# Delaunay Graphs for various geometric objects

A Thesis

Submitted For the Degree of

**Master of Science (Engineering)**

in the Faculty of Engineering

by

**Akanksha Agrawal**

Computer Science and Automation

Indian Institute of Science

BANGALORE – 560 012

NOVEMBER 2014

*To my family*

# Acknowledgements

I offer my sincerest gratitude to my research supervisor, Prof. Sathish Govindarjan, whose constant support and encouragement have been quintessential in the development of this thesis in particular and my own growth as a researcher in the field of Computer Science. I have been fortunate to have an advisor who gave me the freedom to explore on my own, and at the same time the guidance to recover when my steps faltered.

My greatest thanks to Dr. Neeldhara Misra for being an excellent mentor and a never-ending source of answers to all my questions about research, life and everything else. Special thanks to Dr. Saurabh Ray for his valuable suggestions with some of the problems discussed in this thesis.

I would like to thank Subramanya Bharadwaj, Abhijeet Khopkar and Pradeesha Ashok for their valuable guidance about pursuing research and for helping me enhance technical and personal skills. I thank my other lab mates Shivraj, Deepak, Jasine, Sumesh, Abhirukh, Aniket, Ninad, Prachi, Roohani and Shreyas for making my time spent in the lab enjoyable and for all the wonderful coffee-table discussions.

I am extremely indebted to the chairman of the Department, Dr. Y. Narahari for his invaluable support. I would take this opportunity to acknowledge Prof. Dilip P. Patil who has always encouraged me to pursue research and helped me improve my technical writing. I would like to thank all the faculty members of our department whose knowledge and experience, directly or indirectly has influenced me. I thank all the office-staff members for making my stay in the department comfortable and hassle-free. I am also thankful to other staff members who have made my stay in the department enjoyable. I am grateful to the Department of Computer Science and Automation for providing

necessary infrastructure and resources to accomplish my research work. Many thanks to the Indian Institute of Science for all the academic and non-academic facilities that made my stay at the institute amazing.

I must admit I learned a lot about Mathematics from the discussions with Govind Sharma and Sayantan Mukherjee. Discussions with Rohit Vaish on algorithms helped me improve my way of thinking with the problems. I am highly indebted to them for these wonderful discussions.

Life at IISc. would have been quite boring without my friends and batch-mates. I express my deepest gratitude to all of them for making my stay at IISc a memorable one. I am especially thankful to Abhijeet, Amit, Arpita, Narendra, Prateek, Rohit, Soham, Arun and Abhinav for all their care, support and all the weekend hangouts that made my stay at the institute exciting and wonderful.

Thanks to the Kung-Fu Club, Throw-ball club and swimming club at the institute where I improved my physical strength and spent a very enjoyable time with the club-members. I would also acknowledge IISc Gymkhana for providing all the recreational facilities.

I want to thank the Ministry of Human Resource Development, Department of Education for the scholarship to support during the course.

It was all made possible for me by my family's continual support and their unconditional love. I am thankful to my brother Anmol for his love and care, talking to whom is always refreshing. I am deeply indebted to my parents and grandparents for all their love and encouragement to pursue whatever I am interested in. I would not be able to return an iota of what I have received from them - still, I will feel content to dedicate this little piece of work to them.

# Publications based on this Thesis

1. Akanksha Agrawal, Sathish Govindarajan, and Neeldhara Misra. Vertex cover gets faster and harder on low degree graphs. *Computing and Combinatorics - 20th International Conference, COCOON 2014, Atlanta, GA, USA, August 4-6, 2014. Proceedings, volume 8591 of Lecture Notes in Computer Science*, pages 179-190. Springer, 2014.

# Abstract

Given a set of $n$ points $P \subset \mathbb{R}^2$, the Delaunay graph of $P$ for a family of geometric objects $\mathcal{C}$ is a graph defined as follows: the vertex set is $P$ and two points $p, p' \in P$ are connected by an edge if and only if there exists some $C \in \mathcal{C}$ containing $p, p'$ but no other point of $P$. Delaunay graph of circle is often called as Delaunay triangulation as each of its inner face is a triangle if no three points are co-linear and no four points are co-circular. The dual of the Delaunay triangulation is the Voronoi diagram, which is a well studied structure. The study of graph theoretic properties on Delaunay graphs was motivated by its application to wireless sensor networks, meshing, computer vision, computer graphics, computational geometry, height interpolation, etc.

The problem of finding an optimal vertex cover on a graph is a classical NP-hard problem. In this thesis we focus on the vertex cover problem on Delaunay graphs for geometric objects like axis-parallel slabs and circles (Delaunay triangulation).

1. We consider the vertex cover problem on Delaunay graph of axis-parallel slabs. It turns out that the Delaunay graph of axis-parallel slabs has a very special property — its edge set is the union of two Hamiltonian paths. Thus, our problem reduces to solving vertex cover on the class of graphs whose edge set is simply the union of two Hamiltonian Paths. We refer to such a graph as a *braid graph*.

   Despite the appealing structure, we show that deciding $k$-vertex cover on braid graphs is NP-complete. This involves a rather intricate reduction from the problem of finding a vertex cover on 2-connected cubic planar graphs.

2. Having established the NP-hardness of the vertex cover problem on braid graphs,

we pursue the question of improved fixed parameter algorithms on braid graphs. The best-known algorithm for vertex cover on general graphs has a running time of $\mathcal{O}(1.2738^k + kn)$ [CKX10]. We propose a branching based fixed parameter tractable algorithm with running time $\mathcal{O}^\star(1.2637^k)$ for graphs with maximum degree bounded by four. This improves the best known algorithm for this class, which surprisingly has been no better than the algorithm for general graphs. Note that this implies faster algorithms for the class of braid graphs (since they have maximum degree at most four).

3. A *triangulation* is a 2-connected plane graph in which all the faces except possibly the outer face are triangles, we often refer to such graphs as *triangulated graphs*. A *chordless-NST* is a triangulation that does not have chords or separating triangles (non-facial triangles).

   We focus on the computational problem of optimal vertex covers on triangulations, specifically chordless-NST. We call a triangulation *Delaunay realizable* if it is combinatorially equivalent to some Delaunay triangulation. Characterizations of Delaunay triangulations have been well studied in graph theory. Dillencourt and Smith [DS96] showed that *chordless-NST*s are Delaunay realizable. We show that for chordless-NST, deciding the vertex cover problem is NP-complete. We prove this by giving a reduction from vertex cover on 3-connected, triangle free planar graph to an instance of vertex cover on a chordless-NST.

4. If the outer face of a triangulation is also a triangle, then it is called a *maximal planar graph*. We prove that the vertex cover problem is NP-complete on maximal planar graphs by reducing an instance of vertex cover on a triangulated graph to an instance of vertex cover on a maximal planar graph.

# Contents

# List of Tables

# List of Figures

# Keywords

Delaunay triangulation, Delaunay graphs, Triangulations, Maximal-planar graphs, Axis-parallel slabs, Induced objects, Hitting Set, Vertex cover, NP-completeness, Fixed-parameter tractable algorithm, Above-guarantee parameterized algorithms.

# Chapter 1

# Introduction

Given a set of $n$ points $P \subset \mathbb{R}^2$, the *Delaunay graph* of $P$ for a family of geometric objects $\mathcal{C}$ is a graph defined as follows: the vertex set is $P$ and two points $p, p' \in P$ are connected by an edge if and only if there exists some $C \in \mathcal{C}$ containing $p, p'$ but no other point of $P$. The study of graph theoretic properties on Delaunay graphs was motivated by its application to wireless sensor networks, meshing, computer vision, computer graphics, computational geometry, height interpolation, etc.

Delaunay graph of circles is often called as *Delaunay triangulation* as each of its inner face is a triangle if no three points are co-linear and no four points are co-circular. The problem of triangulating the point set is an important problem in computational geometry. A Delaunay triangulation is a particular triangulation of the point set with the property that the circumcircle of any triangle does not contain any other point in its interior. It has been shown that among all possible triangulations, the Delaunay triangulation maximizes the minimum angle [Sib78] and hence it avoids long, skinny triangles.

The dual of the Delaunay triangulation is the *Voronoi diagram*, which is a well studied structure (see the book by Okabe et al. [OBSC09]). Delaunay triangulation contains important subgraphs like Gabriel graphs, Relative neighborhood graphs and Euclidean minimum spanning trees. Dillencourt [Dil96] showed that finding Hamiltonian cycle in Delaunay triangulation is NP-complete. Dillencourt [Dil90b] showed that Delaunay

triangulations are 1-tough and hence have a perfect matching. Efficient algorithms for constructing a Delaunay triangulation for a given point set are known [GKS92, CS89].

Even et al. [ELRS02] introduced the notion of *conflict free coloring* in various geometrically defined hypergraphs to solve the frequency assignment problems in cellular telephone networks. The estimation of minimum number of colors in a conflict-free coloring leads to the question of finding large independent sets in the Delaunay graph for the corresponding geometric object. The Delaunay graph of circles is a planar graph and hence has an independent set of size at least $n/4$. Chen et al. [CPST09] showed that there are $n$ element point sets in the plane whose Delaunay graph of axis-parallel rectangle has independent set of size at most $\mathcal{O}(n\frac{\log^2 \log n}{\log n})$.

The problem of finding an optimal vertex cover on a graph is a classical NP-hard problem. The dual of the vertex cover problem is the independent set problem. Vertex cover problem is a well studied problem even on restricted graphs classes. For some restricted graph classes like chordal graphs [Gav72], apple-free graphs [BLM10], (hole, dart)-free graphs [BCK12], etc. vertex cover is polynomial time solvable. Unfortunately, for many graph classes the problem remains NP-hard, like for 2-connected cubic graphs [Moh01], triangle-free graphs [Ueh96], etc.

The problem of finding a vertex cover is a special case of the *hitting set* question, when each set contains exactly two elements. We describe the hitting set problem for geometric objects induced by a point set. Let $P$ be a set of $n$ points in $\mathbb{R}^2$ and let $\mathcal{R}$ be the family of all distinct objects of a particular kind (disks, rectangles, triangles, ...), such that each object in $\mathcal{R}$ has a distinct tuple of points from $P$ on its boundary. For example, $\mathcal{R}$ could be the family of $\binom{n}{2}$ axis parallel rectangles such that each rectangle has a distinct pair of points of $P$ as its diagonal corners. $\mathcal{R}$ is called the set of all objects induced (spanned) by the point set $P$. An interesting question is to compute a minimum set of points in $P$ that "hits" all the objects in $\mathcal{R}$, i.e. a minimum sized subset $Q \subseteq P$ such that for each $R \in \mathcal{R}$, $R \cap Q \neq \varnothing$.

Various questions related to geometric objects induced by a point set have been studied in the last few decades. A classical result in discrete geometry is the *First Selection*

*Lemma* [BF84] which shows that there exists a point that is present in a constant fraction of triangles induced by $P$. Another interesting question is to compute a minimum set of points in $P$ that "hits" all the objects in $\mathcal{R}$, i.e. a minimum sized subset $Q \subseteq P$ such that for each $R \in \mathcal{R}$, $R \cap Q \neq \varnothing$. This is a special case of the classical Hitting Set problem, which we will refer to as *Hitting Set for Induced Objects*. For most geometric objects, it is not known if the *Hitting Set for induced objects* problem is polynomial time solvable. It is known to be polynomial time solvable for special objects skyline rectangles and halfspaces [CG14]. Recently, Rajgopal et al. [RAG+13] showed that this problem is NP-complete for lines. On the other hand, the hitting set problem, when asked in the context of induced geometric objects, often turns out to be exactly the vertex cover problem on Delaunay Graph for the corresponding geometric object.

In this thesis, we show that the vertex cover problem on Delaunay graphs for geometric objects like axis-parallel slabs and circles is NP-complete. Having known the problem to be NP-complete, there are several methods to cope with computational intractability like approximation algorithms, average-case analysis, randomized algorithms and heuristic methods, etc. A direct way of attacking NP-hard problems is providing a deterministic, exact algorithm. However, in this case, one has to deal with exponential running times. There have been increasing interest in faster exact solutions for NP-hard problems. Despite their exponential running times, these algorithms may be interesting from a theoretical as well as a practical point of view. *Fixed parameter tractable algorithms* is one of the approach to deal with the exponential running time which form a variant of exact, exponential time solutions mainly for NP-hard problems. In the next section we describe the *Fixed parameter tractable* algorithms more formally.

## 1.1 Fixed parameter tractable algorithms

A parameterization of a decision problem is a function that assigns an integer parameter $k$ to each input instance $x$. A parameterized problem is *fixed parameter tractable (FPT)* if it admits an algorithm with running time $f(k) \cdot |x|^c$ on input $x$ and parameter $k$, where

$f$ is an arbitrary function depending only on $k$ and $c$ is a constant. Fixed parameter tractability has been studied for various NP-complete problems [Nie06] like vertex cover, feedback vertex cover, graph bipartization, edge bipartization, graph crossing number, etc.

For instance, the NP-complete *vertex cover* problem is: given an undirected graph $G = (V, E)$ and a nonnegative integer $k$, does $G$ have a vertex cover of size at most $k$? In parameterized complexity theory, $k$ is called the parameter. The vertex cover problem is known to be fixed parameter tractable [Nie06]. So for the $v$ertex cover problem, the goal is to find a vertex cover of size at most $k$ in time $\mathcal{O}(c^k)$, and the "race" involves exploring algorithms that reduce the value of the constant $c$.

Vertex cover is one of the most well-studied problems in the context of fixed parameter algorithm design. The best-known algorithm for vertex cover on general graphs has a running time of $\mathcal{O}(1.2738^k + kn)$ [CKX10]. It enjoys a long list of improvements even on special graph classes. In particular, even for sub-cubic graphs (where the maximum degree is at most three, and the problem remains NP-complete), Xiao [Xia10] proposed an algorithm with running time $\mathcal{O}^\star(1.1616^k)$, improving on the previous best record of $\mathcal{O}^\star(1.1864^k)$ by Razgon [Raz09] and $\mathcal{O}^\star(1.1940^k)$ by Chen, Kanj and Xia [CKX03].

The standard parameterization of vertex cover uses $k$ as the parameter. However, in settings where we know a-priori that the value of $k$ is guaranteed to be large, then the standard parameter is not the ideal choice for parameterization. It is more reasonable to ask the so-called *above-guarantee* question, which is the following: is there a vertex cover of size at most $\mu(G) + \delta$? Here, $\mu(G)$ is an appropriate lower bound on the size of the vertex cover, for instance, the size of a maximum matching in $G$. We treat the difference between the optimal vertex cover and the known lower bound, namely $\delta$, as the parameter. We refer the reader to [NRRS12] for the state of the art. Apart from being a very natural question, it has turned out to be a particularly central problem when it was discovered that a number of problems (including, notably, Odd Cycle Transversal and Almost 2-SAT) reduce to Above Guarantee Vertex Cover.

## 1.2 Delaunay graph of axis-parallel slabs

Axis-parallel slabs are a special class of axis-parallel rectangles where two horizontal or two vertical sides are unbounded. Each pair of points $(x_1, y_1), (x_2, y_2) \in \mathbb{R}^2$ induces two axis-parallel slabs of the form $(-\infty, +\infty) \times [y_1, y_2]$ and $[x_1, x_2] \times (-\infty, +\infty)$. The Delaunay graph of axis-parallel rectangle is a well studied structure [AP13, Cha12, CPST09, ELRS02].

We focus on the computational problem of vertex cover on Delaunay graph of axis-parallel slabs. Note that this is even more structured than general axis-parallel rectangles, as it turns out that the corresponding Delaunay graph has a very special property — its edge set is the union of two Hamiltonian paths (refer Section 3.1) which we call as a braid graph. Thus, our problem reduces to solving vertex cover on braid graphs.

Despite the appealing structure, we show that – surprisingly – deciding $k$-vertex cover on this class of graphs is NP-complete. This involves a rather intricate reduction from the problem of finding a vertex cover on 2-connected cubic planar graphs. The reduction is a two step process. We transform the 2-connected cubic planar graph to a four regular graph with some special structure. We also appeal to the fact that the edge set of four-regular graphs can be partitioned into two two-factors, and the main challenge in the reduction involves stitching the components of the two-factors into two Hamiltonian paths while preserving the size of the vertex cover in an appropriate manner.

Having established the NP-hardness of the problem, we pursue the question of improved fixed parameter algorithms on this special case. Typically, these algorithms involve extensive case analysis on a cleverly designed search tree. We propose a branching algorithm with a running time $\mathcal{O}^\star(1.2637^k)$ for graphs with maximum degree at most four. This improves the best known algorithm for this class, which surprisingly has been no better than the algorithm for general graphs. Note that this implies faster algorithms for the class of braid graphs.

An application of computing vertex covers on Delaunay graph of axis-parallel slabs is the hitting set problem for induced axis-parallel slabs: Let $P$ be a set of points in the plane and $\mathcal{C}$ be the family of all axis-parallel slabs that contains some two points of $P$.

We would like to find a minimum sized subset $Q \subseteq P$ such that for each axis-parallel slab $C \in \mathcal{C}$, $C \cap Q \neq \varnothing$. We show in Section 3.1 that hitting all axis-parallel slabs in $\mathcal{C}$ is equivalent to the vertex cover problem on Delaunay graph of axis-parallel slabs for $P$.

## 1.3 Delaunay triangulations

A *triangulation* is a 2-connected plane graph in which all the faces except possibly the outer face are triangles, we often refer to such a graph as a *triangulated graph*. If the outer face is also a triangle, then it is called a *maximal planar graph*. Delaunay graph of circles is often called as Delaunay Triangulations, since each face is a triangle when the points are in general position (no four points co-circular and no three points co-linear). A *chordless-NST* is a triangulation that does not have chords or separating triangles (non-facial triangles) [DS96]. Triangulations and maximal planar graphs are important subclasses of planar graphs that have been studied with respect to various graph theoretic parameters [Dil90a, KP10, TW11]. Delaunay triangulation is another important subclass of triangulations.

Maximal independent set, connected dominating set on local Delaunay graphs (a variant of Delaunay triangulation) finds applications in network routing and design of network backbone [ALW$^+$03]. Deletion of independent set of vertices is used for decimation of points for terrain simplification where the underlying triangulations used for modeling are generally Delaunay triangulations [JS98]. Minimum vertex cover on 3-connected planar graphs finds application in the illumination problem of three dimensional convex polyhedra [DG95].

We focus on the computational problem of optimal vertex covers on triangulations, specifically *chordless-NST*s. Surprisingly, not much is known on the computational complexity of computing optimal vertex covers, dominating sets, etc. on triangulations or Delaunay triangulation. We call a triangulation *Delaunay realizable* if it is combinatorially equivalent to some Delaunay triangulation. Characterizations of Delaunay triangulations have been well studied in graph theory. Determining if a triangulation is

Delaunay realizable can be done in polynomial time [HMS00]. However, it is not known if the triangulation can be realized (embedded on $\mathbb{R}^2$) as a Delaunay triangulation of a point set in polynomial time. Dillencourt and Smith [DS96] showed that *chordless-NST*s are Delaunay realizable. We show that for chordless-NSTs, deciding the vertex cover problem is NP-complete. We prove this by giving a reduction from vertex cover on 3-connected, triangle free planar graph to an instance of vertex cover on a chordless-NST. Note this implies that the vertex cover problem on graphs realizable as Delaunay triangulation and vertex cover on triangulations is NP-complete. We extend this reduction to show that the problem of vertex cover on maximal-planar graphs is also NP-complete.

## 1.4 Organization of rest of the thesis

Chapter 2 sets up the basic notations and terminologies used in this thesis. Chapter 3 gives the proof of NP-completeness of vertex cover problem on Delaunay graph for axis-parallel slabs and a fixed parameter tractable algorithm for the vertex cover problem on graphs with maximum degree four. Chapter 4 gives the proof of NP-completeness of vertex cover problem on chordless-NST and on maximal planar graphs.

# Chapter 2

# Preliminaries

We establish some of the basic notations and terminologies that will be used in the thesis. We refer the reader to [Die12] for details on standard graph theoretic notation and terminology we use in this thesis.

We denote the set of natural numbers by $\mathbb{N}$ and set of real numbers by $\mathbb{R}$. For a natural number $n$, we use $[n]$ to denote the set $\{1, 2, \ldots, n\}$. For a finite set $A$ we denote by $\mathfrak{S}_A$ the set of all permutations of the elements of set $A$. To describe the running times of our algorithms, we will use the $\mathcal{O}^*$ notation. Given $f : \mathbb{N} \to \mathbb{N}$, we define $\mathcal{O}^*(f(k))$ to be $\mathcal{O}(f(k) \cdot p(n))$, where $p(\cdot)$ is some polynomial function. That is, the $\mathcal{O}^*$ notation suppresses polynomial factors in the running-time expression.

**Graphs.** In the following, let $G = (V, E)$ be a graph. For any non-empty subset $W \subseteq V$, the subgraph of $G$ induced by $W$ is denoted by $G[W]$; its vertex set is $W$ and its edge set consists of all those edges of $E$ with both endpoints in $W$. For $W \subseteq V$, by $G \setminus W$ we denote the graph obtained by deleting the vertices in $W$ and all edges which are incident to at least one vertex in $W$. For a graph $G$ we denote the set of vertices of $G$ by $V(G)$ and set of edges by $E(G)$. We denote a subgraph $H$ of $G$ by $H \subseteq G$, where $V(H) \subseteq V(G)$ and $E(H) \subseteq \{(u, v) \in E(G) | u, v \in V(H)\}$.

For $v \in V$, we denote the open-neighborhood of $v$ by $N(v) = \{u \in V | (u, v) \in E\}$, closed-neighborhood of $v$ by $N[v] = N(v) \cup \{v\}$, second-open neighborhood by

$N_2(v) = \{u \in V | \exists u' \in N(v) \text{ s.t. } (u, u') \in E\}$ and second-closed neighborhood by $N_2[v] = N_2(v) \cup N[v]$. For a pair of vertices $u$ and $v$, the common neighborhood of $u$ and $v$ is the set of vertices that are adjacent to both $u$ and $v$. In this context, a vertex $w$ is called a *private neighbor* of $u$ if $(w, u)$ is an edge and $(w, v)$ is not an edge. We denote the degree of a vertex $v$ in graph $G$ by $d_G(v)$ and drop the subscript $G$ when the context is clear. A *simplicial* vertex is a vertex whose neighborhood forms a clique. By $K_n$ we denote the complete graph on $n$ vertices.

**Definition 1** (Vertex cover)**.** A *vertex cover* is a subset of vertices $S \subseteq V$ such that $G \setminus S$ has no edges.

The parameterized *Vertex Cover* problem is as follows:

**Input:** A graph $G = (V, E)$ and a nonnegative integer $k$.

**Question:** Is there a subset $S \subseteq V$ with $|S| \leq k$ such that $S$ is a vertex cover of $G$?

**Definition 2** (Path)**.** A *path* in a graph is a sequence of distinct vertices $v_0, v_1, \ldots, v_{k-1}$ such that $(v_i, v_{i+1})$ is an edge for all $0 \leq i < k - 1$.

**Definition 3** (Hamiltonian Path)**.** A *Hamiltonian path* of a graph $G$ is a path featuring every vertex of $G$.

**Definition 4** (Cycle)**.** A *cycle* in a graph is a sequence of distinct vertices $v_0, v_1, \ldots, v_{k-1}$ such that $(v_i, v_{i+1})$ is an edge for all $0 \leq i \leq k - 1$ (index computed modulo $k$).

**Definition 5** ($n$-Wheel)**.** A graph $W$ on $n \geq 4$ vertices is an *$n$-wheel* if there is a special vertex $s$ called the center of the wheel, such that $W \setminus \{s\}$ is an induced cycle and $(s, w) \in E$, $\forall w \in V(W) \setminus \{s\}$.

**Definition 6** ($k$-factor)**.** A $k$-regular spanning subgraph of graph $G$ is called a *$k$-factor* of $G$.

**Definition 7** (Delaunay Graph)**.** Given a set of $n$ points $P \subset \mathbb{R}^2$, the *Delaunay graph* of $P$ for a family of geometric objects $\mathcal{C}$ is a graph defined as follows: the vertex set is $P$ and two points $p, p' \in P$ are connected by an edge if and only if there exists some $C \in \mathcal{C}$ containing $p, p'$ but no other point of $P$.

**Definition 8** (Triangulation)**.** A 2-connected plane graph is called as a *triangulation* or a *triangulated graph* if all its faces except possibly the outer face are triangles.

**Definition 9** (Maximal planar graph)**.** A triangulation with the outer face as a triangle is called as a *maximal planar graph.*

For a plane graph $G$ with outer-face $f$, an edge $(v, v')$ with $v, v' \in V(f)$ is called a *chord* if $(v, v')$ is not an edge of the outer-face. We call an outer-face vertex $v$ to be *chordal* if $\exists v' \in V(f)$ s.t. $(v, v') \in E$ is a chord, otherwise it is a non-chordal vertex.

**Definition 10** (Chordless-NST)**.** A *chordless-NST* is a triangulation that does not have chords or separating triangles (non-facial triangles).

**Definition 11** (Braid graphs)**.** A graph $G$ on the vertex set $[n]$ is a *braid* graph if the edges of the graph is the union of two Hamiltonian paths. In other words, there exist permutations $\sigma$, $\tau$ of the vertex set for which $E(G) = \{(\sigma(i), \sigma(i+1)) \mid 1 \leq i \leq n-1\} \cup \{(\tau(i), \tau(i+1)) \mid 1 \leq i \leq n-1\}$.

**Parameterized Complexity.** A parameterized problem $\Pi$ is a subset of $\Gamma^* \times \mathbb{N}$, where $\Gamma$ is a finite alphabet. An instance of a parameterized problem is a tuple $(x, k)$, where $x$ is a classical problem instance, and $k$ is called the parameter. A central notion in parameterized complexity is *fixed parameter tractability (FPT)* which means, for a given instance $(x, k)$ of a problem, we can decide it in time $f(k) \cdot p(|x|)$, where $f$ is an arbitrary function of $k$ and $p$ is a polynomial in the input size.

A branching based algorithm is an algorithm in which at each of the steps we make alternative choices regarding some of the items of the solution to a given problem. We split the problem instance into several instances (branches) that are further processed independently and recursively, resulting in a tree of instances. The branching rules are designed in such a way that the parameter value drops (reduces) in every branch. A branching vector is a vector indicating the drop in the parameter in each of the branch. We will be using the standard notation with regards to branching vectors as described in [Nie06].

The algorithm for vertex cover that we propose is a branching based algorithm. When we say that we *branch on a vertex* $v$, we mean that we recursively generate two instances, one where $v$ belongs to the vertex cover, the other where $v$ does not belong to the vertex cover. This is a standard method of exhaustive branching, where the size of the vertex cover drops respectively, by one and $d(v)$ in the two branches (since the neighbors of $v$ are forced to be in the vertex cover when $v$ does not belong to the vertex cover).

**Preprocessing a simplicial vertex.** For a vertex $v$, if the graph induced by $N[v]$ is a clique, then it is easy to see that there is a minimum vertex cover containing $N(v)$ and not containing $v$ (by a standard shifting argument). We can therefore, preprocess the graph in such a situation by deleting $N[v]$ and reducing the size of the vertex cover by $|N(v)|$.

**Folding a degree two vertex.** In our algorithms we make extensive use of the *folding* technique, as described in past work [CKJ99, CKX06]. This allows us to preprocess vertices of degree two in polynomial time, while also reducing the size of the vertex cover sought by one. We briefly describe how we might handle degree two vertices in polynomial time. Suppose $v$ is a degree two vertex in the graph $G$ with two neighbors $u$ and $w$ such that $u$ and $w$ are not adjacent to each other. We construct a new graph $G'$ as follows: remove the vertices $v$, $u$, and $w$ and introduce a new vertex $v^\star$ that is adjacent to all neighbors of the vertices $u$ and $w$ in $G$ (other than $v$). We say that the graph $G'$ is obtained from the graph $G$ by "folding" the vertex $v$, and we say that $v^\star$ is the vertex generated by folding $v$, or simply that $v^\star$ is the *folded vertex* (when the context is clear). It turns out that the folding operation preserves equivalence, as shown below.

**Proposition 1.** *[CKJ99, Lemma 2.3] Let $G$ be a graph obtained by folding a degree two vertex $v$ in a graph $G$, where the two neighbors of $v$ are not adjacent to each other. Then the graph $G$ has a vertex cover of size bounded by $k$ if and only if the graph $G'$ has a vertex cover of size bounded by $(k-1)$.*

# Chapter 3

# Vertex cover on Delaunay graphs of axis-parallel slabs

In this chapter, we focus on the problem of vertex cover on Delaunay graph of axis-parallel slabs. We show that the hitting set for the class of axis-parallel slabs induced by a point set $P$ is exactly the vertex cover of the Delaunay graph of axis-parallel slabs for $P$. It turns out that the Delaunay graph of axis-parallel slab has a very special property — its edge set is the union of two Hamiltonian paths. Thus, our problem reduces to solving vertex cover on the class of *Braid graphs* (refer Definition 11).

We show that deciding the vertex cover on this class of graphs is NP-complete. Having established the NP-completeness of the problem, we pursue the question of fixed parameter tractable algorithms on graphs with maximum degree bounded by four which includes the family of Braid graphs. Typically, these algorithms involve extensive case analysis on a cleverly designed search tree. We propose a branching based fixed parameter tractable algorithm with a running time $\mathcal{O}^\star(1.2637^k)$ for graphs with maximum degree bounded by four.

## 3.1 Hitting set for induced axis-parallel slabs

Consider a point set $P$, if an axis-parallel slab (say horizontal axis-parallel slab) induced by $p, q \in P$ contains another point, say $r \in P$, then hitting the horizontal axis-parallel slabs induced by $p$ and $r$ will hit the axis-parallel slab induced by $p$ and $q$. An analogous argument can be given for a vertical axis-parallel slab induced by $p, q \in P$ containing another point $r \in P$. Therefore, it is sufficient to hit only empty horizontal and vertical axis-parallel slabs. In the above setting this amounts to hitting all consecutive slabs in the horizontal and vertical directions. Note that the Delaunay graph of axis-parallel slabs has an edge $(p, q) \in E$ if and only if there is an empty axis-parallel slab passing through $p$ and $q$. Thus minimum hitting set for all axis-parallel slabs is exactly the same as computing a minimum vertex cover on the Delaunay of axis-parallel slabs for the point set $P$.

In Lemma 2 and Lemma 3 we show that the problem of finding an optimal hitting set for the family of all axis-parallel slabs induced by a point set is equivalent to the problem of finding a vertex cover of a braid graph with the associated permutation $\sigma$ and $\tau$ defining the two Hamiltonian paths.

**Lemma 2.** *The problem of finding a vertex cover on a braid graph $G$, with $V(G) = [n]$ and the associated permutations $\sigma, \tau$ can be reduced to an instance of hitting set for the collection of all axis-parallel slabs induced by a point set.*

*Proof.* Given an instance of vertex cover on a braid graph $G$, with $V(G) = [n]$ and permutations $\sigma$ and $\tau$, we create $n$ points in $\mathbb{R}^2$ in an $(n \times n)$-grid as follows. We assume, by renaming if necessary, that $\sigma$ is the identity permutation. For every $1 \leq i \leq n$, we let $p_i = (i, \tau^{-1}(i))$.

$p_i, p_j$ are adjacent to each with respect to $x$-coordinates if and only if $i$ and $j$ are adjacent to each other in $\sigma$. Similarly $p_i, p_j$ are adjacent to each with respect to $y$-coordinates if and only if $i$ and $j$ are adjacent to each other in $\tau$. This implies that $(i, j) \in E(G)$ if and only if there exists an empty axis-parallel slab containing $p_i$ and $p_j$. Thus for the braid graph $G$, we have an instance of hitting all axis-parallel slabs induced

by a point set.

□

**Lemma 3.** *The problem of finding a hitting set for all induced axis-parallel slabs by a point set $P$ can be reduced to the problem of finding a vertex cover on a braid graph.*

*Proof.* From the given point set $P$, we sort the points in $P$ according to their $x$-coordinates to obtain a permutation $\sigma$ of the point set. Similarly, we sort with respect to $y$-coordinate to get a permutation $\tau$. Note that there exists a empty axis-parallel slab between two points if and only if they are adjacent with respect to at least one of the $x$- or $y$-coordinates. These are, on the other hand, precisely the edges in the braid graph with $\sigma$ and $\tau$ as the permutations. □

## 3.2 NP-completeness of vertex cover on Braid graphs

In this section, we show that the problem of determining a vertex cover on the class of braid graphs is hard even when the permutations of the braid graph are given as the input.

The intuition for the hardness is the following. Consider a four-regular graph. By Corollary 2.1.5(Petersen 1891) [Die12], we know that the edges of a four-regular graph can be partitioned into two sets, each of which would be a 2-factor (refer Definition 6) in the graph. A 2-factor is a two-regular graph which is essentially disjoint union of cycles. In other words, every four-regular graph can be thought of as an union of two collections of disjoint cycles, defined on the same vertex set. It is conceivable that these cycles can be patched together into paths, leading us to a braid graph. As it turns out, for such a patching, we need to have some control over the cycles in the decomposition to begin with. So we start with an instance of vertex cover on a 2-connected cubic planar graph, morph such an instance to a four-regular graph while keeping track of a special cycle decomposition, which we later exploit for the "stitching" of cycles into Hamiltonian paths.

Formally, therefore, the proof is by a reduction from vertex cover on a 2-connected cubic planar graph to an instance of vertex cover on a braid graph, noting that [Moh01] shows the NP-hardness of vertex cover on 2-connected cubic planar graphs. We describe the construction in two stages, first showing the transformation to a four-regular graph and then proceeding to illustrate the transformation to a braid graph.

### 3.2.1 2-connected cubic planar graphs to 4-regular graphs

Consider an instance of vertex cover on a 2-connected cubic planar graph $G = (V, E)$, where $|V| = n$. We transform this graph to a four-regular graph in two steps. This transformation is important because for turning a four-regular graph into an union of two Hamiltonian paths, we need the underlying decomposition into 2-factors to have certain properties, which we will ensure in the first half of the reduction.

The transformation from 2-connected cubic planar graphs to four-regular graphs happens in two stages. First, we replace certain edges with gadgets that only serve to elongate certain paths in the graph. This is a technical artifact that will be useful later. Next, we add gadgets that increase the degree of every vertex so as to obtain a four-regular graph.

We begin by making two copies of $G$, say, $G_u$ and $G_v$. We let $G_u = (V_u, E_u), G_v = (V_v, E_v)$ where, $V_u = \{u_0, u_1, ..., u_{n-1}\}, V_v = \{v_0, v_1, ..., v_{n-1}\}$. We take two copies of $G$ to make the number of vertices multiple of four which will be useful in the later stages.

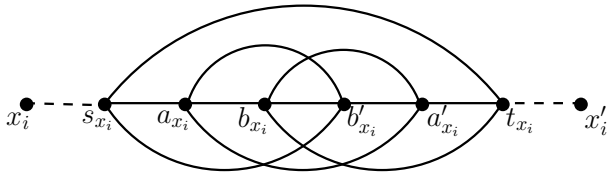



Figure 3.1: Gadget $J_{x_i}$ with solid lines showing gadget edges and dotted lines its connection to outside vertices.

It is shown in [CS12] that a planar cubic graph with no cut edge has exponentially many perfect matchings. Since we have a 2-connected cubic planar graph, there are

evidently no cut edges and maximum matching is known to be polynomial time computable by [MV80]. Let $M_u = \{(u_0, u'_0), (u_1, u'_1), .., (u_{\frac{n}{2}-1}, u'_{\frac{n}{2}-1})\}$ be a perfect matching of $G_u$ (after renaming of vertices if necessary). Let the corresponding matching of $G_v$ be $M_v = \{(v_0, v'_0), (v_1, v'_1), ..., (v_{\frac{n}{2}-1}, v'_{\frac{n}{2}-1})\}$.

For $x \in \{u, v\}$ we now describe how to construct $\hat{G}_x$ from $G_x$. We use gadget $J_{x_i}$ shown in Figure 3.1, where the subscript $x_i$, $i$ corresponds to the numbering of edge $(x_i, x'_i)$ in the matching $M_x$ of graph $G_x$. We define the sequence $\rho(J_{x_i}) := (s_{x_i}, a_{x_i}, b_{x_i}, b'_{x_i}, a'_{x_i}, t_{x_i})$, notice that this is a path in $J_{x_i}$ and we also have the sequence $\rho'(J_{x_i}) := (s_{x_i}, t_{x_i}, b_{x_i}, a'_{x_i}, a_{x_i}, b'_{x_i}, s_{x_i})$, which is a cycle in $J_{x_i}$. It is easy to see that all edges in $J_{x_i}$ is either in the path $\rho(J_{x_i})$ or in the cycle $\rho'(J_{x_i})$. We will refer to these sequences later, when we are specifying how the reduced graph is an union of two Hamiltonian paths.

We construct $\hat{G}_x$ from $G_x$ as follows: Fix a matching $M_x = \{(x_0, x'_0),(x_1, x'_1),...,(x_{\frac{n}{2}-1}, x'_{\frac{n}{2}-1})\}$. Replace every edge in this matching with the gadget $J_{x_i}$. More formally, we have:

$$V(\hat{G}_x) = V(G_x) \uplus \left( \bigcup_{0 \le i < n/2} V(J_{x_i}) \right),$$

$$E(\hat{G}_x) = \left( E(G_x) \setminus \left( \bigcup_{0 \le i < n/2} \{(x_i, x'_i)\} \right) \right) \uplus \left( \bigcup_{0 \le i < n/2} (E(J_{x_i}) \uplus \{(x_i, s_{x_i}), (t_{x_i}, x'_i)\}) \right).$$

In Lemma 4 we show the minimum number of vertices required to cover edges in $E(J_{x_i}) \cup \{(x_i, s_{x_i}), (x'_i, t_{x_i})\}$ based on whether $x_i, x'_i$ are included in the vertex cover or not.

**Lemma 4.** *Consider the graph $J_i = (V(J_{x_i}) \uplus \{x_i, x'_i\}, E(J_{x_i}) \uplus \{(x_i, s_{x_i}), (t_{x_i}, x'_i)\})$ described above. Let $S$ be an optimal vertex cover of $J_i$, and let $S_i$ denote $S \cap V(J_{x_i})$. If $S$ includes none of $x_i, x'_i$, then $|S_i| = 5$, otherwise $|S_i| = 4$.*

*Proof.* The vertices $a_{x_i}, b_{x_i}, b'_{x_i}, a'_{x_i}$ induces a $K_4$, therefore any vertex cover includes at least 3 vertices among them. The following cases arise depending on whether $x_i, x'_i$ is included in $S$ or not.

Case 1 If $x_i, x_i' \notin S$ then we are forced to include $s_{x_i}, t_{x_i}$ in $S_i$. So we need at least five vertices in $S_i$ and note that $X := \{s_{x_i}, t_{x_i}, a_{x_i}, a'_{x_i}, b_{x_i}\}$ is a set of five vertices that covers all edges in $E(J_i)$.

Case 2 Suppose $S$ picks exactly one of $x_i$ and $x_i'$. In particular, let us say that $x_i \notin S$, then we have $s_{x_i}$ in $S_i$. Therefore, we need at least four vertices in $S_i$. Note that $X := \{s_{x_i}, a_{x_i}, b_{x_i}, a'_{x_i}\}$ is a set of four vertices that covers all edges in $E(J_i)$.

Case 3 Finally, let $x_i, x_i' \in S$. Since $(s_{x_i}, t_{x_i}) \in E(J_i)$, at least one of $s_{x_i}, t_{x_i}$ should be in $S$. The rest of the argument is analogous to the previous case.

□

In Lemma 5, we establish the relation between the vertex cover of $G_x$ and vertex cover of $\hat{G}_x$.

**Lemma 5.** *For $x \in \{u, v\}$, the graph $G_x$ admits a vertex cover of size $p$ if and only if the graph $\hat{G}_x$ has a vertex cover of size $(p + 2n)$.*

*Proof.* In the forward direction, consider a vertex cover $S$ of $G_x$. Note that for every matching edge $(x_i, x_i') \in M_x$, $S \cap \{x_i, x_i'\} \neq \varnothing$. So for each $J_{x_i}$ corresponding to $(x_i, x_i') \in M_x$ we need four more vertices to cover $E(J_{x_i}) \cup \{(x_i, s_{x_i}), (t_{x_i}, x_i')\}$ by Lemma 4. But $|M_x| = n/2$, so $p + 4\frac{n}{2} = p + 2n$ vertices are sufficient to cover $E(\hat{G}_x)$.

In the reverse direction, let $S$ be a vertex cover of size $(p + 2n)$ for $\hat{G}_x$, and let $S' = S \cap V(G_x)$. For each gadget $J_{x_i}$, $0 \leq i < n/2$, we require at least four vertices from $V(J_{x_i})$ by Lemma 4. Therefore, for $S$ to cover all other edges (except the matching edge) of $G_x$ in $\hat{G}_x$ we are left with at most $p$ vertices. Note that $S'$ covers all edges in $G_x$ except possibly edges $(x_i, x_i') \in M_x$. For each gadget $J_{x_i}$ inserted between $(x_i, x_i') \in M_x$ we know if both of $x_i, x_i' \notin S$ then from Lemma 4 we require five additional vertices to cover $E(J_{x_i}) \cup \{(x_i, s_{x_i}), (t_{x_i}, x_i')\} \subset E(\hat{G}_x)$. For covering the edge $(x_i, x_i')$ in $G_x$ we need at least one of $x_i, x_i'$. We can modify $S$ to include any one of $x_i, x_i'$ and four more vertices from $V(J_{x_i})$ (note that the size of the vertex cover remains unchanged after this

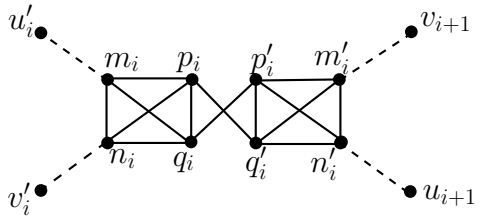



Figure 3.2: Gadget $\tilde{J}_i$ with solid lines showing gadget edges and dotted lines its connection to outside vertices.

operation). After repeating this operation for all matching edges where it is necessary, we have that $S'$ forms a vertex cover of $G_x$ of size at most $p$.

□

We now turn to the gadgets that add to the degree of every vertex in the graph, turning it into a four-regular graph. For this we need a gadget, which we refer to as $\tilde{J}_i$ (indexed by $i$), as shown in Figure 3.2. Vertices $p_i, q_i, m_i, n_i$ and $p'_i, q'_i, m'_i, n'_i$ respectively induce two complete graphs. Therefore, irrespective of whether $u'_i, v'_i, u_{i+1}, v_{i+1}$ is included in the vertex cover or not, we require at least 6 vertices among the vertices of $\tilde{J}_i$. Further, we can include $m_i, n_i, m'_i, n'_i, p_i, p'_i$ in vertex cover to cover edges incident to at least one of the vertices in $V(\tilde{J}_i)$. Now we are ready to describe the actual construction.

We arbitrarily order the edges in matching, say $M_u = \{(u_0, u'_0), (u_1, u'_1),\ ..., (u_{\frac{n}{2}-1},\ u'_{\frac{n}{2}-1}\ )\}$ of $G_u$. The corresponding ordering of matching of $G_v$ is $M_v = \{(v_0, v'_0), (v_1\ , v'_1), ..., (v_{\frac{n}{2}-1},\ v'_{\frac{n}{2}-1})\}$.

We will follow this ordering in every step of reduction wherever required. Note that all vertices of $G_x$ in $\hat{G}_x$ are still degree three vertices, where $x \in \{u, v\}$. We insert the gadget $\tilde{J}_i$ between edges $(u_i, u'_i), (u_{i+1}, u'_{i+1}) \in M_u$ and $(v_i, v'_i), (v_{i+1}, v'_{i+1}) \in M_v$ to increase the degree of the vertices $u'_i, u_{i+1}, v'_i$ and $v_{i+1}$. We do this by adding edges $(u'_i, m_i), (u_{i+1}, n'_i), (v'_i, n_i), (v_{i+1}, m'_i)$ for $0 \leq i \leq \frac{n}{2} - 1$, where the index is computed modulo $n/2$. We refer to the graph constructed above as $\tilde{G}$. It is easy to see that $\tilde{G}$ thus obtained is a 4-regular graph.

In Lemma 6, we establish the relation between vertex cover of $\tilde{G}$ and vertex cover of $\hat{G}_u \uplus \hat{G}_v$.

**Lemma 6.** *The graph $\hat{G} = \hat{G}_u \uplus \hat{G}_v$ has a vertex cover of size $p$ if and only if, $\tilde{G}$ has a vertex cover of size $(p + 3n)$.*

*Proof.* In the forward direction, suppose $\hat{G}$ has a vertex cover of size $p$. Since $\hat{G} \subseteq \tilde{G}$, only extra edges in $\tilde{G}$ are those adjacent to vertices of gadgets $\tilde{J}_i$ for $0 \leq i < n/2$. From $V(\tilde{J}_i)$ if we include vertices $\{m_i, m'_i, n_i, n'_i, p_i, p'_i\}$, then they will cover all edges that are adjacent to at least one vertex in $V(\tilde{J}_i)$, for all $0 \leq i < n/2$. This implies a $p + 3n$ sized vertex cover for $\tilde{G}$.

In the reverse direction, let $S$ be a vertex cover of size $p + 3n$ for $\tilde{G}$, and let $S' = V(\hat{G}) \cap S$. For each $\tilde{J}_i, 0 \leq i < n/2$, we need at least 6 vertices to cover edges adjacent to $V(J_i)$. Therefore $|S'| \leq p$. It is easy to see that $S'$ is a vertex cover for $\hat{G}$. □

Again, for ease of describing paths at the end of this discussion, we define $\tau(\tilde{J}_i) = (m_i, p_i, n_i, q_i, p'_i, m'_i, q'_i, n'_i)$ and $\tau'(\tilde{J}_i) = (n_i, m_i, q_i, p_i, q'_i, p'_i, n'_i, m'_i)$ be the two paths that cover all vertices and edges of $\tilde{J}_i$. Combining the two steps of the reduction above, we have the following.

**Corollary 1.** Let $G_{uv} = G_u \uplus G_v$. The graph $G_{uv}$ admits a vertex cover of size $p$ if and only if the graph $\tilde{G}$ has a vertex cover of size $(p + 7n)$.

**2-factor decomposition of $\tilde{G}$.** From Corollary 2.1.5(Petersen 1891) [Die12], every $2k$-regular graph has a 2-factor as a subgraph. Here, we give an explicit partition of $\tilde{G}$ into two 2-factors, namely $H, H'$. Initially, let $H, H'$ be empty (no vertices). We have fixed a matching $M_u = \{(u_0, u'_0), (u_1, u'_1), ..., (u_{\frac{n}{2}-1}, u'_{\frac{n}{2}-1})\}$ of $G_u$ and $M_v = \{(v_0, v'_0), (v_1, v'_1), ..., (v_{\frac{n}{2}-1}, v'_{\frac{n}{2}-1})\}$ of $G_v$ corresponding to which $\tilde{G}$ was constructed.

Recall that for the construction of $\tilde{G}$, we deleted the matching edge $(x_i, x'_i)$ and inserted gadget $J_{x_i}$, for $0 \leq i < \frac{n}{2}$ and $x \in \{u, v\}$. We increased the degree of each vertices $u'_j, u_{j+1} \in V(G_u) \subset V(\tilde{G})$ corresponding to matching edges $(u_j, u'_j), (u_{j+1}, u'_{j+1})$ and $v'_j, v_{j+1} \in V(G_v) \subset V(\tilde{G})$ corresponding to matching edge $(v_j, v'_j), (v_{j+1}, v'_{j+1})$ by connecting it to gadget $\tilde{J}_i$, for $0 \leq j < \frac{n}{2}$ (index computed modulo $n/2$).

We will construct two 2-factors which will be convenient for us in the next phase of the reduction. These 2-factors will be highly structured in the following sense. The first

cycle in both 2-factors will involve all the vertices of $G_u$ and $G_v$ respectively, and some vertices from the gadgets. The rest of the graph now decomposes into a collection of cycles which get distributed in a natural way. We now describe this formally.

The first cycle $C_0$ in $H$ contains all vertices of $\hat{G}_u$ and $\tilde{J}_i$, $0 \leq i < \frac{n}{2}$ that are in $\tilde{G}$. Similarly the first cycle $C'_0$ in $H'$ contains all vertices of $\hat{G}_v$ and $\tilde{J}_i$, $0 \leq i < \frac{n}{2}$.

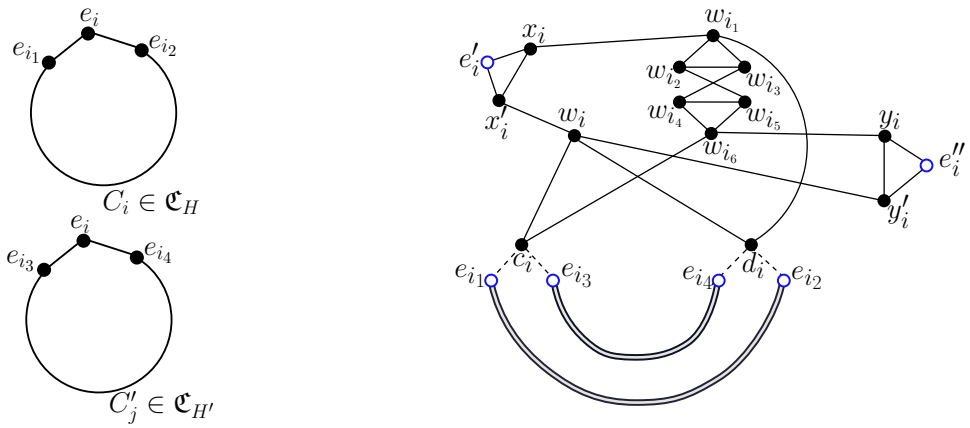
Specifically, the cycle $C_0$ is $u_0 \to \rho(J_{u_0}) \to u'_0 \to \tau(\tilde{J}_0) \to u_1 \to \rho(J_{u_1}) \to u'_1 \to \tau(\tilde{J}_1) \to ... \to u_{\frac{n}{2}-1} \to \rho(J_{u_{\frac{n}{2}-1}}) \to u'_{\frac{n}{2}-1} \to \tau(\tilde{J}_{\frac{n}{2}-1}) \to u_0$. Similarly, the cycle $C'_0$ is $v_0 \to \rho(J_{v_0}) \to v'_0 \to \tau'(\tilde{J}_0) \to v_1 \to \rho(J_{v_1}) \to v'_1 \to \tau'(\tilde{J}_1) \to ... \to v_{\frac{n}{2}-1} \to \rho(J_{v_{\frac{n}{2}-1}}) \to v'_{\frac{n}{2}-1} \to \tau'(\tilde{J}_{\frac{n}{2}-1}) \to v_0$.

For $H$ to be one of the factors we require cycles in $H$ containing vertices of $\hat{G}_v$ present in $\tilde{G}$, so we include the cycles $C_{i+1} = \rho'(J_{v_i})$, for $0 \leq i < n/2$ and let $\mathcal{C}_J = \{C_1, C_2, ..., C_{\frac{n}{2}}\}$. Since we have already used two degrees of vertices of $G_v$ we are left with degree two from each vertex which forms a disjoint union of cycles, so we arbitrarily order these cycles from $C_{\frac{n}{2}+1}, C_{\frac{n}{2}+2}, ..., C_k$ and include them in $H$. We let $\mathcal{C}_v = \{C_{\frac{n}{2}+1}, C_{\frac{n}{2}+2}, ..., C_k\}$. Similarly we include in $H'$ the cycles $C'_{i+1} = \rho'(J_{u_i})$, for $0 \leq i < n/2$ and refer to them by $\mathcal{C}'_J$ and include the graph $\hat{G}_u$ after deleting already included edge in corresponding cycle and refer to cycles obtained after deletion by $\mathcal{C}_u = \{C'_{\frac{n}{2}+1}, C'_{\frac{n}{2}+2}, ..., C'_k\}$.

### 3.2.2 4-regular graph with 2-factoring $H, H'$ to a braid graph

Let $\mathfrak{C}_H, \mathfrak{C}_{H'}$ be set of cycles in $H$ and $H'$ respectively. We will choose $V_H \subseteq V(\tilde{G})$ such that $\forall C \in \mathcal{C}_J \cup \mathcal{C}_v$, $|V(C) \cap V_H| = 1$ and $\forall C' \in \mathcal{C}'_J \cup \mathcal{C}_u$, $|V(C') \cap V_H| = 1$. We will delete vertices in $V_H$ and insert some other set of vertices, the aim is to break all cycles at some vertices and stitch together the broken cycles in $H$ to form one of the Hamiltonian path and similarly form the other Hamiltonian path using the cycles in $H'$.

Let $V_1 = \{b_{x_i} \in V(J_{x_i}) \ | x \in \{u, v\}$ and $0 \leq i < \frac{n}{2}\}$. Observe that the cycles in $\mathcal{C}_u$ and $\mathcal{C}_v$ are disjoint among them (no common vertex), so we select one vertex from each cycle $C \in \mathcal{C}_u \cup \mathcal{C}_v$ and include it in $V_2$. Let $V_H = V_1 \uplus V_2$. Note that the vertex selected from the cycles in $\mathcal{C}_u$ and $\mathcal{C}'_J$ are present in the cycle $C_0$, similarly vertex selected from




(a) (b)

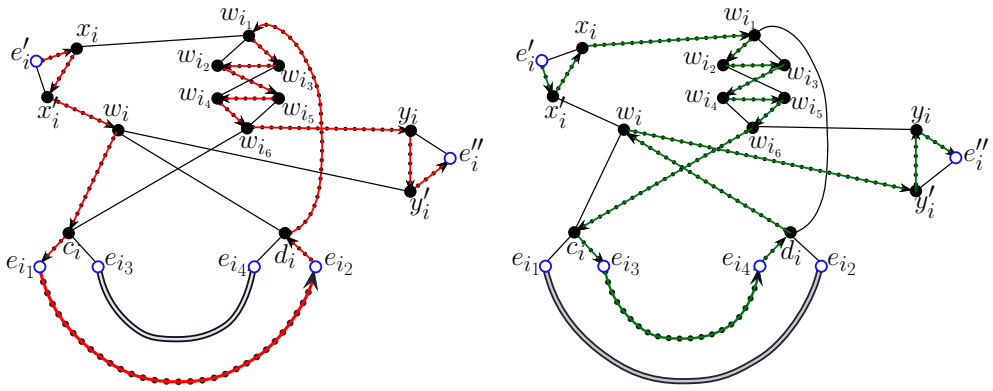
Figure 3.3: (a) Cycles $C_i \in \mathfrak{C}_H, C'_j \in \mathfrak{C}_{H'}$, $C_i, C'_j$ containing $e_i$. (b) Gadget $W_i$, with dotted lines showing its connection to neighbors of $e_i$.

cycles in $\mathcal{C}_v$ and $\mathcal{C}_J$ are present in the cycle $C'_0$.

We let the vertices in $V_H$ from cycles $C_1, C_2, ..., C_k$ to be $e_1, e_2, ..., e_k$ and the vertices in $V_H$ from cycles $C'_1, C'_2, ..., C'_k$ to be $e_{k+1}, e_{k+2}, ..., e_{2k}$. We give the gadgets that we will use for breaking cycles that ensures the Hamiltonian property and relates the vertex cover of $\tilde{G}$ to the vertex cover of new graph constructed.

**Gadgets used for reduction.** We use a gadget $W_i$ as shown in Figure 3.3(b) (indexed by $i$). The two paths from $e'_i$ to $e''_i$ in Figure 3.4(a), 3.4(b), each cover all the vertices of $W_i$ and together cover all the edges of $W_i$.

We create graph $G_F$ as follows. Initially we have $G_F = \tilde{G}$. We choose a vertex $e_i \in V_H$. As $\tilde{G}$ is a 4-regular graph therefore $e_i$ has four neighbors say $e_{i_1}, e_{i_2}, e_{i_3}$ and $e_{i_4}$ in $V(\tilde{G})$. Let $e_{i_1}, e_{i_2}$ be neighbor of $e_i$ in cycle $C_i \in \mathfrak{C}_H$ and $e_{i_3}, e_{i_4}$ be neighbors of $e_i$ in cycle $C'_j \in \mathfrak{C}_{H'}$. We delete $e_i$ from the graph $G_F$ and insert $W_i$ and add edges $(e_{i_1}, c_i), (e_{i_3}, c_i), (e_{i_2}, d_i)$ and $(e_{i_4}, d_i)$ (refer Figure 3.3). Note that when $e_i$ is a vertex in the cycle $C_i$ then $1 \leq i \leq k$. The proof for the construction for the case when $k+1 \leq i \leq 2k$ is analogous to the case when $1 \leq i \leq k$. Also note that for $1 \leq i \leq k$, $e_i$ is present in the cycle $C'_0 \in \mathfrak{C}_{H'}$, therefore we have $j = 0$.




(a) (b)

Figure 3.4: (a),(b) The two paths from $e'_i$ to $e''_i$ each covering all the vertices of $W_i$ and together covering all the edges of $W_i$.

In $\tilde{G}$ if at least one of $e_{i_1}, e_{i_2}, e_{i_3}, e_{i_4}$ is not chosen in vertex cover, we have to include $e_i$ in the vertex cover. So by $W_i$ we ensure that if at least one of $e_{i_1}, e_{i_2}, e_{i_3}, e_{i_4}$ is not chosen in the vertex cover of $G_F$, then at least one of $c_i$ or $d_i$ has to be included in vertex cover of $G_F$. Lemma 7 says that if at least one of $c_i$ or $d_i$ is included in the vertex cover then the size of the vertex cover of $G_F$ increases exactly by one. This indicates that in the vertex cover for $\tilde{G}$ we need to include vertex $e_i$.

**Lemma 7.** *The minimum vertex cover of $W_i$ has size 9 and does not contain $c_i$ or $d_i$. Any vertex cover of $W_i$ containing $c_i$ or $d_i$ has to be of size at least 10.*

*Proof.* The vertex sets $\{e'_i, x_i, x'_i\}$, $\{e''_i, y_i, y'_i\}$, $\{w_{i_1}, w_{i_2}, w_{i_3}\}$, $\{w_{i_4}, w_{i_5}, w_{i_6}\}$ forms $K_3$'s and $(w_i, a_i)$ is an edge, therefore we require minimum of 9 vertices to cover edges of $W_i$. If we have $V' = \{x_i, x'_i, y_i, y'_i, w_{i_1}, w_{i_2}, w_{i_4}, w_{i_6}, w_i\}$ then we can cover all edges of $W_i$ with 9 vertices.

If at least one of $c_i$ or $d_i$ is forced in the vertex cover, say $c_i$ is forced then apart from the $K_3$'s present we have an edge $(w_i, d_i)$, so including $c_i$ we need at least 10 vertices. Similar argument holds if $b_i$ is forced in the vertex cover. If we have $V' = \{x_i, x'_i, y_i, y'_i, w_{i_1}, w_{i_2}, w_{i_4}, w_{i_6}, c_i, d_i\}$ then we can cover all edges of $W_i$ with 10 vertices. □

In Figure 3.5(b) we show the gadget $\tilde{W}_i$ used for reduction and in Figure 3.6 we show

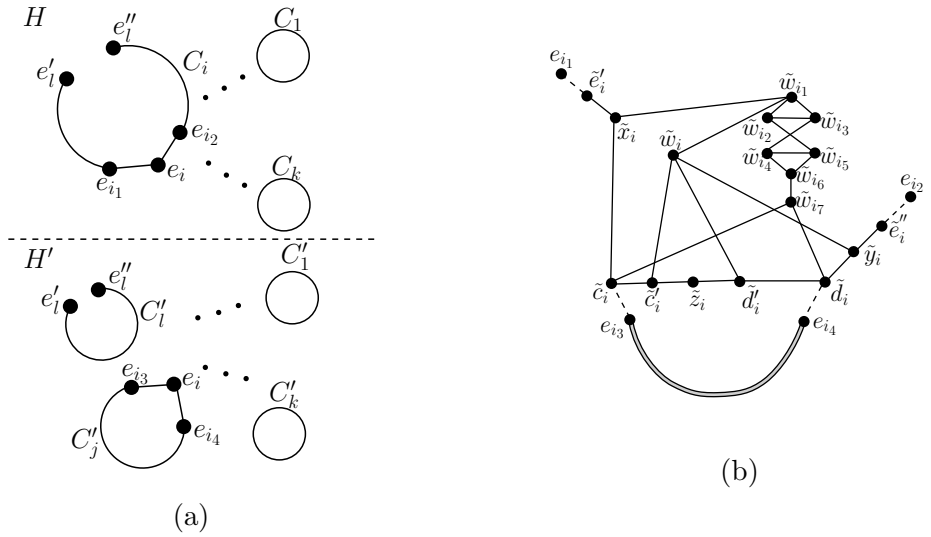


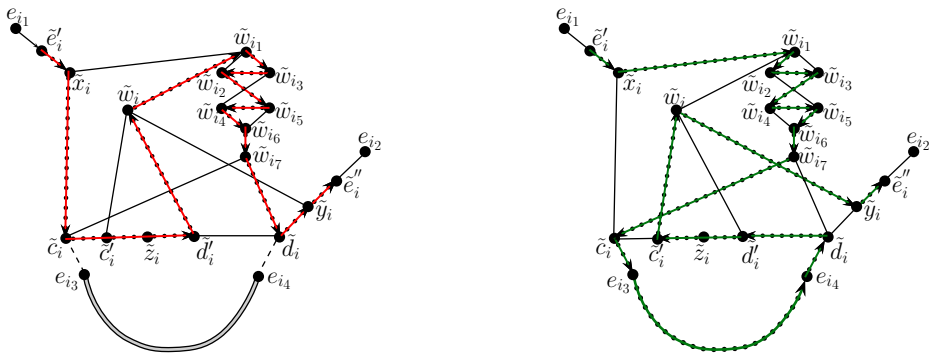

Figure 3.5: (a) Cycle $C_i$ already broken and deletion of vertex $e_i$ needed to break $C'_j$. (b) Gadget $\tilde{W}_i$, with dotted lines showing its connection to neighbors of $e_i$.

that there are no extra edges other than the two paths covering all vertices of $\tilde{W}_i$ from $\tilde{e}'_i$ and $\tilde{e}''_i$. The need of this gadget is when we have a cycle in which more than one vertex is deleted (in our case the cycles $C_0$ and $C'_0$), and we do not have a path for at least one neighbor of deleted vertex to attach to gadget $W_i$ (refer Figure 3.5(a)).

Let $e_i \in V_H$ be a vertex with neighbors $e_{i_1}, e_{i_2}, e_{i_3}$ and $e_{i_4}$. The vertex $e_i$ is deleted from $G_F$ and gadget $\tilde{W}_i$ is inserted as shown in Figure 3.7 to get a graph $G'_F$. If in the vertex cover of $G_F$ one of $e_{i_1}, e_{i_2}, e_{i_3}$ or $e_{i_4}$ is not included in the vertex cover then we need to include $e_i$ in the vertex cover of $G_F$. By $\tilde{W}_i$ we ensure that if at least one of $e_{i_1}, e_{i_2}, e_{i_3}$ or $e_{i_4}$ is not included in the vertex cover of $G'_F$, then one of $\tilde{e}'_i, \tilde{e}''_i, \tilde{c}_i$ or $\tilde{d}_i$ has to be included in the vertex cover of $G'_F$. In Lemma 8 we prove that the minimum number of vertices required to cover edges of $\tilde{W}_i$ is 9 and if at least one $\tilde{e}'_i, \tilde{e}''_i, \tilde{c}_i$ or $\tilde{d}_i$ is included in the vertex cover then we require at least 10 vertices to cover all the edges of $\tilde{W}_i$.

**Lemma 8.** *The minimum vertex cover of $\tilde{W}_i$ has 9 vertices and does not contain any of $\tilde{e}'_i, \tilde{e}''_i, \tilde{c}_i$ or $\tilde{d}_i$. Any vertex cover of $\tilde{W}_i$ containing at least one of $\tilde{e}'_i, \tilde{e}''_i, \tilde{c}_i$ or $\tilde{d}_i$ has to be of size at least 10.*

*Proof.* The vertex sets $\{\tilde{w}_{i_1}, \tilde{w}_{i_2}, \tilde{w}_{i_3}\}$ and $\{\tilde{w}_{i_4}, \tilde{w}_{i_5}, \tilde{w}_{i_6}\}$ forms $K_3$'s, so we require at




(a) (b)

Figure 3.6: (a),(b) The two paths from $e'_i$ to $e''_i$ each covering all the vertices of $\tilde{W}_i$ and together covering all the edges of $\tilde{W}_i$.

least 4 vertices among them. Note that $\{(\tilde{e}'_i,\ \tilde{x}_i),(\tilde{e}''_i\ ,\tilde{y}_i),(\tilde{z}_i,\ \tilde{d}'_i),(\tilde{d}_i,\tilde{w_{i_7}}),(\tilde{c}_i,\tilde{c}'_i)\}$ are vertex disjoint edges, so we require at least 9 vertices to cover all edges of $\tilde{W}_i$. If $V' = \{\tilde{w}_{i_1},\tilde{w}_{i_2},\tilde{w}_{i_4},\tilde{w}_{i_6},\ \tilde{w}_{i_7},\tilde{x}_i,\tilde{y}_i,\tilde{c}'_i,\tilde{d}'_i\}$ then $V'$ can cover all edges of $\tilde{W}_i$ with 9 vertices.

Following are the cases when at least one of $\tilde{e}'_i,\tilde{e}''_i,\tilde{c}_i$ or $\tilde{d}_i$ is forced in the vertex cover.

Case 1 If $\tilde{e}'_i$ is forced in the vertex cover then we have $\{(\tilde{x}_i,\tilde{c}_i),(\tilde{c}'_i,\tilde{z}_i),\ (\tilde{w}_i,\tilde{d}'_i),\ (\tilde{d}_i,\tilde{w}_{i_7}),\ (\tilde{y}_i,\tilde{e}''_i)\}$ as vertex disjoint edges left to be covered.

Case 2 If $\tilde{e}''_i$ is forced in the vertex cover then we have $\{(\tilde{x}_i,\tilde{e}'_i),(\tilde{c}'_i,\tilde{z}_i),(\tilde{w}_i,\tilde{d}'_i),\ (\tilde{c}_i,\tilde{w}_{i_7}),\ (\tilde{y}_i,\tilde{d}_i)\}$ as vertex disjoint edges left to be covered.

Case 3 If $\tilde{c}_i$ is forced in the vertex cover then we have $\{(\tilde{x}_i,\tilde{e}'_i),(\tilde{c}'_i\tilde{w}_i),(\tilde{z}_i,\tilde{d}'_i),\ (\tilde{d}_i,\tilde{w}_{i_7}),\ (\tilde{y}_i,\tilde{e}''_i)\}$ as vertex disjoint edges left to be covered.

Case 4 Similarly, for $\tilde{d}_i$ forced in the vertex cover then we have $\{(\tilde{x}_i,\tilde{e}'_i),(\tilde{c}'_i,\tilde{w}_i),(\tilde{z}_i,\tilde{d}'_i),\ (\tilde{c}_i,\tilde{w}_{i_7}),\ (\tilde{y}_i,\tilde{e}''_i)\}$ as vertex disjoint edges left to be covered.

In each of the above case we require at least 6 more vertices. Moreover, $V' = \{\tilde{e}'_i,\tilde{e}''_i,\tilde{c}_i,\tilde{d}_i,\ \tilde{w}_i,\tilde{z}_i,\ \tilde{w}_{i_1},\tilde{w}_{i_2},\tilde{w}_{i_4},\tilde{w}_{i_6}\}$, can cover all edges of $\tilde{W}_i$ with 10 vertices when at least one of $\tilde{e}'_i,\tilde{e}''_i,\tilde{c}_i,\tilde{d}_i$ is in the vertex cover. □

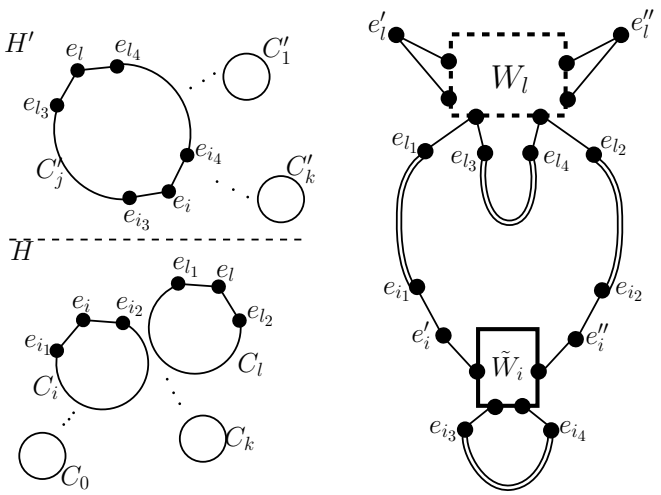



Figure 3.7: Schematic of connecting gadget $W_l$ and $\tilde{W}_i$ in one cycle when a cycle has to be broken at more than one point.

**The overall connection.** The vertex $e_k \in V_H$ is present in cycle $C'_0$ and the vertex $e_{2k} \in V_H$ is present in cycle $C_0$. We delete vertex $e_k$ from $\tilde{G}$ and insert the gadget $W_k$. Similarly we delete vertex $e_{2k}$ and insert gadget $W_{2k}$ appropriately as described in the construction. Note that the path of the cycles $C_0, C'_0$ are used for the paths of $W_k$ and $W_{2k}$. We also note that all the vertices $e_i \in V_H$, for $1 \leq i \leq k-1$ are present in the broken cycle $C'_0$. Similarly all the vertices $e_i \in V_H$, for $k+1 \leq i \leq 2k-1$ are present in the broken cycle $C_0$. So we do not have the path available to insert the gadget $W_i$. So for all other vertices in $V_H$ except $e_k, e_{2k}$, we insert the gadget $\tilde{W}_i$, for the corresponding $i$. Refer Figure 3.7 for the illustration of inserting gadget $W_l$ and $\tilde{W}_i$ in the same cycle. For our case $l$ is either $k$ or $2k$ and $1 \leq i < k$ or $k+1 \leq i < 2k$.

The final gadget used in the reduction is denoted by $W_{C_i}$, and is shown in Figure 3.8. We use this gadget for connecting broken cycles. It is easy to see that the gadget $W_{C_i}$ itself is an union of two Hamiltonian paths, and that we need four vertices to cover all the edges. If we include $t_{i_1}, t_{i_2}, t_{i_3}$ and $t_{i_5}$ in the vertex cover then we can cover all the edges adjacent to at least one vertex of $W_{C_i}$ (these are the vertices to which we connect other vertices in the graph to connect the broken cycles).

For cycle $C_i \in \mathfrak{C}_H$ let the vertex from which the specified path (the path that is

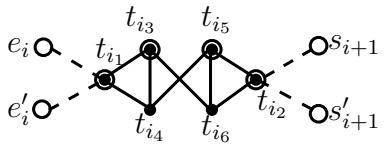



Figure 3.8: Gadget $W_{C_i}$, empty circle are the outside vertices to which $W_{C_i}$ is connected.

Hamiltonian with respect to $C_i$ and inserted gadget to break cycle) starts be $s_i$ and where it ends be $e_i$, similarly for cycles $C'_i \in \mathfrak{C}_{H'}$ let the start vertex be $s'_i$ and end vertex be $e'_i$. For a cycle $C_i, C_{i+1} \in \mathfrak{C}_H$ and $C'_i, C'_{i+1} \in \mathfrak{C}_{H'}$ we insert the gadget $W_{C_i}$ and add edges $(e_i, t_{i_1}), (s_{i+1}, t_{i_2}), (e'_i, t_{i_1}), (s'_{i+1}, t_{i_2})$, for $0 \leq i < k$, where $k$ is the number of cycles in $H$. It is easy to see that after making these additions, the set of edges of the graph is exactly a union of two Hamiltonian paths. One of the Hamiltonian paths starts from $s_0$ and ends at $e_k$ and second Hamiltonian path starts at $s'_0$ and ending at $e'_k$ (refer Figure 3.9). Note that in the Figure 3.9 the two Hamiltonian paths obtained from $H$ and $H'$ respectively are defined on the same set of vertices, but for the sake of clarity broken cycles of $H$ and $H'$ are shown separately. We call the final graph as $\tilde{G}_F$ after the above modification to $\tilde{G}$.

Putting together all the constructions described above and using the translations of the vertex cover at every stage, we have the following result.

**Lemma 9.** *The graph $\tilde{G}$ admits a vertex cover of size $p$ if and only if $\tilde{G}_F$ has a vertex cover of size at most $p + 22k$, where $k + 1$ is the number of cycles in $H$.*

*Proof.* In the forward direction, consider a $p$-sized vertex cover $S$ of $\tilde{G}$. If for some $v_i \in V_H$, $v_i$ is not included in $S$, then by Lemma 7 and Lemma 8 we know 9 vertices are sufficient to cover edges of $W_i$ or $\tilde{W}_i$ (which ever is inserted). Similarly if $v_i$ is included in $S$, we get a corresponding increase in the size of vertex cover for $\tilde{G}_F$ exactly by one from Lemma 7 and Lemma 8. Also for covering edges of $W_{C_i}$ and all other edges adjacent to vertices of $W_{C_i}$, 4 vertices are necessary and sufficient. Therefore, we have a $p + 9.2.k + 4.k = p + 22k$ sized vertex cover for $\tilde{G}_F$. ($|V_H| = 2k$).

In the reverse direction, for each of $W_{C_i}$ $0 \leq i < k$, we require at least 4 vertices. To cover the edges in each of $W_i$ and $\tilde{W}_i$ we require at least 9 vertices by Lemma 7 and

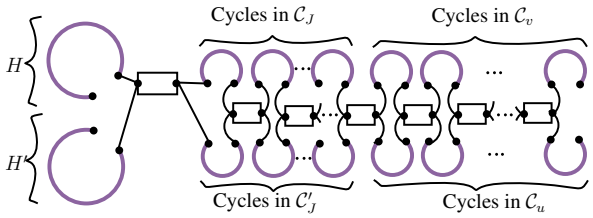



Figure 3.9: Overall Connection, showing outline of the two hamiltonian paths constructed from cycles in $H$ and $H'$ (rectangles representing copies gadget of $W_{C_i}$).

Lemma 8. Therefore to cover the edges of $\tilde{G}$ present in $\tilde{G}_F$ we are left with at most $p$ vertices. Consider the vertices $e_i$ that were deleted from $\tilde{G}$ to construct $\tilde{G}_F$. If any one of the neighbor of $e_i$ is not included in the vertex cover of $\tilde{G}_F$, then by Lemma 7 and Lemma 8 for the corresponding $W_i$ (or $\tilde{W}_i$) we need at least 10 vertices in the vertex cover of $\tilde{G}_F$, indicating that we need to include $e_i$ in the vertex cover of $\tilde{G}$.

□

**Corollary 2.** The graph $G_{uv} = G_u \cup G_v$ has a vertex cover of size at most $p$ if and only if $\tilde{G}_H$ has a vertex cover of size $p + 29k$.

**Theorem 10.** *The problem of finding a vertex cover of size at most $k$ in a braid graph is* NP-complete.

*Proof.* Follows from the Corollary 2. □

## 3.3 An improved branching algorithm

In this section we describe an improved FPT algorithm for the vertex cover problem on graphs with maximum degree at most four. The algorithm is essentially a search tree, and the analysis is based on the branch-and-bound technique. The input to the algorithm is denoted by a pair $(G, k)$, where $G$ is a graph, and the question is whether $G$ admits a vertex cover of size at most $k$.

We work with $k$, the size of the vertex cover sought, as the measure — sometimes referred to as the *budget*.

**Preprocessing.** We begin by eliminating simplicial vertices (refer paragraph on Preprocessing a simplicial vertex, Chapter 2). We delete $N[v]$ and reduce the budget to $k - |N(v)|$. Our algorithm makes extensive use of the *folding* technique, as described in past work [CKJ99, CKX06] (refer paragraph on Folding a degree two vertex, Chapter 2). Note that the new vertex generated by the folding operation can have more than four neighbors, especially if the vertices adjacent to the degree two vertex have, for example, degree four to begin with. We call a degree two vertex *foldable* if after folding, the folded vertex has degree at most four and *easily foldable* if after folding the folded vertex has degree at most three. In the preprocessing step we will fold only easily foldable vertices.

Typically, we ensure a reasonable drop on all branches by creating the following win-win situation: if a vertex is foldable, then we fold it. If it is not, then there are sufficiently many vertices in the second neighborhood of the vertex, and this usually leads to a good branching vector. Also, during the course of the branching, we appeal to a couple of simple facts about the structure of a vertex cover, which we state below.

**Lemma 11.** *[CKJ99, First part of Lemma 3.2] Let $v$ be a vertex of degree three in a graph $G$. Then there is a minimum vertex cover of $G$ that contains either all three neighbors of $v$ or at most one neighbor of $v$.*

This follows from the fact that a vertex cover that contains $v$ (where $d(v) = 3$) and two of its neighbors can be easily transformed into one, of the same size, that omits $v$ and contains all of its neighbors.

**Lemma 12.** *If $x, a, y, b$ form a cycle of length four in $G$ (in that order), and the degree of $a$ and $b$ in $G$ is two, then there exists an optimal vertex cover that does not pick $a$ or $b$ and contains both $x$ and $y$.*

*Proof.* Let $S$ be an optimal vertex cover. Any vertex cover must pick at least two vertices among $x, y, a, b$. If $x$ and $y$ belong to $S$, then clearly $S$ does not contain $a$ and $b$ (otherwise $S \setminus \{a, b\}$ would continue to be a vertex cover, contradicting optimality). If $S$ does not contain $x$ (or $y$, or both), then $S$ must contain both $a$ and $b$. Note that

$(S \setminus \{a, b\}) \cup \{x, y\}$ is a vertex cover whose size is at most $|S|$, and is a vertex cover of the desired requirements. □

**Overall Algorithm.** To begin with, the branching algorithm tries to branch mainly on a vertex of degree three or two. If the input graph is four-regular, then we simply branch on an arbitrary vertex to create two instances, both of which have at least one vertex of degree at most three. We note that this is an off-branching step, in the future, the algorithm maintains the invariant that at each step, the smaller graph produced has at least one vertex whose degree is at most three. After this, we remove all the simplicial vertices and then fold all easily-foldable vertices.

The branching algorithm that we propose assumes that we will always find a vertex whose degree is bounded by three to branch on, therefore it is important to avoid the situation where the graph obtained after folding all available degree two vertices is completely devoid of vertices of degree bounded by three (which is conceivable if all degree three vertices are adjacent to degree two vertices that in turn get affected by the folding operation). Therefore, we apply the folding operation somewhat tactfully− we apply it only when the degree two vertex is *foldable*. We avert the danger of leading ourselves to a four-regular graph recursively by explicitly ensuring that vertices of degree at most three are created whenever a folded vertex has degree four.

Consider a degree two vertex that is not easily foldable. For any degree two vertex $v$ with neighbors $u$ and $w$, note that there exists an optimal vertex cover that either contains $v$ or does not contain $v$ and includes both its neighbors. Indeed, if an optimal vertex cover $S$ contains, say $v$ and $u$, then note that $(S \setminus \{v\}) \cup \{w\}$ is a vertex cover of the same size. So we branch on the vertex $v$:

1. When $v$ does not belong to the vertex cover, we pick $u, w$ in the vertex cover, leading to a drop of two in the measure.

2. When $v$ does belong to the vertex cover, we have that $N(u) \cup N(w)$ must belong to the vertex cover, and we know that $|N(u) \cup N(w) \setminus \{v\}| \geq 4$ (otherwise, $v$ would be easily-foldable), and this leads to a drop of at least five in the measure.

So we either preprocess degree two vertices in polynomial time, or branch on them with a branching vector of $(2, 5)$. At the leaves of this branching tree, if we have a sub-cubic graph, then we employ the algorithm of [Xia10]. Otherwise, we have at least one degree three vertex which is adjacent to at least one degree four vertex. We branch on these vertices next. The case analysis is based on the neighborhood of the vertex — broadly, we distinguish between when the neighborhood has at least one edge, and when it has no edges. The latter case is the most demanding in terms of a case analysis. For the rest of this section, we describe all the scenarios that arise in this context.

For this part of the algorithm, we can always assume that we are given a degree three vertex with a degree four neighbor. Let $v$ be a degree three vertex, and let $N(v) := \{u, w, x\}$, where we let $u$ denote a degree four vertex.

**Degree three vertices with edges in their neighborhood.** Note that $u, w, x$ does not form a triangle, otherwise $v$ would be a simplicial vertex and we would have handled it earlier. So, we deal with the case when $N(v)$ is not a triangle, but has at least one edge. If $(w, x)$ is an edge, then we branch on $u$:

1. When $u$ does belong to the vertex cover, we delete $u$ from the graph, and we are left with $v, w, x$ being a triangle where $v$ is a degree two vertex, and therefore we may pick $w, x$ in the vertex cover — together, this leads to a drop of three in the measure.

2. When $u$ does not belong to the vertex cover, we pick four of its neighbors in the vertex cover, leading to a drop of four in the measure.

On the other hand, if $(w, x)$ is not an edge, then there is an edge incident to $u$. Suppose the edge is $(u, w)$ (the case when the edge is $(u, x)$ is symmetric). In this case, we branch on $x$ exactly as above. The measure may drop by three when $x$ does not belong to the vertex cover, if $x$ happens to be a degree three vertex. Therefore, our worst-case branching vector in the situation when $N(v)$ is not a triangle, but has at least one edge is $(3, 3)$.

Before going further on the case analysis, we describe a branching strategy for some specific situations — these mostly involve two non-adjacent vertices that have more than two neighbors in common, with at least one of them of degree four. This will be useful in scenarios that arise later. We consider the case when a degree four vertex $p$ non-adjacent

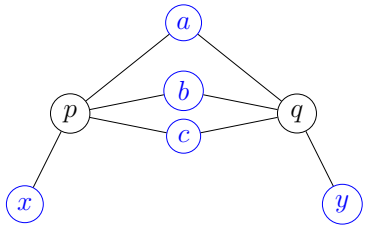



Figure 3.10: The case involving at least three common neighbors.

to a vertex $q$ has at least three neighbors in common, say $a, b, c$ and let $x$ be the other neighbors of $p$ that may or may not be adjacent to $q$. Notice that there always exists an optimal vertex cover that either contains both $p$ and $q$ or omits both $p$ and $q$. To see this, consider an optimal vertex cover $S$ that contains $p$ and omits $q$. Then, $S$ clearly contains $a, b, c$. Notice now that $T := (S \setminus \{p\}) \cup \{x\}$ is also a vertex cover, and $T$ contains neither $p$ or $q$, and has the same size as $S$. This suggests the following branching strategy:

1. If $p$ and $q$ both belong to the vertex cover, then the measure clearly drops by two. We proceed by deleting $p$ and $q$ from $G$. Now note that the degree of the vertices $\{a, b, c\}$ reduces by two and they become vertices of degree one or two (note that they cannot be isolated because we always begin by eliminating vertices of degree two by preprocessing or branching). If any one of these vertices is simplicial or foldable then we preprocess it or fold it respectively. Otherwise, we branch on $a$:

   (a) when $a$ does not belong to the vertex cover, we pick its neighbors in the vertex cover, leading to a drop of two in the measure.

   (b) when $a$ does belong to the vertex cover, we have that its second neighborhood must belong to the vertex cover, and this leads to a drop of six in the measure since $a$ is not foldable.

2. If $p$ and $q$ are both omitted from the vertex cover, then we pick $N(p) \cup N(q)$ in the vertex cover and the measure drops by at least four.

Depending on the situations the branching vector can be one of the following:

- $a$ is simplicial or foldable in $G \setminus \{p, q\}$. $(3, 4)$
- $a$ is not foldable in $G \setminus \{p, q\}$. $(4, 8, 4)$

We refer to the branching strategies outlined above as the **CommonNeighborBranch** strategy.

**Degree three vertices whose neighborhoods are independent.** Here we consider several cases. Broadly, we have two situations based on whether $u, w, x$ have any common neighbors or not (apart from $v$).

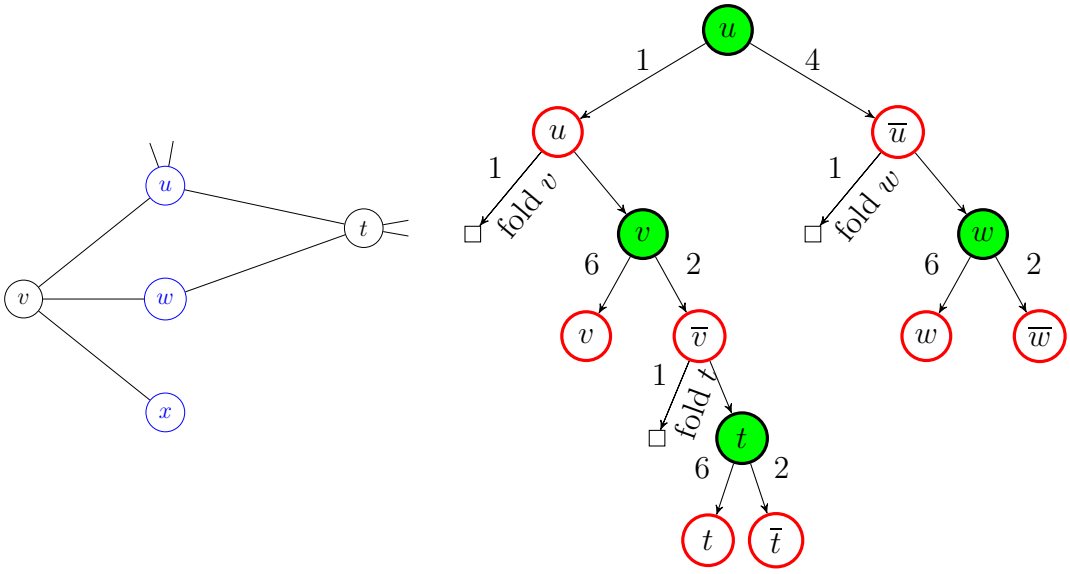



Figure 3.11: Scenario A, Case 1: The situation (left) and the suggested branching (right).

First, suppose there exists a vertex $t$ that is adjacent to at least two vertices in $N(v)$. Here, let us begin by considering the situation when $t$ is adjacent to $u$ and one other vertex. We will call this **Scenario A**.

In this scenario, we distinguish two cases based on the degree of $t$, and whether $t$ is adjacent to a degree four vertex or not. Here after for ease in specification we will refer to a degree one vertex also as a foldable vertex.

**Case 1: The vertex $t$ has degree four.** Here, we branch on $u$ as follows. We let $(t, w)$ to be an edge in the graph.

1. If $u$ belongs to the vertex cover, then we delete $u$ from $G$. Then, if $v$ is foldable in the resulting graph, then we fold $v$. Otherwise, we branch further on $v$:

   (a) When $v$ does belong to the vertex cover, we have that $N(w) \cup N(x)$ must belong to the vertex cover, and we know that $|N(w) \cup N(x) \setminus \{v\}| \geq 5$ (otherwise, $v$ would be foldable), and this leads to a drop of at least six in the measure.

   (b) When $v$ does not belong to the vertex cover, we pick $w, x$ in the vertex cover, leading to a drop of two in the measure. Here, we delete $v, w$ and $x$, after which $t$ becomes a degree two vertex. If $t$ is foldable in the resulting graph, then we fold $t$. Otherwise, we branch on $t$. Let $t', t''$ denote the two neighbors of $t$.

   *When $t$ does belong to the vertex cover*, we have that $N(t') \cup N(t'')$ must belong to the vertex cover, and this leads to a drop of at least six in the measure. *When $t$ does not belong to the vertex cover*, we pick $t', t''$ in the vertex cover, leading to a drop of two more in the measure.

2. If $u$ does not belong to the vertex cover, then we pick all of its neighbors in the vertex cover. Since the degree of $u$ is four, this leads the measure to drop by four. Also, after removing $N[u]$ from $G$, the vertex $w$ lose two neighbors (namely $v$ and $t$). If it is foldable, then we proceed by folding the said vertex. Otherwise, we branch further on $w$, letting $w', w''$ denote the neighbors of $w$.

   (a) When $w$ does belong to the vertex cover, we have that $N(w') \cup N(w'')$ must belong to the vertex cover and this leads to a drop of six in the measure.

(b) When $w$ does not belong to the vertex cover, we pick $w', w''$ in the vertex cover, leading to a drop of two in the measure,

Depending on the situations that arise, the branching vectors can be one of the following (we use $S$ to denote the vertex cover that will be output by the algorithm):

- $w$ is foldable in $G \setminus N[u]$, and $v$ is foldable in $G \setminus \{u\}$. $(2, 5)$
- $w$ is foldable in $G \setminus N[u]$, $v$ is not foldable in $G \setminus \{u\}$, and $t$ is foldable in $(G \setminus \{u\}) \setminus N[v]$. $(7, 4, 5)$
- $w$ is foldable in $G \setminus N[u]$, $v$ is not foldable in $G \setminus \{u\}$, and $t$ not foldable in $G \setminus \{u\} \setminus N[v]$. $(7, 9, 5, 5)$
- $w$ is not foldable in $G \setminus N[u]$, and $v$ is foldable in $G \setminus \{u\}$. $(2, 10, 6)$
- $w$ is not foldable in $G \setminus N[u]$, $v$ is not foldable in $G \setminus \{u\}$, and $t$ is foldable in $G \setminus \{u\} \setminus N[v]$. $(7, 4, 10, 6)$
- $w$ is not foldable in $G \setminus N[u]$, $v$ is not foldable in $G \setminus \{u\}$, and $t$ is not foldable in $G \setminus \{u\} \setminus N[v]$. $(7, 9, 5, 10, 6)$

The reason we needed to have $d(t) = 4$ in the case above was to ensure that we have a vertex that we can either fold or branch on in the graph $G \setminus N(v)$, which is the situation that arises when $v$ is not foldable, and $N(v)$ is included in the vertex cover. If $w$ and $x$ both have degree three, then $v$ is indeed foldable and the branching above gives the desired guarantee. Otherwise, if $t$ has degree three and in particular $(t, x)$ is an edge, then $t$ becomes isolated in this situation, and we have no clear way of further progress.

We now continue our case analysis. Recall that we would like to address the situation that $t$ is degree three and all of its neighbors are common with $v$, and further that both of $w$ and $x$ have degree four (otherwise $v$ would be foldable and we can apply Scenario A, Case 1).

**Case 2: The vertex $t$ has degree three, the vertices $w, x$ have degree four, and $(t, x) \in E$.** Here, we let $u'$ and $u''$ denote the neighbors of $u$. Our case analysis is now based on the degrees of these vertices.

*Case 2.I: At least one of $u'$ or $u''$ has degree three*: Suppose, without loss of generality, that $u'$ has degree three. Note that $x$ and $u$ are two non-adjacent vertices. The vertices $v$ and $t$ are already in their common neighborhood. If they have more common neighbors, then we branch according to the **CommonNeighborBranch** strategy. Otherwise, we branch on the vertex $w$ as follows.

1. If $w$ belongs to the vertex cover, then we delete $w$ from $G$. Here, the measure drops by one. In the remaining graph, branch on $u$:

   (a) When $u$ does belong to the vertex cover, then we also pick $x$ in the vertex cover (see Lemma 11). Further, we fold $u'$ if it is foldable, otherwise we branch on $u'$:

   *When $u'$ does belong to the vertex cover*, we have that its second neighborhood must belong to the vertex cover, and this leads to a further drop of six in the measure.

   *When $u'$ does not belong to the vertex cover*, we pick its neighbors in the vertex cover, leading to a drop of two in the measure.

   (b) When $u$ does not belong to the vertex cover, we pick the neighbors of $u$ in the vertex cover. Since $u$ is not in the vertex cover, and $w$ is in the vertex cover, we know by Lemma 11 that the neighbors of $x$ must be in the vertex cover. Note that $u$ and $x$ have no common neighbors other than $v$ and $t$, otherwise the **CommonNeighborBranch** strategy would apply. Therefore, we have that the measure drops by at least six more (the vertex $u$ has at least four neighbors and $x$ has at least two private neighbors).

2. If $w$ does not belong to the vertex cover, then we pick all of its neighbors in the vertex cover. This immediately leads the measure to drop by four. Also, after

removing $N[w]$ from $G$, the vertex $u$ loses two neighbors (namely $v$ and $t$). If $u$ is foldable we fold $u$, otherwise we branch on $u$:

(a) When $u$ does belong to the vertex cover, we have that $N(u') \cup N(u'')$ must be in the vertex cover and this leads to a drop of six in the measure.

(b) When $u$ does not belong to the vertex cover, we pick $u', u''$ in the vertex cover, leading to a drop of two in the measure.

Depending on the situations that arise, the branching vectors can be one of the following:

- $u$ is foldable in $G \setminus N[w]$, and $u'$ is foldable in $G \setminus \{u, w\}$. $(4, 7, 5)$
- $u$ is foldable in $G \setminus N[w]$, and $u'$ is not foldable in $G \setminus \{u, w\}$. $(9, 5, 7, 5)$
- $u$ is not foldable in $G \setminus N[w]$, and $u'$ is foldable in $G \setminus \{u, w\}$. $(4, 7, 10, 6)$
- $u$ is not foldable in $G \setminus N[w]$, and $u'$ is not foldable in $G \setminus \{u, w\}$. $(9, 5, 7, 10, 6)$

*Case 2.II: Both $u'$ or $u''$ have degree four*: Here, we branch on $u'$, as described below.

1. If $u'$ belongs to the vertex cover, then we delete $u'$ from $G$. Here, the measure drops by one. In the remaining graph, branch on $u$:

   (a) When $u$ does belong to the vertex cover, remove $u$ from $G$. In the remaining graph, the vertices $v$ and $t$ loses one neighbor each (namely $u$), and are now vertices of degree two. Note that $v, w, t, x$ now form a $C_4$, and since $v$ and $t$ have degree two, we may pick $w$ and $x$ in the vertex cover without loss of generality (see Lemma 12).

   (b) When $u$ does not belong to the vertex cover, we pick the neighbors of $u$ in the vertex cover. Also, after removing $N[u]$ from $G$, the vertices $w$ and $x$ lose two neighbors each (namely $v$ and $t$). If either of them are foldable, then we proceed by folding. Notice that none of them become isolated because degree two vertices are eliminated. Otherwise, we branch on $w$:

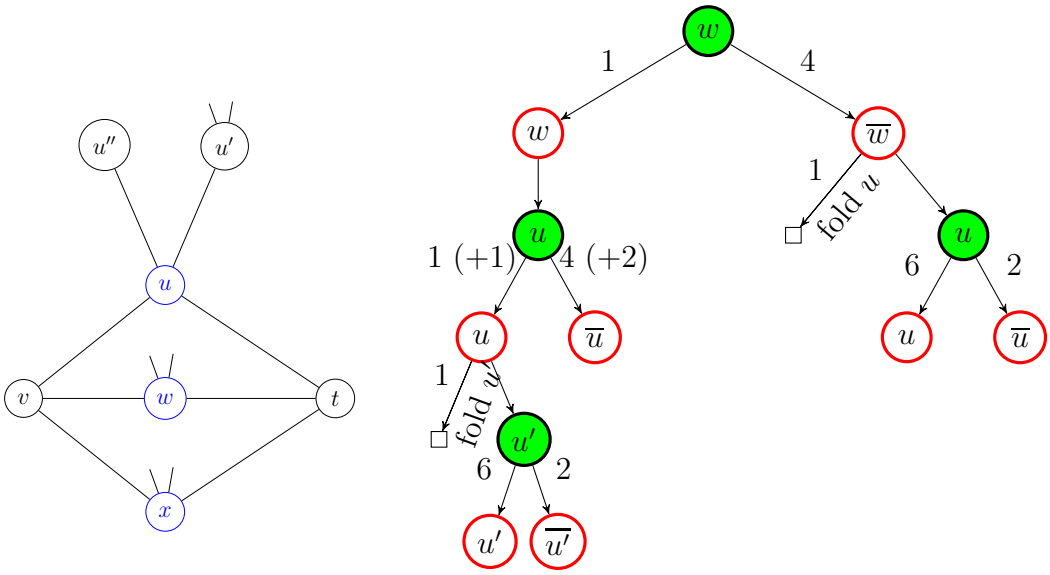



Figure 3.12: Scenario A, Case 2.I: The situation (left) and the suggested branching (right).

*When $w$ does belong to the vertex cover*, we have that its second neighborhood must belong to the vertex cover, and this leads to a drop of six in the measure.

*When $w$ does not belong to the vertex cover*, we pick its neighbors in the vertex cover, leading to a drop of two in the measure.

2. If $u'$ does not belong to the vertex cover, then we pick all of its neighbors in the vertex cover. This immediately leads the measure to drop by four. Also, after removing $N[u']$ from $G$, the vertices $v$ and $t$ loses one neighbor each (namely $u$), and are now vertices of degree two. Note that $v, w, t, x$ now form a $C_4$, and since $v$ and $t$ have degree two, we may pick $w$ and $x$ in the vertex cover without loss of generality (see Lemma 12).

If one of $w$ or $x$ is foldable in $G \setminus N[u]$, then we have a branching vector of $(4, 5, 6)$. Otherwise, we have a branching vector of $(4, 10, 6, 6)$.

This completes the description of **Scenario A**, where we assumed that $t$ was adjacent to $u$. Now, let us turn to the situation when $t$ is not adjacent to $u$. Since we are in the

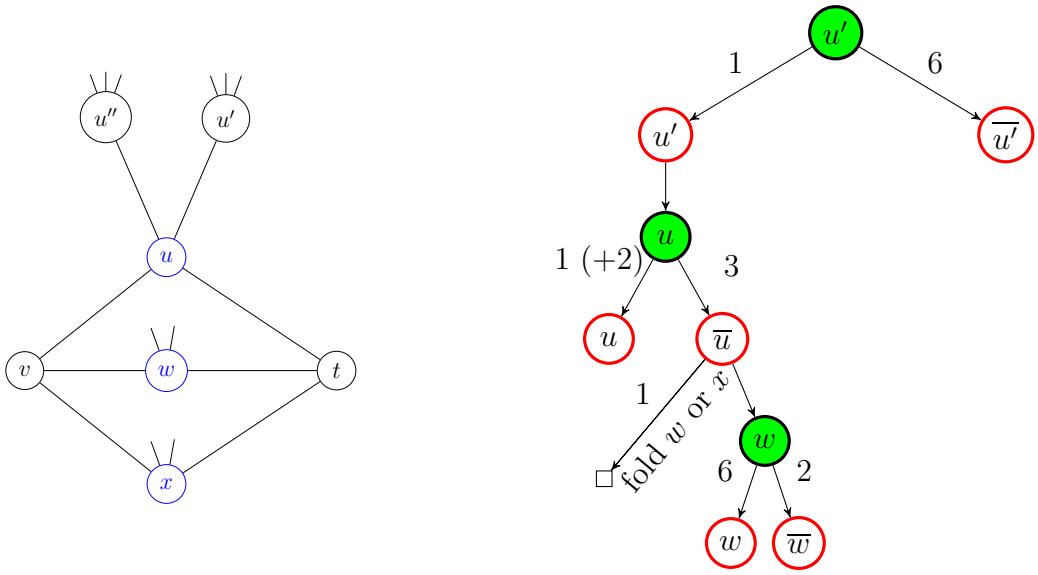



Figure 3.13: Scenario A, Case 2.II: The situation (left) and the suggested branching (right).

setting where $t$ is adjacent to two neighbors of $v$, this implies that $w$ and $x$ are both neighbors of $t$. In fact, we can even assume that both $w$ and $x$ are vertices of degree three, otherwise we would be in **Scenario A** by a simple renaming of vertices. We call this setup **Scenario B**.

Here, we simply branch on the vertex $u$, as follows:

1. If $u$ belongs to the vertex cover, then we delete $u$ from $G$. In the resulting graph, $v$ is evidently a foldable degree two vertex, so we fold $v$. Notice that the measure altogether drops by two in this branch.

2. If $u$ does not belong to the vertex cover, then we pick all its neighbors in the vertex cover. After removing $N[u]$, note that $w$ and $x$ have degree two. If either of them are foldable, then we proceed by folding. Otherwise, we branch on $w$:

   (a) When $w$ does belong to the vertex cover, we have that its second neighborhood must belong to the vertex cover, and this leads to a drop of six in the measure.

   (b) When $w$ does not belong to the vertex cover, we pick its neighbhors in the vertex cover, leading to a drop of two in the measure.

Note that if $w$ is foldable in $G \setminus N[u]$, then we have a branch vector of $(2, 5)$, otherwise, we have a branch vector of $(2, 6, 10)$. Now we have covered all the cases that arise when the neighbors of $v$ have a shared neighbor other than $v$, which we called $t$.

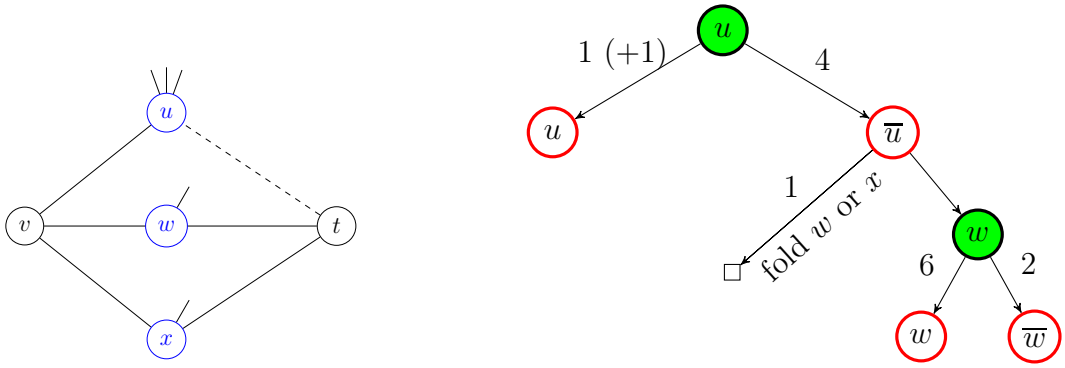



Figure 3.14: Scenario B: The situation (left) and the suggested branching (right).

The remaining case is when the vertices $u, w, x$ have no common neighbors other than $v$. We call this **Scenario C**. Here, we have cases depending on the degree of $w$ and $x$ — the first setting is when both $w, x$ are vertex of degree three. Second when both have degree four and third when one has degree three and other has degree four.

**When the degree of both $w$ and $x$ is three**: This branching is identical to the branching for **Scenario B**. Note that the important aspect there was the fact that $v$ is foldable in $G \setminus \{u\}$, which continues to be the case here. It is easy to check that all other elements are identical.

**When the degree of both $w$ and $x$ is four**: In this case, we branch on $w$.

1. If $w$ belongs to the vertex cover, then we delete $w$ from $G$. Here, the measure drops by one. In the remaining graph, branch on $v$:

   (a) When $v$ does belong to the vertex cover and we are in case when $w$ is in vertex cover, we know by Lemma 11 that the neighbors of $u$ and $x$ must be in the vertex cover. Note that $u, w$ and $x$ have no common neighbors other than $v$ (or we would be in **Scenario A** or **Scenario B**). But, both $u$ and $x$ are degree four vertices with no vertex common in their neighborhood other than

$v$, so we include the neighbors of $u$ and $x$ in the vertex cover and delete from graph $N[u] \cup N[x] \cup \{w\}$, with a total drop in the measure by 8.

(b) When $v$ does not belong to the vertex cover, we pick the neighbors of $v$ in the vertex cover and the measure drops by three.

2. If $w$ does not belong to the vertex cover, then we pick all of its neighbors in the vertex cover and we branch on $x$.

   (a) When $x$ does belong to the vertex cover, and we have that $w$ is not in the vertex cover, we know by Lemma 11 that neighbors of $u$ and $w$ must be in vertex cover. So we include neighbors of $u$ and $x$ in the vertex cover, to get a total drop of 8.

   (b) When $x$ does not belong to the vertex cover, we include all neighbors of $x$ in the vertex cover to get a total drop of 7.

Here we have the branch vector as $(8, 3, 8, 7)$.

**When the degree of $w$ is four and $x$ is three**: We let the two other neighbors of $x$ to be $x_1, x_2$. If $x_1$ is a degree four vertex and $x_2$ is a degree three vertex (or vice-versa) then we have a degree three vertex $x$ adjacent to two degree three vertices $v, x_2$ and a degree four vertex $x_1$ and we can apply the rules in **Scenario $C$: case** 1. So we are left with two cases one when both $x_1, x_2$ are degree three vertices and second when both $x_1, x_2$ are degree four vertices.

*Both $x_1, x_2$ are degree three vertices*: In this case we branch on $u$.

1. When $u$ does belong to the vertex cover then we branch on $v$.

   (a) When $v$ does belong to the vertex cover and we know $u$ is in vertex cover, we know by Lemma 11 that neighbors of $w, x$ are in vertex cover. So we include neighbors of $w, x$ in vertex cover and get a drop in measure by 7.

   (b) When $v$ does not belong to the vertex cover then we include neighbhors of $v$ in the vertex cover, to get a drop in the measure by 3.

2. When $u$ does not belong to the vertex cover. Then we include neighbors of $u$ in the vertex cover and delete $N[u]$ from the graph. Now $x$ is a degree two vertex and $|(N(x_1) \cup N(x_2)) \setminus \{x\}| \leq 4$, so we fold $x$ to get a drop of one more in the measure.

   Here we have the branch vector as $(7, 3, 5)$.

*Both $x_1, x_2$ are degree four vertices*: We branch on $x_1$.

1. When $x_1$ does belong to the vertex cover, we branch on $x$.

   (a) When $x$ does belong to the vertex cover, we know by Lemma 11 that neighbors of $v$ and $x_2$ are in vertex cover. So we include neighbors of $v$ and $x_2$ in the vertex cover and get a total drop of 7 in the measure.

   (b) When $x$ does not belong to the vertex cover, we get an immediate drop in measure by 3, now we branch on $u$.

   *When $u$ does belong to the vertex cover* and we are in the case when $x$ does not belong to the vertex cover, we know by Lemma 11 that neighbors of $w, x$ must be in vertex cover. So we include neighbors of $w, x$ in the vertex cover and get a total drop in the measure by 7

   *When $u$ does not belong to the vertex cover* then we include neighbors of $u$ in the vertex cover and get a total drop of 6 in the measure.

2. When $x_1$ does not belong to the vertex cover, we get an immediate drop of four in the measure. We delete $N[x_1]$ from the graph, after deletion $v$ is a degree two vertex. If $v$ is foldable we fold $v$ and get a drop of 1, otherwise we branch on $v$.

   (a) When $v$ does belong to the vertex cover then we include the neighborhood of $u$ and $w$ in the vertex cover and get a total drop in the measure of at least 10.

   (b) When $v$ does not belong in the vertex cover then we include neighbors of $v$ in the vertex cover and get a total drop of 6.

   If $v$ is foldable in $G \setminus N[x_1]$ we have the branch vector $(7, 7, 6, 5)$, otherwise the branch vector is $(7, 7, 6, 10, 6)$.

| Scenario | Cases | Branch Vector | $c$ |
|---|---|---|---|
| Degree Two (not easily foldable) | | $(2,5)$ | 1.2365 |
| Degree Three Edge in $N(v)$ | | $(3,3)$ | 1.2599 |
| CommonNeighborBranch | | $(3,4)$ | 1.2207 |
| | | $(4,8,4)$ | 1.2465 |
| Scenario A | Case 1 | $(2,5)$ | 1.2365 |
| | | $(7,4,5)$ | 1.2365 |
| | | $(7,9,5,5)$ | 1.2498 |
| | | $(2,10,6)$ | 1.2530 |
| | | $(7,4,10,6)$ | 1.2475 |
| | | $(7,9,5,10,6)$ | 1.2575 |
| | Case 2 (I) | $(4,7,5)$ | 1.2365 |
| | | $(9,5,7,5)$ | 1.2498 |
| | | $(4,7,10,6)$ | 1.2475 |
| | | $(9,5,7,10,6)$ | 1.2575 |
| | Case 2 (II) | $(4,5,6)$ | 1.2498 |
| | | $(4,10,6,6)$ | 1.2590 |
| Scenario B | | $(2,5)$ | 1.2365 |
| | | $(2,6,10)$ | 1.2530 |
| Scenario C | Case 1 | $(2,6,10)$ | 1.2530 |
| | Case 2 | $(8,3,8,7)$ | 1.2631 |
| | Case 3 | $(7,3,5)$ | 1.2637 |
| | | $(7,7,6,5)$ | 1.2519 |
| | | $(7,7,6,10,6)$ | 1.2592 |

Table 3.1: The branch vectors and the corresponding running times across various scenarios and cases for algorithm of maximum degree four graphs.

Note that the correctness of the algorithm is implicit in the description, and follows from the fact that the cases are exhaustive and so is the branching. The branch vectors are summarized in Table 3.1. We have consequently, the following theorem.

**Theorem 13.** *The* Vertex Cover *problem on graphs that have maximum degree at most four can be solved in* $\mathcal{O}^\star(1.2637^k)$ *worst-case running time.*

## 3.4 Conclusions

In this chapter we showed that the problem of hitting all axis-parallel slabs induced by a point set $P$ is equivalent to the problem of finding a vertex cover on a braid graph. We established that this problem is NP-complete. Finally, we also gave an algorithm for vertex cover on graphs of maximum degree four whose running time is $\mathcal{O}^\star(1.2637^k)$. It would be interesting to know if we can exploit the structure of braid graphs to design an algorithm with better running time for computing an optimal vertex cover on them.

# Chapter 4

# Vertex cover on triangulations

In this chapter, we focus on the computational problem of finding optimal vertex covers on triangulations, specifically chordless-NST. Uehara [Ueh96] showed that the $k$-vertex cover problem is NP-complete on 3-connected, triangle free planar graphs. We show that for chordless-NST (refer Definition 10), deciding the vertex cover problem is NP-complete by giving a reduction from an instance of vertex cover problem on 3-connected triangle free planar graph. Dillencourt and Smith [DS96] showed that *chordless-NST*s are Delaunay realizable. Note this implies that the vertex cover problem on graphs realizable as Delaunay triangulation and vertex cover on triangulations is NP-complete. We extend this to show that problem of vertex cover on maximal-planar graphs (refer Definition 9) is NP-complete.

## 4.1 NP-completeness of vertex cover on chordless-NST

In this section, we show that the problem of deciding $k$-vertex cover on chordless-NSTs is NP-complete. We reduce an instance of $k$-vertex cover on *3-connected triangle-free planar* graphs to an instance of $k'$-vertex cover on a chordless-NST.

Let $G$ be a *3-connected triangle-free planar graph* with $|V(G)| = n$ and $|E(G)| = m$. We will construct a chordless-NST graph $G'$. Initially we have $G' = G$. We consider a

fixed straight line planar embedding of $G'$. Let $f_0, f_1, ..., f_{k-1}$ be the faces of $G'$, with $f_{k-1}$ being the outer face. For each face $f_i$, we denote the vertices on the face $f_i$ by $v_{i_0}, v_{i_1}, ..., v_{i_{n_i-1}}$ (forming cycle in that order), where $n_i$ is the number of vertices on the face $f_i$, for $0 \leq i < k$.

Our goal with the construction of $G'$ is to get a triangulation of each inner face of $G$. Informally, we add inside every face $f$ of length $\ell$, a cycle $C_f$ of length $\ell$, and we then place a very large wheel inside this cycle. We triangulate these two layers by distributing edges uniformly. The size of the wheel and the budget of the reduced instance is chosen so that any vertex cover of $G'$ cannot afford to leave out any vertex from $C_f$ or the center of the wheel — this would require the vertex cover to pick too many vertices from the wheel. In particular, the trade-off is such that we can afford to pick half the vertices from the wheel (which would be required), but picking the neighborhood of a vertex $v$ from $C_f$ on the wheel, overshoots the budget. Once the structure of the vertex cover on the gadgets is forced, the original edges can be easily covered by the original vertex cover. We now turn to a formal description.

For face $f_i$ of $G'$, where $0 \leq i < k-1$ we insert a cycle $C_i = (u_{i_0}, u_{i_1}, ..., u_{i_{n_i-1}})$ (in that order) inside $f_i$ and add edges $(v_{i_j}, u_{i_j}), (v_{i_j}, u_{i_{j+1}})$ to $E(G')$, for $0 \leq j < n_i$ (index modulo $n_i$). Inside cycle $C_i$ we insert a wheel $W_i$ on $2c.n_i + 1$ vertices, where $c = 4n$ and the cycle of the wheel being $w_{i_0}, w_{i_1}, ..., w_{i_{2c.n_i-1}}$ and $x_i$ being the center of the wheel. For $u_{i_j}$, $0 \leq j < n_i$ we add $2c+1$ edges, $(u_{i_j}, w_{i_{2c.j+l}})$, $0 \leq l \leq 2c$ (index modulo $2c.n_i$) (refer Figure 4.1). Note the figure gives a qualitative illustration of triangulating a face $f_i$. The actual number of vertices in the cycle or wheel as shown in the figure does not correspond to the number of vertices as given by the construction.

**Lemma 14.** *The graph $G$ admits a vertex cover of size $p$ if and only if $G'$ has a vertex cover of size $p + (k-1) + (c+1)(2m - n_{k-1})$, where $c = 4n$, $k$ is the number of faces in $G$ and $n_{k-1}$ is the number of vertices on the outer face.*

*Proof.* Without loss of generality we may assume $p < n$. In the forward direction consider a vertex cover $S$ of $G$, where $|S| = p$. The only edges in $G'$ that might not be covered by $S$ are those adjacent to vertices of cycle $C_i$ or wheel $W_i$, for $0 \leq i < k-1$. If we add

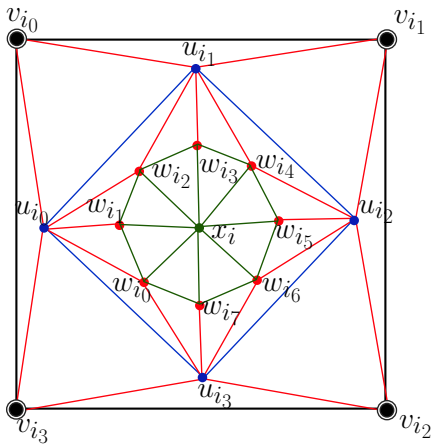



Figure 4.1: Face $f_i$, triangulated by inserting vertices $u_i$ forming a cycle and vertices $w_{i_j}$ forming a wheel.

to $S$, all vertices of $C_i$, $0 \leq i < k-1$ and for each wheel $W_i$, $0 \leq i < k-1$, include alternate vertices and the center of the wheel $x_i$, it is easy to see that we can cover all edges of $G'$, and the number of extra vertices $x$ that we have added to $S$ is given by the expression below:

$$= \sum_{i=0}^{k-2} (n_i + \left\lceil \frac{2c.n_i}{2} \right\rceil + 1)$$

$$= (k-1) + \sum_{i=0}^{k-2} (c+1).n_i$$

$$= (k-1) + (c+1)(2m - n_{k-1})$$

So we have a vertex cover of $G'$ with size at most $p + (k-1) + (c+1)(2m - n_{k-1})$.

In the reverse direction, let $S$ be a vertex cover of size $p+(k-1)+(c+1)(2m-n_{k-1})$ for $G'$. We argue that all the vertices of cycle $C_i$, $0 \leq i < k-1$ must be present in $S$. Suppose there exists a $C_j$ corresponding to the face $f_j$ of $G$, such that we have not included a vertex $u_j \in V(C_j)$ in the vertex cover. But then we need to include all the vertices that are neighbors of $u_j$ in the vertex cover and by construction we have $2c+1$ neighbors in the corresponding wheel $W_j$. Other vertices that are left in wheel $W_j$ forms a cycle (with some extra edges), so we need at least $\lceil (2c.n_j + 1 - (2c+1))/2 \rceil$ more vertices. From the cycle $C_j$ we need at least $\lceil n_j/2 \rceil$ vertices and from remaining cycles $C_i$ and

wheel $W_i$, we need atleast $1 + 2c.n_i/2 + \lceil n_i/2 \rceil$ more vertices for $0 \leq i < k-1, i \neq j$. So the total number of vertices $y$ needed is given by the expression below:

$$y = (2c+1) + \left\lceil \frac{2c.n_j + 1 - (2c+1)}{2} \right\rceil + \left\lceil \frac{n_j}{2} \right\rceil + \sum_{i=0, i\neq j}^{k-2} (1 + \frac{2c.n_i}{2} + \left\lceil \frac{n_i}{2} \right\rceil)$$

$$= (2c+1) + c(n_j - 1) + \left\lceil \frac{n_j}{2} \right\rceil + \sum_{i=0, i\neq j}^{k-2} (1 + c.n_i + \left\lceil \frac{n_i}{2} \right\rceil)$$

$$= c + \sum_{i=0}^{k-2} (1 + c.n_i + \left\lceil \frac{n_i}{2} \right\rceil)$$

$$= c + (k-1) + \sum_{i=0}^{k-2} (c.n_i + \left\lceil \frac{n_i}{2} \right\rceil)$$

$$\geq c + (k-1) + \frac{2c+1}{2} \sum_{i=0}^{k-2} n_i$$

$$= c + (k-1) + \frac{2c+1}{2} (2m - n_{k-1})$$

$$= c + (k-1) + (c+1)(2m - n_{k-1}) - \frac{1}{2}.(2m - n_{k-1})$$

Since $G$ is a planar graph, so number of edges $m$ can be at most $3n - 6$ and each face has at least four vertices, therefore we have

$$y \geq 4n - \frac{1}{2}.(2(3n-6) - 4) + (k-1) + (c+1)(2m - n_{k-1})$$

$$> n + (k-1) + (c+1)(2m - n_{k-1})$$

which is a contradiction to the assumption that $G'$ has a vertex cover of size at most $p + (k-1) + (c+1)(2m - n_{k-1})$ since $p < n$. Therefore, we cannot leave any vertex from cycle $C_i$, $0 \leq i < k-1$. Total number of vertices included by including all vertices of $C_i$, $0 \leq i < k-1$ is $x = \sum_{i=0}^{k-2} n_i = (2m - n_k)$. To cover all the wheels $W_i$, $0 \leq i < k-1$ (Note we have covered all edges with one vertex in wheel and other vertex not in the wheel) we require at least $c.n_i + 1$ vertices from $W_i$. So the total number of vertices required to

cover all edges of the wheels is at least $y = \sum_{i=0}^{k-2}(c.n_i + 1) = c(2m - n_k) + (k - 1)$. So for covering all other edges (edges of $G$) we are left with at most $p$ vertices. □

**Theorem 15.** *Deciding k-vertex cover on chordless-NST is* NP-complete.

*Proof.* Follows from Lemma 14. □

**Lemma 16.** *Graph $G'$ is realizable as a Delaunay Triangulation.*

*Proof.* $G'$ does not have a chord since $G$ was 3-connected and the construction of $G'$ did not alter the outer face. Also since $G$ is a triangle free graph, $G'$ does not have any separating triangle (non-facial triangles) induced by vertices of $G$. Note that all the extra cycles (non-facial) and wheels that we insert in $G$ to create $G'$ are of length greater than three. Hence $G'$ does not have any non-facial triangle.

It is shown in [DS96] that all chordless-NSTs are realizable as combinatorially equivalent Delaunay triangulations. It follows that $G'$ is combinatorially realizable as Delaunay triangulation. □

**Theorem 17.** *Deciding k-vertex cover on graphs realizable as Delaunay triangulations is* NP-complete.

*Proof.* Follows from Theorem 15 and Lemma 16. □

**Corollary 3.** Deciding $k$-vertex cover on triangulated graph is NP-complete.

## 4.2 NP-completeness of vertex cover on maximal planar graphs

In this section we show that the vertex cover problem on maximal planar graphs is NP-complete. We give a reduction from an instance of vertex cover on triangulated graph (which is known to be NP-complete from Section 4.1) to an instance of vertex cover on a maximal planar graph. Given an instance of $k$-vertex cover on a triangulated graph $G$ (outer face not necessarily a triangle) on $n$ vertices, we construct a maximal planar

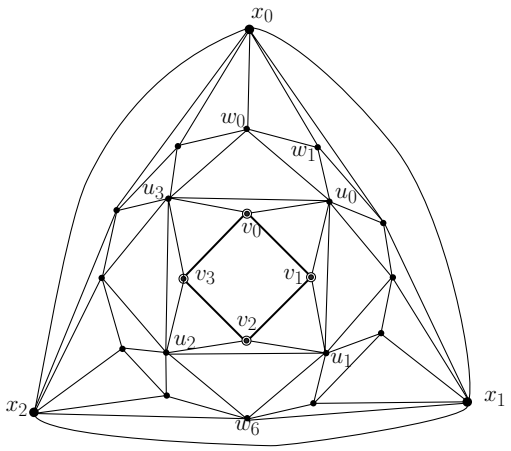



Figure 4.2: Outer face of graph $G'$ is a triangle $C_3 = (x_0, x_1, x_2)$. The region between $C_3$ and the outer face of $G$ is triangulated by adding vertices $u_i$ forming a cycle $C$ and vertices $w_i$ forming a cycle $C'$.

graph $G'$ corresponding to $G$ maintaining appropriate relationship between the sizes of the vertex cover. We fix a straight line embedding of $G$ in plane (non-crossing edges) with the non-triangular face (if any) as the outer face. We let the outer face in this fixed embedding of $G$ to be $f$, with the vertices forming the outer face as $v_0, v_1, ...v_{n'-1}$ (forming a cycle in that order).

Initially we have $G' = G$. We insert a cycle $C = (u_0, u_1, ..., u_{n'-1})$ (in that order) outside $f$ and add edges $(u_i, v_i), (u_i, v_{i+1})$ to $E(G')$, for $0 \leq j < n'$ (index modulo $n'$). Outside cycle $C$ we insert another cycle $C' = (w_0, w_1, ..., w_{cn'-1})$ (in that order), where $c = 6n$. For each vertex $u_i$, $0 \leq i < n'$ we add $c + 1$ edges $(u_i, w_{ci+j})$, $0 \leq j \leq c$, (index modulo $cn'$), . Outside $C'$ we add cycle $C_3 = (x_0, x_1, x_2)$, and for each vertex $x_i$, $0 \leq i < 3$ add edges $(x_i, w_{ci+j})$, $0 \leq j \leq 2n.n'$ (refer Figure 4.2). Note the figure gives a qualitative illustration of construction of maximal planar graph. The actual number of vertices in the cycle as shown in the figure does not correspond to the number of vertices as given by the construction.

**Lemma 18.** *Graph $G$ admits a vertex cover of size $p$ if and only if $G'$ has a vertex cover of size $p + n'(3.n + 1) + 3$.*

*Proof.* Without loss of generality we may assume $p < n$. In the forward direction consider

a vertex cover $S$ of $G$, $|S| = p$. The only edges in $G'$ that might not be covered by $S$ are those adjacent to vertices of cycle $C, C'$ and $C_3$. If we add to $S$, all vertices of $C$, alternate vertices of cycle $C'$ and all the three vertices of $C_3$, then it is easy to see that all the edges of $G'$ are covered, and the number of extra vertices that we added to $S$ is $n'+6n.n'/2+3 = n'(3n+1)+3$. So we have a vertex cover for $G'$ of size $p+n'(3n+1)+3$.

In the reverse direction consider a vertex cover $S$ of $G'$ of size $p+n'(3n+1)+3$. We first argue that all the vertices of cycle $C$ must be present in $S$. Suppose $S$ does not contain at least one vertex say $u \in V(C)$, then we need to include all its $6n+1$ neighbors of $u$ in $C'$ and from the remaining $6n.n' - (6n+1)$ vertices in $C'$ forming a path we need to include at least $\lfloor (6n.n' - (6n+1))/2 \rfloor$ vertices in the vertex cover. We need at least 2 vertices to cover the edges of $C_3$ and $\lceil n'/2 \rceil$ vertices to cover edges of the cycle $C$. Hence the number of vertices required to cover the edges of $G' \setminus G$ is at least $x$ which is given by:

$$x = (6n+1) + \left\lfloor \frac{6n.n' - (6n+1)}{2} \right\rfloor + 2 + \left\lceil \frac{n'}{2} \right\rceil$$

$$\geq 3n + 3.n.n' + 1 + \frac{n'}{2}$$

But, we know that $p + n'(3.n+1) + 3 \leq 2n + 3n.n' + 2 < x$, $(n \geq 3)$ a contradiction on the assumption of size of vertex cover of $G'$.

By a similar argument we can show that all the vertices of cycle $C_3$ must be present in $V_c$. Therefore, we have to include all $n'$ vertices in the cycle $C$, and all three vertices of $C_3$. Now we require at least $6n.n'/2 = 3n.n'$ alternate vertices of $C'$ to cover edges that are incident to vertices in $C'$. So we require at least $n'(3n+1)+3$ vertices to cover edges in $E(G') \setminus E(G)$, which implies a $p$-sized vertex cover for $G$. □

**Theorem 19.** *Deciding $k$-vertex cover on maximal planar graphs is* NP-complete.

*Proof.* Follows from Lemma 18 and Corollary 3. □

## 4.3 Discussions and concluding remarks

In this chapter we showed that the vertex cover problem is NP-complete for chordless-NST. This implies the NP-completeness of vertex cover on triangulations that are Delaunay realizable. It remains an open direction whether we can exploit the structure of a Delaunay triangulation of a point set on the plane (Delaunay triangulation with its realization on the plane) to solve the vertex cover problem optimally on them. It is not known whether given a triangulation $T$ which is known to be Delaunay realizable we can actually find a point set $P$ in the plane realizing $T$ as a Delaunay triangulation of $P$ [HMS00].

Having established the NP-completeness of vertex cover on triangulations it remains an open direction whether we can exploit the structure of triangulation or Delaunay triangulation to get a better algorithm (FPT or approximation) to compute vertex cover on them. Dillencourt [Dil90b] showed that Delaunay triangulations are 1-tough and hence have a perfect matching. It remains an open direction whether we can obtain above-guarantee FPT algorithm parameterized by the difference of the size of the vertex cover and the size of maximum matching (perfect matching in our case) for Delaunay triangulations.

# Bibliography


[ALW+03] Khaled Alzoubi, Mo Li, Yu Wang, Peng-Jun Wan, and Ophir Frieder. Geometric spanners for wireless ad hoc networks. *IEEE Transactions on Parallel and Distributed Systems*, 14(4):408–421, 2003.

[AP13] Eyal Ackerman and Rom Pinchasi. On coloring points with respect to rectangles. *Journal of Combinatorial Theory, Series A*, 120(4):811 – 815, 2013.

[BCK12] M. Basavaraju, L.S. Chandran, and T. Karthick. Maximum weight independent sets in hole- and dart-free graphs. *Discrete Applied Mathematics*, 160(16):2364 – 2369, 2012.

[BF84] Endre Boros and Zoltan Füredi. The number of triangles covering the center of an n-set. *Geometriae Dedicata*, 17:69–77, 1984.

[BLM10] Andreas Brandstädt, Vadim V Lozin, and Raffaele Mosca. Independent sets of maximum weight in apple-free graphs. *SIAM Journal on Discrete Mathematics*, 24(1):239–254, 2010.

[CG14] Timothy M. Chan and Elyot Grant. Exact algorithms and APX-hardness results for geometric packing and covering problems. *Computational Geometry: Theory and Applications*, 47(2):112–124, February 2014.

[Cha12] Timothy M. Chan. Conflict-free coloring of points with respect to rectangles and approximation algorithms for discrete independent set. In *Proceedings of the Twenty-eighth Annual Symposium on Computational Geometry*, SoCG '12, pages 293–302, New York, NY, USA, 2012. ACM.

[CKJ99] Jianer Chen, Iyad A. Kanj, and Weijia Jia. Vertex cover: Further observations and further improvements. In Peter Widmayer, Gabriele Neyer, and Stephan Eidenbenz, editors, *Graph-Theoretic Concepts in Computer Science*, volume 1665 of *Lecture Notes in Computer Science*, pages 313–324. Springer Berlin Heidelberg, 1999.

[CKX03] Jianer Chen, Iyad A. Kanj, and Ge Xia. Labeled Search Trees and Amortized Analysis: Improved Upper Bounds for NP-Hard Problems. In Toshihide Ibaraki, Naoki Katoh, and Hirotaka Ono, editors, *Algorithms and Computation*, volume 2906 of *Lecture Notes in Computer Science*, pages 148–157. Springer Berlin Heidelberg, 2003.

[CKX06] Jianer Chen, Iyad A. Kanj, and Ge Xia. Improved parameterized upper bounds for vertex cover. In Rastislav Krlovi and Pawe Urzyczyn, editors, *Mathematical Foundations of Computer Science 2006*, volume 4162 of *Lecture Notes in Computer Science*, pages 238–249. Springer Berlin Heidelberg, 2006.

[CKX10] Jianer Chen, Iyad A. Kanj, and Ge Xia. Improved upper bounds for vertex cover. *Theoretical Computer Science*, 411(40-42):3736–3756, 2010.

[CPST09] Xiaomin Chen, János Pach, Mario Szegedy, and Gábor Tardos. Delaunay graphs of point sets in the plane with respect to axis-parallel rectangles. *Random Struct. Algorithms*, 34(1):11–23, January 2009.

[CS89] Kenneth L. Clarkson and Peter W. Shor. Applications of random sampling in computational geometry, ii. *Discrete & Computational Geometry*, 4(1):387–421, 1989.

[CS12] Maria Chudnovsky and Paul Seymour. Perfect matchings in planar cubic graphs. *Combinatorica*, 32(4):403–424, 2012.

[DG95] Gautam Das and Michael T Goodrich. On the complexity of approximating and illuminating three-dimensional convex polyhedra. In *Algorithms and Data Structures*, pages 74–85. Springer, 1995.

[Die12] Reinhard Diestel. *Graph Theory, 4th Edition*, volume 173 of *Graduate texts in mathematics*. Springer, 2012.

[Dil90a] M.B. Dillencourt. Hamiltonian cycles in planar triangulations with no separating triangles. *Journal of Graph Theory*, 14(1):31–49, 1990.

[Dil90b] Michael B Dillencourt. Toughness and delaunay triangulations. *Discrete & Computational Geometry*, 5(1):575–601, 1990.

[Dil96] Michael B. Dillencourt. Finding hamiltonian cycles in Delaunay triangulations is NP-complete. *Discrete Applied Mathematics*, 64(3):207 – 217, 1996.

[DS96] Michael B. Dillencourt and Warren D. Smith. Graph-theoretical conditions for inscribability and delaunay realizability. *Discrete Mathematics*, 161(13):63 – 77, 1996.

[ELRS02] Guy Even, Zvi Lotker, Dana Ron, and Shakhar Smorodinsky. Conflict-free colorings of simple geometric regions with applications to frequency assignment in cellular networks. In *Proceedings of the 43rd Symposium on Foundations of Computer Science*, FOCS '02, pages 691–700, Washington, DC, USA, 2002. IEEE Computer Society.

[Gav72] F. Gavril. Algorithms for minimum coloring, maximum clique, minimum covering by cliques, and maximum independent set of a chordal graph. *SIAM Journal on Computing*, 1(2):180–187, 1972.

[GKS92] Leonidas J Guibas, Donald E Knuth, and Micha Sharir. Randomized incremental construction of Delaunay and Voronoi diagrams. *Algorithmica*, 7(1-6):381–413, 1992.

[HMS00] Tetsuya Hiroshima, Yuichiro Miyamoto, and Kokichi Sugihara. Another proof of polynomial-time recognizability of delaunay graphs. *IEICE Transactions on Fundamentals of Electronics, Communications and Computer Sciences*, 83(4):627–638, 2000.

[JS98] B. Jünger and J. Snoeyink. Selecting independent vertices for terrain simplification. *Proceedings of WSCG*, 1998.

[KP10] Erika L.C. King and Michael J. Pelsmajer. Dominating sets in plane triangulations. *Discrete Mathematics*, 310(1718):2221 – 2230, 2010.

[Moh01] Bojan Mohar. Face covers and the genus problem for apex graphs. *Journal of Combinatorial Theory, Series B*, 82(1):102 – 117, 2001.

[MV80] S. Micali and Vijay V. Vazirani. An $O(\sqrt{|V|}.|E|)$ algorithm for finding maximum matching in general graphs. In *21st Annual Symposium on Foundations of Computer Science, 1980*, pages 17–27, 1980.

[Nie06] Rolf Niedermeier. *Invitation to Fixed Parameter Algorithms (Oxford Lecture Series in Mathematics and Its Applications)*. Oxford University Press, USA, March 2006.

[NRRS12] N.S. Narayanaswamy, Venkatesh Raman, M.S. Ramanujan, and Saket Saurabh. LP can be a cure for Parameterized Problems. In Christoph Dürr and Thomas Wilke, editors, *29th International Symposium on Theoretical Aspects of Computer Science (STACS 2012)*, volume 14 of *Leibniz International Proceedings in Informatics (LIPIcs)*, pages 338–349, Dagstuhl, Germany, 2012. Schloss Dagstuhl–Leibniz-Zentrum fuer Informatik.

[OBSC09] Atsuyuki Okabe, Barry Boots, Kokichi Sugihara, and Sung Nok Chiu. *Spatial tessellations: concepts and applications of Voronoi diagrams*, volume 501. John Wiley & Sons, 2009.

[RAG+13] Ninad Rajgopal, Pradeesha Ashok, Sathish Govindarajan, Abhijit Khopkar, and Neeldhara Misra. Hitting and piercing rectangles induced by a point set. In Ding-Zhu Du and Guochuan Zhang, editors, *Computing and Combinatorics*, volume 7936 of *Lecture Notes in Computer Science*, pages 221–232. Springer Berlin Heidelberg, 2013.

[Raz09] Igor Razgon. Faster computation of maximum independent set and parameterized vertex cover for graphs with maximum degree 3. *Journal of Discrete Algorithms*, 7(2):191–212, 2009.

[Sib78] Robin Sibson. Locally equiangular triangulations. *The computer journal*, 21(3):243–245, 1978.

[TW11] Mu-Tsun Tsai and Douglas B West. A new proof of 3-colorability of eulerian triangulations. *Ars Mathematica Contemporanea*, 4(1), 2011.

[Ueh96] Ryuhei Uehara. NP-completeness of the problems on a restricted graph. Technical Report TWCU-M-0004, Tokyo Woman's Christian University, 1996.

[Xia10] Mingyu Xiao. A note on vertex cover in graphs with maximum degree 3. In My T. Thai and Sartaj Sahni, editors, *Computing and Combinatorics*, volume 6196 of *Lecture Notes in Computer Science*, pages 150–159. Springer Berlin Heidelberg, 2010.