TREE-BASED DECOMPOSITIONS OF GRAPHS ON SURFACES AND APPLICATIONS TO THE TRAVELING SALESMAN PROBLEM

A Thesis
Presented to
The Academic Faculty

by

Torsten Inkmann

In Partial Fulfillment
of the Requirements for the Degree
Doctor of Philosophy in
Algorithms, Combinatorics, and Optimization

School of Mathematics
Georgia Institute of Technology
April 2008

TREE-BASED DECOMPOSITIONS OF GRAPHS ON SURFACES AND APPLICATIONS TO THE TRAVELING SALESMAN PROBLEM

Approved by:

Professor William J. Cook, Advisor
School of Industrial and Systems Engineering
Georgia Institute of Technology

Professor Robin Thomas, Advisor
School of Mathematics
Georgia Institute of Technology

Professor Zdeněk Dvořák
School of Mathematics
Georgia Institute of Technology

Professor Robert G. Parker
School of Industrial and Systems Engineering
Georgia Institute of Technology

Professor Xingxing Yu
School of Mathematics
Georgia Institute of Technology

Date Approved: December 4th, 2007

ACKNOWLEDGEMENTS

I would like to thank both of my advisors, Professors William Cook and Robin Thomas, for their support and guidance. Apart from stating the obvious, namely that having two leading researchers in their respective fields to work with and learn from has been invaluable for my academic work, I would like to thank them in particular for everything else besides thinking about math problems with me. While it may seem self-evident for some to expect dependability, availability, fairness and most of all patience from an advisor, this is not always the case and I would like to express my gratitude to Bill and Robin for excelling also in these disciplines.

On the academic side, I would furthermore like to thank Professors Zdeněk Dvořák, Gary Parker and Xingxing Yu for serving on my thesis committee, and Professors Ilya Hicks and Hisao Tamaki for discussions on my research.

The past five years at Georgia Tech have been a truly unique experience, and I would like to thank everyone who has been part of that in one way or another. In particular I would like to thank all my friends that I have met in Atlanta and elsewhere, and all my friends from Germany who managed to stay in my life despite my extended period of absence.

Doing research in Mathematics can be challenging at times (both Mathematicians and others will easily recognize this as a slight understatement), and there has been no other single more important factor than my family in helping me overcome these challenges. The unconditional support of my sister and my parents is the backbone of my life and this thesis.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

SUMMARY

The tree-width and branch-width of a graph are two well-studied examples of parameters that measure how well a given graph can be decomposed into a tree structure. In this thesis we give several results and applications concerning these concepts, in particular if the graph is embedded on a surface.

In the first part of this thesis we develop a geometric description of tangles in graphs embedded on a fixed surface (tangles are the obstructions for low branch-width), generalizing a result of Robertson and Seymour. We use this result to establish a relationship between the branch-width of an embedded graph and the carving-width of an associated graph, generalizing a result for the plane of Seymour and Thomas. We also discuss how these results relate to the polynomial-time algorithm to determine the branch-width of planar graphs of Seymour and Thomas, and explain why their method does not generalize to surfaces other than the sphere.

We also prove a result concerning the class $\mathcal{C}_{2k}$ of minor-minimal graphs of branch-width $2k$ in the plane, for an integer $k \geq 2$. We show that applying a certain construction to a class of graphs in the projective plane yields a subclass of $\mathcal{C}_{2k}$, but also show that not all members of $\mathcal{C}_{2k}$ arise in this way if $k \geq 3$.

The last part of the thesis is concerned with applications of graphs of bounded tree-width to the Traveling Salesman Problem (TSP). We first show how one can solve the separation problem for comb inequalities (with an arbitrary number of teeth) in linear time if the tree-width is bounded. In the second part, we modify an algorithm of Letchford et al. using tree-decompositions to obtain a practical method for separating a different class of TSP inequalities, called simple DP constraints, and study their effectiveness for solving TSP instances.

CHAPTER I

INTRODUCTION

1.1 Tree-based decomposition methods and graphs embedded on surfaces

In their seminal series of papers resulting in the proof of the Graph Minor Theorem [64], Robertson and Seymour introduced several new ways of decomposing graphs into a tree-like structure. Two of these techniques are tree-decompositions and branch-decompositions, together with the corresponding parameters tree-width and branch-width, measuring how well a graph can be decomposed (for definitions, see Chapter 2.3). These closely related decompositions not only play a major role throughout the proof of the Graph Minor Theorem, but also have applications elsewhere and consequently became the subject of intense research in their own right.

Apart from their relevance in graph minor theory, tree- and branch-decompositions also are interesting from an algorithmic point of view: A large class of NP-hard optimization problems on graphs can be solved in polynomial time (or often linear time) provided the width of the input graph is bounded by a constant (for references of results of this type, see 2.3). Although these algorithms are very appealing from a theoretical point of view, especially since they are usually based on a conceptually simple dynamic programming approach, the large constants typically involved present challenges for the practicability of these methods.

In the proof of the Graph Minor Theorem, Robertson and Seymour also make heavy use of the concept of embedding graphs on surfaces. Although tree- and branch-decompositions are defined for arbitrary graphs, it turns out that there are close connections between graphs of bounded tree- or branch-width and graphs embedded on surfaces:

For example, Robertson and Seymour show that for any fixed planar graph H, a class of graphs $\mathcal{G}$ has bounded tree-width if and only if none of the graphs in $\mathcal{G}$ contains H as a minor [60].

In [63], Robertson and Seymour establish a characterization of the representativity (for a definition, see 2.2) of a graph embedded in a surface other than the sphere, in terms of 'respectful tangles'. A tangle (for a definition see 2.3.2) is an obstruction to low branch-width, i.e. a graph has a tangle of order k (for some integer $k \geq 2$) if and only if its branch-width is at least k. In particular, their result implies that if a graph G has an embedding of representativity k on some surface Σ, then its branch-width is at least k.

The proof of the above-mentioned result is based on a geometric description for respectful tangles, called slopes. In the case where Σ is the sphere, all tangles are trivially respectful, and the results about slopes from [63] form the basis of a polynomial-time algorithm by Seymour and Thomas [67] to compute the branch-width of planar graphs.

1.1.1 Contributions of this thesis

In Chapters 3 and 4 of this thesis we establish some further results concerning tree-based decompositions of graphs embedded on a surface.

In Chapter 3, we present a framework for a natural generalization of some of the results from Graph Minors XI [63]. Our first main result (Theorem 3.4.2) uses this framework to show that arbitrary tangles can be described in a geometric way, thus generalizing the geometric description for respectful tangles from [63]. We also show that the second main result of [63] no longer holds in this generalized setting (Theorem 3.5.4), and provide an infinite family of counterexamples.

For a planar graph G, the branch-width of G is equal to half of the carving-width of the medial graph (for definitions, see 2.3), as was established in [67]. We give examples showing that this equality no longer holds on surfaces other than the sphere, and use the above geometric description of arbitrary tangles to prove that the two quantities are within a factor of 2 in general, up to an additive error term depending on the surface (Theorem 3.6.1).

In Chapter 4, we study minor-minimal planar graphs of fixed branch-width. We show that if H is a minor-minimal graph embedded in the projective plane with face-width k, and G is a planar double cover of H, then G is minor-minimal of branch-width $2k$ (Theorem 4.1.1). We also disprove the tempting conjecture that all minor-minimal planar graphs of even branch-width arise in this way, and construct minor-minimal planar graphs of branch-width $2k + 1$, for each integer $k \geq 2$, which are not double covers of any graphs.

Chapter 5 contains some results using tree-decompositions in the context of the Traveling Salesman Problem, which we introduce in the next section.

1.2 The Traveling Salesman Problem

The (symmetric) traveling salesman problem, or TSP, is one of the most studied problems in combinatorial optimization: Given a complete (undirected) graph G_n on n vertices, with non-negative edge-costs c_e for every edge, the objective is to find a cycle of minimum total cost, containing all vertices of G_n. In the context of the TSP, the vertices of G_n are referred to as 'cities', and a cycle of length n is called a 'tour'.

The significance of the TSP stems from the fact that it has been used as a vehicle for developing and testing a large number of techniques which have been successful both in the theory and practice of combinatorial optimization (for a comprehensive treatment of the subject, see for example [3]).

One particular technique arises from the close connection to linear programming.

If we introduce a 0-1 variable x_e for each edge $e \in E(G_n)$, then the TSP can easily be modelled as an integer programming problem. Since it is NP-hard to solve this problem, one can first solve the linear programming problem (LP) obtained from relaxing the constraints $x_e \in \{0,1\}$ to $0 \leq x_e \leq 1$. Suppose we find an optimal solution $\hat{x}$ to this LP. If $\hat{x}$ is integer, it corresponds to an optimal tour and we have solved the TSP. In the seminal paper by Dantzig, Fulkerson and Johnson in 1954 [27], the authors propose the following approach, called the *cutting plane method*, for the case that $\hat{x}$ is fractional: Find a linear inequality, called a *cutting plane*, which is satisfied by any integer vector corresponding to a tour, but is violated by the fractional solution $\hat{x}$. Then add this inequality to the LP formulation, resolve the LP and repeat the process.

It is usually desirable to restrict the type of the inequality added to the LP to belonging to a pre-specified class of inequalities $\mathcal{I}$ valid for all tours. Hence one arrives at what is called the *separation* problem: Given a (fractional) solution $\hat{x}$, find an inequality in $\mathcal{I}$ which is violated by $\hat{x}$ (or decide that none exists).

Many of the interesting classes of inequalities can be viewed as arising from (weighted) hypergraphs defined on the support graph $\hat{G}$, i.e. the subgraph of G consisting only of edges with $\hat{x}_e > 0$. In particular, two of the most important classes of inequalities, the *subtour constraints* and the *comb inequalities* arise in this way (for definitions, see Section 2.4). While the separation problem for the subtour constraints can be solved efficiently both in theory and practice ([42], [56]), it is a major open problem whether there exists a polynomial-time algorithm for the general comb separation problem.

For certain special cases efficient methods for the comb separation problem exist. Padberg and Rao [57] showed how to separate 'blossom inequalities', i.e. combs where each tooth consists of exactly two vertices. Carr [18] showed how to solve the comb separation problem restricted to combs with a fixed number of teeth in polynomial time, but the method is impractical even for combs with only three teeth.

Another class of inequalities that has received a considerable attention recently are the *domino-parity inequalities* (or DP inequalities) defined by Letchford in [50]. The DP inequalities are a generalization of comb inequalities, and have proven to be very helpful in solving some of the largest TSP instances (see [21]). In [50], it is shown how to separate DP inequalities in polynomial time in the case where $\hat{G}$ is planar, and in [34], the authors give an algorithm to separate so-called *simple DP-inequalities*. However, although the latter algorithm runs in polynomial time, the runtime bound of $\mathcal{O}(n^2m^2 \log n^2/m)$ makes it unclear whether or not this method can be used for larger TSP instances.

1.2.1 Contributions of this thesis

In Chapter 5 of this thesis, we describe some separation routines for the case that the support graph $\hat{G}$ has bounded tree-width.

We first show how the comb separation problem (for an arbitrary number of teeth) can be solved in linear time if $\hat{G}$ has bounded tree-width (Theorem 5.1.2), using some facts about the structure of violated combs. Although the assumption of bounded tree-width is natural for the support graph $\hat{G}$, it appears that the method is primarily of theoretical interest due to the high constants involved in the algorithm.

In the second part we discuss how to modify and extend the above-mentioned algorithm of [34] using tree-decompositions in order to generate all violated simple DP inequalities (5.2.1), and provide some computational results on the effectiveness of simple DP inequalities for solving the TSP. The method also demonstrates that tree- or branch-decomposition based methods may present valid alternatives in practice not only for NP-hard problems, but also for problems which are solvable by an algorithm whose runtime is bounded by a polynomial of large degree.

CHAPTER II

PRELIMINARIES

In this chapter we review the basic concepts and results relevant for this thesis.

2.1 General graph theory

A *graph* $G = (V, E)$ consists of a finite set $V = V(G)$ and a finite collection $E = E(G)$ of unordered pairs of (not necessarily distinct) elements of V. The elements of V are called *vertices*, and the elements of E are called *edges*. If $e = \{u, v\}$ is an edge, then the vertices u, v are called the ends of e, and we use the notation $e = uv$. In that case, we say that u and v are adjacent, that e is incident with its ends u, v and we use the notation $u \sim v$ and $v \sim e$. An edge with identical ends is called a *loop*, and two edges $e, e' \in E(G)$ are called *parallel* if they have the same ends. A *simple graph* is a graph with no loops or parallel edges. A graph that is not simple is sometimes called a *multigraph*.

For a subset X of a fixed ground-set Y, we use X^c to denote the set $Y \setminus X$, and $\mathbb{1}_X(\cdot)$ to denote its indicator function, i.e. $\mathbb{1}_X(x) = 1$ if $x \in X$ and 0 otherwise. If $X, X' \subseteq V(G)$ are two sets of vertices, we use the notation $E(X, X')$ to denote the set of X-X' edges, i.e. the edges with one end in X and the other in X'. A set $F \subseteq E(G)$ is a *cut* if $F = E(X, X^c)$ for some $\emptyset \neq X \subsetneq V(G)$, and we write $\delta(X) = E(X, X^c)$, and $\delta(v)$ if X consists of a single vertex v. A *bond* is a cut $\delta(X)$ so that X, X^c are connected in G. It is an easy fact that in a connected graph, bonds are precisely the minimal cuts.

The *degree* of a vertex v is given by $|\delta(v)|$, plus twice the number of loops at v (if any). We denote the degree of $v \in V(G)$ by $d_G(v)$ or simply $d(v)$. The set of *neighbors* of v is defined as $N(v) = \{w \in V(G) \setminus \{v\} \mid v \sim w\}$. For a set of edges

$F \subseteq E(G)$, we let $\partial(F)$ denote the set of all vertices of G which are incident with an edge in F and an edge in F^c.

For $X \subseteq V(G)$, we write $G[X]$ for the subgraph of G *induced* by X, i.e. the subgraph of G with vertex-set X and all edges of G with both ends in X. Similarly for $F \subseteq E(G)$, we denote by $G[F]$ the subgraph *induced* by F, i.e. the subgraph of G consisting of the edges of F and their ends. For graphs G, G', the graph $G \cup G'$ has vertices $V(G) \cup V(G')$ and edge-set $E(G) \cup E(G')$, and the graph $G \cap G'$ is defined similarly.

A *separation* in a graph G consists of two subgraphs $A, B \subseteq G$ such that $E(A \cap B) = \emptyset$ and $A \cup B = G$. Hence we can think of a separation as a partition of $E(G)$ into two parts. The *order* of the separation (A, B) is $|V(A \cap B)|$.

A graph is called *eulerian* if all vertices have even degree. It is well-known that a graph is eulerian if and only if it can be written as a union of edge-disjoint cycles.

For further basic definitions such as connected graphs, walks, cycles, (internally disjoint) paths, trees, bipartite graphs, contraction of edges, minors and other standard concepts we refer the reader to the book by Diestel [30].

2.2 Graphs on surfaces

We will only mention the most important concepts related to graphs embedded on surfaces. For more details, we refer the reader to the book by Mohar and Thomassen [53], and to the introduction of [63].

A *surface* is a connected, compact Haussdorff space which is locally homeomorphic to an open disc in $\mathbb{R}^2$. Throughout this thesis, Σ will denote a surface. For an integer $g \geq 0$ we denote by $\mathbb{S}_g$ the orientable surface of genus g (i.e. with g handles), and for an integer $k \geq 1$ we denote by $\mathbb{N}_k$ the non-orientable surface of genus k (i.e. with k crosscaps). A proof of the following well-known theorem can be found in [53].

Theorem 2.2.1 (Classification of surfaces, theorem 3.1.3. in [53]). *Every* *surface is homeomorphic to precisely one of the surfaces $\mathbb{S}_g$ for $g \geq 0$, or $\mathbb{N}_k$ for $k \geq 1$.*

In particular, the surface $\mathbb{S}_0$ is the sphere, $\mathbb{S}_1$ is the torus and $\mathbb{N}_1$ is the projective plane.

A *curve* in Σ is a continuous function $\phi : [0,1] \to \Sigma$. We usually do not distinguish between the function ϕ and its image $\{\phi(x)|0 \leq x \leq 1\}$ in Σ. A curve ϕ is *simple* if ϕ is injective, it is *closed* if $\phi(0) = \phi(1)$ and *constant* if $\phi(x) = \phi(y)$ for all $x, y \in [0,1]$. A *simple closed curve* ϕ is a closed curve where ϕ is injective on $(0,1)$. Two closed curves ϕ_0, ϕ_1 are *(freely) homotopic* (in $Z \subseteq \Sigma$) if there is a continuous function $\psi : [0,1] \times [0,1] \to Z$ such that:

$$\begin{aligned}
\psi(x,0) &= \phi_0(x) \text{ for } x \in [0,1], \\
\psi(x,1) &= \phi_1(x) \text{ for } x \in [0,1]. \\
\psi(0,t) &= \psi(1,t) \text{ for } t \in [0,1].
\end{aligned}$$

A closed curve is *null-homotopic* or *contractible* (in Z) if it is homotopic to a constant curve in Z, and *non-contractible* otherwise. A *closed (open) disc* in Σ is a closed (open) subset $D \subseteq \Sigma$ that is homeomorphic to the closed (open) unit disc in $\mathbb{R}^2$. It is a well known fact that a simple closed curve is contractible if and only if it bounds a disc ([32], theorem (1.7)).

We denote the closure of a set $Z \subseteq \Sigma$ by $\bar{Z}$ or $cl(Z)$, and the boundary of a closed set Z by $\partial(Z)$.

A set $Z \subseteq \Sigma$ is *simply-connected* if Z is (arc-wise) connected and every simple closed curve is null-homotopic in Z.

We say a graph $G = (V, E)$ is *embedded* in Σ if the elements of V are distinct points in Σ, and every edge $e = uv$ of G corresponds to a simple curve with ends u, v (or simple closed if $u = v$), such that any two such curves are disjoint except possibly their ends. (For a more detailed definition, see [53]). A collection of such points and curves corresponding to a graph $G = (V, E)$ is called an *embedding* of G in Σ, and we write $G \hookrightarrow \Sigma$ to mean that G is embedded in Σ. We usually do not distinguish between a graph and its embedding. In particular, for a graph $G = (V, E)$ embeddable in Σ, we use $e \in E$ both for a pair of elements of V as well as for the open set given by the curve corresponding to e, without its endpoints. If we want to make the distinction explicit, we use the notation $u(e)$ to denote the open set in Σ corresponding to e, and $U(G)$ to denote the union of the points and curves corresponding to G in Σ.

A *face* or *region* of G in Σ is an arcwise connected component of $\Sigma \setminus U(G)$. The set of regions of a Σ-embedded graph G will be denoted by $R(G)$. It is easy to see that every edge is incident with exactly two (possibly equal) regions.

If $G = (V, E)$ is a connected graph embedded in Σ, its *dual* is the (multi-)graph G^* embedded in Σ where $V(G^*) = R(G)$, and the edges of G^* correspond to those of G in the usual way, i.e. if $r_1, r_2 \in R(G)$ are the two regions incident with an edge $e \in E$, then there is an edge $e^* \in E(G^*)$ with ends r_1, r_2. For a vertex, edge or region x in G, we use x^* to denote the corresponding region, edge or vertex in the dual G^*, respectively. We will assume that $u(e) \cap u(e^*)$ consists of a single point, called the *midpoint* and denoted by $x(e)$ or $x(e^*)$, and we require that the point r^* in the dual embedding corresponding to the region $r \in R(G)$ to be contained in the (open) set r. Hence G and its dual embedding G^* intersect only in the points $x(e)$, for $e \in E(G)$.

For a set of regions $Z \subseteq R(G)$, we denote by $\partial(Z)$ the subgraph of G induced by the edges of G which are incident with a region in Z and a region not in Z.

A graph is *2-cell embedded* if every region is an open disc. In a 2-cell embedded graph, every face is bounded by a (single) closed walk. The following well-known theorem can be found on page 85 of [53]:

Theorem 2.2.2 (Euler's formula). *Let G be a connected (multi-)graph which is*

2-cell embedded in a surface Σ. If G has n vertices, m edges and f faces in Σ, then

$$n - m + f = \chi(\Sigma), \tag{2.2.1}$$

where $\chi(\Sigma) = 2 - 2g$ if $\Sigma = \mathbb{S}_g$, and $\chi(\Sigma) = 1 - k$ if $\Sigma = \mathbb{N}_k$.

A cycle C in an embedded graph G is *non-contractible* if the simple closed curve $U(C)$ is non-contractible. For formal definitions of two-sided and one-sided cycles, as well as cutting along a cycle we refer the reader to Chapter 4 of [53].

Suppose P, P' are two paths with ends $u, v \in V(G)$, which are internally disjoint (i.e. $P \cap P' = \{u, v\}$). Then P and P' are *homotopic* if the cycle $P \cup P'$ is contractible, and we write $P \sim P'$ in this case.

We frequently make use of the following simple fact, called the '3-path condition' (for a proof see [69] or [53], proposition 4.3.1):

Proposition 2.2.3. *Let G be embedded in Σ. Let $v, v' \in V(G)$, let P_1, P_2, P_3 be internally disjoint v-v' paths, and let $C_{i,j} = P_i \cup P_j$ for $1 \leq i < j \leq 3$. Then if two of the cycles $C_{i,j}$ are contractible, so is the third.*

Let $G \hookrightarrow \Sigma$ where $\Sigma \neq \mathbb{S}_0$. The minimum number of points that any non-contractible closed curve intersects $U(G)$ in is called the *face-width* or *representativity* of the embedded graph G, and is denoted by $fw(G)$. (If $G \hookrightarrow \mathbb{S}_0$, $fw(G)$ is defined to be ∞). The notion of face-width was introduced in [61].

In particular we have $fw(G) \geq 1$ if and only if G is 2-cell embedded. A non-contractible simple closed curve intersecting $U(G)$ in exactly $fw(G)$ points is called a *noose*. It is easy to see that there is a noose which intersects G only in vertices.

Let G be a 2-cell embedded graph in Σ. A very useful auxiliary graph when dealing with face-width is the *radial graph* (or sometimes called the *vertex-face multigraph*), denoted by $\mathcal{R}_G$. Its vertex-set $V(\mathcal{R}_G)$ is the union of $R(G)$ and $V(G)$, and for every region $r \in R(G)$, we insert (possibly parallel) edges $v_i r$ for $i = 1, ..., k$ if

$W = v_1 e_1 v_2 ... v_k e_k v_1$ is the facial walk bounding r. Note that by definition, the radial graph of G and its dual G^* are the same, and all three can be embedded in Σ.

The dual of the radial graph $\mathcal{R}_G$ is called the *medial graph* of G, denoted by $\mathcal{M}_G$. Hence the vertices of $\mathcal{M}_G$ correspond to edges of G, and regions of $\mathcal{M}_G$ correspond to vertices or regions of G. Again we have that the medial graph of G and G^* are the same by definition.

The *edge-width* of G is defined to be the length of a shortest non-contractible cycle in G, and is denoted by $ew(G)$. The face-width of an embedding can be expressed in terms of edge-width of the radial graph (see e.g. 5.5.4 in [53]):

Proposition 2.2.4. *Let G be a 2-cell embedded graph in Σ and a $\mathcal{R}$ be its radial graph. Then*

$$2 \cdot fw(G) = ew(\mathcal{R}). \tag{2.2.2}$$

If G has high face-width, then every vertex of v has a large 'planar neighborhood' [61], and for this reason many properties from planar graphs carry over to embeddings with high face-width. As an example, we state the following useful fact (see e.g. 5.5.11 in [53]):

Proposition 2.2.5. *Let $G \hookrightarrow \Sigma$ and its dual G^* be 2-cell embedded, and let $\mathcal{R}$ denote their radial graph. Then the following statements are equivalent:*

(1) All facial walks of G are cycles.

(2) All facial walks of G^* are cycles.

(3) $fw(G) \geq 2$ and G is 2-connected.

(4) $fw(G^*) \geq 2$ and G^* is 2-connected.

(5) $\mathcal{R}$ has no parallel edges.

2.3 Tree-based decompositions

In this section we review some definitions and results related to tree-based decompositions such as tree-decompositions, branch-decompositions and carvings. Tree- and branch-decompositions were introduced in the graph minors series by Robertson and Seymour, and carvings were defined by Seymour and Thomas in [67].

2.3.1 Branch-width

The notion of a branch-decomposition was first defined in [62] (in fact their definition is for hypergraphs, but we only state it for graphs).

Definition 2.3.1. A *branch-decomposition* of a graph G is pair (T, η), where T is a tree whose internal vertices have degree exactly three, and η is a bijection from $\mathcal{L}$, the set of leaves of T, to $E(G)$.

For an edge $e \in E(T)$, let T_1, T_2 be the two components of $T \setminus e$. For $i = 1, 2$, let $E_i = \eta(\mathcal{L} \cap V(T_i)) \subseteq E(G)$ and let τ_e be the order of the separation $(G[E_1], G[E_2])$ in G (separations were defined in Section 2.1). The *width* of a branch-decomposition is defined as $max_{e \in E(T)} \tau_e$. The *branch-width* of G, denoted by $bw(G)$, is defined to be the minimum width over all branch-decompositions of G. A branch-decomposition of G is *optimal* if it has width $bw(G)$.

In [62], Robertson and Seymour derive a duality theorem for branch-width using the notion of a *tangle*.

Definition 2.3.2. Let $k \geq 1$ be an integer. A *tangle of order k* in a graph G is a family $\mathcal{T}$ of separations of order $< k$ satisfying the following axioms:

(TD1) For every separation (A, B) of order $< k$, $\mathcal{T}$ contains exactly one of (A, B) and (B, A).

(TD2) If $(A_i, B_i) \in \mathcal{T}$ for $i = 1, 2, 3$, then $A_1 \cup A_2 \cup A_3 \neq G$.

(TD3) If $(A, B) \in \mathcal{T}$ then $V(A) \neq V(G)$.

Theorem 2.3.3 ([62], page 166). *Let $k \geq 2$ be an integer. Then $bw(G) \leq k$ if and only if G has no tangle of order $k+1$.*

It is NP-hard to compute $bw(G)$ in general ([67]), however Seymour and Thomas showed how to compute $bw(G)$ in polynomial-time for planar graphs by an algorithm called the 'ratcatcher method' ([67]). Based on these ideas, Tamaki and Gu [38] improved the runtime of finding an optimal branch-decomposition of a planar graph to $\mathcal{O}(n^3)$, and Hicks discussed how to implement the method in practice ([41], [40]).

2.3.2 Tree-width

The notions of tree-decompositions and tree-width were introduced in [59].

Definition 2.3.4. A *tree decomposition* $\mathcal{T}$ of a graph $G = (V, E)$ is a pair $(T, (X_t)_{t \in V(T)})$, where T is a tree and the sets X_t (called *bags*) associated to the vertices of T are subsets of $V(G)$ satisfying the following axioms:

(T1) For every $v \in V(G)$ there is a $t \in V(T)$ so that $v \in X_t$.

(T2) For every edge $uv \in E(G)$ there is a $t \in V(T)$ so that $\{u, v\} \subseteq X_t$.

(T3) For every $v \in V(G)$, the nodes t such that $v \in X_t$ induce a subtree T_v of T.

The *width* of $\mathcal{T}$ is defined to be $max_{t \in V(T)} |X_t| - 1$. The *tree-width* of G, denoted by $tw(G)$, is the minimum width over all tree-decompositions of G. A tree-decomposition of G is called *optimal* if it has width $tw(G)$.

An (infinite) class of graphs $\mathcal{G}$ is said to have *bounded tree-width [branch-width]* if there is some integer k so that $tw(G) \leq k$ $[bw(G) \leq k]$ for all $G \in \mathcal{G}$. The following result shows that a class of graphs has bounded branch-width if and only if it has bounded tree-width. A proof can be found in [62].

Lemma 2.3.5. *For any graph G, we have*

$$bw(G) \le tw(G) + 1 \le \frac{3}{2} bw(G). \tag{2.3.1}$$

While it is NP-hard to compute $tw(G)$ (or $bw(G)$) in general, Bodlaender [13] has shown that $tw(G) \le k$ can be decided in linear time if k is fixed (i.e. not part of the input). It is an open problem in the area of tree-decompositions whether $tw(G)$ can be computed exactly for planar graphs.

Graphs of bounded tree-width not only play a major role in the proof of the graph minor theorem, but are also highly interesting in terms of algorithms. In particular, a great variety of problems which are NP-hard in general can be solved in polynomial or often even linear time if the input graph is restricted to have bounded tree-width for some fixed integer k. For example every graph problem that can be expressed in terms of certain types of logical formulae can be solved in polynomial time ([25], [26], [23], [24], [4], [16]). For a survey on the algorithmic importance of tree-width, see for example [11].

The standard approach to exploit bounded tree-width is to use dynamic programming on the tree-decomposition. For both designing and implementing such algorithms, it is often helpful to assume that the tree-decomposition at hand is of a particularly simple form, which we introduce next (see e.g. [14] for a similar definition). If T is a rooted tree and $v, w \in V(T)$, then w is a *descendent* of v if $w \neq v$ and w belongs to the subtree rooted at v, and w is a *child* of v if w is a descendent of v which is adjacent to v.

Definition 2.3.6. A tree-decomposition $(T, (X_t)_{t \in V(T)})$ of a graph G is called *nice* if T is a rooted tree in which every node $t \in V(T)$ has at most 2 children, and is of one of the following types:

(1) A LEAF node t is a leaf of T.

(2) An INTRO node t has a unique child node t' and satisfies $X_{t'} \subsetneq X_t$.

(3) A FORGET node t has a unique child node t' and satisfies $X_t = X_{t'} \setminus \{v\}$ for some $v \in X_{t'}$.

(4) A JOIN node t has two children t_1, t_2 and satisfies $X_t = X_{t_1} = X_{t_2}$.

It is not hard to see how to convert any tree-decomposition into a nice tree-decomposition of the same width in linear time, so that the number of nodes in the new tree is at most $\mathcal{O}(|V(G)|)$ - a detailed description (for a slightly stronger version of being 'nice') can be found e.g. in [65].

2.3.3 Carving-width

The notions of a carving and carving-width were introduced by Seymour and Thomas in [67] in order to provide a polynomial-time algorithm for computing the branch-width of planar graphs. In fact their notions are slightly more general, but we will not need those here.

Definition 2.3.7. Let G be a graph with $|V(G)| \geq 2$. A *carving* of G is pair (T, η), where T is a tree whose internal vertices have degree three, and η is a bijection from $\mathcal{L}$, the set of leaves of T, to $V(G)$.

For an edge $e \in E(T)$, let T_1, T_2 be the two components of $T \setminus e$, let $X_i = \eta(\mathcal{L} \cap V(T_i)) \subseteq V(G)$ and let τ_e denote $|\delta(X_i)|$ for $i = 1, 2$. The *width* of the carving (T, η) is defined as $\max_{e \in E(T)} \tau_e$, and the *carving-width* of G, denoted by $cw(G)$, is defined as the minimum width over all carvings of G. A carving of G is *optimal* if it has width $cw(G)$.

Notice that if we replace $V(G)$ by $E(G)$ in the above definition (so that leaves of T correspond to edges of G), and let τ_e denote the number of vertices of G incident with edges in $\eta(\mathcal{L} \cap V(T_i))$ for $i = 1, 2$, then we recover the definitions for branch-decompositions and branch-width.

If a graph has high carving-width, then either there is some vertex of high degree, or a *tilt* of high order in G:

Definition 2.3.8. Let G be a graph with $|V(G)| \geq 2$, and $k \geq 1$ be an integer. A *tilt* in G of order k is a collection $\mathcal{B}$ of subsets $X \subseteq V(G)$ with $|\delta(X)| < k$ such that

(B1) $\mathcal{B}$ contains exactly one of X, X^c for every $X \subseteq V(G)$ of order $< k$.

(B2) If $X_i \in \mathcal{B}$ for $i = 1, 2, 3$, then $X_1 \cup X_2 \cup X_3 \neq V(G)$.

(B3) $\{v\} \in \mathcal{B}$ for all $v \in V(G)$.

Theorem 2.3.9 ((4.3) in [67]). *Let G be a graph with $|V(G)| \geq 2$ and let $k \geq 1$ be an integer such that $|\delta(v)| < k$ for all $v \in V(G)$. Then G has carving-width $\geq k$ if and only if G has a tilt of order k.*

For planar graphs, we have an additional useful relationship:

Theorem 2.3.10 ((7.2) in [67]). *Let G be a connected planar graph with $|E(G)| \geq 2$, and $\mathcal{M}$ be its medial graph. Then*

$$cw(\mathcal{M}) = 2 \cdot bw(G). \tag{2.3.2}$$

The 'ratcatcher method' from [67] in fact computes the carving-width of a planar graph. This is possible since for planar graphs, we have a computationally tractable obstruction to high carving-width, called an *antipodality*:

Definition 2.3.11. Let G be a non-null connected planar graph with planar dual G^*. An *antipodality* in G of range $\geq k$ is a function α with domain $E(G) \cup R(G)$, such that for all $e \in E(G)$, $\alpha(e)$ is a non-null subgraph of G and for all $r \in R(G)$, $\alpha(r)$ is a non-empty subset of $V(G)$, satisfying:

(A1) If $e \in E(G)$, then no end of e belongs to $V(\alpha(e))$.

(A2) If $e \in E(G)$, $r \in R(G)$, and e is incident with r, then $\alpha(r) \subseteq V(\alpha(e))$, and every component of $\alpha(e)$ has a vertex in $\alpha(r)$.

(A3) If $e \in E(G)$ and $f \in E(\alpha(e))$ then every closed walk of G^* using e^* and f^* has length $\geq k$.

Theorem 2.3.12 ((4.1) in [67]). *Let G be a connected planar graph with $|V(G)| \geq 2$ and let $k \geq 0$ be an integer. Then G has carving-width at least k if and only if either $|\delta(v)| \geq k$ for some vertex v, or G has an antipodality of range $\geq k$.*

Hence in order to compute $bw(G)$ for a planar graph, by Theorem 2.3.12 it suffices to compute the maximum order of an antipodality in the medial graph of G, which is what the 'ratcatcher method' accomplishes. In Chapter 3, we explore which of these relationships hold for general surfaces.

2.4 The Traveling Salesman Problem

Let $G_n = (V, E)$ denote the complete graph on n vertices and $m = n(n-1)/2$ edges with non-negative costs $c_e \in [0, \infty)$ for every edge $e \in E$. The vertices of G_n are referred to as cities, and a hamiltonian cycle in G_n (i.e. a cycle passing through all cities) is called a *tour*. The *(symmetric) traveling salesman problem* or TSP is to find a minimum cost hamiltonian cycle in G_n. The problem is known to be NP-hard (as shown in [45]).

The following integer programming formulation of the TSP was first introduced by Dantzig, Fulkerson and Johnson in [27]. If x is a vector with m entries x_e corresponding to the edges of E, we use the notation $x(F) = \sum_{e \in F} x_e$ for $F \subseteq E$ and $x(A, B) = x(E(A, B))$ for $A, B \subseteq V$.

$$\min \quad \sum_{e \in E} c_e x_e \tag{2.4.1}$$

$$\text{subject to} \quad x(\delta(v)) = 2 \;\forall v \in V \tag{2.4.2}$$

$$x(\delta(U)) \geq 2 \;\forall U \subsetneq V,\; |U| \geq 3 \tag{2.4.3}$$

$$x \in \{0, 1\}^m. \tag{2.4.4}$$

The convex hull of the integer points satisfying all of the above constraints is called the *(symmetric) traveling salesman polytope*, and will be denoted by $TSP(n)$. If we relax condition 2.4.4 to $x \in [0,1]^m$, we obtain a linear programming problem (LP) called the *subtour relaxation*:

$$\min \quad \sum_{e \in E} c_e x_e \tag{2.4.5}$$

$$\text{subject to} \quad x(\delta(v)) = 2 \ \forall v \in V \tag{2.4.6}$$

$$x(\delta(U)) \geq 2 \ \forall U \subsetneq V, \ |U| \geq 3 \tag{2.4.7}$$

$$x \in [0,1]^m. \tag{2.4.8}$$

The polytope defined by the constraints 2.4.6 - 2.4.8 is called the *subtour polytope*, and is denoted by $SP(n)$. The constraints 2.4.6 are called *degree-constraints*, and the constraints 2.4.7 are called *subtour elimination constraints* or simply *subtour constraints*.

Let $\mathcal{I}$ be a class of inequalities (such as the subtour constraints) that are satisfied for every tour, and hence by every point in $TSP(n)$. The *separation problem* for $\mathcal{I}$ is the following problem: Given a vector $x \in \mathbb{R}^m$, find an inequality from $\mathcal{I}$ which is violated by x, or decide that no such inequality exists. An algorithm solving this problem is called an *(exact) separation algorithm.* In light of the cutting plane method (introduced in [27]) it is desirable to have polynomial-time algorithms to solve the separation problem for given classes $\mathcal{I}$ of inequalities. More generally, a fundamental theorem of Grötschel, Lovász and Schrijver [35] closely relates the separation problems to optimization problems.

Although the subtour relaxation is defined by exponentially many constraints, the afore-mentioned theorem of [35] implies that this LP can be solved in polynomial time, since we can solve the separation problem for the subtour constraints in polynomial time ([42], [56]). Hence one usually assumes $x \in SP(n)$ when solving the separation problem for a class of valid inequalities for the TSP. Unfortunately exact polynomial-time separation algorithms are only known to exist in special cases for most of the other interesting classes of inequalities for the TSP.

2.4.1 Comb inequalities

Since $TSP(n)$ is a proper subset of $SP(n)$ for $n \geq 6$, it is of great interest to find inequalities satisfied by every tour which are not of the type (2.4.6) - (2.4.8), in order to get a better approximation of $TSP(n)$. Many different classes of such inequalities for the TSP have been defined and studied. Arguably the most important one is the class of *comb inequalities*, first defined in [36] and [20]:

Definition 2.4.1. Let $d \geq 3$ be an odd integer and let $\mathcal{C} = \{H, T_1, ..., T_d\}$ be a collection of nonempty subsets of V satisfying the following conditions:

(1) $T_i \cap T_j = \emptyset$ for $1 \leq i < j \leq d$.

(2) $T_i \cap H \neq \emptyset$ for $1 \leq i \leq d$.

(3) $T_i \setminus H \neq \emptyset$ for $1 \leq i \leq d$.

Then the inequality

$$x(\delta(H)) + \sum_{i=1}^{d} x(\delta(T_i)) \geq 3d + 1 \tag{2.4.9}$$

is called a *comb inequality*, and $\mathcal{C}$ is called a comb. The sets $T_1, ..., T_t$ are called the *teeth* and the set H is called the *handle* of the comb.

All comb inequalities define facets of $TSP(n)$, as shown in [37]. If $|T_i| = 2$ for each tooth, the associated inequality is called a *2-matching inequality* or *blossom inequality* [31]. Blossom inequalities can be separated in polynomial time [57]. If d is fixed, the separation problem for combs with d teeth can be solved in time $\mathcal{O}(n^{2d})$, as shown by Carr in [18].

2.4.2 Domino parity inequalities

A generalization of comb inequalities are the *domino-parity inequalities*, introduced by Letchford in [50]. In order to define them, we first need to introduce some auxiliary concepts.

Definition 2.4.2. Let $A, B \subset V$ be two non-empty subsets such that $A \cap B = \emptyset$ and $A \cup B \neq V$. Then the pair (A, B) is called a *domino*, and

$$x(\delta(A \cup B)) + x(A, B) \geq 3 \tag{2.4.10}$$

is called a *domino inequality*.

It is easy to see that the domino inequalities are valid for $SP(n)$: just add the subtour constraints 2.4.7 for A, B and $A \cup B$ and divide by two.

Definition 2.4.3. Let $\{E_1, ..., E_k\} \subseteq E(G_n)$ be a collection of edge-sets. Then $\{E_1, ..., E_k\}$ *supports a cut* $\delta(H)$ if there is a set $H \subseteq V(G_n)$ so that $e \in \delta(H)$ if and only if $e \in E_1 \triangle ... \triangle E_k$, i.e. if and only if e belongs to an odd number of the sets $E_1, ..., E_k$.

Definition 2.4.4. Let $d \geq 3$ be an odd integer, let (A_i, B_i) for $i = 1, ..., d$ be dominoes and let $F \subseteq E(G_n)$ so that $\mathcal{D} = \{E(A_1, B_1), ..., E(A_d, B_d), F\}$ supports a cut $\delta(H)$ for some $H \subseteq V(G_n)$. Then the corresponding *domino-parity inequality* (or DP inequality) is given by

$$\sum_{i=1}^{d}(x(\delta(A_i \cup B_i)) + x(A_i, B_i)) + x(F) \geq 3d + 1. \tag{2.4.11}$$

The set H is called the *handle* of the inequality.

Note that a comb inequality with teeth $T_1, ..., T_d$ and handle H can be viewed as a DP inequality if we set $A_i = T_i \cap H$, $B_i = T_i \setminus H$ and $F = \delta(H) - \bigcup_{i=1}^{d} E(A_i, B_i)$. We will also refer to the domino $(T_i \cap H, T_i \setminus H)$ as a tooth. In fact in the literature, dominos in general DP inequalities are occasionally referred to as teeth as well, but we will only use the term ‘teeth’ in the context of comb inequalities.

We next provide a proof (following [50]) that the DP inequalities are valid for the TSP (and hence so are comb inequalities).

Theorem 2.4.5 ([50]). *Let $d \geq 3$ be an odd integer, let (A_i, B_i) for $i = 1, ..., d$ be dominoes and let $F \subseteq E(G_n)$ so that $\mathcal{D} = \{E(A_1, B_1), ..., E(A_d, B_d), F\}$ supports a cut $\delta(H)$ for some $H \subseteq V(G_n)$. Then every $x \in TSP(n)$ satisfies inequality 2.4.11.*

Proof. It suffices to show this if x is the incidence vector of a tour in G_n. Adding all domino inequalities for the dominoes (A_i, B_i) and the inequality $x(F) \geq 0$ yields

$$\sum_{i=1}^{d} (x(\delta(A_i \cup B_i)) + x(A_i, B_i)) + x(F) \geq 3d. \tag{2.4.12}$$

For an edge $e \in E(G_n)$, let μ_e denote the number of sets in $\mathcal{D}$ containing e. Using this notation, we can rearrange the terms in the above equation to get:

$$\sum_{i=1}^{d} x(\delta(A_i \cup B_i)) + \sum_{e \in E(G_n)} \mu_e x_e \geq 3d. \tag{2.4.13}$$

If we split the edges in the second sum with respect to membership in $\delta(H)$, we obtain:

$$\sum_{i=1}^{d} x(\delta(A_i \cup B_i)) + x(\delta(H)) + \sum_{e \in \delta(H)} (\mu_e - 1)x_e + \sum_{e \notin \delta(H)} \mu_e x_e \geq 3d. \tag{2.4.14}$$

Since $x(\delta(X))$ is even for every $X \subseteq V(G_n)$, and μ_e is odd if and only if $e \in \delta(H)$ since $\mathcal{D}$ supports the cut $\delta(H)$, we have that the left hand side is an even integer. Hence the inequality remains valid if we add one to the right hand side. □

The *surplus* (with respect to a vector $\hat{x}$) of a linear inequality $a \cdot x \geq b$ is defined as $a \cdot \hat{x} - b$, and the inequality is *violated* by $\hat{x}$ if the surplus is negative.

The following simple observation is important in the design of separation routines for DP (and comb) inequalities.

Proposition 2.4.6. *If $\mathcal{D} = \{(A_1, B_1), ..., (A_d, B_d), F\}$ defines a DP inequality, and $\hat{x} \in SP(n)$, then the following statements hold:*

(1) The DP inequality is violated by at most one.

(2) The surplus of the DP inequality is given by the sum of the surplus values of its domino inequalities, plus $\hat{x}(F) - 1$.

(3) If the DP inequality is violated, then for each domino (A_i, B_i), $i = 1, \ldots, d$, the associated domino inequality has surplus < 1.

Proof. The first two statements follow immediately from the derivation of the DP inequality in the proof of Theorem 2.4.5: The fact that he DP inequality is obtained by adding 1 to the right-hand side of a sum of valid inequalities for $\hat{x} \in SP(n)$ (namely the domino inequalities and $x(F) \geq 0$) implies (1) and (2). Since a DP inequality is violated if and only if its surplus is negative, and all inequalities used in its derivation have non-negative surplus as $\hat{x} \in SP(n)$, statement (2) implies (3). $\square$

CHAPTER III

BRANCH-WIDTH OF EMBEDDED GRAPHS

3.1 Introduction

In this chapter we develop a geometric description of tangles in an embedded graph (Theorem 3.4.2), and prove a result on the relationship of the branch-width of an embedded graph and the carving-width of the medial graph (Theorem 3.6.1). We also discuss how these results relate to the polynomial-time algorithm of [67] to compute the branch-width of a planar graph, and explain why the method used there does not seem to generalize to arbitrary surfaces (Section 3.7).

In their Graph Minors XI paper [63], Robertson and Seymour introduced several new concepts with the goal of establishing a distance function for graphs embedded on a surface with high face-width. In particular, they showed that a special type of tangles (see Definition 2.3.2), called *respectful tangles*, are the dual obstructions for the face-width of an embedding. A tangle $\mathcal{T}$ of order θ in a graph $G \hookrightarrow \Sigma$ is called *respectful* if for every simple closed curve F in Σ with $F \cap U(G) \subseteq V(G)$ and $|F \cap V(G)| < \theta$, there is a closed disc $\triangle \subseteq \Sigma$ bounded by F so that the separation $(G \cap \triangle, G \cap cl(\Sigma \setminus \triangle))$ belongs to $\mathcal{T}$. One of the main results of [63] is the following:

Theorem 3.1.1 ((4.1) in [63]). *Let G be a graph that is 2-cell embedded in a surface Σ other than the sphere, and let $\theta \geq 1$ be an integer. Then G has face-width $\geq \theta$ if and only if there is a respectful tangle in G of order θ.*

Recall that general tangles (see 2.3.2) are the obstructions to low branch-width: The branch-width of a graph G is given by the maximum order of a tangle in G (Theorem 2.3.3). Since respectful tangles are a special kind of tangle, Theorem 3.1.1

implies that the face-width of a Σ-embedded graph G is a lower bound on its branch-width, if $\Sigma \neq \mathbb{S}_0$. However this lower bound can be arbitrarily bad: it is easy to construct embedded graphs of low face-width, but arbitrarily high branch-width - for example just take a large planar grid and add a few edges to achieve the desired face-width.

In order to prove Theorem 3.1.1, Robertson and Seymour arrive at what could be viewed as a 'geometric' description of respectful tangles, called *slopes*. These slopes and the results in [63] based on them turn out to be a key step in the 'ratcatcher method' for computing the branch-width of a planar graph [67]. While computing the branch-width and thus a tangle of maximum order is NP-hard for general graphs, Seymour and Thomas show that this difficulty disappears on planar graphs. Their method essentially computes the maximum order of a slope (in an appropriate graph), which gives the maximum order of a respectful tangle. But since the graph is planar, the extra condition of being 'respectful' is vacuous, and so by Theorem 2.3.3, we obtain the branch-width of the graph.

A natural question to ask is whether the 'ratcatcher method' can be extended to compute the branch-width of a graph G embedded in a surface other than the sphere. In particular, can one obtain a similar 'geometric' description of arbitrary tangles in G (not just respectful ones)?

In Sections 3.2, 3.3, 3.6, we generalize some of the concepts and results from [63]. In particular, we introduce a generalized version of slopes and prove that they characterize tangles in embedded graphs (Theorem 3.4.2), thus answering the above question in the positive. As an application, we prove a result relating the branch-width of an embedded graph to the carving-width of its medial graph (Theorem 3.6.1), generalizing a result from [67].

We also show that the second main theorem of [63] no longer holds in this generalized setting (Theorem 3.5.4), and in Section 3.7, we explain why this, together with the other results from this chapter, implies that the 'ratcatcher method' does not seem to generalize to higher surfaces.

3.2 Spots, borders and generalized slopes

Robertson and Seymour defined their notion of a slope as follows, which we will refer to as a 'traditional slope' (in contrast to the generalized slopes we define later):

Definition 3.2.1. Let $\theta \geq 1$ be a half integer, and let $G \hookrightarrow \Sigma$ be 2-cell embedded. A *(traditional) slope* in G of order θ is a function ins which assigns to every cycle C of G of length $< 2\theta$ a closed disc $ins(C) \subseteq \Sigma$, bounded by $U(C)$, such that

S1) if C_1, C_2 are cycles of G, both of length $< 2\theta$, and $U(C_1) \subseteq ins(C_2)$, then $ins(C_1) \subseteq ins(C_2)$,

S2) if P_0, P_1, P_2 are three paths of G of positive length, with the same ends, mutually internally disjoint, and the cycles $P_1 \cup P_2, P_0 \cup P_2, P_0 \cup P_1$ all have length $< 2\theta$, then one of $ins(P_1 \cup P_2), ins(P_0 \cup P_2), ins(P_0 \cup P_1)$ includes the other two.

Remark 3.2.2. If G has a slope of order θ, then in particular every cycle of length $< 2\theta$ must be contractible, i.e. $fw(G) \geq 2\theta$.

Remark 3.2.3. If we aim to define a slope in a graph G (where every cycle C of length $< 2\theta$ is contractible), then we only have a choice how to define $ins(C)$ if Σ is the sphere, because otherwise C bounds exactly one disc (which would have to be chosen as $ins(C)$).

Similarly to a tangle, a slope is used to define the notion of a 'small' side. Each contractible cycle (of length $< 2\theta$) separates the surface into two components, and suppose we call the side $ins(C)$ the 'small' side. Suppose a graph $G \hookrightarrow \Sigma$ has a slope of order θ. Then the slope axioms will imply that Σ is not the union of three 'small' sides. To get the correspondence to respectful tangles, Robertson and Seymour extend the notion of a 'small' side from insides of cycles to arbitrary sets of regions, and show that $R(G)$ (the set of regions of G) is not the union of three 'small' sets of regions. In our generalization, we use the same strategy: First define a 'small side' for 'simple' separations, and then extend the notion of a 'small side' to prove the latter result for our generalized version of slopes.

The reason that cycles show up in the definition of a traditional slope is that a circle is the only way to separate the surface into precisely two connected components (so that their boundary is 'short') if there are no short non-contractible curves. However if the condition of high face-width is dropped, there are other ways to separate the surface into exactly two parts, and we have to extend our definition of the slope function *ins* to those as well. This motivates the following definition:

Definition 3.2.4. A graph B with no isolated vertices and embedded in Σ (not necessarily 2-cell) is called a *border* if B has exactly two regions, and every edge of B is incident with both of them. A (closed) set $S \subseteq \Sigma$ is called a *spot* if there is a border B embeddable in Σ so that S is the closure of exactly one of the regions of B. If a border B is a subgraph of a graph G, we say B is a *border in* G.

Example 3.2.5. In the projective plane $\mathbb{N}_1$, one can show that there are exactly two types of borders: Contractible cycles, and two non-contractible cycles intersecting in a single vertex.

On the torus $\mathbb{S}_1$, a border B is of one of the following types (see also Figure 1):

- B is a contractible cycle.
- $B = C_1 \cup C_2$, where C_1, C_2 are two disjoint non-contractible, homotopic cycles.
- $B = C_1 \cup C_2$, where C_1, C_2 are two non-contractible homotopic cycles, intersecting in exactly one vertex.
- $B = C_1 \cup C_2 \cup C_3$, where C_1, C_2, C_3 are non-contractible, pairwise non-homotopic cycles intersecting in a single common vertex (and are disjoint otherwise).

Note that there are two types of borders for the second and third case, depending on the homotopy type of the two cycles involved (either C_1 and C_2 are both meridians or both equators on the torus).

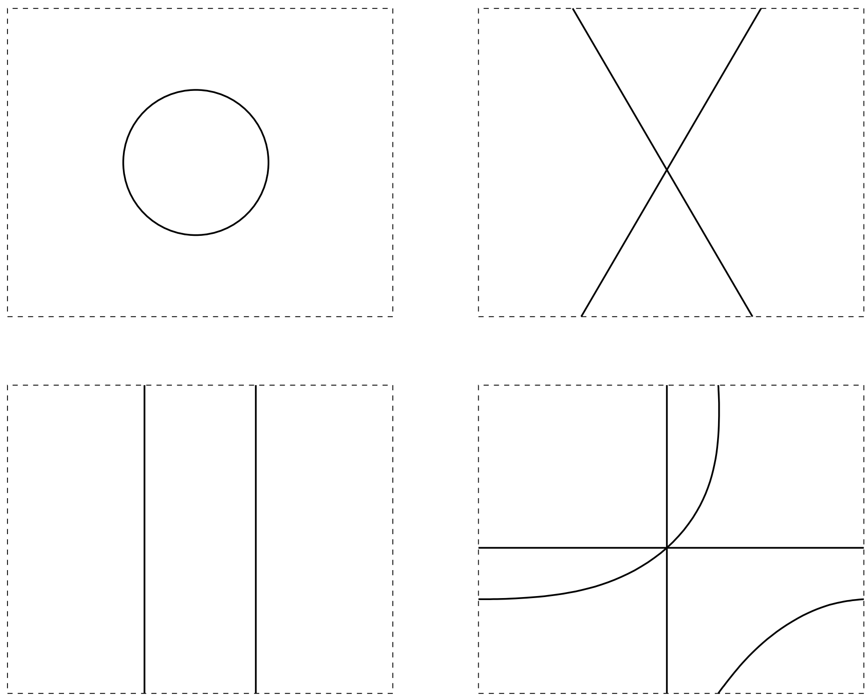


Figure 1: Borders on the torus

We next list some easy facts about borders which we will need later.

Proposition 3.2.6. *Every border B is either a separating cycle, or the edge-disjoint union of at least two non-separating cycles.*

Proof. At a vertex $v \in V(B)$, every edge adjacent to v in B is incident with both regions r_1 and r_2 of B. It follows that v has even degree, so B is eulerian and hence the edge-disjoint union of cycles $C_1, ..., C_k$. If say C_1 is separating, then the graph C_1 has two distinct regions, and since the regions of B are contained in the regions of C_1, we have that $k = 1$ and $B = C_1$. $\square$

It will sometimes be helpful to construct a 'dual' of an embedded graph, even if that graph is not connected (typically one does not define a dual of a disconnected graph, since many standard theorems would no longer hold, but we won't need any of those here). Let $G \hookrightarrow \Sigma$ be an embedded graph, not necessarily connected. Then we define a new embedded (multi-)graph $G^\star$, with vertex-set $V(G^\star) = R(G)$, and edges $E(G^\star)$ as follows: For every $e \in E(G)$, if e is incident with regions $r_1, r_2 \in R(G)$, then we add an edge $e^\star = r_1 r_2$ to $E(G^\star)$. Note that $G^\star$ is always connected, and if G is connected, then $G^\star$ is simply the dual of G in the standard graph-theoretic sense, denoted by G^*. For $X \subseteq E(G)$, we denote by $X^\star$ the corresponding set of edges in $E(G^\star)$.

Proposition 3.2.7. *Let G be Σ-embedded, and let $G^\star$ be as above. If B is a border in G, then $E(B)^\star$ is a bond in $G^\star$, and conversely, if $\delta(X)$ is a bond in $G^\star$, then $\delta(X) = E(B)^\star$ for some border B in $G^\star$.*

Proof. Suppose B is a border in G. The two regions of B partition the regions of G into (two non-empty) classes, corresponding to a partition $X \cup X^c = V(G^\star)$ where $G^\star[X]$, $G^\star[X^c]$ are connected, i.e $\delta(X)$ is a bond in $G^\star$. Since every edge in $E(B)$ is incident with both regions of B, it follows that $\delta(X) = E(B)^\star$.

Conversely, if $\delta(X)$ is a bond in $G^\star$, then the subgraph B of G induced by the edges corresponding to $\delta(X)$ has exactly two regions (since $G^\star[X]$, $G^\star[X^c]$ are connected and non-empty), and each edge $e \in E(B)$ is incident with both of them, since $e^\star$ is an X - X^c edge in $G^\star$. □

Recall that for $Z \subseteq R(G)$, $\partial(Z)$ denotes the subgraph of G induced by all edges of G which are incident with a region in Z, and one not in Z.

Proposition 3.2.8. *Let G be Σ-embedded, and let $r \in R(G)$. Let $r_1, \ldots, r_k$ be the regions of the graph $\partial(r)$ other than r. Then $\partial(r_i)$ is a border B_i for $i = 1, \ldots k$, and $\partial(r)$ is the edge-disjoint union of $B_1 \cup \ldots \cup B_k$.*

Proof. Both statements follow immediately from the fact that $\partial(r_i) \subseteq \partial(r)$, i.e. every edge in $\partial(r_i)$ is incident with r and r_i, for $i = 1, \ldots, k$. □

Lemma 3.2.9. *For every surface Σ there is a non-negative constant $c(\Sigma)$, depending only on Σ, so that if $B \hookrightarrow \Sigma$ is a border, then*

$$|E(B)| \leq |V(B)| + c(\Sigma), \tag{3.2.1}$$

where $c(\Sigma) = (2 - \chi(\Sigma)) \cdot \max\{2, -3\chi(\Sigma)/2\}$ if Σ is orientable, and $c(\Sigma) = (2 - \chi(\Sigma)) \cdot \max\{2, -3\chi(\Sigma)\}$ otherwise.

Proof. Note that for any connected graph H embedded in Σ, we have $|E(H)| \leq |V(H)| + |R(H)| - \chi(\Sigma)$, where $\chi(\Sigma)$ denotes the Euler characteristic of Σ: If the embedding is 2-cell, equality holds by Theorem 2.2.2, and if not, we can obtain a 2-cell embedding of H in some surface Σ' with $\chi(\Sigma') > \chi(\Sigma)$ and hence applying Theorem 2.2.2 there yields the result. Let $B_1, B_2, ..., B_k$ be the components of B. Applying the above to each B_i and summing all k inequalities yields

$$|E(B)| \leq |V(B)| + k(2 - \chi(\Sigma)) \tag{3.2.2}$$

since $|R(B_i)| \leq 2$ as B has exactly two regions (in fact we have $|R(B_i)| = 1$ unless B is connected). If $k \leq 2$ we are done, so assume B has at least 3 components. Pick a non-contractible cycle C_i in each B_i for $i = 1 \ldots k$ (this is possible by Proposition 3.2.6). Then any two of them are non-homotopic: If say C_1 was homotopic to C_2, then $C_1 \cup C_2$ separates the surface into exactly two connected parts (since C_1 and C_2 are disjoint), and hence it follows that $B = C_1 \cup C_2$, contradicting our assumption that $k \geq 3$. As shown in [52] (see also p. 107 of [53]), we have that $k \leq c''$ for some constant c'' depending only on Σ (in fact $c'' \leq g$ if $g \leq 1$, and $c'' \leq 3g - 3$ otherwise). Hence combining this with inequality 3.2.2 yields the result (recall that $\chi(\Sigma) = 2 - 2g$ if Σ is orientable, and $\chi(\Sigma) = 1 - g$ if it is not). $\square$

The basic idea for generalized slopes is to replace cycles by borders. However care has to be taken with respect to the following two aspects.

First, in order to prove an exact correspondence to tangles, it is not sufficient to set the length of a border to be the total number of vertices (or edges): In fact, we need to define the 'length' of a border with respect to a weight function on the vertices. Such a weight function is also needed in [63], but can be avoided at the level of slopes because in the application, the weight of a cycle in a traditional slope is always exactly half of the number of vertices in the cycle, a fact which no longer holds for general borders.

More specifically, a border B in the radial graph $\mathcal{R}(G)$ of G (where the slope will be defined in the application for tangles) induces a separation in the original graph G, and hence we should define the weight of a border to be exactly $|V(B) \cap V(G)|$, the order of the corresponding separation in G. If B is a cycle, then its weight will always be exactly half its length, as $\mathcal{R}(G)$ is bipartite (with bipartition $V(G) \cup R(G)$). However for a general border B in $\mathcal{R}(G)$, $|V(B) \cap V(G)|$ need not be a fixed fraction of either $|V(B)|$ or $|E(B)|$: For example if B consists of two homotopic, noncontractible cycles on the torus intersecting in a single vertex x (so that $|V(B)|$ is odd), then $|V(B) \cap V(G)| \in \left\{\lfloor\frac{|V(B)|}{2}\rfloor, \lfloor\frac{|V(B)|}{2}\rfloor + 1\right\}$, depending on whether or not $x \in V(G)$. Hence the correct notion of length or weight to consider is the following:

Definition 3.2.10. For a fixed set $\Omega \subseteq V(G)$ and an integer $\theta \geq 1$, we define the Ω-*weight* or simply the *weight* of a subgraph $G' \subseteq G$ to be $w(G') = |V(G') \cap \Omega|$ and say that G' is θ-*light* or simply *light* if $w(G') < \theta$. Similarly if $X \subseteq V(G)$, then its weight $w(X)$ is defined as $|X \cap \Omega|$.

Second, what may seem to be the most natural definition of a generalized slope (in light of the above comments) will only be called a *pre-slope*. We will define the new (stronger) notion of a slope later when we need it, and point out why such a stronger notion is necessary.

Three sets of $Z_1, Z_2, Z_3 \subseteq R(G)$ form a *partition* (Z_1, Z_2, Z_3) of $R(G)$ if they are pairwise disjoint and have union $R(G)$.

Definition 3.2.11. Let $\theta \geq 1$ be an integer, let G be a Σ-embedded graph and $\Omega \subseteq V(G)$. A *pre-slope* in G of order θ (with respect to Ω) is a function ins which assigns to every light border B in G a spot $ins(B)$, bounded by B, such that

S1) if B_1, B_2 are light borders in G, and $U(B_1) \subseteq ins(B_2)$, then $ins(B_1) \subseteq ins(B_2)$,

S2) if (Z_1, Z_2, Z_3) is a partition of $R(G)$ so that $\partial(Z_i) = B_i$ for some light border B_i for $i = 1, 2, 3$, then $ins(B_1) \cup ins(B_2) \cup ins(B_3) \neq \Sigma$.

The following terms were defined in [63], and we now redefine them in our more general setting. Let G be embedded in Σ, and let ins be a pre-slope of order θ in G. We say a subgraph $G' \subseteq G$ is *confined* if every border in G' is light. In that case we define $ins(G')$ to be the union of $U(G')$ and $ins(B)$ for all borders B in G'. Clearly $ins(G')$ is the union of vertices, edges and regions of G'. A set $X \subseteq \Sigma$ is *captured* by G if $X \subseteq ins(B)$ for some light border B in G. Finally, a set $Z \subseteq R(G)$ is called *small* if $\partial(Z)$ is a confined graph and $Z \subseteq ins(\partial(Z))$.

With the new definitions, we now follow the steps from [63] in order to establish the relationship with general tangles. The majority of those intermediate steps and their proofs are essentially identical to the ones presented in [63]. The only change is often to replace the original concepts by the generalized ones introduced in this thesis (i.e. cycles by borders, traditional slopes by (pre-)slopes etc.), and observe that the statements remain true. In such cases, we still provide the proofs here for better readability, but clearly point out if they are essentially the same as in [63].

For the remainder of this section, we assume G is a Σ-embedded graph with a pre-slope of order $\theta \geq 1$ (with respect to some set $\Omega \subseteq V(G)$).

For later reference, we state the following obvious fact:

Proposition 3.2.12 ((4.2) from [63]). *Let $G' \subseteq G$. Then the restriction of ins to the set of light borders of G' is a pre-slope of order θ in G'.*

Proposition 3.2.13. *If B_1, B_2 are two light borders in a graph G, then $ins(B_1) \cup ins(B_2) \neq \Sigma$.*

Proof. Suppose not, i.e. $ins(B_1) \cup ins(B_2) = \Sigma$. Let $e \in E(B_1)$, with regions $r_1, r_2 \in R(G)$ incident at e. Then exactly one of them is contained in $ins(B_1)$ (since B_1 is a border), say r_1. Hence by our assumption we must have $r_2 \subseteq ins(B_2)$, and therefore $u(e) \subseteq ins(B_2)$. Since $e \in E(B_1)$ was arbitrary and B_1 has no isolated vertices, it follows that $U(B_1) \subseteq ins(B_2)$, and so by S1), $ins(B_1) \subseteq ins(B_2)$, contrary to our initial assumption (clearly $ins(B_2) \neq \Sigma$). $\square$

The first main step is to prove a result analogous to (4.3)(ii) from [63]. However the proof here is very different from (and somewhat more complicated than) the original one in [63] - in particular statement (4.3)(i), which is used in the proof of (4.3)(ii) in [63], no longer holds.

Lemma 3.2.14 ((4.3)(ii) in [63]). *Let $G' \subseteq G$ be confined, B be a border in G' with $ins(B)$ maximal. Then for every edge $e \in E(B)$, G' does not capture the region of G' incident with e not included in $ins(B)$.*

Proof. Let $e \in E(B)$. Since B is a border in G', e is adjacent to two distinct regions of G'. Let r be the region incident with e and $r \not\subseteq ins(B)$. Suppose r is captured by G', i.e. there is a border B' in G' with $r \subseteq ins(B')$, and hence also $e \subseteq ins(B')$.

Claim 3.2.14.1. $ins(B)$ and $ins(B')$ cross, i.e. $ins(B') \cap ins(B) \neq \emptyset$, $ins(B') \setminus ins(B) \neq \emptyset$, $ins(B) \setminus ins(B') \neq \emptyset$ and $ins(B) \cup ins(B') \neq \Sigma$.

Proof of claim. The first inequality holds since $e \subseteq ins(B') \cap ins(B)$, and the second holds because of the region r. If $ins(B) \subseteq ins(B')$, then equality would hold by maximality of $ins(B)$, contrary to the second inequality, and so the third inequality holds as well. The last inequality is Lemma 3.2.13. $\square$

Now consider the graph $B \cup B'$.

Claim 3.2.14.2. For every region $r' \in R(B \cup B')$ with $r' \subseteq (ins(B))^c$, $\partial(r')$ contains an edge of B.

Proof of claim. Suppose not, i.e. $E(\partial(r')) \subseteq E(B') \setminus E(B)$ and in particular r' is a region of B'. Since B' is a border, we either have $ins(B') = cl(r')$ or $ins(B') = (r')^c$. Now the first case cannot happen, since $e \subseteq ins(B')$, but $\partial(r')$ and hence $cl(r')$ contains no edge of B by assumption. Hence we have $ins(B') = (r')^c$. But since $r' \subseteq (ins(B))^c$, we have $ins(B') = (r')^c \supseteq ins(B)$, contrary to Claim 3.2.14.1. □

Now let H be the graph $(B \cup B')^\star$, as defined before Proposition 3.2.7. Since B is a subgraph of $B \cup B'$, every region of $B \cup B'$ is either contained in $ins(B)$, or disjoint from it. Let $X_1 \subseteq V(H)$ consist of the regions of $B \cup B'$ contained in $ins(B)$, and $X_2 = V(H) \setminus X_1$ be the regions in $(ins(B))^c$. In particular $\delta(X_1) = \delta(X_2) = E(B)^\star$, so since B is a border in $B \cup B'$, we have that $H[X_i]$ is connected for $i = 1, 2$ by Proposition 3.2.7. Note that $|X_2| \geq 2$: If $|X_2| = 1$, i.e. $(ins(B))^c$ is a region of $B \cup B'$, we would have $U(B') \subseteq ins(B)$, and so by S1) $ins(B') \subseteq ins(B)$, contrary to Claim 3.2.14.1.

Since $H[X_2]$ is a connected graph on at least two vertices, we can choose $r' \in X_2$ so that $H[X_2] \setminus r'$ is connected. Let $B'' = \partial(r')$ and $B''' = \partial(X_2 \setminus r')$ be subgraphs of $B \cup B'$.

Claim 3.2.14.3. B'' and B''' are light borders in $B \cup B'$.

Proof of Claim. B'' partitions $V(H)$ into r' and $X_1 \cup (X_2 \setminus r')$. By Claim 3.2.14.2, there is an X_1 - $(X_2 \setminus r')$ edge in H , and since $H[X_1]$ and $H[X_2 \setminus r']$ are connected, we have that $\delta_H(r')$ is a bond in H, and so by Proposition 3.2.7, B'' is a border since $\delta_H(r') = E(B'')^\star$. Similarly B''' partitions $V(H)$ into $X_2 \setminus r'$ and $X_1 \cup r'$, and since r' is incident with an element of X_1 in H by Claim 3.2.14.2, applying Proposition 3.2.7 establishes that B''' is a border in $B \cup B'$. Clearly B'', B''' are light since $B \cup B'$ is confined. □

Note that $ins(B'') \neq (r')^c$: If $ins(B'') = (r')^c$, then since $r' \in X_2$ means $r' \subseteq (ins(B))^c$, this would imply $ins(B'') \supseteq ins(B)$. From the maximality of B, equality holds, and so $(ins(B))^c$ consists only of r', contrary to $|X_2| \geq 2$. Similarly if $ins(B''')$ consists of the closure of the regions of $X_2 \backslash r' = X_1 \cup r'$, this contradicts the maximality of B since $r' \subseteq (ins(B))^c$. Hence $ins(B'') = cl(r')$, $ins(B''')$ is the closure of the union of the regions in $X_2 \backslash r'$, and therefore we would have $ins(B) \cup ins(B'') \cup ins(B''') = \Sigma$, contrary to S2) (note that $(X_1, X_2 \setminus \{r'\}, \{r'\})$ is a partition of $R(G)$). This completes the proof of Lemma 3.2.14. $\square$

Lemma 3.2.15 ((4.4) in [63]). *If G' is confined, then $\Sigma \setminus ins(G')$ is a region of G'.*

Proof. The proof is essentially the same as in [63], and we only include it for completeness. We first claim that some region of G' is not captured by G'. If G' contains no border then G' captures no region at all. If G' has a border, we can choose a border B with $ins(B)$ maximal, and so the claim follows from Lemma 3.2.14.

It remains to show that there is (at most) one region of G' not captured by G'. We will show by induction on $|E(G')|$ that if r'_1, r'_2 are distinct regions of G', then at least one of them is captured by G'.

First consider the case that G' contains no border. Then G' has only one region, for otherwise if $r \in R(G')$ is not the only region of G', then the graph $\partial(r)$ has a region other than r, which is bounded by some border in $\partial(r) \subseteq G'$ by Proposition 3.2.8, contradiction. Hence there is nothing to prove for this case.

Now assume that G' contains a border B. Let $r' \in R(G')$ be a region captured by B, and let e be an edge of G' incident with r'. Let $G'' = G' \setminus e$, and let r''_1, r''_2 be regions of G'' with $r'_i \subseteq r''_i$, for $i = 1, 2$. Since G' captures r', we may assume that $r'_1, r'_2 \neq r'$, for otherwise our claim holds. Hence at least one of r'_1, r'_2 is not incident with e (since e is incident with r'), say r'_1. Therefore we have $r'_1 = r''_1$, and so $r''_1 \neq r''_2$ since $r'_2 \nsubseteq r'_1 = r''_1$. By induction, G'' captures at least one of the two distinct regions r''_1, r''_2, and hence at least one of r'_1, r'_2, and so the the claim follows since $G'' \subseteq G'$. $\square$

Lemma 3.2.16 ((4.5) in [63]). *Let $G' \subseteq G$ be confined, and let $r' \in R(G')$. Suppose G captures r'. Then so does G'.*

Proof. The statement is the same as in [63], but the proof is somewhat different. Let B be a border in G capturing r', i.e. $r' \subseteq ins(B)$. Let $r_1, \ldots, r_k$ be the regions of the graph $\partial(r')$ other than r'. By Proposition 3.2.8, $B_i = \partial(r_i)$ is a border in $\partial(r') \subseteq G'$, for $i = 1, \ldots, k$. Note that all B_i are light since G' is confined. Let r be a region of G that is not captured by B. Then $r \neq r'$ implies $r \subseteq r_{i_0}$, for some $i_0 \in \{1, \ldots, k\}$. Now $B_{i_0} \subseteq \partial(r')$ implies $U(B_{i_0}) \subseteq ins(B)$, and so S1) implies that $ins(B_{i_0}) \subseteq ins(B)$. If $ins(B_{i_0}) = cl(r_{i_0})$, then $r \subseteq r_{i_0} \subseteq ins(B_{i_0}) \subseteq ins(B)$, contrary to r not being captured by B. Hence $ins(B_{i_0}) = (r_{i_0})^c$. But then $r' \subseteq (r_{i_0})^c = ins(B_{i_0})$, and so r' is captured by B_{i_0}, and hence by G'. ☐

Since the proof of the following fact from [63] is easy and does not use (pre-)slopes, we will not repeat it here.

Proposition 3.2.17 ((5.1) in [63]). *Let $Z \subseteq R(G)$, let $G' \subseteq G$ and let $r' \in R(G')$. If there are regions $r_1 \in Z$, $r_2 \in R(G) \setminus Z$ with $r_1, r_2 \subseteq r'$, then there are such regions r_1, r_2 which in addition are incident with some edge e of G with $e \notin E(G')$.*

We conclude this section by a result on maximal borders, which will be useful later.

Lemma 3.2.18. *Let B be a border in a confined graph G' with $ins(B)$ maximal, and let B' be a border in G' with $ins(B') \nsubseteq ins(B)$. Then $ins(B) \cap ins(B') \subseteq V(B) \cap V(B')$. In particular, no edge or region of G' is captured by both B and B'.*

Proof. Clearly $ins(B) \cap ins(B')$ is the union of vertices, edges and regions of G'. First suppose there is some region $r_1 \in R(G')$ with $r_1 \subseteq ins(B) \cap ins(B')$. Let r_2 be a region captured by B' but not by B (such a region exists by our assumption $ins(B') \nsubseteq ins(B)$). By Proposition 3.2.17 (applied to the graph B', where we chose

r' to be the unique region of B' captured by B', and Z to consist of the regions of G' captured by B), we may assume there is an edge e incident with r_1 and r_2, in particular $e \in E(B)$. But r_2 is captured by G' (through B'), contrary to Lemma 3.2.14. So $ins(B) \cap ins(B')$ does not contain any regions.

Suppose there is an edge e captured by B and B'. Let r_1, r_2 be the two regions incident at e. Since e is captured, each of B, B' captures at least one of r_1, r_2. Since $ins(B) \cap ins(B')$ contains no region, we may assume $r_1 \subseteq ins(B) \setminus ins(B')$ and $r_2 \subseteq ins(B') \setminus ins(B)$. But then in particular $e \in E(B)$ and r_2 is contrary to Lemma 3.2.14.

Finally if $v \in ins(B) \cap ins(B')$, then for some edge $e \in \delta(v)$, $e \subseteq ins(B)$, and for some $e' \in \delta(v)$, $e' \subseteq ins(B')$, so by the above, v belongs to $\partial(ins(B)) \cap \partial(ins(B')) = B \cap B'$, as claimed. □

3.3 Small sets

As in the previous section, let G be a Σ-embedded graph with a pre-slope of order θ, with respect to some $\Omega \subseteq V(G)$. Recall that $Z \subseteq R(G)$ is *small* if $\partial(Z)$ is confined (i.e. every border in $\partial(Z)$ has weight $< \theta$), and $Z \subseteq ins(\partial(Z))$.

The next two steps are identical to [63], and their proofs are obtained from the original ones by replacing cycles with borders, and traditional slopes with pre-slopes.

Lemma 3.3.1 ((5.2) in [63]). *If $Z \subseteq R(G)$ and $\partial(Z)$ is light, then exactly one of Z, $R(G) \setminus Z$ is small.*

Proof. Note that $\partial(Z)$ is confined since it is light, and so by Lemma 3.2.15 there is a unique region r_0 of $\partial(Z)$ not captured by $\partial(Z)$. Hence a region $r \in R(G)$ is captured by $\partial(Z)$ if and only if $r \nsubseteq r_0$. In particular not every region of $R(G)$ is contained in r_0, and so at most one of $Z, R(G) \setminus Z$ is small.

Suppose for a contradiction that neither is small, i.e. there exist $r_1 \in Z$, $r_2 \in R(G) \setminus Z$ with $r_1, r_2 \subseteq r_0$. By Proposition 3.2.17, we may choose r_1, r_2 both incident with some edge of G not in $E(\partial(Z))$, contrary to the fact that $r_1 \in Z$ and $r_2 \notin Z$. $\square$

Lemma 3.3.2 ((5.3) in [63]). *If $Z \subseteq R(G)$ is small, $Z' \subseteq Z$ and $\partial(Z')$ is light, then Z' is also small.*

Proof. Suppose $r \in Z'$. We have to show that r is captured by $\partial(Z')$. Since $Z' \subseteq Z$ and Z is small, there is a light border B in $\partial(Z)$ capturing r. Let r' be the region of $\partial(Z') \subseteq G$ containing r.

Claim 3.3.2.1. r' is captured by B.

Proof of claim. Suppose not, and let X be the regions of G captured by B. Since $X \cap Z'$ contains some (namely r) but not every region of G included in r', by Proposition 3.2.17 we may pick regions $r_1, r_2 \in R(G)$ with $r_1, r_2 \subseteq r'$ so that $r_1 \in X \cap Z'$, $r_2 \notin X \cap Z'$, and both are incident with some edge $e \notin \partial(Z')$. But $r_1 \in Z'$ and $e \notin \partial(Z')$ implies $r_2 \in Z'$, and so $r_2 \notin X$. But $r_1 \in X$ implies $e \in E(\partial(X)) = E(B) \subseteq E(\partial(Z))$, a contradiction since both of r_1, r_2 belong to Z' and hence to Z. $\square$

Hence by the claim, r' is captured by G, and by Lemma 3.2.16, the subgraph $\partial(Z')$ of G captures $r' \in R(\partial(Z'))$ since it is light and hence confined by assumption. Therefore $r \subseteq r'$ is captured by $\partial(Z')$, as required. This proves Lemma 3.3.2. $\square$

We now come to discussing the key lemma in this context. In [63], it is shown that if G contains a traditional slope, then no three small sets Z_1, Z_2, Z_3 make up all of $R(G)$. If G is the radial graph of an embedded graph H, then a separation in H corresponds to a partition of $R(G)$ into two sets Z, Z^c, so clearly this result is needed to establish the second (and most important) axiom for a tangle in H.

Unfortunately, the existence of a pre-slope is not enough to obtain such a result: Below we give a simple example of a graph that has a pre-slope of order four, and yet its set of regions is the union of three small sets.

For $k, l \geq 2$, the $k \times l$ *toroidal grid* $G_{k \times l}$ is the graph embedded in the torus $\mathbb{S}_1$ with vertices $V(G) = \{(i,j) \mid 1 \leq i \leq k,\ 1 \leq j \leq l\}$, and an edge $(i,j)(i',j')$ for

$1 \le i, i' \le k$, $1 \le j, j' \le l$ whenever $|i - i'| + |j - j'| = 1$, or $i = i'$ and $\{j, j'\} = \{1, l\}$, or $\{i, i'\} = \{1, k\}$ and $j = j'$ (if $k = 2$ or $l = 2$, this creates parallel edges). We say that $G_{k \times l}$ has k rows and l columns, and we assume that $G_{k \times l}$ is embedded in $\mathbb{S}_1$ in the natural way where every face is bounded by a disc whose boundary contains exactly 4 edges (see Figure 2).

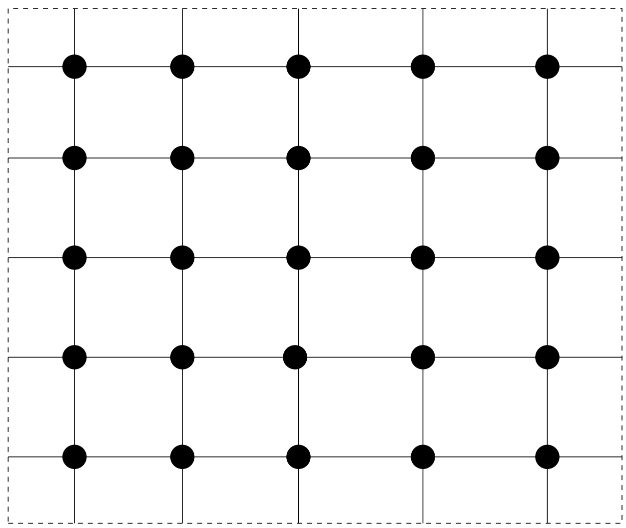


Figure 2: The natural embedding of the 5×5 toroidal grid $G_{5 \times 5}$ on the torus

Example 3.3.3. Let G be the 2×3 toroidal grid embedded in $\mathbb{S}_1$, and let $\mathcal{R}$ be the radial graph of G (also embedded in $\mathbb{S}_1$). We first construct a pre-slope of order 4 in $\mathcal{R}$ (with respect to $\Omega = V(G)$), by defining $ins(B)$ for each border of weight ≤ 3. It is not hard to see that there are only three types of such borders: Either B is a contractible four-cycle bounding some region of $\mathcal{R}$ (and hence has weight two), or B is a contractible 6-cycle bounding two regions (and hence has weight 3), or $B = \partial(Z)$ for some $Z \subseteq R(\mathcal{R})$ with $|Z| = 4$, and the common intersection of the boundaries of the 4 regions in Z contains a unique vertex $v \in V(G)$. In the latter case, such a border B has weight 3, and the 4 regions of Z correspond to $\delta(v)$ in G, for the common vertex v (and hence there are exactly 6 such borders B).

If B is a border of the first or second type, we let $ins(B)$ be the (closure of the) one or two regions of $\mathcal{R}$ bounded by B, and if B is of the third type, we define $ins(B)$

to be the (closure of the) 4 regions in Z.

We claim that this defines a pre-slope in $\mathcal{R}$: Clearly the first axiom holds. Suppose that $\Sigma = ins(B_1) \cup ins(B_2) \cup ins(B_3)$ for three borders B_1, B_2, B_3 of weight at most three. Since $|R(\mathcal{R})| = |E(G)| = 12$, it follows that all B_i must be of the third type. Hence there are vertices v_1, v_2, v_3 in G with $\delta(v_i)$ corresponding to $ins(B_i)$, for $i = 1, 2, 3$. But $E(G) \neq \delta(v_1) \cup \delta(v_2) \cup \delta(v_3)$ for any choice of v_1, v_2, v_3, so $\Sigma = ins(B_1) \cup ins(B_2) \cup ins(B_3)$ cannot hold. Hence ins is a pre-slope of order 3 in $\mathcal{R}$.

However there are three small sets $Z_1, Z_2, Z_3 \subseteq R(\mathcal{R})$ (with respect to ins) forming a partition (Z_1, Z_2, Z_3) of $R(\mathcal{R})$: Let $A_1, A_2 \subseteq E(G)$ be the edges of the two disjoint non-contractible cycles of length three in G, and $B_1, B_2, B_3 \subseteq E(G)$ be the edges of the three disjoint non-contractible 2-cycles. Let Z_1 be the regions corresponding to $A_1 \cup B_1$, Z_2 the regions corresponding to $A_2 \cup B_2$ and Z_3 be the regions corresponding to B_3 (see Figure 3).

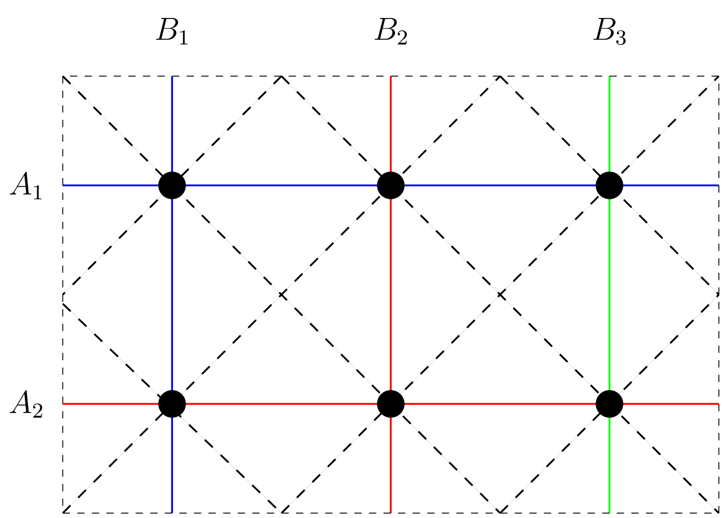


Figure 3: Decomposing the 2×3 grid

It is straightforward to check that Z_1, Z_2, Z_3 are small (in particular $\partial(Z_i)$ has weight ≤ 3 for $i = 1, 2, 3$), and that $Z_1 \cup Z_2 \cup Z_3 = R(\mathcal{R})$.

What goes 'wrong' in the above example is the following: If we have a graph G

embedded with high face-width (or in particular, a planar graph embedded in the sphere), then the optimal branch-decompositions have a simple structure: there is always some optimal branch-decomposition where every separation induced by the decomposition is such that (at least) one side forms a disc in the radial graph $\mathcal{R}$. This implies that it is enough to require that no three small sets bounded by single (contractible) cycles make up all of $R(\mathcal{R})$: If it is possible at all to have that $R(\mathcal{R})$ is the union of three small sets, then such a configuration indicates that G has a branch-decomposition $\mathcal{B}$ of order $< \theta$, and hence there is some branch-decomposition $\mathcal{B}^*$ of order $< \theta$ with the above property, which in turn would give a way to write $R(\mathcal{R})$ as the union of three small sets bounded by cycles.

However if we are not on the sphere, and the graph G embedded in Σ has face-width lower than $bw(G)$, then there need not be an optimal branch-decomposition with the above mentioned property. In the above example, one can check that *any* optimal branch-decomposition (of width 3) will contain some vertex in the tree where the three separations in G induced by it each contain a non-contractible cycle of length at least two in their 'small' side (for example, the partition $(A_1 \cup B_1) \cup (A_2 \cup B_2) \cup (A_3 \cup B_3)$ of $E(G)$ can be extended to an optimal branch-decomposition of order 3 of G).

More generally, in the radial graph $\mathcal{R}$, a non-contractible cycle of length $k \geq 2$ in an embedded graph G corresponds to a set Z (of k regions of the radial graph $\mathcal{R}$) which is bounded by a 'chain' of k edge-disjoint 4-cycles. In particular, the graph $\partial(Z)$ would have $k+1$ regions in this case, and so $\partial(Z)$ is not a border.

What all of this shows is that in general, we can not expect to have optimal branch-decompositions whose separations correspond just to single borders in the radial graph, and so in order to get the exact relationship with tangles (and hence branch-width), we need to strengthen the second axiom of a pre-slope to deal with structures like those arising from non-contractible cycles in G.

With this motivation in mind, we now define the notion of a *cluster*, which leads to the definition of a (generalized) slope.

Suppose B is a confined subgraph of G such that $B = B_0 \cup \ldots \cup B_{k-1}$, where $B_0, \ldots, B_{k-1}$ are the borders in B with $ins(B_i)$ maximal, for $i = 0, \ldots, k-1$. By Lemma 3.2.18, the union is edge-disjoint, and hence B is eulerian by Proposition 3.2.6.

Remark 3.3.4. Note that B has exactly $k+1$ regions: By Lemma 3.2.15, there is a unique region r_0 of B which is not a subset of $ins(B)$ (in fact $r_0 = \Sigma \backslash ins(B)$), and the other k regions are given by $ins(B_i) \setminus B_i$ for $i = 0, \ldots, k-1$, since $ins(B_i) \cap ins(B_j)$ contains no region for $i \neq j$, by Lemma 3.2.18.

We now define a bipartite multigraph $\mathcal{S}_B$, capturing the structure of B. Let $\mathcal{B} = \{B_0, \ldots, B_{k-1}\}$, and let $\mathcal{V}^{2+} = \{v \in V(B) : d_B(v) > 2\}$. For $v \in \mathcal{V}^{2+}$, let $e_0, e_1, \ldots, e_{2d-1}$ be the edges of B incident at v (in cyclic order). By the above remark, every e_i is incident with the region r_0, and some region corresponding to a maximal border B_i of B. Hence there are exactly d (not necessarily distinct) regions $B_0^v, \ldots, B_{d-1}^v$ incident at v which are distinct from r_0, say with $\{e_{2j}, e_{2j+1}\} \in E(B_j^v)$ for $j = 0, \ldots, d-1$ (see Figure 4).

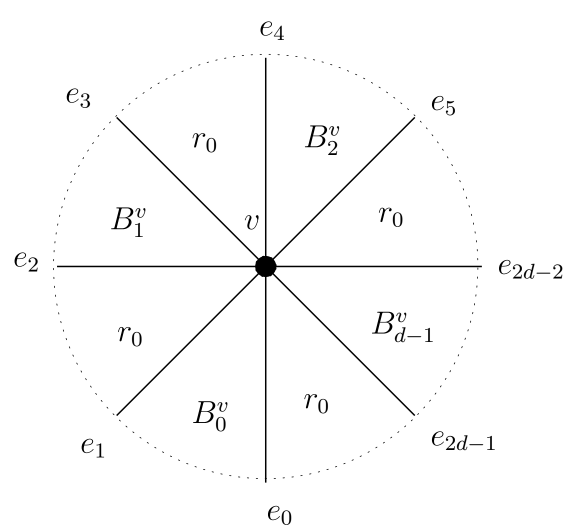


Figure 4: Construction of the structure graph $\mathcal{S}_B$

Definition 3.3.5. Suppose B is a confined subgraph of G such that B is the (edge-disjoint) union of the maximal borders $B_1, \ldots, B_k$ in B. Then the *structure graph* $\mathcal{S}_B$ of B is the (embedded) bipartite multigraph with bipartition $V(\mathcal{S}_B) = \mathcal{V}^{2+} \cup \mathcal{B}$, and $E(\mathcal{S}_B)$ defined as follows: For a vertex $v \in \mathcal{V}^{2+}$, if $\delta_B(v) = \{e_0, e_1, \ldots, e_{2d-1}\}$ are the edges incident at v in B, and $B_0^v, \ldots B_{d-1}^v \subseteq \mathcal{B}$ are as above, then $E(\mathcal{S}_B)$ contains edges vB_j^v for $j = 0, \ldots, d-1$.

Remark 3.3.6. Note that $\mathcal{V}^{2+}$ is allowed to be empty (for example if B consists of a single contractible cycle), in which case $V(\mathcal{S}_B)$ is an independent set.

Remark 3.3.7. If $v \in \mathcal{V}^{2+}$, then $d_B(v) = 2 \cdot d_{\mathcal{S}_B}(v)$.

Definition 3.3.8. Suppose C is a confined subgraph of G so that C is the (edge-disjoint) union of the maximal borders in C. Then C is called a *cluster* if the structure graph $\mathcal{S}_C$ is connected, and if $|V(\mathcal{S}_C)| > 1$, then $\mathcal{S}_C \setminus v$ is connected for every $v \in \mathcal{V}^{2+}$.

Note in particular that any border C is a cluster (in this case, $V(\mathcal{S}_C)$ has exactly one vertex in the class $\mathcal{B}$).

The structure graph was essentially defined in the proof of (5.4) in [63]. However in that setting, it turns out that $\mathcal{S}_B$ can not have cycles, i.e. $\mathcal{S}_B$ is a forest. This is no longer true in our setting, as the following example shows: Suppose C is the union of k light contractible cycles $C_1, \ldots C_k$ (where $ins(C_i)$ is a closed disc for each i), so that for $i = 1, \ldots, k$, $C_i \cap C_{i+1}$ (mod k) consists of a single vertex v_i (with $v_i \neq v_j$ for $i \neq j$), and C contains a non-contractible cycle. Then C is (typical example of) a cluster, and the structure graph $\mathcal{S}_C$ is a non-contractible cycle of length $2k$. For a slightly more general example, see Figure 5.

The following fact about maximal clusters will be useful later:

Lemma 3.3.9. *Let G be a confined graph, let C_1, C_2 be distinct clusters in G so that $ins(C_1)$ is maximal. Then $ins(C_1) \cap ins(C_2) = \emptyset$, or $ins(C_1) \cap ins(C_2) = \{v\}$ for some $v \in C_1 \cap C_2$.*

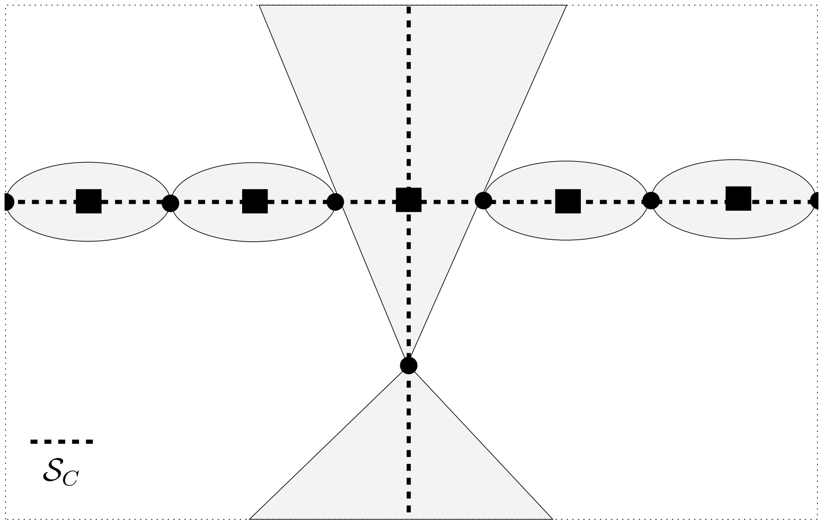


Figure 5: An example of a cluster C with its structure graph $\mathcal{S}_C$ on the torus

Proof. Let C_i be the edge-disjoint union of maximal borders $B_i^1, \ldots, B_i^{k_i}$ in G, let $\mathcal{S}_i$ be the structure graph of $B_i^1 \cup \ldots \cup B_i^{k_i}$, and let $\mathcal{B}_i = \{B_i^1, \ldots, B_i^{k_i}\}$ for $i = 1, 2$. Let $\mathcal{S}$ be the structure graph of $C_1 \cup C_2$. Then either $\mathcal{S}$ is not connected (i.e. has components $\mathcal{S}_1$, $\mathcal{S}_2$), or there is a unique cutvertex v in $\mathcal{S}$ which belongs to $V(C_1 \cup C_2)$ and has degree > 2 in $C_1 \cup C_2$ (for otherwise $C_1 \cup C_2$ would be a cluster contrary to the maximality of C_1). Clearly $v \in V(C_1) \cap V(C_2)$, since v is not a cutvertex for $\mathcal{S}_1$ or $\mathcal{S}_2$. Hence $\mathcal{S}_1 \cap \mathcal{S}_2 = \emptyset$ or $\mathcal{S}_1 \cap \mathcal{S}_2 = \{v\}$. In either case, $\mathcal{B}_1 \cap \mathcal{B}_2 = \emptyset$, i.e. no border belongs to C_1 and C_2, and so by Lemma 3.2.18, it follows that $ins(C_1) \cap ins(C_2) \subseteq V(C_1) \cap V(C_2)$ (since $ins(C_i) = ins(B_i^1) \cup \ldots \cup ins(B_i^{k_i}$ for $i = 1, 2$). But any vertex in $V(C_1) \cap V(C_2)$ must have degree > 2 in $C_1 \cup C_2$, and so $V(C_1) \cap V(C_2) \subseteq V(\mathcal{S}_1) \cap V(\mathcal{S}_2)$. Consequently $ins(C_1) \cap ins(C_2)$ is either empty or consists of $v \in V(C_1) \cap V(C_2)$, as claimed. $\square$

With the notion of a cluster, we can finally define the notion of a (generalized) slope.

Definition 3.3.10. Let $\theta \geq 1$ be an integer, and let $G \hookrightarrow \Sigma$. A *(generalized) slope* in G of order θ is a function ins which assigns to every light border B of G a spot $ins(B)$, bounded by B, such that

S1) if B_1, B_2 are light borders in G, and $U(B_1) \subseteq ins(B_2)$, then $ins(B_1) \subseteq ins(B_2)$,

S2) if (Z_1, Z_2, Z_3) is a partition of $R(G)$ so that $\partial(Z_i) = C_i$ for some light cluster C_i, for $i = 1, 2, 3$, then $ins(C_1) \cup ins(C_2) \cup ins(C_3) \neq \Sigma$.

Remark 3.3.11. Any slope is a pre-slope (of the same order).

It is easy to see that the second axiom is equivalent to requiring that in a partition (Z_1, Z_2, Z_3) where each Z_i is bounded by a light cluster, at least one of Z_1, Z_2, Z_3 is not small. Note that this fixes the problem in the previous example, since the partition into three small sets we constructed showed that there did not exist a generalized slope of order 4.

We are now ready to prove the key lemma for this section. The overall structure of the proof is the same as the one in [63], but some of the individual steps require modifications.

Lemma 3.3.12 ((5.4) in [63]). *If ins is a generalized slope and (Z_1, Z_2, Z_3) is a partition of $R(G)$, then at least one of Z_1, Z_2, Z_3 is not small.*

Proof. Suppose Z_1, Z_2, Z_3 form a counterexample with $\partial(Z_1, Z_2, Z_3)$ minimal, where $\partial(Z_1, Z_2, Z_3) = \partial(Z_1) \cup \partial(Z_2) \cup \partial(Z_3)$. Then all Z_i are small and moreover $Z_i \neq \emptyset$ for $i = 1, 2, 3$ by Lemma 3.3.1. We aim to show that $\partial(Z_1)$ is a cluster.

Claim 3.3.12.1. There is a cluster $C_1 \subseteq \partial(Z_1)$ such that at most one vertex of C_1 is incident with an edge of $\partial(Z_1)$ not in $ins(C_1)$.

Proof of claim. Let $\mathcal{C} = \{C_1, \ldots C_k\}$ be the set of all maximal clusters in $\partial(Z_1)$, i.e. the clusters with $ins(C_i)$ maximal. Note that every edge e of $\partial(Z_1)$ is contained in $ins(B)$ for some border B in $\partial(Z_1)$, since the region at e belonging to Z_1 is captured by $\partial(Z_1)$ as Z_1 is small. Hence every edge of $\partial(Z_1)$ is contained in $ins(C_i)$ for some cluster $C_i \in \mathcal{C}$.

Let $C = C_1 \cup \ldots \cup C_k$. Then C is the edge-disjoint union of the maximal borders in $\partial(Z_1)$, by Lemma 3.3.9.

Let $\mathcal{S}_C$ be the structure graph of C, and let $\mathcal{W}$ be the set of cutvertices of $\mathcal{S}_C$ that belong to $\mathcal{V}^{2+}$. Then Lemma 3.3.9 implies that $\mathcal{W}$ consists exactly of all vertices belonging to at least two distinct maximal clusters in $\mathcal{C}$. If $\mathcal{W}$ is empty, then C is a cluster, i.e. $k = 1$, and so by the above remarks $\partial(Z_1) \subseteq ins(C_1)$. Hence we may assume $\mathcal{W} \neq \emptyset$.

Construct a new bipartite graph $\mathcal{F}$ with bipartition classes $\mathcal{C}$ and $\mathcal{W}$, where we put an edge between a maximal cluster $C_i \in \mathcal{C}$ and a cutvertex $v \in \mathcal{W}$ whenever $v \in V(C_i)$. Then the maximality of the clusters in $\mathcal{C}$ implies that $\mathcal{F}$ has no cycles (if $\mathcal{F}$ had a cycle D, then the clusters in $D \cap \mathcal{C}$ would form a single larger cluster), and so $\mathcal{F}$ is a forest. As noted above, every $v \in \mathcal{W}$ has degree at least two in $\mathcal{F}$.

Without loss of generality suppose that the cluster C_1 is a leaf of $\mathcal{F}$. If $e = uv \in E(\partial(Z_1))$ is not in $ins(C_1)$, then e belongs to $ins(C')$ for some maximal cluster $C' \neq C_1$ in $\mathcal{C}$ by the initial remark. Suppose that $v \in V(C_1)$. Then $ins(C_1) \cap ins(C') = \{v\}$ and $v \in V(C_1) \cap V(C')$ by Lemma 3.3.9, and so $v \in \mathcal{W}$ as noted above. Hence v must be the unique neighbor of C_1 in $\mathcal{F}$, and so C_1 is as stated in the claim. $\square$

The remaining steps are now analogous to the ones from [63] (again using clusters and borders instead of cycles), but we include them for the sake of readability.

Let C_1 be as in Claim 3.3.12.1, and pick a vertex $v_1 \in V(C_1)$ such that no other $v \in V(C_1)$ is incident with an edge of $\partial(Z_1)$ that is not in $ins(C_1)$. Let

$$A = \{r \in R(G) : r \subseteq ins(C_1))\} \tag{3.3.1}$$

be the set of all regions captured by C_1, and let $A^c = R(G) \setminus A$ be all other regions. The strategy is to first show that A includes only regions of Z_1 (since otherwise we will be able to add all of A to either Z_2 or Z_3, while maintaining a partition into small sets), and later that that $A = Z_1$, i.e. Z_1 is captured by a single cluster.

Claim 3.3.12.2. If $e \in E(C_1)$, then one region incident with e belongs to $A \cap Z_1$, and the other one belongs to $A^c \cap (Z_2 \cup Z_3)$.

Proof of claim. Let $e \in E(C_1)$. Then e is incident with a region r in A, and a region s in A^c (since $ins(C_1)$ is bounded by C_1). Let B be the (unique) border in C_1 which contains e (and hence captures r, but not s). Since $B \subseteq C_1$ is a maximal border in the confined graph $\partial(Z_1)$, Lemma 3.2.14 implies that s is not captured by $\partial(Z_1)$, and so $s \in R(G) \setminus Z_1 = Z_2 \cup Z_3$. Hence $e \in E(C_1) \subseteq E(\partial(Z_1))$ means that $r \in Z_1$. $\square$

For $v \in V(C_1)$, let $R(v)$ denote the regions incident with v which are not in A. Note that by Claim 3.3.12.2, $R(v) \subseteq Z_2 \cup Z_3$ for all $v \neq v_1$.

For $i = 2, 3$, let $W_i = \{v \in V(C_1) : v \neq v_1, \quad R(v) \subseteq Z_i\}$ (i.e. vertices in W_i are only incident with regions of Z_i, and regions captured by $ins(C_1)$).

Fix an edge $f \in E(C_1)$ which is incident at v_1. By Claim 3.3.12.2, $f \in E(\partial(Z_1) \cap \partial(Z_2))$ or $f \in E(\partial(Z_1) \cap \partial(Z_3))$. Since so far we had symmetry between Z_2 and Z_3, we may assume that $w(W_2) \geq w(W_3)$, and if equality holds, we may assume that $f \in \partial(Z_2)$.

Claim 3.3.12.3. $E(C_1 \cap \partial(Z_2)) \neq \emptyset$.

Proof of claim. Either $W_2 \neq \emptyset$, or $|W_2| = 0$ and so $|W_3| = |W_2| = 0$, in which case the above edge f is as desired. $\square$

For $i = 1, 2, 3$, let $Z'_i = Z_i \cap A^c$ be the regions of Z_i not captured by C_1.

Claim 3.3.12.4. For $i = 1, 2, 3$, $\partial(Z'_i) \subseteq \partial(Z_i)$, and Z'_i is small.

Proof of claim. Let $e \in \partial(Z'_i)$ be incident with regions $r \in Z'_i$ and $s \notin Z'_i$. Then $r \in Z_i$ and r is not captured by C_1, and s is either captured by C_1, or belongs to Z_i^c (or both). Suppose for a contradiction that $e \notin \partial(Z_i)$, i.e. by the previous statement s belongs to Z_i and is captured by C_1. Then $e \in C_1$, and by Claim 3.3.12.2 $s \in Z_1$, and so $r \notin Z_1$. In particular, $i \neq 1$ since $r \in Z_i$, and so $s \notin Z_i$ (since $s \in Z_1$), implying that $e \in \partial(Z_i)$ after all.

To see that Z_i' is small, note that $Z_i' \subseteq Z_i$, Z_i is small, and that $w(\partial(Z_i')) \leq w(\partial(Z_i))$ by the inclusion we just proved, and so Z_i' is small by Lemma 3.3.2. □

The next step is to show that moving the regions of A to Z_2 does not increase the weight of the boundary.

Claim 3.3.12.5. $w(\partial(Z_2 \cup A)) \leq w(\partial(Z_2))$.

Proof of claim. Define

$$X = V(\partial(Z_2)),$$

$$Y = V(\partial(Z_2 \cup A)),$$

$$D = \{v_1\} \setminus X.$$

In particular $|D| \in \{0, 1\}$. We have to show $w(X) - w(Y) \geq 0$, i.e. that $w(X \setminus Y) - w(Y \setminus X) \geq 0$.

Note that if $v \in W_2$, then $v \in V(C_1)$ is incident only with regions in A and Z_2 (and in particular with a region in Z_1 and one in Z_2), and so $W_2 \subseteq X \setminus Y$.

Similarly let $v \in Y \setminus X$. Since $\partial(Z_2 \cup A) \subseteq \partial(Z_2) \cup \partial(A)$, we have that $v \in V(\partial(A)) = V(C_1)$. Since $v \notin X$, but $v \in V(C_1)$ we must have that no regions at v are in Z_2, by Claim 3.3.12.2. It follows that $v \in W_3 \cup D$, i.e. we have $Y \setminus X \subseteq W_3 \cup D$.

Hence we obtain that $w(X \setminus Y) - w(Y \setminus X) \geq w(W_2) - w(W_3) - |D|$. But if $w(W_2) = w(W_3)$, then the special edge f at v_1 belongs to $E(\partial(Z_2))$ (as assumed before Claim 3.3.12.3), and so $|D| = 0$. So in either case, $w(X \setminus Y) - w(Y \setminus X) \geq 0$, as desired. □

For a partition (X, Y, Z) of $R(G)$, let $\partial(X, Y, Z)$ denote the graph $\partial(X) \cup \partial(Y) \cup \partial(Z)$. (The proof of the second part of the following claim is slightly different from [63]).

Claim 3.3.12.6. $E(\partial(Z_1)) \cap E(\partial(Z_3)) \subseteq E(C_1)$, and $A \subseteq Z_1$.

Proof of claim. $(Z_1', Z_2 \cup A, Z_3')$ is a partition of $R(G)$, and Z_1', Z_3' are small as shown in Claim 3.3.12.4. Note that $\partial(Z_1', Z_2 \cup A, Z_3') \subseteq \partial(Z_1, Z_2, Z_3)$, and in fact the inclusion is proper because by Claim 3.3.12.3, there is an edge e on C_1 in $\partial(Z_2)$, and by Claim 3.3.12.2, e is incident with a region in $A \cap Z_1$ and one in $A^c \cap Z_2$, so e is not in $\partial(Z_1', Z_2 \cup A, Z_3')$. Hence the minimality of $\partial(Z_1, Z_2, Z_3)$ (together with Claim 3.3.12.4) implies that $Z_2 \cup A$ is not small. But $\partial(Z_2 \cup A)$ is light by Claim 3.3.12.5, and so $(Z_2 \cup A)^c = Z_1' \cup Z_3'$ is small by Lemma 3.3.1.

Hence $(A, Z_2', Z_1' \cup Z_3')$ is a partition into small sets. It is easily checked that $\partial(A, Z_2', Z_1' \cup Z_3')$ is a subgraph of $\partial(Z_1, Z_2, Z_3)$ (using Claim 3.3.12.2), and so by minimality we have equality. Now let $e \in E(\partial(Z_1)) \cap E(\partial(Z_3))$ be a $Z_1 - Z_3$ edge. Then the previous statement implies that e must be an $A - (Z_1' \cup Z_3')$ edge, so in particular e belongs to $\partial(A) = C_1$, as desired.

For the second part of the claim, suppose there is a region $r \in A \setminus Z_1$, i.e. r is captured by some border $B \subseteq C_1$, but $r \in Z_2 \cup Z_3$. Clearly $ins(B)$ contains a region in $r' \in Z_1 \cap A$ by Claim 3.3.12.2, and since $ins(B) \setminus U(B)$ is a region of C_1 (C_1 is the edge-disjoint union of its maximal borders), Proposition 3.2.17 implies we may assume that there is an edge $e \notin E(C_1)$ which is incident with r and r'. Hence e belongs to $\partial(Z_1) \subseteq \partial(Z_1, Z_2, Z_3)$, but the fact that both regions at e belong to A means that e is not an edge of $\partial(A, Z_2', Z_1' \cup Z_3')$, a contradiction since we showed that these two boundary graphs are identical. ☐

Claim 3.3.12.7. $Z_1 = A$.

Proof of claim. Suppose not. By Claim 3.3.12.6, $\partial(Z_1)$ consists not just of the cluster C_1, i.e. it has at least two clusters. Hence we can pick a cluster $C_1' \neq C_1$ of $\partial(Z_1)$ which is also a leaf in the forest $\mathcal{F}$ defined in the proof of Claim 3.3.12.1. By Claim 3.3.12.6 applied to C_1', there exists $j \in \{2, 3\}$ such that $E(\partial(Z_1)) \cap E(\partial(Z_j)) \subseteq E(C_1')$

(note that we broke the symmetry between Z_2 and Z_3 with respect to Z_1 to get Claim 3.3.12.6).

But by Claim 3.3.12.3 and Claim 3.3.12.2, C_1 already contains an edge in $E(\partial(Z_1)) \cap E(\partial(Z_2))$, so $j \neq 2$ since C_1 and C_1' are edge-disjoint by Lemma 3.3.9.

Therefore $j = 3$, and so Claim 3.3.12.6 implies that in fact $E(\partial(Z_1)) \cap E(\partial(Z_3)) = \emptyset$. Hence there are no $Z_1 - Z_3$ edges in $\partial(Z_1, Z_2, Z_3)$, i.e. $\partial(Z_1, Z_2, Z_3) = \partial(Z_2)$, and so every region of $R(G)$ is captured by $\partial(Z_2)$, contrary to Lemma 3.2.15. □

By Claim 3.3.12.7, $\partial(Z_1) = \partial(A) = C_1$, i.e. Z_1 is bounded by a single cluster. Similarly Z_2 and Z_3 are bounded by single clusters, and since (Z_1, Z_2, Z_3) is a partition of $R(G)$, this contradicts the second slope axiom and completes the proof of the lemma. □

The final step is to extend the previous result to the case where the Z_i may have nonempty intersection- this is done by a simple uncrossing argument which is identical to the one in [63], so we will not repeat it here.

Theorem 3.3.13 ((5.5) in [63]). *Let $G \hookrightarrow \Sigma$, let ins be a slope of order θ in G with respect to $\Omega \subseteq V(G)$. If $Z_1, Z_2, Z_3 \subseteq R(G)$ are small, then $Z_1 \cup Z_2 \cup Z_3 \neq R(G)$.*

3.4 Tangles and slopes

We are now ready to prove the aforementioned result about the correspondence of tangles and slopes. For this we will also use the following easy fact from [63]:

Proposition 3.4.1. *Let G be a graph, and $\mathcal{T}$ be a tangle of order $\theta \geq 1$. If $(A, B) \in \mathcal{T}$, $(A', B') \in \mathcal{T}$ and $(A \cup A', B \cap B')$ has order $< \theta$, then $(A \cup A', B \cap B') \in \mathcal{T}$.*

Given a tangle it is always possible to construct a slope of the same order in the radial graph, but for the reverse direction we need to impose a mild extra condition to avoid the 'trivial' slopes that are mentioned in the introduction of [63]. For a region r bounded by a closed walk $W = v_1 e_1 v_2 \dots e_{k-1} v_k$, define the *perimeter weight* of r

to be $\sum_{i=1}^{k} \mathbb{1}_\Omega(v_i)$. A slope is called *even* if for every region $r \in R(G)$ of perimeter weight $< \theta$, there is a border B in G of weight $< \theta$ with $r \subseteq ins(B)$.

Theorem 3.4.2. *Let $G \hookrightarrow \Sigma$ be 2-cell embedded and $\mathcal{R}$ be its radial graph. Let $\theta \geq 1$ be an integer. Then there is a tangle of order θ in G if and only if there is an even slope in $\mathcal{R}$ of order θ and with respect to $\Omega = V(G)$.*

Proof. Suppose there is a tangle $\mathcal{T}$ of order θ in G. Set $\Omega \subseteq V(\mathcal{R})$ to be $V(G)$, and let B be a border in $\mathcal{R}$ of weight $k < \theta$. Then the two spots bounded by B partition the regions of $\mathcal{R}$ into two sets Z_C and Z_D, corresponding to two complementary separations (C, D) and (D, C) in G. A vertex $v \in V(G)$ is incident with edges of both C and D if and only if, in $\mathcal{R}$, it is incident with a region from both Z_C and Z_D, i.e. if and only if $v \in V(B) \cap V(G)$. Hence the order of (C, D) is equal to k, and by T1), exactly one of $(C, D), (D, C)$ belongs to $\mathcal{T}$, say (C, D). In that case, we define $ins(B)$ to be the spot corresponding to Z_C, and we claim that this defines a slope in $\mathcal{R}$.

It is easy to see that S1) holds: Suppose $U(B') \subseteq ins(B)$ for some light borders B, B'. If $ins(B') \nsubseteq ins(B)$ then $ins(B) \cup ins(B') = \Sigma$ and so every region of $\mathcal{R}$ is contained in at least one of $ins(B), ins(B')$. But by our definition of ins, we get two separations (C_B, D_B) and $(C_{B'}, D_{B'})$ in $\mathcal{T}$ with $C_B \cup C_{B'} = G$, a contradiction to T2).

It is easy to check from our definition of $ins(B)$ that ins defines a pre-slope. We now show that in fact ins defines an even slope.

For a small set $Z \subseteq R(\mathcal{R})$ of weight $k < \theta$, and the separation (C_Z, D_Z) in G *associated* to it is the separation of order k with $E(C)$ corresponding to Z.

Claim 3.4.2.1. Let Z be a small set. Then $(C_Z, D_Z) \in \mathcal{T}$.

Proof of claim. Let $B_1, \ldots, B_k$ be the maximal borders in $\partial(Z)$, and let Z_i be the regions in $ins(B_i)$, for $i = 1, \ldots k$. Since Z is small, we have $Z \subseteq Z_1 \cup \ldots \cup Z_k$.

Now suppose Z is a counterexample with k minimum, and subject to that, choose $|Z|$ maximum. Then it follows that $Z = Z_1 \cup \ldots \cup Z_k$, and Lemma 3.2.18 implies that $\partial(Z) = B_1 \cup \ldots \cup B_k$. Since Z is a counterexample, we have $k \geq 2$ (for otherwise $k = 1$, i.e. $\partial(Z) = B_1$ is a single border and the claim holds by construction of the function ins). Now Z_1 and $Z' = Z_2 \cup \ldots \cup Z_k$ are small by Lemma 3.3.2 (note that $\partial(Z') = B_2 \cup \ldots \cup B_k$), and so $(C_{Z_1}, D_{Z_1}) \in \mathcal{T}$ and $(C_{Z'}, D_{Z'}) \in \mathcal{T}$ by minimality of Z. Since $Z = Z_1 \cup Z'$ and $\partial(Z)$ has weight $< \theta$, applying Proposition 3.4.1 yields that $(C_Z, D_Z) \in \mathcal{T}$, so Z was not a counterexample after all. $\square$

The above claim readily implies that S2) holds: If (Z_1, Z_2, Z_3) is a partition of $R(\mathcal{R})$ (so that each $\partial(Z_i)$ is a cluster), then not all three can be small by the above claim and tangle axiom T2).

It remains to show that the slope ins is even: Let r_e be a region of $R(\mathcal{R})$ (corresponding to an edge $e \in E(G)$) with perimeter weight $< \theta$, so that in particular $\partial(r_e)$ has weight $< \theta$. Let e have ends v_1, v_2 in G, and be incident with regions r_1, r_2. Then in $\mathcal{R}$, r_e is bounded by the contractible walk $W = v_1 r_1 v_2 r_2$. Then W is a 4-cycle, or the (edge-disjoint) union of two (contractible) cycles of length 2, i.e. $W = \partial(r_e)$ is the union of two light borders B_1 and B_2 (not necessarily distinct).

If $r_e \subseteq ins(B_i)$ for $i = 1$ or $i = 2$, we are done, so assume not. It follows that $\Sigma = ins(B_1) \cup ins(B_2) \cup r_e$. Let Z_i be the regions of $R(\mathcal{R})$ captured by B_i, for $i = 1, 2$. Then the associated separations (C_{Z_1}, D_{Z_1}), (C_{Z_2}, D_{Z_2}) belong to $\mathcal{T}$ by definition of ins, and by result (2.7) from [62], the separation $(e, G \setminus e) \in \mathcal{T}$ since $\mathcal{T}$ is a tangle and e has $< \theta$ ends by assumption. But $E(C_{Z_1} \cup C_{Z_2}) = G \setminus e$, contrary to axiom T2). Hence r_e is captured by some light border in $\mathcal{R}$, and ins is even as desired.

Conversely, suppose ins is an even slope of order θ in $\mathcal{R}$. Let (C, D) be a separation of order $k < \theta$ in G, and let Z_C, Z_D be the induced (complementary) sets of regions of $\mathcal{R}$. As above, we have that $\partial(Z_C)$ has weight k, and so by Lemma 3.3.1, exactly one of Z_C, Z_D is small, say Z_C. In that case we put $(C, D) \in \mathcal{T}$ (as opposed to (D, C)), and we claim that $\mathcal{T}$ is a tangle.

Clearly T1) holds, and T2) holds by Theorem 3.3.13. It remains to verify T3). By result (2.7) of [62], it suffices to check that $(e, G \setminus e) \in \mathcal{T}$ for every $e \in E(G)$ with $< \theta$ ends. Let r_e be the region of $\mathcal{R}$ corresponding to such an edge e. Let $W = \partial(r_e)$ as above. In particular, the perimeter weight of W equals the number of ends of e, which is $< \theta$ by assumption. As ins is even, there is a light border B with $r_e \subseteq ins(B)$. Let Z be the set of regions captured by B. Then $\{r_e\} \subseteq Z$, Z is small and $w(\partial(r_e)) < \theta$, so $\{r_e\}$ is small by Lemma 3.3.2. But by construction of $\mathcal{T}$ we have $(e, G \setminus e) \in \mathcal{T}$, so T3) holds and $\mathcal{T}$ is a tangle. This completes the proof of Theorem 3.4.2. $\square$

3.5 Restraints and the capturing theorem

The second main result of [63] employs the concept of a restraint. Given an embedded graph G and a traditional slope of order θ, a *restraint* (of length $k < 2\theta$) in [63] is a set $X \subseteq \Sigma$ which is bounded by a closed contractible walk W of length $k < 2\theta$ so that $ins(W) = X$ (the original definition is slightly more topological).

Now suppose that the graph G is bipartite, let Ω be one of the bipartition classes and suppose ins is a (generalized) slope of order θ, i.e. ins assigns an inside to every border of Ω-weight $< \theta$. Then we can define the notion of a restraint as before, i.e. $X \subseteq \Sigma$ is a restraint if X is bounded by a closed contractible walk W of length $|W| < 2\theta$ and $ins(W) = X$. Note that since G is bipartite, $|W| < 2\theta$ implies that W has Ω-weight at most $|W|/2 < \theta$ (multiple vertices are only counted once for the Ω-weight) and so W is confined, ensuring that $ins(W)$ is defined.

For a fixed point $x \in \Sigma$ and an integer $k \leq 2\theta$, let $\mathcal{C}_x^k$ be the set of points $y \in \Sigma$ which are captured from x, i.e. all points y for which there is a restraint X of length $< k$ containing x and y. The following theorem is a reformulation of (8.12) from [63]:

Theorem 3.5.1. *Let G be a 2-cell embedded graph in Σ with a traditional slope of order θ. Then for every $x \in \Sigma$ and every integer $k \leq 2\theta$, $\Sigma \setminus \mathcal{C}_x^k$ contains a region of*

G.

We call this result the 'capturing theorem': It says that in a graph with a traditional slope of order 2θ, we can not 'capture' all of Σ (with restraints) from any fixed point $x \in \Sigma$. This theorem turns out to be the foundation of the 'ratcatcher' method for computing the branch-width of a planar graph in polynomial time [67]. Very roughly speaking, in the ratcatcher game, if the ratcatcher is at a point x in a graph with branch-width at least θ, then the 'noisy' areas of the sphere correspond to the areas captured by restraints containing x , and the above theorem guarantees that there is always some 'quiet' area for the rat to avoid capture.

However it turns out that if we replace traditional by generalized slopes, the analogous statement of the above theorem no longer holds, as we show in Theorem 3.5.4, the main result of this section.

We start by a lemma about the $k \times k$ toroidal grid $G = G_{k\times k}$, with rows $i_1, \ldots, i_k$ and columns $j_1, \ldots, j_k$ (we will use the the symbols i and j to denote both the index and the subgraph given by a row i or column j). Let $Row(G)$ be the set of rows, and $Col(G)$ be the set of columns of G. Let (A_1, A_2) be a separation in G. We say a row i is *full* if all edges in i belong to $E(A_1)$, or all belong to $E(A_2)$. A row is *mixed* if it is not full. Full and mixed columns are defined similarly.

Remark 3.5.2. A mixed row or column contains at least two elements of $V(A_1 \cap A_2)$.

A *full cross* in A_d is a subgraph of G consisting of a full row and a full column contained in A_d, for $d = 1, 2$. We denote by r_d the number of full rows in A_d, and by c_d the number of full columns in A_d, for $d = 1, 2$.

Lemma 3.5.3. *Let $G \hookrightarrow \mathbb{S}_1$ be the $k \times k$ toroidal grid, for $k \geq 2$, and let (A_1, A_2) be a separation of order at most $2k - 3$ in G. Then there is a unique $d_0 \in \{1, 2\}$ so that $r_{d_0} + c_{d_0} \leq 1$, i.e. A_{d_0} contains at most one full row or column, but not both.*

Proof. We first show that there is at least one side A_{d_0} as described. The following claim shows that not all of r_1, r_2, c_1, c_2 can be positive.

Claim 3.5.3.1. One of A_1, A_2 does not contain a full cross.

Proof of claim. Suppose not, i.e. A_d contains a full row i_d and a full column j_d, for $d = 1, 2$. For a row or column l, let $x(l) = |V(A_1 \cap A_2) \cap V(l)|$ denote the contribution of l to $V(A_1 \cap A_2)$. Define

$$I = \{i \in Row(G) \mid i \neq i_1, i_2 \text{ and } x(i) = 1\},$$

$$J = \{j \in Col(G) \mid j \neq j_1, j_2 \text{ and } x(j) = 1\}.$$

Without loss of generality assume that $|J| \geq |I|$. Note that if $j \in J$ (in particular $j \neq j_1, j_2$), then the unique vertex in $V(A_1 \cap A_2)$ on j is either (i_1, j) or (i_2, j), i.e. belongs to i_1 or i_2, depending on whether column j is contained in A_1 or A_2. In particular, this implies that $|J| + 2 \leq x(i_1) + x(i_2)$, and so we have

$$\begin{aligned}
|V(A_1 \cap A_2)| &= \sum_{\substack{i \notin I \\ i \neq i_1, i_2}} x(i) + \sum_{i \in I} x(i) + x(i_1) + x(i_2) \\
&\geq 2(k - |I| - 2) + |I| + x(i_1) + x(i_2) \\
&\geq 2(k - |I| - 2) + |I| + 2 + |J| \\
&= 2k - 2 - |I| + |J|.
\end{aligned}$$

Hence by our assumption that $|J| \geq |I|$, we have $|V(A_1 \cap A_2)| \geq 2k - 2$, a contradiction. ☐

By the above claim, there exists a side A_{d_0} with no full rows or no full columns, for some $d_0 \in \{1, 2\}$. Without loss of generality assume A_{d_0} has no full rows. Therefore we may assume that A_{d_0} has at least two full columns, since otherwise $r_{A_{d_0}} + c_{A_{d_0}} \leq 1$ as desired. But any full row of the other side of the separation induced by A_{d_0} contains at least two vertices of $V(A_1 \cap A_2)$, so since A_{d_0} contains no full rows, each row in $Row(G)$ contributes at least two to $|V(A_1 \cap A_2)|$, i.e. $|V(A_1 \cap A_2)| \geq 2k$, a contradiction. Hence there exists a side A_{d_0} with $r_{A_{d_0}} + c_{A_{d_0}} \leq 1$.

It is easy to see that such a side A_{d_0} is unique: If (A_1, A_2) has order $\leq 2k-3$, then there are at most $(2k-3)/2 < k-1$ mixed rows (by Remark 3.5.2), i.e. at least two rows are full, and similarly there are at least two full columns. Hence $r_{A_1} + r_{A_2} + c_{A_1} + c_{A_2}$ is always at least 4, and so not both A_1, A_2 can be as in the lemma. $\square$

Note that Lemma 3.5.3 is tight: A separation in $G_{k \times k}$ where one side consists of a single cross has order $2k-2$, and both sides contain a full row and a full column. Also clearly Lemma 3.5.3 is false for the planar $k \times k$ grid, since for example there are separations of order k where both sides contain $\lfloor k/2 \rfloor$ full rows, or $\lfloor k/2 \rfloor$ full columns (take 'half' of the grid).

Theorem 3.5.4. *For every $0 < \epsilon < \frac{2}{\sqrt{3}} - 1$, there is an integer $\theta_0 \geq 0$ so that the following holds: For every integer $\theta \geq \theta_0$ there is a graph $G = G(\theta)$ 2-cell embedded in $\Sigma = \mathbb{S}_1$ so that G has a slope of order $(1+\epsilon)\theta$, but $C_x^{2\theta} = \Sigma$ for every $x \in \Sigma$.*

Proof. The graph in question will be the radial graph of a toroidal grid of appropriate size. Let G_k be the $k \times k$ toroidal grid for $k \geq 2$, and let $1 < \rho < \frac{2}{\sqrt{3}}$ be irrational.

Claim 3.5.4.1. There is an integer k_0 so that for every $k \geq k_0$, G_k contains a tangle of order $\lceil \rho k \rceil$.

Proof of claim. We will construct such a tangle $\mathcal{T}$ as follows: Let (A, B) be a separation of order $\leq \rho k$, i.e. of order $\leq \lceil \rho k \rceil - 1$ since ρ is irrational. In particular $\rho k < 2k - 2$, for $k \geq 3$, and so exactly one of A, B satisfies the condition of Lemma 3.5.3. If $r_A + c_A \leq 1$ we put $(A, B) \in \mathcal{T}$, and $(B, A) \in \mathcal{T}$ otherwise. Then axiom T1) holds, and it clearly T3) holds.

We now verify that T2) holds as well. Let $(A_i, B_i) \in \mathcal{T}$ for $i = 1, 2, 3$. We have to show that $A_1 \cup A_2 \cup A_3 \neq G_k$. Let $A = A_i$, for a fixed $i \in \{1, 2, 3\}$, and let A^c be the complementary side of the separation (A, A^c). Define

$$
\begin{aligned}
m_r &= |\{i \mid \text{row } i \text{ is mixed }\}|, \\
a_r &= |\{i \mid \text{row } i \text{ contains at least one edge of } A\ \}|, \\
f_r &= |\{i \mid \text{row } i \text{ is full }\}|,
\end{aligned}
$$

so that $a_r = m_r + f_r$, and we define m_c, a_c analogously for columns. We have $a_c = m_r + f_r$. Note that $m_r, m_c \leq \frac{\rho k}{2}$, since every mixed row or column contributes at least two to $|V(A \cap B)|$, and that $f_r + f_c \leq 1$ by construction of the tangle $\mathcal{T}$. Also if $v = (i, j) \in V(G_k)$ is only incident with edges in A (i.e. $\delta(v) \subseteq E(A)$), then row i and column j contain an edge in A (in fact at least two), and so the number of such vertices is bounded by $a_r \cdot a_c$. Hence we obtain:

$$
\begin{aligned}
|\{v \in V(G_k) \mid \delta(v) \cap E(A) \neq \emptyset\}| &= |V(A \cap B)| + |\{v \in V(G_k) \mid \delta(v) \subseteq E(A)\}| \\
&\leq \rho k + (m_r + f_r)(m_c + f_c) \\
&\leq \frac{\rho^2}{4} k^2 + \Big(\frac{f_r + f_c}{2} + 1\Big) \rho k + f_r f_c \\
&\leq \frac{\rho^2}{4} k^2 + \frac{3\rho}{2} k + 1.
\end{aligned}
$$

Since $\rho < \frac{2}{\sqrt{3}}$, we can choose an integer k_0^A so that for all $k \geq k_0^A$,

$$
|\{v \in V(G_k) \mid \delta(v) \cap A \neq \emptyset\}| < \frac{k^2}{3} = \frac{n}{3},
$$

where $n = |V(G_k)|$. If we choose such integers for each of A_1, A_2, A_3 and let k_0 denote their maximum, then we get that some $v_0 \in V(G_k)$ is not incident with any edge in $E(A_1) \cup E(A_2) \cup E(A_3)$, i.e. $v_0 \notin V(A_1 \cup A_2 \cup A_3)$. Hence T2) holds and $\mathcal{T}$ is a tangle of order $\lceil \rho k \rceil$, as claimed. □

Hence by Theorem 3.4.2, the radial graph $\mathcal{R}_k = \mathcal{R}(G_k)$ of G_k has a slope of order $\lceil \rho k \rceil$, for $k \geq k_0$.

Conversely, we claim that if $x \in \Sigma$, and $y \notin \Sigma$, then in $\mathcal{R}_k$ there is a restraint of length $\leq 2k + 4 = 2(k + 2)$ capturing x and y (note that $k + 2 < \rho k$ for k sufficiently large): It suffices to consider the case when x and y are midpoints of edges of G_k, i.e. x and y are in the interior of regions of $\mathcal{R}_k$ (if x is in the interior of a region r_e in $\mathcal{R}_k$, then any restraint capturing x also captures $cl(r_e)$). Fixing arbitrary edges $e, f \in E(G_k)$, it is easy to check that their midpoints $x(e)$ and $x(f)$ are captured by a common restraint of length $\leq 2k+4$ in $\mathcal{R}_k$: Given a fixed e, for most choices of f one can pick a contractible walk (in $\mathcal{R}_k$) consisting of the two four-cycles bounding the regions r_e, r_f, together with a shortest path of length $\leq k-1$ joining them (the path is traversed twice in the walk), and if that walk has length $2k+6$ (i.e. the shortest path has length exactly $k-1$), then one can take two such shortest paths which are disjoint and whose endpoints on each of the two four-cycles are incident, and thus we obtain a restraint consisting of a contractible cycle of length $2k+4$ (see Figure 6).

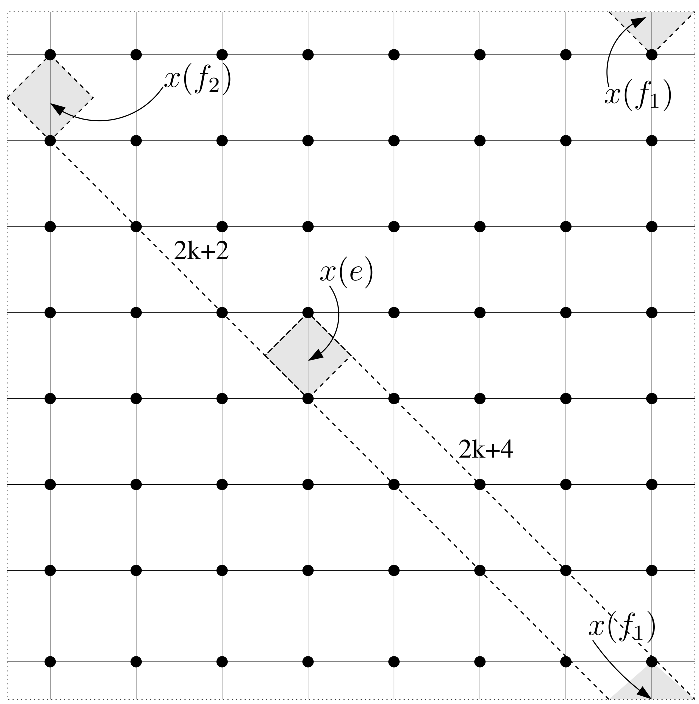


Figure 6: Capturing points $x(f_1)$, $x(f_2)$ from $x(e)$ for the $k \times k$ grid with $k = 8$

Hence any $x, y \in \Sigma$ are captured by a restraint of length $\leq 2(k+2)$, implying that $C_x^{2(k+3)} = \Sigma$, for any $k \geq 2$.

Let $k \geq 2$, let $\theta = k+3$, and let $\epsilon' = \rho - 1$, so that $0 < \epsilon' < \frac{2}{\sqrt{3}} - 1$. Then G_k has a slope of order at least $\rho k = \rho\theta - 3\rho = (1+\epsilon')\theta - 3(1+\epsilon')$ for all $\theta \geq 5$.

Now suppose $0 < \epsilon < \frac{2}{\sqrt{3}} - 1$ is fixed. Choose ϵ' so that $\epsilon < \epsilon' < \frac{2}{\sqrt{3}} - 1$. Then for some integer θ'_0, $(1+\epsilon)\theta \leq (1+\epsilon')\theta - 3(1+\epsilon')$ for all $\theta \geq \theta'_0$, and it follows that for $\theta_0 = \max(\theta'_0, 5)$, the radial graph of the $(\theta-3)\times(\theta-3)$ toroidal grid is as desired, for every $\theta \geq \theta_0$. This proves Theorem 3.5.4. □

In the proof of Theorem 3.5.4 we show that if k is sufficiently large, then $bw(G_{k\times k}) \geq ck$, for some constant $c \approx 1.15$. In fact one can verify that already for small values of k, $bw(G_{k\times k}) > k$, in contrast to the fact that the planar $k \times k$ grid has branch-width exactly k. For instance, the following example shows that $bw(G_{3\times 3}) = 4$:

Example 3.5.5. It is easy to see that $G = bw(G_{3\times 3}) \leq 4$, for example by noting that G can not have a tangle of order 5: Define a *cross* $X(i,j)$ in G to be the subgraph consisting of row i and column j, for $1 \leq i,j \leq 3$. Let $(X(i,j),Y)$ be the separation induced by a cross (i.e. Y has edges $E(G)\setminus E(X(i,j))$). Then $(X(i,j),Y)$ has order 4, and it is easy to see that in a tangle of order at least 5, the small side must be the cross (i.e. $(X(i,j),Y)$ is in the tangle). But $X(1,1)\cup X(2,2)\cup X(3,3) = E(G)$, so we can not construct a tangle of order 5 or higher in G, i.e. $bw(G) \leq 4$.

Conversely, for every separation of order (A,B) of order $\leq 3 = 2k-3$, there is a unique side which contains at most one full row or column, but not both, by Lemma 3.5.3. We then construct a tangle $\mathcal{T}$ of order 4 by putting (A,B) into $\mathcal{T}$ if $r_A + c_A \leq 1$, and $(B,A) \in \mathcal{T}$ otherwise. Clearly T1) and T3) hold. If (A,B) has order ≤ 3 and A satisfies $r_A + c_A \leq 1$, then it is not hard to check that $|E(A)| \leq 4$ (consider the cases $r_A = c_A = 0$ and say $r_A = 1$, $c_A = 0$), and since $|E(G)| = 18\ 3 \cdot 4$, T3) holds as well. Hence $bw(G_{3\times 3}) = 4$, as claimed.

It is also not hard to see that $bw(G_{k\times k}) < 2k$ (for $k \geq 2$), and it would be interesting to determine $bw(G_{k\times k})$ exactly for any k (clearly the argument in Theorem 3.5.4 is not tight).

Another interesting family of graphs in this context is given by the $k \times l$ toroidal grids for $l \geq 2k$: It is not too hard to see that $bw(G_{k\times l}) = 2k$ for those graphs, but one would need restraints of order approximately $2l$ (in the radial graph) to capture all of Σ from a fixed point x. Hence if $l \gg 2k$, then we need restraints much higher than $2bw(G_{k\times l})$ to capture all of Σ.

3.6 Carvings and branch-decompositions

Recall that for a planar graph G, Seymour and Thomas proved that $2bw(G) = cw(\mathcal{M}(G))$ (for the precise result see Theorem 2.3.10), where $\mathcal{M}(G)$ denotes the medial graph of G (on the sphere).

In this section we prove a result similar to Theorem 2.3.10, for graphs embedded on arbitrary surfaces:

Theorem 3.6.1. *For every surface Σ there is a non-negative constant $c(\Sigma)$, so that if G is a Σ-embedded graph with $|E(G)| \geq 2$ and $\mathcal{M}$ is its medial graph, then*

$$2bw(G) \leq cw(\mathcal{M}) \leq 4bw(G) + c(\Sigma). \tag{3.6.1}$$

The constant $c(\Sigma)$ can be chosen as $c(\Sigma) = 4c$, where c is the constant from Lemma 3.6.6.

We start by proving an easy fact about embedded graphs. Note that the statement below is vacuous if Σ is the sphere, since graphs as described can not be planar.

Proposition 3.6.2. *Let $G \hookrightarrow \Sigma$ be connected. Suppose that G has minimum degree at least two, and that G has exactly one region. Let D_3 be the vertices of G of degree at least three. Then*

$$\sum_{v\in D_3} d(v) \leq 6(1-\chi(\Sigma)),$$

where $\chi(\Sigma)$ denotes the Euler characteristic of Σ.

Proof. We may assume that G is 2-cell embedded in Σ; if not, there is a 2-cell embedding of G into a surface Σ' (with exactly one region) with $\chi(\Sigma') \geq \chi(\Sigma)$, and hence the inequality holds for Σ if it holds for Σ'.

Using $m = |E(G)|$ and $n = |V(G)|$, we have

$$m = \sum_{v \in V(G)} (d(v)/2) = n + \sum_{v \in V(G)} (d(v)/2 - 1) = n + \sum_{v \in D_3} (d(v)/2 - 1).$$

Hence by Euler's formula Theorem 2.2.2, we get

$$2(1 - \chi(\Sigma)) = \sum_{v \in D_3} (d(v) - 2).$$

Since $d(v) - 2 \geq 1$ for $v \in D_3$, we have that $|D_3| \leq 2(1 - \chi(\Sigma))$, and so

$$\begin{aligned} \sum_{v \in D_3} d(v) &= 2(1 - \chi(\Sigma)) + \sum_{v \in D_3} 2 \\ &= 2(1 - \chi(\Sigma)) + 2|D_3| \\ &\leq 6(1 - \chi(\Sigma)). \end{aligned}$$

□

The following lemma turns out to be the key fact for the approximate upper bound in Theorem 3.6.1.

Lemma 3.6.3. *For any surface Σ, there is a non-negative constant $c(\Sigma)$, so that the following holds: Suppose G is a bipartite graph embedded in Σ, where $\Omega \subseteq V(G)$ is one bipartition class, and let ins be a pre-slope of order θ in G, with respect to Ω. Suppose C is a cluster in G. Then we have*

$$m_C \leq 4w(C) + c(\Sigma),$$

where $m_C = |E(C)|$, and $w(C) = |V(C) \cap \Omega|$ is the weight of C. The constant $c(\Sigma)$ can be chosen as $c(\Sigma) = \max\{2c, 12(1 - \chi(\Sigma))\}$, where c is the constant from Lemma 3.2.9.

Proof. Let $\mathcal{S}_C$ be the structure graph of C (as defined in 3.3.5, i.e. the bipartite graph with bipartition classes $\mathcal{V}^{2+} = \{v \in V(C) \mid d_C(v) > 2\}$ and $\mathcal{B} = \{B_1, \ldots, B_k\}$ (where $B_1, \ldots, B_k$ are the maximal borders in C), as defined in 3.3.5.

First suppose $|V(\mathcal{S}_C)| = 1$, i.e. C is a (single) border. Since C is bipartite, we have

$$\begin{aligned} m_C &= \sum_{v \in \Omega \cap V(C)} d(v) \\ &= \sum_{v \in \Omega \cap V(C)} 2 + \sum_{v \in \Omega \cap V(C)} (d(v) - 2) \\ &\leq 2w(C) + \sum_{v \in V(C)} (d(v) - 2) \\ &= 2w(C) + 2(m_C - n_C) \end{aligned}$$

and so by applying Lemma 3.2.9 to the border C, we get $m_C \leq 2w(C) + c$, for a constant c depending only on Σ, as desired.

Hence we may assume that $|V(\mathcal{S}_C)| \geq 2$, so that in particular, C contains at least two distinct maximal borders.

Claim 3.6.3.1. $\mathcal{S}_C$ has minimum degree at least two.

Proof of claim. Clearly $\mathcal{S}_C$ has minimum degree at least one, since $\mathcal{S}_C$ is connected (by the definition of a cluster) and $|V(\mathcal{S}_C)| \geq 2$. Suppose that $w \in V(\mathcal{S}_C)$ has degree one, say $v \sim w$. Note that $w \notin \mathcal{V}^{2+}$, because $w \in \mathcal{V}^{2+}$ would imply $d_{\mathcal{S}_C}(w) = 1/2 \cdot d_C(w) > 1$. Hence $w \in \mathcal{B}$, and $v \in \mathcal{V}^{2+}$. But then $\mathcal{S}_C \setminus v$ is not connected (since $|\mathcal{B}| \geq 2$ by assumption), contrary to the fact that C is a cluster. $\square$

We assume that $\mathcal{S}_C$ is embedded in Σ in its 'natural' embedding, described as follows:

The graph C is confined, and hence by Lemma 3.2.15, $\Sigma \setminus ins(C)$ is one region of C, say r_0. Since $C = B_1 \cup \ldots B_k$, we have that C has precisely k regions $r_1, \ldots, r_k$ other than r_0, where $r_i = ins(B_i) \setminus U(B_i)$ (note that $ins(B_i) \cap ins(B_j)$ contains neither edges nor regions, by Lemma 3.2.18). Now the natural embedding of $\mathcal{S}_C$ is obtained by placing a vertex b_i for each maximal border $B_i \in \mathcal{B}$ contained in r_i, and connecting it to all $v \in \mathcal{V}^{2+} \cap V(B_i)$ so that $u(vb_i) \setminus v$ is contained in r_i (see Figure 5 for an example).

Claim 3.6.3.2. The natural embedding of $\mathcal{S}_C$ has exactly one region.

Proof of claim. Let $e = vb_i \in E(\mathcal{S}_C)$, and s_1, s_2 be the regions of $\mathcal{S}_C$ incident at e. We show that $s_1 = s_2$. Let D be a disk (of small positive radius) centered at the midpoint $x(e)$ of e, such that $D \cap U(\mathcal{S}_C) \subseteq u(e)$, and let x_i be a point in $D \cap s_j$, for $j = 1, 2$ (in particular $x_j \notin U(\mathcal{S}_C)$). Let f_1, f_2 be the two edges of $E(B_i)$ (where B_i is the border represented by b_i) forming an angle at v, so that e appears between f_1 and f_2 (i.e. in the cyclic order of the edges at v in $E(B_i) \cup E(\mathcal{S}_C)$, the edges appear in the order f_1, e, f_2). If y_j is the midpoint of f_j, then there is a path P_j with ends x_j and y_j, contained in $(s_j \cap r_i) \cup \{y_j\}$, for $j = 1, 2$, where r_i is the region bounded by B_i as above. But since f_j is a r_0-r_j edge (as $f_j \in E(B_i)$) for $j = 1, 2$, there is a path P with ends y_1, y_2 whose interior is contained in r_0. Hence $P \cup P_1 \cup P_2$ is a x_1, x_2 path contained in $s_1 \cup s_2 \cup \{y_1\} \cup \{y_2\} \cup r_0$. But $y_j \notin U(\mathcal{S}_C)$ and $r_0 \cap U(\mathcal{S}_C) = \emptyset$ (since $U(\mathcal{S}_C) \subseteq ins(C) = r_0^c$) by construction of the embedding of $\mathcal{S}_C$, implying that $P \cup P_1 \cup P_2$ is disjoint from $U(\mathcal{S}_C)$ but connects $x_1 \in s_1$ to $x_2 \in s_2$, and so $s_1 = s_2$, as desired. $\square$

By the above two claims, we can apply Proposition 3.6.2 to the graph $\mathcal{S}_C$, and hence the following computation yields the desired result:

$$\begin{aligned}
m_C &= \sum_{v \in \Omega \cap V(C)} d_C(v) && (\Omega \cap V(C) \text{ is one bipartition class of } C) \\
&= \sum_{\substack{v \in \Omega \cap V(C) \\ d_C(v) \leq 4}} d_C(v) + \sum_{\substack{v \in \Omega \cap V(C) \\ d_C(v) \geq 6}} d_C(v) && (C \text{ is eulerian}) \\
&\leq 4w(C) + 2 \cdot \sum_{\substack{v \in V(\mathcal{S}_C) \\ d_{\mathcal{S}_C}(v) \geq 3}} d_{\mathcal{S}_C}(v) && (\text{since } d_C(v) = 2d_{\mathcal{S}_C}(v)) \\
&\leq 4w(C) + 2 \cdot 6(1 - \chi(\Sigma)) && (\text{by Proposition 3.6.2}).
\end{aligned}$$

Note that $6(1-\chi(\Sigma)) \geq 0$, since otherwise Σ is the sphere, and it is easy to see that in that case, any cluster is a single contractible cycle (and so $|V(\mathcal{S}_C)| = 1$, a case we discussed above). This proves Lemma 3.6.3. $\square$

Remark 3.6.4. From the first part of the above proof, we see that the multiplicative constant in Lemma 3.6.3 can be improved from 4 to 2 if every cluster consists only of a single border. The constant of 4 is tight for clusters which are a non-contractible chain of contractible 4-cycles (i.e. the cluster C consists of (edge-disjoint) contractible four-cycles $C_1, \ldots, C_k$ so that any two consecutive ones intersect in a unique vertex (in Ω), and C is the union of two (edge-disjoint) non-contractible cycles).

Remark 3.6.5. If Σ is the projective plane (so that $\chi(\Sigma) = 1$), then the above proof gives that for any cluster C, $m_C \leq \max\{2w(C) + c, 4w(C)\}$, where c is the constant from Lemma 3.2.9. However the additive constant is needed, because if C is a border consisting of two non-contractible cycles intersecting in a single vertex v (belonging to Ω), then $m_C = 2w(C) + 1$. In fact, c can be chosen to be one in the projective plane, but the example in the previous remark also applies to the projective plane, so in general the multiplicative constant of 4 is needed.

In order to prove Theorem 3.6.1, we first need to show how to construct an even slope from a tilt of appropriate order (in [67], this was done for traditional slopes in result (4.4)). Tilts were defined in 2.3.8.

Theorem 3.6.6. *Let G be a connected graph embedded in Σ, and let G^* denote its dual graph. Assume that G^* is bipartite, where $\Omega \subseteq V(G^*)$ is one bipartition class. If G has a tilt of order k for some integer $k \geq 1$, then G^* has an even slope (with respect to Ω) of order at least $k/4 - c$, with $c = c(\Sigma)/4$, where $c(\Sigma)$ is the constant from Lemma 3.6.3.*

Proof. Let $\mathcal{T}$ be a tilt in G of order k. In order to construct an even slope of order $k/4 - c$ in G^*, we need to define $ins(B)$ for every border B in G^* of weight $w(B) < k/4 - c$. Let B be such a border, let $\triangle_1, \triangle_2$ be the two spots bounded by it (i.e. the closures of the two components of $\Sigma \setminus B$), and let $X_i = V(G) \cap \triangle_i$, for $i = 1, 2$ (so that $X_1 \cup X_2 = V(G)$ and $X_1 \cap X_2 = \emptyset$). Then $\delta(X_1) = \delta(X_2) = E(B)^*$, and so by Lemma 3.6.3, we have $|\delta(X_1)| = |E(B)| \leq 4w(B) + c(\Sigma) < k - 4c + c(\Sigma) = k$. Therefore exactly one of X_1, X_2 belongs to $\mathcal{T}$, say X_1, in which case we set $ins(B) = \triangle_1$.

We claim that ins defines an even slope in G^*, of order $k/4 - c$.

Clearly S1) holds: If $U(B_1) \subseteq ins(B_2)$ but $ins(B_1) \not\subseteq ins(B_2)$ for two borders B_1, B_2 of weight $< k/4 - c$, then $ins(B_1) \cup ins(B_2) = \Sigma$ and hence $(V(G) \cap ins(B_1)) \cup (V(G) \cap ins(B_2)) = V(G)$, contradicting the second tilt axiom. We next verify that axiom S2) holds. Let $Z_1, Z_2, Z_3 \subseteq R(G^*)$ be bounded by clusters C_1, C_2, C_3 with $w(C_i) < k/4 - c$ for $i = 1, 2, 3$. Let $X_i \subseteq V(G)$ be the vertices corresponding to the elements of Z_i. Then $\delta(X_i) = E(C_i)^*$, and so as before, Lemma 3.6.3 implies $|\delta(X_i)| = |E(C_i)| \leq 4w(C) + c(\Sigma) < k$ for $i = 1, 2, 3$. Therefore exactly one of $X_i, V(G) \setminus X_i$ belongs to the tilt $\mathcal{T}$, and by our definition of ins, we have $X_i \in \mathcal{T}$ for $i = 1, 2, 3$. But if $ins(C_1) \cup ins(C_2) \cup ins(C_3) = \Sigma$, then $X_1 \cup X_2 \cup X_3 = V(G)$, contrary to the second tilt axiom, and so S2) holds.

It remains to check that ins is even. Let r_v be a region of G^*, corresponding to a vertex $v \in V(G)$. We will show that $r_v \subseteq ins(B)$ for some light border B. The proof is analogous to the corresponding part of the proof of (4.4) in [67]:

Choose a set $X \in \mathcal{T}$ maximal with respect to $v \in X$ and $G[X]$ connected - this is possible since $\{v\} \in \mathcal{T}$ by the third tilt axiom. It is then shown in (4.4) of [67] that $\delta(X)$ is a bond in G, and so by Proposition 3.2.7, $\delta(X)^*$ forms the edge-set of a border B in G^*. Since $X \in \mathcal{T}$ implies in particular that $|\delta(X)| < k$, we deduce as before that B is light, and $X \in \mathcal{T}$ together with $v \in X$ implies that $r_v \subseteq ins(B)$, as desired. $\square$

We are now ready to prove Theorem 3.6.1.

Proof of Theorem 3.6.1. Let $\beta = bw(G)$ and $\kappa = cw(\mathcal{M})$. The first part of the proof of Theorem 2.3.10 in [67] shows that a carving-decomposition of order $2k$ in $\mathcal{M}$ yields a branch-decomposition of order k for G. This proof works on any surface and hence we have $2\beta \leq \kappa$.

The proof of the second inequality $\kappa \leq 4\beta + c(\Sigma)$ is similar (but slightly simpler since we are dealing with graphs only) to the corresponding part from [67]. Since $\mathcal{M}$ is 4-regular, we either have $|\delta(v_e)| < \kappa$ for all $v_e \in V(\mathcal{M})$, or $\kappa \leq 4$. In the latter case, the desired inequality holds, unless $\beta = 0$, which is not the case since we assumed $|E(G)| \geq 2$. Hence by Theorem 2.3.9, there is a tilt of order κ in $\mathcal{M}$ (since $\mathcal{M}$ has carving-width κ), and so by Theorem 3.6.6, the radial graph $\mathcal{M}^* = \mathcal{R}(G)$ contains an even slope of order at least $\kappa/4 - c$, for some non-negative constant c. By Theorem 3.4.2, G contains a tangle of order at least $\kappa/4 - c$, and so by Theorem 2.3.3, we have $\beta \geq \kappa/4 - c$, yielding the desired inequality if we choose $c(\Sigma) = 4c$. $\square$

The existence of a difference between $cw(\mathcal{M}(G))$ and $2bw(G)$ for surfaces other than the sphere, suggested by Theorem 3.6.1, already shows up in very small examples:

Example 3.6.7. Let G be the 2×2 toroidal grid, embedded in the natural way on $\mathbb{S}_1$ with face-width two. Then G has 4 vertices and 4 pairs of parallel edges (in fact we can make the graph simple by subdividing each edge, which will not change its branch-width or the carving-width). We will show that $bw(G) = 2$, while $cw(\mathcal{M}(G)) = 8$.

Note that G has no tangle of order three: Let v_1, v_2 be two non-adjacent vertices of G, and let $A_i = G[\delta(v_i) \cup v_i]$, for $i = 1, 2$. Then the separation (A_1, A_2) has order two, and it is easy to see that we can have neither (A_1, A_2) nor (A_2, A_1) in a tangle (both A_1 and A_2 can be built by adding one edge at a time while maintaining that the corresponding separations have order at most two, so both (A_1, A_2) and (A_2, A_1) should belong to the tangle, but this would violate the second tangle axiom since $A_1 \cup A_2 = G$). In fact it is easy to use the separation (A_1, A_2) to construct a branch-decomposition of with two. Since clearly $bw(G) \geq 2$, we have $bw(G) = 2$.

On the other hand, $\mathcal{M}(G)$ has carving-width at least 8: If T is the tree of a carving-decomposition of $\mathcal{M}(G)$, then there is an edge of T so that both sides of the corresponding partition X, X^c of $V(\mathcal{M}(G))$ have size at least three (note that $|V(\mathcal{M}(G))| = 8$). Suppose $|X| \leq |X^c|$, so that $|X| \in \{3, 4\}$. Since $\mathcal{M}(G)$ is triangle-free and $\mathcal{M}(G)$ is 4-regular, it follows that $|\delta(X)| \geq 8$, and so any carving-decomposition will have width at least 8. Conversely, it is easy to give a carving-decomposition of width 8 for $\mathcal{M}(G)$ (e.g. by starting with the analogue of the above separation (A_1, A_2) in $\mathcal{M}(G)$), and so $cw(\mathcal{M}(G)) = 8$.

In Example 3.5.5 we showed that the branch-width of the 3×3 toroidal grid is 4, and using somewhat similar arguments to those in Lemma 3.5.3, one can show that the carving-width of its medial graph (on the torus) is 10, hence providing another example where $2bw(G) < cw(\mathcal{M}(G))$.

In fact it would be interesting to determine the branch-width of the $k \times k$ toroidal grid for any k exactly, as well as the carving-width of the corresponding medial graphs. In Theorem 3.5.4, we have seen that if G is the $k \times k$ toroidal grid, then $bw(G) \geq (1 + \epsilon)k$ (at least for k sufficiently large), and it is also not hard to see that $bw(G) \leq 2k - 2$ for $k \geq 2$ (in fact is seems to be the case that $bw(G) < (1 - \delta)2k$ for some $0 < \delta < 1$).

We believe that $cw(\mathcal{M}(G)) > 2bw(G)$ for all k for these graphs, although it may not be the case that these examples are tight for the bounds in Theorem 3.6.1. Nevertheless, based on the remark following Lemma 3.6.3, we conjecture that the multiplicative constant of 4 in Theorem 3.6.1 is best possible, in fact for all surfaces other than the sphere.

3.7 Conclusion

The main motivation for the results of this chapter was to investigate whether the 'ratcatcher method' of [67] can be generalized to compute the branch-width of a graph embedded in a surface other than the sphere.

The three main results from this chapter (Theorem 3.4.2, Theorem 3.5.4, Theorem 3.6.1) combined show that this is not the case. This is because of the following: Combining various results from [63] and [67], we have that in the plane, the following statements are equivalent for any planar graph G, its associated medial graph $\mathcal{M}(G)$ and radial graph $\mathcal{R}(G) = \mathcal{M}(G)^*$:

(1) G has has branch-width at least θ.

(2) G has a tangle of order θ.

(3) $\mathcal{R}(G)$ has a (traditional) even slope of order θ (defining $ins(C)$ for every cycle of length $< 2\theta$, or equivalently of weight $< \theta$).

(4) $\mathcal{M}(G)$ has an antipodality of order 2θ, or a vertex v with $\delta(v) \geq 2\theta$.

(5) $\mathcal{M}(G)$ has carving-width at least 2θ.

The ratcatcher method actually computes the maximum order of an antipodality, and hence by the above equivalences the branch-width of a planar graph G.

Now '(1) $\Leftrightarrow$ (2)' holds for arbitrary graphs, and we have shown (Theorem 3.4.2) that '(2) $\Leftrightarrow$ (3)' generalizes to arbitrary surfaces with our extended notion of slopes.

However result Theorem 3.6.1 and the examples given in the previous section indicate that (1) $\Leftrightarrow$ (5) no longer holds, so any exact algorithm to determine $cw(\mathcal{M}(G))$ would only yield an approximation algorithm for $bw(G)$, unlike in the plane.

Moreover, the concept of antipodalities, as defined in [67], no longer corresponds to either $bw(G)$ or $cw(\mathcal{M}(G))$: Consider for example the $k \times l$ toroidal grid $G_{k\times l}$, for $l \gg 2k$: It is easy to see that $bw(G_{k\times l}) \leq 2k$ (in fact equality holds), and that $cw(\mathcal{M}(G_{k\times l})) \leq 4k$. However it is also not hard to see that the maximum order of an antipodality in $\mathcal{M}(G_{k\times l})$ is proportional to l, i.e. on surfaces other than the sphere, the maximum order of an antipodality is unrelated to branch-width or carving-width in general.

This is actually not surprising: The reason that the medial graphs of $G_{k\times l}$ allow such high order antipodalities is that from a fixed point on the surface, we cannot capture all other points on the surface by contractible walks unless we use walks of order proportional to l. In fact, in order to try to save (4) $\Leftrightarrow$ (5), it would be natural to define a more powerful notion of antipodalities which do not only allow 'capturing' by contractible walks, but other simple non-contractible configurations (e.g. two disjoint, homotopic non-contractible cycles with an 'inside' defined for them). In other words, one could try to define a generalized version of antipodalities in order to get an obstruction for carving-width in embedded graphs, similar to how we defined our generalized slopes to get exact obstructions for the branch-width of embedded graphs.

However more powerful antipodalities will save neither (4) $\Leftrightarrow$ (5), nor (3) $\Leftrightarrow$ (4) exactly, as implied by the main result of Section 3.5: If G is a graph so that $\mathcal{R}(G)$ is as in Theorem 3.5.4, i.e. $R(G)$ has a slope of order $(1+\epsilon)\theta$, but we can capture all of Σ by restraints of length $< 2\theta$, for ϵ, θ chosen appropriately, then clearly (3) $\Leftrightarrow$ (4) can not be saved since introducing more powerful antipodalities only worsens Theorem 3.5.4. Moreover, the fact that we can capture all of Σ with short restraints in $\mathcal{R}(G)$ means in particular that the dual of $\mathcal{R}(G)$, namely $\mathcal{M}(G)$, allows no antipodality of order 2θ or higher, and so the same will hold true if we make the antipodalities more powerful. But since such a G has branch-width $\geq (1+\epsilon)\theta$ by Theorem 3.4.2, we have $cw(\mathcal{M}(G)) \geq 2(1+\epsilon)\theta$ by the first inequality of Theorem 3.6.1, i.e. $cw(\mathcal{M}(G))$ exceeds the maximum order of a traditional or more powerful antipodality.

In conclusion, we have shown that the equivalences between $(1), (2), (3)$ still hold exactly with the appropriate generalizations, that $(1) \Leftrightarrow (5)$ only holds as an approximate version in general, and we argued that neither $(4) \Leftrightarrow (5)$ nor $(3) \Leftrightarrow (4)$ can be saved in an exact version. However while traditional antipodalities are unrelated to carving-width or slopes in general (as the above example $G_{k\times l}$ shows), it may in principle be possible to obtain an approximate version (within a constant factor) of $(3) \Leftrightarrow (4)$ or $(4) \Leftrightarrow (5)$ by using more general antipodalities.

In terms of polynomial-time computability, recall that the ratcatcher method actually computes the carving-width (of a planar graph), by computing the maximum order of an antipodality. Hence the above arguments imply that even if one defined more general antipodalities (corresponding (approximately) to carving-width on higher surfaces), this would at best yield a constant factor approximation algorithm for branch-width by Theorem 3.6.1, for a fixed surface.

Another approach would be to try to compute the maximum order of a (generalized) slope directly, although it is not clear whether this is possible in polynomial time. For example, a first very simple algorithmic challenge (which would probably also appear if one would try to compute some version of a generalized antipodality) would be to compute a border of minimum weight of fixed homotopy type, e.g. among all borders consisting of two non-contractible, homotopic, disjoint cycles, compute one of minimum weight (it is not known in general whether it is NP-hard to compute a shortest non-contractible cycle on an arbitrary surface [69]).

Since it is unclear whether there is a polynomial-time algorithm for computing the branch-width of embedded graphs exactly in general, we mention two other possible approaches: Since it is known that $bw(G) \leq \mathcal{O}(\sqrt{|V(G)|})$ for graphs embedded on a surface (implied by results from [1]), it is possible to determine $bw(G)$ exactly in sub-exponential time (see e.g. [28]).

On the other hand, if G is embedded on Σ with genus g, the following simple recursive algorithm (suggested by Thomas, private communication) gives a 2^g approximation algorithm which runs in polynomial time if g is bounded by a constant:

(1) Find a shortest noncontractible closed curve ϕ intersecting G in a set X of r vertices.

(2) Remove X from G to obtain a graph G' embedded in a surface Σ' of lower genus, and let $\beta' = bw(G')$. Run the algorithm recursively to get an approximation $\hat{\beta}'$ for β'.

(3) Output $\hat{\beta} = \max\{r, \hat{\beta}'\}$.

The proof of correctness is simple: Letting $\beta = bw(G)$, we have that $\beta \geq r$ by results from [63], $\beta \geq \beta'$ since G' is a subgraph of G, and $\beta \leq \beta' + r$, since deleting r vertices from G decreases the branch-width by at most r, implying that

$$\max\{r, \beta'\} \leq \beta \leq \beta' + r \leq 2\max\{r, \beta'\}. \tag{3.7.1}$$

Hence in each iteration (i.e. each time we move from G to G', decreasing the genus) we lose a factor of 2, each of the g iterations can be implemented in polynomial time, and so the algorithm is as claimed.

CHAPTER IV

MINOR-MINIMAL PLANAR GRAPHS OF FIXED BRANCH-WIDTH

t

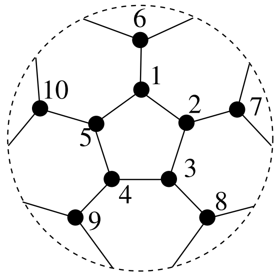
4.1 Introduction

Given a graph H embeddable in the projective plane, one can apply the following construction to obtain a planar graph G, called a *planar double cover* of H: Fix a circle γ in $\mathbb{R}^2$ and let θ, θ' be the closures of the two components of $\mathbb{R}^2 \setminus \gamma$. Draw one copy of H in θ and one in θ', so that diagonally opposite points on γ are either disjoint from both copies of H, or correspond to the same vertex of H or the midpoint of the same edge $e \in E(H)$ in both copies. In this way, we obtain an embedding of a planar graph G where every vertex, edge and region of H is represented twice in the drawing of G. See Figure 7 for an example where H is the Petersen graph, and G is the Dodecahedron.

The above construction shows that every projective planar graph has a planar cover (a graph G is a *cover* of a graph H if there is a surjective map $f : V(G) \to V(H)$ such that for every $v, w \in V(G)$, $vw \in E(G)$ if and only if $f(v)f(w) \in E(H)$). The question of when a graph has a planar cover is the subject of a conjecture by Negami [55]:

Conjecture 4.1.1 (Negami). If a connected graph H is covered by a planar graph G, then H is planar or projective planar.

In this chapter, we show that this construction relates minor-minimal embeddings



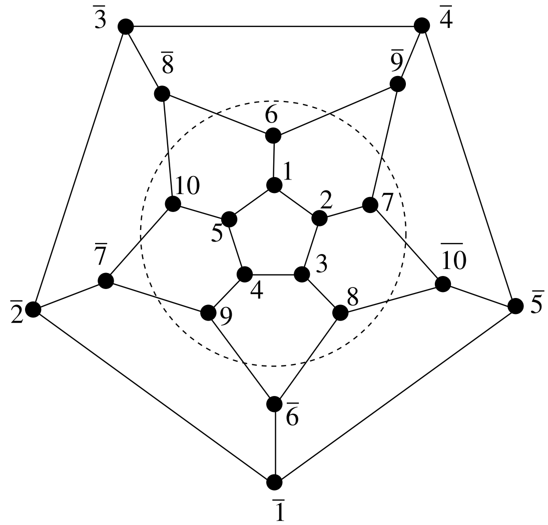


Figure 7: An embedding of the Petersen graph in $\mathbb{P}$ and its planar double cover, the Dodecahedron

in the projective plane with face-width k to minor-minimal planar graphs of branch-width $2k$.

More precisely, for an integer $l \geq 2$, let $\mathcal{C}_l$ be the minor-minimal planar graphs of branch-width exactly l. Since this is the obstruction set for minor-minimal planar graphs of branch-width $l-1$, the Graph Minor Theorem [64] implies that $|\mathcal{C}_l|$ is finite for every l.

It would be interesting to determine the structure of the classes $\mathcal{C}_l$. For small values of l, a complete description is known [62]: $\mathcal{C}_1 = \{K_2\}$ (where K_2 denotes the complete graph on 2 vertices), $\mathcal{C}_2 = \{P_2\}$ (where P_2 denotes a path of length 2), and $\mathcal{C}_3 = \{K_4\}$ (in fact the planarity condition is not needed here). For $k = 4$, $\mathcal{C}_4$ consists of exactly two graphs, namely the cube and its dual, the octahedron (there are two other non-planar, minor-minimal graphs of branch-width 4, see [44], [29], [15]). It easy to see that each of these two graphs is a planar double cover of one of the two minor-minimal embeddings of face-width 2 in the projective plane. The main result of this chapter shows that this is not a coincidence:

Theorem 4.1.1. *Let $k \geq 2$ be an integer, and let G be a double cover of a projective-planar minor-minimal graph of face-width k. Then G is minor-minimal of branch-width $2k$.*

Of course a tempting conjecture would be that all graphs in $\mathcal{C}_{2k}$ arise in this way for $k \geq 2$, i.e. every $G \in \mathcal{C}_{2k}$ is a planar double cover of a minor-minimal graph embedded in $\mathbb{P}$ of face-width k. In particular, the structure of the minor-minimal graphs of face-width k in the projective plane is completely understood, and all such graphs have been determined explicitly for $k = 2$ and $k = 3$ ([71], [8]).

Unfortunately that conjecture is false, as we show in 4.3.

4.2 Proof of Theorem 4.1.1

We first give a more rigorous definition of a planar double cover. Let H be a graph embeddable in the projective plane, let γ be a circle in $\mathbb{R}^2$, and let θ_1, θ_2 be the closures of the two components of $\mathbb{R}^2 \setminus \gamma$. Draw two copies H_i of H in θ_i for $i = 1, 2$ (with the usual convention that diagonally opposite points on γ are identified) so that the conditions below are satisfied. We denote the two copies corresponding to a vertex $v \in V(H)$ by $v_i \in V(H_i)$, for $i = 1, 2$.

(1) γ intersects H_i for $i = 1, 2$ either only in vertices, or only in midpoints of edges.

(2) $v_1 \in V(H_1)$ is on γ if and only if $v_2 \in V(H_2)$ is on γ.

(3) $p \in \gamma$ is the midpoint of an edge $e_1 = v_1w_1 \in E(H_1)$ if and only if it is the midpoint of an edge $e_2 = v_2w_2 \in E(H_2)$.

If we do not identify opposite points on γ, we naturally obtain a plane graph G from the two copies of H where every vertex and every edge of H is represented exactly twice in G. We say a planar graph G is a planar double cover of a projective planar graph H if an embedding of G can be constructed from H as described above.

Let $H \hookrightarrow \mathbb{P}$ be minor-minimal of face-width k, and G be a planar double cover of H. We first establish $bw(G) \geq 2k$ by constructing an antipodality in 4.2.1, and then show how a certain way of 'sweeping' through G can be used to prove that G is minor-minimal of branch-width $2k$.

For both parts, we often need to consider corresponding structures of a projective planar graph H and its planar double cover G. If x_H is a vertex, edge or region of H, and H is not planar, then there are exactly two elements $x, \bar{x}$ (vertices, edges or regions) in G corresponding to x. For an element x in G, we call $\bar{x}$ the *complementary* element in G. If G' is a subgraph of G which contains no complementary vertices, the *complementary subgraph* of G' is the subgraph consisting of the complementary vertices and edges of G (in particular, this applies to facial cycles).

We say $x, \bar{x}$ are *lifts* of x_H into G, and x_H is the *projection* of x and $\bar{x}$ into H. If P is a path in G with complementary ends $v, \bar{v}$, but no other pair of complementary vertices on P, then the cycle C in H whose edges are the projections of $E(P)$ is called the *projection* of P, and P is called a *lift* of C. Hence the edges of the cycle C in H and those of the lift P are in 1-1 correspondence, and in particular $|C| = |P|$.

Remark 4.2.1. If H is connected, and neither H nor its dual H^* contain any loops, then complementary elements in G are never adjacent to each other: If a vertex v is incident with $\bar{v}$, in G, this would correspond to a loop of H, and regions $r, \bar{r}$ of G being adjacent corresponds to a loop in H^*. Hence complementary edges $e, \bar{e}$ cannot share an endpoint either.

Remark 4.2.2. There is an easy way to check whether a cycle C is contractible in H: Suppose that H is drawn in a closed disc bounded by a circle γ (where diagonally opposite points are identified) so that γ intersects H only in midpoints of edges. Assign a negative sign to edges which 'cross' γ (i.e. whose midpoints are on γ), and a positive sign to all other edges. Then it is easy to see that C is contractible if and only if it contains an even number of negative edges.

4.2.1 Lower bound on branch-width

Let H, G be as in the beginning of 4.2. We now construct an antipodality of appropriate range in the medial graph of G , which in light of Theorem 2.3.12 and Theorem 2.3.10 will give a lower bound of $2k$ on the branch-width of G.

It is easy to see that the radial graph $\mathcal{R}_G$ of G is in fact a planar double cover of $\mathcal{R}_H$, and similarly, $\mathcal{M}_G$ is a double cover of $\mathcal{M}_H$.

Recall that vertices of $\mathcal{M} = \mathcal{M}_G$ correspond to edges of $\mathcal{R}_G$, each edge of $\mathcal{M}$ corresponds to two edges in G forming an angle at vertex $v \in V(G)$ with a region $r \in R(G)$, and faces of $\mathcal{M}$ correspond to vertices and regions of $V(G)$.

Lemma 4.2.3. *Let $H \hookrightarrow \mathbb{P}$ be minor-minimal of face-width $k \geq 2$, and let G be a planar double cover of H. Let $\mathcal{M}$ be the medial graph of G. Then $\mathcal{M}$ has an antipodality of range $4k$.*

Proof. We need to assign a subgraph of $\mathcal{M}$ to each edge in $\mathcal{M}$, and a set of vertices of $\mathcal{M}$ to each face of $\mathcal{M}$. Let $a_{v,r}$ be an edge in $\mathcal{M}$, whose two adjacent regions in $\mathcal{M}$ are $v \in V(G)$ and $r \in R(G)$. Let e_1, e_2 denote the ends of $a_{v,r}$ in $\mathcal{M}$. Then in G, the edges e_1, e_2 form an angle at $v \in V(G)$ and are consecutive in the boundary of the face $r \in R(G)$. For an edge $a_{v,r}$ of $\mathcal{M}$, an edge e of $E(G)$, a region π of $\mathcal{M}$ let $a_{\bar{v},\bar{r}}$, $\bar{e}$, $\bar{\pi}$ be the complementary edge or region, respectively. Finally for a region π of $\mathcal{M}$, let C_π denote the cycle bounding that region (Note that π corresponds either to a vertex or a region of G, and that every face of the medial graph $\mathcal{M}$ is indeed bounded by a cycle: It is easy to see that the G has no loops or cutedges, and this implies that $\mathcal{M}$ is a 2-connected plane graph). We now define the function α as follows:

$$\alpha(a_{v,r}) = C_{\bar{v}} \cup C_{\bar{r}} \text{ for every } a_{v,r} \in E(\mathcal{M}),$$

$$\alpha(\pi) = V(C_{\bar{\pi}}) \text{ for every } \pi \in R(\mathcal{M}).$$

We claim that α defines an antipodality of range $4k$ in $\mathcal{M}$.

It is easy to check that $\alpha 1)$ holds: The ends of the edge $a_{v,r}$ are e_1 and e_2. Suppose for a contradiction that say $e_1 \in V(\alpha(a_{v,r}))$. If $e_1 \in V(C_{\bar{v}})$, then in G, we have that e_1 has ends v and $\bar{v}$, a contradiction as pointed out in Remark 4.2.1. Similarly if $e_1 \in V(C_{\bar{r}})$, we get that r and $\bar{r}$ are adjacent in G^*, again yielding a contradiction. Hence $\alpha 1)$ holds.

Clearly $\alpha 2)$ holds, since $\alpha(a_{v,r})$ is connected (both $C_{\bar{v}}$ and $C_{\bar{r}}$ contain the edge $a_{\bar{v},\bar{r}}$), and by definition, $\alpha(v)$ and $\alpha(r)$ are subsets of $V(\alpha(a_{v,r}))$.

We now verify that $\alpha 3)$ holds. Suppose for a contradiction that for some edge $l_1 = a_{v,r}$ of $\mathcal{M}$ and some edge l_2 of $\alpha(a_{v,r})$, there exists a closed walk W in $\mathcal{M}^* = \mathcal{R}_G$ of length less than $4k$, containing both l_1^* and l_2^*. By definition of $\alpha(a_{v,r})$, at least one of $\bar{v}$, $\bar{r}$ is an end of l_2^*, say $\bar{v}$. Hence W can be divided into two subwalks W_1, W_2 with endpoints v and $\bar{v}$, and assume W_1 is a shortest $v - \bar{v}$ subwalk of W. Then W_1 is actually a path: By definition, all the internal vertices of W_1 are distinct from $v, \bar{v}$, and so if l_1^* or l_2^* are on W_1, then they must be one of the ending edges of W_1. Hence if W_1 contained a cycle, then the cycle would contain neither l_1^* nor l_2^*, and we could remove it to obtain a shorter walk, contrary to the choice of W_1. Therefore W_1 is a $v - \bar{v}$ path, and since $|W_1| \leq |W_2|$, we have $|W_1| < 2k$.

Let P be a subpath of W_1 with complementary ends $w, \bar{w}$, but no other pair of complementary vertices on P. Without loss of generality our embedding of $\mathcal{R}_G$ is a double cover of $\mathcal{R}_H$ such that both copies of $\mathcal{R}_H$ are of the type described in Remark 4.2.2. Since the endpoints of P are complementary, it follows that there are an odd number of edges on P of negative sign. Hence the projection of P into $\mathcal{R}_H$ is a noncontractible cycle of length less than $2k$, contrary to the fact that $ew(\mathcal{R}_H) = 2fw(H) = 2k$. We deduce that any walk W as described must have length at least $4k$, and so $\alpha 3)$ holds. $\square$

4.2.2 Upper bound on branch-width

Let H, G be as before (i.e. H is a minor-minimal embedding of face-width k in the projective plane, and G is a planar double cover of H), and $\mathcal{R}_H, \mathcal{R}_G$ be their radial graphs. The goal of this section is to show that the branch-width of G becomes strictly less than $2k$ if we delete or contract an edge.

The idea is roughly as follows: Start with an arbitrary noncontractible cycle C_0 of minimum length in the radial graph $\mathcal{R}_H$, and find a region r_1 of $\mathcal{R}_H$ whose boundary cycle intersects C_0 in exactly two adjacent edges. Now move C_0 'across' r_1 to obtain a new noncontractible cycle C_1. If we are careful with the selection of regions to move across, we can continue this process until we have covered every region of $\mathcal{R}_H$ exactly once, and are back to the initial cycle C_0.

In H, this defines an ordering of the edges so that at each step, the boundary of the set of covered edges will have at most $2k$ vertices, and it follows that H has a branch-decomposition of order at most $2k$. In fact, we can use the same order of regions of $\mathcal{R}_H$ to cover all regions of $\mathcal{R}_G$, by first covering m pairwise non-complementary regions, and then in a second phase covering all the complementary regions. Again this will define a branch-decomposition of G of order at most $2k$, and in fact the method can be extended to show that the branch-width decreases below $2k$ if we delete or contract an edge (actually this last step is all we need to show G is minor-minimal of branch-width $2k$, since we have already shown $bw(G) \geq 2k$).

The crucial part in the above method is to ensure that we can always make a next step, i.e. find a suitable region to move across. To show that this is always possible, we use the following concept of a *straight decomposition*, which is helpful to characterize minor-minimal embeddings of fixed face-width:

Definition 4.2.4. Let v be a vertex of degree four in an embedded graph G. Then for an edge e incident with v, define the *opposite* edge of e to be the edge incident with v at distance two from e in the cyclic ordering at v.

Definition 4.2.5. Let G be a four-regular, Σ-embedded graph. A closed walk S in G with the property that any two consecutive edges on S are opposite for some vertex v is called a *straight walk* in G. We say S *crosses* the edge $e^* \in E(G^*)$ if $e \in E(S)$. The *straight decomposition* of G is the collection of all straight walks in G.

The following result is implicitly proved in [51], and is explicitly stated and generalized in [66].

Theorem 4.2.6. *Let $H \hookrightarrow \mathbb{P}$, let $\mathcal{M}$ be its medial graph and let $k \geq 2$ be an integer. Then H is minor-minimal of face-width k if and only if the straight decomposition of $\mathcal{M}$ is a collection of noncontractible cycles any two of which intersect in exactly one vertex.*

Remark 4.2.7. Since there are only two homotopy classes in $\mathbb{P}$, any two non-contractible closed curves are homotopic, and intersect in at least one point.

As a first application (Corollary 4.2.10) which will be useful later, we show that noncontractible cycles of length $2k$ in the radial graph of a minor-minimal projective planar graph of face-width k can only intersect facial cycles in a restricted way.

Definition 4.2.8. If C is a cycle in an embedded graph G, a region $r \in R(G)$ is *C-nice* if $\partial(r) \cap C$ is a path of length at most two (the intersection is allowed to be empty).

Lemma 4.2.9. *Let $G \hookrightarrow \mathbb{P}$ be simple of edge-width $ew(G) = 2k$ for $k \geq 2$, such that every face is bounded by a four-cycle, and G has no separating four-cycles. Let C be a noncontractible cycle of length $2k$ in G. Then every $r \in R(G)$ is C-nice.*

Proof. Let C_r denote the four-cycle bounding r. First suppose that C_r contains an edge vv' with $v, v' \in V(C)$ but $vv' \notin E(C)$.

Then C can be written as the union of two $v - v'$ paths P, P' not containing vv'. Let $C' = P \cup vv'$ and $C'' = P' \cup vv'$ be two cycles whose union is $C \cup vv'$. Since C is noncontractible, it follows from Proposition 2.2.3 applied to the three paths P, P', vv' that at least one of C', C'' is noncontractible, say C'. Since $|C'| \leq |C| = ew(G)$, we have $|C'| = |C|$ and so $|P'| = 1$, a contradiction since the edge vv' was not in C by assumption (and G has no parallel edges).

Hence there is no such edge vv', which implies that $|E(C_r) \cap E(C)| \leq 2$. If $|E(C_r) \cap E(C)|$ is two or one, then the intersection must consist of a path, for otherwise there would be an edge vv' as above. Therefore we may assume $|E(C_r) \cap E(C)| = 0$, and furthermore $|V(C_r) \cap V(C)| \geq 2$, since otherwise we are done. In fact we may assume that the intersection of C_r and C consists of precisely two non-adjacent vertices v, w on C_r, since otherwise we again get an edge vv' as above.

Let P, P' denote the two paths with ends v, w whose union is C, and let P'' be one of the two $v - w$ paths whose union is C_r. Note that $|P''| = 2$ as $|C_r| = 4$, and $|P|, |P'| \geq 2$: If say P consists only of a single edge, then $P \cup P''$ forms a contractible triangle (since $ew(G) \geq 4$) which is non-facial, a contradiction since it is easy to see that G can not have separating odd cycles. Since C is noncontractible, we have that P is not homotopic to P'. Since there are only two homotopy classes in $\mathbb{P}$, we have that $P'' \sim P$ or $P'' \sim P'$. Without loss of generality $P'' \sim P$. Since replacing P by P'' in C yields again a noncontractible cycle, it follows that $|P| \leq |P''| = 2$ and so $|P| = 2$. But then $P'' \cup P$ is a contractible four-cycle, and hence bounds a face r' by assumption. However then the boundaries of the two faces r, r' intersect in a path of length two (namely P), and such a configuration would yield a separating four-cycle (G is not the union of the boundaries of r and r', since P' is disjoint from them), contradiction. □

Corollary 4.2.10. *Let $H \hookrightarrow \mathbb{P}$ be minor-minimal of face-width k for $k \geq 2$, and let C be a noncontractible cycle of length $2k$ in the radial graph $\mathcal{R}$ of H. Then every region r of $\mathcal{R}$ is C-nice.*

Proof. We show that Lemma 4.2.9 applies: Since $k \geq 2$, the radial graph $\mathcal{R}$ is a simple graph of edge-width $2k$ by Proposition 2.2.4, in which every region is bounded by a four-cycle. Suppose $\mathcal{R}$ has a separating four-cycle C'. Then C' bounds a disc θ containing at least two distinct regions $r, r' \in R(\mathcal{R})$. Let S_1, S_2 be the two straight walks containing r, and let S_3, S_4 be the two straight walks containing r'. Each of S_1, S_2 must cross C' in two distinct edges, since S_1, S_2 are noncontractible. However at least one of S_3, S_4 is distinct from both S_1 and S_2 (otherwise S_1 and S_2 meet in r and r', contrary to Theorem 4.2.6), say S_3. But then S_3 cannot cross C' since every edge of C' is either already crossed by S_1 or S_2, and so S_3 is contained in the disc bounded by C', a contradiction since S_3 is non-contractible. $\square$

In order to show that we can 'sweep' the projective plane by moving a non-contractible cycle C_0 of length $2k$ across faces, we first need to prove some preliminary results about plane graphs, which will be applied to the plane graph obtained from cutting open along C_0.

A *plane near-quadrangulation* is a plane graph G where every region apart from possibly the infinite region r_∞ is bounded by a four-cycle.

Definition 4.2.11. Let $\mathcal{S} = \{S_i\}_{i \geq 1}$ be a collection of cycles in the dual G^* (which may have parallel edges or loops) of a plane near-quadrangulation G, satisfying the following conditions:

(1) Any two consecutive edges on S_i incident with a common vertex $r \neq r_\infty$ of G^* are opposite.

(2) All cycles in $\mathcal{S}$ are edge-disjoint.

(3) Every edge $e^* \in E(G^*)$ is in a (unique) cycle S_i.

The cycles S_i are called the *straight cycles* (for G). The collection $\mathcal{S}$ of straight cycles is *non-degenerate* if any two distinct straight cycles intersect in r_∞ and exactly one other region $r \in V(G^*) \setminus r_\infty$.

Proposition 4.2.12. *Let $\mathcal{S}$ be a non-degenerate collection of straight cycles for a plane near-quadrangulation G, let C be a cycle in G bounding a disc θ and let r be a region of G in θ. Then every straight cycle $S \in \mathcal{S}$ containing r crosses C in at least two edges.*

Proof. It suffices to show that S crosses C in at least one edge, i.e. that there is an edge $e \in E(C)$ with $e^* \in E(S)$. Suppose not, so that $U(S) \subseteq \theta$. Let S' be the unique straight cycle in $\mathcal{S}$ containing r which is different from S. Since $U(S), U(S')$ are circles in $\mathbb{R}^2$ which cross in the point representing the region r, we have that there must be a second point $r' \in V(S \cap S')$ where they cross. In particular $r' \in V(S)$ implies $r' \subseteq \theta$, so $r' \neq r_\infty$, contrary to the assumption that $\mathcal{S}$ was non-degenerate. $\square$

In order to solve our original problem (construct a 'sweep' in $\mathbb{P}$), we will solve the following problem in the plane: Suppose G' is a near-quadrangulation whose infinite face is bounded by a cycle $P \cup Q$, where P and Q are paths with certain properties. Then we show how to transform Q onto P by moving across faces in G'.

We now define more formally what we mean by 'moving across faces'.

Definition 4.2.13. Let C be a cycle (or path) in an embedded graph, and let r be a region bounded by a 4-cycle C_r so that $C \cap C_r$ is a path P of length two. Then r is called a *2-face* for C. Let P' be the path of length two with edge-set $E(C_r) \setminus E(C)$. We denote by $C \triangle r$ the cycle (or path) obtained from replacing P by P' in C, and we say $C \triangle r$ is obtained from C by *moving across* r.

Remark 4.2.14. Note that $C, C \triangle r$ have the same length, and that if C is a cycle, one of them is contractible if and only if the other one is by Proposition 2.2.3.

The following concept of a *matching pair* will be used to locate a face to move across in each step.

Definition 4.2.15. Let G be a plane near-quadrangulation with a non-degenerate collection of straight cycles $\mathcal{S}$. Let C be a cycle in G bounding a disc θ, let $e \in E(C)$

and r the unique region of G in θ incident with e. Let $S \in \mathcal{S}$ be a straight cycle containing r which does not cross e. Follow S in both directions starting in r, and let $\tau_1(e)$ and $\tau_2(e)$ be the two (distinct) edges where S first crosses C (note that these edges exist by Proposition 4.2.12). For $i = 1, 2$, let $\tau_i(r)$ be the region of G in θ incident with $\tau_i(e)$. Then $(e, \tau_i(e))$ is called a *matching pair* for $i = 1, 2$.

For a matching pair $(e, \tau(e))$, let T be the $r - \tau(r)$ sub-path of S contained in θ, and let $\tilde{T}$ be a sub-path of C with ending edges e and $\tau(e)$ (note that e and $\tau(e)$ are distinct edges, since S crosses through $\tau(e)$, but not through e). Then $\tilde{T}$ is called a *matching path* for the pair $(e, \tau(e))$.

Lemma 4.2.16. *Let G be a plane near-quadrangulation with a non-degenerate collection of straight cycles $\mathcal{S}$. Let C be a cycle bounding a disc θ so that every region contained in θ is C-nice. For an edge $e \in E(C)$, let $(e, \tau(e))$ be a matching pair with matching path $\tilde{T}$. Then there is a face $r_0 \subseteq \theta$ so that $\partial(r_0) \cap \tilde{T}$ contains a path of length two.*

Proof. Suppose the statement is false for some edge $e \in E(C)$ and a corresponding path $\tilde{T}$, and consider a counterexample with $\tilde{t} = |\tilde{T}|$ minimum. Clearly $\tilde{t} \geq 2$ since $e, \tau(e)$ are distinct by definition. Let $r, S, T, \tilde{T}$ be as in Definition 4.2.15, and let $r' = \tau(r)$. Then $r \neq r'$, since otherwise r is as desired because in that case the edges $e, \tau(e) \subseteq E(\partial(r))$ share an end.

Let S' be the straight cycle containing r' which does not cross through $\tau(e)$. Let $e = x_1x_2$ and $\tau(e) = x_{\tilde{t}}x_{\tilde{t}+1}$, where x_1 and $x_{\tilde{t}+1}$ are the end-vertices of $\tilde{T}$. Let $e_1, e_2, \ldots, e_{\tilde{t}}$ be the edges of $\tilde{T}$, with $e_1 = e$ and $e_{\tilde{t}} = \tau(e)$ (see Figure 8).

Let $[x_2, r]$ be a line segment between x_2 and the point r^* so that $[x_2, r] \subseteq cl(r)$, and let $[r', x_{\tilde{t}}]$ be a line segment between the point r' and $x_{\tilde{t}}$ so that $[r', x_{\tilde{t}}] \subseteq cl(r')$. Note that if T is the $r - r'$ subpath of S in θ, then the closed curve given by $[x_2, r] \cup T \cup [r', x_{\tilde{t}}] \cup \tilde{T}$ is simple since $r \neq r'$, and hence bounds an open disc $\theta' \subseteq \theta$. Since the circle $U(S')$ crosses the boundary of $cl(\theta')$ at the point r', there exists a second point ρ different from r' where $U(S')$ intersects the boundary of $cl(\theta')$. Pick such a point ρ so that the interior of the segment between r' and ρ of S' is contained in θ'.

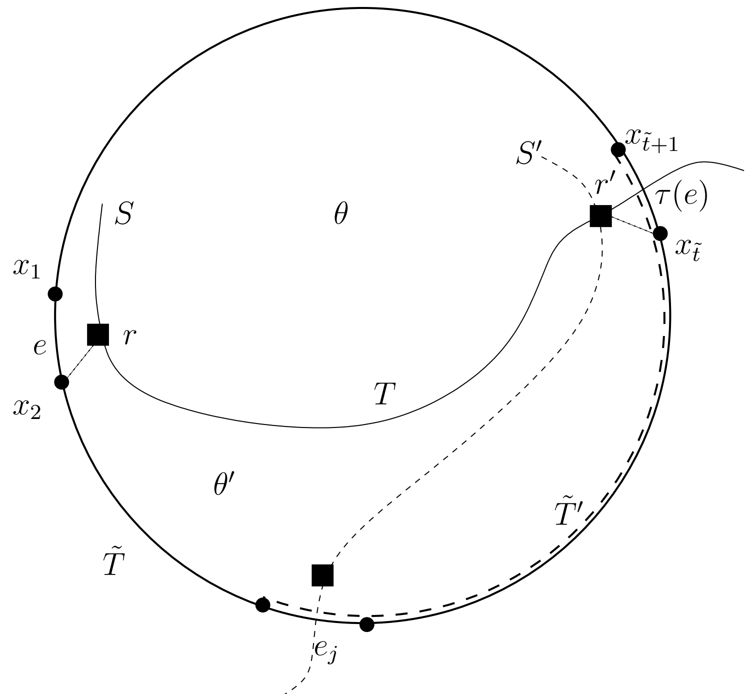


Figure 8: Configuration in proof of Lemma 4.2.16

Suppose $\rho \in [x_2, r]$. We claim that we have $\rho = r$ in this case. Clearly $\rho \neq x_2$, since x_2 is a vertex of $\mathcal{R}$, and S' is a subgraph of the dual of $\mathcal{R}$. If $\rho \in (x_2, r]$, then $U(S') \cap r \neq \emptyset$, and so $r \in V(S')$. Hence S' intersects S in two distinct points r and r', contrary to the assumption that the collection of straight cycles was non-degenerate. Similarly one shows that $\rho \in [r', x_{\tilde{t}}]$ implies $\rho = r'$.

But then by the definition of θ' and the fact that S' is a subgraph of the dual of $\mathcal{R}$, ρ is either a vertex of T, or the midpoint of an edge e_j of $\tilde{T}$. However the first case is impossible since ρ is distinct from r', and yet both belong to $S \cap S'$.

Hence we may assume that ρ is the midpoint of an edge e_j on $\tilde{T}$. Note that $j \neq 1$, since the midpoint of e is not contained in θ'. But then $(\tau(e), e_j)$ form a matching pair: The straight cycle S' contains r' but does not cross $\tau(e)$. If we define $\tilde{T}'$ to be the subpath of C with ending edges $\tau(e)$ and e_j which does not contain the edge

$e = e_1$, then $\tilde{T}'$ is a subpath of $\tilde{T}$, and is strictly shorter since it does not contain e_1. Hence $(\tau(e), e_j)$ contradicts the minimality of $(e, \tau(e))$. ☐

Corollary 4.2.17. *Let G and C be as in Lemma 4.2.16, and assume $C = P \cup Q$ for two paths P, Q with ends $v, v' \in V(C)$. Then there is a face $r \subseteq \theta$ such that at least one of $\partial(r) \cap P$ and $\partial(r) \cap Q$ contains a path of length two.*

Proof. By applying Lemma 4.2.16 for an arbitrary edge on C, we obtain a face $r \subseteq \theta$ with $\partial(r) \cap C$ containing a path of length at least two. If r is not as desired, then this path contains at least one of v, v' as an internal vertex. Without loss of generality assume the former, i.e. there are edges $e, f \in E(\partial(r))$ incident at v with $e \in E(P)$ and $f \in E(Q)$.

Let $S_1, S_2 \in \mathcal{S}$ be the two straight cycles containing r, where S_1 crosses through f and S_2 crosses through e. We have two possible choices for each of $\tau(e)$ and $\tau(f)$, and we choose them so that $\tau(e) \neq f$ and $\tau(f) \neq e$. Clearly $\tau(e) \neq \tau(f)$ since the cycles in $\mathcal{S}$ are edge-disjoint. Let r_e and r_f be the regions in θ incident with $\tau(e)$ and $\tau(f)$, let $\tilde{T}_e$ be the matching path on C with ending edges $e, \tau(e)$ not containing f, and let $\tilde{T}_f$ be the matching path on C with ending edges $f, \tau(f)$ not containing e (see Figure 9).

Note that the edges $e, f, \tau(e), \tau(f)$ must appear in the cyclic order $e, \tau(e), \tau(f), f$ on C: If they do not, then the order is $e, \tau(f), \tau(e), f$ and it is easy to see that $S_1 \cap S_2$ contains a second point besides r in θ, contradicting the non-degeneracy of $\mathcal{S}$. In particular, this implies that $\tilde{T}_e$ and $\tilde{T}_f$ intersect in v and at most one other vertex of C. Hence v' is an internal vertex for at most one of them, say $\tilde{T}_f$. But in that case $\tilde{T}_e$ contains neither v nor v' as an internal vertex, implying $\tilde{T}_e \subseteq P$, and applying Lemma 4.2.16 to $(e, \tau(e))$ and $\tilde{T}_e$ yields a face as desired. Similarly if v' is not internal for $\tilde{T}_f$, then $\tilde{T}_f \subseteq Q$ and applying Lemma 4.2.16 to $(f, \tau(f))$ and $\tilde{T}_f$ gives the desired face. ☐

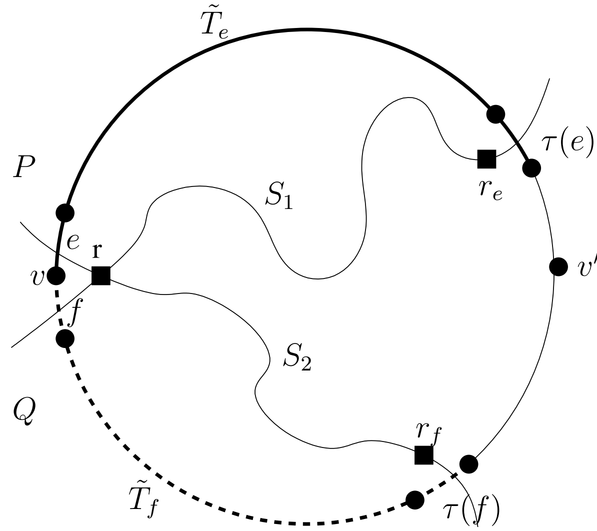


Figure 9: Configuration in proof of Corollary 4.2.17

Definition 4.2.18. A path P in a plane graph G is *good* if the following holds: Every $r \in R(G)$ is P'-nice for any path P' that can be obtained from P by moving across 2-faces (i.e. $P' = P \triangle r_1 \ldots \triangle r_k$, where r_i is a 2-face for the path $P \triangle r_1 \ldots \triangle r_{i-1}$ for all $i \geq 1$).

Lemma 4.2.19. *Let G be a plane near-quadrangulation with a non-degenerate collection of straight cycles $\mathcal{S}$, and suppose that the infinite face is bounded by a cycle $P \cup Q$, where P, Q are two good paths with ends v, v'. Then there is an order $r_1, \ldots, r_p$ of the elements of $R(G) \setminus r_\infty$, and paths $Q_0, \ldots, Q_p$ in G so that:*

(1) Q_j is a $v - v'$ path for $j = 1, 2, \ldots, p$,

(2) $Q_0 = Q$,

(3) $Q_p = P$,

(4) $Q_j = Q_{j-1} \triangle r_j$ where r_j is a 2-face for Q_{j-1}, for $j = 1, 2, \ldots, p$.

Proof. Notice that the lemma is symmetric in P and Q: If there is a sequence of regions and paths as stated moving Q onto P, then reversing the order of this sequence moves P onto Q.

We prove the lemma by induction on $|R(G)|$. If $|R(G)| = 2$, then $P \cup Q$ bounds a unique region r and so $|P \cup Q| = 4$. Since P, Q are both good, it follows that $|\partial(r) \cap P| = |\partial(r) \cap Q| = 2$ and hence $P = Q \triangle r$, as desired.

Hence assume $|R(G)| \geq 3$. Let r be the face obtained from applying Corollary 4.2.17 to the cycle $P \cup Q$. By the introductory remark, we may assume that $\partial(r) \cap Q$ contains a path of length two (for otherwise $\partial(r) \cap P$ does, and we reverse the roles of P and Q).

Since Q is good by assumption, r is a 2-face for Q. Let $Q_1 = Q \triangle r$, and let $w_1, e_1, w_2, e_2, w_3, e_3, w_4, e_4$ be the 4-cycle $\partial(r)$ so that $e_1, e_2 \in E(Q) \cap \partial(r)$. Consider the subgraph $G' = G \setminus \{w_2\}$. Then $R(G') = R(G) \setminus \{r\}$ and G' is a near-quadrangulation whose infinite face is bounded by the closed walk $P \cup Q_1$.

Note that $E(\partial(r)) \cap E(P) = \emptyset$: If say $e_3 \in E(P)$, then the straight cycle S containing r and crossing e_1 and e_3 has length two (since in that case e_1, e_3 are on the boundary of the infinite face), and since $\mathcal{S}$ is non-degenerate, it would follow that $|\mathcal{S}| = 2$, contrary to our assumption that $|R(G)| \geq 3$.

We now distinguish two cases depending on whether or not $w_4 \in V(P)$ (or equivalently whether or not $P \cup Q_1$ is a cycle).

First suppose $w_4 \notin V(P)$. Then $P \cup Q_1$ is a cycle, and contracting the (parallel) edges with ends r, r_∞ in the two straight cycles from $\mathcal{S}$ which contain r yields a non-degenerate collection of straight cycles $\mathcal{S}'$ for G'. Since $|R(G')| = |R(G)| - 1$, we can apply induction to the subgraph G' and the paths P, Q_1 (which are good in G' by definition) to obtain an order $r_2, \dots, r_p$ for the faces of $R(G) \backslash r_\infty \backslash r$, and corresponding paths Q_j for $j = 2, \dots, p$. Then setting $Q_0 = Q$, the sequence $Q_0, Q_1, \dots, Q_p$ is as desired for G.

Now suppose $w_4 \in V(P)$. Let P^1, P^2 be the two subpaths of P with ends w_4 and v, v' respectively, and let Q^1, Q^2 be the subpaths of Q_1 with ends w_4 and v, v', respectively. Then $P \cup Q_1$ is the (edge-disjoint) union of the two cycles $P^i \cup Q^i$ for

$i = 1, 2$. Let G^i be the subgraph of G' bounded by $P^i \cup Q^i$, for $i = 1, 2$. Then each G^i is a near-quadrangulation bounded by a cycle, and the non-degenerate collection $\mathcal{S}^i$ of straight cycles for G^i is naturally obtained from $\mathcal{S}$, so we can apply induction to each G^i. Let $r^i_1, r^i_2, \ldots, r^i_{p^i}$ be the order of the interior regions of $R(G^i) \subseteq R(G) \backslash r_\infty \backslash r$ which moves Q^i onto P^i in G^i, for $i = 1, 2$ (note that $R(G) \setminus r_\infty$ consists of r and the interior regions of G^1, G^2). But then if we let $Q_0 = Q$, $Q_{j+1} = Q_j \triangle r^1_j$ for $j = 1, 2, \ldots, p^1$ and $Q_{j+1+p^1} = Q_{j+p^1} \triangle r^2_j$ for $j = 1, 2, \ldots, p^2$, then it is easily checked that the sequence $Q_0, Q_1, \ldots, Q_p$ is as desired. □

Corollary 4.2.20. *Let $H \hookrightarrow \mathbb{P}$ be minor-minimal of face-width $k \geq 2$, let C_0 be a non-contractible cycle of length $2k$ in the radial graph $\mathcal{R}$ of H, and fix a vertex $v_0 \in V(C_0) \cap V(G)$. Then there exist an order of the regions of $\mathcal{R}$, say $r_{e_1}, \ldots, r_{e_m}$, and cycles $C_1, \ldots, C_m$ in $\mathcal{R}$ so that*

(1) Each C_j is noncontractible and has length $2k$, for $j = 1, 2, \ldots, m$,

(2) $C_m = C_0$,

(3) $v_0 \in V(C_j)$ for $j = 0, 1, \ldots, m$,

(4) $C_j = C_{j-1} \triangle r_{e_j}$ for $j = 1, 2, \ldots, m$,

(5) $\partial(F_j) = (C_0 \setminus C_j) \cup (C_j \setminus C_0)$ with $F_j = \{r_{e_1}, \ldots, r_{e_j}\}$ for $j = 1, 2, \ldots, m$.

Proof. Let $\mathcal{R}'$ be the graph obtained from $\mathcal{R}$ by cutting open along C_0. Then $\mathcal{R}'$ is a near-quadrangulation (since the regions of $\mathcal{R}'$ other than the infinite region correspond to the regions of $\mathcal{R}$), and the cycle C'_0 bounding the infinite face of $\mathcal{R}'$ is the union of two paths P, Q, each of which corresponds to C_0 in $\mathcal{R}$. In particular, P, Q have ends v, v' corresponding to the vertex v_0. Moreover, P and Q are good, since every path P' in $\mathcal{R}'$ obtained from either P or Q by moving across 2-faces corresponds to a non-contractible cycle of length $2k$ in $\mathcal{R}$ (see Remark 4.2.14), and so Corollary 4.2.10 implies that every region $r \in R(\mathcal{R}') \setminus r_\infty$ is P'-nice. Let $r_{e_1}, r_{e_2}, \ldots, r_{e_m}$ be the order of $R(\mathcal{R}')$ and let $Q_0, Q_1, \ldots Q_m$ be the paths obtained from applying Lemma 4.2.19. Then in $\mathcal{R}$, each of the v-v' paths Q_j corresponds to a cycle C_j containing v_0, and since C_j is obtained from C_0 by moving across 2-faces, each has length $2k$ and is non-contractible (Remark 4.2.14). Condition (5) is easily seen to hold by induction, so the sequence $C_0, C_1, \ldots, C_m$ is as desired. $\square$

We now extend the 'sweep' from the previous result to a planar double cover of H.

Corollary 4.2.21. *Let $H \hookrightarrow \mathbb{P}$ be minor-minimal of face-width $k \geq 2$, and let G be a planar double cover of H. Let $\mathcal{R}$ be the radial graph of H, and let $\mathcal{R}_G$ be the radial graph of G. Let C_0 be a noncontractible cycle of length $2k$ in $\mathcal{R}$, v_0 be a vertex on C_0, and let P_0 be a path of length $2k$ in $\mathcal{R}_G$ which is a lift of C_0. Let $v, \bar{v}$ be the complementary ends of P_0, corresponding to v_0.*
Then there exist an order of the regions of $\mathcal{R}_G$, say $r(e_1), r(e_2), \ldots, r(e_{2m})$, and paths $P_1, P_2, \ldots, P_{2m}$ in $\mathcal{R}_G$ so that

(1) Each P_i is a $v - \bar{v}$ path of length $2k$ for $i = 1, 2, \ldots, 2m$,

(2) $P_{2m} = P_0$,

(3) $P_i = P_{i-1} \triangle r(e_i)$ for $i = 1, 2, \ldots, 2m$,

(4) $\partial(F_i) = (P_0 \backslash P_i) \cup (P_i \backslash P_0)$ with $F_i = \{r(e_1), r(e_2,) \ldots, r(e_i)\}$ for $i = 1, 2, \ldots, 2m$.

Proof. Let $r_{e_1}, \ldots, r_{e_m}$ be the order of $R(\mathcal{R})$ and $C_1, C_2, \ldots, C_m$ be the cycles obtained from Corollary 4.2.20. We now use this to construct the order of the $2m$ faces of $\mathcal{R}_G$ and the paths P_i, for $1 \leq i \leq 2m$.

Suppose we have constructed the paths up to P_{i-1} with properties (1), (3), (4), for $i \leq m$, and so that P_j is a lift of C_j, for $j \leq i - 1$. In $\mathcal{R}$, let $f^1_{\mathcal{R}}, f^2_{\mathcal{R}}$ be the two edges in $E(C_{i-1})$ the boundary of the 2-face r_{e_i} for C_{i-1} which satisfies $C_i = C_{i-1} \triangle r_{e_i}$. Since P_{i-1} is a lift of the noncontractible cycle C_{i-1}, we have that P_{i-1} contains at most one edge of every pair of complementary edges. In particular, P_{i-1} contains exactly one of the two complementary copies of each of $f^1_{\mathcal{R}}$, $f^2_{\mathcal{R}}$ in $\mathcal{R}_G$, say f^h, for $h = 1, 2$. Let $r(e_i) \in R(\mathcal{R}_G)$ be the unique copy of $r_{e_i} \in R(\mathcal{R})$ which contains f^h in its boundary in $\mathcal{R}_G$, for $h = 1, 2$. Then $r(e_i)$ is a 2-face for P_{i-1}, and if we let $P_i = P_{i-1} \triangle r(e_i)$, then $(1), (3), (4)$ hold for all $j \leq i$, and P_i is a lift of C_i into $\mathcal{R}_G$.

In particular, P_m is a lift of $C_m = C_0$, so P_m is either equal to P_0 or its complementary path $\bar{P}_0$. Note that by (4), F_m contains exactly one of each of the two complementary faces of $\mathcal{R}_G$, so by the formula for $\partial(F_m)$, $P_m \neq P_0$ and hence $P_m = \bar{P}_0$. We now again use the sequence $r_{e_1}, r_{e_2}, \ldots, r_{e_m}$ to define $P_{m+1}, P_{m+2}, \ldots, P_{2m}$, by letting $P_{i+m} = P_{i-1+m} \triangle \overline{r(e_i)}$ for $i = 1, 2, \ldots, m$: It is easy to check that for a 2-face r_{e_i} with edges f^1_R, f^2_R in P_{i-1}, we have that in $\mathcal{R}_G$, the corresponding edges on P_{i+m} (which is a lift of C_i) are $\bar{f}^1, \bar{f}^2$, and therefore $\overline{r(e_i)}$ is a 2-face for P_{i+m}. Hence (1), (3), (4) hold by construction, and we have $F_{2m} = R(\mathcal{R}_G)$, implying that $P_{2m} = P_0$, as desired for (2). $\square$

It is easy to see that Corollary 4.2.21 and Corollary 4.2.20 can be used to define a linear branch-decomposition (i.e. a branch-decomposition where every internal vertex is adjacent to at least one leaf) of order $2k$ for G and H, respectively. In fact, we can use the ordering of $E(G)$ implied by Corollary 4.2.21 to construct a branch-decomposition of G/e and $G \setminus e$ for any edge $e \in E(G)$, as we show in the following lemma (we actually construct a carving-decomposition of the medial graph $\mathcal{M}_G$, which in turn gives a branch-decomposition of G).

Lemma 4.2.22. *Let $H \hookrightarrow \mathbb{P}$ be minor-minimal of face-width $k \geq 2$, and let G be a planar double cover of H. Then G/e and $G \setminus e$ have branch-width at most $2k - 1$ for any edge $e \in E(G)$.*

Proof. Let $\mathcal{R}$ be the radial graph of G, and $\mathcal{R}_H$ be the radial graph of H. Fix an edge $e \in E(G)$. Let r_e be the region of $\mathcal{R}$ corresponding to e, and let C be the cycle bounding r_e in $\mathcal{R}$. Let f_1, f_2, f_3, f_4 be the consecutive edges of C. Let G' be obtained from G by either deleting or contracting e, and let $\mathcal{R}'$, $\mathcal{M}'$ be its radial and medial graph, respectively. Notice that the radial graph $\mathcal{R}'$ of G/e and $G \setminus e$ is obtained from $\mathcal{R}$ by identifying either f_1 with f_2 and f_3 with f_4, or f_1 with f_4 and f_2 with f_3. In particular, the faces of $\mathcal{R}'$ correspond to the faces of $\mathcal{R}$ different from r_e. Without loss of generality suppose that $\mathcal{R}'$ is obtained from $\mathcal{R}$ by identifying edges f_1 with f_2 and f_3 with f_4.

By Theorem 2.3.10, it suffices to show that $cw(\mathcal{M}') < 4k$. Since H is minor-minimal of face-width k, it follows that there is a noncontractible cycle C_0 of length $2k$ in $\mathcal{R}_H$ such that $E(C \cap C_0) = \{f_1, f_2\}$ (we can draw $\mathcal{R}_H$ and $\mathcal{R}$ so neither f_1 nor f_2 are edges crossing the circle γ separating the two copies of $\mathcal{R}_H$, and hence we can also think of f_1, f_2 as being edges in (one copy of) $\mathcal{R}_H$). Fix a vertex $v_0 \in V(C_0)$ other than the one which is incident with both f_1 and f_2, and let P_0 be a path of length $2k$ in $\mathcal{R}$ which is a lift of C_0, with end-vertices $v, \bar{v}$ corresponding to v_0, and containing both f_1 and f_2 (or more precisely, one of each of the two copies of f_1, f_2 which are incident in $\mathcal{R}$). Let $P_1, P_2, \ldots, P_{2m}$ be the $v - \bar{v}$ paths in $\mathcal{R}$ obtained from Corollary 4.2.21 for P_0, let $e_1, e_2, \ldots, e_{2m}$ be the induced ordering of $E(G) = R(\mathcal{R}) = V(\mathcal{M})$, and let F_i be as in Corollary 4.2.21 for $i = 1, 2, \ldots, m$.

We have to show that the induced carving of $V(\mathcal{M}')$ (obtained from the sequence $e_1, e_2, \ldots, e_{2m}$ with e removed) has width strictly less than $4k$. Since a partition of $V(\mathcal{M}')$ corresponds to a partition of $R(\mathcal{R}')$, it suffices to show that $|\partial(F_i')| < 4k$ in $\mathcal{R}'$, where $F_i' = \{r_{e_1}, r_{e_2}, \ldots, r_{e_i}\} \setminus \{r_e\}$ for all $i = 1, 2, \ldots, 2m$.

We first claim that $|\partial(F_i')| \leq |\partial(F_i)|$: Suppose f is an edge in $\partial(F_i') \setminus \partial(F_i)$, and is incident with faces r_1, r_2 in $\mathcal{R}'$. Since $f \in \partial(F_i') \setminus \partial(F_i)$, it follows that f is one of the two edges obtained from identifying f_1 with f_2, or f_3 with f_4, say the former. But now depending on whether or not $r(e) \in F_i$ (i.e. whether $e = e_j$ for some $j \leq i$), we have that exactly one of f_1 and f_2 belongs to $\partial(F_i)$ in $\mathcal{R}$, so f is accounted for and the claim follows .

We now show that $|\partial(F_i')| < 4k$. If $|\partial(F_i)| < 4k$ we are done by the above claim, hence we may assume that $|\partial(F_i)| = 4k$. In this case since $|P_0| = |P_i| = 2k$ for all i and $\partial(F_i) \subseteq P_0 \cup P_i$ by (4) of Corollary 4.2.21, we must have that $\partial(F_i) = P_0 \cup P_i$. However by our choice of v_0, it is easy to see that $|\partial(F_i')|$ decreases (in fact by two), since we identify two edges on P_0, namely f_1 with f_2. Hence $|\partial(F_i')| < 4k$ for all $1 \leq i \leq 2m$, as desired. $\square$

The proof of the main result now follows easily from the previous results:

Proof of theorem Theorem 4.1.1.

By Lemma 4.2.3 and Theorem 2.3.12, G has branch-width at least $2k$. Since deletion or contraction of a single edge decreases the branch-width by at most one, Lemma 4.2.22 then implies that G has branch-width exactly $2k$, and moreover that G is minor-minimal. $\square$

4.3 Other minor-minimal planar graphs of fixed branch-width

As before, let $\mathcal{C}_l$ be the minor-minimal planar graphs of branch-width (exactly) l, for $l \geq 0$. We now give some examples showing that in general, the class $\mathcal{C}_{2k}$ does not consist only of double covers of minor-minimal projective planar graphs.

For $k = 3$, there are 7 non-isomorphic minor-minimal planar graphs of branch-width 6 which arise as planar double covers of the 7 minor-minimal embeddings of face-width 3 in the projective plane (the latter were determined in [8] and [71]). However there are more graphs in $\mathcal{C}_6$:

Figure 10 shows some graphs in $\mathcal{C}_6$ (these graphs were found and checked to be minimal by a computer), but none of them can be a double cover of another graph, since they all have either an odd number of vertices or edges (or both).

None of these graphs is self-dual, so we get 4 more members of $\mathcal{C}_6$ by taking the

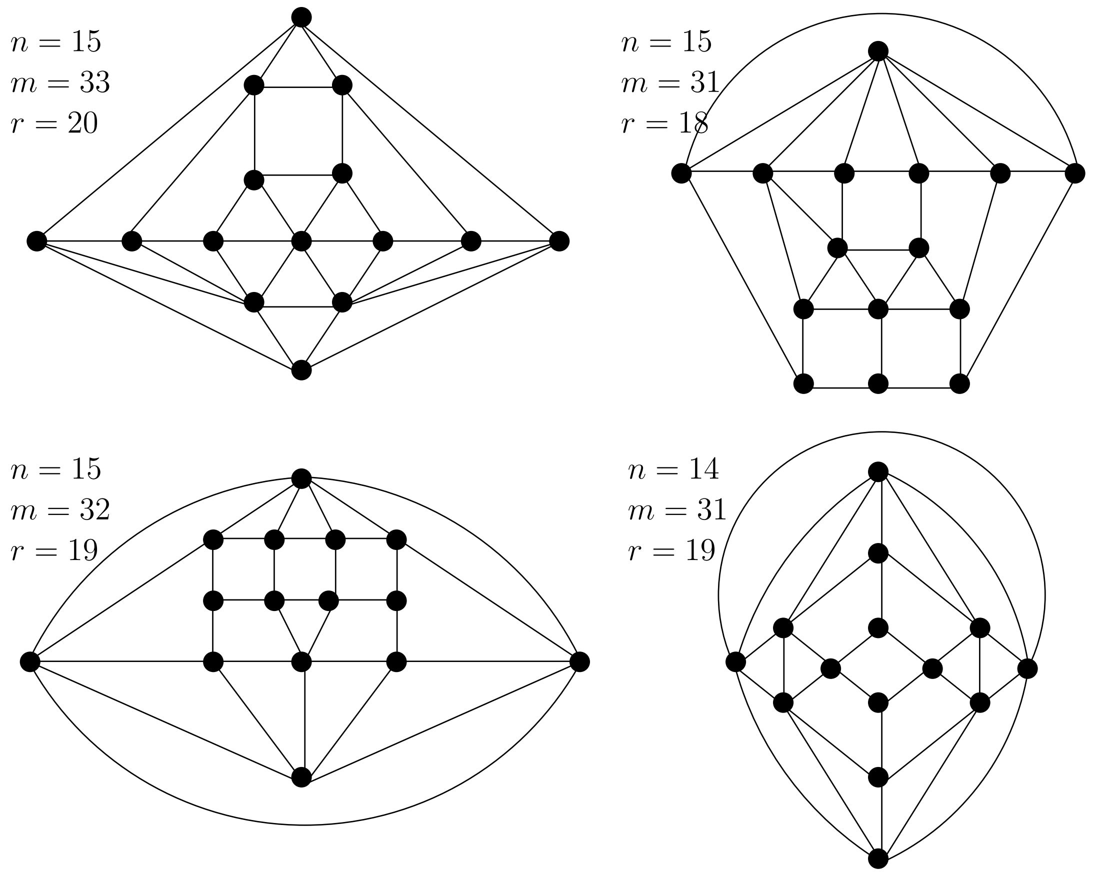


Figure 10: Minor-minimal graphs of branch-width 6 which are not double covers

planar dual of each. Hence $|\mathcal{C}_6| \geq 15$ (in fact there are probably more minor-minimal planar graphs of branch-width 6, since our computer search was not designed to be exhaustive). The graphs in Figure 10 appear not to have too much structure, and so we expect that the class $\mathcal{C}_{2k}$ will not allow an easy complete characterization in general, in particular it is not generated by taking double covers of other graphs.

Although the double cover construction discussed in this chapter always yields a graph of even branch-width, it is also interesting to look at the classes $\mathcal{C}_{2k+1}$: For every $k \geq 2$, $\mathcal{C}_{2k+1}$ contains a graph that is not the double cover of any other graph: Let $C_{k\times(2k+1)}$ be the planar $k \times (2k+1)$ circular grid with k circular rows and $2k+1$ columns. Now add a new vertex that is connected to all $2k+1$ vertices on the innermost cycle, to obtain a graph G_{2k+1} (see Figure 11).

It is easy to see that $bw(C_{k\times(2k+1)}) \leq 2k$ (in fact equality holds), and so $bw(G_{2k+1}) \leq$

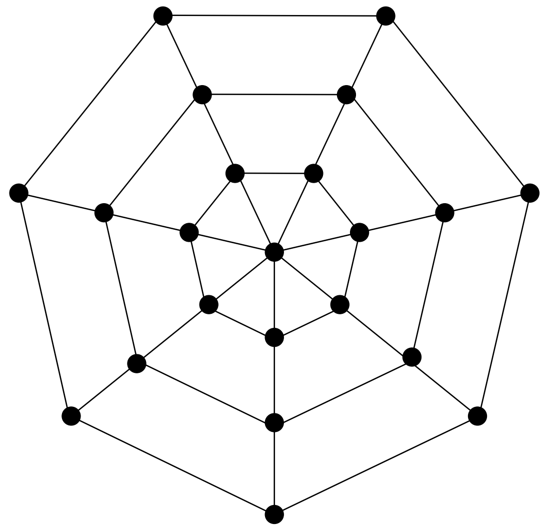


Figure 11: The graph G_{2k+1} for $k = 3$

$2k+1$. In fact one can show that G_{2k+1} has branch-width exactly $2k+1$ and is minor-minimal, i.e. $G_{2k+1} \in \mathcal{C}_{2k+1}$ for every $k \geq 2$ (G_{2k+1} is self-dual, so we do not get a second graph this way). But clearly G_{2k+1} is not a double cover of any graph, since it has a unique vertex of degree $2k+1$ for $k \geq 2$.

CHAPTER V

APPLICATIONS OF TREE-DECOMPOSITIONS TO THE TSP

In this chapter we discuss some applications of tree-decompositions to the problem of finding violated comb and DP-inequalities (for definitions, see Section 2.4). In particular, we show how to solve the comb separation problem for an arbitrary number of teeth if the input graph has bounded tree-width, and we modify an algorithm by Letchford to produce all violated simple DP-inequalities using tree-decompositions.

5.1 Separation of comb inequalities using tree-decompositions

The separation problem for comb inequalities is the following:

Problem 5.1.1 (Separation problem for comb inequalities). Given a vector $\hat{x} \in SP(n)$, find a comb inequality that is violated by $\hat{x}$, or decide that no such inequality exists.

In this section we prove the following result:

Theorem 5.1.2. *The comb separation problem can be solved in time $\mathcal{O}(n)$ if the support graph $\hat{G}$ satisfies $tw(\hat{G}) \leq \theta$ for some fixed integer θ.*

Remark 5.1.3. It is straightforward to design a linear time algorithm (for bounded tree-width) if the number of teeth is fixed to be at most some constant d: Roughly speaking, what we need to keep track of in a partial solution at a node t is to which one of the d teeth every vertex $v \in X_t$ belongs (or if it belongs to the handle or the 'outside' of the comb), and whether each of the d teeth candidates already represents a full tooth (i.e. it crosses the handle) or not. Hence the amount of information to be stored at any node t is roughly $\mathcal{O}(d^\theta \cdot 2^d)$, where θ is the width of the decomposition used. Therefore this only yields a polynomial-time algorithm if the number of teeth is bounded by a constant. Hence to prove Theorem 5.1.2, we have to make use of structural properties of violated combs.

For a tree T of a tree-decomposition of a graph G, we define the the following vertex sets and subgraphs for each node $t \in V(T)$ (see Figure 12):

$$Y_t = \bigcup_{s \text{ is a descendent of } t} X_s. \tag{5.1.1}$$

$$Z_t = V(G) \setminus Y_t \setminus X_t, \tag{5.1.2}$$

$$G_t = G[Y_t \cup X_t], \tag{5.1.3}$$

$$G'_t = G[Z_t \cup X_t]. \tag{5.1.4}$$

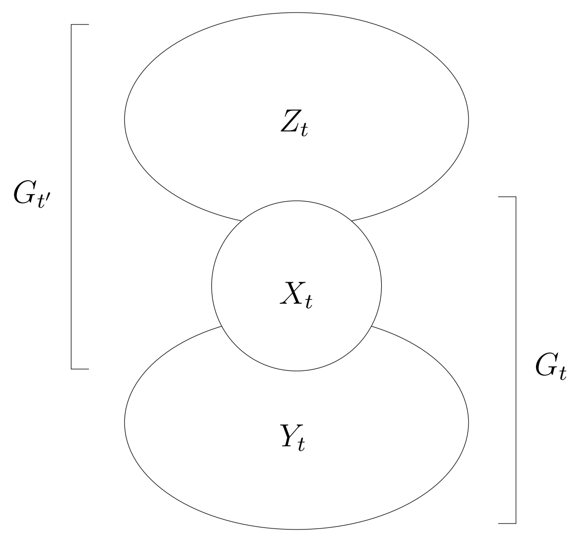


Figure 12: Vertex sets and subgraphs at a node t in a tree-decomposition

In the following, we will use G to denote the support graph $\hat{G}$ for a solution $\hat{x} \in SP(n)$.

Before we describe the algorithm and its proof of correctness in full detail, we give a sketch of the method to illustrate the overall idea. Suppose that we have a nice tree decomposition $(T, (X_t)_{t \in V(T)})$ (see Definition 2.3.6) of G available, i.e. in particular T is a rooted tree and each node has at most two children and is of one of the four types LEAF, INTRO, JOIN, FORGET. The algorithm we describe follows the standard dynamic programming principle typically used in algorithms based on tree-decompositions: At each node t, we will store a table of characteristics of partial solutions which can potentially be extended to a full solution. After computing all these tables in a bottom-up manner for T, the information computed at the root indicates whether there is a full solution or not. This technique was first used for tree-decompositions in [5] and [9].

In our case, a partial solution for a node t will be a 'partial comb', i.e. a partition of $V(G_t)$ (into a handle, teeth, and vertices outside of the comb), together with an associated value computed analogous to the violation of a comb inequality (for rigorous definitions, see below). In particular, a 'partial comb' can have 'partial teeth' (A, B) with $A = \emptyset$ or $B = \emptyset$ on G_t. For a given node $t \in V(T)$, we can partition the partial solutions into equivalence classes, where two partial solutions in G_t are equivalent if they 'agree' on X_t with respect to the above mentioned partition. Each partial solution is then represented by a *characteristic*, which is identical for equivalent partial solutions. In order for this framework to give the desired linear time algorithm, we need to show it suffices for each equivalence class at node t to store a characteristic of minimum value ('violation') - this is done in Lemma 5.1.6. The key to this is that at a node t, if a characteristic corresponds to a partial solution which extends to a proper violated comb in G, then every 'partial tooth' must have a non-empty intersection with the bag X_t - this will follow from an observation about the structure of violated combs (Lemma 5.1.4). Therefore it suffices to keep track only of characteristics corresponding to partial solutions with this intersection property (such partial solutions will be called 'promising'), and it follows that there are only a constant number of characteristics to store at each node, leading to a linear time algorithm.

We now proceed to carefully define the above-mentioned notions, and prove that the characteristics we define are indeed sufficient to solve the comb separation problem. In our presentation, we use the terminology from [14], which contains a description of the general technique of using dynamic programming on a tree-decomposition.

A *solution* to Problem 5.1.1 is a collection of sets $\mathcal{C} = \{H, T_1, ..., T_d\}$ (for some odd integer $d \geq 3$) representing a comb, together with a (negative) value λ representing the surplus of the violated comb inequality, with respect to the given vector $\hat{x}$ (recall that a comb inequality is violated if and only if its surplus is negative).

A *labelled partition* of a set $X \subseteq V(G)$ is a partition of X into sets $H, R, A_1, B_1, ..., A_k, B_k$ (for some $k \geq 0$), where some of the sets may be empty, except not both A_i and B_i for a given $i \leq k$. The set R is called the *rest* and $H \cup A_1 \cup ... \cup A_k$ is called the *handle*. A pair (A_i, B_i) is called a *partial tooth* if exactly one of A_i, B_i is empty, and a *full tooth* if both A_i, B_i are non-empty. We write T_i for $A_i \cup B_i$. The *contribution* of an edge $e \in E(G[X])$ (for the labelled partition of X) is defined as

$$x_e \cdot (\mathbb{1}_{\delta(H \cup A_1 \cup ... \cup A_k)}(e) + \sum_{i=1}^{k} \mathbb{1}_{\delta(T_i)}(e)),$$

where $\mathbb{1}_A(\cdot)$ denotes the indicator function of a set A, and all cuts in the definition are in $G[X]$. The *value* λ of the labelled partition is the sum of the contributions of all edges in $E(G[X])$, minus $3k+1$. Notice that if $X = V(G)$, $k \geq 3$ is odd, and all sets A_i, B_i are non-empty, then the labelled partition defines a comb of surplus λ.

A *partial solution* ϕ (at a node t) is a labelled partition of $V(G_t)$, together with its value λ and the integer k, the number of (full or partial) teeth. We assume that the teeth (A_i, B_i) of ϕ are ordered in some canonical way (for a fixed node t), so that partial solutions which are identical up to renumbering of the teeth receive the same labels.

A partial solution is *extendible* if there is a violated comb $\mathcal{C}' = \{H', T'_1, ..., T'_d\}$,

$(d \geq k)$ so that

$$H \subseteq H', \tag{5.1.5}$$

$$R \subseteq V(G) \setminus (H' \cup T_1' \cup ... \cup T_d'), \tag{5.1.6}$$

$$B_i \subseteq T_i' \setminus H' \; 1 \leq i \leq k, \tag{5.1.7}$$

$$A_i \subseteq T_i' \cap H' \; 1 \leq i \leq k. \tag{5.1.8}$$

We now define a notion of equivalence between partial solutions. Given a partial solution ϕ, its *characteristics* ξ consists of

(1) the labelled partition induced by ϕ on X_t,

(2) an additional boolean label *partial* for each tooth of ϕ intersecting X_t, indicating whether it is full or partial in ϕ,

(3) a boolean label $\beta = k \mod 2$, indicating whether the number of teeth is odd or even.

Two partial solutions ϕ, ϕ' are *equivalent* if their characteristics are identical.

Suppose $\phi = (\{H, R, A_1, B_1, ..., A_k, B_k\}, \lambda)$ is a partial solution at t, and $\psi = \{H', R', A_1', B_1', ..., A_{k'}', B_{k'}'\}$ is a labelled partition of a set $X \subseteq V(G_{t'})$ which satisfies $X \cap V(G_t) = X_t$. If ψ agrees with ϕ on X_t we write $\phi \oplus \psi$ to denote the partial solution induced by the labelled partition ϕ and ψ on $V(G_t) \cup X$, together with its value λ and the number of teeth.

The following lemma, observed by Letchford in [50], is crucial to the correctness of our approach. Recall that that comb inequalities form a special case of DP inequalities, where the dominos are called teeth. Following Letchford, we say a set $A \subseteq V(G)$ is connected if $G[A]$ is connected.

Lemma 5.1.4. *Suppose (A, B) is a domino of a DP inequality violated by $\hat{x} \in SP(n)$. Then the sets $A, B, A \cup B$ and their complements in $V(\hat{G})$ are connected in $\hat{G}$.*

Proof. We only prove this for A, since the proof for the other sets is analogous. Suppose A is not connected, i.e. A is the disjoint union of two non-empty sets A_1, A_2 with $E(A_1, A_2) = \emptyset$. Then $x(E(A_1, A_2)) = 0$ and since $x \in SP(n)$, we have

$$\hat{x}(\delta A) = \hat{x}(\delta A_1) + \hat{x}(\delta A_2) \geq 4 \tag{5.1.9}$$

i.e. the subtour constraint for $\delta(A)$ has surplus at least two. Hence the domino inequality (2.4.10) for (A, B) has surplus at least one, which is impossible by Proposition 2.4.6 if (A, B) is used in a violated DP inequality. □

Corollary 5.1.5. *Suppose (A, B) is a partial tooth in an extendible partial solution ϕ at t. Then $X_t \cap (A \cup B) \neq \emptyset$.*

Proof. Without loss of generality $B = \emptyset$, and $A \neq \emptyset$. Consider a violated comb extending ϕ and let (A', B') be the tooth which satisfies $A \subseteq A'$. Now $A' \cap V(G_t) = A \neq \emptyset$ and $B' \cap V(G_t) = B = \emptyset$ since the comb is an extension of ϕ. Hence $B' \subseteq Z_t$. Since $A' \cup B'$ is connected by Lemma 5.1.4, there is an $A' - B'$ edge in G. Since X_t is a cutset separating Y_t from Z_t, and A' is connected by Lemma 5.1.4, it follows that $A' \cap X_t \neq \emptyset$, and hence $A \cap X_t \neq \emptyset$, as desired. □

A partial solution ϕ at t is called *promising* (at t) if every partial tooth of ϕ has non-empty intersection with X_t. Hence Corollary 5.1.5 says that every extendible partial solution at t is promising at t.

The following lemma will be the key to showing that it suffices to keep only one representative per equivalence class of characteristics at any node t, namely one with minimum value λ.

Lemma 5.1.6. *Suppose ϕ, ϕ' are two equivalent partial solutions at t with $\lambda_\phi \leq \lambda_{\phi'}$. Then if ϕ' is extendible and ϕ is promising, then ϕ is extendible.*

Proof. Suppose $\phi' \oplus \psi$ is a partial solution (at the root) extending ϕ' which corresponds to a violated comb, so that $\lambda_{\phi' \oplus \psi} < 0$. We claim that $\phi \oplus \psi$ induces a comb with $\lambda_{\phi \oplus \psi} \leq \lambda_{\phi' \oplus \psi}$, i.e. of greater or equal violation.

Claim 5.1.6.1. $\phi \oplus \psi$ has no partial teeth.

Proof of claim. Suppose $T_i = (A, \emptyset)$ is a partial tooth of $\phi \oplus \psi$. Then $A \cap V(G_t) \neq \emptyset$: If $A \subseteq Z_t$, then there is a full tooth (A', B) of $\phi' \oplus \psi$ with $A' \cap (X_t \cup Z_t) = A$ and $B \subseteq Y_t$, and so $A' \cup B$ would be disconnected, contrary to Lemma 5.1.4. But if $A \cap V(G_t) \neq \emptyset$ then there is a partial tooth $(A \cap V(G_t), \emptyset)$ in ϕ (since ϕ and $\phi \oplus \psi$ agree on $V(G_t)$). Since ϕ is promising, $(A \cap V(G_t)) \cap X_t \neq \emptyset$, i.e. we have $A \cap X_t \neq \emptyset$. Since the labelled partitions (on X_t) of ϕ and ϕ' are identical, we have that $\phi' \oplus \psi$ contains a (full) tooth $T'_i = (A', B')$ with $A \subseteq A'$ and $B' \cap X_t = \emptyset$. Since the *partial* labels of ϕ and ϕ' are identical on X_t, we have that $(A' \cap V(G_t), B' \cap V(G_t))$ is partial for ϕ', i.e. $B' \cap V(G_t) = \emptyset$, and so $B' \subseteq Z_t$. But then A and B' form the tooth T_i of $\phi \oplus \psi$ and hence $B' = \emptyset$, contrary to the fact that $T'_i = (A', B')$ was full. □

To compute $\lambda_{\phi' \oplus \psi}$ from $\lambda_{\phi'}$, we have to add the contribution of the edges of $E(Z_t)$ plus those from $E(X_t, Z_t)$, and subtract the number of teeth T_i of $\phi' \oplus \psi$ entirely contained in Z_t. Since ϕ and ϕ' agree on X_t, and since $\phi' \oplus \psi$ agrees with $\phi \oplus \psi$ on $Z_t \cup X_t$, we have $\lambda_{\phi \oplus \psi} - \lambda_{\phi} = \lambda_{\phi' \oplus \psi} - \lambda_{\phi'}$, and so $\lambda_{\phi \oplus \psi} \leq \lambda_{\phi' \oplus \psi}$ since $\lambda_{\phi} \leq \lambda_{\phi'}$. In particular we have $\lambda_{\phi \oplus \psi} < 0$. It is easy to check that the parity of the number of teeth of $\phi \oplus \psi$ and $\phi' \oplus \psi$ is identical, and hence odd. Notice that $k \neq 1$ since $\lambda_{\phi \oplus \psi} < 0$ and it is easy to see that a 'comb' with a single tooth would violate a subtour constraint (and we assumed $\hat{x} \in SP(n)$). It follows that $\phi \oplus \psi$ corresponds to a (proper) violated comb as well, and so ϕ is extendible, as desired. □

5.1.1 The algorithm

We only focus on deciding whether there exists a violated comb inequality in G or not. It is straightforward to extend the algorithm so that it actually computes a violated comb if one exists.

The algorithm can be described as follows:

INPUT: The support graph $G = \hat{G}$ for a vector $\hat{x} \in SP(n)$, satisfying $tw(G) = \theta$ for some constant θ.

OUTPUT: YES if there is a comb inequality violated by $\hat{x}$, and NO otherwise.

ALGORITHM:

(1) Compute a nice tree-decomposition $(T, (X_t)_{t \in V(T)})$ of width $tw(G)$ for G.

(2) For each $t \in V(T)$ compute a table of characteristics, using the characteristics stored at its child node(s). For each distinct characteristic ξ computed at t we store a triple (ξ, λ, k), as described in the following four cases:

(2a) At a LEAF node t, generate all possible labelled partitions of X_t, compute k and λ for each and store a triple (ξ, λ, k) of minimum value λ for each characteristic ξ.

(2b) At a FORGET node t, the characteristics stored at the child node t' induce characteristics for t, and we store a triple (ξ, λ, k) of minimum value λ for those ξ in which every partial tooth intersecting $X_{t'}$ also intersects X_t.

(2c) At an INTRO node t, for every characteristic stored at the child node t' compute all possible extensions of the labelled partition into X_t, and again store a triple (ξ, λ, k) of minimum value for the characteristics at t obtained in this way.

(2d) At a JOIN node t with children t_1 and t_2, we pair each characteristic from t_1 with those characteristics from t_2 whose labelled partitions are identical on $X_t = X_{t_1} = X_{t_2}$ (note that in particular the *partial* labels are not required to be identical). We compute the characteristics at t arising this way and again we store a triple (ξ, λ, k) of minimum value λ.

(3) Output YES if and only if there is a characteristic ξ at the root node $\lambda < 0$, $\beta = 1$ (i.e. $k = 1 \mod 2$) and all *partial* labels set to FALSE.

We now prove that Algorithm 5.1.1 works correctly and can be implemented to run in linear time. This will imply Theorem 5.1.2.

Proof of Theorem 5.1.2. It suffices to show that algorithm Algorithm 5.1.1 correctly decides (in linear time) whether there is a violated comb or not: If the output is YES, then it is straightforward to actually construct a violated comb in linear time using the characteristics stored at the nodes (using the standard approach for constructing solutions in a tree-decomposition based algorithm).

Clearly if the algorithm outputs YES, then there is a comb with $\lambda < 0$, i.e. a violated comb (note that $k \neq 1$ if $\lambda < 0$ and $\hat{x} \in SP(n)$, as discussed earlier).

Conversely, we will show that if there is a violated comb, then the algorithm will output YES. This will follow from the next two claims.

Claim 5.1.6.2. If the algorithm stores (ξ, λ, k) at t, then there exists a promising partial solution ϕ at t with k teeth, value λ and characteristic ξ.

Proof of claim. Clearly this holds for LEAF nodes. Suppose the statement holds at the children t_i at a node t. Then if t is an INTRO or JOIN node, the statement holds since $X_{t_i} \subseteq X_t$ in those cases. Hence suppose t is a FORGET node. If the algorithm stores (ξ, λ, k) at t, then by (2b) there is a triple (ξ_1, λ, k) at the child t_1 in which every partial tooth of ξ_1 also intersects X_t. Hence there is a promising ϕ_1 at t_1 for ξ_1, and since ξ is the restriction of ξ_1 to X_t, the claim follows. ☐

Claim 5.1.6.3. If there is a violated comb in G, then for every $t \in V(T)$, the algorithm stores some triple (ξ, λ, k) at t for which there exists an extendible partial solution ϕ at t with k teeth, value λ and characteristic ξ.

Proof of claim. Clearly the statement holds if t is a LEAF node, so suppose that the algorithm stores a triple (ξ_i, λ_i, k_i) at for each child t_i of t corresponding to an extendible partial solution ϕ_i.

If t is an INTRO node, then some extension of the labelled partition of ξ_1 to X_t will correspond to an extension of ϕ_1, and hence the characteristic ξ stored by the algorithm will be as desired.

If t is a JOIN node, the algorithm stores the characteristic ξ of the partial solution $(\phi_1 \oplus \phi_2)$ at t, say with values λ^* and k^*. We claim that (ξ, λ^*, k^*) is as desired. Let $(\xi_1^*, \lambda_1^*, k_1^*)$, $(\xi_2^*, \lambda_2^*, k_2^*)$ be the triples from t_1, t_2 which caused the algorithm to store the triple (ξ, λ^*, k^*) at t. By Claim 5.1.6.2, there are promising partial solutions ϕ_1^*, ϕ_2^* at t_1, t_2 for those two triples. Their union $\phi_1^* \oplus \phi_2^*$ is a partial solution which is promising at t, equivalent to $\phi_1 \oplus \phi_2$ and satisfies $\lambda_{\phi_1^* \oplus \phi_2^*} \leq \lambda_{\phi_1 \oplus \phi_2}$. Hence by Lemma 5.1.6, $\phi_1^* \oplus \phi_2^*$ is extendible (since it is easy to check that $\phi_1 \oplus \phi_2$ is extendible), as desired.

Finally if t is a FORGET node, let ϕ be the restriction of ϕ_1 to X_t, and ξ its characteristic at t. Now ϕ is extendible since ϕ_1 is extendible, and by Corollary 5.1.5, ϕ is promising at t. Hence the algorithm stores some triple (ξ, λ^*, k^*) at t in step (2b). Hence there is some triple $(\xi_1^*, \lambda^*, k^*)$ stored at t_1 (with the restriction of ξ_1^* to X_t being ξ), and by Claim 5.1.6.2 we get a promising partial solution ϕ_1^*. Then ξ and the restriction ϕ^* of ϕ_1^* to X_t are as desired: Since ϕ^* is equivalent to ϕ and promising for t (as its characteristic ξ was selected in step (2c)), ϕ is extendible and $\lambda_{\phi^*} \leq \lambda_\phi$, Lemma 5.1.6 implies that ϕ^* is extendible as well. $\square$

Since Claim 5.1.6.3 holds in particular for the root node t_0, the partial solution ϕ obtained from the claim (whose characteristic ξ is stored at t_0) will be a proper violated comb, causing the algorithm to output YES since the three conditions of (3) are satisfied.

For the runtime of Algorithm 5.1.1, notice that all steps can be implemented to run in time $\mathcal{O}(n)$:

For step (1), we can use an algorithm of Bodlaender [13] to find a tree-decomposition of optimal width since G has bounded tree-width. This tree-decomposition can then easily be converted to a nice tree-decomposition in linear time - see e.g. [65].

For the four cases (2a)-(2d), notice that the number of equivalence classes of characteristics possible at each node t is bounded by $((2\theta+4)\cdot 2\cdot 2)^{\theta+1}$, since $|X_t| \leq \theta+1$ and there at most $2(\theta+1)+2$ possible labels $H, R, A_1, B_1, ..., A_k, B_k$ for any vertex of X_t. Also note that the characteristics at t together with their values λ and k can be computed from the information stores at its child nodes and the graph $G[X_t]$ in constant time.

Since T has $\mathcal{O}(n)$ nodes, it follows that the algorithm can be implemented in time $\mathcal{O}(n)$. □

Since we always store characteristics of minimum value, the method will actually produce a comb that has maximal violation (i.e. minimum surplus) among all combs in G. However the violation need not be one, unlike the method of [33] which finds maximally violated combs in planar support graphs.

Unfortunately it does not appear to be the case that Algorithm 5.1.1 can be turned into a practical algorithm, even for small values of $tw(G)$. In fact even restricting the method to searching for violated combs with exactly three teeth (and using some straight-forward techniques to reduce the amount of characteristics considered) does not seem to be practical without further ideas on how to dramatically reduce the number of characteristics stored.

5.2 Separating simple DP inequalities using tree-decompositions

Recall that a DP inequality (defined in 2.4.4) is given by dominos $\{(A_i, B_i)\}$ with $A_i, B_i \subseteq V(\hat{G})$ for $i = 1, \ldots, d$ with $d \geq 3$ odd, and a set $F \subseteq E(\hat{G})$, where $\hat{G}$ denotes the support graph of a given $\hat{x} \in SP(n)$. In analogy to the more specific class of comb inequalities, a domino is also sometimes called a tooth.

It is not known in general if the separation problem for DP inequalities can be solved in polynomial time, or whether it is NP-hard. A polynomial-time separation routine for the case that $\hat{G}$ is planar was developed in [50].

In [34], the authors give a polynomial-time separation method for so-called simple DP inequalities.

A domino (A, B) is called *simple* if $|A| = 1$ or $|B| = 1$, or both. A *simple DP inequality* is a DP inequality in which all of its dominos are simple. Hence simple DP inequalities are a natural generalization of the 2-matching inequalities from [31].

The algorithm of [34] is fairly involved, both conceptually and computationally, and as described it only produces a single violated simple DP inequality, if one exists.

In this section we show how one can use tree-decompositions to overcome some of these difficulties if the tree-width of the involved graphs is bounded. In Section 5.2.1 we explain how to separate all (or if desired a large selection of) simple DP inequalities using tree-decompositions (assuming low tree-width for the graphs involved), and in Section 5.2.2 we provide some computational results on the effectiveness of (a subclass of) simple DP inequalities for the TSP. To our knowledge, no implementation of the algorithm of [34] is available and no computational studies on its effectiveness have been conducted.

5.2.1 The algorithm

We first give an overview of the separation algorithm from [34], and then show how to modify it using tree-decompositions. For a detailed description of the original algorithm and in particular a full proof of correctness, we refer the reader to [34].

If $(i, S) = (\{i\}, S)$ is a simple domino (where $S \subseteq V(\hat{G})$, and $i \in V(\hat{G}) \setminus S$, the vertex i is called the *root* of the domino (i, S), and S the *body* of (i, S) (if $S = \{j\}$ for some $j \neq i$, then both i and j can serve as body or root). The associated *simple domino inequality* is inequality 2.4.10 for $(A, B) = (i, S)$ and can be rewritten as

$$x(\delta(S)) - x(E(\{i\}, S)) \geq 1, \tag{5.2.1}$$

since $x(\delta(i)) = 2$. Note that in their description, the authors of [34] use the '$\leq$' form of DP inequalities rather than the equivalent '$\geq$' form used in this thesis, and also use the term 'tooth' instead of 'domino', in analogy to the more specific comb inequalities. By Theorem 2.4.6 a (simple) DP inequality is violated if and only if the sum of the surplus on each inequality used in the derivation (both domino inequalities and non-negativity constraints) is < 1.

In Theorem 2.4.5, we derived DP inequalities by adding domino inequalities, degree-constraints and non-negativity constraints $x_e \geq 0$, and then adding one to the (odd) right-hand side. Equivalently, we could have added those inequalities with factors $1/2$, and in the end rounded the right-hand side. An inequality obtained by adding valid inequalities with coefficient $1/2$ and rounding is called a $\{0, 1/2\}$-*cut* (see [17]). The main idea of [34] is the following: If a subclass of $\{0, 1/2\}$-cuts satisfies certain conditions, then one can construct an auxiliary graph in which a violated inequality corresponds to an odd cut of value equal to one minus its violation, and hence the separation problem can be solved by computing a minimum odd cut in that graph.

Since the class of simple DP inequalities does not quite satisfy those conditions in general, the authors first focus on simple DP inequalities where every domino inequality has surplus $< 1/2$. For a fixed vertex i, define the weight $w(S)$ of the body S (with respect to i) to be the surplus of the domino inequality given by i and S, and for an edge $e = ij \in V(\hat{G})$, define its weight $w(e)$ to be $\hat{x}_e$. Hence a simple *DP* inequality is violated if and only if the sum of the weights of the used bodies and non-negativity constraints from $e \in F$ is < 1. Say a domino (i, S) is *light* if $w(S) < 1/2$, and in that case the body S is called i-*light*. It turns out that if one only allows light dominos, then the conditions mentioned above hold and one can construct an auxiliary graph as follows: As shown in [34], for a fixed $i \in V(\hat{G})$, the i-light bodies form a laminar family (after eliminating bodies yielding equivalent domino inequalities). Let

F_i be the rooted forest representing the laminar decomposition of the family of i-light bodies, where the body S is a child of S' if and only is $S \subseteq S'$.

Then one can construct a new weighted graph $\mathcal{G}_{1/2}$ as follows: For each i, connect the roots of each subtree of F_i to a new vertex v_i to get a tree T_i rooted at v_i, and connect all vertices $v_1, ..., v_n$ (where $n = |V(\hat{G})|$) to a new vertex v^*. For each body S of a root i, set the weight of the unique edge to its parent in T_i to be $w(S)$, and label those edges as odd. Define the weight of the edges v^*v_i to be 0, and label them as even. Finally, for each $e = ij \in E(\hat{G})$, let w_j^i be the unique highest vertex in the rooted tree T_i so that no body S in the subtree rooted at w_j^i contains the vertex j (notice that w_j^i exists and is unique because the bodies are laminar sets, and possibly $w_j^i = v_i$). Then we add a new (even) edge $w_j^i w_i^j$ of weight $w(e)$ to $E(\mathcal{G}_{1/2})$. This completes the construction of $\mathcal{G}_{1/2}$ (see Figure 13).

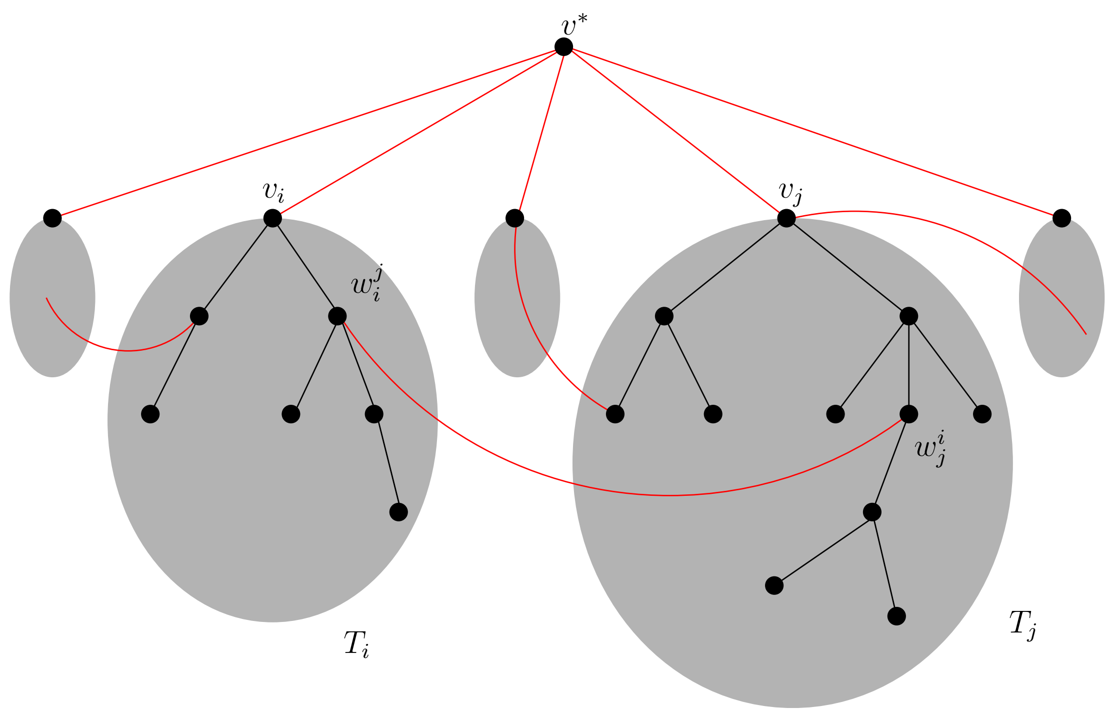


Figure 13: The graph $\mathcal{G}_{1/2}$ (odd edges are black, even edges are red)

Then it can be shown that every simple light DP inequality of violation $1 - \theta$ corresponds to an odd cut (defined as containing an odd number of edges) in $\mathcal{G}_{1/2}$ of weight θ, and vice versa. This is a special case of a more general result on $\{0, 1/2\}$-cuts from [17]. Hence to solve the separation problem for simple light DP inequalities, one needs to construct $\mathcal{G}_{1/2}$ and compute a minimum odd cut, which is exactly what is done in the basic version of the algorithm of [34].

If one also wants to include *heavy* dominos (i.e. dominos with surplus in $[1/2, 1)$), then the authors show that this can be done by a similar construction. For a fixed heavy domino (i, S), remove the forest F_i from $\mathcal{G}_{1/2}$ and instead add a new odd edge at v_i of weight $w(S)$, to obtain a new graph $\mathcal{G}_{(i,S)}$. Again one can show that the odd cuts of $\mathcal{G}_{(i,S)}$ correspond to either a simple light DP inequality, or a simple DP inequality using the unique heavy domino (i, S).

Hence to get a complete separation algorithm for the set of simple DP inequalities, one first computes a minimum odd cut in $\mathcal{G}_{1,2}$, and then a minimum odd cut in $\mathcal{G}_{i,S}$ for each heavy domino (i, S).

In fact this leads to the following template to compute all (non-equivalent) violated simple DP inequalities (two DP inequalities are equivalent if one can be obtained from the other by replacing one or more dominos (A, B) by $A, V(\hat{G}) \setminus (A \cup B))$:

INPUT: The support graph $\hat{G}$ for a vector $\hat{x} \in SP(n)$

OUTPUT: All (or one) non-equivalent violated simple DP inequalities in $\hat{G}$

(1) Compute the graph $\mathcal{G}_{1/2}$.

(2) Compute all (or one) odd cut(s) of value < 1 in $\mathcal{G}_{1/2}$.

(3) For each heavy domino (i, S):

(4) Compute the graph $\mathcal{G}_{i,S}$.

(5) Find all (or one) odd cut(s) of value < 1 in $\mathcal{G}_{i,S}$.

(6) Return all (or one) non-equivalent simple DP inequalities obtained from the cuts in (2) and (5).

In order to analyze the runtime, we need to specify how to find the set of bodies needed to construct the graphs $\mathcal{G}_{1,2}$ and $\mathcal{G}_{i,S}$, and how to find odd cuts of small value in those graphs. In [34], the authors only aim at finding one violated inequality, so they simply need to compute a minimum odd cut in each of the graphs, e.g. using the algorithm from [57] which runs in time $\mathcal{O}(n^2 m \log(n^2/m))$ on a graph with n vertices and m edges. In order to construct the graphs $\mathcal{G}_{1/2}$, one needs to generate all dominos of weight < 1, the 'candidate dominos'. For this part it suffices to generate all $X \subseteq V(\hat{G})$ with $x(\delta(X)) < 3$, which can be done in time $\mathcal{O}(nm(m + n \log n))$ (where $n = |V(\hat{G})|$ and $m = |E(\hat{G})|$) using the standard algorithm of [54] (there are only $\mathcal{O}(n^2)$ of such sets by [39]). It follows that the number of possible dominos is at most $\mathcal{O}(n^3)$, as shown in [34] (each X can be a body S or $X = S \cup \{i\}$).

In this basic form, the authors give a total runtime bound of $\mathcal{O}(n^9 \log n)$, based on the fact that loop (3) is executed $\mathcal{O}(n^3)$ times, and the fact that the graphs $\mathcal{G}_{1/2}$ and $\mathcal{G}_{i,S}$ have $\mathcal{O}(n^2)$ edges and vertices. Since this version is impractical, the authors present several ways how to reduce the amount of computation needed. In particular, they show that one can reduce the number of light and heavy dominos that need to be considered, and they show one can modify $\mathcal{G}_{1/2}$ to handle several (but not all) heavy dominos rooted at the same vertex i simultaneously. In particular only a total of $\mathcal{O}(m)$ graphs similar to $\mathcal{G}_{i,S}$ are needed in this version to deal with heavy dominos (as opposed to $\mathcal{O}(n^3)$ before), and the modified versions of both $\mathcal{G}_{1/2}$ and $\mathcal{G}_{i,S}$ have only $\mathcal{O}(n)$ vertices and $\mathcal{O}(m)$ edges, leading to a total runtime of $\mathcal{O}(n^2 m^2 \log(n^2/m))$ to find a violated simple DP inequality.

Now suppose we want to generate all violated simple DP inequalities. From the discussions, the two critical points in any implementation of scheme 5.2.1 are to find all cuts of weight < 3 in $\hat{G}$, and to generate all odd cuts of value < 1 in certain auxiliary graphs.

Both of these problems can easily be solved using tree-decompositions (assuming the tree-width of the respective input graph is bounded). Computing a minimum (or maximum) in linear time is straightforward if a tree-decomposition (of bounded width) is given: Simply traverse the tree in bottom-up fashion, and each node t generate a full set of characteristics (corresponding to cuts in the graph $G_t = G[X_t]$, where as before X_t is the bag at t) from the characteristics stored at the children of t. We omit further details since this is exactly the standard dynamic programming approach used for tree-decomposition based algorithms (see e.g. [11], or the exposition in the previous section for an example). Computing a minimum odd cut is just as easy, the only change being that we keep two otherwise equivalent copies of characteristics at t, corresponding to the best even and odd cut on G_t with the given characteristic on X_t. Notice that in this standard approach it makes no difference whether we are looking for a cut of minimum value, or a cut bounded above by a certain value, as in the application here.

Having traversed the tree in bottom-up fashion, it is also straightforward to generate all cuts of value < 3 (or any other value). Let $\mathcal{C}_3$ be the set of minimum cuts $\delta(X)$ of value < 3 in a graph G, let $c_3 = |\mathcal{C}_3|$ and let T be the tree of a tree-decomposition of bounded width. Then one can generate all members of $\mathcal{C}_3$ in time $\mathcal{O}(n \cdot c_3)$, where $n = |V(G)|$ (note that $|E(G)| = \mathcal{O}(n)$ since G has bounded tree-width). To do so, simply traverse T top-down and use all characteristics stored at a node t with value < 3 to assemble cuts. More precisely, suppose ϕ is a partial solution on $G \setminus (V(G_t) \setminus X_t))$ (i.e. the graph G'_t with the notation of the previous section) that corresponds to a characteristic ξ at t. Then we extend ϕ with all characteristics ξ' stored at a child t' of t which are compatible with ξ on X_t (i.e. with all ancestors ξ' of ξ at t'), and where $\xi'.val + \xi.val$, minus the weight of the edges in $X_t \cap X_{t'}$ that are cut-edges for ϕ, is < 3. If this quantity is < 3, then there is at least one extension of ϕ to a cut in all of G of precisely this quantity, and so any partial solution we are building in this top-down process will eventually be extended to a cut of value < 3 in G. Hence it follows that we can build all of these cuts in time $\mathcal{O}(n \cdot c_3)$, i.e. in time

$\mathcal{O}(n^3)$ since $c_3 = \mathcal{O}(n^2)$ by [39].

The method to compute all odd cuts of value < 1 is analogous. In fact, this approach of generating all solutions within a factor of the optimal solution works for any problem which can be solved with the standard dynamic programming method for using tree-decompositions.

Hence if we assume that the tree-width of the graphs $\mathcal{G}_{1/2}$ and $\mathcal{G}_{i,S}$ is bounded, using tree-decompositions reduces the runtime of the basic version of the algorithm of [34] from $\mathcal{O}(n^9 \log n)$ to $\mathcal{O}(n^5 \log n)$, and if we only consider light dominos, we have a reduction from $\mathcal{O}(n^6 \log n)$ to $\mathcal{O}(n^3)$. Moreover, generating *all* violated simple DP inequalities (i.e. all odd cuts of value < 1 in the appropriate graphs) only adds a factor equal to the number of violated DP inequalities to the respective runtimes using tree-decompositions.

In the improved version of Algorithm 5.2.1 discussed in the second part of [50], the time to find a violated simple light DP inequality is now dominated by the time it takes to reduce the set of candidate dominos to $\mathcal{O}(n)$, namely $\mathcal{O}(n^4)$ (see analysis in the proof of (5.3) of [34]), and if one also includes heavy dominos, the total runtime is bounded by $\mathcal{O}(n^4 \log n)$ (which also includes the additional reduction of the set of heavy dominos that are considered). Performing the odd cut and near min-cut computations using tree-decompositions would only save a factor of $\log n$ asymptotically in the total runtime since the reduction of the light and heavy dominos still takes time $\mathcal{O}(n^4)$, but the time spent in odd-cut computations would still be reduced from $\mathcal{O}(n^4 \log n)$ to $\mathcal{O}(n^2)$.

The assumption that $tw(\hat{G}) \leq c$ for some not too large constant c is realistic. If $\hat{G}$ is optimal, $\hat{G}$ will simply be a cycle (and hence have tree-width 2), and even if $\hat{x}$ is only a solution of good quality, one would expect that $\hat{G}$ is still similar to a cycle and hence its tree-width is not too high. In practice, this indeed seems to be the case, as indicated by the test results in 5.2.2.

However in general, $tw(\hat{G}) \le c$ does not necessarily imply that $tw(\mathcal{G}_{1/2})$ (or $tw(\mathcal{G}_{i,S})$) is bounded, as we show in the following example.

Example 5.2.1. Notice that if in $\mathcal{G}_{1/2}$ we contract all edges in each tree T_i, for $i = 1, \ldots n$ (where $n = |V(\hat{G})|$), and delete the vertex v^*, then the remaining graph is precisely $\hat{G}$ (or conversely, we can think of $\mathcal{G}_{1/2}$ as arising from $\hat{G}$ by first adding a new vertex v^* connected to all of $V(\hat{G})$, then replacing each $i \in V(\hat{G})$ by a tree T_i, and rerouting edges ij of $\hat{G}$ so that they are now connecting some vertex in T_i to some vertex in T_j).

Now suppose G is the $(2k+1) \times (2k+1)$ planar grid (for $k \ge 1$) on $4k^2 + 4k + 1$ vertices, with columns $1, \ldots, 2k+1$, and contract the k even numbered columns to obtain a graph G' on $k + (2k+1)(k+1) = 2k^2 + 4k + 1$ vertices (see Figure 14).

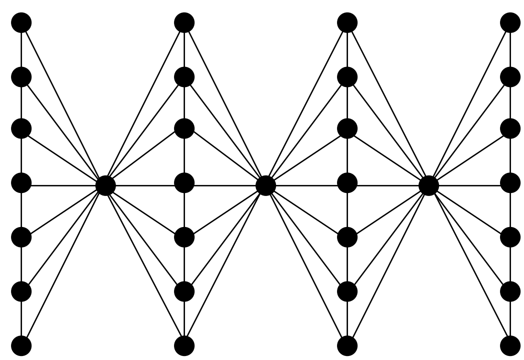


Figure 14: The graph G' (for $k = 3$) from Example 5.2.1

We have $tw(G) = 2k + 1$ (shown in [62]), and it is easy to see that $tw(G') = 3$: Clearly $tw(G') \ge 3$ since it has a K_4 minor, and $tw(G') \le 3$ since e.g. G' does not contain any of the 5 obstruction graphs for tree-width 3 as a minor (see e.g. [6]) (it is also easy to construct a tree-decomposition of width 3).

Hence if $\hat{G}$ is isomorphic to G', and $\mathcal{G}_{1/2}$ is isomorphic to G plus the vertex v^* connected to the $2k+1$ vertices in the first row of G (i.e. $\mathcal{G}_{1/2}$ is obtained from $\hat{G}$ by replacing each vertex by a path of length $2k+1$ as described above, and adding v^* with its neighbors), then $tw(\hat{G}) = 3$, $|V(\mathcal{G}_{1/2})| = \mathcal{O}(|V(\hat{G})|)$, $|E(\mathcal{G}_{1/2})| = \mathcal{O}(|E(\hat{G})|)$, and yet $tw(\mathcal{G}_{1/2}) = \Omega\left(\sqrt{|V(\hat{G})|}\right)$.

However in practice this complication does not seem to happen, as observed in the next section.

5.2.2 Computational results

We now present some computational results on the effectiveness of simple DP inequalities for TSP computations. For that purpose we implemented a reduced version of Algorithm 5.2.1 using tree-decompositions (available upon request from the author of this thesis), and integrated it into the TSP solver Concorde by Applegate, Bixby, Chvátal and Cook (Concorde is available at www.tsp.gatech.edu). More specifically, Concorde works in iterations where in each iteration or *round*, it starts with an LP solution $\hat{x}$ and uses various methods to find violated inequalities from a given class, and we added an option so that Concorde can use our implementation to find simple DP inequalities in each round.

In practice, a separation routine which finds only one violated inequality in $\hat{G}$ is essentially useless, since the pool of cutting planes maintained by Concorde typically contains hundreds or more violated inequalities, so adding a single inequality will typically not cause a significant change in the LP solution. On the other hand, trying to generate all violated simple DP inequalities is typically not practical either, since Concorde also limits the number of inequalities to be used for solving the next LP. Hence in general, the goal is to provide a cutting plane algorithm which produces a moderate number (no more than a few hundred) of violated inequalities from a given class. Once one has an algorithm to generate more than one violated inequality from a given class, typically a selection problem arises, i.e. one must decide on a strategy which inequalities to look for and add to the cut pool, since adding all is usually not an option. Some possible selection strategies were suggested for example in [2].

For our implementation, we focus only on simple light DP inequalities (for reasons explained below). We first use a tree-decomposition of $\hat{G}$ to generate all candidate light dominos and build $\mathcal{G}_{1/2}$, as described above. In our tests, $\mathcal{G}_{1/2}$ typically had more then $20,000$ vertices, up to more than $200,000$ in the largest examples. Since we can not generate all odd cuts of value < 1 in $\mathcal{G}_{1/2}$ (this would take too long and produce too many inequalities), we use the tree-decomposition of $\mathcal{G}_{1/2}$ to repeatedly find minimum odd cuts, as we describe now. Initially fix some $v_0 \in V(\mathcal{G}_{1/2})$ and find a minimum odd cut $\delta(X_1)$ in $\mathcal{G}_{1/2}$ with $v_0 \in X_1$. Then pick a vertex $v_1 \notin X_1$, and use the tree-decomposition of $\mathcal{G}_{1/2}$ to find an odd cut $\delta(X_2)$ of minimum value subject to the constraint that $\{v_0, v_1\} \subseteq X_2$. Then repeat this procedure as long as the returned cuts have value < 1 (clearly this will happen eventually, since no previously produced cut can be found again). It is very easy to modify the standard tree-decomposition method to find minimum odd cuts so that it only considers cuts where a subset of vertices is forced to be on a particular side. Obviously other methods for generating a selection of odd cuts using the tree-decomposition are possible, but besides being very simple to implement this method has the advantage of forcing the generated cuts to 'cover' different parts of the graph, so that the returned DP inequalities differ substantially.

In this way, we obtain a conceptually simple and fast routine for generating a moderate sample of violated simple light DP inequalities (for discussions on the performance, see below).

We did not include 'heavy' simple DP inequalities (i.e. DP inequalities where one domino has surplus $\geq 1/2$) in our implementation, for the following reasons (for our arguments, we assume that $m = \mathcal{O}(n)$, as is the case for graphs of bounded tree-width). In Algorithm 5.1.1, separating heavy inequalities is quite expensive. Certainly the runtime of $\mathcal{O}(n^5 \log n)$ for the basic version of the algorithm (using tree-decompositions) leaves little hope for a practical implementation. But even in the faster (and more complicated) version of the simple DP separation algorithm of

[34], computing heavy inequalities is still very expensive and inefficient: Instead of computing odd cuts only in $\mathcal{G}_{1/2}$, one would have to compute odd cuts in $\mathcal{O}(|E(\hat{G}|)$ graphs (which are modified versions of $\mathcal{G}_{1/2}$). In fact in their analysis the authors show that for every vertex $i \in V(\hat{G})$, one needs up to $|\delta(i)|$ such graphs to model heavy domino inequalities as odd cuts, yielding a total of $2m$ graphs for which we would have to find odd cuts in, as opposed to just 1 for detecting violated light inequalities. Moreover, the techniques for reducing the number of light and heavy domino inequalities in the improved version of the algorithm already take time $\mathcal{O}(n^4)$ by themselves. Apart from the computation difficulties, there is an even more important reason why implementing an exact separation routine for DP inequalities seems of little practical value. While there may be a lot of heavy *domino* inequalities (up to $\mathcal{O}(mn)$ in total, as shown in [34]), it is likely that there are not too many violated heavy *DP* inequalities in practice. A single heavy domino already contributes at least 1/2 to the surplus of the DP inequality, and so the other dominos (at least 2) have to have very small surplus in order to keep the surplus total below one. In particular, any violated simple DP inequality has at most one heavy domino, and no maximally violated (i.e. surplus total 0) simple DP inequality can be heavy. Finally, our implementation usually already returns a sufficient number of cuts (unless the tree-width is very large, see below), so additionally using a much slower method to look for a few extra cuts which have small violation does not seem worthwhile.

Like any tree-decomposition based algorithm, the performance of our implementation depends on the width of the tree-decomposition used. All our tree-decompositions were computed using a simple Min-Degree heuristic (discussed e.g. in [7]). In the following tests results, when we speak of the tree-width of a problem instance, we mean the width of the tree-decomposition computed by the Min-Degree heuristic, for a graph $\mathcal{G}_{1/2}$ in that instance.

As it turned out, the tree-width computed for $\mathcal{G}_{1/2}$ increased only marginally over that of its minor $\hat{G}$ (usually by at most one or two), in contrast to the theoretical worst case described in Example 5.2.1.

In our application, as long as the tree-width was below 20 then our simple DP implementation performed well and was usually able to generate a sufficient number of inequalities. If the tree-width was between 20 and 23, the algorithm typically becomes quite slow, and sometimes even runs out of memory altogether before a single inequality is found (we also set a maximum time bound of 180 seconds per round). Therefore if we detected that the tree-width of $\mathcal{G}_{1/2}$ was 24 or higher in a given round, we aborted the algorithm and did not try to compute an odd cut for that round.

All of our tests were run on 2.66 GHz Intel Xeon machines with 2 GB of memory. We initially chose two sets of instances for our tests: all TSP instances from TSPLIB [58] between 1,000 and 2,000 cities, and all TSPLIB instances between 3,000 and 10,000 cities. Each of tests described in the following was conducted for 10 different input seeds for Concorde, and all tests results reported here are averages of those runs.

For each problem, we first generated a 'good' LP solution by running the standard version of Concorde (with options -B -mC32) to the point where it would usually start branching. Then starting off that solution, we ran the following first set of tests:

'standard':	run Concorde again (for reference)
'simpleDP':	run Concorde with our simple DP implementation turned on
'planarDP':	run Concorde with the planar DP separator turned on

The last test uses a heuristic for finding (general) violated DP inequalities, which was described and shown to be highly effective computationally in [21]. This heuristic first 'approximates' $\hat{G}$ by a planar graph, and then generates violated (general) DP inequalities, adopting the exact method from [50].

The results of this first set of tests are shown in Table 1. Notice that a few problems from TSPLIB between 1,000 and 2,000 cities are not listed here, since the standard version of Concorde was already able to solve all or most instances of them to optimality. The table contains the following data: Columns 'gap' and 'gap %' list the absolute value and the percentage of the initial gap (in terms of the optimal solution), obtained from running Concorde once with -B and -mC32 (the gap of an LP solution $\hat{x}$ is defined as the value of the optimal solution, minus the value of $\hat{x}$). Columns 'std', 'sDP', 'pDP' list the improvement in the gap over the initial solution from the tests 'standard', 'simpleDP' and 'planarDP', respectively. More precisely, if Δ_0 is the gap of the initial solution, and Δ is the gap of the new solution, then the column lists the quantity $((\Delta_0 - \Delta)/\Delta_0) \cdot 100$. The remaining three columns contain statistics pertaining to the width of the tree-decompositions used. We say a round (as defined above) was *successful* if our implementation was able to generate more than one violated simple DP inequality in that round. The column 'success' displays the percentage of successful rounds for a given test instance. The column 'tw' contains the average tree-width over all rounds (successful or not) for that instance, and the column 'aborted' contains the percentage of runs which were aborted due to the tree-width being higher than 23 or the algorithm running out of memory.

The three columns on tree-width data give an indication to what extent our implementation was actually able to generate cutting planes. If 'success' was low, then this was due either to the fact that the tree-width was too high, or there were simply no violated inequalities to be detected (e.g. if the solution was getting very close to being optimal). Which one of the two cases appeared can be inferred from looking at the other two columns, 'aborted' and 'tw'. In particular, if 'success' was very low (below 10%) due to high tree-width, we did not perform any further tests on that instance since our implementation can not consistently contribute violated inequalities in such cases.

Table 1: Results of tests 'standard', 'simpleDP' and 'planarDP'

Problem	gap	gap %	std	sDP	pDP	success	tw	aborted
dsj1000	4822.77	0.0258%	3.7%	59.9%	89.4%	79.4%	10.4	0.0%
u1060	66.59	0.0297%	1.7%	25.2%	30.7%	76.1%	11.2	0.0%
vm1084	61.25	0.0256%	6.0%	77.6%	97.5%	77.4%	12.0	0.0%
pcb1173	9.09	0.0160%	5.2%	17.3%	37.6%	32.4%	9.4	0.0%
d1291	87.78	0.1728%	5.7%	17.7%	51.9%	84.8%	18.4	4.1%
rl1323	186.97	0.0692%	3.5%	22.2%	42.2%	64.6%	13.1	0.0%
fl1400	273.17	1.3572%	1.2%	19.9%	49.8%	100.0%	9.9	0.0%
fl1577	31.05	0.1396%	9.6%	22.1%	85.7%	95.3%	17.7	3.1%
vm1748	101.84	0.0303%	2.4%	62.6%	96.8%	92.9%	15.4	0.0%
u1817	130.03	0.2273%	2.3%	8.8%	75.9%	61.0%	21.3	22.9%
rl1889	263.63	0.0833%	5.5%	12.3%	18.3%	92.6%	15.1	0.0%
pcb3038	46.33	0.0336%	1.3%	7.1%	—	4.0%	25.4	86.2%
fl3795	85.24	0.2963%	8.1%	20.6%	77.2%	98.9%	13.2	0.0%
fnl4461	21.48	0.0118%	4.2%	10.7%	—	8.4%	24.9	82.4%
rl5915	227.88	0.0403%	9.0%	26.1%	—	5.0%	23.5	63.9%
rl5934	169.25	0.0304%	12.3%	15.5%	—	2.7%	24.2	74.9%
pla7397	8531.04	0.0367%	13.5%	35.7%	36.1 %	100.0%	15.2	0.0%

There are two main observations one can make looking at the Table 1. Our implementation was able to handle all instances of the first class (1,000-2,000 cities) without problems (i.e. most rounds were successful), but on all but two of the larger instances, the tree-width was too high in order for our implementation to work properly. We also ran some tests on some larger instances (namely rl11849 and usa13509), and also found that the tree-width was too large (roughly 30) to be handled by our implementation. There are several ways to get around this problem that may be worthwhile investigating: For example one could try to use other methods than the Min-Degree heuristic to try to obtain better tree-decompositions (although compared to other methods, the Min-Degree heuristic offers a good trade-off between runtime and quality of the decomposition), and one could also try to develop methods to approximate the input graph (in our case $\mathcal{G}_{1/2}$) by a graph of lower tree-width (e.g. by deleting or contracting specific edges), similar to what was done for approximating non-planar by planar graphs in [21].

Second, adding the simple DP separation routine brought significant improvements over just running Concorde a second time in essentially all cases where the 'success' rate was high (and even in some cases where only a small percentage of the rounds was successful). Nevertheless, it is just as clear from the data that the planar DP separation routine from [21] is even more effective in improving the LP bound.

We also performed a second round of tests, in order to see whether our exact simple DP separation routine finds some cuts that the heuristic planar DP separation routine misses. For that purpose, we compared the outcomes of the following two tests:

'planarDP again':	run Concorde with the planar DP separator turned on, starting from the solution of test 'planarDP'
'simpleDP+planarDP':	run Concorde with both the planar and simple DP separator turned on, starting from the solution of test 'planarDP'

The results of these tests are given in Table 2. We only ran this second set of tests on those examples where our implementation was not hindered by high tree-width, and we also did not run additional tests on vm1084 and vm1748, since the solutions resulting from the first 'planarDP' test were already very close to being optimal (in each case, at least half of the seeds were solved to optimality). Columns 'pDP again' and 'sDP + pDP' indicate the improvement of the gap by 'planarDP again' and 'simpleDP+planarDP' (respectively) over the gap from the test 'planar DP' of the first set. The remaining three columns are measuring the performance of the tree-width part, as above.

The results in 2 are more mixed than those in 1. The first thing to notice is that the percentage of successful rounds is quite low for a lot of instances, although the

Table 2: Results of tests 'planarDP', 'planarDP again' and 'simpleDP+planarDP'

Problem	pDP gap	pDP again	sDP + pDP	success	tw	aborted
dsj1000	509.48	2.2%	2.6%	12.8%	12.0	0.0%
u1060	46.16	0.2%	1.4%	26.2%	11.2	0.0%
pcb1173	5.66	0.0%	0.3%	12.6%	12.1	0.0%
d1291	42.25	3.4%	11.4%	11.5%	23.0	57.8%
rl1323	108.13	1.3%	1.4%	29.6%	13.5	0.0%
fl1400	137.18	26.3%	56.6%	100.0%	11.3	0.0%
fl1577	4.45	13.2%	20.1%	85.9%	15.1	0.0%
u1817	31.27	5.9%	6.2%	6.2%	24.2	70.9%
rl1889	215.54	3.7%	3.1%	50.0%	14.8	1.1%
fl3795	19.45	25.9%	21.2%	86.0%	13.8	0.0%
pla7397	5449.79	11.0%	16.7%	90.8%	15.7	0.0%

tree-width is not problematic (except d1291 and u1817). This is due to the fact that on those examples, it frequently happens that in a given round, no simple (light) DP inequality is violated. However looking at the individual log-files, it is usually the case that the planar DP separator does not find any violated general DP inequalities either. For those examples, the results of 'planarDP again' and 'simpleDP+planarDP' were essentially identical, and neither brought significant improvements.

On the other hand, for examples where the success rate was high (fl1400, fl1577, fl3795 and pla7397), both runs brought significant improvements (11% to 56%), and in 3 of those 4 examples adding the simple DP inequalities to the mix made a noticeable positive contribution over just using the planar DP separator.

However on fl3795, somewhat surprisingly, the 'simpleDP+planarDP' tests actually performed worse (on average over 10 runs) than the 'planarDP' tests. The same phenomenon occurs on rl1889 (where about 50% of the rounds were successful). In both cases, the actual difference was relatively small (the average gap in absolute values was worse by 0.9 and 1.1, respectively). We ran additional tests (100 additional seeds) for 'rl1889', and here the problem disappeared: While the average gap for 'planarDP again' was 210.06, the average gap for 'simpleDP+planarDP' was

207.20, which meant an improvement of 1.4% and 2.8% respectively over the average gap from ‘planarDP’ of 213.09. Hence while it can happen for single seeds that the ‘planarDP again’ version produces a slightly better gap because different inequalities are actually used in the cutpool, this seems to be the exception. In fact looking at the log files for the original 10 runs of rl1889, there was only one instance where ‘simpleDP+planarDP’ performed significantly worse, and the other 9 produced essentially no change of the gap in either direction. Similarly for fl3795, only 2 runs had a decrease in gap improvement, one had a significant improvement and the other 7 produced about the same gap for both tests.

In summary, our computational results imply the following conclusions:

1. Using a separation method for simple DP inequalities gives considerable improvements in the gap over running Concorde in standard mode.

2. If compared individually, simple DP inequalities do not seem to be as effective as the more general DP inequalities returned by the planar DP separation heuristic.

3. If the planar DP heuristic is unable to provide a decent amount of cuts, then the exact separation routine typically finds no violated cuts either, and conversely if the exact separation routine is unable to produce cuts despite the fact that the tree-width is low, then the planar DP heuristic usually finds no (or very few) violated general DP inequalities as well.

4. If there are enough violated simple DP inequalities left in $\hat{G}$, then enabling the simple DP separator in addition to the DP heuristic improves the gap noticeably (although not dramatic) in most cases, although there may be individual cases where the performance actually decreases slightly.

5. Simple tree-decomposition heuristics find tree-decompositions of width 10 to 25

(on average) for the graphs $\hat{G}$ tested here (in fact for the larger graphs $\mathcal{G}_{1/2}$), making them accessible for tree-decomposition based methods.

Our implementation of a separation routine for simple (light) DP inequalities is relatively basic, and leaves room for improvement. As discussed earlier, we only focus on generating light simple DP inequalities, so one obvious improvement would be to try to get a fast practical method for generating heavy simple DP inequalities as well. However as we noted earlier, even apart from the practical challenges of obtaining a fast method for this, it appears unlikely that this would lead to significant progress in improving the gap, since for one heavy simple DP inequalities can never be violated by much, and moreover (as mentioned in (5)), once there are no more violated light simple DP inequalities, even the more general planar DP heuristic has trouble providing cuts, suggesting that there are in particular essentially no violated heavy simple DP inequalities left either in such cases.

Second, we chose a very simple approach to generating a selection of violated inequalities each round, namely to simply keep forcing vertices to one side of the cut, and finding single odd cuts by repeatedly traversing the tree-decomposition. Clearly one can try other methods of getting a good selection of cuts from the tree-decomposition of $\mathcal{G}_{1/2}$, e.g. by building all partial solutions stored in the tree up to a certain depth, and then completing them to distinct odd cuts in $\mathcal{G}_{1/2}$. In particular one could try to generate more inequalities per round in this way- while our implementation typically returns about 50-150 violated inequalities per round (when it was not hindered by high tree-width or too few violated inequalities left in $\hat{x}$) before fixing vertices leads to increasing the value of the best odd cut to 1, the heuristic for planar general DP inequalities usually passes 500 inequalities per round to Concorde, about half of which typically make it to the cut-pool.

Third, in terms of the tree-decomposition part, one can try to obtain better tree-decompositions by using more sophisticated methods, or try to actively decrease the tree-width of $\hat{G}$ (and $\mathcal{G}_{1/2}$) without destroying too many violated inequalities, in order to be able to deal with larger problems.

5.3 Conclusion

The idea of using tree-based decompositions in the context of the TSP first originated in the paper by Cook and Seymour [22], where the authors use branch-decompositions to find an optimal tour in a graph which is the union of a small number of 'good' tours. In this chapter we have demonstrated two possible ways of exploiting low tree-width in the context of the TSP. As we have seen, the support graph of (reasonably good) LP solutions $\hat{x}$ tends to have relatively low tree-width. This is to be expected, as the support graph of a good LP solution should look similar to a single cycle, which has tree-width two. What is equally important is that even the simplest heuristics actually find tree-decompositions for these graphs whose width is in a range where optimization methods can be applied in practice. Of course the idea of using tree-based decomposition methods on the support graph can also be applied to other classes of inequalities.

The results of this chapter also give further evidence (after e.g. [22], [48], [70], [47], [46], [49], [43], [19]) that algorithms working on tree-decompositions (or branch-decompositions) of bounded width are not only of theoretical interest, but may in practice sometimes represent an alternative to traditional methods such as integer programming or complex polynomial-time algorithms. Some of the potential advantages in particular compared to complex polynomial-time algorithms are listed below:

1. Many polynomial-time methods only generate a *single* optimal solution (in the case of [34] one violated DP inequality), and it is usually not easy to generate *all* (or a even a large number of) optimal solutions for an instance. In the dynamic programming approach typically used for tree-decomposition based methods, constructing all (near-) optimal solutions instead of a single one requires only

small changes to the algorithm, in contrast to standard polynomial-time algorithms. Even if generating all solutions is not an option, the tree-decomposition is a good starting point for getting a selection of optimal or near-optimal solutions. In the case of finding violated inequalities for the TSP, this is crucial: Finding only a single violated inequality will usually not help to increase the LP bound on larger problems.

2. Even standard polynomial-time techniques such as max-flow based techniques may become infeasible to use on very large graphs, while the only critical limitation for tree-decomposition based algorithms is usually the width of the decomposition used. For example in 5.2 we compute all cuts within a factor of 1.5 of the minimum cut on graphs with more than 5000 edges, and compute a minimum odd cut in graphs of up to 200,000 vertices within a few seconds, which is at the very least a challenge for implementations of the corresponding standard methods ([54], [57]).

3. Once an implementation for the general dynamic programming framework of the tree-decomposition based methods is available, it is relatively easy to adapt it to solve different problems such as minimum (or maximum) (odd) cut, max clique, chromatic number and many others, or introduce additional side constraints (like we did when searching for minimum odd cuts with some vertices fixed).

4. Although the constant factors in the theoretical analysis of many problems solvable in linear time on graphs of bounded tree-width may seem large, in practice these methods often still work well if the width of the decomposition used is reasonably high, as illustrated e.g. in Section 5.2.2. Further evidence of the practicability of tree-decomposition based methods is given Chapter A of the appendix of this thesis, where we report computational results for such algorithms on a large collection of graphs from various practical applications.

Apart from problem specific techniques, using better methods to find good tree-decompositions and other generally applicable methods to reduce memory requirements (see e.g. [10]) would likely increase the number of instances that can be handled.

APPENDIX A

COMPUTATIONAL RESULTS FOR SOLVING STANDARD PROBLEMS ON GRAPHS OF BOUNDED TREE-WIDTH

In 5.2.1 and 5.2.2, we demonstrated that tree-decompositions can actually be useful in practice, even if the width of the decomposition used is moderately high. In fact the tree-decomposition part of the algorithm in 5.2.1 was implemented as a module of a more general framework. In this appendix we briefly describe this framework, and give some computational results for solving different NP-hard problems on a large variety of problem instances from practical applications.

In order to test the practicability of tree-decomposition based methods for several NP-hard problems, we implemented a version of the standard dynamic programming approach for tree-decomposition based algorithms. The implementation was done in the Java programming language (version 1.5), and is available upon request from the author of this thesis. The key component of the package is a template class which contains the basic mechanisms to store information in a (nice) tree-decomposition tree and traverse it bottom-up in order to compute information at the root node. The package also contains classes representing the abstract notions of partial solutions characteristics and full solutions, as described in [14] or in section 5.1. In order to add an algorithm for a specific problem, such as computing the chromatic number, all that has to be done is to create a subclass of the above template classes (using Java's inheritance framework) and override methods which are problem-specific , such as generating all characteristics at a LEAF node, or extending characteristics at an INTRO node. The present version of the package contains implementations for

- finding a minimum (odd) cut,
- finding a maximum (odd) cut,
- finding all (odd) cuts below a certain value,
- computing an optimal coloring, and computing the chromatic number,
- computing a maximum independent set,
- computing a maximum clique.

The above classes are straightforward implementations of the standard dynamic programming approach, and contain no sophisticated data structures or algorithms. The only exception is the class for computing an optimal coloring of a graph. Apart from first running a first-fit heuristic to get an upper bound on the chromatic number (which can be used to prune solutions during the tree-decomposition part), we use a non-trivial scheme for encoding characteristics of partial solutions in order to save memory. For coloring, the characteristic of a partial solution at a node t mainly consists of a partition of the bag X_t into (independent) sets. If $X_t = \{0, \ldots, k-1\}$, then a standard way to encode a partition of X_t is a vector $a = (a_0, a_1, \ldots, a_{k-1})$ of length k, satisfying the following property:

$$a_0 = 0, \quad a_i \leq 1 + \max(a_0, a_1, \ldots, a_{i-1}) \quad \forall\, 0 \leq i \leq k-1. \tag{A.0.1}$$

It is easy to see that such vectors are in 1-1 correspondence with partitions of $\{0, 1, \ldots, k-1\}$. For example the partition $\{\{0,1,4\},\{2,3\}\}$ of the set $\{0,1,2,3,4\}$ could be encoded as $a = (0,0,1,1,0)$. In our implementation, we exploit the fact that $a_i \leq i$ to store a partition of a set of up to 19 elements in a single 64 bit variable. This yields significant memory savings over the straightforward approach of representing each a_i by a separate integer, but limits this approach to sets X_t of size at most 19 (of course one could use e.g. a second 64 bit variable to deal with larger sets). Therefore the coloring algorithm can only handle tree-decompositions of width at most 18, but our test examples in fact indicate that this is already on the border of what is doable in practice with the standard dynamic programming approach.

In order to estimate up to what tree-width the standard dynamic programming approach works in practice, we created a test collection of 95 instances of graphs of low tree-width, from a wide spectrum of practical applications. Most of these graphs can be found in the online library 'TreewidthLIB' created by Bodlaender and van den Broek [12], which contains a large collection of graphs of low tree-width. Our particular focus was on graphs where the minimum degree heuristic (see e.g. [7]) finds a tree-decomposition of width at most 20, since it seems that in practice, one would need rather specific conditions to be met in order to handle instances of width much higher than that. Our test collection consists of the following instances (for more specific information about the practical applications that the specific instances arise in, we refer the reader to the references given below).

- All 16 graphs from frequency assignment problems discussed in [48]: celar06, celar06pp, celar07, celar07pp, celar08, celar08pp, celar09, celar09pp, graph05, graph06, graph06pp, graph11, graph12, graph12pp, graph13, graph13pp. All of these graphs are contained in the online library TreewidthLIB [12].

- Several groups of graphs used for testing in [48] which arise from coloring problems: The Mycielsky group (myciel3, myciel4, myciel5, myciel6, myciel7), the 'miles' group (miles250, miles500, miles500, miles1000, miles1500) and the 'queen' group (queen5_5, queen6_6, queen7_7, queen8_8, queen9_9, queen10_10, queen11_11). All of these 17 graphs are contained in the online library TreewidthLIB [12].

- Graphs from a website maintained by N.J.A. Sloane of some difficult instances for computing a maximum independent set [68]. Our test collection consists of the 7 graphs of the 'dc' family with at most 512 vertices, namely 1dc.64, 1dc.128, 1dc256, 1dc512, 2dc.128 and 2dc.512.

- 4 large instances arising during the execution of the algorithm described in 5.2.1: G_S_1, G_S_2, G_S_3, G_S_4. The instances are available upon request from the author of this thesis.

- Graphs stored in TreewidthLIB [12] with the following property: The minimum degree heuristic (as implemented by the authors of TreewidthLIB) computes a tree-decomposition of width at least 10 and at most 20. For our test compilation we have included all of these graphs, with the exception that if TreewidthLIB contains different (pre-processed) versions of the same graph, we only chose the original one. All graphs not mentioned in one of the above four categories belong to this group, and some graphs in the first category also satisfy this property. Among others, this group contains several graphs arising from Delauney triangulations of TSP instances.

For each of the above 95 instances, we used the above-described modules to compute the maximum clique number $\omega(G)$, the size of a maximum independent set $\alpha(G)$, the chromatic number $\chi(G)$ and the maximum cut in the given graph G. All tests were carried out on AMD Athlon workstations with a dual-core 1.8 GHz processor and 2GB of memory. A detailed list of all results is given in Table 4 at the end of this chapter. The runtimes given there do not contain the time for executing the minimum degree heuristic to find a tree-decomposition (which was typically less than a second). An entry of '-' indicates that the algorithm failed, which was either due to insufficient memory, or in the case of the coloring the width of the decomposition used being 19 or higher. The results are ordered first by increasing tree-width, and then by increasing number of vertices (hence roughly increasing in the difficulty of the instance for a tree-decomposition based approach).

Table 3 contains a short summary of the results of Table 4: For each problem we list the minimum width for which at least one instance failed ('min tw unsolved'), the maximum width for which at least one instance completed ('max tw solved'), and the number of instances that were solved successfully in between these two bounds ('solved in between').

Table 3: Overall performance of tree-decomposition based methods

Problem	min tw unsolved	max tw solved	solved in between
$\omega(G)$	43	313	12 / 16
$\alpha(G)$	57	188	3 / 12
$\chi(G)$	14	18	5 / 22
max cut in G	19	21	4 / 4

We next list a few observations one can make from Tables 3 and 4. Despite the fact that our test collection of is rather large and diverse, there are some restrictions to keep in mind, which we list as well.

- Each of the 4 problems (max clique, max independent set, chromatic number, max cut) are typically solvable for much higher tree-width in practice than what one can expect from the theoretical worst case analysis.

- The problems 'chromatic number' and 'max cut' exhibit a relatively tight threshold behavior in practice with respect to the width of the decomposition, in the following sense. Most problems below a certain width can be solved very efficiently, and essentially no problems above the threshold can be solved. Only if the width is in a small range around that threshold does the solvability depend on the possible additional structure or the size of the actual problem instance.

- It appears that computing $\omega(G)$ and $\alpha(G)$ is easier in practice than solving the other two problems. Part of this can be explained by the fact that the size of the optimal set (clique or independent set) in the successfully solved instances of our test collection with very high width (> 30) is usually much smaller than the width of the decomposition. In that case, the dynamic programming approach is exponential only in the size of the optimal set. Also, there are only relatively few instances in our test collection of width higher than 20, so the data in the summary Table 3 should be interpreted in the sense that one can typically expect to compute $\omega(G)$ and $\alpha(G)$ without problems if the tree-width is at most 20, and sometimes also for much higher values.

In fact the special structure of the maximum clique problem makes it in principle possible to use tree-decompositions in an even simpler way: Since any maximum clique is contained in one of the bags of a tree-decomposition, one could use the best performing exact method to compute a maximum clique in each bag, so that $\omega(G)$ is simply the maximum of those clique sizes. This could potentially improve the scope of exact methods for the maximum clique problem for those cases where the tree-width is much smaller than the size (i.e. number of vertices) of the input graph.

The focus of our implementation of the dynamic programming framework for tree-decomposition based methods was to show that already a straightforward implementation of this algorithmic template can be useful for solving non-trivial instances of hard problems. For example, for the graph G_S_2 with more than 150,000 vertices (and a tree-decomposition of width 14), it takes 22 seconds to compute $\alpha(G)$, under 7 seconds to compute $\omega(G)$ and 1 minute to compute a maximum cut. Apart from the above mentioned methods for computing $\chi(G)$, little effort has been made to optimize memory consumption or running time of our implementation. Hence the running times reported here can certainly be improved significantly, e.g. by making use of further pruning techniques at each node of the tree-decomposition, or by using more efficient data structures. Given the fact that the computational results in Table 4 are already promising, it would be of interest to develop more optimized versions of the concepts presented here, and investigate how they compare to other exact methods for solving NP-hard problems in practice.

Table 4: Computational results for solving standard problems

Problem	n	m	tw	ω	sec	α	sec	χ	sec	max cut	sec
myciel3	11	20	5	2	<0.1	5	<0.1	4	<0.1	16	<0.1
graph12pp	61	122	5	5	<0.1	17	<0.1	5	0.2	86	<0.1
celar09pp	67	165	7	8	0.2	18	<0.1	8	<0.1	112	0.1
miles250	128	387	9	8	0.2	44	<0.1	8	<0.1	263	0.4
eil51.tsp	51	140	10	3	<0.1	16	0.2	4	3.1	95	0.1
huck	74	602	10	11	0.5	27	<0.1	11	<0.1	191	0.4
celar02	100	311	10	8	0.1	34	<0.1	8	0.1	213	0.3
att532	532	729	10	4	<0.1	239	0.2	4	8.3	37149	0.2
myciel4	23	71	11	2	<0.1	11	<0.1	5	6.4	55	0.5
water	32	123	11	6	0.1	12	0.1	6	0.6	84	0.3
BN_16-pp-014	34	156	11	7	0.1	12	<0.1	7	0.6	124	0.8
oesoca+	67	208	11	10	0.2	35	<0.1	10	0.5	141	0.7
celar06pp	82	327	11	11	0.2	21	<0.1	11	<0.1	214	0.9
rat99.tsp	99	279	11	4	0.1	32	0.1	4	12.9	188	0.3
celar06	100	350	11	11	0.4	31	<0.1	11	<0.1	233	0.9
munin1	189	466	11	4	0.1	87	0.6	4	9.0	282	0.3
1bx7	41	195	12	7	0.1	10	<0.1	7	1.2	127	0.7
kroC100.tsp	100	286	12	4	0.1	32	0.1	4	88.8	192	0.6
lin105.tsp	105	292	12	4	<0.1	34	0.1	4	108.7	198	0.5
pr124.tsp	124	318	12	3	<0.1	47	0.3	4	59.1	220	0.4
anna	138	423	12	7	0.4	85	0.1	7	1.2	307	2.1
pr152.tsp	152	428	12	4	0.1	51	0.3	4	26.7	289	0.5
fl417.tsp	417	1179	12	4	0.1	132	0.1	4	409.2	797	1.0
pigs	441	806	12	3	0.1	225	0.4	3	107.3	591	1.4
1ubq	74	211	13	5	0.1	30	<0.1	5	7.7	153	1.9
pr76.tsp	76	218	13	4	0.1	24	0.1	4	177.9	146	0.9
david	87	406	13	11	0.4	36	<0.1	11	0.4	267	3.3
G_S_3	39559	45109	13	4	2.6	19630	5.6	4	648.3	43306	9.7
kroE100.tsp	100	283	14	4	0.1	32	0.1	4	1081.8	191	0.7
kroB150.tsp	150	436	14	4	0.1	48	0.2	-	-	292	5.0
fl3795	2103	3973	14	4	0.2	950	1.6	-	-	3394	7.5
G_S_4	62791	69330	14	4	3.1	31203	11.7	-	-	67180	23.9
G_S_2	153410	160239	14	5	6.6	76517	22.0	-	-	157985	60.6
eil76.tsp	76	215	15	3	0.2	24	0.6	-	-	145	3.2
rd100.tsp	100	286	15	4	0.1	31	0.4	-	-	192	4.0
pr136.tsp	136	377	15	3	0.1	44	0.2	-	-	256	3.2
kroA150.tsp	150	432	15	4	0.1	48	0.7	-	-	290	9.3

Table 4: continued

Problem	n	m	tw	ω	sec	α	sec	χ	sec	max cut	sec
celar05	200	681	15	8	0.3	66	0.2	8	187.2	458	6.3
G_S_1	71351	78536	15	4	1.5	35476	19.1	-	-	76192	39.2
a280.tsp-pp-002	92	257	16	3	0.1	29	0.5	-	-	174	4.9
kroB100.tsp	100	284	16	3	0.1	31	0.7	-	-	191	7.8
kroA100.tsp	100	285	16	4	<0.1	32	0.1	-	-	192	1.9
eil101.tsp	101	290	16	3	<0.1	32	0.2	-	-	195	7.9
u159.tsp	159	431	16	4	0.1	55	0.8	-	-	294	6.2
kroB200.tps	200	580	16	4	0.1	63	0.6	-	-	389	39.4
celar03	200	721	16	9	0.2	64	0.2	9	266.6	482	10.0
celar11	340	975	16	9	0.2	125	0.4	9	541.2	682	8.2
kroA200.tsp	200	586	17	4	0.1	62	1.7	-	-	392	37.6
pr299.tsp	299	864	17	4	0.2	97	1.7	-	-	581	50.3
celar01	458	1449	17	8	0.3	159	0.6	-	-	988	166.2
queen5_5	25	160	18	5	0.1	5	0.2	5	35.2	100	12.7
celar07pp	162	764	18	11	0.1	42	0.1	11	146.6	497	25.2
celar07	200	817	18	11	0.3	60	<0.1	11	142.0	538	25.4
tsp225.tsp	225	622	18	3	0.1	73	1.2	-	-	423	168.7
celar04	340	1009	18	10	0.3	117	1.4	-	-	696	53.3
celar09	340	1130	18	11	0.2	112	1.7	-	-	764	38.6
celar10	340	1130	18	11	0.1	112	0.1	-	-	764	27.6
celar08pp	365	1539	18	11	0.8	98	1.3	-	-	1013	164.6
celar08	458	1655	18	11	0.9	148	1.1	-	-	1106	292.9
p654.tsp	654	1806	18	4	0.3	219	1.0	-	-	1229	29.6
1i07	59	397	19	8	0.1	12	0.1	-	-	245	24.6
1b67	68	559	19	9	0.3	12	0.1	-	-	340	69.4
rat195.tsp	195	562	19	4	0.1	61	1.2	-	-	378	81.9
link	724	1738	19	4	0.3	301	270.2	-	-	-	-
d2103	2103	2737	19	4	0.4	1010	39.0	-	-	127901	151.4
myciel5	47	236	20	2	0.1	23	0.4	-	-	180	330.4
knights8_8	64	168	20	2	<0.1	32	6.1	-	-	168	536.3
ch150.tsp	150	432	20	4	0.2	48	2.4	-	-	290	251.4
d198.tsp	198	571	20	4	0.1	63	4.8	-	-	384	238.8
pr264.tsp	264	772	21	3	0.2	84	4.7	-	-	517	538.7
graph06pp	119	348	22	6	<0.1	43	25.0	-	-	-	-
miles500	128	1170	27	20	140.1	18	0.5	-	-	-	-
queen6_6	36	290	28	6	0.3	6	0.6	-	-	-	-
graph05	100	416	28	9	0.2	25	43.6	-	-	-	-
1dc.64	64	543	32	7	0.8	10	0.5	-	-	-	-
myciel6	95	755	35	2	1.2	47	310.3	-	-	-	-
queen7_7	49	476	38	7	1.2	7	1.3	-	-	-	-
miles750	128	2113	43	-	-	12	0.4	-	-	-	-

Table 4: continued

Problem	n	m	tw	ω	sec	α	sec	χ	sec	max cut	sec
queen8_8	64	728	49	8	0.9	8	27.5	-	-	-	-
miles1000	128	3216	54	-	-	8	0.7	-	-	-	-
graph06	200	843	57	9	1.3	-	-	-	-	-	-
queen9_9	81	1056	66	9	6.4	9	34.1	-	-	-	-
1dc.128	128	1471	67	8	2.7	-	-	-	-	-	-
myciel7	191	2360	78	2	2.4	-	-	-	-	-	-
queen10_10	100	1470	80	10	5.5	-	-	-	-	-	-
miles1500	128	5198	83	-	-	5	1.1	-	-	-	-
2dc.128	128	5173	89	-	-	5	1.6	-	-	-	-
graph12	340	1256	98	6	1.0	-	-	-	-	-	-
queen11_11	121	1980	101	11	29.0	-	-	-	-	-	-
graph11	340	1425	103	8	0.2	-	-	-	-	-	-
1dc.256	256	3839	138	9	83.0	-	-	-	-	-	-
graph13pp	456	1874	138	7	0.3	-	-	-	-	-	-
graph13	458	1877	138	7	0.3	-	-	-	-	-	-
2dc.256	256	17183	188	-	-	7	273.2	-	-	-	-
1dc.512	512	9727	313	10	2240	-	-	-	-	-	-
2dc.512	512	54895	416	-	-	-	-	-	-	-	-

REFERENCES

[1] Alon, N., Seymour, P., and Thomas, R., "A separator theorem for nonplanar graphs," *J. Amer. Math. Soc.*, vol. 3, no. 4, pp. 801–808, 1990.

[2] Andreello, G., Caprara, A., and Fischetti, M., "Embedding $\{0, \frac{1}{2}\}$-cuts in a branch-and-cut framework: a computational study," *INFORMS J. Comput.*, vol. 19, no. 2, pp. 229–238, 2007.

[3] Applegate, D. L., Bixby, R. E., Chvátal, V., and Cook, W. J., *The traveling salesman problem*. Princeton Series in Applied Mathematics, Princeton, NJ: Princeton University Press, 2006. A computational study.

[4] Arnborg, S., Lagergren, J., and Seese, D., "Easy problems for tree-decomposable graphs," *J. Algorithms*, vol. 12, no. 2, pp. 308–340, 1991.

[5] Arnborg, S. and Proskurowski, A., "Linear time algorithms for NP-hard problems restricted to partial k-trees," *Discrete Appl. Math.*, vol. 23, no. 1, pp. 11–24, 1989.

[6] Arnborg, S., Proskurowski, A., and Corneil, D. G., "Forbidden minors characterization of partial 3-trees," *Discrete Math.*, vol. 80, no. 1, pp. 1–19, 1990.

[7] Bachoore, E. H. and Bodlaender, H. L., "New upper bound heuristics for treewidth," Tech. Rep. UU-CS-2004-036, Department of Information and Computing Sciences, Utrecht University, 2004.

[8] Barnette, D. W., "Generating projective plane polyhedral maps," *J. Combin. Theory Ser. B*, vol. 51, no. 2, pp. 277–291, 1991.

[9] Bern, M. W., Lawler, E. L., and Wong, A. L., "Linear-time computation of optimal subgraphs of decomposable graphs," *J. Algorithms*, vol. 8, no. 2, pp. 216–235, 1987.

[10] Betzler, N., Niedermeier, R., and Uhlmann, J., "Tree decompositions of graphs: saving memory in dynamic programming," *Discrete Optim.*, vol. 3, no. 3, pp. 220–229, 2006.

[11] Bodlaender, H. L., "A tourist guide through treewidth," *Acta Cybernet.*, vol. 11, no. 1-2, pp. 1–21, 1993.

[12] Bodlaender, H. and van den Broek, J.-W., "TreewidthLIB," *http://people.cs.uu.nl/hansb/treewidthlib/index.php*.

[13] Bodlaender, H. L., "A linear time algorithm for finding tree-decompositions of small treewidth," *SIAM Journal on Computing*, vol. 25, pp. 1305–1317, 1996.

[14] Bodlaender, H. L., “Treewidth: algorithmic techniques and results,” in *Mathematical foundations of computer science 1997 (Bratislava)*, vol. 1295 of *Lecture Notes in Comput. Sci.*, pp. 19–36, Berlin: Springer, 1997.

[15] Bodlaender, H. L. and Thilikos, D. M., “Graphs with branchwidth at most three,” *J. Algorithms*, vol. 32, no. 2, pp. 167–194, 1999.

[16] Borie, R. B., Parker, R. G., and Tovey, C. A., “Automatic generation of linear-time algorithms from predicate calculus descriptions of problems on recursively constructed graph families,” *Algorithmica*, vol. 7, no. 5-6, pp. 555–581, 1992.

[17] Caprara, A. and Fischetti, M., “$\{0, \frac{1}{2}\}$-Chvátal-Gomory cuts,” *Math. Programming*, vol. 74, no. 3, Ser. A, pp. 221–235, 1996.

[18] Carr, R., “Separating clique trees and bipartition inequalities having a fixed number of handles and teeth in polynomial time,” *Math. Oper. Res.*, vol. 22, no. 2, pp. 257–265, 1997.

[19] Christian Jr., W. A., *Linear-time algorithms for graphs of bounded branchwidth*. PhD thesis, Rice University, Texas, United States, 2003.

[20] Chvátal, V., “Edmonds polytopes and weakly Hamiltonian graphs,” *Math. Programming*, vol. 5, pp. 29–40, 1973.

[21] Cook, W., Espinoza, D., and Goycoolea, M., “A study of domino-parity and k-parity constraints for the TSP,” in *Integer programming and combinatorial optimization*, vol. 3509 of *Lecture Notes in Comput. Sci.*, pp. 452–467, Berlin: Springer, 2005.

[22] Cook, W. and Seymour, P., “Tour merging via branch-decomposition,” *INFORMS J. Comput.*, vol. 15, no. 3, pp. 233–248, 2003.

[23] Courcelle, B., “The monadic second-order logic of graphs. III. Tree-decompositions, minors and complexity issues,” *RAIRO Inform. Théor. Appl.*, vol. 26, no. 3, pp. 257–286, 1992.

[24] Courcelle, B. and Mosbah, M., “Monadic second-order evaluations on tree-decomposable graphs,” in *Graph-theoretic concepts in computer science (Fischbachau, 1991)*, vol. 570 of *Lecture Notes in Comput. Sci.*, pp. 13–24, Berlin: Springer, 1992.

[25] Courcelle, B., “Graph rewriting: an algebraic and logic approach,” in *Handbook of theoretical computer science, Vol. B*, pp. 193–242, Amsterdam: Elsevier, 1990.

[26] Courcelle, B., “The monadic second-order logic of graphs. I. Recognizable sets of finite graphs,” *Inform. and Comput.*, vol. 85, no. 1, pp. 12–75, 1990.

[27] Dantzig, G., Fulkerson, R., and Johnson, S., "Solution of a large-scale traveling-salesman problem," *J. Operations Res. Soc. Amer.*, vol. 2, pp. 393–410, 1954.

[28] Demaine, E. D., Fomin, F. V., Hajiaghayi, M., and Thilikos, D. M., "Subexponential parameterized algorithms on bounded-genus graphs and H-minor-free graphs," *J. ACM*, vol. 52, no. 6, pp. 866–893 (electronic), 2005.

[29] Dharmatilake, J., *Binary Matroids With Branch-Width Three.* PhD thesis, Ohio State University, 1994.

[30] Diestel, R., *Graph theory*, vol. 173 of *Graduate Texts in Mathematics.* Berlin: Springer-Verlag, third ed., 2005.

[31] Edmonds, J., "Maximum matching and a polyhedron with $0,1$-vertices," *J. Res. Nat. Bur. Standards Sect. B*, vol. 69B, pp. 125–130, 1965.

[32] Epstein, D. B. A., "Curves on 2-manifolds and isotopies," *Acta Math.*, vol. 115, pp. 83–107, 1966.

[33] Fleischer, L. and Tardos, É., "Separating maximally violated comb inequalities in planar graphs," *Math. Oper. Res.*, vol. 24, no. 1, pp. 130–148, 1999.

[34] Fleischer, L. K., Letchford, A. N., and Lodi, A., "Polynomial-time separation of a superclass of simple comb inequalities," *Math. Oper. Res.*, vol. 31, no. 4, pp. 696–713, 2006.

[35] Grötschel, M., Lovász, L., and Schrijver, A., *Geometric algorithms and combinatorial optimization*, vol. 2 of *Algorithms and Combinatorics: Study and Research Texts.* Berlin: Springer-Verlag, 1988.

[36] Grötschel, M. and Padberg, M. W., "On the symmetric travelling salesman problem. I. Inequalities," *Math. Programming*, vol. 16, no. 3, pp. 265–280, 1979.

[37] Grötschel, M. and Padberg, M. W., "On the symmetric travelling salesman problem. II. Lifting theorems and facets," *Math. Programming*, vol. 16, no. 3, pp. 281–302, 1979.

[38] Gu, Q.-P. and Tamaki, H., "Optimal branch-decomposition of planar graphs in $O(n^3)$ time," in *Automata, languages and programming*, vol. 3580 of *Lecture Notes in Comput. Sci.*, pp. 373–384, Berlin: Springer, 2005.

[39] Henzinger, M. and Williamson, D. P., "On the number of small cuts in a graph," *Inform. Process. Lett.*, vol. 59, no. 1, pp. 41–44, 1996.

[40] Hicks, I. V., "Planar branch decompositions. I. The ratcatcher," *INFORMS J. Comput.*, vol. 17, no. 4, pp. 402–412, 2005.

[41] Hicks, I. V., "Planar branch decompositions. II. The cycle method," *INFORMS J. Comput.*, vol. 17, no. 4, pp. 413–421, 2005.

[42] Hong, S., *A Linear Programming Approach for the Traveling Salesman Problem.* PhD thesis, 1972.

[43] Jensen, F. V., Lauritzen, S. L., and Olesen, K. G., "Bayesian updating in causal probabilistic networks by local computations," *CSQ—Comput. Statist. Quart.*, vol. 5, no. 4, pp. 269–282, 1990.

[44] Johnson, E. and Robertson, N. Private communication.

[45] Karp, R. M., "Reducibility among combinatorial problems," in *Complexity of computer computations (Proc. Sympos., IBM Thomas J. Watson Res. Center, Yorktown Heights, N.Y., 1972)*, pp. 85–103, New York: Plenum, 1972.

[46] Koster, A. M. C. A., van Hoesel, C. P. M., and Kolen, A. W. J., "Optimal solutions for a frequency assignment problem via tree-decomposition," in *Graph-Theoretic Concepts in Computer Science (WG '99)* (Widmayer, P., Neyer, G., and Eidenbenz, S., eds.), vol. 1665 of *Lecture Notes in Computer Science*, pp. 338–349, Springer-Verlag, 1999.

[47] Koster, A. M. C. A., van Hoesel, C. P. M., and Kolen, A. W. J., "Solving frequency assignment problems via tree-decomposition," *Electronic Notes in Discrete Mathematics*, vol. 3, 1999.

[48] Koster, A. M. C. A., Bodlaender, H. L., and van Hoesel, S. P. M., "Treewidth: computational experiments," in *1st Cologne-Twente Workshop on Graphs and Combinatorial Optimization*, vol. 8 of *Electron. Notes Discrete Math.*, p. 4 pp. (electronic), Amsterdam: Elsevier, 2001.

[49] Lauritzen, S. L. and Spiegelhalter, D. J., "Local computations with probabilities on graphical structures and their application to expert systems," *J. Roy. Statist. Soc. Ser. B*, vol. 50, no. 2, pp. 157–224, 1988. With discussion.

[50] Letchford, A. N., "Separating a superclass of comb inequalities in planar graphs," *Math. Oper. Res.*, vol. 25, no. 3, pp. 443–454, 2000.

[51] Lins, S., "A minimax theorem on circuits in projective graphs," *J. Combin. Theory Ser. B*, vol. 30, no. 3, pp. 253–262, 1981.

[52] Malnič, A. and Mohar, B., "Generating locally cyclic triangulations of surfaces," *J. Comb. Theory Ser. B*, vol. 56, no. 2, pp. 147–164, 1992.

[53] Mohar, B. and Thomassen, C., *Graphs on surfaces.* Johns Hopkins Studies in the Mathematical Sciences, Baltimore, MD: Johns Hopkins University Press, 2001.

[54] Nagamochi, H., Nishimura, K., and Ibaraki, T., "Computing all small cuts in undirected networks," in *Algorithms and computation (Beijing, 1994)*, vol. 834 of *Lecture Notes in Comput. Sci.*, pp. 190–198, Berlin: Springer, 1994.

[55] NEGAMI, S., "Re-embedding of projective-planar graphs," *J. Combin. Theory Ser. B*, vol. 44, no. 3, pp. 276–299, 1988.

[56] PADBERG, M. and RINALDI, G., "An efficient algorithm for the minimum capacity cut problem," *Math. Programming*, vol. 47, no. 1, (Ser. A), pp. 19–36, 1990.

[57] PADBERG, M. W. and RAO, M. R., "Odd minimum cut-sets and b-matchings," *Math. Oper. Res.*, vol. 7, no. 1, pp. 67–80, 1982.

[58] REINELT, G., "Tsplib - a traveling salesman library," *ORSA Journal of Computing*, vol. 3, pp. 376–384, 1991.

[59] ROBERTSON, N. and SEYMOUR, P. D., "Graph minors. II. Algorithmic aspects of tree-width," *J. Algorithms*, vol. 7, no. 3, pp. 309–322, 1986.

[60] ROBERTSON, N. and SEYMOUR, P. D., "Graph minors. V. Excluding a planar graph," *J. Combin. Theory Ser. B*, vol. 41, no. 1, pp. 92–114, 1986.

[61] ROBERTSON, N. and SEYMOUR, P. D., "Graph minors. VII. Disjoint paths on a surface," *J. Combin. Theory Ser. B*, vol. 45, no. 2, pp. 212–254, 1988.

[62] ROBERTSON, N. and SEYMOUR, P. D., "Graph minors. X. Obstructions to tree-decomposition," *J. Combin. Theory Ser. B*, vol. 52, no. 2, pp. 153–190, 1991.

[63] ROBERTSON, N. and SEYMOUR, P. D., "Graph minors. XI. Circuits on a surface," *J. Combin. Theory Ser. B*, vol. 60, no. 1, pp. 72–106, 1994.

[64] ROBERTSON, N. and SEYMOUR, P. D., "Graph minors. XX. Wagner's conjecture," *J. Combin. Theory Ser. B*, vol. 92, no. 2, pp. 325–357, 2004.

[65] SCHEFFLER, P., "A practical linear time algorithm for disjoint paths in graphs with bounded tree-width," Tech. Rep. 396/1994, TU Berlin, FB Mathematik, 1994.

[66] SCHRIJVER, A., "Decomposition of graphs on surfaces and a homotopic circulation theorem," *J. Combin. Theory Ser. B*, vol. 51, no. 2, pp. 161–210, 1991.

[67] SEYMOUR, P. D. and THOMAS, R., "Call routing and the ratcatcher," *Combinatorica*, vol. 14, no. 2, pp. 217–241, 1994.

[68] SLOANE, N., "Challenge problems: Independent sets in graphs," *http://www.research.att.com/ njas/doc/graphs.html*.

[69] THOMASSEN, C., "Embeddings of graphs with no short noncontractible cycles," *J. Combin. Theory Ser. B*, vol. 48, no. 2, pp. 155–177, 1990.

[70] VERWEIJ, B., *Selected Applications of Integer Programming: A Computational Study*. PhD thesis, Universiteit Utrecht, Utrecht, The Netherlands, 2000.

[71] Vitray, R., "The 2- and 3-representative projective planar embeddings," *J. Combin. Theory Ser. B*, vol. 54, no. 1, pp. 1–12, 1992.