A Computational Approach for Finding 6-List-Critical Graphs on the Torus

Félix Moreno Peñarrubia
Treball de Final de Grau

Grau en Matemàtiques, FME-UPC
Grau en Enginyeria Informàtica, FIB-UPC

May 2023

Director: Zdeněk Dvořák   *Tutor: Oriol Serra Albó*

Abstract

Coloring graphs embedded on surfaces is an old and well-studied area of graph theory. Thomassen proved that there are finitely many 6-critical graphs on any fixed surface and provided the explicit set of 6-critical graphs on the torus. Later, Postle proved that there are finitely many 6-list-critical graphs on any fixed surface. With the goal of finding the set of 6-list-critical graphs on the torus, we develop and implement algorithmic techniques for computer search of critical graphs in different list-coloring settings.

Keywords: graphs on surfaces, list coloring, graph algorithms.
MSC2020 codes: 05C10, 05C15, 68R10.

Resum

La coloració de grafs dibuixats a superfícies és un àrea antiga i molt estudiada de la teoria de grafs. Thomassen va demostrar que hi ha un nombre finit de grafs 6-crítics a qualsevol superfície fixa i va proporcionar el conjunt explícit dels grafs 6-crítics al torus. Després, Postle va demostrar que hi ha un nombre finit de grafs 6-llista-crítics a qualsevol superfície fixa. Amb l'objectiu de trobar el conjunt de grafs 6-llista-crítics al torus, desenvolupem i implementem tècniques algorítmiques per la cerca per ordinador de grafs crítics en diferents situacions de coloració per llistes.

Paraules clau: grafs en superfícies, coloració amb llistes, algorismes sobre grafs.
Codis MSC2020: 05C10, 05C15, 68R10.

Resumen

La coloración de grafos dibujados en superficies es un área antigua y muy estudiada de la teoría de grafos. Thomassen demostró que hay un número finito de grafos 6-críticos en cualquier superficie fija y proporcionó el conjunto explícit de los grafos 6-críticos en el toro. Después, Postle demostró que hay un número finito de grafos 6-lista-críticos en cualquier superficie fija. Con el objetivo de encontrar el conjunto de grafos 6-lista-críticos en el toro, desarrollamos e implementamos técnicas algorítmicas para la búsqueda por ordenador de grafos críticos en diferentes situaciones de coloración por listas.

Palabras clave: grafos en superficies, coloración con listas, algoritmos sobre grafos.
Códigos MSC2020: 05C10, 05C15, 68R10.

Contents

Chapter 1

Introduction

In this chapter, we lay out the basic definitions of graph theoretical and topological concepts used in this thesis, as well as the background results which contextualize our research.

We follow the exposition of Diestel [6], of Mohar and Thomassen [17] and of Postle [18].

1.1 Graphs and Surfaces

1.1.1 Graph Theory Terminology

Here we lay out the basic definitions of graph theory that we will be using in this thesis. We expect the reader to already have some familiarity with the concepts, and therefore we will not delve into explainations and will just state the definitions for the purpose of completeness and disambiguation.

Definition 1.1.1. A *graph* G is a pair $(V(G), E(G))$ consisting of a finite set $V(G)$ and a set $E(G)$ of two-element subsets of $V(G)$. We call the elements of $V(G)$ vertices and the elements $\{u, v\}$ of $E(G)$ edges, which we often denote as uv.

Definition 1.1.2. Two vertices u, v are *adjacent* if $uv \in E(G)$.

For an edge $e = \{u, v\} \in E(G)$, we say e is *incident* with u and v.

The *neighborhood* of $v \in V(G)$, denoted by $N(v)$, is the set of all vertices in G adjacent to v. The *degree* of a vertex v is $d(v) = |N(v)|$.

Definition 1.1.3. Two graphs G and H are *isomorphic* if there exists a bijection f between $V(G)$ and $V(H)$ any two vertices u and v in G are adjacent if and only if $f(u)$ and $f(v)$ are adjacent in H.

Definition 1.1.4. For a graph $G = (V, E)$, we say a graph $G' = (V', E')$ is a *subgraph of* G if $V' \subseteq V$, $E' \subseteq E$ and for all $uv \in E'$ we have $u, v \in V'$. We denote G' being a subgraph of G by $G' \subseteq G$, and denote a *proper* subgraph (a

subgraph in which $E' \subsetneq E$) by $G' \subset G$. We say that a subgraph $G' = (V', E')$ is *induced* if E' contains all the edges in E whose endpoints are in V'. If $X \subset V$, we denote by $G[X]$ the induced subgraph with the vertex set X. If $G' = (V', E) \subset G = (V, E)$, we denote by $G \setminus G'$ the graph $G[V \setminus V']$, and if $X \subset V$ we denote by $G \setminus X$ the graph $G[V \setminus X]$. If e is an edge, $G \setminus e$ is the graph $(V, E \setminus \{e\})$.

Definition 1.1.5. A *connected component* of a graph is the induced graph by the vertices of an equivalence class over the transitive closure of the adjacency relation on the vertices.

A graph is *connected* if it has only one connected component.

A vertex $v \in V(G)$ is a *cutvertex* if $G \setminus \{v\}$ is not connected. A graph G is *2-connected* if it has no cutvertices.

Definition 1.1.6. The *complete graph in n vertices*, denoted by K_n, is the graph with a vertex set of size n and edges between any pair of vertices. A *path* is a connected graph with two vertices, the *endpoints*, with degree 1 and all the other vertices with degree 2. The path is said to be of *length* ℓ if it has ℓ edges. A *cycle* is a connected graph with all vertices of degree 2. The cycles is said to be of length ℓ if it has ℓ edges and vertices. The cycle of length ℓ is denoted by C_ℓ.

Definition 1.1.7. The *join* of two graphs $G_1 = (V_1, E_1)$ and $G_2 = (V_2, E_2)$, denoted by $G_1 + G_2$, is the graph $(V_1 \cup V_2, E_1 \cup E_2 \cup (V_1 \ltimes V_2))$, where $(V_1 \ltimes V_2)$ denotes the set $\{v_1, v_2 : v_1 \in V_1, v_2 \in V_2\}$.

Definition 1.1.8. The *distance* $d(u, v)$ between two vertices u and v in G is the length of the shortest path between them.

In one section of this thesis, we will also need the variant of *directed graphs*.

Definition 1.1.9. A *directed graph* G is a pair $(V(G), E(G))$ consisting of a finite set $V(G)$ and a set $E(G)$ of ordered pairs of elements of $V(G)$. We call the elements of $V(G)$ vertices and the elements (u, v) of $E(G)$ edges, which we often denote as $\overrightarrow{uv}$.

Definition 1.1.10. The *indegree* $d^+(v)$ of a vertex $v \in V(G)$ of a directed graph G is the number of edges of the form $\overrightarrow{uv}$ in G. The *outdegree* $d^-(v)$ is the number of edges of the form $\overrightarrow{vu}$ in G. A directed graph is said to be *eulerian* if for all vertices v, $d^+(v) = d^-(v)$.

1.1.2 Surfaces

Definition 1.1.11. A *surface* is a connected compact 2-dimensional manifold without boundary.

Example 1.1.12. The *sphere* S_0 is the surface defined by the set $\{(x, y, z) \in \mathbb{R}^3 : x^2 + y^2 + z^2 = 1\}$ with the euclidean metric inherited from $\mathbb{R}^3$.

The *torus* S_1 is the surface defined by the set $\{(x, y, z) \in \mathbb{R}^3 : (\sqrt{x^2 + y^2} - 2)^2 + z^2 = 1\}$ with the euclidean metric inherited from $\mathbb{R}^3$.

Note that, even though we have defined the above example surfaces as geometric objects in $\mathbb{R}^3$, we think of surfaces as *topological* objects, and therefore consider two surfaces equivalent if they are *homeomorphic*.

An important result in topology is that all surfaces defined in this way can be classified:

Theorem 1.1.13 (Classification Theorem of Surfaces)**.** *Every surface is homeomorphic to one of the following surfaces:*

- S_0, *the sphere.*

- S_k, *the surface obtained from the sphere by performing the operation of adding a handle $k \geq 1$ times.*

- N_k, *the surface obtained from the sphere by performing the operation of adding a crosscap $k \geq 1$ times.*

The operation of *adding a handle* in a surface Σ can be thought of as deleting the interiors of two small disks T_1 and T_2 from the surface Σ and identifying the boundaries of T_1 and T_2. The operation of *adding a crosscap* in a surface Σ can be thought of as deleting the interior of a small disk T of Σ and identifying the diametrically opposite points of T. In this thesis, we will be mainly concerned only with the two surfaces S_0 and S_1, so we do not delve into the topological details of this construction.

We often represent surfaces via their *fundamental polygon*. A fundamental polygon is a polygon with a labelling and an orientation of its edges so that the corresponding surface is obtained by identifying the edges with the same labels along the specified orientations. Theorem 1.1.13 can be restated as that every surface is homeomorphic to a surface obtained from some type of fundamental polygon. See Figure 1.1 for representations of S_0 and S_1 as fundamental polygons.

Finally, we define the following topological invariant for surfaces:

Definition 1.1.14. The *Euler characteristic* of a surface Σ is $\chi(\Sigma) = 2 - 2k$ if $\Sigma = S_k$ and $\chi(\Sigma) = 2 - k$ if $\Sigma = N_k$. The *Euler genus* of a surface Σ is $g(\Sigma) = 2 - \chi(\Sigma)$.

1.1.3   Embedding Graphs in Surfaces

Let X be a topological space.

Definition 1.1.15. An *arc* in X is the image of a continuous injective function $f : [0, 1] \to X$. A *closed curve* in X is the image of a continuous injective function $f : S^1 \to X$, where S^1 is the circle.

Definition 1.1.16. A *graph embedded in X* is a graph G in which each element of $V(G)$ is a point in X, together with an associated arc A_{uv} in X for each edge $uv \in E(G)$, such that the interior of each arc A_{uv} is disjoint from any vertices of G and its endpoints are u, v.
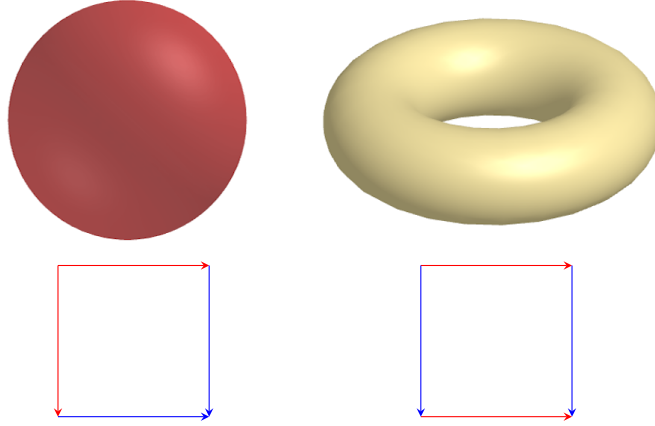
Figure 1.1: Representation of S_0 and S_1 as embedded manifolds in $\mathbb{R}^3$ and as fundamental polygons.

The existence of an arc between two points of X determines an equivalency relation which partitions X into equivalence classes known as *arcwise connected components*.

Definition 1.1.17. A *face* of a graph G embedded in X is an arcwise connected component of $X \setminus \bigcup_{uv \in E(G)} A_{uv}$.

The graph G is *2-cell-embedded* if every face is homeomorphic to an open disk.

Definition 1.1.18. A *plane graph* is a graph G embedded in the plane. A *planar graph* is a graph G for which there exists an embedding of G into the plane.

If G is a plane graph, then there exists an unbounded face of G. We say that the boundary walk of the infinite face of G is the *outer walk* of G. We say than an edge e of G is a chord of the outer walk of G if the edge does not lie on the boundary of the infinite face but both its ends do.

We have the following result for plane graphs:

Theorem 1.1.19. *In a 2-connected plane graph, each face is bounded by a cycle.*

Therefore, when talking about 2-connected graphs, we may refer to the outer walk as the *outer cycle*.

Note also that technically plane graphs are not graphs embedded in surfaces, since the plane is not a surface according to our definition (it is not compact). But by compactifying the plane we see that embedding a connected graph in the plane is in some sense "equivalent" to embedding the graph in S_0, and we will interchangeably refer to plane graphs and graphs embedded in the sphere when convenient.

Theorem 1.1.20 (Euler's formula)**.** *Let G be a 2-cell-embedded graph in a surface Σ. If G has V vertices, E edges and F faces, then*

$$V - E + F = \chi(\Sigma)$$

Definition 1.1.21. A *homotopy* between two functions f and g from a space X to a space Y is a continuous map $G : X \times [0,1] \to Y$ such that $G(x,0) = f(x)$ and $G(x,1) = g(x)$. Two functions are *homotopic* or *homotopically equivalent* if there is an homotopy between them.

Definition 1.1.22. A *contractible cycle* of a graph G embedded in a surface is a cycle in the graph whose embedding is the image of a closed curve homotopic to a constant map. The *edge-width $ew(G)$* of an embedded graph G is the length of the smallest non-contractible cycle in G.

1.2   Graph Coloring

Problems related to *coloring* are a fundamental part of graph theory. Although there are many variants, the original one and the most important is *vertex coloring*.

Definition 1.2.1. A *vertex coloring* of a graph G is a function $\phi : V(G) \to \mathbb{N}$. The vertex coloring is said to be *proper* if $\forall uv \in E(G), \phi(u) \neq \phi(v)$.

We think of this as assigning one color to each vertex of the graph, so that adjacent vertices are assigned different colors. This interpretation comes from the origin of the problem in *map coloring*, in which we have to color a political map assigning colors to countries so that neighboring countries are assigned different colors in order to distinguish them. A quantity of interest is the number of colors required for a proper coloring of the graph:

Definition 1.2.2. A vertex coloring ϕ is said to be a *k-coloring* if $|\mathrm{Im}\phi| = k$. A graph G is said to be *k-colorable* if it admits a proper k-coloring. The *chromatic number $\chi(G)$* of a graph G is the minimum k such that G is k-colorable.

In the map coloring context, we study vertex coloring for *planar* graphs. The following remarkable result was the origin of this area of mathematics:

Theorem 1.2.3 (Four color theorem)**.** *For all planar graphs G, $\chi(G) \leq 4$.*

This theorem, originally conjectured in 1852, was proven by Appel and Haken in 1976 [4, 5]. Their proof achieved some notoriety due to use of computers to process a lengthy case analysis.

A natural generalization of the above problem is to study the chromatic number of graphs embedded in surfaces other than the plane. The following result, due to Heawood in 1890 [14], generalizes the four color theorem to surfaces other than the plane:

8

Theorem 1.2.4 (Heawood). *Let Σ be a surface with Euler genus $g(\Sigma) \geq 1$. Any graph embedded in Σ can be colored with*

$$H(\Sigma) = \left\lfloor \frac{7 + \sqrt{1 + 24g(\Sigma)}}{2} \right\rfloor$$

colors.

We call $H(\Sigma)$ the *Heawood number* of the surface.

The very interesting result is that this bound is tight for all surfaces with $g(\Sigma) \geq 1$ except for N_2, the Klein bottle. (It is also tight for the sphere S_0, but Heawood's proof does not work for this case). This was finally proved after much work by Ringel and Youngs:

Theorem 1.2.5 (Ringel-Youngs [22]). *For every surface $\Sigma \neq N_2$, $K_{H(\Sigma)}$ embeds into Σ.*

A more recent approach to problems of coloring graphs on surfaces is to ask how the graphs that are not colorable with a certain number of colors look like, and see if there is any algorithmic insight to obtain from that. For example, is $K_{H(\Sigma)}$ the only graph which is not $(H(\Sigma) - 1)$-colorable? Any other graph which contains $K_{H(\Sigma)}$ will also chromatic number at least $H(\Sigma)$, so in order to properly ask this question we need the concept of *critical graphs*.

Definition 1.2.6. A graph G is *k-critical* if $\chi(G) = k$ but $\chi(H) < k$ for any proper subgraph $H \subset G$.

k-critical graphs are the minimal obstructions for k-colorability:

Observation 1.2.7. $\chi(G) \geq k \iff G$ *contains a k-critical graph as a subgraph.*

Theorem 1.2.8 (Dirac; Albertson, Hutchinson [1]). *For $\Sigma \neq N_2, g(\Sigma) \geq 1$, $K_{H(\Sigma)}$ is the only $H(\Sigma)$-critical graph embeddable in Σ.*

This means that determining $(H(\Sigma) - 1)$-colorability for graphs embedded in $H(\Sigma)$ is equivalent to finding an $H(\Sigma)$-clique.

Is it possible that other simple characterizations exist for graphs on surfaces which are not k-colorable for $k < H(\Sigma) - 1$? The answer is affirmative:

Theorem 1.2.9 ([26]). *For any surface Σ and $k \geq 6$, there exist only finitely many k-critical graphs embeddable in Σ.*

This was proved by Thomassen after previous results by Dirac and Gallai for $k \geq 8$ and $k \geq 7$. It is the best possible bound for k since Fisk [11] proved the existence of infinitely many 5-critical graphs on the torus.

Let us briefly discuss the algorithmic implications of this result. For a fixed graph H, it is possible to check whether H is a subgraph of G in time polynomial in the size of G. In fact, by a result of Eppstein [9], for graphs G in a fixed surface it is possible to test subgraph isomorphism in linear time. Therefore,
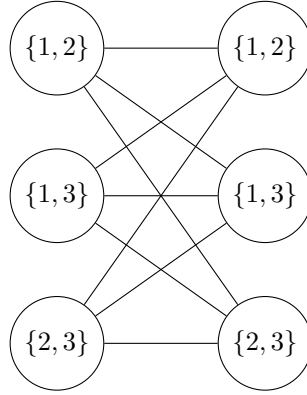
Figure 1.2: An example of a bipartite graph which is not L-colorable for a 2-list-assignment L.

for $k \geq 6$ there exists an algorithm for determining k-colorability in linear time for graphs on a fixed surface, by testing subgraph isomorphism with each of the k-critical graphs in the finite list.

We can ask what is the complexity of testing k-colorability with $k < 6$ for graphs in fixed surfaces. 1-colorability and 2-colorability can be determined in linear time. 3-colorability is NP-complete even for planar graphs [13]. The complexity of 4-colorability in surfaces other than the sphere remains an open problem.

1.3 List Coloring

A variant of graph coloring called *list coloring* was introduced by Vizing [29] and Erdős, Rubin and Taylor [10].

Definition 1.3.1. Let G be a graph. A *list assignment* for G is a function $L : V(G) \to 2^{\mathbb{N}}$. An *L-coloring* of G for a list assignment L is a (proper) coloring ϕ such that $\phi(v) \in L(v) \, \forall v \in V(G)$.

Definition 1.3.2. A *k-list-assignment* is a list assignment L with $|L(v)| \geq k \, \forall v \in V(G)$. A graph G is *k-list-colorable* or *k-choosable* if there exists an L-coloring of the graph **for all** k-list-assignments L. The *list chromatic number* or *choosability* $\chi_\ell(G)$ is the least integer so that G is $\chi_\ell(G)$-list-colorable.

Note the following:

Observation 1.3.3. *G is k-list-colorable $\implies$ G k-colorable, so $\chi_\ell(G) \geq \chi(G)$.*

The reason for this is that by setting $L(v) = \{1, 2, \ldots, k\}$ one retrieves the usual coloring.

But there exist graphs with $\chi_\ell(G) > \chi(G)$: consider $K_{3,3}$ with the list assignment given as in Figure 1.2.

A motivation for this variant of coloring is to consider coloring problems in the usual graph vertex coloring setting in which some of the vertices have already been colored. In this case, the remaining vertices do not have all colors available but only a subset of them (the ones not already picked by colored neighbors) and the subsets for each of the vertices can be different.

We can ask if there is an analogue of the four color theorem for list coloring. In their paper from 1979, Erdős, Rubin and Taylor conjectured the following:

Conjecture 1.3.4 ([10])**.** *1. There exists a planar graph G with $\chi_\ell(G) \geq 5$.*

2. For all planar graphs G, $\chi_\ell(G) \leq 5$.

The first part of the conjecture was proved by Voigt [30] in 1993 providing an example with 238 vertices. One year later, Thomassen proved the second part of the conjecture:

Theorem 1.3.5 (Thomassen's theorem [24])**.** *For all planar graphs G, $\chi_\ell(G) \leq 5$.*

Thomassen actually proved a stronger theorem:

Theorem 1.3.6 (Thomassen's stronger theorem)**.** *Let G be a plane (embedded) graph with outer walk C, and let L be a list assignment satisfying:*

- *$|L(v)| \geq 5$ for all internal vertices.*

- *$|L(v)| \geq 3$ for all $v \in V(C) \setminus \{x, y\}$ where x, y are a pair of adjacent vertices.*

- *$|L(x)| = |L(y)| = 1$, $L(x) \neq L(y)$.*

Then G is L-colorable.

Proof. Suppose we have a counterexample with minimal $|V(G)|$. It is clear that for $|V(G)| \leq 3$ the theorem is true, so we assume $|V(G)| \geq 4$.

First we prove that G is 2-connected. Assume it is not. Then, we have two subgraphs $G_1, G_2 \subset G$ with $G_1 \cup G_2 = G$ and $G_1 \cap G_2 = \{v\}$, with v a cutvertex. Assume, without loss of generality, that $x, y \in G_1$. By minimality of G, G_1 is $L_{\restriction G_1}$ colorable. Let ϕ_1 be a coloring of G_1. Now, let w be a neighbor of v in the outer face of G_2 and consider the list assignment L' for G_2 for which $L'(v) = \{\phi_1(v)\}$, $L'(w) = c$ for some arbitrary $c \in L(w)$, and $L'(u) = L(u) \, \forall u \in V(G_2) \setminus \{v, w\}$. Note that G_2 and L' satisfy the hypothesis of the theorem. Therefore, by minimality of our counterexample, G_2 has a L'-coloring ϕ_2. But now note that, since $\phi_1(v) = \phi_2(v)$, the coloring $\phi(u) = \phi_i(u)$ if $u \in V(G_i)$ is well-defined and is an L-coloring of G, contradiction.

Hence, G is 2-connected and the outer walk C is a cycle. Now we prove that there is no chord in C. The proof is similar to the above argument. Assume there is a chord vw. Then, we have two subgraphs $G_1, G_2 \subset G$ with $G_1 \cup G_2 = G$,
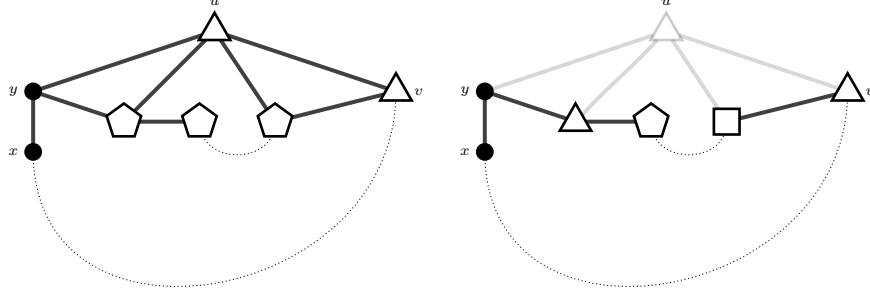
Figure 1.3: Illustration of Thomassen's reduction. At most 2 colors are erased from the lists of u's neighbors.

$G_1 \cap G_2 = \{v, w\}$ and $x, y \subset G_1$. By minimality of G, G_1 has an L-coloring ϕ_1. If we set $L'(v) = \{\phi_1(v)\}$, $L'(w) = \{\phi_1(w)\}$, and $L'(u) = L(u)$ for all other vertices $u \in V(G_2) \setminus \{v, w\}$, then G_2 is L'-colorable and a coloring of G can be constructed.

Now we have that G has no chords or cutvertices. Let u be the neighbor of y in the outer face other than x and let v be the neighbor of u in the outer face other than y (possibly $v = x$). Let $\{c_1, c_2\} \subseteq L(u) \setminus L(y)$. Now, let G' be the graph obtained by removing u from G and let L' be the list assignment for G' in which $\{c_1, c_2\}$ are removed from the lists of the neighbors of u other than v. G' satisfies the hypothesis of the theorem: every vertex in the outer face has list size at least 3, since each of those vertices is either a vertex previously in the outer face of G all of which have their previous lists (the only neighbors of u in the outer face are u and y, since G has no chords), or a previously interior vertex, which has had at most 2 of its ≥ 5 colors removed. So G' has an L'-coloring, which can be extended to an L-coloring of G by coloring u with one of c_1 or c_2 (whichever is not in use by v), contradiction. $\qquad \square$

We can also extend the notion of critical graphs to list coloring:

Definition 1.3.7. A graph is G is *L-critical* for some list assignment L if G has no L-coloring but every proper subgraph $H \subset G$ has an L-coloring. A graph is *k-list-critical* if there exists a $(k-1)$-list assignment L such that G is L-critical. A graph is *minimal k-list-critical* if $\chi_\ell(G) = k$ but $\chi_\ell(H) < k$ for every proper subgraph $H \subset G$.

Notice that here the analogous definition to k-critical graphs for the usual coloring is the definition of **minimal** k-list-critical (terminology from [23], there it is shown that such graphs are in fact the minimal k-list-critical ones with respect to subgraph containment), and the actual definition of k-list-critical is a little different than what one would immediately expect. The reason for this is that in theoretical arguments it is usually more convenient to work with a

pair (G, L) of a graph with a fixed k-list-assignment, and therefore the concept of L-critical graph is more useful.

There are also other criticality notions we will use. In coloring problems, it is useful to consider when a precoloring of a subgraph does not *extend* to the entire graph, that is, there is no coloring of the entire graph under certain constraints which agrees with the coloring of the subgraph.

Definition 1.3.8 (Extending)**.** Let G be a graph, $T \subseteq G$ a subgraph, and L a list assignment for G. For an L-coloring ϕ of T, we say that ϕ *extends* to an L-coloring of G if there exists an L-coloring ψ of G s such that $\phi(v) = \psi(v)$ for all $v \in V(T)$.

It is also interesting to consider graphs which are critical in this setting. To do so, we use the following definition (from [20]):

Definition 1.3.9 (T-critical)**.** Let G, T, L be as above. The graph G is T-*critical with respect to* L if $G \neq T$ and for every proper subgraph $G' \subset G$ such that $T \subseteq G'$, there exists an L-coloring of T that extends to an L-coloring of G', but does not extend to an L-coloring of G. If the list assignment L is clear from context, we just say T-*critical*.

Definition 1.3.10 (ϕ-critical)**.** Let, G, T, L be as above. The graph G is ϕ-critical for a coloring ϕ of T if ϕ extends to every proper subgraph of G containing T but not to G.

In a way similar to the general notion of criticality, we have that graphs for which colorings of T do not extend contain a non-trivial T-critical subgraph:

Lemma 1.3.11. *Let G be a graph, T a subgraph, and L a list assignment for G. If there is an L-coloring ϕ of T that does not extend to G, then G contains a subgraph H with $T \subsetneq H$ which is ϕ-critical, and hence also T-critical with respect to $L_{\restriction_H}$.*

Proof. Let ϕ be the coloring of T that does not extend and let H be a minimal subgraph of G for which ϕ does not extend. Now note that H is ϕ-critical by construction. $\square$

Lemma 1.3.12. *Let G be a graph, T a subgraph, L a list assignment for G, and $H \supset T$ a subgraph of G which is minimal with respect to the following property: for every L-coloring ϕ of T that extends to H, ϕ also extends to G. Then H is T-critical.*

Proof. Suppose not. Then, H contains a proper subgraph H' so that every L-coloring ϕ that extends to H' also extends to H and hence to G. But then H' is a smaller subgraph with that property, contradiction. $\square$

We also have the following lemma from [20]:

Lemma 1.3.13. *Let T be a subgraph of a graph G such that G is T-critical with respect to a list assignment L. Let $A, B \subseteq G$ be such that $A \cup B = G$ and $T \subseteq A$. Then $G[V(B)]$ is $A[V(A) \cap V(B)]$-critical.*

Proof. Let $G' = G[V(B)]$ and $S = A[V(A) \cap V(B)]$. If $G' = S$ there is nothing to say, suppose otherwise that $G' \neq S$ and note that therefore G' contains an edge not in S (in fact, all isolated vertices of G' must be in S). Suppose for a contradiction that G' is not S-critical. Then, by taking a maximal proper subgraph that defies the defintion, there exists an edge $e \in E(G') \setminus E(S)$ such that every L-coloring of S that extends to $G' \setminus e$ also extends to G'. Since G is T-critical and $e \notin E(T)$, there exists a coloring $\phi_{\upharpoonright_T}$ of T that extends to an L-coloring ϕ of $G \setminus e$, but does not extend to an L-coloring of G. However, by the choice of e, the restriction $\phi_{\upharpoonright_S}$ extends to an L-coloring ϕ' of G'. Let ϕ'' be such that $\phi''(v) = \phi'(v) \, \forall v \in V(G')$ and $\phi''(v) = \phi(v) \, \forall v \in V(G) \setminus V(G')$. Now, since $A \cup B = G$, ϕ'' is an L-coloring of G extending $\phi_{\upharpoonright_T}$, a contradiction. $\qquad\square$

For us, it will be more useful in this form:

Lemma 1.3.14 (Gluing Lemma)**.** *Let T be a subgraph of a graph G such that G is T-critical with respect to a list assignment L. Let $A, B \subseteq G$ be such that $A \cup B = G$. Then $G[V(B)]$ is $(A[V(A) \cap V(B)] \cup T)$-critical.*

Proof. Apply 1.3.13 to $A' = A \cup T$ and $B' = B$. $\qquad\square$

The reason we decided to name it "Gluing lemma" in this work is that it is useful to visualize the graph G as made of two separate pieces, A and B, which are glued together along $A[V(A) \cap V(B)]$. In our approach we will frequently use the fact that all T-critical graphs can be "decomposed" in this way.

We will also use the following simple lemmas:

Lemma 1.3.15 (Extension Lemma)**.** *Let G be a T-critical graph. If T' is a subgraph with $T \subseteq T' \subset G$, then G is T'-critical.*

Proof. For any subgraph H with $T \subseteq T' \subseteq H \subset G$, there exists a coloring $\phi_{\upharpoonright_T}$ of T that extends to a coloring ϕ of H but not to G by T-criticality. Then, $\phi_{\upharpoonright_{T'}}$ is a coloring of T' that extends to H but not to G. $\qquad\square$

Lemma 1.3.16 (Duplication Lemma)**.** *Let G be a T-critical graph with respect to a list assignment L and let G' be a graph , $T' \subset G'$ be a subgraph, L' be a list assignment for G' and $f : V(G') \to V(G)$ be a function such that:*

1. *$f_{\upharpoonright_{V(G') \setminus V(T')}}$ is an isomorphism between $G'[V(G') \setminus V(T')]$ and $G[V(G) \setminus V(T)]$.*

2. *For all $v \in V(T')$, $f(v) \in V(T)$ and f is surjective.*

3. *$L'(v) = L(f(v))$ for all $v \in V(G')$.*

4. *For every $v \in V(G)$, the image of the set $N(v) \cap V(T')$ under f is equal to $N(f(v)) \cap V(T)$.*

Then, G' is T'-critical with respect to L'.

Proof. Let H' be any subgraph with $T' \subseteq H' \subset G'$ and let H be the image of H' under f. H can be seen to satisfy $T \subseteq H \subset G$, and by T-criticality there is a coloring $\phi_{\restriction_T}$ that extends to a coloring ϕ of H but does not extend to G. Consider the L'-coloring ψ of H' defined by $\psi(v) = \phi(f(v))$. Then, $\psi_{\restriction_{T'}}$ is a coloring that extends to H' but does not extend to G', because if it did extend to a coloring ψ' of G' then $\phi_{\restriction_T}$ would extend to G by setting the color $\psi'(f^{-1}(v))$ to each vertex $v \in V(G) \setminus V(T)$. $\qquad\square$

Often, in list coloring we want to impose some restriction on the sizes of the lists, but we don't want the same sizes for all the vertices as in k-list-colorability. We will use the following terminology:

Definition 1.3.17. Let G be a graph and $f : V(G) \to \mathbb{N}$ a function. We say a list assignment L is a *f-list-assignment* if $|L(v)| \geq f(v)$ for all $v \in V(G)$. We say G is *f-list-colorable* if G is L-colorable for every f-list-assignment L. We say G is *f-list-critical* if G is L-critical for some f-list-assignment L.

And when drawing graphs whose vertices have prescribed list sizes by such an f, we will use the pictorial notation of Figure 1.4 to denote the list sizes of the vertices.

Figure 1.4: Shapes used for the vertices of the graphs according to their list sizes.

Finally, we can ask if there are results for list coloring of graphs on surfaces as there were for the usual coloring. The argument of Heawood also works for list coloring, so for every surface Σ the graphs embedded in Σ are all $H(\Sigma)$-list-colorable.

We have the analogous result that $K_{H(\Sigma)}$ is the only obstruction for $(H(\Sigma) - 1)$-list-colorability:

Theorem 1.3.18 ([16]). *For a surface Σ with $g(\Sigma) \geq 1$, $\Sigma \neq N_2$, $K_{H(\Sigma)}$ is the only minimal $H(\Sigma)$-list-critical graph embeddable in Σ.*

And Postle proved in his PhD thesis in 2012 the result analogous to Thomassen's:

Theorem 1.3.19 ([18]). *For $k \geq 6$ there exist only finitely many k-list-critical graphs embeddable in a given surface Σ.*

The goal we set out for this thesis is to find the explicit list of 6-list-critical graphs for the torus.

Chapter 2

Critical Graphs on the Torus

In this chapter we delve into the details of critical and list-critical graphs on the torus, including Postle's approach for list-critical graphs on surfaces and Thomassen's approach for 6-critical graphs on the torus. From studying these, we develop our own, computer-aided approach for 6-list-critical graphs on the torus.

2.1 An Overview of Postle's Approach

Here we briefly explain Postle's approach in [18] to obtain the result on the finiteness of 6-list-critical graphs in general surfaces, mentioning specially those intermediate results or definitions we will also use in our approach. The results obtained by Postle are very non-explicit in the sense that the (unspecified) constant in the size bounds for the graphs he obtains is extremely large and hence useless for our purpose of finding an explicit characterization. Nevertheless, given that our approach is primarily guided by this work we consider it of interest to provide a exposition of the main argument.

The results developed in [18] in 2012 have been published successively in journal articles afterwards, often with improvements in exposition or in the strength of the result. We refer to the corresponding published article in the discussion of each particular result.

2.1.1 Notation and Terminology

Postle works mainly in a setting similar to the hypothesis of Thomassen's stronger theorem: list assignments L which have list sizes of length at least 5 for interior vertices and at least 3 for exterior vertices with some exceptions. This setting is encapsulated in the concept of *canvas*.

Definition 2.1.1 (Canvas)**.** We say that (G, S, L) is a *canvas* if G is a connected plane graph with outer walk C, S is a subgraph of C, and L is a list assignment such that $|L(v)| \geq 5 \, \forall v \in V(G) \setminus V(C)$ and $|L(v)| \geq 3 \, \forall v \in V(C) \setminus V(S)$. If S is a path, we say (G, S, L) is a *path-canvas* or a *wedge*. If $S = C$ and C is a cycle, then (G, C, L) is a *cycle-canvas*.

Note: in some places like [20], the term "canvas" is used for what Postle calls in [18] "cycle-canvas".

We can restate Thomassen's Stronger Theorem in these terms:

Theorem 2.1.2. *If (G, P, L) is a path-canvas and $|V(P)| \leq 2$, then G is L-colorable.*

Definition 2.1.3 (Critical canvas)**.** We say that a canvas (G, S, L) is *critical* if it is S-critical with respect to L.

In the context of canvases, we usually talk of *separating vertices* or *separating subgraphs* as graphs which, when removed, disconnect vertices of S from other vertices in S.

2.1.2 Variations on Thomassen's Condition

Much of the technical work on Postle's thesis relies in a careful study of what happens if one varies the condition on Theorem 1.3.6. One of the most elegant (and also useful) results is the following strengthening to Thomassen's Stronger Theorem, originally conjectured by Hutchinson in [15].

Theorem 2.1.4 (Two Lists of Size Two Theorem [19])**.** *If G is a plane graph with outer cycle C, $v_1, v_2 \in C$ and L is a list assignment with $|L(v)| \geq 5$ for all $v \in V(G) \setminus V(C)$, $|L(v)| \geq 3$ for all $v \in V(C) \setminus \{v_1, v_2\}$, and $|L(v_1)| = |L(v_2)| = 2$, then G is L-colorable.*

Or, in the language of canvases:

Theorem 2.1.5. *If (G, S, L) is a canvas with $|V(S)| = 2$ and $|L(v)| \geq 2$ for $v \in S$, then G is L-colorable.*

This theorem is not true when one of the two vertices has list of size 1. In fact, Postle characterizes exactly when it fails:

Definition 2.1.6 (Coloring Harmonica)**.** Let G be a plane graph and L a list assignment for G. Given an edge uv and a vertex w both from the outer face of G, we say that (G, L) is a *coloring harmonica from uv to w* if either:

- G is a triangle with vertex set $\{u, v, w\}$ and $L(u) = L(v) = L(w)$ with $|L(u)| = 2$, or

- There exists a vertex z incident with the outer face of G such that uvz is a triangle in G, $L(u) = L(v) \subseteq L(z)$, $|L(u)| = |L(v)| = 2$, $|L(z)| = 3$, and the pair (G', L') is a coloring harmonica from z to w, where G' is obtained by deleting **one or both** of the vertices u, v and L' is obtained from L by $L'(z) = L(z) \setminus L(u)$ and $L'(x) = L(x)$ for all other vertices $z \neq x \in V(G')$.

Given two vertices u, w in the outer face of G, we say (G, L) is a *coloring harmonica from u to w* if there exist vertices x, y incident with the outer face of G such that uxy is a triangle in G, $|L(u)| = 1$, $L(x) \setminus L(u) = L(y) \setminus L(u)$, $|L(x) \setminus L(u)| = 2$, and (G', L') is a coloring harmonica from xy to w, where G' is obtained from G by removing u, and L' is obtained from L by seting $L'(x) = L'(y) = L(x) \setminus L(u)$ and $L'(z) = L(z)$ for every $z \in V(G') \setminus \{x, y\}$.

We say that (G, L) is a coloring harmonica if it is a coloring harmonica from uv to w or a coloring harmonica from u to w for some u, v, w as specified earlier.
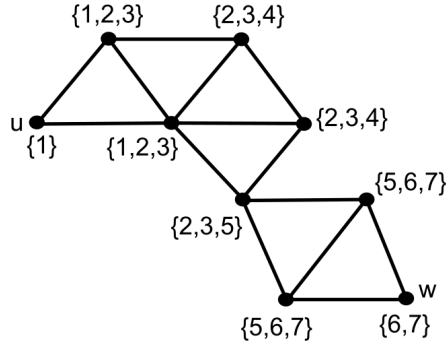
Figure 2.1: Example of coloring harmonica from u to w (taken from [21])

See the example in Figure 2.1 (from [21]) for some clarity with respect to this mutually recursive definition. Note that the definition makes it clear that graphs which contain a coloring harmonica as a subgraph are not L-colorable.

Theorem 2.1.7 (One List of Size One and One List of Size Two Theorem [21]). *Let G be a plane graph with outer cycle C, let $p_1, p_2 \in V(C)$, and let L be a list assignment with $|L(v)| \geq 5$ for all $v \in V(G) \setminus V(C)$, $|L(v)| \geq 3$ for all $v \in V(C) \setminus \{p_1, p_2\}$, $|L(p_1)| \geq 1$ and $|L(p_2)| \geq 2$. Then G is L-colorable if and only if the pair (G, L) does not contain a coloring harmonica from p_1 to p_2.*

Studying conditions of the sizes of the lists in the boundary in which the graph is not L-colorable like this one is also useful, because such conditions arise when dealing when reductions and therefore characterizing which are the critical graphs in such settings can give fruitful results.

Thomassen already studied when does the coloring of a path of length 2 not extend:

Definition 2.1.8 (Bellows). We say that a path-canvas (G, P, L) with $P = p_0 p_1 p_2$ is a *bellows* (terminology from [18]) or a *generalized wheel* (terminology from [27]) if either:

- G has no interior vertices and its edge set consists of the edges of the outer cycle plus all edges from p_1 to vertices of the outer cycle. In this case, we say that (G, P, L) is a *fan*.

18

- G has one interior vertex u and its edge set consists of the edges of the outer cycle plus all edges from u to vertices of the outer cycle. In this case, we say that (G, P, L) is a *turbofan*.

- G can be formed by gluing two smaller bellows from the edges $p_1 p_2$ and $p_0 p_1$ respectively.

Theorem 2.1.9 ([27], Theorem 3). *If $T = (G, P, L)$ is a path-canvas with path length 2, then G is L-colorable unless T has a bellows as a subcanvas.*

Postle studies when the coloring of two paths of length 1 does not extend. He finds the following obstruction:

Definition 2.1.10 (Accordion). We say that a canvas $T = (G, P_1 \cup P_2, L)$ with P_1, P_2 distinct paths of length 1 is an *accordion* with *ends* P_1, P_2 if T is a bellows with $P_1 \cup P_2$ path of length 2 or T is the gluing of two smaller accordions $T_1 = (G_1, P_1 \cup U, L)$ with ends P_1, U and $T_2 = (G_2, P_2 \cup U, L)$ with ends U, P_2 along a chord $U = u_1 u_2$ where $|L(u_1)|, |L(u_2)| \leq 3$.

The main result he obtains is that if the two paths are sufficiently far apart, then the graph contains a proportionally large accordion or a coloring harmonica as a subgraph.

Theorem 2.1.11 (Bottleneck Theorem, loosely stated). *If $T = (G, P \cup P_0, L)$ is a canvas with P, P_0 distinct edges of C with $d(P, P_0) \geq 14$, then either there exists an L-coloring of G, or there exists a subcanvas $(G_0, U_1 \cup U_2, L)$ of T where $d_{G_0}(U_1, U_2) = \Omega(d_G(P, P_0))$ which is an accordion or a coloring harmonica.*

This result, along with coloring and structural properties of accordions and harmonicas, is often used as a technical lemma when proving the following results.

2.1.3 Linear Bound on Critical Cycle-Canvases

Postle proves the following result:

Theorem 2.1.12 ([20]). *Let G be a plane graph with outer cycle C and L a 5-list-assignment for G. Let H be a minimal subgraph of G such that every L-coloring of C that extends to an L-coloring of H also extends to an L-coloring of G. Then H has at most $19|V(C)|$ vertices.*

Or, equivalently stated in the language of critical canvases:

Theorem 2.1.13. *If (G, C, L) is a critical cycle-canvas, then $|V(G)| \leq 19|V(C)|$.*

The equivalence of the two statements is given by Lemma 1.3.12. This result is interesting in its own right because by Lemma 1.3.14, all faces of a T-critical graph which do not separate vertices from T are in fact critical cycle-canvases, and therefore what the result tells us is that for such graphs there is only finitely

many kinds of faces that can appear for each given cycle length. This gives us a lot of information of how critical graphs look like.

A first observation that can be made is that critical cycle-canvases (in which C is indeed a simple cycle) are 2-connected, so each face is bounded by a cycle:

Lemma 2.1.14. *If (G, C, L) is a critical cycle-canvas, then it is 2-connected.*

Proof. If G is not 2-connected, then there exist subgraphs A, B such that $A \cup B = G$ with $|V(A) \cap V(B)| \leq 1$ and $|V(B) \setminus V(A)| \geq 1$. Assume $C \subseteq A$ and apply Lemma 1.3.14 to get that B is $A[V(A) \cap V(B)]$-critical, contradicting Theorem 1.3.6. $\square$

The key result in Postle's proof of the linear bound for cycles is the following theorem about the structure of critical cycle-canvases:

Theorem 2.1.15 (Cycle Chord or Tripod Theorem)**.** *If (G, C, L) is a critical cycle-canvas, then either*

1. *C has a chord in G, or*

2. *there exists a vertex $v \in V(G) \setminus V(C)$ with at least three neighbors on C such that at most one of the faces of $G[\{v\} \cup V(C)]$ includes a vertex or edge of G.*

Using this result, Postle carefully examines what happens near the boundary cycle in order to define some quantities related to sums of lengths of faces and proves that certain inequalities with those quantites are mantained when adding tripods in critical canvases.

2.1.4 The Two Precolored Triangles Theorem

Next, Postle proves the following theorem:

Theorem 2.1.16. *There exists d such that the following holds. Let G be a planar graph and T_1, T_2 triangles in G at distance at least d. Let L be a 5-list-assignment of G. Then, every L-coloring of $T_1 \cup T_2$ extends to an L-coloring of G.*

(Postle refers to the graphs with two precolored triangles and all other vertices with list sizes at least 5 as *prism-canvases*, even though they are not a type of canvas).

The value of d that Postle obtains is not explicitly stated, but it is on the order of 100. However, we conjecture that 4 or 5 suffices.

Conjecture 2.1.17. *Let G be a planar graph and T_1, T_2 triangles in G with distance between them at least 5. Let L be a 5-list-assignment of G. Then, every L-coloring of $T_1 \cup T_2$ extends to an L-coloring of G.*

The argument that Postle uses to prove his result is as follows. First, he proves that one can precolor a path between the two triangles in such a way that, when deleting the path and deleting the corresponding colors from the lists of neighboring vertices, all remaining non-precolored vertices have lists of size at least 3. The proof of this begins with the simple observation that each vertex outside a shortest path has at most 3 neighbors inside the path. Using planarity properties, a shortest path can be found so that it can be colored in such a way that the vertices with 3 neighbors inside the path only see two different colors from their lists.

After precoloring and deleting the path between the two triangles, a canvas $(G, P_1 \cup P_2, L)$ is obtained. If there was a precoloring of the triangles that did not extend, then the canvas contains a critical canvas, and by Theorem 2.1.11 it contains a proportionally long accordion or harmonica. Postle proves that this (together with some technical details related to how the path between the triangles was chosen) implies that in the original graph there must be a long chain of separating triangles so that the graph between each separating triangle pertains to one of three very specific types, which he calls tetrahedral, octahedral or hexadecahedral bands.

Finally, he proves that for a sufficiently long chain of this type, any precoloring of the innermost and outermost triangles extends to the whole chain. This proves the theorem, because of the following observation:

Proposition 2.1.18. *Let G be a plane graph with L a list assignment, T_1, T_2 two facial triangles with T_1 bounding the infinite face of G, and T_1', T_2' two triangles such that T_1' is a separating triangle between T_1 and T_2' and T_2' is a separating triangle between T_1' and T_2. Denote by $G[T_1', T_2']$ the subgraph comprised between the two triangles T_1', T_2'. If there exists some L-coloring of $T_1 \cup T_2$ that does not extend to G, then there exists some L-coloring of $T_1' \cup T_2'$ that does not extend to $G[T_1', T_2']$.*

Proof. By Theorem 1.3.6, the coloring on T_1 extends to $G[T_1, T_1']$ and the coloring on T_2 extends to $G[T_2', T_2]$. The coloring of $T_1' \cup T_2'$ given by this extensions can not extend to $G[T_1', T_2']$ by the assumption that the original coloring of $T_1 \cup T_2$ did not extend to G. $\square$

2.1.5 Cylinder-Canvases and Hyperbolic Families

Finally, Postle generalizes the previous setting of critical prism-canvases to *critical cylinder-canvases*, where a cylinder-canvas is like a prism-canvas but the precolored cycles can have length greater than 3. It generalizes Theorem 2.1.16 by showing that for every pair of cycle lengths, there exists a distance d (depending logarithmically on the cycle sizes) so that all critical cylinder-canvases have the cycles at distance at most d. It also generalizes Theorem 2.1.12 by showing that critical cylinder-canvases have a bound on their size linear in the sum of the cycle lengths.

Then, he defines *hyperbolic families*: families of graphs embedded in surfaces which have certain properties related to having planar graphs resulting from

performing *excisions* in the surface always having bounded size. We do not delve into the details because the topological arguments for the general case are not of interest for us, but we will explain some similar operations on the torus in the section explaining our approach.

The idea here is that the linear bounds on the cycle-canvases and the cylinder-canvases make the 6-list-critical graphs on the torus be an hyperbolic family. Then, the result we want about the finiteness of 6-list-critical graphs is proven for the more general setting of hyperbolic families. In fact, it is proven that the number of vertices of graphs in such a family embeddable in a surface is bounded linearly by the genus of the surface.

The general setting of hyperbolic surfaces allows Postle to prove more properties and bounds for 6-list-critical graphs on surfaces, such as:

Theorem 2.1.19. *Let G be a graph 2-cell-embedded in a surface Σ and let L be a 5-list-assignment. If $ew(G) \geq O(\log g(\Sigma))$, then G is L-colorable.*

2.2 Critical Graphs on the Torus for (usual) Vertex Coloring

In this section we discuss the result from Thomassen in [25] that characterizes the critical graphs for 5-coloring (not 5-list-coloring) on the torus.

2.2.1 The Critical Graphs

Theorem 2.2.1 ([25]). *A graph G embeddable on the torus is 5-colorable if and only if it does not contain the following subgraphs:*

- *K_6.*

- *$C_3 + C_5$.*

- *$K_2 + H_7$, where H_7 is a 7-vertex graph known as the Moser spindle.*

- *T_{11}, where T_{11} is a triangulation of the torus with 11 vertices.*

Where $+$ denotes the join of two graphs: their disjoint union with all pairs of vertices from different graphs joined by edges.

If a graph is not 5-colorable, it is not 5-list-colorable, so all graphs that contain any of the above subgraphs are not 5-list-colorable. We conjecture that this characterizes the 5-list-colorable graphs on the torus too:

Conjecture 2.2.2. *A graph G embeddable on the torus is 5-list-colorable if and only if it does not contain the following subgraphs: K_6, $C_3 + C_5$, $K_2 + H_7$, T_{11}.*

This means that those are the minimal 6-list-critical graphs on the torus. Note that there may be additional 6-list-critical graphs embeddable on the torus, but what we are conjecturing is that they all contain those subgraphs. For example:
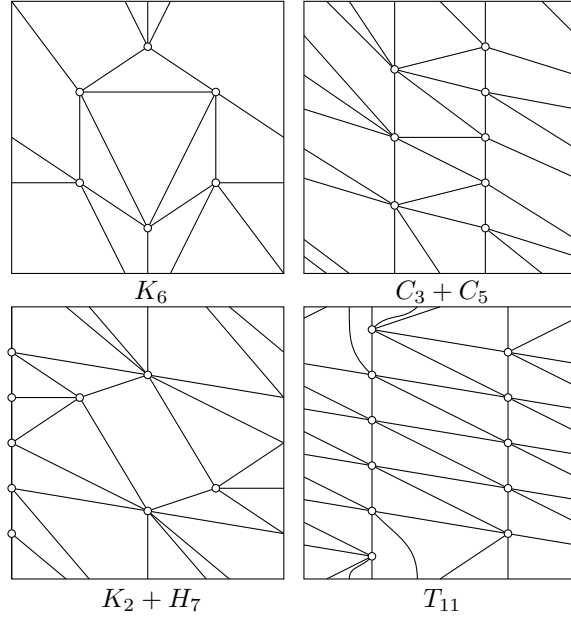
Figure 2.2: 6-critical graphs embedded on the torus.

Observation 2.2.3. K_7 *is* 6-*list-critical.*

Proof. Consider the following 5-list-assignment for K_7: $L(v_1) = L(v_2) = L(v_3) = L(v_4) = L(v_5) = \{1, 2, 3, 4, 5\}$, $L(v_6) = L(v_7) = \{1, 2, 3, 4, 6\}$. K_7 is not L-colorable, since there are only 6 available colors. But any subgraph is L-colorable. Let's give a coloring ϕ for $K_7 \setminus v_i v_j$. If $i, j \leq 5$, then setting $\phi(v_i) = \phi(v_j) = 5$ and $\phi(v_7) = 6$ leaves 4 vertices to be colored with 4 colors. If $i \leq 5$ and $j \geq 6$, then setting $\phi(v_i) = \phi(v_j) = 1$, $\phi(v_{13-j}) = 6$ leaves 4 vertices to be colored with 4 colors. If $\{i, j\} = \{6, 7\}$, then $\phi(v_i) = \phi(v_j) = 6$ leaves 5 vertices to be colored with 5 colors.

Hence, K_7 is L-critical for a 5-list-assignment L, and is therefore 6-list-critical. □

2.2.2 An Overview of Thomassen's Approach

Thomassen's article where he characterizes the graphs on the torus ([25]) predates his result on finitely many 6-critical graphs for all surfaces ([26]). For the characterization of 6-critical graphs on the torus, he only uses elementary, relatively straighforward arguments that work on specifically in the torus. We briefly summarize his approach here in order to discuss which arguments can be reused for the list-coloring case.

First, Thomassen considers the case when the minimum degree is at least 6.

Proposition 2.2.4. *If a graph G embedded on the torus has $\delta(G) \geq 6$, then:*

1. *G is 6-regular.*

2. *G is a triangulation of the torus.*

Proof. We apply Euler's formula: let V, E, F be the number of vertices, edges and faces in the embedding, respectively. We have that $\delta(G) \geq 6 \implies V \leq \frac{1}{3}E$ with equality iff G is 6-regular, and $F \leq \frac{2}{3}E$ with equality iff G is a triangulation. Then $0 = V - E + F \leq \frac{1}{3}E - E + \frac{2}{3}E = 0$, so we have equality on both inequalities. $\qquad\square$

Using the proposition above, Thomassen then proves the following:

Proposition 2.2.5 (3.2 in [25]). *Let G be a 6-regular graph on the torus. If G contains a vertex v, such that $\{v\} \cup N(v)$ induces a nonplanar graph, then $G = K_7$ or G is obtained from K_8 or K_9 by deleting the edges of a 1-regular or 2-regular subgraph.*

The study of 6-regular graphs on the torus without vertices whose neighborhood induces a nonplanar graph was already done by Thomassen in his previous paper [28], in the context of finding all tilings of the torus in order to prove a conjecture by Babai about vertex-transitive graphs.

He obtains the following result:

Theorem 2.2.6. *Let G be a graph embedded on the torus with $\delta(G) \geq 6$. Then G is 5-colorable unless $G = K_7$ or $G = T_{11}$.*

This part of the argument is the one that can be generalized to list coloring, but it is not useful to our approach, so we don't delve into the details.

Let us now describe Thomassen's argument for general graphs.

He assumes a minimum counterexample G_0 to Theorem 2.2.1 (the counterexample has minimum number of vertices, maximum number of edges restricted to that, and some other assumptions about details we will not discuss here). By the previous result, there must be a vertex $v_0 \in V(G_0)$ with degree ≤ 5, and the degree of v_0 is in fact equal to 5 by minimality of the counterexample.

Consider two vertices $x, y \in N(v_0)$ which are not adjacent (if all the vertices of $N(v_0)$ were adjacent, then G_0 would contain K_6, a contradiction). Let G_{xy} be the graph obtained from $G_0 \setminus \{v_0\}$ by identifying the vertices x and y. G_{xy} can be embedded in the torus by modifying the embedding of G_0. If G_{xy} were 5-colorable, then we would have a 5-coloring of G_0 by assigning the same color to x and y and coloring v_0 with a color not appearing in its 5 neighbors. Hence, G_{xy} is not 5-colorable and by minimality of our counterexample it contains K_6, $C_3 + C_5$, $K_2 + H_7$ or T_{11}.

The above argument works for all pairs x, y of non-adjacent vertices in $N(v_0)$, so potentially we can have many different obstructions for each of the corresponding G_{xy} subgraphs. But we can prove that, by minimality, there can not be much else in G_0 apart from these obstructions arising from all the G_{xy} subgraphs. More precisely:

Proposition 2.2.7. *For any non-adjacent $x, y \in N(v_0)$, let G'_{xy} a copy of K_6, $C_3 + C_5$, $K_2 + H_7$ or T_{11} in G_{xy}, and let G''_{xy} be the induced subgraph of G_{xy} by the vertex set of G'_{xy}. Then G_0 consists of v_0, $N(v_0)$, the edges between vertices of $\{v_0\} \cup N(v_0)$, and the union over all non-adjacent $x, y \in N(v_0)$ of the graph obtained from G''_{xy} by splitting the contracted vertex into x and y.*

Proof. We will prove that the graph described above, which is a subgraph of G_0, is not 5-colorable. This means, by the assumptions of minimality of vertices and maximality of edges, that G_0 is in fact equal to that subgraph.

If the subgraph had a 5-coloring, then two non-adjacent vertices x, y of $N(v_0)$ would have the same color. But by then identifying the two vertices we can get a 5-coloring of G''_{xy}, which contains the non-5-colorable subgraph G'_{xy}, contradiction.

$\square$

Note that, since the maximum number of vertices in a critical graph is 11, this means that G_0 has at most $(11-1) \cdot \binom{5}{2} + 6 = 106$ vertices, and hence what remains is a finite problem.

Thomassen uses some more arguments to narrow down the remaining possibilities for G_0, but we can already see an important point of failure of this argument for list-coloring: in the proof of Proposition 2.2.7, it is used that a necessary and sufficient condition for a coloring of $G_0 \setminus \{v_0\}$ to extend to v_0 is that two neighbors of v_0 have the same color. In list coloring, this condition is not necessary. So we cannot conclude that the minimum counterexample is the union of the graphs induced by the obstructions in G_{xy} and the argument breaks down here.

2.3 Our Approach

The bounds on the size of 6-list-critical graphs on the torus given by Postle's approach are way too large to be of any use to us in our search of the explicit list for the torus. However, we can take inspiration in his approach and in particular in how it was useful to carefully study critical canvases. The key modification is that, instead of proving abstract bounds for sizes of critical canvases as in Theorem 2.1.12, we will be working with the explicit, full list of critical canvases of a certain type and size.

In order to obtain those canvases, we will be using computer search, so this will be a computer-aided proof, like the Four Color Theorem's (but using very different techniques). The reason we consider it feasible to perform this search is that results like Theorem 2.1.15 give very structured descriptions of critical canvases - so we don't face enormously huge space of searching among all the planar graphs with a given size bound, but instead have a much more restricted search space.

Once we have generated some critical canvases, our idea is to take advantage of them in a way similar to Postle's approach - but this time, since we will work
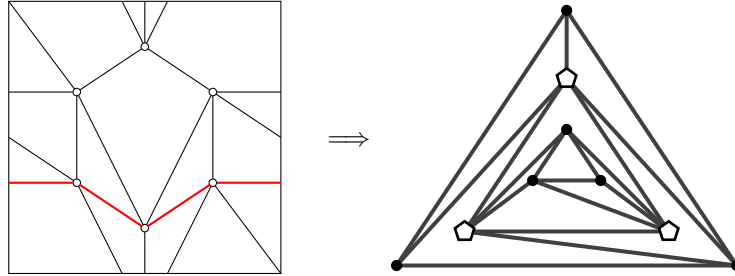
Figure 2.3: Cutting K_6 into a critical prism-canvas.

with the explicit canvases instead of loose bounds, the hope is that we can use them to obtain also the explicit list of 6-list-critical graphs.

Since the torus is not a complicated surface, there are simple manipulations that relate 6-list-critical graphs on the torus with critical prism-canvases and cycle-canvases, as we will see now.

2.3.1 Cutting Through a Non-Contractible Cycle

Consider the torus S_1. If one cuts through non-contractible curve one obtains a cylinder, which can then be projected to the plane.

We can do this to get planar graphs from graphs 2-cell-embedded on the torus: we can cut through a non-contractible cycle, duplicate the vertices on the cycle so they appear on both sides of the resulting cylinder, and then project this cylinder into the plane. See Figure 2.3 for an example with K_6, one of the 6-critical graphs embedded on the torus.

The reason why this is useful to us is the following observation:

Observation 2.3.1. *Let G be a graph embedded on the torus, let G' be a graph obtained by this procedure from G, let C_1, C_2 be the two cycles of G; corresponding to the non-contractible cycle C of G we cut through, let L be a list assignment for G and let L' be the corresponding list assignment for G' (with the same lists for all vertices, duplicated for C_1 and C_2).*

If G is L-critical, then G' is $(C_1 \cup C_2)$-critical with respect to the list assignment L'.

Proof. Consider a subgraph $(C_1 \cup C_2) \subseteq H' \subset G'$ and the corresponding subgraph $C \subseteq H \subset G$. By L-criticality of G, H is L-colorable with a coloring ϕ. Consider the corresponding coloring ϕ' of H'. Then, $\phi'_{\restriction C_1 \cup C_2}$ is a coloring of $(C_1 \cup C_2)$ which extends to H' but not to G', since if it extended to G' we would have a L'-coloring of G' in which the corresponding vertices of C_1 and C_2 have the same colors and therefore we would be able to retrieve an L-coloring of G. $\square$

This, together with Theorem 2.1.16, already places a restriction on how the 6-list-critical graphs on the torus can be: if the graph has a non-contractible
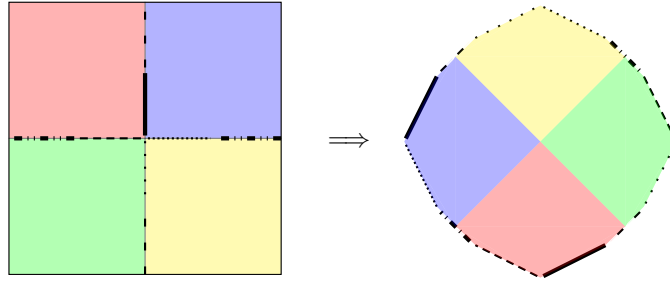
Figure 2.4: Cutting the torus into the plane.
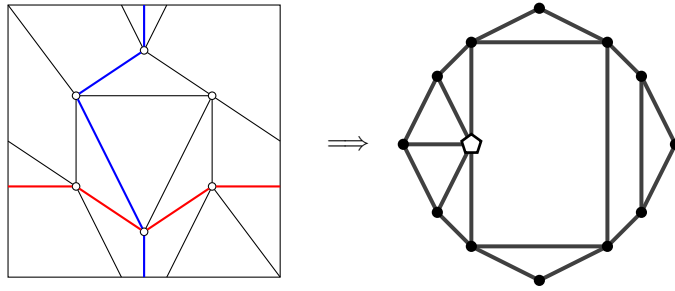
Figure 2.5: Cutting K_6 into a critical cycle-canvas.

triangle, then it must also have a not too large non-contractible cycle not homotopically equivalent to the non-contractible triangle, since if the two precolored triangles are at distance d in the resulting planar graph, then there is such a non-contractible cycle of length $\leq d + 1$ in the original graph.

An interesting implication is also that we can do the process backwards: if we have a critical prism-canvas, we can "fuse the two triangles" to get a graph on the torus which is *possibly* 6-list-critical.

2.3.2 Cutting Across Two Non-Contractible Cycles

Instead of just cutting through one non-contractible cycle to get a cylinder and project it to the plane, we can cut through two non-homotopically-equivalent non-contractible cycles to get the plane, as in Figure 2.4. This can be thought also as first cutting through a cycle to get a cylinder, and then cutting through the cylinder to get the plane.

The idea here is again that we can use this to get critical planar graphs from critical graphs on the torus, but now we get a cycle-canvas instead of a prism-canvas. See Figure 2.5 for an example again with K_6.

Observation 2.3.2. *Let G be a L-critical graph embedded on the torus with L a 5-list-assignment, and let G' be the corresponding planar graph resulting from this procedure with outer face C corresponding to the two cycles we cut through.*

27

Then (G', C, L) is a critical cycle-canvas.

Proof. Similar argument to the previous observation. □

If the cycles we cut through have lengths a and b, the resulting cycle-canvas has cycle length $2(a + b)$. This gives a bound for the sizes of 6-list-critical graphs in terms of the sizes of their non-contractible cycles via Theorem 2.1.12. However, here the more interesting idea is the possibility of running the process backwards: that is, of generating candidates for 6-list-critical graphs from critical cycle-canvases. That is because, as we will see, we have a procedure for the generation of all critical cycle-canvases by computer.

2.3.3 Our Plan for Graphs With a Non-Contractible Triangle

Using the above two constructions, we can already devise a plan for finding all 6-list-critical graphs on the torus which have at least one non-contractible triangle.

1. Generate all critical cycle-canvases with small cycle sizes (ideally at least up to size 14 or, better yet, 16). Also, depending on the approach followed in the next step, it will also be necessary to generate small critical path-canvases or other kinds of critical graphs.

2. Prove Conjecture 2.1.17. Here, we will try to take advantage of the computer-generated graphs in order to obtain the tightest possible bound instead of Postle's loose bound.

3. By cutting through the non-contractible triangle of any 6-list-critical graph, we obtain a critical prism-canvas, and therefore we can reconstruct all possible candidates from either the list of all critical prism-canvases (if we obtained such a list from the previous step) or a list of all critical cycle-canvases of size ≤ 16 (by cutting across the non-contractible triangle and the other non-contractible cycle of length ≤ 5 corresponding to the shortest path between the two precolored triangles in the prism-canvas).

This plan only works for 6-list-critical graphs with edge-width 3, which is the case for the 6-critical graphs for the previous section. We don't expect to have additional 6-list-critical graphs with larger edge-width, especially given Theorem 2.1.19, so our hope would be to find an argument independent of this plan to prove that any 6-list-critical graph on the torus must have a non-contractible triangle.

The rest of this thesis will be dedicated to describe how to carry out this plan. For the first step, we will explain how to perform the computer generation of such critical graphs in Chapters 3 and 4. Chapter 3 will be about the computer representation of the canvases and the algorithms used for the generation of candidates to be critical canvases, while Chapter 4 will be on how to test that such candidates are indeed critical. The techniques explained in Chapters 3

and 4 will be applicable not only to cycle-canvases but also to other types of canvases or graphs with prescribed list sizes more generally.

For the second step, in Chapter 5 we will explain different approaches to proving Conjecture 2.1.17 using the computer-generated graphs. As of the writing of this thesis, we have not been able find a proof of this results, because there have been computational and conceptual obstacles in the approaches we have tried. We will also discuss those setbacks.

Finally, in Chapter 6 we will summarize the computational results obtained by the implementation of the ideas discussed in the previous chapters and we will expose the partial results we have been able to obtain with respect to the third step of the plan.

Chapter 3

Generation of Critical Graphs

In this chapter we describe algorithms for processing and generating critical canvases via computer search.

3.1 Representation of Canvases

The first step is deciding how to represent canvases in our algorithms. Recall that a canvas T is a tuple (G, S, L) where G is a plane graph, S is a subgraph of the outer face and L is a list assignment for G satisfying some conditions. Here, in each algorithm we will usually be working with one particular family of canvases at a time, for example cycle-canvases or path-canvases with a fixed size of cycle or path, so the information about the subgraph S can be "implicit" in each different representation for each different algorithm instead of working with a general representation that allows all canvases. Also, in some scenarios we will be working with conditions on the list assignment L which are different from the ones in the definition of canvas. In this section we intend to just expose some general ideas about how the representation of graphs in this context can be done, which will be afterwards applied in different scenarios.

The most important thing to state is that we will not be interested in storing the list assignment L at all. This is because there is a significant combinatorial explosion in the number of list assignments to be considered and we are interested in the graphs themselves, not the list assignments. Also, most of the results we will be using such as Lemma 1.3.14 or Theorem 2.1.15 are directly related to subgraphs and not list assignments, and while they are in theory stated with respect to a fixed list assignment, it is more useful in practice to not consider the list assignment at all.

Thus, when we generate all critical canvases (G, S, L), what we will actually be doing is generate all pairs (G, S) such that there exists some list assignment L so that (G, S, L) is a critical canvas. In some scenarios, we will also be interested
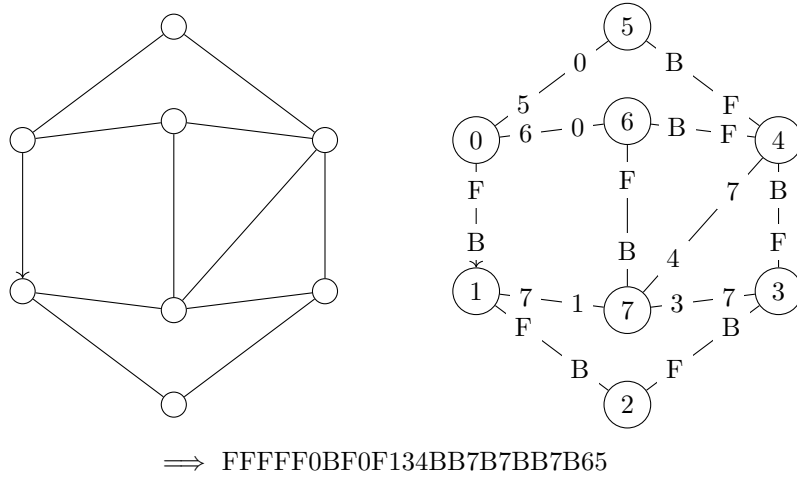
$\implies$ FFFFF0BF0F134BB7B7BB7B65

Figure 3.1: Example of a DFS traversal transcript for a graph. It is the lexico-graphically smallest transcript in this case.

in storing the prescribed size of the list assignment for each vertex: that is, we will be storing a tuple (G, S, f) with $f : V(G) \to \mathbb{N}$ so that we will only be considering list assignments L with $|L(v)| = f(v)$, but other than that we will not store information about the actual list assignment.

We store the information of the graph G with an adjacency list. We will also be interested in storing the planar embedding of the graph: to do so, we order the edges in the adjacency list of each vertex according to their clockwise order in the embedding (as in a *rotation system*). This information, together with the information of which vertices are in the outer face, is enough to reconstruct the embedding.

We will want to test when two canvases are isomorphic. More generally, we will want to have a canonical form for each canvas, so that given a set of canvases $\mathcal{S}$ and a new canvas T, we can check whether there is a canvas isomorphic to T in $\mathcal{S}$ by checking the presence of the corresponding canonical form of T in an associative array with the canonical forms of the canvases in $\mathcal{S}$.

In order to produce the canonical form, define the *transcript of the DFS traversal starting at edge $u \to v$* as the string generated by procedure dfsTranscript in Algorithm 1. That is, we do a depth-first traversal of the graph following the edges on each vertex in clockwise order, assigning labels to vertices based in the order in which we first visit them and storing information for each edge we visit in the traversal: F for edges towards a new vertex in the traversal, B for edges towards the immediately previous vertex in the traversal stack (which signifies the end of the edges for the current vertex and the return to the previous vertex of the stack) and the label of the other endpoint for other edges. See Algorithm 1 for details.

We compute such string for all the edges of the outer face as starting edges,

Algorithm 1: Canonization of Plane Graphs.

```
/* Returns lexicographically smallest DFS Transcript       */
function canonicalForm(G)
    s ← "#";
    for uv in outer face of G do
        t ← dfsTranscript(G, u, v);
        if t <ℓ s then
            s ← t;
        end
    end
    return s;
end
function dfsTranscript(G, u, v)
    s ← "";
    A ← array(|V(G)|);
    A[u] ← firstIdentifier();
    ;      /* A is the array that stores identifiers of already
           visited vertices, initially initialized with ∅ for all
           vertices */
    for w neighbor of u in clockwise order starting with v do
        dfs(G, w, u, s, A);
    end
    return s;
end
function dfs(G, u, p, s, A)
    if A[u] ≠ ∅ then
        s ← s + A[u];
        return;
    end
    A[u] ← nextIdentifier();
    s ← s + 'F';
    for v ≠
      p neighbor of u in clockwise order starting in the next neighbor after p
      do
        dfs(G, v, u, s, A);
    end
    s ← s + 'B';
end
```

and we take the lexicographically smallest one as the canonical form. It is clear that two plane graphs have the same canonical string if and only if they have isomorphic (in terms of the rotation system) embeddings.

3.2 Generation of Critical Cycle-Canvases

Our algorithm for the generation of critical cycle-canvases is based on Theorem 2.1.15. This theorem says that every critical cycle-canvas can either be decomposed into two smaller critical cycle-canvases through a chord in the outer face, it can be decomposed into a "tripod", a vertex v with at least 3 neighbors in C, and a smaller critical cycle-canvas contained in the only nonempty face incident with v. In these decompositions, it is possible that instead of a smaller critical canvas we get an empty canvas, which is technically not critical.

This implies that we can generate all critical cycle-canvases from smaller cycle-canvases by gluing cycle-canvases through outer face edges to get a canvas with a chord, or by adding a tripod to the outside of a cycle-canvas. We then have to check whether the resulting canvas is indeed critical, since the decomposition into two smaller critical cycle-canvases is a necessary but not sufficient condition for criticality. We will see how to do this in Chapter 4.

If we are generating cycle-canvases with cycle length ℓ, then a chord partitions the cycle-canvas into two cycle-canvases of length a, b with $a, b \geq 3$ and $a + b = \ell + 2$, so $a, b \leq \ell - 1$ and therefore if we have generated all cycle-canvases with cycle length $< \ell$ we can generate all cycle-canvases with cycle length ℓ with a chord. In the case of adding a tripod, though, if the vertex v of the tripod is adjacent to only three adjacent vertices in the outer face, then the smaller cycle-canvas has the same cycle length as the larger cycle-canvas.

In order to resolve this, what we do is first generate all the cycle-canvases obtained from cycle-canvases with smaller cycle size, enqueue the resulting critical canvases, and then process the canvases from the queue and add tripods to three consecutive vertices in all possible ways, enqueueing the new critical cycle-canvases that are found. See Algorithm 2 for a detailed description.

Note that we only need to add tripods with 3 adjacent neighbors since vertices with a larger number of neighbors in the outer face can be obtained by first adding chords and then adding finally adding a tripod with 3 neighbors. However, often we are interested in just generating chordless critical canvases. In that case, we do need to add tripods of all sizes. The modified algorithm for chordless critical cycle-canvases is Algorithm 3.

3.3 Generation of Critical Wedges

We are will be not only interested in generating critical cycle-canvases, but also critical path-canvases or wedges. There are infinitely many of those for path length greater than 1, but as we will see in coming sections, we will be able to have a finite number of them if we impose additional conditions.

Algorithm 2: Generation of Critical Cycle-Canvases.

```
/* Generate critical canvases of cycle size ℓ, including empty one
*/
```
function *generateCriticalCycleCanvases(ℓ)*

    for $i = 3, \ldots, \ell - 1$ **do**

        $S_i \leftarrow$ generateCriticalCycleCanvases(i);

    end

    $S \leftarrow \{\text{emptyCycle}(\ell)\}$;

    for $a = 3, \ldots, \ell - 1$ **do**

        $b \leftarrow \ell - a + 2$;

        for $G_1 \in S_a$ **do**

            for $G_2 \in S_b$ **do**

                $T \leftarrow$ fuseChordSet(G_1, G_2);

                ;   `/* Set of cycle-canvases obtained by fusing` G_1 `and` G_2 `along outer cycle edges in all possible ways */`

                for $G \in T$ **do**

                    if $G \notin S$ *AND isCritical(G)* **then**

                        $S \leftarrow S \cup \{G\}$;

                  end

                end

            end

        end

    end

    for $k = 3, \ldots, \ell - 1$ **do**

        for $G_1 \in S_k$ **do**

            $T \leftarrow$ addTripodSet($G_1, \ell - k + 3, 3$);

            ;   `/* Set of cycle-canvases obtained by adding a tripod with 3 neighbors in the outer face to get a cycle-canvas of length` ℓ `in all possible ways */`

            for $G \in T$ **do**

                if $G \notin S$ *AND isCritical(G)* **then**

                    $S \leftarrow S \cup \{G\}$;

                end

            end

        end

    end

    $Q \leftarrow$ Queue(S);

    while Q *is not empty* **do**

        $G_1 \leftarrow$ first(Q);

        dequeue(Q);

        $T \leftarrow$ addTripodSet($G_1, 3, 3$);

        for $G \in T$ **do**

            if $G \notin S$ *AND isCritical(G)* **then**

                $S \leftarrow S \cup \{G\}$;

                enqueue(Q, G);

            end

        end

    end

    return S;

end

Algorithm 3: Generation of Chordless Critical Cycle-Canvases.

```
/* Generate chordless critical canvases of cycle size ℓ,
   including empty one                                    */
```
function *generateChordlessCriticalCycleCanvases(ℓ)*

    for $i = 3, \ldots, \ell - 1$ **do**

        $S_i \leftarrow$ generateChordlessCriticalCycleCanvases(i);

    end

    $S \leftarrow \{\text{emptyCycle}(\ell)\}$;

    for $k = 3, \ldots, \ell - 1$ **do**

        for $j = 3, \ldots, \ell - k + 3$ **do**

            for $G_1 \in S_k$ **do**

                $T \leftarrow$ addTripodSet($G_1, \ell - k + 3, j$);

                for $G \in T$ **do**

                    if $G \notin S$ *AND* *isCritical*(G) **then**

                        $S \leftarrow S \cup \{G\}$;

                    end

                end

            end

        end

    end

    $Q \leftarrow$ Queue(S);

    while Q *is not empty* **do**

        $G_1 \leftarrow$ first(Q);

        dequeue(Q);

        $T \leftarrow$ addTripodSet($G_1, 3, 3$);

        for $G \in T$ **do**

            if $G \notin S$ *AND* *isCritical*(G) **then**

                $S \leftarrow S \cup \{G\}$;

                enqueue(Q, G);

            end

        end

    end

    return S;

end

Fortunately, we also have an analogue of Theorem 2.1.15 for wedges:

Theorem 3.3.1 (Wedge Chord or Tripod Theorem)**.** *If (G, P, L) is a 2-connected critical wedge, then either*

1. *The outer walk C has a chord in G, or*

2. *there exists a vertex $v \in V(G) \setminus V(P)$ with at least three neighbors on P such that at most one of the faces of $G[\{v\} \cup V(P)]$ includes a vertex or edge of G.*

Proof. Assume not. Then, we will show that every L-coloring of P extends to an L-coloring of G, contradiction. Let ϕ be any L-coloring of P, $G' = G \setminus P$ and $L'(v) = L(v) \setminus \{\phi(u) : u \in V(P)$ neighbor of $v\}$ for each $v \in V(G')$. Note that $|L'(v)| \geq 5$ for every interior vertex of G'. Let C' be the outer walk of G' and let v_1, v_2 be the two vertices of G' that were adjacent to the two endpoints of P in G. Note that $|L'(v)| \geq 3$ for all $v \in C' \setminus \{v_1, v_2\}$ since G was 2-connected and had no chords, and $|L(v_1)|, |L(v_2)| \geq 2$. Hence, G' is L'-colorable by Theorem 2.1.4. $\qquad\square$

(This version of the theorem is slightly different than the one proved by Postle in [18]).

Based on this theorem, we can design an algorithm to generate critical wedges similar to the one used to generate critical canvases. The main additional considerations we need to take into account are:

-   We have to generate non-2-connected critical wedges by gluing smaller critical wedges along the cutvertices. Observe that the cutvertices must necessarily be part of P.

-   Now adding a chord whose endpoint is the vertex next to the endpoints of the precolored path produces a wedge with the same path length, so we need to add those chords in the part of the algorithm using a queue.

-   Chords with one endpoint in the precolored path and another endpoint outside the precolored path need to be treated separately from chords with two endpoints in the precolored path. Note that it is not possible to have a chord with two endpoints outside the precolored path (or one endpoint in an endpoint in the precolored path) because of Theorem 1.3.6.

Algorithm 4: Generation of Critical Wedges.

```
/* Generate critical wedges of path size ℓ, assuming those
of path size < ℓ have been already generated and are stored
in variable Wᵢ, and the biconnected ones are in Bᵢ.        */
```
function *generateCriticalWedges(ℓ)*
    $B_\ell \leftarrow$ generateBiconnectedCriticalWedges(ℓ);
    $W_\ell \leftarrow B_\ell \cup$ generateNonBiconnectedCriticalWedges(ℓ);
end
function *generateBiconnectedCriticalWedges(ℓ)*
    $S \leftarrow$ emptyCycle(ℓ) $\cup$ wedgesFromPathChords(ℓ) $\cup$
     wedgesFromNonPathChords(ℓ) $\cup$ wedgesFromTripods(ℓ);
    $Q \leftarrow$ Queue(S);
    while Q *is not empty* **do**
        $G_1 \leftarrow$ first(Q);
        dequeue(Q);
        $T \leftarrow$ addTripodSet($G_1, 3, 3$);
        $T \leftarrow T \cup$ addSizeTwoWedgesInEndpointsSet(G_1);
```
        ;   /* Set of wedges obtained by fusing wedges of size
         two in the endpoints of wedge G₁ in all possible
         ways */
```
        for $G \in T$ **do**
            if $G \notin S$ *AND isCritical*(G) **then**
                $S \leftarrow S \cup \{G\}$;
                enqueue(Q, G);
            end
        end
    end
    return S;
end
function *generateNonBiconnectedCriticalWedges(ℓ)*
    $S \leftarrow$ emptyPath(ℓ);
    for $a = 2, \ldots, \ell - 2$ **do**
        $b \leftarrow \ell - a$;
        for $G_1 \in W_a$ **do**
            for $G_2 \in W_b$ **do**
                $G \leftarrow$ fuseEndpoints(G_1, G_2);
                if $G \notin S$ *AND isCritical*(G) **then**
                    $S \leftarrow S \cup \{G\}$;
                end
            end
        end
    end
    return S;
end

Algorithm 5: Functions Generating Critical Wedges from Smaller Path Lengths

---

function *wedgesFromPathChords(ℓ)*
  $S \leftarrow \{\}$;
  for $a = 2, \ldots, \ell - 1$ **do**
    for $G' \in B_a$ **do**
      $T \leftarrow \text{extendPathChordSet}(G', \ell - a)$;
      ;   /* Set of wedges obtained by adding $\ell - a$ consecutive new
       precolored vertices between two adjacent precolored vertices
       of G', making the previous edge of G' a chord in the path.
       */
      for $G \in T$ **do**
        if $G \notin S$ *AND isCritical*(G) **then**
          $S \leftarrow S \cup \{G\}$;
        end
      end
    end
  end
  return S;
end
function *wedgesFromNonPathChords(ℓ)*
  $S \leftarrow \{\}$;
  for $a = 4, \ldots, \ell - 1$ **do**
    $b \leftarrow \ell - a + 3$;
    for $G_1 \in B_a$ **do**
      for $G_2 \in B_b$ **do**
        $T = \text{fuseChordSet}(G_1, G_2)$;
        for $G \in T$ **do**
          if $G \notin S$ *AND isCritical*(G) **then**
            $S \leftarrow S \cup \{G\}$;
          end
        end
      end
    end
  end
  return S;
end
function *wedgesFromTripods(ℓ)*
  $S \leftarrow \{\}$;
  for $k = 3, \ldots, \ell - 1$ **do**
    for $j = 3, \ldots, \ell - k + 3$ **do**
      for $G' \in B_k$ **do**
        $T \leftarrow \text{addTripodSet}(G', \ell - k + 3, j)$;
        for $G \in T$ **do**
          if $G \notin S$ *AND isCritical*(G) **then**
            $S \leftarrow S \cup \{G\}$;
          end
        end
      end
    end
  end
  return S;
end

Chapter 4

Criticality Testing

In this chapter we describe algorithms used to determine list-criticality of graphs. Recall that we are not storing the explicit list assignment L for our graphs, so what we want to check is whether there exists a L so that the graph is critical with respect to that L. However, even if L was fixed, to determine criticality would still be a computationally hard problem. What we will do instead is check for weaker properties, and therefore admit some false positives, that is, some graphs we identify as list-critical for which actually no suitable L exist. Our hope is that the tests will be exhaustive enough so that finiteness results such as Theorem 2.1.12 still hold for the weaker properties we are testing, and our algorithms terminate. We will see that indeed, the algorithms described here work very well in practice at discarding non-critical graphs.

We do not have a fixed list assignment for our graph, but we do have prescribed list sizes f for each vertex. So in this section we will see how to test for f-criticality.

4.1 Degree Properties

We can start with an easy observation:

Observation 4.1.1. *In a f-list-critical graph, $d(v) \geq f(v)$ for all vertices v.*

So if we find a vertex with degree less than the prescribed list size, we can conclude that the graph is not list-critical. However, this is a very weak test. We can incorporate another test concerning the vertices with $d(v) = |L(v)|$: there is the following result by Gallai showing that the subgraph induced by those vertices must have a certain structure, generalizing the classical Brooks theorem for vertex coloring:

Theorem 4.1.2 (Gallai [12]). *Let G be a f-list-critical graph and let H be the subgraph of H induced by the vertices with $d(v) = f(v)$. Then each 2-connected component of H is a complete graph or an odd cycle.*
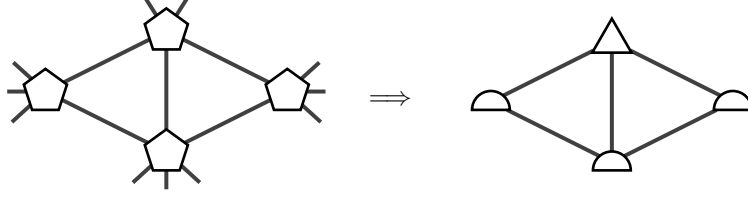
Figure 4.1: Illustration of 4.2.1: from a subgraph, we get a graph with prescribed list sizes.
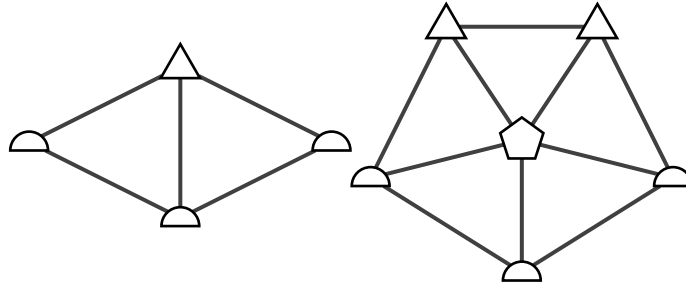

Figure 4.2: Some colorable reducible configurations.

4.2 Reducible Configurations

The *method of reducible configurations* is a usual technique in graph coloring problems. It consists in identifying subgraphs or other structures that can not appear in critical graphs. We have the following observation:

Proposition 4.2.1. *Let G be an f-list-critical graph, and let H be an induced subgraph of G such that $\forall v \in H, f(v) > d_{G \setminus H}(v)$, where $d_{G \setminus H}(v)$ is the number of neighbors of v which are not in H. Then H is not g-list-colorable, where $g(v) = f(v) - d_{G \setminus H}(v)$.*

Proof. If $G = H$, it is immediate. Assume $H \subsetneq G$, and let G be L-critical. Let ϕ be a coloring of $G \setminus H$, and let L' be the g-list-assignment of H given by $L'(v) = L(v) \setminus \{\phi(u) : u \in N_G(v), u \notin H\}$. Then H is not L'-colorable: since otherwise, the L-coloring ϕ of $G \setminus H$ would extend to G, contradiction. □

We can consider Observation 4.1.1 to be a particular case of Proposition 4.2.1. In Figure 4.2 we see a couple of examples of small graphs that are always f-colorable, so one possible test we can add to our criticality testing procedure is to search for occurrences of those graphs as induced subgraphs, and if one is found then conclude that the graph is not critical.

However, there are also reducible configurations which are not f-colorable.

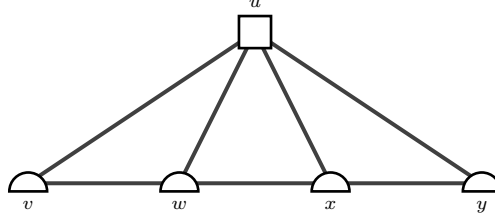**Definition 4.2.2.** A graph G is said to be *f-reducible* if there is a proper

Figure 4.3: A non-colorable reducible configuration.

subgraph $H \subsetneq G$ so that for all f-list-assignments L of G, if H is $L_{\upharpoonright_H}$-colorable then G is L-colorable.

Proposition 4.2.3. *Let G be an f-list-critical graph, and let H be an induced subgraph of G such that $\forall v \in H, f(v) > d_{G \setminus H}(v)$, where $d_{G \setminus H}(v)$ is the number of neighbors of v which are not in H. Then H is not g-reducible, where $g(v) = f(v) - d_{G \setminus H}(v)$.*

Proof. Assume not, and let H' be the corresponding subgraph of H for which all g-colorings extend. Let G be L-critical. There exists an L-coloring ϕ of $G \setminus (H \setminus H')$. Let L' be the g-list-assignment of H given by $L'(v) = L(v) \setminus \{\phi(u) : u \in N_G(v), u \notin H\}$. Note that $\phi_{\upharpoonright_{H'}}$ is an L'-coloring of H', so there must be an L'-coloring ψ of H. But then the coloring Φ given by $\Phi(v) = \psi(v)$ for $v \in H$, $\Phi(v) = \phi(v)$ for $v \in G \setminus H$ is a L-coloring of G, contradiction. $\square$

Observation 4.2.4. *The graph depicted in Figure 4.3 with prescribed list sizes f is f-reducible.*

Proof. Let us characterize the f-list-assignments L for which the graph is not L-colorable. First, note that if $L(v) \neq L(w)$ or $L(x) \neq L(y)$ one can precolor both w and x so that the resulting graph with the corresponding colors removed from the lists is a path with list sizes at least 2 in all vertices except in one endpoint, with list size at least 1, so it is colorable. Therefore, we have $L(v) = L(w)$ and $L(x) = L(y)$. Then, note that $L(v), L(x) \subseteq L(u)$, since otherwise by coloring one vertex with a color not in $L(u)$ one can always color the graph. Therefore, the f-list-assignments for which the graph is not L-colorable are those of the form $L(u) = \{A, B, C, D\}$, $L(v) = L(w) = \{A, B\}$, $L(x) = L(y) = \{C, D\}$. But for those list assignments the graph without edge wx is also not L-colorable, so for any f-list-assignment L the graph is L-colorable if and only if the subgraph without the edge wx is L-colorable. $\square$

4.3 The Alon-Tarsi Method

While checking for the small reducible configurations we found in the previous section is helpful, it is not good enough, because there are larger graphs which

are always f-colorable but do not contain any of the reducible configurations. One could augment the list of configurations to check by manually inspecting those graphs when they are encountered, but this is ineffective and also inefficient because induced subgraph isomorphism testing starts being very expensive with larger subgraphs. We would, then, like to have a systematic method to find when graphs with prescribed list sizes f are f-list-colorable. Alon and Tarsi provided a useful criterion:

Theorem 4.3.1 (Alon-Tarsi, [3]). *Let G be a directed graph on vertices $v_1, \ldots, v_n$, and let L be an assignment of lists to vertices of G such that $|L(v_i)| \geq d^+(v_i) + 1$ for $i = 1, \ldots, n$. If G has a different number of even and odd spanning eulerian subgraphs, then G has an L-coloring.*

Here *even* and *odd* eulerian subgraphs refer to the number of edges. We will explain the proof of this theorem here, since it will give us insight into how to implement it in a more efficient way than enumerating all eulerian subgraphs. For the proof we will need an algebraic result known as Combinatorial Nullstellensatz.

Theorem 4.3.2 (Combinatorial Nullstellensatz [2]). *Let $\mathbb{K}$ be a field and $p(x_1, \ldots, x_n) \in \mathbb{K}[x_1, \ldots, x_n]$ be a nonzero polynomial and let $t_1, \ldots, t_n$ be nonnegative integers such that the degree of p is $t_1 + \ldots + t_n$ and the coefficient of $\prod_{i=1}^n x_i^{t_i}$ in p is nonzero. Let $S_1, \ldots, S_n$ be subsets of $\mathbb{K}$ such that $|S_i| \geq t_i + 1$. Then there exist $a_1 \in S_1, \ldots, a_n \in S_n$ such that $p(a_1, \ldots, a_n) \neq 0$.*

4.3.1 Proof of the Alon-Tarsi Theorem

Definition 4.3.3. For a directed graph G with n vertices $v_1, \ldots, v_n$, we define its *graph polynomial* $p_G(x_1, \ldots, x_n)$ as:

$$p_G(x_1, \ldots, x_n) = \prod_{\overrightarrow{v_i v_j} \in E(G)} (x_j - x_i). \tag{4.1}$$

Observation 4.3.4. *If we associate each color with a different number (we work in, say, $\mathbb{C}$), then ϕ is a proper coloring of G if and only if $p_G(\phi(v_1), \ldots, \phi(v_n)) \neq 0$.*

Proposition 4.3.5. *Let G be as in the hypothesis of the Alon-Tarsi theorem. If the coefficient of p_G at $\prod_{i=1}^n x_i^{d^+(v_i)}$ is non-zero, then G has an L-coloring.*

Proof. The total degree of p_G is $|E(G)| = d^+(v_1) + \ldots + d^+(v_n)$. By the Combinatorial Nullstellensatz, one can find $\phi(v_1) \in L(v_1), \ldots, \phi(v_n) \in L(v_n)$ so that $p_G(\phi(v_1), \ldots, \phi(v_n)) \neq 0$. $\qquad \square$

Observation 4.3.6. *Let G be a directed graph, and denote by $x_G = \prod_{\overrightarrow{v_i v_j} \in E(G)} x_j$. Let D_1 and D_2 be two orientations of a graph, denote by $|D_1 \Delta D_2|$ the number of edges with different direction in the two orientations. We have:*

$$p_G(x_1, \ldots, x_n) = \sum_{D \text{ orientation of } G} (-1)^{|D \Delta G|} x_D \tag{4.2}$$

Proposition 4.3.7. *The absolute value of the coefficient of p_G at $\prod_{i=1}^n x_i^{d^+(v_i)}$ is the difference between odd and even eulerian spanning subgraphs of G.*

Proof. Consider the set $\mathcal{S} = \{D \text{ orientation of } G : x_D = \prod_{i=1}^n x_i^{d^+(v_i)}\}$, that is, the set of orientations which have the same indegrees as G. We claim that this set is in bijection with the set of eulerian spanning subgraphs of G, with a bijection maps orientations with even $|D\Delta G|$ to even spanning subgraphs and orientations with odd $|D\Delta G|$ to odd spanning subgraphs. This implies the result by (4.2).

The bijection is as follows: for each $D \in \mathcal{S}$ map it to the subgraph with edges given by the edges in G which have opposite orientation as edges in D. Since the indegrees of D and G are the same, this subgraph is eulerian, and this map is clearly invertible and hence a bijection. $\qquad\square$

This concludes the proof of Theorem 4.3.1.

4.3.2 Implementation of the Alon-Tarsi Method

Here we will explain how to use Theorem 4.3.1 in practice to check the f-colorability of a graph. The first thing we have to note is that Theorem 4.3.1 is stated with respect to a directed graph, but this is immaterial to our needs. We only have the prescribed list sizes f, and for each such prescription there can be different orientations of that satisfy the $f(v) \geq d^+(v) + 1$ condition, and we could apply the theorem to each of those. So instead of using the combinatorial characterization of the theorem, what we will do is directly compute the polynomial p_G with respect to an arbitrary orientation of the graph, and if any monomial $\prod x_i^{e_i}$ with $e_i < f(i)$ has a nonzero coefficient we will conclude f-colorability.

It will be convenient to think of the computation of the polynomial p_G using the expression (4.2). This way, we can compute the coefficients by enumerating all orientations of the graph and summing the signs. See Algorithm 6 for the implementation.

However, there are multiple improvements that can be made over this naive implementation.

The most immediate one is that we only care about the coefficients of the monomials corresponding to orientations with indegrees less than $f(v)$ in each vertex f (we call such orientations f-*bounded orientations*, and the coefficients of the corresponding terms of the polynomial f-*bounded coefficients*). Therefore we can store only coefficients of the polynomial corresponding to f-bounded orientations. We would also like to generate only f-bounded orientations, so that we do not have to iterate over all $2^{|E(G)|}$ orientations of the graphs.

If we generate orientations by a orienting each edge one by one in a recursive backtracking fashion, we want to know when to cut a branch that is not going to lead to an f-bounded orientation. The easiest way is to cut a branch when the branch trivially does not correspond anymore to an f-bounded orientation, that is, when the indegree of some of the vertices already reaches $f(v)$. It can be

---
Algorithm 6: Naive Alon-Tarsi.

; /* Returns true if the Alon-Tarsi method determines that G is f-colorable. */
function *alonTarsi(G, f)*

    $S \leftarrow \text{allOrientations(G)}$;

    ; /* allOrientations(G) generates all orientations of G (e. g. by orienting each edge by recursive backtracking) */

    $p_G \leftarrow \text{emptyAssociativeArray()}$;

    ; /* We represent the polynomial p_G by an associative array mapping the array of n integers $e_1,\ldots,e_n$ to the coefficient of $\prod x_i^{e_i}$ initialized to 0 on all values. */

    for $D \in S$ **do**

        $s \leftarrow 1$;

        $e \leftarrow \text{array}(n)$;

        for $\{u,v\} \in E(G)$ **do**

            ; /* We pick an arbitrary order for u,v in each edge in order to determine an arbitrary orientation of G (e. g. set $u < v$ in the integer labeling we use to represent G). */

            if $\overrightarrow{uv} \in E(D)$ **then**

                $e_v \leftarrow e_v + 1$;

            else

                $s \leftarrow -s$;

                $e_u \leftarrow e_u + 1$;

        end

        $p_G[e] \leftarrow p_G[e] + s$;

    end

    for $(e, c_e) \in p_G$ **do**

        ; /* Iterate over the coefficients of the polynomial and if there is some nonzero coefficient of an appropiate monomial, return success. */

        if $e_v < f(v) \,\forall v \in V(G)$ **then**

            if $c_e \neq 0$ **then**

                return true;

            end

        end

    end

    return false;

end

done in a more sophisticated way by reducing the problem of checking whether a partial orientation can be extended to an f-bounded orientation to a problem of maximum bipartite matching. This way, all the branches that do not lead to an f-bounded orientation can be immediately discarded by this test, and we can obtain an enumeration of all f-bounded orientations in polynomial time for each f-bounded orientation.

However, it turns out it is more efficient to just do it in the trivial way since the overhead of solving the bipartite matching subproblems is not worth the more eager cutting of branches. The trivial branch cutting can be improved by selecting the next edge that is going to be oriented following some heuristic that makes it more likely for branches to be cut off earlier. For example, we can orient first the edges that whose orientation is forced (because one of the endpoints has already indegree $f(v) - 1$) and then prioritize the ones whose both endpoints have indegrees close to $f(v)$.

Implementing the above improvements already makes a very substantial impact in the execution time of the algorithm, but it is still slow when the number of f-bounded orientations is very large and storing all the f-bounded coefficients of the polynomial can also incur in a large memory usage.

There is a different approach which can work a bit better in practice, based not on enumerating all orientations but on computing the contribution of each edge to the polynomial separately. We can think of it as using (4.1) instead of (4.2) and computing the product $\prod_{\overrightarrow{v_i v_j} \in E(G)} (x_i - x_j)$ term by term. The important ideas in making this be actually faster than the enumerating orientations approach are:

- We *truncate* the partial results so that we don't store non-f-bounded coefficients: we only care about f-bounded coefficients, so we can just avoid storing the other coefficients of the polynomial, not only in the final result but also in the intermediate results we get after multiplying each $(x_i - x_j)$ term.

  In a similar fashion to the previous discussion in which we were enumerating f-bounded orientations, given the intermediate polynomial after multiplying some of the terms we can check whether each monomial will have some contribution to a f-bounded coefficient in the final result – i.e., whether the partial orientations corresponding to the indegree sequence corresponding to the monomial can be extended to a f-bounded orientations – by solving a bipartite matching problem. But, just as before, we find that the overhead of performing this extendability test is not worth the additional pruning and that it is more efficient in practice to just discard non-f-bounded monomials.

- We also discard monomials whose coefficients are 0 in the intermediate results and don't store them in memory. This is the most important optimization and it is what makes this approach better in practice than the one about enumerating orientations, since in "hard-to-color" graphs we expect that we will already start seeing many zeroes in intermediate

results and this can contribute to an important reduction in time and memory usage.

- As before, the order in which we process edges matters. We want to have many truncations and zero coefficents early, so that we store fewer data for our partial results. This can be done heuristically by prioritizing edges with small values of f at their endpoints.

Further improvements are possible, but we do not describe them here since this is already the implementation we have actually used for this project. For a more detailed analysis and comparison of techniques for the efficient implementation of the Alon-Tarsi method, see [8].

Remember that the Alon-Tarsi method does not always detect colorable graphs successfully. For example, the second graph in Figure 4.2 is not recognized by Alon-Tarsi as a colorable graph. In addition, for bigger graphs Alon-Tarsi is significantly slower than the search for fixed small induced subgraphs. And remember that we also have non-colorable reducible configurations, for which Alon-Tarsi does not apply. So it is still advisable to first check for the small reducible configurations we found above before running Alon-Tarsi.

4.4 Recursive Colorability Testing

We know that if our graph has a reducible configuration as an induced subgraph, then it can not be f-critical. And we can determine whether a induced subgraph is an always-colorable reducible configuration using the Alon-Tarsi method. So one possible idea would be to apply Alon-Tarsi on every induced subgraph to check if our graph has any reducible configuration. We can easily see that this is not very efficient – we run the already slow Alon-Tarsi method on an exponential number of graphs. On the other extreme, we can run the Alon-Tarsi method only on the entire graph, falsifying f-criticality only if it is f-colorable. This is not good, either – we can easily imagine graphs which are not f-colorable but are not f-critical either, because they contain some reducible configuration as an induced subgraph.

In fact, as Figure 4.4 shows, this scenario can happen very easily when, for example, generating critical cycle-canvases, because it suffices to have a triangle with list sizes 2 to make the graph non-f-colorable. The specific reducible configuration that appears in Figure 4.4 is one of the small ones that we can check specifically before running Alon-Tarsi, but the more general scenario of having a reducible configuration plus a non-colorable subgraph does also often happen with larger reducible configurations and it is important that we deal with it automatically, using Alon-Tarsi to detect the reducible configuration.

What we do is the following: if Alon-Tarsi returns a negative answer to being applied to the entire graph, we try to find a *minimal* non-colorable subgraph which is causing this negative answer (such as the triangle in Figure 4.4). We do that by arbitrarily removing vertices and checking whether the graph is still non-f-colorable, and doing so until we have a graph which becomes f-colorable

Algorithm 7: Optimized Alon-Tarsi with truncated multiplication.

```
;  /* Returns true if the Alon-Tarsi method determines that G
   is f-colorable.  */
function alonTarsi(G, f)
    p_G ← emptyAssociativeArray();
    E ← sortEdges(E(G), f);
    ;            /* sortEdges sorts the edges according to some
     heuristic, e.g.  increasing by sum of values of f at
     endpoints */
    for {u, v} ∈ E do
        q ← emptyAssociativeArray();
        ;                                    /* q = p_G(x_u − x_v) */
        for (e, c_e) ∈ p_G do
            if e_u + 1 < f(u) then
                r ← e;
                r_u ← r_u + 1;
                q[r] ← q[r] + c_e;
            end
            if e_v + 1 < f(v) then
                r ← e;
                r_v ← r_v + 1;
                q[r] ← q[r] − c_e;
            end
        end
        p_G ← emptyAssociativeArray();
        ;    /* Update p_G with the nonzero coefficients of q */
        . for (e, c_e) ∈ q_G do
            if c_e ≠ 0 then
                p_G[e] ← c_e;
            end
        end
    end
    for (e, c_e) ∈ p_G do
        if c_e ≠ 0 then
            return true;
        end
    end
    return false;
end
```

47
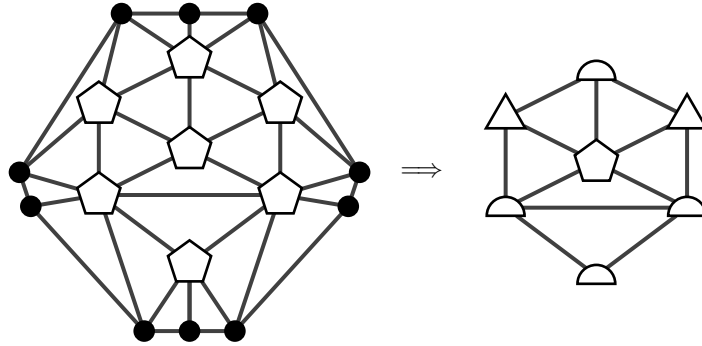
Figure 4.4: An example of a non-f-colorable graph with a reducible configuration obtained from a cycle-canvas.

when any vertex is removed. Then we remove subgraph H (and reduce the list sizes of vertices neighboring H correspondingly, like when considering reducible configurations) and recursively apply the test to the remaining graph. We see that this works for the graph in Figure 4.4: first, the triangle is removed and then the reducible configuration is identified. This recursive approach also handles more general cases when there are multiple non-colorable subgraphs obstructing the colorability of the entire graph. Overall, the results of using this heuristic are very satisfactory.

Algorithm 8: Recursive Colorability Testing.
---

function *containsColorableSubgraph(G)*

    if G *is empty* **then**

        | return false;

    end

    if *alonTarsi(G)* **then**

        | return true;

    end

    $H \leftarrow$ minimalNonColorable(G);

    return containsColorableSubgraph(precolorSubgraph(G, H));

    ;   /* `precolorSubgraph`(G, H) `returns the graph` G `removing the subgraph` H `and decrementing the list sizes of the neighbors of vertices of` H `appropiately.` */

end

function *minimalNonColorable(G)*

    for $v \in V(G)$ **do**

        if *not alonTarsi(removeVertex(G,v))* **then**

            | return minimalNonColorable(removeVertex(G,v));

        end

    end

    return G;

end

Chapter 5

Approaches to the Two Precolored Triangles Theorem

In this chapter we explain different approaches to proving Conjecture 2.1.17.

5.1 Cycle-Canvas Strangulation

We can generate all critical prism-canvases whose two triangles are at distance d from critical cycle-canvases with cycle size $2d + 6$. Here is how:

Proposition 5.1.1. *Let G be a plane graph $(T_1 \cup T_2)$-critical with respect to some list assignment L, where T_1 and T_2 are two triangles and P is a shortest path between them of length d. Let G' be the graph obtained by "duplicating" the $d + 1$ vertices of path P, so that the edges of the path P are duplicated and the other neighbors of the duplicated vertices are now neighbors of the vertex corresponding to the side of the path in which the neighbors were in (see Figure 5.1). Let C be the corresponding cycle of length $2d + 6$ that is newly formed with the duplicated vertices and the vertices of the triangles. Then G' is C-critical (with respect to the naturally corresponding list assignment L').*

Proof. By the Extension Lemma (1.3.15), G is $(T_1 \cup T_2 \cup P)$-critical. Now, the result follows by the Duplication Lemma (1.3.16). $\square$

Now, we use this result in the backwards direction: from critical cycle-canvases of cycle size $2d + 6$, we generate candidates for critical prism-canvases by identifying $d+1$ consecutive vertices in the precolored cycle with the opposite segment of $d + 1$ consecutive vertices in the precolored cycle (it is useful to visualize the "inverted" canvas, so that the precolored cycle is no longer the outer face but a cycle bounding an interior face). Note that from each cycle-canvas there are $d + 3$ ways to identify the vertices. After criticality testing all
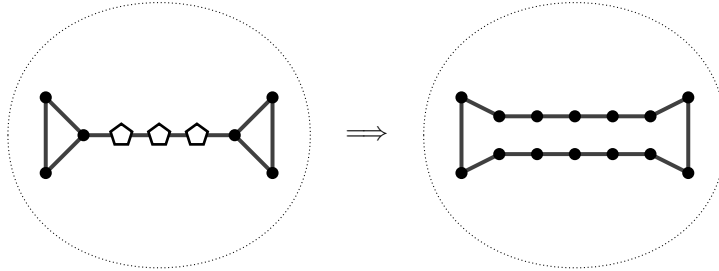
Figure 5.1: Critical cycle-canvas from critical prism-canvas.

candidates from all cycle-canvases of size $2d + 6$, we obtain all critical prism-canvases of distance d.

This can serve to get the list of all critical prism-canvases with the triangles at a certain distance, which is useful for a part of our plan and to experimentally determine the value of the right distance constant in Conjecture 2.1.17. However it is not useful, just by itself, to prove that there are no such critical prism-canvases for large enough distances.

5.2   The Forbidden 3-3 Setting

Recall Postle's approach: Postle, like Thomassen, works in a setting where the vertices of the outer face of the graphs are allowed to have lists of size 3. For proving the Two Precolored Triangles Theorem, this is useful because a shortest path between the triangles can be precolored to obtain a graph with two precolored paths of length 1 and vertices with list size at least 3 in the outer face. The problem is that there are infinitely many critical graphs in such a setting. The question is: can we find some restriction on the list sizes so that:

1. It is restrictive enough so that there are only finitely many $(P_1 \cup P_2)$-critical graphs, where P_1, P_2 are paths of length 1 in the outer face, but

2. It is not too restrictive, so it is useful and allows reductions like in the proof of Thomassen's Theorem.

The setting we propose is: require vertices in the outer face to have list size at least 3, and also additionally require that no two adjacent non-precolored vertices have list size less than 4. Note that in this setting, we cannot have the infinte amount of bellows that prevented path-canvases with path length 2 from being colorable, or the coloring harmonicas that prevented canvases with a path of length 1 and a path of length 0 from being colorable.

In fact, this setting was first introduced in [7] to prove that there are only finitely many P-critical graphs in this setting, where P is a path of length 2,

and as we will see there are also only finitely many critical graphs for longer path lengths and for two paths of length 1.

Definition 5.2.1. We say that (G, S, L) is a *3-3-forbidden canvas* if G is a connected plane graph with outer walk C, S is a subgraph of C, and L is a list assignment such that $|L(v)| \geq 5 \, \forall v \in V(G) \setminus V(C)$, $|L(v)| \geq 3 \, \forall v \in V(C) \setminus V(S)$ and for all $uv \in E(G)$ with $u, v \notin V(S)$ we have that $\max\{|L(u)|, |L(v)|\} \geq 4$. If S is a path, we say (G, S, L) is a *3-3-forbidden wedge* (or just *wedge* in this section). If $S = P_1 \cup P_2$ with P_1, P_2 paths of length at most 1, we say that (G, S, L) is a *3-3-forbidden biwedge*. We say that (G, S, L) is critical if G is S-critical with respect to L.

Since in our program we do not store the list assignments for the graphs but we do store the prescribed list sizes for each vertex (that is, whether a vertex can have a list size of 3 or is forced to have ≥ 4 colors), we will sometimes abuse terminology use the terms critical wedge and critical biwedge to refer to those graphs with prescribed list sizes but not with a fixed list assignment.

5.2.1 The Forbidden 3-3 Reduction

Before we state the reduction that we can use in this setting, we need the following result:

Proposition 5.2.2. *There are only two critical 3-3-forbidden wedges with path length 2:*

1. *The three-vertex graph consisting of $P = p_0 p_1 p_2$ with an additional edge joining p_0 and p_2.*

2. *The four vertex graph consisting of P plus one vertex with list of size three joined to all three vertices of P.*

This is proven in [7] using the reduction we are about to describe, but note that we can also prove it by generating critical wedges using the algorithm in Section 3.3 and seeing that it halts after generating only these two graphs.

Definition 5.2.3. We say that a 3-3-forbidden canvas (G, S, L) is *reducible* if the following holds:

1. It is 2-connected.

2. The outer cycle has no chords.

3. There is a precolored vertex $p_0 \in S$ so that the next four vertices v_1, v_2, v_3, v_4 in clockwise order in the outer cycle are not in S.

Let (G, S, L) be a critical reducible 3-3-forbidden canvas, and consider the list sizes of the vertices v_1, v_2, v_3, v_4 as in the definition of reducible. We are going to consider different cases according to those list sizes, and in each case we will select a subset X of those vertices and a L-coloring on some of the vertices in X.
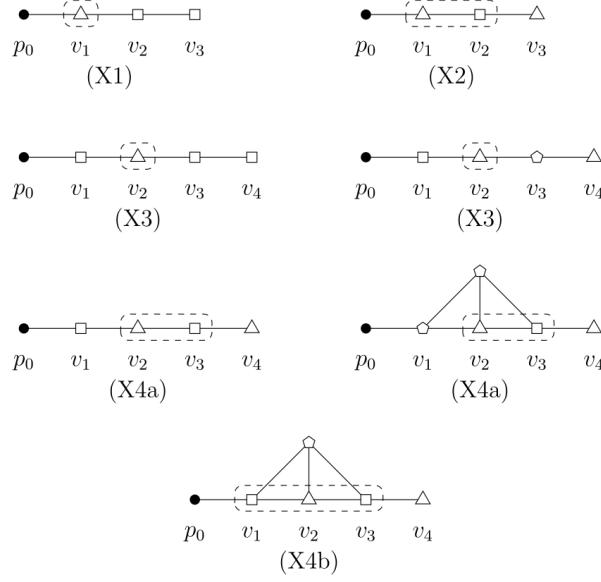
Figure 5.2: Illustration for the cases in the reduction (from [7])

X1 If $|L(v_2)| \geq 4$ and $|L(v_3)| \geq 4$, then $X = \{v_1\}$ and $\phi(v_1) \in L(v_1) \setminus L(p_0)$ is chosen arbitrarily.

X2 If $|L(v_2)| \geq 4$ and $|L(v_3)| = 3$, then $X = \{v_1, v_2\}$ and ϕ is chosen so that $\phi(v_2) \in L(v_2) \setminus L(v_3)$ and $\phi(v_1) \in L(v_1) \setminus (L(p_0) \cup \{\phi(v_2)\})$.

X3 If $|L(v_2)| = 3$, and either $|L(v_4)| \neq 3$ or $|L(v_3)| \geq 5$, then $X = \{v_2\}$ and $\phi(v_2) \in L(v_2)$ is chosen arbitrarily.

X4 If $|L(v_2)| = 3$, $|L(v_3)| = 4$ and $|L(v_4)| = 3$, then:

X4a If v_1, v_2 and v_3 do not have a common neighbor or $|L(v_1)| \geq 5$, then $X = \{v_2, v_3\}$ and ϕ is chosen so that $\phi(v_3) \in L(v_3) \setminus L(v_4)$ and $\phi(v_2) \in L(v_2) \setminus \{\phi(v_3)\}$.

X4b If v_1, v_2 and v_3 have a common neighbor and $|L(v_1)| = 4$, then $X = \{v_1, v_2, v_3\}$ and ϕ is chosen so that $\phi(v_3) \in L(v_3) \setminus L(v_4)$, $\phi(v_1) \in L(v_1) \setminus L(p_0)$ and either at least one of $\phi(v_1)$ and $\phi(v_3)$ does not belong to $L(v_2)$ or $\phi(v_1) = \phi(v_3)$. The vertex v_2 is left uncolored.

Let $G' = G \setminus X$ and let L' be the list assignment obtained from L by removing the colors of the vertices of X according to ϕ from the lists of their neighbors (if a vertex of X is not colored according to ϕ, we do not remove any colors for it). By the choice of X and ϕ, any L'-coloring of G' can be extended to an L-coloring of G, thus G' is not L'-colorable.
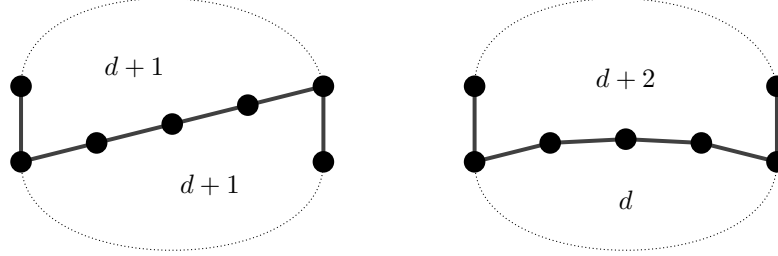
Figure 5.3: Fusing wedges along the shortest path of the biwedge.

This can imply one of these two options:

1. (G', S, L') contains a critical 3-3-forbidden canvas as a subgraph (and one which is smaller than (G, S, L)).

2. (G', S, L') is not a 3-3-forbidden canvas.

The second option can only happen if G' contains two adjacent vertices with list size 3. Let us call those vertices u, v. Because G had no chords and because Lemma 1.3.14 and Proposition 5.2.2 applied to vertices of G with two neighbors in X, we can conclude that u and v form a chord of the outer walk of G'.

5.2.2 Plan for Critical Biwedges

We will apply the above reduction to prove the following:

Conjecture 5.2.4. *Let $(G, P_1 \cup P_2, f)$ be a tuple such that G is a plane graph, P_1, P_2 are two length one paths in the outer walk of G, and f is a list size assignment as in the definition of 3-3-forbidden canvases. If there exists an f-list-assignment L such that $(G, P_1 \cup P_2, L)$ is a critical 3-3-forbidden biwedge, then $(G, P_1 \cup P_2, f) \in \mathcal{S}$, where $\mathcal{S}$ is an explicit finite set (to be determined computationally) such that all the members of $\mathcal{S}$ satisfy $d(P_1, P_2) \leq 4$.*

In order to prove this, we do the following:

We can generate all critical biwedges with distance between paths d by fusing all pairs of critical wedges with path lengths $d+1$ or path lengths d and $d+2$ as in Figure 5.3 and testing criticality. By Lemma 1.3.15 applied to the shortest path between the two paths and Lemma 1.3.14, all critical biwedges can be decomposed in this way.

We let $\mathcal{S}$ be the set of critical biwedges with distance between paths up to 4 obtained by that procedure.

To prove the conjecture, assume for a contradiction that there is a critical biwedge $(G, P_1 \cup P_2, f) \notin \mathcal{S}$, and choose such a counterexample with the smallest $|V(G)|$.

Assume G has a chord or cutvertex. Then, by Lemma 1.3.14, G can be decomposed into two smaller biwedges fused along the chord or cutvertex. We should verify computationally the following statement:

Proposition 5.2.5. *If any $B_1, B_2 \in \mathcal{S}$ are fused along one of their precolored paths to create a biwedge $(G, P_1 \cup P_2, f)$ with $d(P_1, P_2) \geq 5$, then $(G, P_1 \cup P_2, f)$ is not critical.*

So we conclude that G can not have a chord or cutvertex.

We must have $d(P_1, P_2) \geq 5$, since all critical biwedges with distance up to 4 are in $\mathcal{S}$. Now, note that our biwedge is reducible. We apply the reduction. If G contains as a subgraph a smaller critical biwedge, then that critical biwedge is in $\mathcal{S}$ and therefore has distance at most 4 between precolored paths, so G shoud have distance at most 4 between paths, contradiction.

Therefore we have that the resulting graph G' after performing the reduction has a chord uv with two vertices of list size 3. And G' is not L'-colorable, so there exists a coloring of $P_1 \cup P_2$ that does not extend to G', so G' contains a subgraph G'' which is $(P_1 \cup P_2)$-critical with respect to L'. Now, G'' is not a critical biwedge because of the uv chord, but it by Lemma 1.3.14 it can be decomposed into two critical smaller critical biwedges along the chord. Therefore, we must verify computationally the following statement:

Proposition 5.2.6. *If any $B_1, B_2 \in \mathcal{S}$ are fused along one of their precolored paths to create a triple $(G, P_1 \cup P_2, f)$ with **one chord between vertices with prescribed list sizes** 3 **but no other adjacencies between vertices with prescribed list size** 3, then either $(G, P_1 \cup P_2, f)$ is not critical **or** $d(P_1, P_2) \leq 4$.*

In order to reach a contradiction and conclude that such a counterexample must not exist.

5.2.3   Approach for Critical Triangle-Wedges

We can not use Postle's technique of precoloring a shortest path between the two triangles to conclude the result for two triangles from the result for biwedges, because the requirement that no two vertices adjacent vertices have list of size 3 is too strong for that.

Instead, what we can try is to use the reduction again. We generalize the two precolored triangles setting as follows:

Definition 5.2.7. We say that $(G, S = P \cup T, L)$ is a *3-3-forbidden triangle-wedge* if G is a connected plane graph with outer walk C, P is a path of length 1 of C, T is a triangle bounding a face of G (not necessarily incident with C) , and L is a list assignment such that $|L(v)| \geq 5 \, \forall v \in V(G) \setminus (V(C) \cup V(T))$, $|L(v)| \geq 3 \, \forall v \in V(C) \setminus V(S)$ and for all $uv \in E(G)$ with $u, v \notin V(S)$ we have that $\max\{|L(u)|, |L(v)|\} \geq 4$.

Note that the critical prism-canvases can be retrieved from the critical 3-3-forbidden triangle-wedges by selecting those whose outer face is a triangle with list size 3 in the outer face.

We can try a similar strategy as with the biwedges in order to prove that all the critical triangle-wedges belong to a critical set: construct all candidates in which the triangle is at a small distance from the outer face, and then apply the reduction to show that there are no critical graphs with a larger distance.

But the problem is, here constructing all the graphs for which the triangle is at a small distance from the outer face is much more complicated than for the biwedges. It also requires larger critical wedges as building blocks, which we are less likely to be able to be computed in reasonable time. In theory, this approach could be carried out with access to large computational resources, but it seems like something smarter is needed.

Chapter 6

Results and Further Study

In this chapter we describe the computational results we got by executing our implementation of the techniques described in the previous chapters, and briefly discuss next steps for working on the 6-list-critical graphs on the torus problem.

6.1   Computational Results

6.1.1   Generation of Cycle-Canvases

ℓ	#	ℓ	#	ℓ	#
3	1	7	18	11	131221
4	1	8	145	12	1447449
5	2	9	1260	13	16506284
6	5	10	12518	14	–

Table 6.1: Number of critical cycle-canvases by cycle size

In Table 6.1 we can see the number of chordless cycle-canvases generated by our program for sizes from 3 to 13. We see an exponential growth in the number of critical cycle-canvases, which is consistent with the linear number of vertices bound of Theorem 2.1.12. The base of the exponent appears to be approximately 10.

Note that the number corresponds to *candidates* for critical chordless cycle-canvases (also including the empty cycle-canvases, which are considered critical by our program for implementation ease), and the number of actual critical canvases could be slightly lower, especially in the higher cycle sizes. (This applies for all the tables in this chapter).

The generation of the critical cycle-canvases of cycle size 13 takes around 20 hours in our computer. Our program uses too much memory for cycle-canvases of size 14, because of the need to store critical canvases in the queue and in the

associative container used to eliminate duplicates. We see the following possible improvements to ameliorate this:

- Process the cycle-canvases in a depth-first rather than breadth-first way: that is, use a stack rather than a queue. Because the search space of critical cycle-canvases is shallow with respect to the operation of adding a tripod, this will use less memory.

- Do not store the found cycle-canvases, just print them. This brings up the problem of how to handle duplicates: one possible way is to try to define a "canonical" sequence of tripod operations to generate any possible cycle-canvas, and only generate canvases through that canonical sequence of operations. This requires analyzing the symmetries of the graphs.

- Alternatively, compress the information of the generated graphs as much as possible while still retaining the ability to detect duplicates. Applying a compression hash function to the DFS transcript is the simplest example.

Also, we can also improve the time performance of the program via parallelization, since the program does criticality testing different graphs independently, and therefore can easily be parallelized.

We did in fact implement some of the proposals above, but it was not enough to achieve the generation of all critical cycle-canvases of cycle size 14 in our computer. Nevertheless, we believe that generating cycle-canvases of cycle size 14 is feasible and can be achieved by carefully optimizing the time and memory performance of our program or by running the program in a computer with greater resources. However, the exponential growth in the number of critical cycle-canvases suggests that the practical limit might be 14 or 15. In particular, it is not feasible to generate all 10^{10} critical cycle-canvases of cycle size 16, which means that if the appropiate constant in the Two Precolored Triangles Theorem is 5, as we conjecture, we will not be able to verify it by canvas strangulation.

6.1.2 Generation of Prism-Canvases via Strangulation

d	$\#$
1	352
2	1573
3	125

Table 6.2: Number of critical prism-canvases by distance between triangles

Table 6.2 shows the number of prism-canvases obtained by canvas strangulation of cycle-canvases of cycle sizes 8, 10, 12. The stark decline with prism-canvases at distance 3 gives us hopes that the number will reach 0 by distance 5 (or even 4).

Processing the critical cycle-canvases of cycle size 12 to get the prism-canvases at distance 3 takes about 30 minutes in our computer, approximately

the same time the cycle-canvas search program takes to generate the canvases of size 12. The same optimizations, in particular parallelization, can be applied to this program to make it faster.

6.1.3 Forbidden 3-3 Setting Approaches

ℓ	#
1	1
2	3
3	22
4	245
5	3198
6	44945

Table 6.3: Number of critical 3-3-forbidden wedges by path length

d	# 2 Paths	# Path & Vertex
1	22	1
2	212	10
3	116	4
4	14	0

Table 6.4: Number of critical 3-3-forbidden biwedges by distance between precolored paths

Table 6.3 shows the number of critical 3-3 forbidden wedges by path length up to length 6 and Table 6.4 shows the number of biwedges (both those with two precolored paths of length one and those with one path of length one and one path of length zero) generated from those wedges by the distance between precolored paths up to distance 4. We computationally performed the proof steps explained in Section 5.2.2 and therefore proved Conjecture 5.2.4, which we now restate here as two theorems.

Theorem 6.1.1. *There exist only finitely many 3-3-forbidden critical biwedges.*

Theorem 6.1.2. *Let G be a plane graph with outer walk C, let P_1, P_2 be two paths of length 1 in C, and let L be a list assignment for G. If all the following conditions are satisfied:*

1. *$|L(v)| \geq 5 \,\forall v \in V(G) \setminus V(C)$.*

2. *$|L(v)| \geq 3 \,\forall v \in V(C) \setminus (V(P_1) \cup V(P_2))$*

3. *P_1, P_2 are L-colorable.*

4. *There does not exist $uv \in E(G)$ with $|L(u)| = |L(v)| = 3$.*

5. $d(P_1, P_2) \geq 5$.

then G is L-colorable.

6.2 Conclusions and Further Study

We hope that the work done in this thesis can serve as a first step in the quest for finding the set of 6-list-critical graphs on the torus. Based on the results we have gotten above and what we have studied about the problem, we outline some possible next steps to be taken:

- Implement the corresponding optimizations to the critical cycle-canvas search and canvas strangulation programs in order to obtain the list of critical cycle-canvases with cycle size 14 and the list of critical prism-canvases with triangle distance 4. We discussed the details of what those optimizations could be in the previous section.

- Try to prove the Two Precolored Triangles Theorem with the tightest bound. This is the main obstacle we have faced for our plan. Our approaches have not been successful: the 3-3-forbidden setting worked well for the biwedges but it is not conceptually clear how to extend it to the precolored triangle setting. Other approaches that we tried also had problems. Perhaps more work and more complex strategies can make these approaches successful, or perhaps new ideas are needed here.

- Implement gluing of prism-canvases along the precolored triangles to generate new candidates for critical prism-canvases with a separating triangle. This may allow us to get some critical prism-canvases with higher distances (if those exist) without having to strangulate cycle-canvases of large cycle size.

- Implement resconstruction of critical 6-list-critical graphs on the torus from critical cycle-canvases and prism-canvases. This will allow us to see if we have found any counterexample to Conjecture 2.2.2, and will at least allow us to prove a weaker result about the colorability of graphs on the torus with bounded distance between non-contractible triangles even if we do not prove the Two Precolored Triangles Theorem with a good bound.

- Think about what to do for graphs on the torus without non-contractible triangles.

Bibliography

[1]   Michael Albertson and Joan Hutchinson. "The three excluded cases of Dirac's map-color theorem". In: *Annals of the New York Academy of Sciences* 319 (Dec. 2006), pp. 7 –17. DOI: `10.1111/j.1749-6632.1979.tb32768.x`.

[2]   Noga Alon. "Combinatorial nullstellensatz". In: *Combinatorics, Probability and Computing* 8.1-2 (1999), pp. 7–29.

[3]   Noga Alon and Michael Tarsi. "Colorings and orientations of graphs". In: *Combinatorica* 12 (1992), pp. 125–134.

[4]   K. Appel and W. Haken. "Every planar map is four colorable. Part I: Discharging". In: *Illinois Journal of Mathematics* 21.3 (1977), pp. 429 – 490. DOI: `10.1215/ijm/1256049011`.

[5]   K. Appel, W. Haken, and J. Koch. "Every planar map is four colorable. Part II: Reducibility". In: *Illinois Journal of Mathematics* 21.3 (1977), pp. 491 –567. DOI: `10.1215/ijm/1256049012`.

[6]   R. Diestel. *Graph theory*. Springer-Verlag, 2000.

[7]   Zdenek Dvorak, Bernard Lidicky, and Bojan Mohar. "5-choosability of graphs with crossings far apart". In: *Journal of Combinatorial Theory, Series B* 123 (2017), pp. 54–96. DOI: `10.1016/j.jctb.2016.11.004`.

[8]   Zdeněk Dvořák. "An efficient implementation and a strengthening of Alon-Tarsi list coloring method". In: (2023). arXiv: `2301.06571 [cs.DM]`.

[9]   David Eppstein. "Subgraph Isomorphism in Planar Graphs and Related Problems". In: *Journal of Graph Algorithms and Applications* 3.3 (1999), pp. 1–27. DOI: `10.7155/jgaa.00014`.

[10]   Paul Erdos, Arthur L Rubin, and Herbert Taylor. "Choosability in graphs". In: *Proc. West Coast Conf. on Combinatorics, Graph Theory and Computing, Congressus Numerantium*. Vol. 26. 1979, pp. 125–157.

[11]   Steve Fisk. "The nonexistence of colorings". In: *Journal of Combinatorial Theory, Series B* 24.2 (1978), pp. 247–248. ISSN: 0095-8956. DOI: `https://doi.org/10.1016/0095-8956(78)90028-X`.

[12]   T. Gallai. "Kritische Graphen I, II". In: *Publ. Math. Inst. Hungar. Acad. Sci.* 8 (1963), 165–192, 373–395.

[13] M.R. Garey, D.S. Johnson, and L. Stockmeyer. "Some simplified NP-complete graph problems". In: *Theoretical Computer Science* 1.3 (1976), pp. 237–267. ISSN: 0304-3975. DOI: `https://doi.org/10.1016/0304-3975(76)90059-1`.

[14] P. J. Heawood. "Map colour theorem". In: *Quarterly Journal of Mathematics* 24 (1890), pp. 332–338.

[15] J. P. Hutchinson. "On list-coloring extendable outerplanar graphs". In: *ARS MATHEMATICA CONTEMPORANEA* 5.1 (2012), pp. 175–188.

[16] Daniel Král' and Riste Skrekovski. "The last excluded case of Dirac's map-color theorem for choosability". In: *Journal of Graph Theory* 51.4 (2006), pp. 319–354.

[17] B. Mohar and C. Thomassen. *Graphs on Surfaces*. Johns Hopkins University Press, 2001. ISBN: 9780801866890.

[18] Luke Postle. "5-list-coloring graphs on surfaces". PhD thesis. Georgia Institute of Technology, 2012.

[19] Luke Postle and Robin Thomas. "Five-list-coloring graphs on surfaces I. Two lists of size two in planar graphs". In: *Journal of Combinatorial Theory, Series B* 111 (2015), pp. 234–241. ISSN: 0095-8956. DOI: `https://doi.org/10.1016/j.jctb.2014.08.003`.

[20] Luke Postle and Robin Thomas. "Five-list-coloring graphs on surfaces II. A linear bound for critical graphs in a disk". In: *Journal of Combinatorial Theory, Series B* 119 (2016), pp. 42–65. ISSN: 0095-8956. DOI: `https://doi.org/10.1016/j.jctb.2015.12.005`.

[21] Luke Postle and Robin Thomas. "Five-list-coloring graphs on surfaces III. One list of size one and one list of size two". In: *Journal of Combinatorial Theory, Series B* 128 (2018), pp. 1–16. ISSN: 0095-8956. DOI: `https://doi.org/10.1016/j.jctb.2017.06.004`.

[22] Gerhard Ringel and J. W. T. Youngs. "Solution of The Heawood Map-Coloring Problem". In: *Proceedings of the National Academy of Sciences* 60.2 (1968), pp. 438–445. DOI: `10.1073/pnas.60.2.438`.

[23] Michael Stiebitz, Zsolt Tuza, and Margit Voigt. "On list critical graphs". In: *Discrete Mathematics* 309.15 (2009). Cycles and Colourings, pp. 4931–4941. ISSN: 0012-365X. DOI: `https://doi.org/10.1016/j.disc.2008.05.021`.

[24] C. Thomassen. "Every Planar Graph Is 5-Choosable". In: *Journal of Combinatorial Theory, Series B* 62.1 (1994), pp. 180–181. ISSN: 0095-8956. DOI: `https://doi.org/10.1006/jctb.1994.1062`.

[25] C. Thomassen. "Five-Coloring Graphs on the Torus". In: *Journal of Combinatorial Theory, Series B* 62.1 (1994), pp. 11–33. ISSN: 0095-8956. DOI: `https://doi.org/10.1006/jctb.1994.1052`.

[26]   Carsten Thomassen. "Color-Critical Graphs on a Fixed Surface". In: *Journal of Combinatorial Theory, Series B* 70.1 (1997), pp. 67–100. ISSN: 0095-8956. DOI: https://doi.org/10.1006/jctb.1996.1722.

[27]   Carsten Thomassen. "Exponentially many 5-list-colorings of planar graphs". In: *Journal of Combinatorial Theory, Series B* 97.4 (2007), pp. 571–583. ISSN: 0095-8956. DOI: https://doi.org/10.1016/j.jctb.2006.09.002.

[28]   Carsten Thomassen. "Tilings of the Torus and the Klein Bottle and Vertex-Transitive Graphs on a Fixed Surface". In: *Transactions of the American Mathematical Society* 323.2 (1991), pp. 605–635. ISSN: 00029947.

[29]   Vadim G Vizing. "Vertex colorings with given colors". In: *Diskret. Analiz* 29 (1976), pp. 3–10.

[30]   Margit Voigt. "List colourings of planar graphs". In: *Discrete Mathematics* 120.1-3 (1993), pp. 215–219.