Maximal k-Degenerate Spanning Subgraphs

Master's Thesis of

Nadine Davina Krisam

At the Department of Informatics
Institute of Theoretical Informatics

Reviewer:	PD Dr. Torsten Ueckerdt
Second reviewer:	Juniorprofessor Dr. Thomas Bläsius
Advisors:	Paul Jungeblut
	Laura Merker

12th May 2021 – 12th November 2021

Karlsruher Institut für Technologie
Fakultät für Informatik
Postfach 6980
76128 Karlsruhe

I hereby declare that this document has been composed by myself and describes my own work, unless otherwise acknowledged in the text. I also declare that I have read and observed the *Satzung zur Sicherung guter wissenschaftlicher Praxis am Karlsruher Institut für Technologie (KIT).*

Karlsruhe, 12.11.21

. . . .

(Nadine Davina Krisam)

Abstract

In this thesis we discuss maximum k-degenerate spanning subgraphs. A spanning subgraph is a subgraph where only edges were deleted. In a k-degenerate graph G every subgraph of G has a vertex of degree at most k. The degeneracy of a graph G is the minimum integer k such that G is k-degenerate. By deleting edges of a graph the degeneracy can be reduced.

We introduce the term of *k-degenerate skewness* $ds_k(G)$ that denotes the number of edges in a graph G that need to be deleted so that it becomes k-degenerate. We also define the *maximum k-degenerate skewness* $ds_k^{\mathcal{C}}(n)$ for a graph class $\mathcal{C}$ as the maximum of $ds_k(G)$ over all graphs $G \in \mathcal{C}$ with n vertices.

We describe general methods to reduce the degeneracy of a graph. Furthermore, we give bounds for $ds_k^{\mathcal{P}}(n)$ and $ds_k^{\mathcal{B}}(n)$. Here, $\mathcal{P}$ stands for the graph class containing all planar graphs and $\mathcal{B}$ for the graph class containing all bipartite planar graphs.

Zusammenfassung

In dieser Arbeit beschäftigen wir uns mit dem Finden von maximalen aufspannenden Subgraphen, welche k-degeneriert sind. Aufspannende Subgraphen sind Subgraphen in denen nur Kanten gelöscht werden. Ein Graph G ist k-degeneriert, wenn in jedem Subgraphen von G ein Knoten mit Grad höchstens k existiert. Die Degeneriertheit von G ist die kleinste positive Zahl k, für die der Graph k-degeneriert ist.

Wir führen hier den Begriff *k-degenerate skewness* $ds_k(G)$ ein, welcher die Anzahl von Kanten angibt, die in einem Graphen G gelöscht werden müssen, um ihn k-degeneriert zu machen. Für eine Graphklasse $\mathcal{C}$ definieren wir die *maximum k-degenerate skewness* $ds_k^{\mathcal{C}}(n)$. Dies beschreibt die maximale Anzahl von Kanten, sodass das Löschen von so vielen Kanten alle Graphen in $\mathcal{C}$ mit n Knoten k-degeneriert macht.

Wir geben in dieser Arbeit zum einen verschiedene generelle Methoden an Kanten zu löschen, sodass die Degeneriertheit von Graphen reduziert wird. Zum anderen geben wir Schranken für $ds_k^{\mathcal{P}}(n)$ und $ds_k^{\mathcal{B}}(n)$ an. Hier beschreibt $\mathcal{P}$ die Graphklasse der planaren Graphen und $\mathcal{B}$ die Graphklasse der bipartit planaren Graphen.

Contents

1. Introduction

The degeneracy of a graph G is the smallest number k such that for every subgraph $H \subseteq G$ there is a vertex of degree at most k. The graph G is then k'-degenerate for all $k' \geq k$. The degeneracy can be determined in linear time [MB83]. In this thesis we study the following problem.

Problem 1.1: *Given a graph G and an integer k, find the largest subgraph H of G such that H is k-degenerate.*

This problem was motivated by the fact that some algorithms only work for graphs that are k-degenerate for a certain integer k. An example are Lombardi drawings. In their paper, Duncan et al. [Dun+11] proved that every 2-degenerate graph has a circular Lombardi drawing by giving a procedure how to draw them. Because it is not always possible to draw arbitrary k-degenerate graphs, it might help to run the algorithm on a subgraph of the original graph. For example it is possible to run a graph drawing algorithm on a large subgraph H of a graph G and then to draw only the vertices and edges of $G - H$ by hand.

There are different versions of Problem 1.1. For one, the subgraph can be derived by only deleting vertices. Then, the size of a subgraph can be defined as the number of vertices. In this case, the subgraph is an induced subgraph. Thus, here the largest subgraph is the induced subgraph with the largest number of vertices that is k-degenerate. This problem was studied in several papers. For example in [LMZ15] the authors gave a lower bound on the number of vertices of maximum 4-degenerate induced subgraphs of planar graphs. This and other papers are described further in Chapter 3.

In a different version of the Problem 1.1, we only allow deleting edges. A subgraph that is derived by deleting edges is called a *spanning subgraph*. A subgraph is a maximum k-degenerate spanning subgraph if, by deleting one edge less, the graph is no longer k-degenerate. This is the version we study in this thesis. In Figure 1.1 (left) an example is given of a graph G. This graph is not 1-degenerate. A maximum 1-degenerate spanning subgraph G' of G is given in Figure 1.1 (right). For $k \geq 2$ deleting an edge has the same effect as subdividing them. This again is interesting for graph drawings. Subdividing an edge is similar to allowing a bend in an edge.

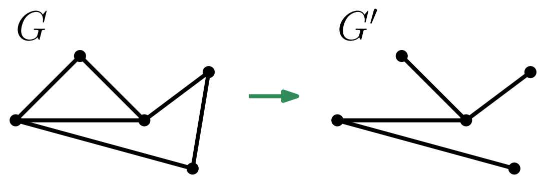

Figure 1.1.: Graph G that is not 1-degenerate and a 1-degenerate maximum spanning subgraph G' of G.

We can further restrict Problem 1.1 by only allowing graphs of a certain graph class. Often, the considered graphs are restricted to planar graphs. This is done to get more concrete results. Also the integer k is often very small. A large part of this thesis focuses on planar graphs and bipartite planar graphs. Planar graphs are already known to be 5-degenerate so the problem is only interesting for integers smaller than 5. Similarly, bipartite planar graphs are 3-degenerate and only integers smaller than 3 need to be considered.

1.1. Contribution

In this thesis, we study Problem 1.1 restricted to spanning subgraphs, as stated above. We define the *k-degenerate skewness* $ds_k(G)$ which allows us to count the number of edges that need to be deleted to obtain a k-degenerate subgraph of G. We also define the *maximum k-degenerate skewness* $ds_k^{\mathcal{C}}(n)$ for a graph class $\mathcal{C}$ as the maximum of $ds_k(G)$ over all graphs $G \in \mathcal{C}$ with n vertices. We discuss several general methods to find edges that should be deleted. We then go on and prove the maximum k-degenerate skewness for the graph class $\mathcal{B}$ of bipartite planar graphs and the graph class $\mathcal{P}$ of planar graphs as shown in Table 1.1. Whereas we do not obtain concrete values for $ds_k^{\mathcal{C}}(n)$ we prove bounds.

Table 1.1.: Maximum k-degenerate skewness for the graph class $\mathcal{B}$ of bipartite planar graphs and the graph class $\mathcal{P}$ of planar graphs.

$ds_k^{\mathcal{C}}(n)$	$\mathcal{C} = \mathcal{B}$		$\mathcal{C} = \mathcal{P}$	
$k = 1$	$= n - 3$	(Lemma 5.1)	$= 2n - 5$	(Lemma 6.1)
$k = 2$	$\geq n/4 - 1$ $\leq n/2 - 1$	(Lemma 5.4) (Lemma 5.8)	$= n - 3$	(Theorem 6.4)
$k = 3$	$= 0$		$> n/4$ $\leq n - 3$	(Lemma 6.7) (Lemma 6.8)
$k = 4$	$= 0$		$\leq 3/5n$	(Lemma 6.9)
$k = 5$	$= 0$		$= 0$	

Additionally, we conjecture that the upper bound for $ds_2^{\mathcal{B}}(n)$ can be reduced to $\frac{n}{4}$.

1.2. Outline

In Chapter 2, we give definitions of the graph parameters considered in this thesis, in particular degenerate skewness. We also prove bounds on the degeneracy for certain graph classes and show how the degeneracy relates to other graph parameters. In Chapter 3, we look at related work. Here we present papers already published in this field and state important results. In Chapter 4, we formulate several methods to obtain k-degenerate subgraphs. These methods often do not result in maximum spanning k-degenerate subgraphs, but instead delete more edges than necessary. However, they are general methods that can be applied to multiple graphs classes. In Chapter 5, we prove bounds for the k-degenerate skewness for bipartite planar graphs. The same is done for planar graphs in Chapter 6. We conclude this thesis with Chapter 7 by suggesting open problems for further research.

2. Preliminaries

2.1. Graph Definitions

A *graph* is a tuple of sets $G = (V, E)$, where V is the set of *vertices* and E the set of *edges*. In this thesis we restrict our graphs to undirected and simple graphs. For a vertex $v \in V$ we denote by $\deg(v)$ the *degree* of v, i.e. the number of edges incident to v. The *order* of a graph G is the number of vertices of G and is denoted by $|V(G)|$. We often use the letter $n = |V(G)|$ for short. In the same way $|E(G)| = m$ denotes the number of edges. A vertex of a certain degree k is denoted by *k-vertex*, a vertex of degree at most k by *k^--vertex* and a vertex of degree at least k by *k^+-vertex*. For edges we define the *weight* of an edge e as the sum of the degrees of the vertices incident to e. Again we denote by *w-edge* an edge with weight w.

A *planar graph* is a graph that can be embedded into the $\mathbb{R}^2$ plane such that vertices are points in the plane and edges are arcs where only the endpoints can intersect each other. The regions of $\mathbb{R}^2$ separated by edges and vertices of a graph in an embedding are called *faces*. We denote by $|F(G)| = f$ the number of faces of a planar graph. For a face $g \in F(G)$ we denote the *length of the face*, that is the number of edges in the cycle bounding the face g, by $l(g)$. Planar graphs have certain properties, like the following. The number of edges m is bounded by the number of vertices n with $m \leq 3n - 6$. Equality holds if and only if the graph is *triangulated*, that means if every face is bounded by exactly three edges and thus forms a triangle. This follows from Euler's formula $n - m + f = 2$. With the pigeonhole principle we can derive that there always exists a vertex v of degree $\deg(v) \leq 5$, that is a 5^--vertex.

A *bipartite graph* is a graph G where the vertex set can be split into two disjoint sets $V = U \dot\cup W$ and every edges of G connects a vertex in U to a vertex in W. A *bipartite planar graph* is a planar graph that is bipartite. Observe that faces of such a graph are always bounded by an even number of edges. A *quadrangulation* of a graph is a graph where every face is bounded by four edges and thus forms a quadrangle. Such a quadrangulation is a maximal bipartite planar graph. This results in the following inequalities:

$$\begin{aligned} m &\leq 2n - 4 \\ 4f &\leq 2m \\ f &\leq n - 2 \end{aligned} \tag{2.1}$$

By pigeonhole principle we know that there always exist a 3^--vertex.

2.2. Definition of Degeneracy

A graph G is *k-degenerate* if each subgraph of G contains at least one vertex of degree at most k. The *degeneracy* of a graph G is the smallest k, such that G is k-degenerate.

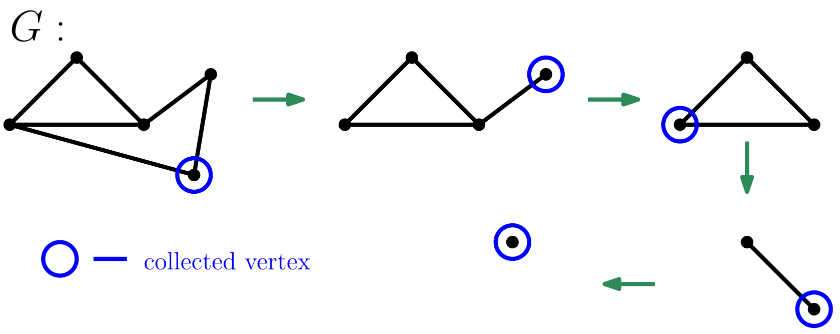

Figure 2.1.: Example of using the algorithm, that always collects a vertex of minimal degree. The example graph G is 2-degenerate.

For a graph G and a vertex v of G the process of *collecting* v describes the removal of v from G. So after collecting v we obtain graph $G - \{v\}$. Let σ be an ordering of $V(G)$. We call it a *collecting ordering* if we collect the vertices of G in this order. We define the *maximum degree of* σ as the the maximum degree of a vertex at the time of its collection. This means that if the maximum degree of σ is equal to k, then k is the smallest integer such that every collected vertex has degree at most k.

In 1983 Matula and Beck developed the following linear time algorithm [MB83]: Given a graph G, take the vertex v with smallest degree and collect it. Repeat this process until the graph is empty. This results in a collecting ordering π. They then went on an proved that the degeneracy of G is equal to the maximum degree of π. We call π a *minimum collecting ordering*. An example of how this algorithm works is given in Figure 2.1.

For every other collecting ordering σ the maximum degree is at least as big as the maximum degree of the minimum collecting ordering.

Lemma 2.1: *Let G be a graph and π a minimum collecting ordering with maximum degree k. For every other collecting ordering σ the maximum degree is k' with $k' \geq k$.*

Proof. Let σ be a collecting ordering with maximum degree k'. Assume $k' < k$. We now collect vertices according to the order π. At some point, the next collected vertex v is a vertex of degree k. Otherwise, the maximum degree of π is smaller than k. Let u be the first vertex according to order σ that has not yet been collected. The degree of u is at most $k' < k$. This is true, because all vertices previous to u in order σ have already been deleted. Because σ is a collecting ordering with maximum degree k', at this point the degree of u is at most k'. By deleting the other vertices previous to v in order π we can only reduce the degree of u further. But then the algorithm would collect vertex u before it would collect vertex v, because it collects the vertex with minimum degree. Therefore either π is not a minimum collecting ordering or the degree of π was not k with $k > k'$. This proves the claim. □

In the following chapters, we often provide an order σ in which we collect vertices of a graph G. Because of Lemma 2.1 we can now use this σ to assure the degeneracy of a graph. We formulate this in the following corollary.

Corollary 2.2: *Let G be a graph and let σ be a collecting ordering with maximum degree k. Then G is k-degenerate.*

The *skewness* of a graph G is defined as the minimum number of edges, that need to be deleted, so that G becomes planar [Kai74]. Similarly, we introduce the parameter *k-degenerate skewness* $ds_k(G)$ of a graph G as the minimum number of edges of G that need to be deleted, such that the graph becomes k-degenerate. Also we introduce the parameter *maximum k-degenerate skewness* $ds_k^{\mathcal{C}}(n)$. It is defined as $ds_k^{\mathcal{C}}(n) = \max\{ds_k(G) \colon G \in \mathcal{C}, |V(G)| = n\}$, the maximum of the minimum number of edges that need to be deleted, such that every graph G of the graph class $\mathcal{C}$ with n vertices, can be made k-degenerate. The following lemma helps with the relations between the skewness parameters.

Lemma 2.3: *For $k \geq 0$, a graph class $\mathcal{C}$ and $n \geq 1$ it holds that $ds_k^{\mathcal{C}}(n) \geq ds_{k+1}^{\mathcal{C}}(n)$.*

Proof. Let $G \in \mathcal{C}$ be a graph where $ds_k^{\mathcal{C}}(n)$ edges need to be deleted, to obtain a k-degenerate graph G'. Being k-degenerate means that every subgraph of G' has a vertex of degree at most k. But then, every subgraph of G' also as a vertex of degree at most $k + 1 > k$ and thus G' is $k + 1$-degenerate. Therefore we need to delete at most $ds_k^{\mathcal{C}}(n)$ edges to make every graph $G \in \mathcal{C}$ $k + 1$-degenerate and this proves the lemma. $\square$

Lemma 2.4: *For $k \geq 0$, graph classes $\mathcal{B}$,$\mathcal{C}$ with $\mathcal{B} \subseteq \mathcal{C}$ and $n \geq 1$ it holds that $ds_k^{\mathcal{B}}(n) \leq ds_k^{\mathcal{C}}(n)$.*

Proof. Every graph $G \in \mathcal{B}$ is also element in $\mathcal{C}$. If G is a graph with maximum $ds_k(G)$ and n vertices in $\mathcal{B}$, then $ds_k(G) = ds_k^{\mathcal{B}}(n)$. Also $ds_k(G) \leq ds_k^{\mathcal{C}}(n)$. Thus $ds_k^{\mathcal{B}}(n) \leq ds_k^{\mathcal{C}}(n)$. $\square$

A graph $G \in C$ is called *edge maximal* for a graph class $\mathcal{C}$, if by adding even one more edge to G resulting in a graph G', $G' \notin \mathcal{C}$.

Lemma 2.5: *For a graph class $\mathcal{C}$ let $\mathcal{C}' \subseteq \mathcal{C}$ be the edge maximal graphs in $\mathcal{C}$. Then for $n \geq 1$ it holds that $ds_k^{\mathcal{C}}(n) \leq ds_k^{\mathcal{C}'}(n)$.*

Proof. Let G be a graph in $\mathcal{C}$ with n vertices. If G is edge maximal, it is also in $\mathcal{C}'$ and $ds_k(G) \leq ds_k^{\mathcal{C}}(n)$ and $ds_k(G) \leq ds_k^{\mathcal{C}'}(n)$. If G is not edge maximal, then we need to prove, that not just $ds_k(G) \leq ds_k^{\mathcal{C}}(n)$ but also $ds_k(G) \leq ds_k^{\mathcal{C}'}(n)$. Let G' be an edge maximal graph in $\mathcal{C}$, that has G as subgraph. The graph G' exists because we can simply add edges to G until adding more would result in the graph not being in $\mathcal{C}$ anymore. Then G' is also in $\mathcal{C}'$ and thus $ds_k(G') \leq \mathcal{C}'$. Also $ds_k(G) \leq ds_k(G')$, because we only added edges. Thus, we need to delete at least as many edges in G' as in G to make G' k-degenerate. But then $ds_k(G) \leq ds_k^{\mathcal{C}'}(n)$, which proves the lemma. $\square$

The edge maximal planar graphs are triangulated. From Lemma 2.5 if follows that we only need to look at triangulated graphs to find the maximum k-degenerate skewness of planar graphs. The same holds for bipartite planar graphs and quadrangulated graphs.

2.2.1. Degeneracy for Different Graph Classes

In this part we look at different graph classes and determine their degeneracy. That means for each class we determine the smallest k, such that every graph in that class is k-degenerate. That does not mean that a specific graph of this graph class cannot have a smaller degeneracy.

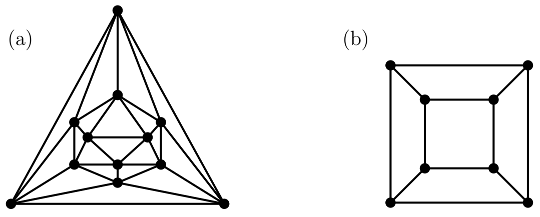

Figure 2.2.: (a) Icosahedron, a planar graph with minimum degree 5. (b) Cube, a bipartite planar graph with minimum degree 3.

Planar Graphs Every planar graph has a vertex of degree at most 5. We also know that every subgraph of a planar graph is a planar graph. Therefore every planar graph is 5-degenerate. There are planar graphs, that have minimum degree 5, for example the Icosahedron, see Figure 2.2 (a). Thus planar graphs have degeneracy 5.

Forests Every forest has at least one leaf. Leaves are defined as vertices of degree 1. Every subgraph of a forest is again a forest. Therefore forests have degeneracy 1.

Bipatite Planar Graphs Every bipartite planar graph has a vertex of degree at most 3. Also every subgraph of a bipartite planar graph is again bipartite and planar. Thus those graphs are all 3-degenerate. There are bipartite planar graphs, that have minimum degree 3, therefore they have degeneracy 3. An example is the cube, as seen in Figure 2.2 (b).

2.2.2. Degeneracy and Other Graph Parameters

Degeneracy is a graph parameter that is not often used. There are other more common parameters that are bounded by the degeneracy or bound it. These can be used to get a feeling of how big or small the degeneracy of a graph can be. For the following paragraphs, we assume that the considered graph is k-degenerate.

Chromatic Numbers The *chromatic number* χ in a graph G is defined as the minimum number of colors needed to label all vertices in G such that no two vertices with the same color are adjacent. Let π be the minimum collecting ordering of G and k its maximum degree. Now take the vertices in the reverse order of π and use a greedy coloring. When a vertex v is assigned a color, it has at most k neighbors that have already been assigned a color because G is k-degenerate. Because this is true for every vertex, we need at most $k+1$ colors to label all vertices. Therefore $\boldsymbol{\chi \leq k+1}$.

Arboricity The *arboricity* a of a graph G is the minimum number of forests that are needed to partition the edges of G. First we show, that $\boldsymbol{a \leq k}$. For this, we map every edge to one of k forests. We take a minimum collecting ordering π and traverse the vertices in this order. When we get to vertex v, there are at most k edges incident to v not assigned to a forest. We map each of these edges to a distinct forest. When we map every edge in this way we get no cycle in a forest because otherwise, we would have needed a vertex v with two edges two vertices that come later in the ordering π than v. This never happened.

Next we show that $\boldsymbol{k \leq 2a - 1}$. Every subgraph of a graph of arboricity a also has arboricity at most a. Thus, every subgraph G' has average degree at most $\frac{2a(n-1)}{n}$, where $n = |V(G')|$. This is true because the average degree of a graph G is $\frac{2|E(G)|}{|V(G)|}$, a forest F has at most $|V(F)| - 1$ edges and the edges of G' can be partitioned in at most a forests. The average degree gives us, that in every subgraph of G we can find at least one vertex of degree $2a - 1$. Therefore the degeneracy k of G must be at most $2a - 1$.

Acyclic Chromatic Number The *acyclic chromatic number* l of a graph G is defined as the smallest size of a vertex partition $\{V_1, \dots, V_l\}$ such that each V_i, $1 \leq i \leq l$, is an independent set and for all i, j, $1 \leq i < j \leq l$ the induced subgraph $G[V_i \cup V_j]$ does not contain a cycle. One can look at the vertices in V_i as the vertices that have been assigned the color i for $1 \leq i \leq l$. We can partition the edges in $\binom{l}{2}$ forests, one for each pair V_i, V_j with $1 \leq i < j \leq l$. Thus, the arboricity $a \leq \binom{l}{2}$. In the paragraph *Arboricity* we have shown $k \leq 2a - 1$, therefore we get $\boldsymbol{k \leq 2\binom{l}{2} - 1}$.

Treewidth The *treewidth* t of a graph G is defined as the size of the largest clique in a chordal completion of G minus 1. We now prove that $\boldsymbol{k \leq t}$. Let us take the order of a perfect elimination ordering σ of the chordal completion. In the ordering σ each vertex v and all neighbors of v, that come later in σ form a clique. The size of the largest clique is at most $t + 1$. Thus, every vertex has at most t neighbors that come later in σ. Then, we can take σ as collecting order and, because of Lemma 2.2, the degeneracy of the graph is at most t.

2.3. Discharging

The *discharging method* was introduced in [Wer04] by Wernicke in 1904, to prove that every planar graph of minimum degree 5, has at least one edge of weight 10 or 11. He used this as part of his proof of the Four Color Theorem. This proof was not complete, but the method is correct all the same. In general, discharging can be used to prove the existence of certain subgraphs or local configurations. A function $c\colon V \mapsto \mathbb{R}$ gives every vertex v a charge $c(v)$. In the case of discharging on planar graphs, faces often get a charge, too. The charge of the vertices (and faces) then gets redistributed. If the existence of a configuration should be proven, the redistribution of the charge under the assumption that such a configuration does not exist leads to a contradiction of some sort.

A method often used is *degree charging.* Here every vertex $v \in V(G)$ gets initial charge $c(v) = \deg(v)$. A redistribution does not change the total sum of charges. Thus statements about the sum of degrees in a graph can now be applied to the charges. After the redistribution of the charge these statements must still be true. But if a certain configuration does not exist, the statements are often no longer satisfied after redistribution. An example for such a statement is the average degree of a graph.

By restricting the type of graph by certain preconditions other contradictions can be made. For example, a precondition can be that the average degree is at most 3. If every vertex has charge at least 3 after redistribution, when certain configurations do not exist, then the precondition is not satisfied. Therefore, the configurations must exist.

In their paper “An Introduction to the Discharging Method via Graph Coloring” Daniel W. Cranston and Douglas B. West explain the discharging technique and give multiple examples how it can be used in different cases [CW17].

3. Related Work

In this chapter we give an overview on already published results similar to those stated in this thesis. We first name papers on degeneracy in Section 3.1. Then, in Section 3.2, we discuss results of papers on finding maximum induced subgraphs of a certain degeneracy. We go on to list papers on generating triangulations and quadrangulations in Section 3.3. At last we name several papers on skewness of non-planar graphs in Section 3.4.

3.1. General Statements on Degeneracy

In their paper "On chormatic number of graphs and set-systems" Erdős and Hajnal define the *coloring number* k [EH66]. In their definition k is the smallest integer such that there exists an ordering σ in which every vertex x has less than k neighbors y that appear before x in σ. This is similar to our definition of degeneracy with the only difference being that for our definition v can have at most k neighbors y that appear before x in σ.

In 1970 Don R. Lick and Arthur T. White defined degeneracy in their paper "k-degenerate Graphs" [LW70]. In this, they introduced the graph classes Π_k that consist of all k-degenerate graphs. In some early observations they stated that Π_0 is equivalent to the class of graphs with no edges and Π_1 is equivalent to the class of graphs with only forests. Furthermore they observed that all outerplanar graphs lie in Π_2 and all planar graphs in Π_5. The following lemma was stated in [LW70] and is the base of the algorithm that produces a minimum collection ordering as described in Section 2.2. We adapt some notation to fit the notation in this thesis.

Lemma 3.1 (Proposition 1 in [LW70]): *A graph G is in Π_k if and only if the vertices in $V(G)$ can be ordered, say $v_1, \dots, v_n$, such that the degree $\deg(v_1) \leq k$ and, in the subgraph induced by the vertices $v_p, \dots, v_n$ of G, $\deg(v_p) \leq k$, for each $1 \leq p \leq n$. In other words, G can be reduced to the degenerate (i.e. trivial) graph K_1 by a sequence of removal of vertices of degree less than or equal to k.*

Other important facts the authers show in their second Proposition are that, if a graph is k-degenerate, then it is also k'-degenerate for all $k' \geq k$ and that, if G is k-degenerate, then also every subgraph of G is k-degenerate. For us also interesting is their third Proposition . Here, they prove that a k-degenerate graph G with n vertices has at most $kn - \binom{k+1}{2}$ edges. We use this proposition later in Section 6.2. There, we want to find a maximum 2-degenerate spanning subgraph of a planar graph and with Proposition 3 we know that such a subgraph has at most $2n - 3$ edges. There are cases in which a graph has at most $kn - \binom{k+1}{2}$ edges but is still not k-degenerate. This is the case for all bipartite planar graphs with minimum degree 3. They are not 2-degenerate but every bipartite planar graph has at most $2n - 4$ edges which is less than $2n - 3$.

Aside from general statements about k-degenerate graphs they also introduced the concept of the point partition number. We call it a vertex partition number p_k. It is the minimum number of induced subgraphs a graph G can be partitioned in, such that every subgraph is k-degenerate. Note that p_0 is the equivalent to the chromatic number.

In Section 2.2.2, we compare degeneracy to other graph parameters. An overview of different graph parameters and how they are related to each other is given by Manuel Sorge et al. in [Sa19]. In Figure 1 in [Sa19] graph parameters are displayed in a Hasse diagram where one parameter is above another parameter if the former bounds the later.

A graph parameter that is closely related to the degeneracy is the chromatic number. In Paragraph *Chromatic Numbers* in Section 2.2.2 we prove that the chromatic number χ of a graph G is at most $k+1$ if G is k-degenerate. This is done by using a minimum collecting ordering for the greedy coloring. Lick and White proved in 1970 that every 4-degenerate planar graph has a 4-coloring (Theorem 7 in [LW70]). They then deduced that all planar graphs that do not have a 4-coloring must be 5-degenerate but not 4-degenerate. Six years later, Appel and Haken proved the Four Color Theorem that all planar graphs have a 4-coloring [AH78].

In Section 2.2.1, we introduce the degeneracy of certain graph classes. It is also possible to determine the degeneracy of graphs without certain subgraphs. For example Wang and Lih proved in 2001 that planar graphs without 5-cycles are 3-degenerate. In 2002 G. Fijavz et al. proved that planar graphs without 6-cycles are also 3-degenerate. For 2-degeneracy, it is a well known fact that planar graphs of girth 6 are 2-degenerate. Jumnongnit and Pimpasalee proved in 2021 that every planar graph without 4-,6-,8-, and 10-cycles is also 2-degenerate.

3.2. Subgraphs of certain Degeneracy

As discussed in Chapter 1, Problem 1.1 can be restricted to the problem of finding the maximum k-degenerate induced subgraph of a graph. In 2018 Lukot'ka, Mazák and Zhu wrote a paper on “Maximum 4-degenerate subgraph of a planar graph” [LMZ15]. They proved in Theorem 1 in [LMZ15] that if a connected planar graph G with n vertices has average degree $d \geq 2$, then there exists a 4-degenerate induced subgraph of G that has at least $(38-d)/36 \cdot n$ vertices. Furthermore in Corollary 3 in [LMZ15] the authors proved that for every planar graph with n vertices there exists a 4-degenerate induced subgraph that has at least $8/9 \cdot n$ vertices. They conjecture that this bound can be raised to $11/12 \cdot n$. To prove these facts, they use discharging, as described in Section 2.3. Here, they give initial charge to vertices and faces. They then define a discharging procedure that leaves all faces with negative charge. Afterwards they prove that a minimal counterexample to Theorem 1 must contain certain vertices with positive charge after the discharging procedure. These vertices do not exists, which proves their claim.

In a paper “3-degenerate induced subgraph of a planar graph” Gu et al. proved a lower bound on the number of vertices in an induced 3-degenerate subgraph of a planar graph [Gu+]. If G is a planar graph with n vertices, then there exists an induced 3-degenerate subgraph of G with at least $(3n+2)/4$ vertices. In [DK18] Dvořák and

Kelly gave a lower bound on the number of vertices in an induced 2-degenerate subgraph of a triangle-free planar graph. They proved that there exists a 2-degenerate induced subgraph with at least $4/5 \cdot n$ vertices for every graph with n vertices.

Alternatively it is also possible to decompose every planar graph G into two induced subgraphs G' and G'' where G' is 2-degenerate and G'' 1-degenerate, i.e. a forest. This was proven 1993 by Carten Thomassen [Tho95]. Note that for this paper Thomassen used the definition that in a k-degenerate graph every subgraph has a vertex of degree less than k, but here we use our definition of degree at most k. Additionaly Thomassen proved in 2000 that it is also possible to decompose a planar graph G into two induced subgraphs G' and G'' where G' is 3-degenerate and G'' is 0-degenerate, i.e. an independent set [Tho01].

The problem how many vertices must be deleted so that a graph becomes k-degenerate for a certain integer k is NP-complete. This was proven by Lewis and Yannakakis [LY80] in 1980. Here they proved that every node-deletion problem is NP-complete for any nontrivial and hereditary graph property. Note that the term *node* is equivalent to our term *vertex*. A graph property is defined as *nontrivial* if there exists an infinite number of graphs that satisfy the property and an infinite number of graphs that do not. A *hereditary* graph property on induced subgraphs is a property that is satisfied by all induced subgraphs of a graph G if G satisfies the property. The graph property whether a graph is k-degenerate for some integer k is nontrivial and hereditary. For example, for $k = 1$ there is an infinite number of graphs that are forests, which all are 1-degenerate. However there is also an infinite number of graphs that are not forests, which are all not 1-degenerate. The same is true for $k > 1$. Degeneracy is hereditary because the definition states that a graph is k-degenerate if and only if every subgraph is k-degenerate. Thus we can apply the statement in [LY80] to Problem 1.1 where the subgraph must be an induced subgraph.

3.3. Generating Triangulations and Quadrangulations

As stated before, a large part of this thesis is done on planar and bipartite planar graphs. We denote the class consisting of planar graphs by $\mathcal{P}$ and the class consisting of bipartite planar graphs by $\mathcal{B}$. To find a sharp lower bound for the maximum k-degenerate skewness $ds_k^{\mathcal{P}}(n)$ and $ds_k^{\mathcal{B}}(n)$ as defined in Section 2.2 we need to find graphs with large k-degenerate skewness. Edge maximal planar graphs are triangulations and edge maximal bipartite planar graphs are quadrangulations. Thus, we search for methods to generate triangulations and quadrangulations.

In 2005 Brinkmann et al. published a paper on “Generation of simple quadrangulations of the sphere” [Bri+05]. A *simple quadrangulation* here means that there are no multi-edges in a graph. They defined four graph classes: The class $\mathcal{L}_1$ of all simple quadrangulations of the sphere. The class $\mathcal{L}_2$ of all simple quadrangulations of the sphere with minimum degree 3. The class $\mathcal{L}_3$ of all 3-connected quadrangulations of the sphere. The class $\mathcal{L}_4$ of all 3-connected quadrangulations of the sphere without separating 4-cycles. They then gave four types of expansion procedures that generate all four graph classes. In Section 5.1 we use graphs of the class $\mathcal{L}_2$ that are generated according to these procedures in [Bri+05].

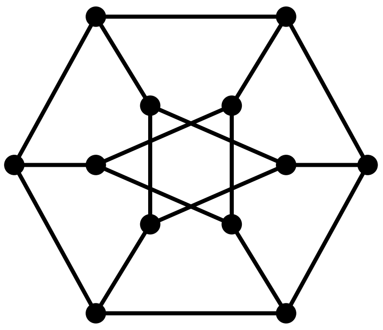

Figure 3.1.: The Petersen graph $P(6, 2)$.

In a different paper from 2005 Brinkmann et al. discussed the "Construction of planar triangulations with minimum degree 5" [BM05]. They distinguished triangulations with minimum degree 5 based on their connectivity. They then listed different operations that create these triangulations. All triangulations are based on the Icosahedron. The graph in Section 6.4 is based on an Icosahedron that was expanded by three operations of type $\mathcal{C}$ as defined in Fig. 1 on Page 3 in [BM05].

3.4. Skewness

In Section 2.2, we introduce the term *k-degenerate skewness*. We derived the name from the term *skewness*. The skewness of a graph is the number of edges that need to be deleted so that a graph becomes planar. Thus, the k-degenerate skewness is defined as the number of edges that need to be deleted so that a graph becomes k-degenerate. There are multiple papers that discuss skewness. In 1974 Kainen used skewness to generalize the 5-color theorem [Kai74]. It is easy to see that every planar graph can be 5-colored. Kainen proved that if the skewness of a graph is at most 2, it can be 5-colored. An important part of the proof is to show that every graph with skewness at most 2 has a vertex of degree at most 5. This can be proven by using the fact that a graph G with skewness at most 2 has at most $3 \cdot |V(G)| - 4$ edges. Kainen later went on and proved that every graph with skewness at most 5 can be 6-colored. In his paper one year later he formalized that every graph with skewness less than $\binom{k}{2}$ for a $k \geq 3$ can be $(k + 2)$-colored. He also proved that that there is a graph with skewness less than $\binom{k}{2}$ for a $k \geq 3$ that can not be $(k + 1)$-colored.

In 2005 G.L. Chia et al. found the skewness for certain generalized Petersen graphs [CL05]. A generalized Petersen graph $P(n, k)$ consists of a set $\{u_i, v_i \,|\, 0 \leq i < n\}$ of vertices and a set $\{u_iu_{i+1}, u_iv_i, v_iv_{i+k} \,|\, 0 \leq i < n$ where indices are take modulo n$\}$. The Petersen graph $P(6, 2)$ is shown in Figure 3.1. They proved that for $P(3k, k)$ the skewness is $\lceil \frac{k}{2} \rceil + 1$ with $k \geq 4$. In later papers, for example in [Gek12] and [CS13] they discussed further bounds for the skewness of Petersen graphs.

In 1977 Liu and Geldmacher proved in their paper "On the deletion of nonplanar edges of a graph" that the problem to find the skewness of a nonplanar graph is NP-complete [Liu77]. However, there are several heuristics for this problem. One was given in 1992 by Robert Cimikowski [Cim92]. His heuristic to find a maximum spanning planar subgraph is based on spanning trees. He guarantees that the size of the resulting subgraph is at least $\frac{2}{3}$ of the size of the optimum. Later he proved that the skewness of the n-dimensional hypercube Q_n is $2^n(n-2) - n \cdot 2^{n-1} + 4$.

4. General Methods to Obtain k-degenerate Subgraphs

In this chapter we look at several methods to find subgraphs of a certain degeneracy. These are obvious approaches to solve this problem and are easy in their execution. These techniques rarely result in subgraphs of maximum size. This means that using these approaches the number of deleted edges for a graph G is often a lot greater than the k-degenerate skewness $ds_k(G)$. But these are methods that can be applied to graphs in different graph classes. Techniques that result in stronger bounds for planar graphs and bipartite planar graphs are introduced in Chapter 6 and Chapter 5.

4.1. 1-degenerate via Spanning Trees

We begin with finding maximum 1-degenerate subgraphs in graphs. In the paragraph Forests in Section 2.2.1 we show, that every forest is 1-degenerate. It is also clear that every 1-degenerate graph G is a forest. Otherwise, the graph would have a cycle C giving us a not 1-degenerate but 2-degenerate subgraph. This violates the condition that every subgraph of G is 1-degenerate as well.

Now, we use a spanning tree algorithm for a graph G. If G is not connected, we find spanning trees in each component so we assume G is connected. We only need a spanning tree and not minimum spanning trees of a given weight function. Thus, we can simply use a depth-first search or breadth-first search and find a spanning tree of G in linear time.

The number of edges in a spanning tree is $|V(G)| - 1$. Thus to obtain a 1-degenerate subgraph G' of a connected graph G we need to delete exactly $|E(G)| - |V(G)| + 1$ edges. Then $ds_1(G) = |E(G)| - |V(G)| + 1$ and this method give us maximum 1-degenerate subgraphs of G.

4.2. Greedy Algorithms

The first general approach we use is a greedy approach. In Section 2.2 we introduce an algorithm that, given G, always deletes a vertex of minimum degree. The degeneracy of G then is the maximum degree k of all collected vertices.

We use a similar algorithm to find a k'-degenerate subgraph G' of a k-degenerate graph G for $k' \leq k$. Take a vertex $v \in V(G)$ with minimum degree. Delete arbitrary edges incident to v until v has degree at most k'. Then collect v and repeat these steps until all vertices are collected. The subgraph G' is then the graph G without the deleted edges. Note, there is a difference between deleted edges, and edges that are removed when a vertex is collected. The later edges are part of G'.

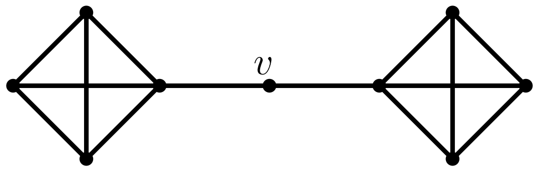

Figure 4.1.: A graph G where the greedy algorithm deletes more edges than necessary.

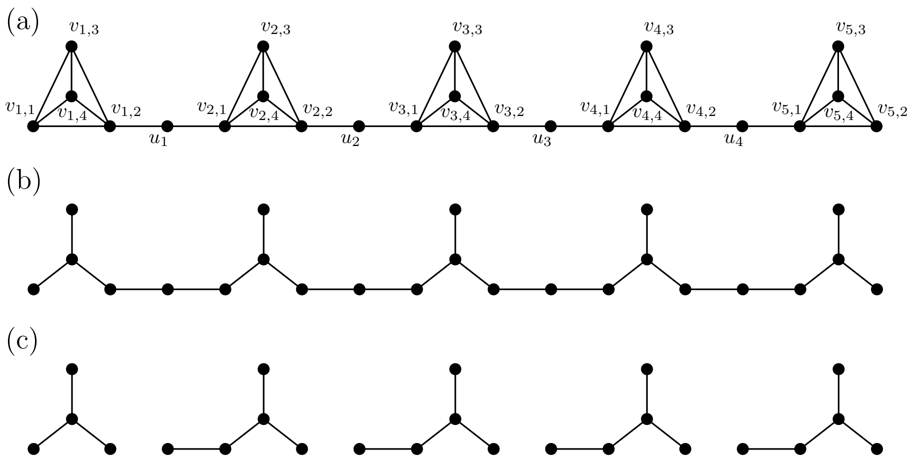

Figure 4.2.: (a) Another example of a graph G where the greedy algorithm deletes more edges than necessary. It consists of $d = 5$ copies of K_4 that are connected by $d - 1$ paths of length 2.
(b) An example of a maximum subgraph of G that is 1-degenerate.
(c) A result of the greedy algorithm.

In general, for a graph G the number of edges that are deleted is greater than $ds_{k'}(G)$. That means, we delete more edges than necessary. This can be shown with the example in Figure 4.1. Here we have a 3-degenerate graph G that we want to make 1-degenerate. The graph G consist of two complete graphs of size 4, i.e. two K_4 graphs, and a path of length 2 connecting them. The vertex v in the middle of the graph has degree 2 and is chosen first in the greedy algorithm. To reduce the degree of v to 1, so that it can be collected, one edge incident to v has to be deleted. Afterwards v can be collected. Then the graph breaks into two components that are both K_4 graphs. Thus the resulting 1-degenerate subgraph is not connected. But all maximum 1-degenerate subgraphs of G are spanning trees. Therefore, we delete at least one edge more than $ds_1(G)$ indicates.

We can extend the graph in Figure 4.1 by using d copies of K_4 graphs and connect them all with $d-1$ paths of length 2. Let $v_{i,1}$,$v_{i,2}$,$v_{i,3}$,$v_{i,4}$ be the vertices of the i-th copy of K_4 for $1 \leq i \leq d$. For $1 \leq i < d$ we connect the i-th and $(i + 1)$-th copy by adding a vertex u_i and edges between $v_{i,2}$ and u_i and between u_i and $v_{i+1,1}$. An example graph G with $d = 5$ is given in Figure 4.2 (a). Graph G has $|V(G)| = 5 \cdot d - 1$ vertices and $|E(G)| = 8 \cdot d - 2$ edges. In Section 4.1 we show that all maximum subgraphs of a graph that are 1-degenerate are spanning trees. Thus a maximum 1-degenerate subgraph G' of G has $|E(G')| = 5 \cdot d - 2$ edges and we need to delete $3 \cdot d$ edges. Graph G' is shown in Figure 4.2 (b). In Figure 4.2 (c) we see an example of the result when using the greedy algorithm to obtain a 1-degenerate subgraph. Figure 4.3 shows how the graph in Figure 4.2 (a) is reduced when using the greedy algorithm. Here, we colored the

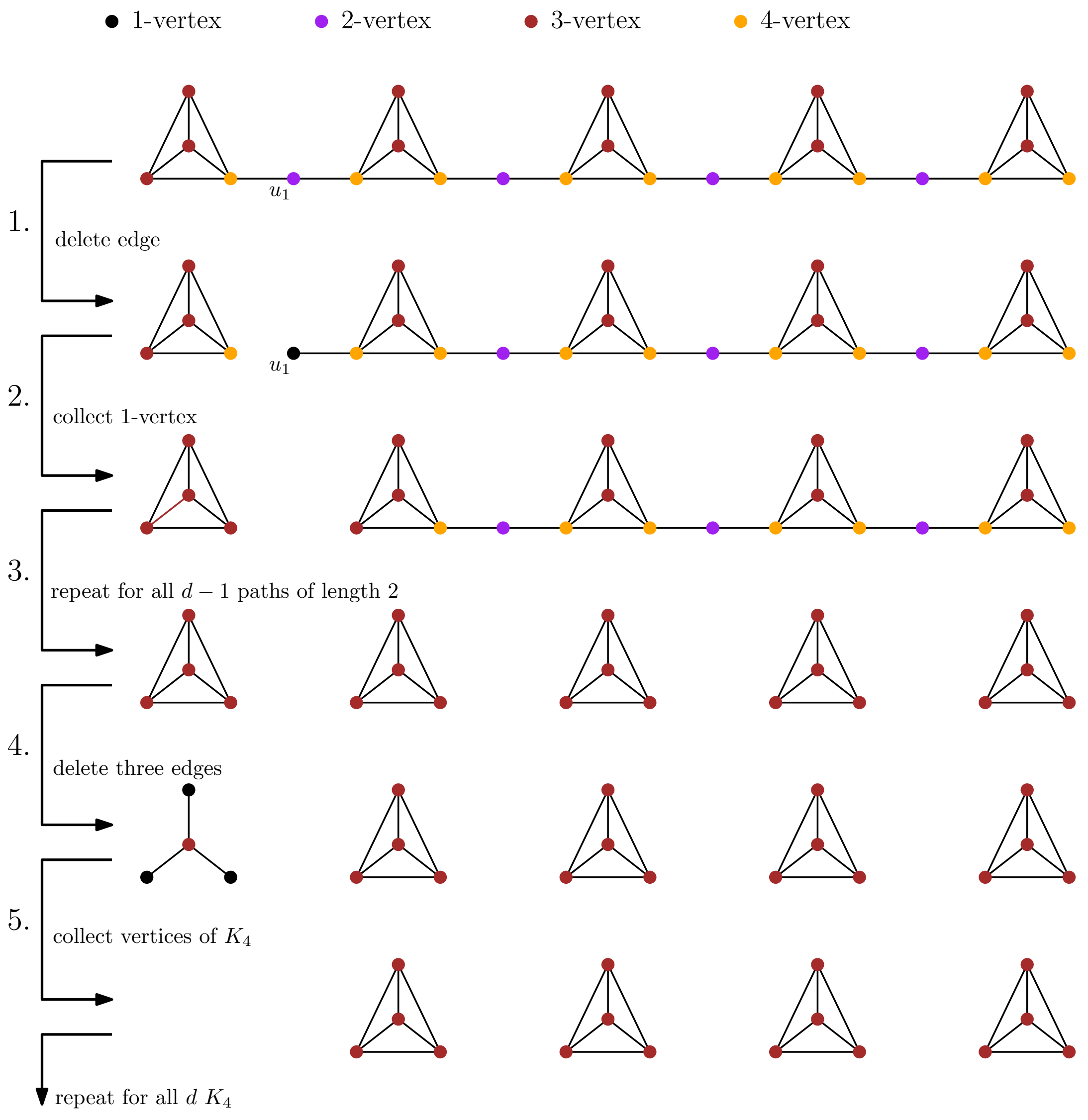

Figure 4.3.: The different steps of the greedy algorithm applied to the graph in Figure 4.2.

vertices according to their degree. First delete an edge incident to a 2-vertex. These are the vertices u_i for $1 \leq i < d$. For $i = 1$ this is shown in the first step in Figure 4.3. Then we collect u_1, which has now degree 1 (see second step in Figure 4.3). In this way we collect all vertices u_i first. What remains are 3-vertices in the d copies of K_4 graphs. These K_4 graphs are not longer connected to each other (see third step in Figure 4.3). To collect all vertices in a K_4 we need to delete three edges. As reference, see steps 4. and 5. in Figure 4.3. Therefore, if we use the greedy algorithm, we delete one edge for each u_i with $1 \leq i < d$. Because we have $d-1$ such vertices u_i we delete $d-1$ edges. Additionally, we delete three edges for every copy of a K_4. We have d such copies, so we delete $3 \cdot d$ edges. This results in a total of $4 \cdot d - 1$ edges that we delete if we use this algorithm. Thus, we delete $d-1$ edges more than necessary. This gives us a lower

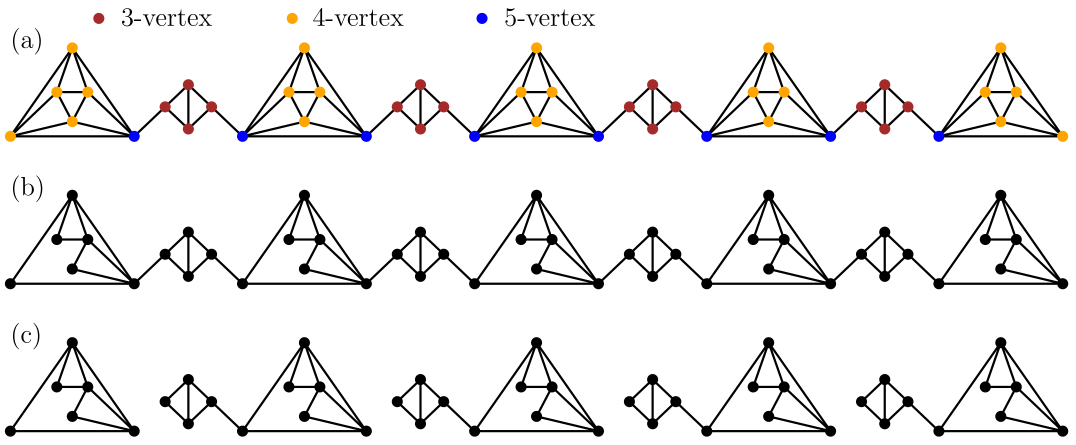

Figure 4.4.: (a) An example of a graph G where the greedy algorithm deletes more edges than necessary while generating a 2-degenerate subgraph.
(b) An example of a maximum subgraph of G that is 2-degenerate.
(c) A result of the greedy algorithm.

bound on the number of edges we delete more when using the greedy algorithm in the worst case. The graph in Figure 4.3 is a planar example. Thus, restricting the graph class we look at to planar graphs does not decrease the bound.

In the previous examples, we only considered 1-degenerate subgraphs. For a different k the number of edges deleted by the greedy algorithm is still larger than the k-degenerate skewness. An example for $k = 2$ is given in Figure 4.4. This is again an example of a planar graph. In Figure 4.4 (a) the original graph G is shown. It consists of $d = 5$ octahedra connected by $d - 1$ copies of the graph H. Graph H is defined as following. It has four vertices called v_1,v_2,v_3,v_4 that form a cycle. Additionally there is an edge between vertices v_2 and v_4. Vertex v_1 is then connected by an edge a copy of the octahedra and v_3 is connected by an edge to a different copy of the octahedra. We, again, color the vertices according to their degree. Note that G is not 2-degenerate. In Figure 4.4 (b) we see a maximum 2-degenerate subgraph of G. To get this subgraph we needed to delete $3 \cdot d$ edges. Figure 4.4 (c) shows a 2-degenerate subgraph of G that is a result of the greedy algorithm. Here, we delete $4 \cdot d - 1$ edges. So, again, we delete $d - 1$ edges more while using the greedy algorithm than necessary.

This type of examples also exist for k-degenerate skewness for $k > 2$ but we do not discuss it further in this thesis. It also would be interesting to study how much worse the greedy algorithm could be in comparison to an optimal algorithm. However, this was not done here either.

Still, we can use the greedy algorithm to prove an upper bound of $ds_k(G)$ for a graph G. Furthermore, we can look at different graph classes $\mathcal{C}$ and find upper bounds for $ds_k^{\mathcal{C}}(n)$. For bipartite planar graphs we find bounds in Section 5.2. For planar graphs we do this in Section 6.6 and Section 6.5.

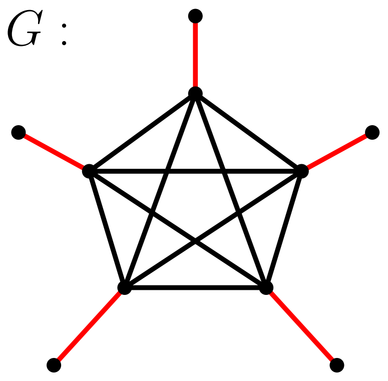

Figure 4.5.: A graph G with maximum matching in red. The graph G is 4-degenerate, as is the graph where the matching is deleted.

4.3. Matchings

One method to identify certain edges that should be deleted to reduce the degeneracy is based on matchings. A *matching* $M \subseteq E(G)$ is a set of edges of a graph G, such that no two edges in M share a vertex. A *maximum matching* M of a graph G is a matching with the maximum number of edges. In this section we discus whether deleting edges in a matching reduces the degeneracy of a graph.

By deleting edges in a matching, the degree of a vertex incident to a matching edge is reduced by one. The degree of vertices not incident to matching edges does not change. To assure that a graph is k-degenerate, every subgraph has to have a vertex of degree at most k. Now the question arises, whether it is enough for a $k+1$-degenerate graph G to delete a maximum matching, so that G becomes k-degenerate. One might think that in every subgraph of G the degree of the vertex with degree at most $k+1$ was reduced by one.

The example in Figure 4.5 contradicts this assumption. Here, we have a graph G that is 4-degenerate because K_5, the complete graph with five vertices, is a subgraph of G. The red edges show a matching of maximum cardinality. But even if we delete the edges in this matching, the K_5 is still a subgraph and the graph is still not 3-degenerate.

In this example the problem is a subgraph that is a complete graph. The next idea was to reduce the considered graph class. Concretely we only considered bipartite planar graphs and tried to prove the upper bound of $\frac{n}{2}$ for $ds_2^{\mathcal{B}}(n)$.

A bipartite planar graph G has a vertex of degree at most 3, because G is 3-degenerate. Let G have minimum degree 3. Then it is not 2-degenerate. If we delete one edge incident to a 3-vertex v it then has degree 2. Then G has a 2-vertex and chances are that it is 2-degenerate. Graph G is 2-degenerate if for every subgraph G' of G with minimum degree 3 we delete an edge incident to a 3-vertex in G'. Because we need to delete at most one edge incident to these vertices, we want to delete edges in a matching.

Again, we can give an example of a bipartite planar graph G and a maximum matching M that contradicts this assumption. In Figure 4.6 (left) the graph G and the matching M (in red) is pictured. The matching M consist of eight edges, while G has 16 vertices. A matching cannot be greater than $\frac{|V(G)|}{2}$. Thus M is a maximum matching.

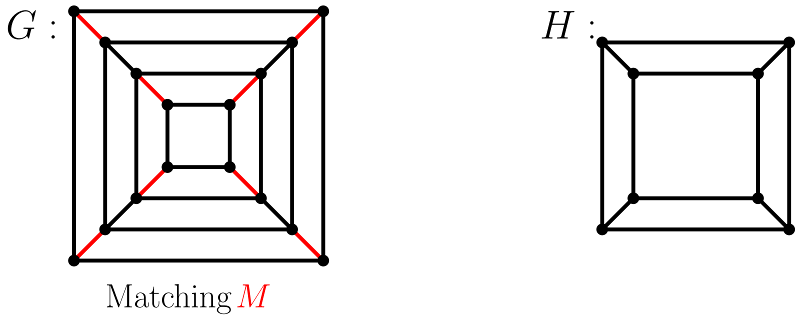

Figure 4.6.: The left graph G shows a bipartite planar graph with a maximum matching M in red. The right graph H is a subgraph of G without the edge of M. Graph H is still not 2-degenerate.

If we delete the edges of M we get a graph $G' = (V(G), E(G) \setminus M)$. This graph has H as subgraph, shown in Figure 4.6 (right). The subgraph H has minimum degree 3 and, thus, is not 2-degenerate. Therefore, G' is not 2-degenerate and deleting the edges in an arbitrary maximum matching is not enough.

We have not found a way to give further restrictions for a matching M, such that deleting the edges of M is enough to reduce the degeneracy by one.

5. 2-degenerate Skewness for Bipartite Planar Graphs

In this chapter, we look at bipartite planar graphs. We denote the class of those graphs with $\mathcal{B}$. Recall that a quadrangulation of a planar graph is an edge maximal bipartite planar graph, that means, no other edge can be added so that it is still bipartite and planar. Because of Lemma 2.5, we use quadrangulations to prove our statements. Observe that every bipartite planar graph can be augmented to a quadrangulation by simply adding edges. In a quadrangulation, every face is bounded by a cycle containing exactly four edges.

To make a quadrangulated graph G 1-degenerate we need to find a spanning tree again like in Section 4.1. There we prove that $ds_1(G) = |E(G)| - |V(G)| + 1$. Here our graph G with $|V(G)| = n$ vertices has only $|E(G)| = 2n - 4$ edges and thus only $2n - 4 - (n - 1) = n - 3$ edges need to be deleted. For quadrangulations this bound is sharp, because only forests are 1-degenerate. Because of Lemma 2.5 we know the following lemma is true.

Lemma 5.1: *Any bipartite planar graph G can be made 1-degenerate by deleting at most $n - 3$ edges, where $n = |V(G)|$. If G is quadrangulated, exactly $n - 3$ edges need to be deleted. Then $ds_1^{\mathcal{B}}(n) = n - 3$.*

In the Paragraph *Bipatite Planar Graphs* in Section 2.2.1, we show that every bipartite planar graph is 3-degenerate. Thus, it is left to find subgraphs that are 2-degenerate.

In Section 5.1 we prove the lower bound $\frac{n}{4} - 1$ for $ds_2^{\mathcal{B}}(n)$ to make a graph $G \in \mathcal{P}$ with $|V(G)| = n$ 3-degenerate. We do this by giving a construction of a family of graphs $\mathcal{F} \subset \mathcal{P}$. For every graph $G \in \mathcal{F}$ with $|V(G)| = n$ the 2-degenerate skewness is $ds_2(G) \leq ds_2^{\mathcal{P}}(n)$. In Section 5.2 and Section 5.3 we give upper bounds for $ds_2^{\mathcal{B}}(n)$ that use generic types of approaches like the greedy algorithm used in Section 6.6 and discharging, as described in Section 2.3. Then, in Section 5.4 we prove a stronger upper bound $ds_2^{\mathcal{B}}(n) \leq \frac{n}{2}$. At last, in Section 5.5 we justify the conjecture that $ds_2^{\mathcal{B}}(n) \leq \frac{n}{4}$.

5.1. Lower Bound

In this part, we give a lower bound on the number of edges that need to be deleted to make any bipartite planar graph 2-degenerate, i.e. $ds_2^{\mathcal{B}}(n)$. For this, we give the construction of a family of graphs, where a specific number of edges dependent on the number of vertices needs to be deleted to obtain a 2-degenerate graph. All graphs in this family are quadrangulations.

We define a family $\mathcal{F}$ by listing all graphs $G_i \in \mathcal{F}$ for $i \geq 2$. Graph G_2 is a 3-cube (or hexahedron), see Figure 5.1 left. This graph is bipartite and planar. The number of vertices and edges is given by $n_2 = |V(G_2)| = 2 \cdot 4 = 8$ and $m_2 = |E(G_2)| = 12 = 2n_2 - 4$. All faces are bounded by cycles of length 4. Now, a graph G_i can be constructed by taking G_{i-1} and adding for each of the four vertices on the cycle bounding the outer

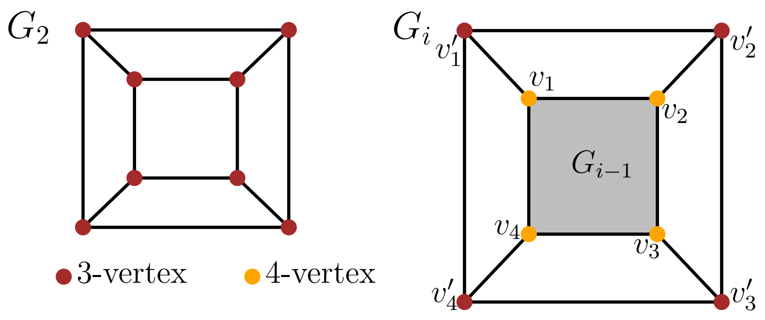

Figure 5.1.: Construction of G_2 (left) and G_i (right).

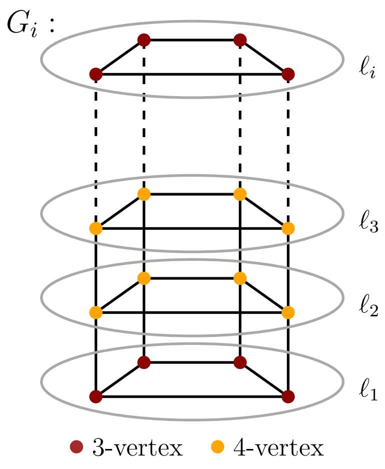

Figure 5.2.: Graph G_i split into i layers.

face v_1, v_2, v_3, v_4 a vertex v_1', v_2', v_3', v_4'. We add edges between vertices v_i and v_i' for $1 \leq i \leq 4$ and between v_i' and v_{i+1}' for $1 \leq i \leq 3$ and last between vertices v_4' and v_1'. For reference see Figure 5.1 right. Thus, we add four vertices and 8 edges in a way that the graph stays quadrangulated and we know $n_i = |V(G_i)| = n_{i-1} + 4 = i \cdot 4$ and $m_i = |E(G_i)| = m_{i-1} + 8 = 2n_{i-1} - 4 + 8 = 2n_i - 4$.

We prove that we need to delete at least $\frac{n}{4} - 1$ edges so that a graph $G \in \mathcal{F}$ with n vertices becomes 2-degenerate. That means for $G_i \in \mathcal{F}$ we need to delete $i - 1$ edges, for $2 \leq i$. First, we sort the vertices of G_i into i layers with four vertices per layer. In the first (lowest) layer ℓ_1 we put the vertices that are not on the cycle bounding the outer face of G_2. In layer ℓ_j we put the vertices that form the cycle bounding the outer face of G_j for $2 \leq j \leq i$. Layer ℓ_i is then the last (highest) layer. All vertices in layers ℓ_1 and ℓ_i have degree 3, vertices in the other layers have degree 4. An example is given in Figure 5.2. Therefore, the minimum degree of G_i is always 3, for all $2 \leq i$, and none of the graphs in $\mathcal{F}$ are 2-degenerate.

Next, we prove that we can make G_i 2-degenerate by deleting $i - 1$ edges.

Lemma 5.2: *For all $i \geq 2$ the graph $G_i \in \mathcal{F}$ can be made 2-degenerate by deleting $i - 1 = \frac{n}{4} - 1$ edges.*

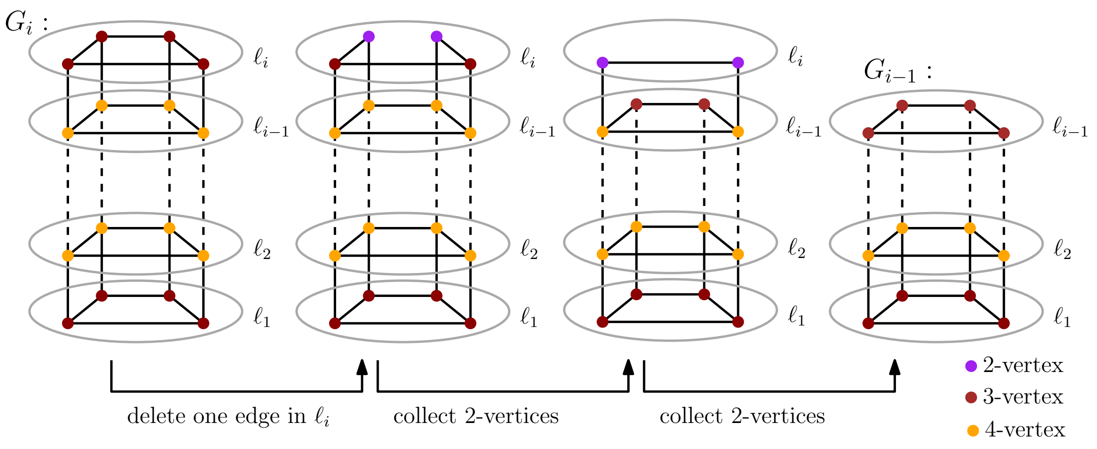

Figure 5.3.: Induction step. Reducing G_i to G_{i-1} by deleting one edge.

Proof. We prove this lemma by induction. First we take G_2. This graph is a 3-cube and it is 3-regular. If we delete any one edge, the remaining graph is 2-degenerate because every collected vertex reduces the degree of its neighbors by at least one which results in them having degree at most 2. Therefore, the statement is true for $i = 2$.

Now, let the lemma is true up to one $i \geq 3$. We next want to prove it is true for G_i as well. For reference look at Figure 5.3. We delete one edge between vertices that are both in the highest layer ℓ_i. Then, those vertices only have degree 2 and we can collect them. This results in the remaining two vertices in layer ℓ_i becoming 2-vertices that can be collected as well. Thus, by deleting one edge we can collect 4 vertices. The graph induced by the remaining vertices is G_{i-1} which can be reduced into a 2-degenerate graph by removing $(i-1)-1$ edges, by induction. Thus, G_i can be reduced to a 2-degenerate graph by deleting $i-1$ edges. □

Now, we show that this is the minimum number edges that need to be deleted.

Lemma 5.3: *For all $i \geq 2$ the graph $G_i \in \mathcal{F}$ is 3-degenerate and not 2-degenerate if only $i-2 = \frac{n}{4} - 2$ edges are deleted.*

Proof. The graph G_2 is 3-degenerate. If we delete $2-2=0$ edges, G_2 is still not 2-degenerate. Now assume that for each $2 \leq j < i$, the graph G_j is 3-degenerate when we delete $j-2$ edges and only becomes 2-degenerate when we delete $j-1$ edges.

We take graph G_i and delete $i-2$ edges. We distinguish what types of edges we deleted. First, let one of the deleted edges be incident to a vertex on layer ℓ_1 or layer ℓ_i. We denote this edge by e and the vertex incident to e on layer ℓ_1 or layer ℓ_i with v. Because of symmetry, we can assume without loss of generality that v is on layer ℓ_i. It is not important whether the second vertex incident to e is on layer ℓ_i or layer ℓ_{i-1}, see Figure 5.4 (a) and (b) respectively. In both cases, G_{i-1} is a subgraph of G_i without edge e. There are $j-3$ other edges that can be deleted if we want to delete only $i-2$ edges in total. By induction, we know that we can not make G_{i-1} 2-degenerate by only deleting $(i-1)-2$ edges. Thus, in this case G_i cannot be 2−*degenerate* by deleting $i-2$ edges.

Next let one of the deleted edges be incident to vertices on adjacent layers. We denote this edge with e and say it is incident to vertex u on layer ℓ_j and to vertex v on layer ℓ_{j+1} for $2 \leq j \leq i-2$. This case is shown in Figure 5.5. Again, we have $i-3$ edges left that

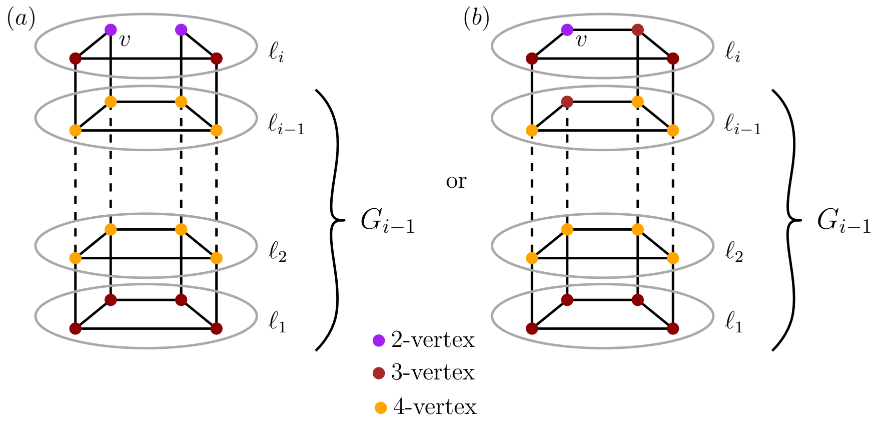

Figure 5.4.: First case in the proof of Lemma 5.3 in which one deleted edge, called e, is incident to vertex v in layer ℓ_i. In (a) e is incident to two vertices in layer ℓ_i. In (b) e is incident to only one vertex in layer ℓ_i. Graph G_{i-1} is in both cases a subgraph.

can be deleted. We split the graph in two parts. Graph G' consists of the layers ℓ_1 to ℓ_j and G'' consists of the layers ℓ_{j+1} to ℓ_i. Then $G' = G_j$ and $G'' = G_{i-j}$. By induction, we know that we need at least $j-1$ edges to make G' 2-degenerate and $i-j-1$ edges to make G'' 2-degenerate. That means we delete $(j-1)+(i-j-1) = i-2$ edges in G' and G'' combined. If we only delete $i-3$ edges in G' and G'' and e, then we delete $i-2$ edges, but G_i is still not 2-degenerate.

For the last case, let all $i-2$ deleted edges be between vertices on the same layer and none incident to a vertex on layer ℓ_1 or layer ℓ_i. If we delete one edge of every layer from ℓ_2 to ℓ_{i-1}, then every vertex still has degree at least 3. In Figure 5.6 (b) we show an example for $i = 5$. Thus, deleting $i-2$ edges in this way does not result in a 2-degenerate graph. Now say we delete more than one edge in one layer. An example for that, again for $i = 5$, is given in Figure 5.6 (c). For every layer in which we delete more than one edge, there is a layer in which we delete no edge. Thus, if we have p distinct layers in which we delete at least two edges, then there must be $q \geq p$ layers in which we delete no edge. Note that $q \geq p+2$ because we have i layers and delete only $i-2$ edges. Thus, we find a subgraph G' whose lowest and highest layer contain no deleted edge and in the layers in between exactly one edge per layer was deleted. In G' all vertices have degree at least 3. Therefore, G' is not 2-degenerate and, because $G' \subseteq G_i$, we know that G_i is not 2-degenerate. Thus, deleting $i-2$ edges in this way still does not give a 2-degenerate graph.

We deduce that no matter how we choose the $i-2$ edge we delete, G_i is not 2-degenerate, which concludes the proof. □

Because G_i has $|V(G_i)| = n = 4 \cdot i$ vertices and $ds_2(G_i) = i-1$, also $ds_3(G_i) = \frac{n}{4} - 1$. This proves the following lower bound for $ds_3^{\mathcal{P}}(n)$.

Theorem 5.4: *The maximum 2-degenerate skewness for bipartite planar graphs is at least $ds_2^{\mathcal{B}}(n) \geq \frac{n}{4} - 1$.*

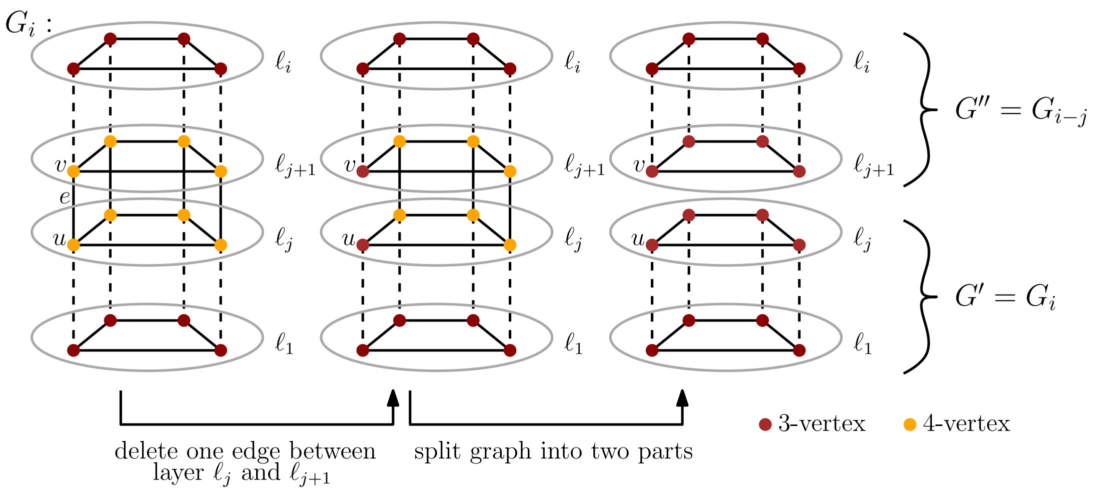

Figure 5.5.: Second case in the proof of Lemma 5.3 in which one deleted edge is between two adjacent layers.

5.2. Upper Bound using Greedy Algorithm

To give a first upper bound for $ds_2^{\mathcal{B}}(n)$, we use the greedy algorithm from Section 4.2.

Lemma 5.5: *To make a bipartite planar graph G 2-degenerate, at most $\frac{2}{3}n$ edges need to be deleted, where $n = |V(G)|$.*

Proof. The greedy algorithm always selects the vertex with minimum degree. We now define a collecting ordering σ by always collecting the vertex of minimum degree. Let $\overline{\deg}(v)$ denote the degree of a vertex v when it is identified as the vertex with minimum degree and next in σ. To make G 2-degenerate we always need to collect vertices of degree at most 2 and thus we delete edges incident to v if $\overline{\deg}(v) > 2$ before it is collected. It is well known that in every bipartite planar graph there is a vertex of degree at most 3. Thus, we need to delete at most one edge to make to reduce the degree to 2. There can be at most $\frac{2}{3}$ vertices, that have $\overline{\deg}(v) > 2$. Otherwise $|E(G)| \geq 3 \cdot (\frac{2}{3}n) = 2n > 2n - 4$ which is a contradiction to the fact that bipartite planar graphs have at most $2n - 4$ edges. Thus, $ds_2^{\mathcal{B}}(n)$ is bounded by $\frac{2}{3}n$. □

5.3. Upper Bound using Discharging

We can give the same upper bound by using the discharging method described in Section 2.3. First, we give a local configuration that has to exist in every quadrangulation. Afterwards, we give a procedure for finding edges that need to be deleted to obtain a 2-degenerate graph. We prove that the obtained subgraph is 2-degenerate by giving an implicit collecting ordering with maximum degree 2. Because of Corollary 2.2, the subgraph is 2-degenerate. We furthermore prove that we did not delete more than $\frac{2}{3}n$ edges for a graph with n vertices.

In the following lemma we prove the existence of certain local configurations. These are also shown in Figure 5.7.

Lemma 5.6: *Every quadrangulation G with minimum degree 3 has a 7^--edge or a 5-vertex with at least four 3-vertices as neighbors.*

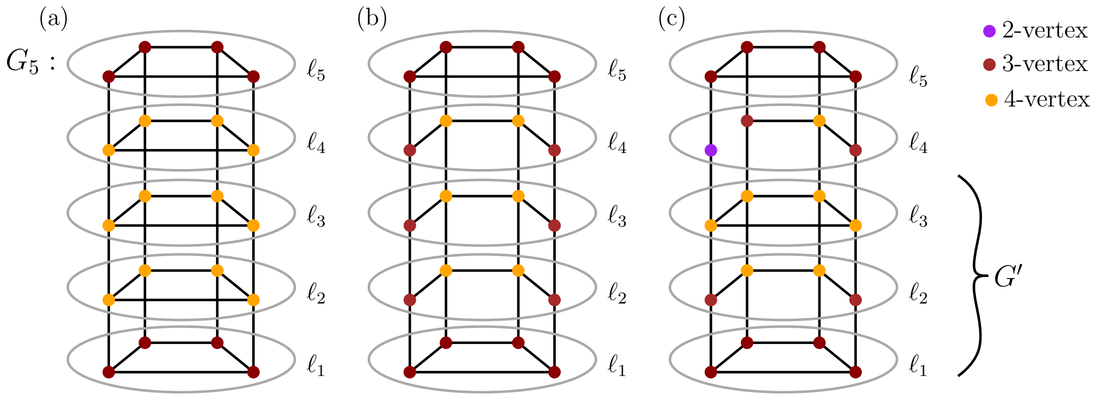

Figure 5.6.: Last case in the proof of Lemma 5.3 in which all $i-2$ deleted edges are between vertices of the same layer. In (a) the graph G_5 is pictured. (b) is an example, where one edge per layer is deleted. (c) is an example, where for some layers more than one edge per layer is deleted. Here subgraph G' is shown as well.

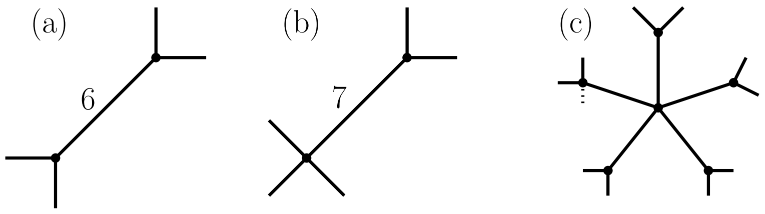

Figure 5.7.: Local configurations where at least one exists in every quadrangulation

Proof. Assume that this is not true. That means, that no edge with weight at most 7 exists and every 5-vertex has at most three 3-vertices. Now, assign every vertex v a charge $c(v) = \deg(v)$. Similar to Chapter 3 in [CW17], we say a vertex v is *happy* if and only if it has charge $c(v) \geq 4$. Because G has minimum degree 3, only 3-vertices are not happy.

Next, we redistribute the charge, to make every vertex happy. Every 3-vertex takes $\frac{1}{3}$ charge from every neighbor. Observe, that 3-vertices do not have neighbors that have degree at most 4, because otherwise there is an edge with weight at most 7. If the neighbor of a 3-vertex is a 5-vertex u, then u has at most three 3-vertices as neighbors in total. That means, it looses at most charge $3 \cdot \frac{1}{3} = 1$, which results in a charge $c(u) = 4$. Thus, the 5-vertex remains happy. If the neighbor of a 3-vertex is a 6^+-vertex u, then it looses at most $\deg(u) \cdot \frac{1}{3}$ charge, and with $\deg(u) \geq 6$, this means that u keeps at least $\deg(u) \cdot \frac{2}{3} \geq 4$ charge and u is still happy.

Therefore, it is possible to redistribute the charge, to make every vertex happy. In Proposition 3.1. in [CW17] the authors proved the *balanced charging equation*:

$$\sum_{v \in V(G)} (\deg(v) - 4) + \sum_{f \in F(G)} (l(F) - 4) = -8$$

For quadrangulations every face has length 4 and this equation is reduced to

$$\sum_{v \in V(G)} (\deg(v) - 4) = -8$$

Because we defined a vertex to be happy if and only if it has charge at least 4, every vertex that is happy contributes a positive amount to the sum and every vertex that is not happy a negative amount. We defined the charge of a vertex as its degree and redistributing the charge did not change the the value of the sum in total. Because every vertex is now happy, only positive amounts are added to the left side of the equation. This is a contradiction to the balanced charging equation, because the right side is negative, which the left side is not. Thus, the assumption that the theorem is wrong is false. □

If graph G is a bipartite planar graph, but not necessarily a quadrangulation, then for a face $f \in F(G)$ the length of f is always $l(f) \geq 4$. Thus, in the charging equation the right summand is always positive. Therefore Theorem 5.6 is true for all bipartite planar graphs.

Now, given a bipartite planar graph G, we give x steps in which we collect at least one vertex and afterward the graph is empty. In each step we have to pay a certain amount for the edges that need to be deleted, so that the vertices in this step can be collected. There are three types of steps.

In the first type, we simply collect all vertices of degree at most 2. We do not need to pay anything for this step, because we do not need to delete any edge. Thus, the payment for steps of the first type is $p_1 = 0$. The number of times we use this step is denoted by x_1.

In the second type, we take a 3-vertex v incident to a 7^{-}-edge and collect v. To collect v, we first have to delete an edge incident to v so that v gets degree 2. Thus, we need to pay $p_2 = 1$ for steps of the second type. We denote the number of times we use this step by x_2. Every time we use a step of the second type, we reduce the number of edges by 3.

In the third type, we collect a 5-vertex v that has at least four neighbors of degree 3 and these neighbors. To collect v we need to delete three edges incident to v, thus the payment for this step is $p_3 = 3$. After we collect v, the degree of neighbors of v is reduced by one and the four neighbors of degree 3 have now degree 2 and can be collected. The number of times we use this step is denoted by x_3. With this step we reduce the number of edges by thirteen.

The following equations hold:

$$\begin{aligned} x_1 + x_2 + x_3 &= x \\ x &\leq n \\ 3x_2 + 13x_3 &\leq 2n - 4 \end{aligned}$$

To give an upper bound on $ds_2^{\mathcal{B}}(n)$, we need the maximal amount payed, when taking the x steps. Observe, that we can always use one of these steps because of Theorem 5.6. The total payed amount is

$$p = p_1 \cdot x_1 + p_2 \cdot x_2 + p_3 \cdot x_3 = x_2 + 3x_3$$

We now have a Linear Problem dependent on n. The payment p is less than $\frac{2}{3}n$. This is true because

$$\begin{aligned} p &= x_2 + 3x_3 && \Leftrightarrow \\ 3p &= 3x_2 + 9x_3 \leq 3x_2 + 13x_3 \leq 2n - 4 && \Leftrightarrow \\ p &\leq \frac{2}{3}n - \frac{4}{3} \end{aligned}$$

Now we have found a way, to make any bipartite planar graph 2-degenerate by deleting less than $\frac{2}{3}n$ edges an thus $ds_2^{\mathcal{B}}(n) < \frac{2}{3}n$.

5.4. Upper Bound

In this section, we use a different method to obtain an upper bound for $ds_2^{\mathcal{B}}(n)$. This method is specifically for 2-degenerate subgraphs of bipartite planar graphs and results in a stronger bound. We prove that every bipartite planar graph G needs to delete at most $\frac{n}{2}$ edges to become 2-degenerate, with $n = |V(G)|$. To show this we first prove the following lemma.

Lemma 5.7: *Given a bipartite planar graph G with minimum degree 3, there exists a vertex v in G, such that $\deg(v) = 3$ and v is incident to three pairwise distinct faces.*

Proof. Let G be connected. Otherwise, we conduct the proof on each of the components. We distinguish two different cases based on the connectivity of G.

Case 1: G is at least 2-connected. Then, we know that every face is bounded by a cycle. We also know that there exists a vertex v of degree 3 because our minimum degree is 3 and every bipartite planar graph is 3-degenerate. But then, it is clear that v is incident to exactly three pairwise distinct faces because, otherwise, those faces would not be bounded by cycles.

Case 2: G is 1-connected. First, we form a block-cut-tree H of G, where every block represents a maximal 2-connected component of G. Two blocks are adjacent if there is one vertex in each block such that the two vertices are adjacent. We now choose an arbitrary leaf of H. This leaf C is a 2-connected component with at least three vertices. Otherwise, we would not have minimum degree 3 in G.

If there is a vertex v in C that has $\deg(v) = 3$ and where every neighbor of v is in C, then we can use Case 1 on C and we are done. This case is illustrated in Figure 5.8 (left). Otherwise, the only vertex of degree 3 is the vertex u, that has one neighbor outside of C. This is shown in Figure 5.8 (right). We know that this vertex exists because G is 3-degenerate and, thus, C is 3-degenerate as well. The vertex u is not incident to three different faces but only to two. Therefore, we want to prove that u cannot be the only 3-vertex in C. Assume that it is. Then every other vertex in C has degree at least 4 (because the minimum degree of G is 3). If we only look at C, then u has degree 2. That results in the following approximation for the number of edges in C:

$$|E(C)| \geq \frac{(|V(C)| - 1) \cdot 4 + 2}{2} = \frac{4 \cdot |V(C)| - 2}{2} = 2|V(C)| - 1$$

But we know that C is still bipartite planar and thus $|E(C)| \leq 2 \cdot |V(C)| - 4$. This leads to a contradiction and we know that u can not be the only 3-vertex in C. Therefore, there is at least one other 3-vertex in C satisfying the lemma. ▢

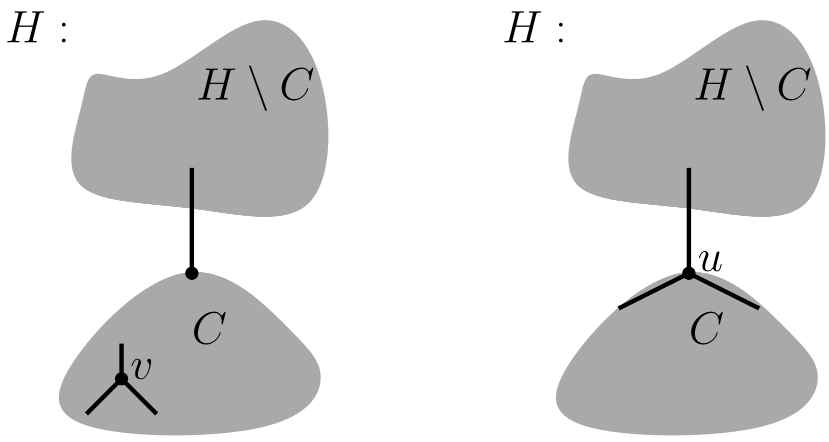

Figure 5.8.: Component C in bock-cut-tree H and two cases for 3-vertices in C.

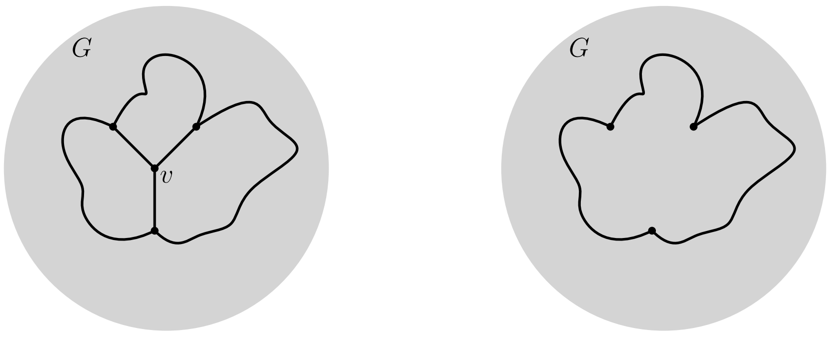

Figure 5.9.: The left graph shows 3-vertex v incident to three distinct faces.
The right graph shows the graph after one edge is deleted and v is collected.

Now we can formulate and prove the actual theorem.

Theorem 5.8: *To make a bipartite planar graph G 2-degenerate, one has to delete at most $\frac{n-2}{2}$ edges, where n is the number of vertices in G. That means $ds_2^{\mathcal{B}}(n) \leq \frac{n-2}{2}$.*

Proof. We want to find a collecting ordering with maximum degree 2. We can collect vertices of degree 2 without problem. If we only have vertices of degree 3 we need to delete an edge incident to a 3-vertex v prior to collecting v. Thus say, that G has minimum degree 3. We know that $|F(G)| = f \leq n-2$ for bipartite planar graphs. (See Equation 2.1.)

Now, take a vertex v with $\deg(v) = 3$ that is incident to three pairwise distinct faces. See Figure 5.9 (left). We know that such a vertex exists because of Lemma 5.7. Next we delete one edge incident to v, and then collect v. The result is shown in Figure 5.9 (right). We go on and collect as many vertices as possible in the resulting graph. Because v was incident to three pairwise distinct faces, the collection of v results in merging those three faces into one and the number of faces goes down by at least two. (It could go down by more than two if other vertices can be collected.) Thus every deletion of an edge results in the reduction of the number of faces by two. Then, at most $\frac{n-2}{2}$ edges need to be deleted to result in a graph that has only one face. This graph is a forest and forests are 1-degenerate. Therefore, by deleting $\frac{n-2}{2}$ edges, we get a 2-degenerate subgraph of G. □

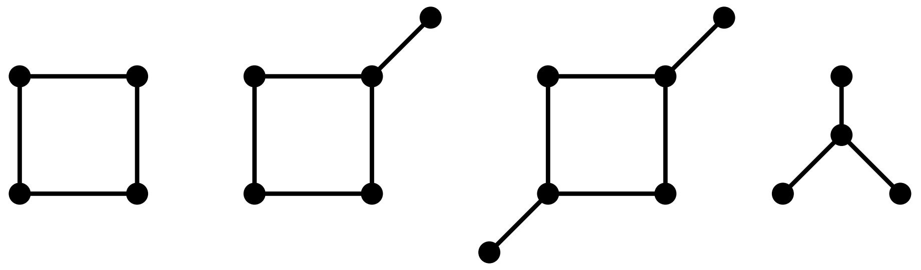
Figure 5.10.: Possible components of the subgraph induced by the vertices of degree 3.

5.5. Conjecture for Stonger Upper Bound

In 2015 Brinkmann et al. published their paper "Generation of simple quadrangulations of the sphere" [Bri+05]. This paper aims to give an understanding what types of bipartite planar graphs exists. Among other things they prove the following lemma (Lemma 2 in Preliminary observations).

Lemma 5.9: *[[Bri+05]] Let G be a simple quadrangulation with minimum degree 3 and let H be a component of the subgraph induced by the vertices of degree 3. Then H is one of the following graphs:*

1. *a cycle of even length at least 8, in which case G is a pseudo-double wheel;*
2. *a path (possibly of a single vertex);*
3. *a cube,in which case $G = H$;*
4. *one of the four graphs of Figure 5.10.*

We use this lemma to give evidence to the following conjecture.

Conjecture 5.10: *For the graph class $\mathcal{B}$ consisting of the bipartite planar graphs, we have $ds_2^{\mathcal{B}}(n) \leq \frac{n}{4}$.*

Lemma 2.5 gives us that we only need to look at quadrangulations. Let G be a quadrangulation. The goal is to find an algorithm that produces a collecting ordering with maximum degree 2. It consists of multiple iterations. In each iteration we collect vertices that have degree at most 2 when they are collected. Before the vertices can be collected, it may happen that edges need to be deleted so that the degree of the vertices is reduced to 2. At the end of an iteration we add edges such that the graph is again a quadrangulation. We want to find an algorithm that collects at least four vertices and deletes at most one edge in each iteration. Then the number of deleted edges is at most $\frac{n}{4}$. Let G' be the graph G after the edges that are deleted in all iterations are removed. Note that edges and vertices that are collected remain in G'. Then G' is 2-degenerate. This holds because we found a collecting ordering with maximum degree 2 and Corollary 2.2 gives us the rest. The number of edges added is not important. We only add edges to simplify the proof. Without the added edges, the degree of vertices is only reduced further. This does not contradict the fact that G' is 2-degenerate.

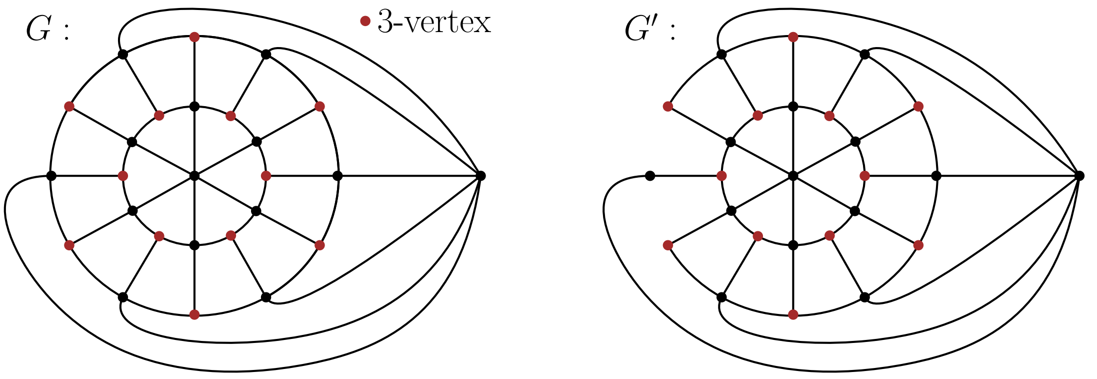

Figure 5.11.: The left graph G shows quadrangulation with minimum degree 3 where the 3-vertices form an independent set. In the right graph G' two edges of G where deleted. Graph G' is 2-degenerate.

We now need to show that, if we need to delete an edge in one iteration, we can collect at least four vertices. If G has degree 2 vertices, these can be collected without deleting an edge. So let G have minimum degree 3. Then no vertex can be collected. Lemma 5.9 states that one of the described four cases occurs. In Cases 1,3 and 4 we can collect at least four vertices by deleting one edge. This is also true if in Case 2 we have a path that consist of at least four vertices. That means that, if we find such subgraphs in G for every iteration, Conjecture 5.10 is true.

The problem with this approach is that it is possible that all components of the subgraph of G induced by vertices of degree 3 are paths of length less than 4. In this case, we need to delete an edge to collect a vertex, but we can not necessarily collect four vertices. There are quadrangulations where this case occurs. The graph G in Figure 5.11 (left) is such an example of this. Here the 3-vertices form an independent set, i.e. each component of the subgraph induced by all 3-vertices consists of a single vertex. Now, when we delete one edge we can only collect one vertex.

We still assume that $ds_2^{\mathcal{B}}(n) \leq \frac{n}{2}$. In the previous algorithm we add edges so that we always have a quadrangulation at the end of each iteration. These added edges are only important for using Lemma 5.9. If we do not add those edges, the degree of some vertices is reduced further. It is then possible that we can collect other vertices without needing to delete more edges. In the graph G in Figure 5.11 (left) for example, it is enough to delete two edges in total. The graph G' in Figure 5.11 (right) is 2-degenerate.

Next, we look at a quadrangulation G that consists of 3-vertices and 6-vertices. Again, the 3-vertices form an independent set. In Figure 5.12 we show a part of this graph. If this pattern is repeated infinitely, then G could be a counterexample to Conjecture 5.10. But because G is finite and every quadrangulation has average degree $4 \cdot |V(G)| - 8$, we know that at the boundary of this graph we have vertices of smaller degree. This again leads us to assume that Conjecture 5.10 is true.

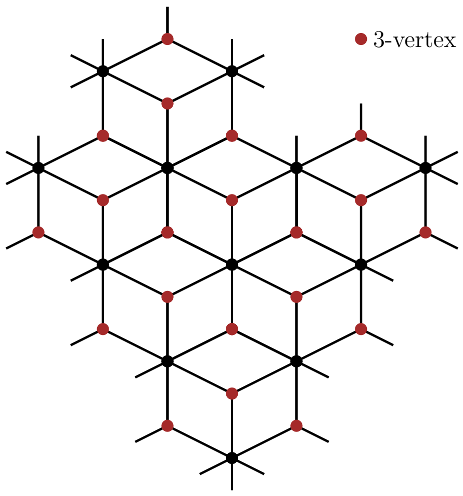

Figure 5.12.: A part of graph G, consisting of 3-vertices and 6-vertices where the 3-vertices form an independent set.

6. k-degenerate Skewness for Planar Graphs

In this Chapter we look at planar graphs. We call the class of planar graphs $\mathcal{P}$. In Section 2.2.1, we have already shown that every planar graph is 5-degenerate. Here, we want to find a way to delete as few edges as possible of a planar graph G such that the obtained graph is k-degenerate for $1 \leq k < 5$. For G to become 0-degenerate, we have to delete every edge. In Section 6.1 and Section 6.2, we introduce procedures to make a planar graph G 1-degenerate and 2-degenerate, respectively. For these cases, it is possible to give an exact value for $ds_1^{\mathcal{P}}(n)$ and $ds_2^{\mathcal{P}}(n)$. In Section 6.3 give a lower bound for $ds_3^{\mathcal{P}}(n)$ by using a similar approach as in Section 5.1. In Section 6.4, we then give an even stronger lower bound for $ds_3^{\mathcal{P}}(n)$. We cannot yet give an upper bound for $ds_3^{\mathcal{P}}(n)$, other than the bound for $ds_2^{\mathcal{P}}(n)$. In Section 6.6, we give an upper bound for $ds_4^{\mathcal{P}}(n)$ by using a greedy algorithm. Last, we show in Section 6.5 that by using the same greedy algorithm to determine $ds_3^{\mathcal{P}}(n)$, we get a higher bound than $ds_2^{\mathcal{P}}(n)$. This means using the greedy algorithm results in an unnecessarily high number of edges, that are deleted.

Observe that if we take a graph $G \in \mathcal{P}$ and its triangulation G', then $ds_k(G)$ is always at most $ds_k(G')$. A triangulation is an edge maximal graph in $\mathcal{P}$. Because of Lemma 2.5, we may restrict the graphs we consider to triangulated graphs to get $ds_k^{\mathcal{P}}(n)$.

6.1. 1-degenerate Skewness

We begin finding maximum 1-degenerate subgraphs in planar graphs. In Section 4.1, we show that for every graph G the degeneracy skewness is $ds_1(G) = |E(G)| - |V(G)| + 1$. Now, take a triangulated graph G. The graph G has $|V(G)| = n$ vertices and $|E(G)| = 3n - 6$ edges. Therefore exactly $3n - 6 - (n - 1) = 2n - 5$ edges need to be deleted. Because of Lemma 2.5, to find $ds_1^{\mathcal{P}}(n)$, we only need to look at triangulated graphs. Thus, we can formulated the following lemma.

Lemma 6.1: *Any planar graph G can be made 1-degenerate by deleting at most $2n - 5$ edges, where $n = |V(G)|$. If G is triangulated, exactly $2n - 5$ edges need to be deleted. Then, $ds_1^{\mathcal{P}}(n) = 2n - 5$.*

6.2. 2-degenerate Skewness

We show that to make any planar graph 2-degenerate, we need to delete at most $n - 3$ edges.

To prove this statement, we use a canonical ordering. De Fraysseix, Pach and Pollack first constructed a method to find such an ordering $\pi = (v_1, \dots, v_n)$ of the vertices of a graph G in [DPP90]. The graph G_i is the graph induced by vertices $\{v_1, \dots, v_i\}$ and C_i as the bounding cycle that forms the external face of G_i. This ordering π is called a *canonical ordering* if for each i with $3 \leq i \leq n$:

C1: G_i is biconnected and internally triangulated,

C2: C_i contains the edge (v_1, v_2),

C3: For $i < n$, vertex v_{i+1} lies in the outer face of G_k and all neighbors of v_{i+1} in G_i appear on the cycle C_i consecutively.

De Fraysseix et al. then went on to prove, that every triangulated graph has such a canonical ordering and that it can by computed in space. Later Goos Kant proved that this is also possible in linear time [Kan96]. The canonical ordering was used to construct an algorithm that gives a planar straight-line drawing of size $(2n-4) \times (n-2)$ for a planar graph with n vertices [DPP90].

The following definitions will help prove the next theorem. For each i with $3 \leq i \leq n$ we know that the neighbors of v_i appear consecutively on C_{i-1}. Let $v_{i,l}$ be the leftmost neighbor of v_i, i.e. the neighbor with minimal distance to v_1 on C_{i-1}, without using edge (v_1, v_2). Also let $v_{i,r}$ be the rightmost neighbor of v_i, i.e. the neighbor with minimal distance to v_2 on C_{i-1}, without using edge (v_1, v_2). Observe that for all neighbors of v_i, the vertices $v_{i,l}$ and $v_{i,r}$ exist. Also note that these are the only neighbors that are on C_i and that could appear on previous bounding cycles C_m for $i \leq m \leq n$. All other neighbors can only be on later bounding cycles C_m for $m < i$. For every vertex $v \neq v_n$ we define a *parent edge*. Let i be the biggest integer with $3 \leq i < n$ such that v is on the bounding cycle C_i but not on C_{i+1}. Then v is adjacent to v_{i+1}. The parent edge of v is then the edge connecting v to v_{i+1}. Observe that this edge always exists and is uniquely defined for v.

We now use the canonical ordering to prove the following lemma.

Lemma 6.2: *For the 2-degenerate skewness of planar graphs we can show $ds_2^{\mathcal{P}}(n) \leq n-3$.*

Proof. Let $G \in \mathcal{P}$ be a triangulated graph with n vertices. Given a canonical ordering $\pi = (v_1, \dots, v_n)$ of the vertices of G, we consider vertices in reverse order σ. We use σ as a collecting ordering of G. Let k be the maximum degree of σ. If $k > 2$ we delete edges such that the maximum degree is reduced to 2. For the obtained graph σ then has maximum degree 2 and Corollary 2.2 gives us that is is 2-degenerate.

We first take v_n, delete certain edges such that it has degree 2 and then we collect v_n. Then, we go on to the vertex v_{n-1} and repeat the process until we collect v_3.

We now define which edges we delete. For each i with $3 \leq i \leq n$, the neighbors of v_i appear consecutively on C_{i-1}. We delete the edges between v_i and all its neighbors in C_{i-1} except for $v_{i,l}$ and $v_{i,r}$. That means, for every u on C_{i-1} that is neighbor of v_1 and not $v_{i,l}$ and $v_{i,r}$, we delete the parent edge of u. This process is illustrated in Figure 6.1.

We stop when we have only vertices v_1, v_2 and v_3 left. These three form a triangle and G_3 is 2-degenerate. Thus, we delete $n-3$ edges in total and obtain a 2-degenerate graph and $ds_2^{\mathcal{P}}(n) \leq n-3$ □

Next we prove that for planar graphs $ds_2^{\mathcal{P}}(n) = n-3$ by proving that $ds_2^{\mathcal{P}}(n)$ is also at least $n-3$.

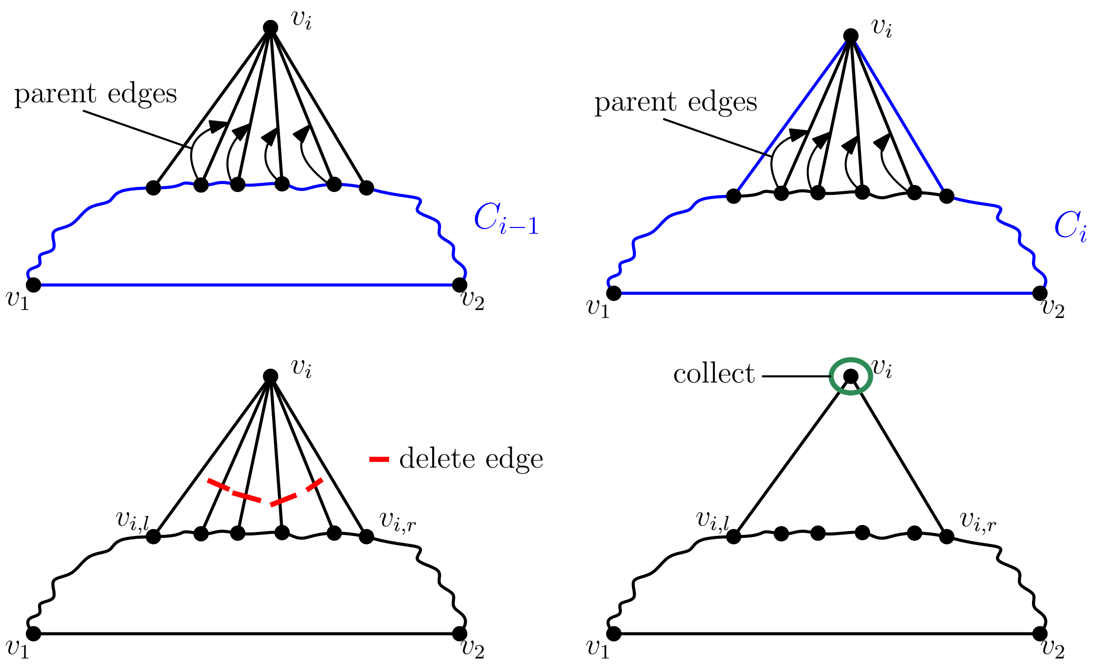

Figure 6.1.: Graph, when collecting v_i

Lemma 6.3: *For the* 2*-degenerate skewness of planar graphs we can show* $ds_2^{\mathcal{P}}(n) \geq n-3$.

Proof. Take a triangulated graph $G \in \mathcal{P}$ with n vertices. For a graph G' with $n' = |V(G')| \geq k$ to be k-degenerate, the number of edges is bounded by $m' = |E(G')| \leq kn - \binom{k+1}{2}$. This was proved by Don R. Lick and Arthur T. White in [LW70] in Proposition 3.

We know that the number of edges in every triangulated graph is $m = |E(G)| = 3n-6$, where $n = |V(G)|$. Now we insert $k = 2$ into the formula and get that $m' \leq 2n - \binom{3}{2} = 2n-3$. That means $m - m' = 3n - 6 - (2n-3) = n-3$ edges have to be deleted for the graph to become 2-degenerate. This means $ds_2(G) \geq n-3$ and thus $ds_2^{\mathcal{P}}(n) \geq n-3$. □

If we combine Lemma 6.2 and Lemma 6.2 we get the following theorem.

Theorem 6.4: *For planar graphs the maximum* 2*-degenerate skewness is* $ds_2^{\mathcal{P}}(n) = n-3$.

6.3. Lower Bound for 3-degenerate Skewness

In this section, we prove the lower bound $ds_3^{\mathcal{P}}(n) \geq \frac{n}{4} - 1$ where $\mathcal{P}$ again denotes the planar graph class. To prove this lower bound, we give a family of certain planar graphs in which each graph G has $ds_3(G) = \frac{|V(G)|}{4} - 1$. The construction of this family and the proof of the 3-degenerate skewness is similar to the proof of the lower bound of $ds_2^{\mathcal{B}}(n)$ in Section 5.1.

First, we define a family $\mathcal{F}' \subset \mathcal{P}$ of graphs. For this we take the family $\mathcal{F}$ from Section 5.1. Recall, Figure 5.1 shows the graphs G_2 and G_i. We also use the definition of layers from that section. In Figure 5.2, a layered drawing of the graph G_i is shown. For every graph $G_i \in \mathcal{F}$ with $i \geq 2$, we create a graph $G_i' \in \mathcal{F}'$ by adding $(i-1) \cdot 4$ edges in the following way: We add four edges between every two adjacent layers ℓ_j and

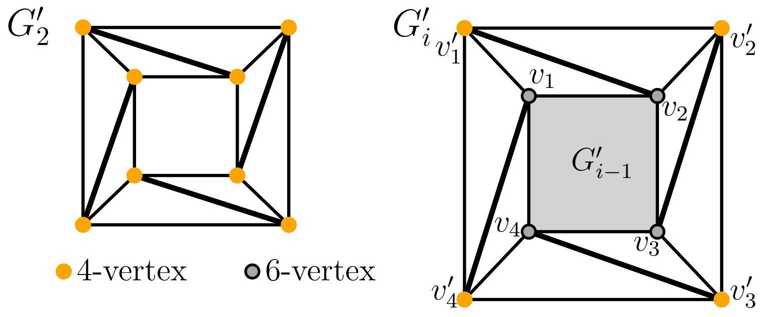

Figure 6.2.: Construction of G'_2 (left) and G'_i (right).

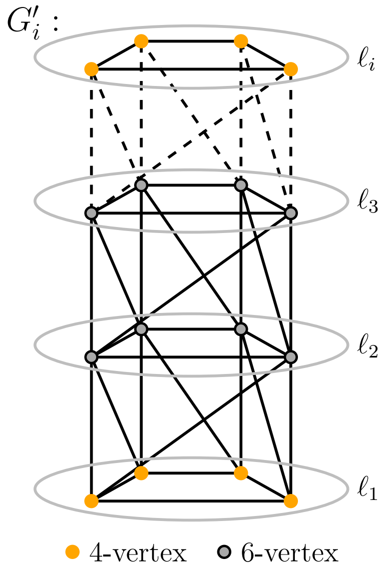

Figure 6.3.: Graph G'_i split into i layers.

ℓ_{j+1} for $1 \leq j \leq i-1$. Let v_1, v_2, v_3, v_4 denote the vertices in layer ℓ_j and v'_1, v'_2, v'_3, v'_4 denote the vertices in layer ℓ_{j+1} in cyclic clockwise ordering. Then, we add an edge between vertices v_i and v'_{i-1} for $1 < i \leq 4$ and between v_1 and v'_4. The edges are added in such a way that the face incident to vertices of layer ℓ_1 is still bounded by a cycle of length 4. The same holds for the vertices of layer ℓ_i. Every other face of G_i is divided into two faces bounded by a cycle of length 3 in G'_i. Also observe that we add the edges so that in G'_i all vertices in layer ℓ_1 and layer ℓ_i have degree 4 and degree 6 in the other layers. Figure 6.2 shows the graphs G'_2 and G'_i. The edges added to G_2 and G_i to create G'_2 and G'_i are shown in bold. Figure 6.3 shows a layered drawing of the graph G'_i.

Now, we show that $ds_3(G'_i) = i - 1$ for $i \geq 2$. First, we prove that we need to delete at most $i-1$ edges.

Lemma 6.5: *For all $i \geq 2$ the graph $G'_i \in \mathcal{F}'$ can be made 3-degenerate by deleting $i-1$ edges.*

Proof. This proof is analogous to the proof of Lemma 5.2. Thus, we again prove this lemma by induction. Graph G'_2 is 4-regular and by deleting one arbitrary edge G'_2 becomes 3-degenerate. Now let the lemma be true up to some $i \geq 3$. We next prove it is true for G'_i as well. Similar to the induction step of Lemma 5.2, after deleting

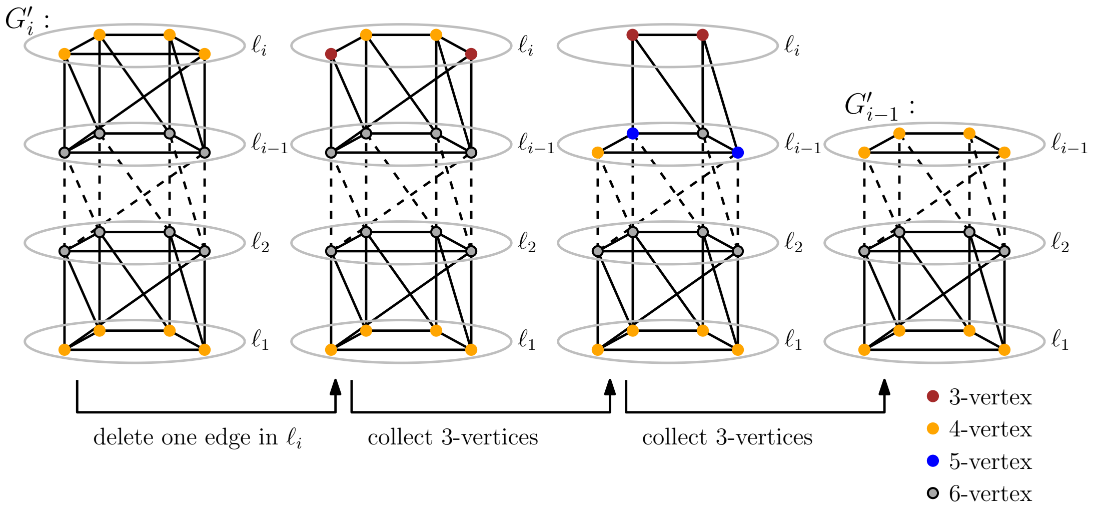

Figure 6.4.: Induction step. Reducing G'_i to G'_{i-1} by deleting one edge.

one edge incident to a vertex of layer ℓ_i, we can collect all vertices in layer ℓ_i. For reference look at Figure 6.4. The remaining graph is G'_{i-1} which can be reduced into a 3-degenerate graph by removing $(i-1)-1$ edges by induction. Thus G'_i can be reduced to a 3-degenerate graph by deleting $i-1$ edges. □

Now it remains to prove that we need to delete at least $i-1$ edges.

Lemma 6.6: *For all $i \geq 2$ the graph $G'_i \in \mathcal{F}'$ is 4-degenerate and not 3-degenerate if at most $i-2$ edges are deleted.*

Similarly to Lemma 6.5, we can adapt the proof of Lemma 5.3. The only difference is that we now use graphs $G'_i \in \mathcal{F}'$ and want to obtain 3-degenerate graphs.

Because G'_i has $|V(G'_i)| = n = 4 \cdot i$ vertices and $ds_3(G'_i) = i-1$, also $ds_3(G'_i) = \frac{n}{4} - 1$. This proves the lower bound for $ds_3^{\mathcal{P}}(n)$.

6.4. Stronger Lower Bound for 3-degenerate Skewness

In Section 5.5 we stated the conjecture that we need to delete at most $\frac{n}{4}$ edges of a bipartite planar graph to obtain a 2-degenerate graph. We can not state a similar conjecture for the 3-degenerate skewness of planar graphs. Instead we can raise the lower bound for $ds_3^{\mathcal{P}}(n)$ even higher than $\frac{n}{4}$. The graph G in Figure 6.5 has 27 vertices, but we need to delete at least seven edges, so that it becomes 3-degenerate. Thus, for $n = 27$ we have $ds_3^{\mathcal{P}}(n) \geq ds_3(G) = \frac{7}{27}n > \frac{n}{4}$. Then we can formulate the following lemma.

Lemma 6.7: *The maximum 3-degenerate skewness for planar graphs is at least $ds_3^{\mathcal{P}}(n) > \frac{n}{4}$.*

To prove that $ds_3(G) = 7$ we look at all combinations of six edges and show that even when we delete those edges the graph is still not 3-degenerate.

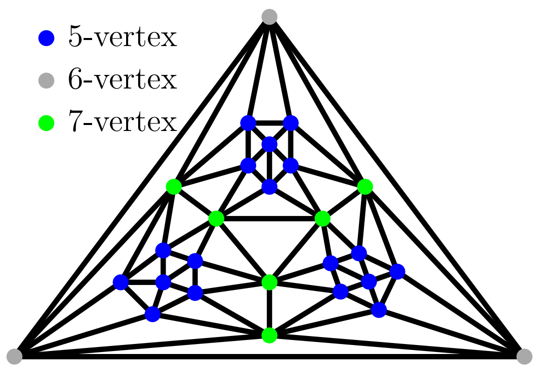

Figure 6.5.: Graph G with 27 vertices, for which $ds_3(G) = 7$

For this, we look at different collecting orderings of G. If we take a collecting ordering σ and it has a maximum degree greater than 3, then we need to delete a certain number of edges, to reduce the degree of the collected vertices. We can stop collecting the vertices in order of σ if we already deleted six edges and it is clear that the remaining graph is still not 3-degenerate.

We now distinguish all different cases of types for the first vertex in σ. From there we consider all subcases for the next collected vertex and so on.

1. The first collected vertex is 7-vertex adjacent to a 6-vertex. For this we need to delete four edges. The new graph is shown in Figure 6.6 (Case 1).
   a) The second collected vertex in σ for which we need to delete an edge is a 5-vertex. Here we need to delete two edges, thus we cannot delete any more edges. If we collect a 5-vertex not adjacent to a 4-vertex, then we cannot collect any more vertices. The remaining graph is not 3-degenerate. Thus, we collect a 5-vertex adjacent to a 4-vertex. The graph in Figure 6.6 (Case 1a)) is a subgraph of the remaining graph. Thus, the graph G without these six edges is still not 3-degenerate.
   b) The second collected vertex in σ for which we need to delete edges is a 4-vertex. We need to delete one edge and can still delete one other edge. The graph G' in Figure 6.6 (Case 1b)) is a subgraph of the remaining graph.
      i. The third collected vertex in σ for which we need to delete an edge is a 4-vertex not adjacent to other 4-vertices. We need to delete one edge before we can collect the 4-vertex. The remaining graph, shown in Figure 6.6 (Case 1b)i) is not 3-degenerate.
      ii. The third collected vertex in σ for which we need to delete an edge is a 4-vertex adjacent to other 4-vertices. No matter which of these we choose, we always find the graph in Figure 6.6 (Case 1b)ii) as subgraph. Thus the graph is not 3-degenerate after we deleted those six edges.

2. The first collected vertex is a 7-vertex adjacent to three other 7-vertices. Again, we need to delete four edges. The resulting graph is shown in Figure 6.7 (Case 2).
   a) The second collected vertex in σ for which we need to delete an edge is a 5-vertex. We need to delete two more edges, before we can collect it. The 5-vertex must be adjacent to a 4-vertex. Otherwise, no other vertex can

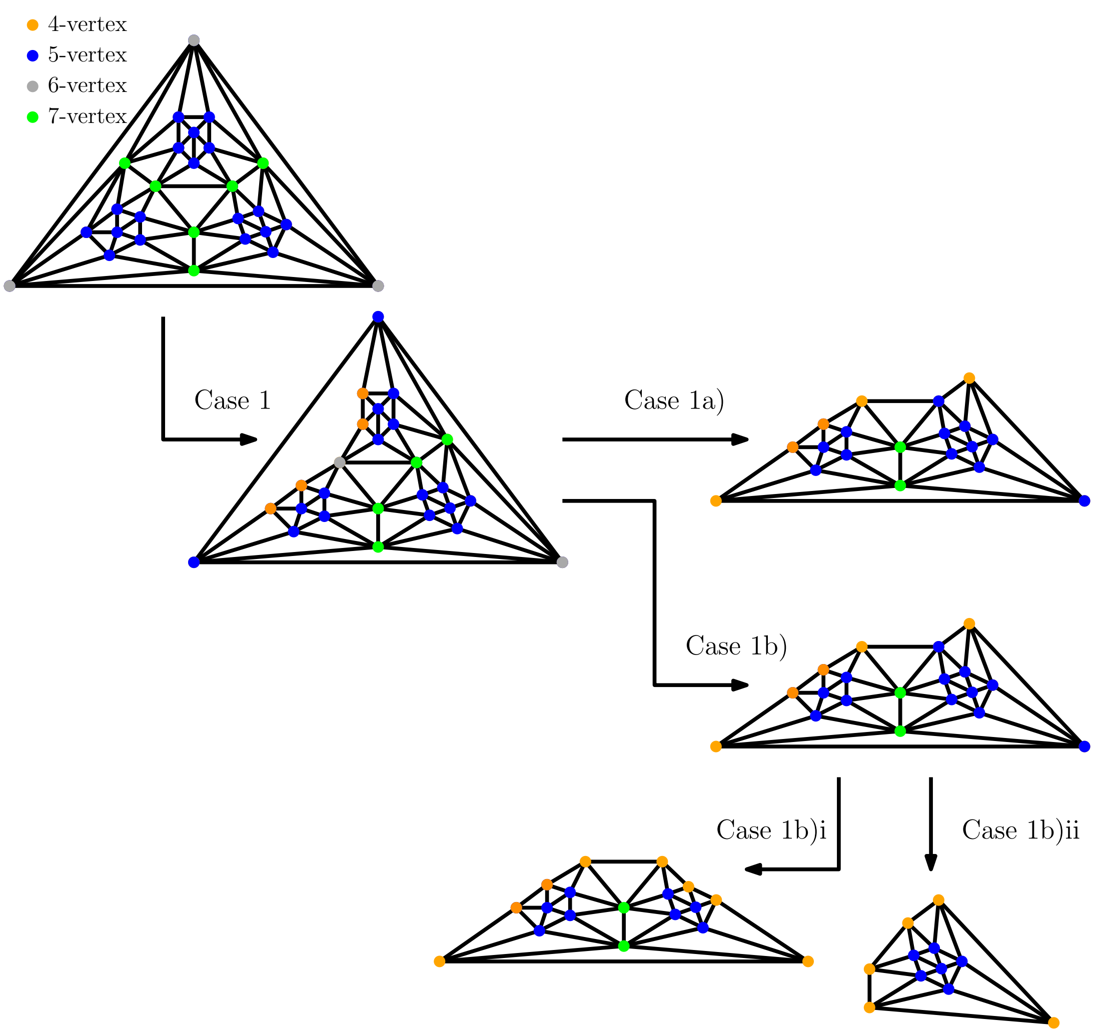

Figure 6.6.: Case 1, to show that for graph G we need to delete at least seven edges so that it becomes 3-degenerate. In this case, we first delete a 7-vertex adjacent to a 6-vertex.

be collected and it is clear, that the remaining graph is not 3-degenerate. No matter which 5-vertex adjacent to a 4-vertex we collect, we always have the graph in Figure 6.7 (Case 2a)) as a subgraph. This subgraph is not 3-degenerate, but we already deleted six edges.

b) The second collected vertex in σ for which we need to delete an edge is a 4-vertex. No matter which 4-vertex we collect, we need to delete one edge and get the graph G' in Figure 6.7 (Case 2b)) as a subgraph.

   i. The third collected vertex in σ for which we need to delete an edge is a 4-vertex not adjacent to other 4-vertices. Again we need to delete one edge, before we can collect the 4-vertex. The remaining graph is shown in Figure 6.7 (Case 2b)i) and is not 3-degenerate.

ii. The third collected vertex in σ for which we need to delete an edge is a 4-vertex adjacent to other 4-vertices. No matter which such vertex we choose, we always find the graph in Figure 6.7 (Case 2b)ii) as a subgraph. Thus the graph is not 3-degenerate after we deleted these six edges.

3. The first collected vertex is a 6-vertex.

4. The first collected vertex is a 5-vertex

This case distinction is not complete. Cases 3. and 4. contain more subcases, because we delete less edges in the beginning. We made a complete case distinction for Case 3 that we did not list here because the procedure stays the same and nothing more can be learned.

To still prove that $ds_3(G) = 7$, we wrote a C++ program. In this program, we first input all 75 edges of G and save them in a graph object using adjacency lists. We then create all possible subgraphs of G where six edges were deleted. This is done by using six `for`-loops that chose six different edges that are then deleted in a copy of the original graph.

The class `graph` we created has a function `degenerate` that checks whether the graph is k-degenerate for a given k. This function simulates the algorithm that gives us a minimum collecting ordering by always collecting the vertex of minimum degree. It saves all vertices in a priority queue where the key of a vertex is its degree. Then it can take the vertex with minimum degree and reduce the degree of its neighbors by one. If a vertex has already been collected like this, it will be ignored. If a vertex of minimum degree has degree greater than k the function returns `false`. Otherwise, the graph is k-degenerate and it returns `true` when all vertices are collected and the priority queue is empty.

The program now tests whether all the subgraphs with six deleted edges are 2-degenerate, by using the `degenerate` function. Should there be a combination of six edges that can be deleted so that G becomes 3-degenerate, it will output this. The code for this program can be found in Appendix A.1. When running the program no combination of six edges is found that when deleted results in a 3-degenerate graph.

6.5. Upper Bound for 3-degenerate Skewness

In the previous two sections, we looked at the lower bound for $ds_3^{\mathcal{P}}(n)$. Because every 2-degenerate graph is also 3-degenerate, we know that $ds_3^{\mathcal{P}}(n) \leq ds_2^{\mathcal{P}}(n) = n - 3$, see Theorem 6.4. Thus we know the following lemma is true.

Lemma 6.8: *To make a planar graph G 3-degenerate at most $n-3$ edges need to be deleted, where $n = |V(G)|$. That means that $ds_3^{\mathcal{P}}(n) \leq n - 3$.*

We can not give a better upper bound for the 3-degenerate skewness for planar graphs. If we use the greedy algorithm described in Section 4.2 to make a planar graph 3-degenerate and the same analysis technique as in Section 5.2, we only get an upper bound of $ds_3^{\mathcal{P}}(n) \leq \frac{6}{5}n - \frac{12}{5}$. This is greater than the upper bound of $n-3$. We did not find a graph G where we delete $\frac{6}{5}|V(G)| - \frac{12}{5}$ when using the greedy algorithms. But we also know of no other analysis technique that gives us a stronger upper bound for the number of deleted edges when using the greedy algorithm. Nonetheless, we prove in the following paragraph the bound of $ds_3^{\mathcal{P}}(n) \leq \frac{6}{5}n - \frac{12}{5}$.

If we follow the greedy algorithm, we always collect the vertex with minimum degree next. Because we only know that our graph G is planar, we only know that the degree of the next vertex v is at most 5. We use the notation $\overline{\deg}(v)$ for the degree of a vertex v at the time it is identified as the vertex with minimum degree and collected next. If $\overline{\deg}(v) = 5$, we need to delete two edges incident to v. If $\overline{\deg}(v) = 4$, we need to delete one edge incident to v. Otherwise we do not need to delete any edge. Let n_5 denote the number of vertices with $\overline{\deg}(v) = 5$ and n_4 denote the number of vertices with $\overline{\deg}(v) = 4$. Like always n denotes the number of vertices in G. We denote with p the number of edges that are deleted. We can formulate the following conditions. The number of vertices with $\overline{\deg}(v) = 5$ and number of vertices with $\overline{\deg}(v) = 4$ combined are at most the total number of vertices in G. Also five times n_5 and four time n_4 must be at most the total number of edges in G. Thus, we have

$$\begin{aligned} n_5 + n_4 &\leq n \quad \text{and} \\ 5n_5 + 4n_4 &\leq 3n - 6. \end{aligned}$$

We can describe the total number p of deleted edges by

$$p = 2n_5 + n_4.$$

We now derive an upper bound for p depending on n.

$$\begin{aligned} & p = 2n_5 + n_4 \\ \Leftrightarrow \quad & \frac{5}{2}p = 5n_5 + \frac{5}{2}n_4 \leq 5n_5 + 4n_4 \leq 3n - 6 \\ \Leftrightarrow \quad & p \leq \frac{6}{5}n - \frac{12}{5} \end{aligned}$$

It might be possible that there are other conditions, that we did not consider here, that reduce the upper bound further. For this thesis $n - 3$ is the best upper bound for $ds_3^{\mathcal{P}}(n)$ that we can give.

6.6. Upper Bound for 4-degenerate Skewness using Greedy Algorithm

In this section, we use the greedy algorithm introduced in Section 4.2. We want to find an upper bound for the number of edges necessary that, when deleted, make a planar graph 4-degenerate.

Lemma 6.9: *To make a planar graph G 4-degenerate, at most $\frac{3}{5}n$ edges need to be deleted, where $n = |V(G)|$. That means that $ds_4^{\mathcal{P}}(n) \leq \frac{3}{5}n$.*

Proof. Let σ be a collecting ordering that always collects the vertex of minimum degree. Like in Lemma 5.5 denote by $\overline{\deg}(v)$ the degree of a vertex v at the time when it will be collected next. We want G to become 4-degenerate and, thus, we reduce the maximum degree of σ by deleting edges incident to v if $\overline{\deg}(v) > 4$. Because G is planar, we know that we can always find a vertex v with $\overline{\deg}(v) \leq 5$. If we take v and delete edges until $\overline{\deg}(v) \leq 4$, we need to delete at most one edge. We can then collect this vertex and repeat these steps until there are no more vertices. We argue that the case that our vertex v with minimum degree is a 5-vertex happens to at most $\frac{3}{5}n$ vertices. The sum of $\overline{\deg}(v)$ for all vertices $v \in V(G)$ is equal to the total number of edges in G. If at least $\frac{3}{5}n$ vertices have $\overline{\deg}(v) = 5$, then we get $|E(G)| \geq 5 \cdot (\frac{3}{5}n) = 3n > 3n - 6$, which is a contradiction to the statement that G is planar. Because we only deleted exactly one edge for each vertex v with $\overline{\deg}(v) = 5$, we get an upper bound of $\frac{3}{5}n$ edges that need to be deleted. This concludes the proof. $\square$

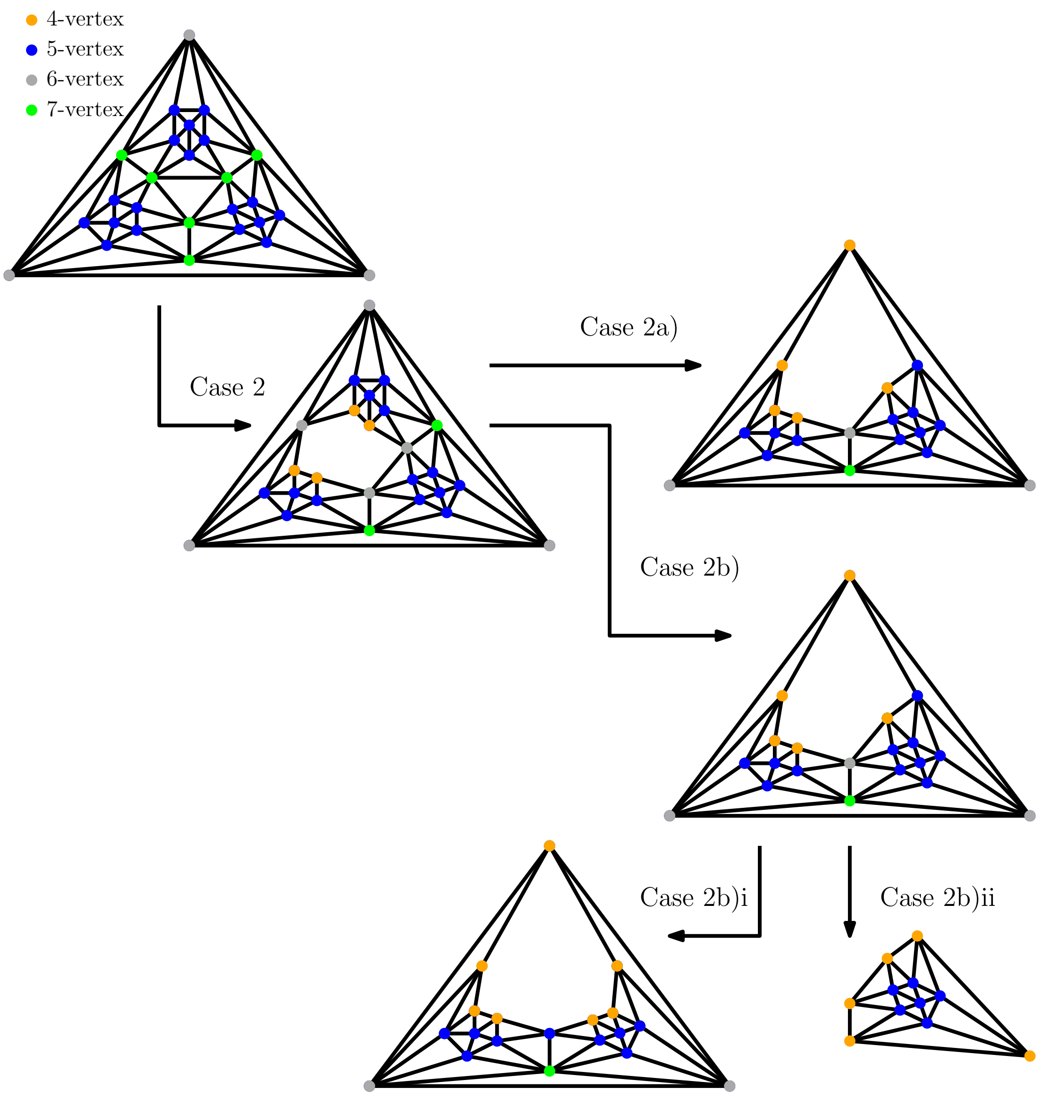

Figure 6.7.: Case 2, to show that for graph G we need to delete at least seven edges so that it becomes 3-degenerate. In this case, we first delete a 7-vertex adjacent to three other 7-vertices.

7. Conclusion

In this thesis, we discussed the problem of finding a maximum spanning k-degenerate subgraph. We introduced the k-degenerate skewness $ds_k(G)$ of a graph as the number of edges that need to be deleted to make G k-degenerate. Furthermore, we defined the maximum k-degenerate skewness $ds_k^{\mathcal{C}}(n)$ of a graph class $\mathcal{C}$ as the maximum of $ds_k(G)$ for all graphs $G \in \mathcal{C}$ with n vertices. We then went on an described several methods to find k-degenerate spanning subgraphs. Later, we restricted the considered graph classes to the class $\mathcal{P}$ of planar graphs and the class $\mathcal{P}$ of bipartite planar graphs. To find 1-degenerate subgraphs, it is enough to find a spanning tree. With this knowledge, we were able to give concrete values for the maximum 1-degenerate skewness for graphs in class $\mathcal{P}$ and class $\mathcal{P}$. We were also able to prove that the maximum 2-degenerate skewness for graphs in $\mathcal{P}$ is $ds_2^{\mathcal{P}}(n) = n - 3$. For the other values, we were only able to give bounds. We proved that the maximum 2-degenerate skewness for graphs in $\mathcal{P}$ lies between $\frac{n}{4} - 1$ and $\frac{n}{2} - 1$. For planar graphs we know that $ds_3^{\mathcal{P}}(n)$ is between $\frac{n}{4}$ and $n - 3$ and $ds_4^{\mathcal{P}}(n)$ is smaller than $\frac{3}{5}n$.

7.1. Outlook

In further research, it might be possible to sharpen the bounds for $ds_3^{\mathcal{P}}(n)$, $ds_4^{\mathcal{P}}(n)$ and $ds_2^{\mathcal{B}}(n)$. For $ds_2^{\mathcal{B}}(n)$, i.e. the maximum 2-degenerate skewness for graphs in $\mathcal{B}$, we already conjecture that the bound can be reduced to $\frac{n}{4}$.

In Section 4.2, we introduced a greedy algorithm that always collects the vertices of minimum degree. As stated before, it would be interesting to study further how much worse the greedy algorithm is in comparison to an optimal algorithm. There are also other greedy algorithms that could be considered. For example we could always delete an edge with minimum or maximum weight. It might also be better to first delete edges incident to vertices with a high degree. In Section 4.3, we discussed that deleting maximum matchings is not enough to reduce the degeneracy of a graph. But it might be possible to change a maximum matching M of a graph G into a set of edges M', so that the degeneracy of $G - M'$ is reduced by one. It is also possible that a matching with certain constrains is enough to reduce the degeneracy of a graph.

In this thesis we mostly restricted ourselves to planar and bipartite planar graphs. There are other graph classes for which the k-degenerate skewness should be studied. One example is chordal graphs. The degeneracy of a chordal graphs is not bounded. But it is still plausible that the maximum k-degenerate skewness can be described depending on the number of vertices in the chordal graph. This is the case because, similar to the minimum collecting ordering for degeneracy, a chordal graph has a perfect elimination ordering π. In that ordering all neighbors of a vertex v that come after v in π form a clique. The degeneracy of a specific chordal graph G is bounded by the size of its biggest clique. To reduce the degeneracy of G it would be interesting to see if the perfect elimination ordering can help to identify a minimum number of edges that need to be deleted.

In Chapter 1 we stated the Problem 1.1, given a graph G and an integer k find a maximum subgraph H of G such that H is k-degenerate. In Chapter 3 we explained how the paper on the NP-completeness of note-deletion problems [LY80] proves that the version of Problem 1.1 that uses induced subgraphs is NP-complete. The same is true for the skewness of non-planar graphs. It is probable that if we look for spanning subgraphs, the problem is also NP-complete. This is yet to be proven.

Furthermore, there are other versions of Problem 1.1 that can be studied. One version of Problem 1.1 is to allow the deletion of edges and vertices. It is also interesting not to look for subgraphs that are k-degenerate, but instead to allow certain modifications of G. In Chapter 1 we already noted that for $k \geq 2$ subdividing an edge by placing a vertex in the middle it the same as deleting the edge. Then the problem is to find the minimum number of edges that need to be subdivided such that the obtained graph is k-degenerate. It would be interesting to see what the effect of such subdivisions are for graph drawing algorithms that only work on k-degenerate graphs for $k \geq 2$. One would be the algorithm for circular Lombardi drawings of 2-degenerate graphs mentioned in [Dun+11].

In a similar way, G could be modified by splitting vertices. To split a vertex v, we replace v with two vertices v_1 and v_2. The vertices v_1 and v_2 are adjacent and every neighbor of v is now neighbor of either v_1 or v_2. Now the problem is formulated as follows: Given a graph G and an integer k, find the minimum number of vertices that need to be split such that the obtained graph is k-degenerate. These versions was not studied in this thesis but would be interesting to study in the future.

Bibliography

[AH78] Kenneth Appel and Wolfgang Haken. “The Four-Color Problem”. In: *Mathematics Today Twelve Informal Essays*. Edited by Lynn Arthur Steen. New York, NY: Springer New York, 1978, pp. 153–180. ISBN: 978-1-4613-9435-8. DOI: *10.1007/978-1-4613-9435-8_7*. URL: *https://doi.org/10.1007/978-1-4613-9435-8_7*.

[BM05] G. Brinkmann and Brendan D. Mckay. “Construction of Planar Triangulations with Minimum Degree 5”. In: *Discrete Math.* Volume 301.2–3 (Oct. 2005), pp. 147–163. ISSN: 0012-365X. DOI: *10.1016/j.disc.2005.06.019*. URL: *https://doi.org/10.1016/j.disc.2005.06.019*.

[Bri+05] Gunnar Brinkmann, Sam Greenberg, Catherine Greenhill, Brendan D. McKay, Robin Thomas, and Paul Wollan. “Generation of simple quadrangulations of the sphere”. In: *Discrete Mathematics* Volume 305.1 (2005), pp. 33–54. ISSN: 0012-365X. DOI: *https://doi.org/10.1016/j.disc.2005.10.005*. URL: *https://www.sciencedirect.com/science/article/pii/S0012365X05005170*.

[Cim92] Robert J Cimikowski. “Graph planarization and skewness”. In: *Congressus Numerantium* (1992), pp. 21–21.

[CL05] G. L. Chia and C. L. Lee. “Crossing Numbers and Skewness of Some Generalized Petersen Graphs”. In: *Combinatorial Geometry and Graph Theory*. Edited by Jin Akiyama, Edy Tri Baskoro, and Mikio Kano. Berlin, Heidelberg: Springer Berlin Heidelberg, 2005, pp. 80–86. ISBN: 978-3-540-30540-8.

[CS13] Gek L. Chia and Kai An Sim. “On the skewness of the join of graphs”. In: *Discrete Applied Mathematics* Volume 161.16 (2013), pp. 2405–2409. ISSN: 0166-218X. DOI: *https://doi.org/10.1016/j.dam.2013.03.023*. URL: *https://www.sciencedirect.com/science/article/pii/S0166218X13001704*.

[CW17] Daniel W. Cranston and Douglas B. West. “An introduction to the discharging method via graph coloring”. In: *Discrete Mathematics* Volume 340.4 (2017), pp. 766–793. ISSN: 0012-365X. DOI: *https://doi.org/10.1016/j.disc.2016.11.022*. URL: *https://www.sciencedirect.com/science/article/pii/S0012365X1630379X*.

[DK18] Zdeněk Dvořák and Tom Kelly. “Induced 2-degenerate Subgraphs of Triangle-free Planar Graphs”. In: *Combinatorics* Volume 25 (2018). DOI: *10.37236/7311*. URL: *https://doi.org/10.37236/7311*.

[DPP90] H. De Fraysseix, J. Pach, and R. Pollack. “How to draw a planar graph on a grid”. In: *Combinatorica* (Mar. 1990). URL: *https://doi.org/10.1007/BF02122694*.

[Dun+11] Christian A. Duncan, David Eppstein, Michael T. Goodrich, Stephen G. Kobourov, and Martin Nöllenburg. "Lombardi Drawings of Graphs". In: *Graph Drawing*. Edited by Ulrik Brandes and Sabine Cornelsen. Berlin, Heidelberg: Springer Berlin Heidelberg, 2011, pp. 195–207. ISBN: 978-3-642-18469-7.

[EH66] P. Erdős and A. Hajnal. "On chromatic number of graphs and set-systems". In: *Acta Mathematica Hungarica* Volume 17.1-2 (1966), pp. 61–99. DOI: *10.1007/bf02020444*. URL: *https://akjournals.com/view/journals/10473/17/1-2/article-p61.xml*.

[Gek12] Chan Lye LEE Gek Ling CHIA. "Skewness of generalized Petersen graphs and related graphs". In: *Frontiers of Mathematics in China* Volume 7.3, 427 (2012), p. 427. DOI: *10.1007/s11464-012-0186-5*. URL: *https://journal.hep.com.cn/fmc/EN/abstract/article_2915.shtml*.

[Gu+] Yangyan Gu, H. A. Kierstead, Sang-il Oum, Hao Qi, and Xuding Zhu. "3-Degenerate induced subgraph of a planar graph". In: *Journal of Graph Theory* Volume n/a.n/a (). DOI: *https://doi.org/10.1002/jgt.22740*. URL: *https://onlinelibrary.wiley.com/doi/abs/10.1002/jgt.22740*.

[Kai74] Paul Kainen. "A Generalization of the 5-Color Theorem". In: *Proceedings of The American Mathematical Society - PROC AMER MATH SOC* Volume 45 (Sept. 1974), pp. 450–450. DOI: *10.1090/S0002-9939-1974-0345861-4*.

[Kan96] Goos Kant. "Drawing Planar Graphs Using the Canonical Ordering". In: *Algorithmica* (July 1996). URL: *https://doi.org/10.1007/BF02086606*.

[Liu77] Peter C Liu. "On the deletion of non-planar edges of a graph". In: *Proceedings of the 10th South-East Conference on Combinatorics Graph Theory, and Computing, Boca Raton, FL, USA*. 1977, pp. 727–738.

[LMZ15] Robert Lukoťka, Ján Mazák, and Xuding Zhu. "Maximum 4-degenerate subgraph of a planar graph". In: *Combinatorics* Volume 22 (2015). DOI: *10.37236/4265*. URL: *https://doi.org/10.37236/4265*.

[LW70] Don R. Lick and Arthur T. White. "k-Degenerate Graphs". In: *Canadian Journal of Mathematics* Volume 22.5 (1970), pp. 1082–1096. DOI: *10.4153/CJM-1970-125-1*.

[LY80] John M. Lewis and Mihalis Yannakakis. "The node-deletion problem for hereditary properties is NP-complete". In: *Journal of Computer and System Sciences* Volume 20.2 (1980), pp. 219–230. ISSN: 0022-0000. DOI: *https://doi.org/10.1016/0022-0000(80)90060-4*. URL: *https://www.sciencedirect.com/science/article/pii/0022000080900604*.

[MB83] David W. Matula and Leland L. Beck. "Smallest-Last Ordering and Clustering and Graph Coloring Algorithms". In: *J. ACM* Volume 30.3 (July 1983), pp. 417–427. ISSN: 0004-5411. DOI: *10.1145/2402.322385*. URL: *https://doi.org/10.1145/2402.322385*.

[Sa19] Manuel Sorge and et al. *The graph parameter hierarchy*. 2019. URL: *https://manyu.pro/assets/parameter-hierarchy.pdf*.

[Tho01] Carsten Thomassen. “Decomposing a Planar Graph into an Independent Set and a 3-Degenerate Graph”. In: *Journal of Combinatorial Theory, Series B* Volume 83.2 (2001), pp. 262–271. ISSN: 0095-8956. DOI: *https://doi.org/10.1006/jctb.2001.2056*. URL: *https://www.sciencedirect.com/science/article/pii/S0095895601920568*.

[Tho95] C. Thomassen. “Decomposing a Planar Graph into Degenerate Graphs”. In: *Journal of Combinatorial Theory, Series B* Volume 65.2 (1995), pp. 305–314. ISSN: 0095-8956. DOI: *https://doi.org/10.1006/jctb.1995.1057*. URL: *https://www.sciencedirect.com/science/article/pii/S009589568571057X*.

[Wer04] P. Wernicke. “Über den kartographischen Vierfarbensatz”. In: *Mathematische Annalen* Volume 58.3 (1904), pp. 413–426. DOI: *10.1007/BF01444968*. URL: *https://doi.org/10.1007/BF01444968*.

A. Appendix

A.1. C++ code for degeneracy check

```cpp
// C++ program to check if, given a graph, there are 6 edges
// that can be deleted such that the resulting graph is
// k-degenerate
#include <bits/stdc++.h>
using namespace std;

struct vertex {
    int id;
    int degree;
};

// This class represents a undirected graph using adjacency
// list representation
class Graph {

    // Pointer to an array containing adjacency lists
    vector<list<int>> adj{};
public:
    Graph(int n) : adj(n) {} // Constructor

    // function to add an edge to graph
    void addEdge(int u, int v);

    // function to remove an edge to graph
    void removeEdge(int u, int v);

    // function to check if graph is k-degenerate
    bool degenerate(int k);

    // gives the number of vertices of graph
    int size() { return adj.size(); }
};

// function to check if graph is k-degenerate
bool Graph::degenerate(int k) {
    vector<char> collected(size(), false);
    vector<char> degree(size(), 0);
    for(int i = 0; i < size(); ++i)
        degree.at(i) = adj.at(i).size();
    auto comparator = [](auto fst, auto scn) {
        return fst.degree > scn.degree ||
          (fst.degree == scn.degree && fst.id > scn.id); };
    std::priority_queue<vertex, std::vector<vertex>,
            decltype(comparator)> queue{comparator};
    for(int i = 0; i < size(); ++i)
        queue.push(vertex{i, degree.at(i)});

    while(!queue.empty()) {
        auto top = queue.top();
        queue.pop();
        if(collected.at(top.id)) continue;

        collected.at(top.id) = true;
        if(degree.at(top.id) != top.degree)
            exit(2);

        if(top.degree > k) {
            return false;
        }

        for(auto neighbor : adj.at(top.id)) {
            if(collected.at(neighbor)) continue;
            degree.at(neighbor) -= 1;
            queue.push(vertex{neighbor, degree.at(neighbor)});
        }
    }
    return true;
}

void Graph::addEdge(int u, int v) {
    adj[u].push_back(v);
    adj[v].push_back(u);
}

void Graph::removeEdge(int u, int v) {
    adj[u].remove(v);
    adj[v].remove(u);
}


struct edge {
    int v1_id;
    int v2_id;
};
```

```
int main() {
    const int k = 3;
    Graph graph(27);

    vector<edge> edges {
        edge{0,6},      edge{0,9},      edge{0,26},
        edge{0,25},     edge{0,20},     edge{0,18},
        edge{1,2},      edge{1,3},      edge{1,4},
        edge{1,10},     edge{1,8},      edge{1,26},
        edge{2,10},     edge{2,13},     edge{2,14},
        edge{2,19},     edge{2,3},      edge{3,19},
        edge{3,23},     edge{3,24},     edge{3,26},
        edge{4,10},     edge{4,5},      edge{4,7},
        edge{4,8},      edge{5,6},      edge{7,8},
        edge{7,5},      edge{7,6},      edge{8,26},
        edge{9,6},      edge{9,7},      edge{9,8},
        edge{9,26},     edge{10,5},     edge{10,12},
        edge{10,13},    edge{11,10},    edge{11,5},
        edge{11,6},     edge{11,0},     edge{11,18},
        edge{11,16},    edge{11,12},    edge{12,13},
        edge{13,14},    edge{14,17},    edge{15,13},
        edge{15,14},    edge{15,12},    edge{15,16},
        edge{15,17},    edge{16,12},    edge{16,17},
        edge{18,16},    edge{18,17},    edge{18,20},
        edge{18,21},    edge{19,14},    edge{19,17},
        edge{19,18},    edge{19,21},    edge{19,23},
        edge{21,20},    edge{21,23},    edge{22,21},
        edge{22,23},    edge{22,24},    edge{22,25},
        edge{22,20},    edge{24,26},    edge{24,23},
        edge{25,20},    edge{25,26},    edge{25,24},
    };

    if(edges.size() != 75) exit(1);
    int m = edges.size();


    for (int j = 0; j < m; ++j) {
        graph.addEdge(edges[j].v1_id, edges[j].v2_id);
    }


    const auto constcopy = graph;

    bool found = false;
    for (int i1 = 0; i1 < m-5; ++i1) {
        for (int i2 = i1+1; i2 < m-4; ++i2) {
            for (int i3 = i2+1; i3 < m-3; ++i3) {
                for (int i4 = i3+1; i4 < m-2; ++i4) {
                    for (int i5 = i4+1; i5 < m-1; ++i5) {
                        for (int i6 = i5+1; i6 < m; ++i6) {
                            auto copy = constcopy;
                            for(auto el : {i1, i2, i3, i4, i5, i6})
                                copy.removeEdge(edges[el].v1_id, edges[el].v2_id);
                            if (copy.degenerate(k)) {
                                stringstream collection;
                                collection << "Found 6 edges such that, when "
                                     << "deleted, the graph is " << k
                                     << "-degenerate" << endl;
                                for(auto el : {i1, i2, i3, i4, i5, i6})
                                    collection << el << "; ";
                                collection << endl;
                                found = true;
                            }
                        }
                    }
                }
            }
        }
    }

    if(found == true) {
        cout << "Found 6 edges such that, when deleted, "
           << "the graph is " << k << "-degenerate" << endl;
    } else {
        cout << "Did not find 6 edges such that, when "
          << "deleted, the graph is " << k << "-degenerate" << endl;
    }
    return 1;
}
```