of ϕ_r with θ_i if it belongs to the branch i in T for $1 \leq i \leq 3$. We recognize the edges adjacent to w by the curves $\Theta^1_\emptyset = \theta_1, \Theta^2_\emptyset = \theta_2$ and $\Theta^3_\emptyset = \theta_3$. Denote these homotopy classes of curves by Φ_G. Now, these free homotopy classes can be realized by disjoint curves as the choices of above/below in the three branches have been specifically designed so that concatenating those curves with θ_1, θ_2 and θ_3 can still be done while preserving disjointedness: see Figure 8.12.

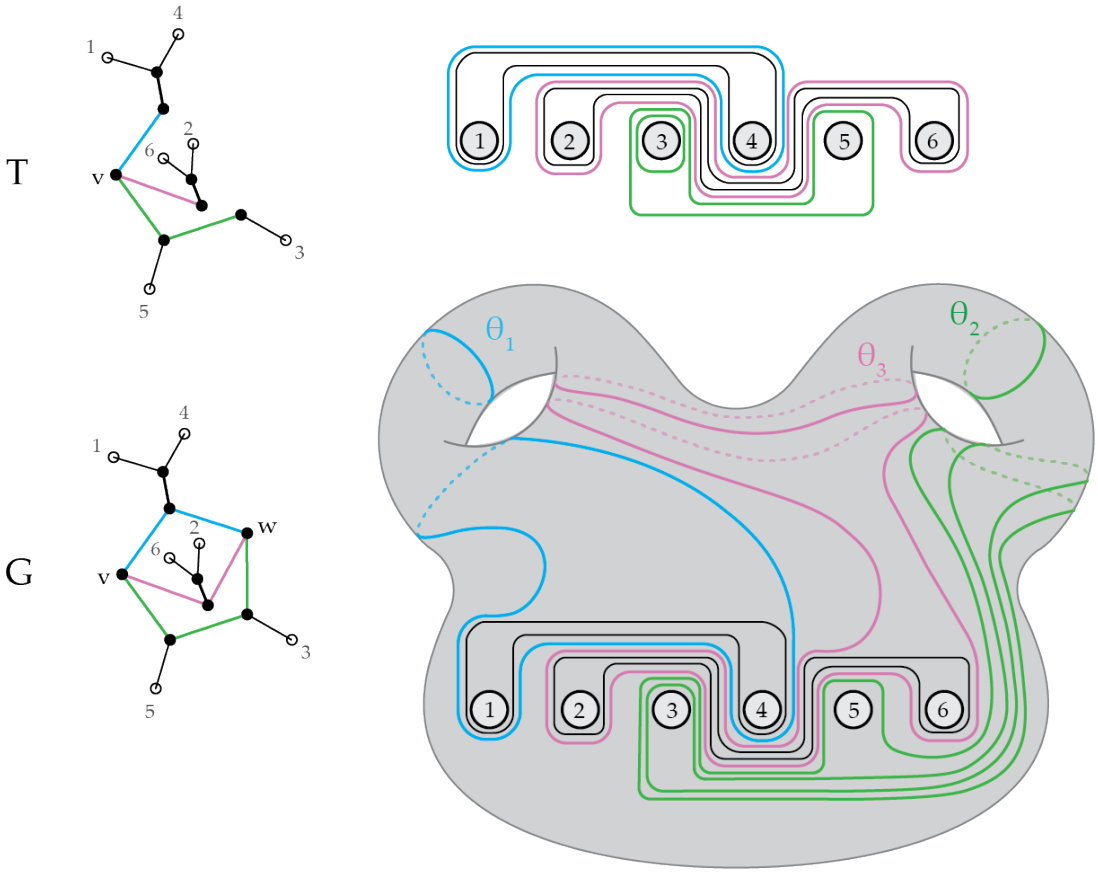

Figure 8.12: The case where the cycles in G are not disjoint. In T the branches with blue, pink and green edges correspond to branches 1,2 and 3, respectively.

The case where G has two disjoint cycles. Choose the vertex v in a cycle of G such that it belongs to the path that connects the two cycles. Remove an edge r_1 from the other cycle and an edge r_2 adjacent to v from the cycle that v belongs to. We recognize r_1 by θ_1 and r_2 by θ_2 in Figure 8.11. The graph $G \setminus \{r_1, r_2\}$ is a tree which we denote by T. Let v be the root of T and proceed as in Lemma 8.4.1 to realize T. We choose the

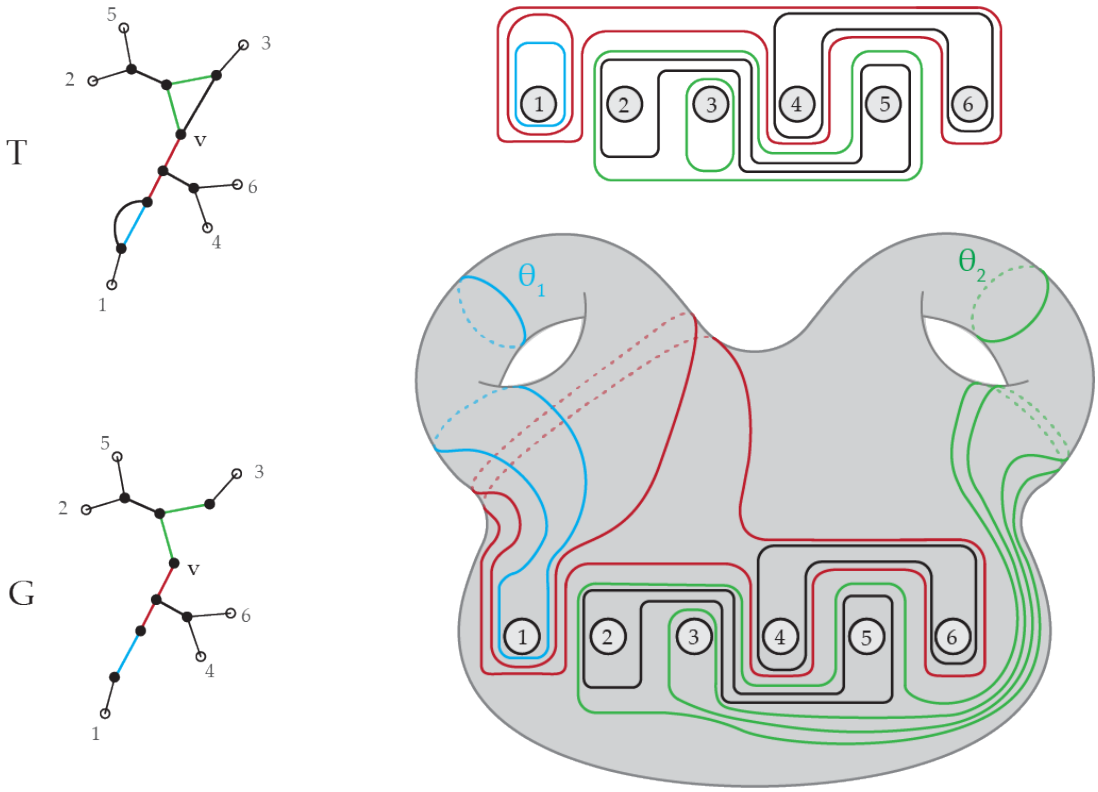

Figure 8.13: The case where the cycles in G are disjoint.

homotopy classes that recognize the edges in the path that connect the two cycles and the edges in the cycle that contains r_1 such that they go above the other punctures. For the edges in the cycle that contains $r-2$, we choose homotopy classes that go below the rest of the punctures. Let Φ_T be the set of homotopy classes of curves that recognize T. We denote the homotopy class in Φ_T that recognizes the edge r in T, by ϕ_r.

In order to recognize G, we proceed as follows. If r is an edge in G that does not belong to any cycle in G nor the path connecting the two cycles, we realize it by ϕ_r. If r belongs to the same cycle as v in G, we concatenate ϕ_r with θ_2 and if it belongs to the other cycle, we concatenate it with θ_1. If r belongs to the path connecting the two cycles, r is recognized by the concatenation of ϕ_r with ω. These curves together with $\Theta^1_\emptyset = \theta_1$ and $\Theta^2_\emptyset = \theta_2$ recognize G; we denote the set containing these curves by Φ_G. As before, the choices of above and below in the homotopy classes have been specifically designed to ensure that the homotopy classes can be realized with disjoint curves, see Figure 8.13.

This finishes the proof of the claim. ■

As in the proofs of the previous lemmas, it now suffices to pick a closed curve in each homotopy class and Λ so that the resulting is in minimal position. By construction, for any type of pants decomposition with dual graph G, the curves in Φ_G are pairwise disjoint and realize this type of pants decomposition. Altogether, the curves that we chose realize all types of pants decompositions.

Finally, we provide an upper bound on the size of Λ. Note that in our construction, it suffices to consider curves in Θ^1_S and Ω_S which go above the punctures they do not encompass, and curves for Θ^2_S which go below the punctures they do not encompass. However, for Θ^3_S, as for Γ_S, we need to consider all possible choices of above and below for the punctures they do not encompass. This yields the following bound.

$$\begin{aligned}
|\Lambda| &= \sum_{S\subset[n]} (|\Theta^1_S| + |\Theta^2_S| + |\Omega_S|) + \sum_{S\subset[n]} (|\Gamma_S| + |\Theta^3_S|) \\
&\leq 3 \times 2^{n-4} + 2 \times \frac{1}{4}(3^{n-4} - 2(n-4) - 1) \\
&< 3^{n-3}
\end{aligned}$$

□

We do not push these studies farther: it would seem that the techniques used in this last proof would require considering curves using arbitrary subsets of the handles, and thus naturally lead to exponentially-sized family of curves. While the resulting base of the exponential could conceivably be smaller than the one we obtain in Lemma 8.5.1, it would therefore not be helpful towards breaking the exponential barrier.

Chapter 9

Conclusion

In this thesis, our focus has been on examining topological problems on graphs and curves embedded on surfaces from a computational perspective. In the following, we will provide a summary of our results and discuss the immediate next steps that lie ahead. Then we introduce some more distant avenues of research that extend beyond the immediate scope of this thesis.

9.1 Summary of results and continuation

1) Decompositions for surfaces and joint crossing numbers. In Chapters 5 and 6, we provided some of the first decompositions of non-orientable surfaces and algorithms to compute them. But our results are restricted to very few types of decompositions. For example, restricting our attention to one-vertex decompositions for non-orientable surfaces, our techniques provide short decompositions of the form $a_1a_1, \ldots, a_la_l, b_1c_1\overline{b}_1\overline{c}_1 \ldots b_kc_k\overline{b}_k\overline{c}_k$ where $l > 0$ and k can be zero; in a sense that we can compute short decompositions in which the non-orientable part is nicely separated from the orientable one. Similarly for orientable surfaces, the only short one-vertex decomposition that is known is the canonical one, $b_1c_1\overline{b}_1\overline{c}_1 \ldots b_gc_g\overline{b}_g\overline{c}_g$. Except this one, no other short decomposition is known, not even a natural decomposition of the form $a_1a_2 \ldots a_g\overline{a}_1\overline{a}_2 \ldots \overline{a}_g$. The best known bound for this decomposition is of length $O(g^2|E(G)|)$ where G is the primal graph embedded on the surface.

More generally, Negami's conjecture which implies that every surface can be decomposed along any shape shortly, remains open.

Conjecture 1. (Negami's conjecture) *There exists a universal constant C such that for any pair of graphs G_1 and G_2 embedded on a surface S, the joint crossing number is at most $C|E(G_1)||E(G_2)|$.*

In this thesis, we described a geometric avenue, first suggested by Hubard, Kaluža, de

Mesmay and Tancer in [50], toward proving Negami's conjecture by asking the following question.

Question 1. *Given a surface S of genus g, does there exist a Riemannian metric on S such that any simple graph embeddable on S can be embedded so that the edges are shortest paths on S?*

We explained that a positive answer to this question implies Negami's conjecture. Although we could not answer the question in its generality, in Chapter 4, we leveled the playing field for both orientable and non-orientable surfaces by providing an $O(g)$-universal shortest path metric for non-orientable surfaces. Furthermore, we introduced an avenue of attack that might lead to a negative answer to this question in Chapter 8.

Although we suspect that Negami's conjecture holds, constructing a universal shortest path metric seems to be more elusive.

2) Degenerate crossing number vs. genus crossing number. In Chapter 7, we studied the conjecture of Mohar regarding degenerate crossing number and genus crossing number of graphs. We disproved the stronger conjecture of Mohar for graphs that are given by a fixed embedding by providing a two-vertex counterexample.

Conjecture 3. ([57, Conjecture 3.4]) *For any positive integer g, every loopless pseudo-triangulation of N_g admits a cross-cap drawing with g cross-caps in which each edge enters each cross-cap at most once.*

Conversely, we managed to provide a systematic way to obtain drawings for most loopless two-vertex embedding schemes that illustrate that *gcr* and *dcr* for these embedding schemes are equal. But our constructions are restricted to loopless two-vertex graphs and still it could be possible that the stronger conjecture of Mohar stands for graphs with more than 2 vertices. We expect that our construction can be extended to bipartite embedding schemes since by contracting edges in a bipartite graph, one can obtain a loopless two-vertex graph.

On the other hand, we know that increasing the number of vertices increases the freedom in achieving a perfect cross-cap drawing. This is because, for any embedding scheme, we can obtain a cross-cap drawing in which each edge enters each cross-cap at most twice using the Schaefer-Štefankovič algorithm. By dragging vertices into cross-caps, we can resolve double entrances for edges. Having more vertices increases the chances of resolving all the double entrances without making another edge enter a cross-cap more than once. This suggests that the two-vertex case is probably the hardest case regarding this conjecture. Since our result completely classifies the cases of loopless two-vertex embedding schemes, we expect for the conjecture to hold in the cases with more than 2 vertices.

Furthermore, the main conjecture of Mohar that is stated for graphs without a fixed embedding remains open.

Conjecture 2. ([57, Conjecture 3.1]) *For every simple graph G, $dcr(G) = gcr(G)$.*

In this conjecture, the embedding scheme is not fixed, which makes it even more flexible than the previous case which in turn makes it more challenging to solve this conjecture. However, by our above discussion, we strongly expect that the conjecture holds.

3) Enumerating curves and arcs on surfaces. In Chapter 8, in an attempt to answer Question 1, we developed an approach based on counting curves in pants decompositions of surfaces. We provided estimates for the cardinality of a universal family of curves that realizes all the pants decompositions of the surface. In particular, we provided a polynomial lower bound and an exponential upper bound for the case of closed surfaces. We explained that proving a super-polynomial bound for the size of a family of curves that realizes all types of pants decompositions in this case, would answer negatively to Question 1. The exponential upper bound provided for this case is directly achieved by using the exponential upper bound provided for the cardinality of a family of curves that realizes all the pants decompositions of a sphere with punctures in which the punctures are labelled. This suggests that this estimation is probably rough, and that it might significantly overshoot the true value.

Devising a general approach to estimate the exact value seems to be hard in this problem. One way to make progress is to estimate the size of such a family for surfaces with low genus. This might shed light on the growth rate of the cardinality or even inspire developing a more systematic way to create such a family. In a joint work with Vincent Delecroix, we have computed this quantity for surfaces up to genus 4. The exact number for surfaces of genus $2, 3$ and 4 is $4, 10$ and 18, respectively (Figure 9.1 depicts an optimal family of curves realizing these numbers). This stirs the speculation that in general the bound is polynomial in g.

9.2 Further research directions

Shortest decompositions and curves

Throughout this thesis, we searched for decompositions that are relatively short, namely of length $O(gn)$ where n is the number of edges in the primal graph of the cross-metric surface and g is its genus. But what if we aim for the shortest decomposition of a specific type? The problem of computing shortest decompositions or curves with particular topological properties has been extensively researched in computational topology. Here, we outline

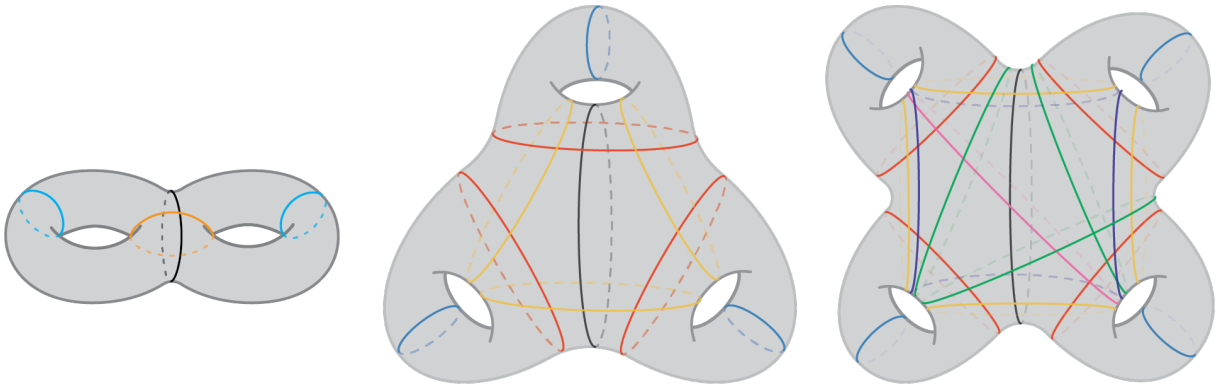

Figure 9.1: Optimal universal families of curves that realize all pants decompositions of surfaces of genus $2, 3$ and 4 respectively.

the best-known results and highlight open problems related to shortest decompositions and curves.

Shortest curves. A well studied problem in the computational topology of surfaces is to compute shortest curves of certain types on a surface. The shortest contractible curve can be computed in polynomial time [13] on a given combinatorial surface. A non-contractible non-separating curve can be computed in $O(n^2 \log n)$ time [33]. However, computing the shortest non-contractible separating curve is NP-hard and there exists an algorithm with running time $g^{O(g)} n \log n$ that computes such a curve [14].

On the other hand, there has been no study on computing shortest curves of certain homeomorphism types on non-orientable surfaces. This is the subject of a work in progress with Éric Colin de Verdière and Denys Bulavka.

Shortest decompositions. In this thesis, we provided polynomial time algorithms to compute canonical decompositions for both orientable and non-orientable surfaces that are relatively short. But in general, the complexity of computing shortest such decompositions is open [34].

The polynomial time algorithm provided in [21], computes an octagonal decomposition of orientable surfaces that is made of closed curves which are as short as possible in their homotopy classes. In Chapter 4, we generalized this decomposition to non-orientable surfaces but in the decomposition that we provided, the curves are not the shortest possible in their homotopy classes. This is due to the fact that the orienting curve that we compute to construct this decomposition is not the shortest. In general, computing the shortest such decompositions of both orientable and non-orientable surfaces is open.

The complexity of computing the shortest pants decomposition of a surface is open even in the case of the Euclidean plane and the Hyperbolic plane with punctures. For these cases, an approximation algorithm that runs in $O(n \log n)$ time is provided in [30] (see also [63]). In [23], a polynomial time algorithm is provided that computes a pants

decomposition that is made of closed curves that are as short as possible in their homotopy classes.

As we mentioned in the introduction, a well-studied problem in computation topology of surfaces is to look for shortest cut graphs. Computing the shortest cut graph is NP-hard but there is an $O(\log^2 g)$-approximation algorithm that computes it in $O(g^2 n \log n)$ time [33]. Can we find a better approximation, such as a constant-factor approximation or even a polynomial-time approximation scheme?

Computational biology playground

Two of the main contributions of this thesis (the results in Chapters 5 and 7), use the insights and an algorithm coming from genome rearrangements in computational biology. This intriguing interplay between computational biology and topology, is motivating to look for other surprising correlations between the two fields.

In Section 3.8, we described two mutations for genomic evolution (reversals and block interchange) and their respective computational formulations to compute the distance between two genomes. We also studied the topology behind the two problems, to establish a foundation to identify the similarities of these problems to our graph embedding setting. We only considered problems in which one type of mutation is possible. However, it is more natural to allow multiple types of mutation to be involved in transforming a set of genes to another. It is especially interesting to study the topology behind the problem when multiple types of mutations are allowed; our (l, k) combination algorithm in Chapter 6 is similar to the case where we allow for both reversals and block interchanges to sort a sequence of genes.

In [43], a polynomial time algorithm is provided that computes the distance between the multi-chromosomal genomes of human and mice by four types of mutations: inversions, translocations, fusions and fissions of chromosomes. This algorithm has led to a scenario of human to mouse evolution with 131 mutations. Here we can ask the following questions. What insights can be gained through the topological interpretation of these operations? Could the topological insight behind the problem simplify the arguments or enhance the computational complexity of the problem? In a broader sense, one might wonder if topological ideas can be leveraged in other problems involving reconstructing the whole phylogenetic tree.

List of publications

[A] Niloufar Fuladi, Alfredo Hubard, Arnaud de Mesmay, *Short topological decompositions of non-orientable surfaces*, published in Discrete & Computational Geometry (DCG), 2023. Preliminary version in the Proceedings of the 38th International Symposium on Computational Geometry (SoCG 2022), Schloss Dagstuhl-Leibniz-Zentrum für Informatik. ArXiv: 2203.06659.

[B] Niloufar Fuladi, Arnaud de Mesmay, Hugo Parlier, *Universal families of arcs and curves on surfaces*, accepted in Israel Journal of Mathematics. ArXiv: 2302.06336.

[C] Niloufar Fuladi, Alfredo Hubard, Arnaud de Mesmay, *Degenerate crossing number and signed reversal distance*. Preliminary version in the Proceedings of the 31st International Symposium on Graph Drawing and Network Visualization, 2023. ArXiv: 2308.10666.

Bibliography

[1] M. J. Alam, M. Fink, and S. Pupyrev, *The bundled crossing number*, in International Symposium on Graph Drawing and Network Visualization, Springer, 2016, pp. 399–412.

[2] N. Alon, *Asymptotically optimal induced universal graphs*, Geometric and Functional Analysis, 27 (2017), pp. 1–32.

[3] D. Archdeacon and C. P. Bonnington, *Two maps on one surface*, Journal of Graph Theory, 36 (2001), pp. 198–216.

[4] L. Auslander, T. A. Brown, and J. W. T. Youngs, *The imbedding of graphs in manifolds*, Journal of Mathematics and Mechanics, (1963), pp. 629–634.

[5] V. Bafna and P. A. Pevzner, *Genome rearrangements and sorting by reversals*, SIAM Journal on computing, 25 (1996), pp. 272–289.

[6] A. Bergeron, *A very elementary presentation of the Hannenhalli-Pevzner theory*, in Annual Symposium on Combinatorial Pattern Matching, Springer, 2001, pp. 106–117.

[7] A. Bergeron, J. Mixtacki, and J. Stoye, *Reversal distance without hurdles and fortresses*, in Annual Symposium on Combinatorial Pattern Matching, Springer, 2004, pp. 388–399.

[8] O. Bernardi and G. Chapuy, *Counting unicellular maps on non-orientable surfaces*, Advances in Applied Mathematics, 47 (2011), pp. 259–275.

[9] B. Bollobás, *The asymptotic number of unlabelled regular graphs*, Journal of the London Mathematical Society, 2 (1982), pp. 201–206.

[10] M. Bonamy, C. Gavoille, and M. Pilipczuk, *Shorter labeling schemes for planar graphs*, SIAM Journal on Discrete Mathematics, 36 (2022), pp. 2082–2099.

[11] A. C. Bura, R. X. Chen, and C. M. Reidys, *On a lower bound for sorting signed permutations by reversals*, arXiv preprint arXiv:1602.00778, (2016).

[12] P. Buser and M. Seppälä, *Triangulations and homology of riemann surfaces*, Proceedings of the American Mathematical Society, 131 (2003), pp. 425–432.

[13] S. Cabello, *Finding shortest contractible and shortest separating cycles in embedded graphs*, ACM Transactions on Algorithms (TALG), 6 (2010), pp. 1–18.

[14] E. W. Chambers, É. Colin de Verdière, J. Erickson, F. Lazarus, and K. Whittlesey, *Splitting (complicated) surfaces is hard*, Computational Geometry: Theory and Applications, 41 (2008), pp. 94–110.

[15] G. Chapuy, *A new combinatorial identity for unicellular maps, via a direct bijective approach*, Advances in Applied Mathematics, 47 (2011), pp. 874–893.

[16] D. A. Christie, *Sorting permutations by block-interchanges*, Information Processing Letters, 60 (1996), pp. 165–169.

[17] V. Cohen-Addad, É. Colin De Verdière, D. Marx, and A. De Mesmay, *Almost tight lower bounds for hard cutting problems in embedded graphs*, Journal of the ACM (JACM), 68 (2021), pp. 1–26.

[18] V. Cohen-Addad and A. de Mesmay, *A fixed parameter tractable approximation scheme for the optimal cut graph of a surface*, in Algorithms-ESA 2015: 23rd Annual European Symposium, Patras, Greece, September 14-16, 2015, Proceedings, Springer, 2015, pp. 386–398.

[19] É. Colin De Verdière, *Topological algorithms for graphs on surfaces*, Habilitation à diriger des recherches, (2012). Available at `http://www.di.ens.fr/~colin/`.

[20] É. Colin De Verdière, *Computational topology of graphs on surfaces*, in Handbook of Discrete and Computational Geometry, J. E. Goodman, J. O'Rourke, and C. Toth, eds., CRC Press LLC, third ed., 2018, ch. 23, pp. 605–636.

[21] É. Colin De Verdière and J. Erickson, *Tightening nonsimple paths and cycles on surfaces*, SIAM Journal on Computing, 39 (2010), pp. 3784–3813.

[22] É. Colin De Verdière, A. Hubard, and A. de Mesmay, *Discrete systolic inequalities and decompositions of triangulated surfaces*, Discrete & Computational Geometry, 53 (2015), pp. 587–620.

[23] É. Colin De Verdière and F. Lazarus, *Optimal pants decompositions and shortest homotopic cycles on an orientable surface*, Journal of the ACM (JACM), 54 (2007), pp. 18–es.

[24] Y. Colin De Verdière, *Comment rendre géodésique une triangulation d'une surface*, L'Enseignement Mathématique, 37 (1991), pp. 201–212.

[25] P. Compeau and P. Pevzner, *Bioinformatics algorithms: an active learning approach*, Active Learning Publishers La Jolla, California, 2015.

[26] M. P. Do Carmo and J. Flaherty Francis, *Riemannian geometry*, vol. 6, Springer, 1992.

[27] T. Dobzhansky and A. H. Sturtevant, *Inversions in the chromosomes of drosophila pseudoobscura*, Genetics, 23 (1938), pp. 28–64.

[28] V. Dujmović, L. Esperet, C. Gavoille, G. Joret, P. Micek, and P. Morin, *Adjacency labelling for planar graphs (and beyond)*, Journal of the ACM (JACM), 68 (2021), pp. 1–33.

[29] C. A. Duncan, M. T. Goodrich, and S. G. Kobourov, *Planar drawings of higher-genus graphs.*, Journal of Graph Algorithms and Applications, 15 (2011), pp. 7–32.

[30] D. Eppstein, *Squarepants in a tree: sum of subtree clustering and hyperbolic pants decomposition*, ACM Transactions on Algorithms (TALG), 5 (2009), pp. 1–24.

[31] D. B. Epstein, *Curves on 2-manifolds and isotopies*, Acta Mathematica, 115 (1966), pp. 83–107.

[32] J. Erickson, *Combinatorial optimization of cycles and bases*, Advances in applied and computational topology, 70 (2012), pp. 195–228.

[33] J. Erickson and S. Har-Peled, *Optimally cutting a surface into a disk*, Discrete & Computational Geometry, 31 (2004), pp. 37–59.

[34] J. Erickson and K. Whittlesey, *Greedy optimal homotopy and homology generators*, in Symposium on Discrete Algorithms, vol. 5, 2005, pp. 1038–1046.

[35] L. Esperet, G. Joret, and P. Morin, *Sparse universal graphs for planarity*, Journal of the London Mathematical Society, (2023).

[36] G. Fertin, A. Labarre, I. Rusu, S. Vialette, and E. Tannier, *Combinatorics of genome rearrangements*, MIT press, 2009.

[37] C. Gavoille and C. Hilaire, *Minor-universal graph for graphs on surfaces*, arXiv preprint arXiv:2305.06673, (2023).

[38] J. Geelen, T. Huynh, and R. B. Richter, *Explicit bounds for graph minors*, Journal of Combinatorial Theory, Series B, 132 (2018), pp. 80–106.

[39] H. Gray, *Anatomy of the human body*, vol. 8, Lea & Febiger, 1878.

[40] J. E. Greene, *On curves intersecting at most once, ii*, arXiv preprint arXiv:1811.01413, (2018).

[41] L. Guth, H. Parlier, and R. Young, *Pants decompositions of random surfaces*, Geometric and Functional Analysis, 21 (2011), p. 1069.

[42] Q. Han and J.-X. Hong, *Isometric embedding of Riemannian manifolds in Euclidean spaces*, vol. 13, American Mathematical Soc., 2006.

[43] S. Hannenhalli and P. A. Pevzner, *Transforming men into mice (polynomial algorithm for genomic distance problem)*, in Proceedings of IEEE 36th annual foundations of computer science, IEEE, 1995, pp. 581–592.

[44] ——, *Transforming cabbage into turnip: polynomial algorithm for sorting signed permutations by reversals*, Journal of the ACM (JACM), 46 (1999), pp. 1–27.

[45] A. Hatcher, *Algebraic Topology*, Cambridge University Press, 2002.

[46] B. Hayes, *Computing science: Sorting out the genome*, American Scientist, 95 (2007), pp. 386–391.

[47] P. Hliněný and G. Salazar, *On hardness of the joint crossing number*, in International Symposium on Algorithms and Computation, Springer, 2015, pp. 603–613.

[48] K. Hormann, B. Lévy, and A. Sheffer, *Mesh parameterization: Theory and practice*, ACM SIGGRAPH ASIA 2008 courses, (2007).

[49] F. W. Huang and C. M. Reidys, *A topological framework for signed permutations*, Discrete Mathematics, 340 (2017), pp. 2161–2182.

[50] A. Hubard, V. Kaluža, A. De Mesmay, and M. Tancer, *Shortest path embeddings of graphs on surfaces*, Discrete & Computational Geometry, 58 (2017), pp. 921–945.

[51] F. István, *On straight-line representation of planar graphs*, Acta scientiarum mathematicarum, 11 (1948), pp. 229–233.

[52] F. Lazarus, *Combinatorial graphs and surfaces from the computational and topological viewpoint followed by some notes on the isometric embedding of the square*

flat torus, Habilitation à diriger des recherches, (2014). Available at `http://www.gipsa-lab.grenoble-inp.fr/~francis.lazarus/Documents/hdr-Lazarus.pdf`.

[53] F. Lazarus, M. Pocchiola, G. Vegter, and A. Verroust, *Computing a canonical polygonal schema of an orientable triangulated surface*, in Proceedings of the seventeenth annual symposium on Computational geometry, 2001, pp. 80–89.

[54] J. Matoušek, E. Sedgwick, M. Tancer, and U. Wagner, *Untangling two systems of noncrossing curves*, Israel Journal of Mathematics, 212 (2016), pp. 37–79.

[55] A. Miller, *Asymptotic bounds for permutations containing many different patterns*, Journal of Combinatorial Theory, Series A, 116 (2009), pp. 92–108.

[56] M. Mitzenmacher and E. Upfal, *Probability and computing: Randomization and probabilistic techniques in algorithms and data analysis*, Cambridge university press, 2017.

[57] B. Mohar, *The genus crossing number*, ARS Mathematica Contemporanea, 2 (2009), pp. 157–162.

[58] B. Mohar and C. Thomassen, *Graphs on surfaces*, vol. 10, JHU press, 2001.

[59] S. Negami, *Crossing numbers of graph embedding pairs on closed surfaces*, Journal of Graph Theory, 36 (2001), pp. 8–23.

[60] S. R. Nicholls, N. Scherich, and J. Shneidman, *Large 1-systems of curves in non-orientable surfaces*, Involve, 16 (2023), pp. 127–139.

[61] J. Pach and G. Tóth, *Degenerate crossing numbers*, Discrete & Computational Geometry, 41 (2009), pp. 376–384.

[62] J. D. Palmer and L. A. Herbon, *Plant mitochondrial dna evolved rapidly in structure, but slowly in sequence*, Journal of Molecular evolution, 28 (1988), pp. 87–97.

[63] S.-H. Poon and S. Thite, *Pants decomposition of the punctured plane*, arXiv preprint cs/0602080, (2006).

[64] P. Przytycki, *Arcs intersecting at most once*, Geometric and Functional Analysis, 25 (2015), pp. 658–670.

[65] R. Rado, *Universal graphs and universal functions*, Acta Arithmetica, 9 (1964), pp. 331–340.

[66] R. B. Richter and G. Salazar, *Two maps with large representativity on one surface*, Journal of Graph Theory, 50 (2005), pp. 234–245.

[67] I. Rivin, *Counting cycles and finite dimensional lp norms*, Advances in applied mathematics, 29 (2002), pp. 647–662.

[68] M. Schaefer, *The graph crossing number and its variants: A survey*, The electronic journal of combinatorics, (2012).

[69] M. Schaefer and D. Štefankovič, *The degenerate crossing number and higher-genus embeddings*, Journal of Graph Algorithms and Applications, 26 (2022), pp. 35–58.

[70] T. Shinbrot and W. Young, *Why decussate? Topological constraints on 3D wiring*, The Anatomical Record: Advances in Integrative Anatomy and Evolutionary Biology, 291 (2008), pp. 1278–1292.

[71] K. Stephenson, *Introduction to circle packing: The theory of discrete analytic functions*, Cambridge University Press, 2005.

[72] J. Stillwell, *Classical topology and combinatorial group theory*, vol. 72, Springer Science & Business Media, 1993.

[73] S. Vulliemoz, O. Raineteau, and D. Jabaudon, *Reaching beyond the midline: why are human brains cross wired?*, The Lancet Neurology, 4 (2005), pp. 87–99.