Technical Report AC-TR-19-007

September 2019

A Lower Bound for the Smallest Uniquely Hamiltonian Planar Graph with Minimum Degree Three

Benedikt Klocker, Herbert Fleischner, and Günther R. Raidl

A Lower Bound for the Smallest Uniquely Hamiltonian Planar Graph with Minimum Degree Three

Benedikt Klocker*, Herbert Fleischner, Günther R. Raidl

Institute of Logic and Computation, TU Wien,
Favoritenstraße 9-11/192-01, 1040 Vienna, Austria

Abstract

Bondy and Jackson conjectured in 1998 that every planar uniquely hamiltonian graph must have a vertex of degree two. In this work we verify computationally Bondy and Jackson's conjecture for graphs with up to 25 vertices. Using a reduction we search for graphs that contain a stable fixed-edge cycle or equivalently a stable cycle with one vertex of degree two. For generating candidate graphs we use plantri and for checking if they contain a stable fixed-edge cycle we propose three approaches. Two of them are based on integer linear programming (ILP) and the other is a cycle enumeration algorithm. To reduce the search space we prove several properties a minimum planar graph with minimum degree three containing a stable fixed-edge cycle must satisfy, the most significant being triangle freeness. Comparing the three algorithms shows that the enumeration is more effective on small graphs while for larger graphs the ILP-based approaches perform better. Finally, we use the enumeration approach together with plantri to check that there does not exist a planar graph with minimum degree three which contains a stable fixed-edge cycle with 24 or fewer vertices.

Keywords: uniquely hamiltonian graphs, integer linear programming, stable cycles, minimum counterexample

1. Introduction

The class of hamiltonian graphs is a widely studied research field in graph theory. A special subclass of those graphs are the *uniquely hamiltonian* graphs, the graphs that contain exactly one hamiltonian cycle. Already in 1946 Smith proved that every edge of a 3-regular graph is contained in an even number of hamiltonian cycles, which was published by Tutte [1]. Clearly, this result implies that 3-regular graphs can never be uniquely hamiltonian. This was then further improved by Thomason [2] who showed that all graphs where all vertices have odd degrees cannot be uniquely hamiltonian. As an implication we get that every uniquely hamiltonian graph must contain at least two vertices of even degree.

The aforementioned early results show that vertex degrees play an important role for uniquely hamiltonian graphs. Bondy and Jackson [3] provided an upper bound of $c\log_2(8n)+3$ with $c\approx 2.41$ for the smallest vertex degree $\delta(G)$ for a uniquely hamiltonian graph G. Abbasi and Jamshed [4] could improve the upper bound to $c\log_2(n)+2$ with $c\approx 1.71$. Naturally, the question arises how tight this upper bound is.

Already in 1980 Entringer and Swart [5] found uniquely hamiltonian graphs with minimum degree three. Their graphs have only two vertices of degree four and all other vertices have degree three, which is as close as we can get to 3-regular uniquely hamiltonian graphs by Thomason's theorem. Sheehan conjectured in 1975 that there do not exist 4-regular uniquely hamiltonian graphs. If we allow parallel edges, Fleischner [6]

*Corresponding author.
Email addresses: `klocker@ac.tuwien.ac.at` (Benedikt Klocker), `fleischner@ac.tuwien.ac.at` (Herbert Fleischner), `raidl@ac.tuwien.ac.at` (Günther R. Raidl)

constructed infinitely many 4-regular uniquely hamiltonian multigraphs. This construction could not be extended to simple graphs and therefore Sheehan's conjecture is still an open problem.

Another question raised by Bondy [7] asks whether there exists a uniquely hamiltonian graph with minimum degree four. This question got answered by Fleischner [8] who constructed an infinite family of uniquely hamiltonian graphs that only have vertices of degree four and 14. The big gap between the best known minimum degree $\delta(G) = 4$ and the upper bound $\delta(G) \leq c \log_2(n) + 2$ remains an open problem.

If we consider only planar graphs Bondy and Jackson [3] provided a much smaller upper bound of $\delta(G) \leq 3$ by showing that a planar uniquely hamiltonian graph has at least two vertices of degree two or three. Clearly, there exist planar uniquely hamiltonian graphs with $\delta(G) = 2$, for example the cycle graphs C_n. Bondy and Jackson conjectured that every planar uniquely hamiltonian graph contains a vertex of degree two, i.e. $\delta(G) \leq 2$ for planar uniquely hamiltonian graphs, which would close the gap for the case of planar graphs.

Since the construction of uniquely hamiltonian graphs with minimum degree four by Fleischner was quite surprising for the scientific community, one may guess that there exist also planar uniquely hamiltonian graphs with minimum degree three. Unfortunately, the techniques used by Fleischner to construct uniquely hamiltonian graphs with minimum degree four cannot directly be applied to planar graphs. This motivates the computer-aided search for such graphs on which we will focus in this article. Especially, we will systematically search through planar graphs up to a certain order to either find a counterexample to the conjecture of Bondy and Jackson or establish a lower bound for the order of a uniquely hamiltonian planar graph with minimum degree three. Such a lower bound of 18 vertices was established recently by Goedgebeur et al. [9] who developed a construction procedure for generating all non-isomorphic graphs with a given number of hamiltonian cycles, which works especially well for a small number of cycles like in the case of uniquely hamiltonian graphs. As we will see in Theorem 6.2 we will improve this lower bound to 25 vertices. Furthermore, also with the goal to find a uniquely hamiltonian planar graph with minimum degree three, we developed in a previous work [10] an algorithm for finding uniquely hamiltonian graphs with minimum degree three with a small crossing number. Although, we could not find any planar graphs or even graphs with crossing number one in this work, we obtained many graphs of different orders with crossing number two.

Instead of directly searching for uniquely hamiltonian planar graphs we will focus in this work on searching for a planar graph with minimum degree three containing a so called stable fixed-edge cycle, or short SFE-cycle. In Section 2 we show that such a graph would imply the existence of a uniquely hamiltonian planar graph with minimum degree three. As we will see we can prove strong properties for a minimum planar graph with minimum degree three that contains an SFE-cycle. This helps us to search for such a graph more effectively.

Our overall approach is to use the tool plantri [11] to generate representants of all isomorphism classes of planar graphs with minimum degree three of some given order. Then we check for each of these graphs if they contain an SFE-cycle or not. To that end we formulate three problem variants that decide for a given input graph if it contains an SFE-cycle, a dominating SFE-cycle, or a dominating SFE-cycle with some additional properties, respectively. The third problem variant is used for checking if a graph is a potential minimum counterexample. We prove in Section 4 that the stable cycle of a minimum counterexample must satisfy those additional properties. In Section 4 we also prove some properties a graph must satisfy to be a minimum counterexample, one of which is the strong property of triangle freeness. To effectively generate such candidate graphs we use plantri to construct dual graphs with minimum degree four.

For solving the three problem variants we propose an integer linear programming (ILP) based approach [12] and an enumeration scheme. Using both approaches helps us to verify the results and allows us comparing the performances for different graph sizes. As already mentioned the main result of this article is a new lower bound of 25 vertices for the smallest uniquely hamiltonian planar graph with minimum degree three.

In this work we only consider undirected simple graphs. Furthermore, we focus on planar graphs. In Section 2 we present the reductions from uniquely hamiltonian to stable cycles and the first two of the three problems that we consider. Two different solution approaches for solving the problems are then presented in Section 3. In Section 4 we focus on properties a minimum counterexample must satisfy and formulate the third problem variant. We then apply our algorithms presented in Section 3 on different instance groups and summarize the results in Section 6. Finally, we conclude with Section 7 and propose some further research ideas.

2. Reduction to Stable Fixed-Edge Cycles

In this section we reduce the search for uniquely hamiltonian planar graphs with minimum degree three to the search for planar graphs with minimum degree three with a stable fixed-edge cycle.

We will start with defining what we mean by a stable fixed-edge cycle.

Definition 2.1. Let G be a graph. A *fixed-edge cycle*, or short an *FE-cycle*, is a pair (C, e) where C is a cycle of G and e an edge occurring in C. An FE-cycle (C, e) is called *maximal* if there is no FE-cycle (C', e) in G with $V(C') \supsetneq V(C)$. Moreover, a FE-cycle (C, e) is called *dominating* if C is edge-dominating in G, i.e. for all edges $e = vw \in G$ either v or w is in $V(C)$. Finally, a maximal FE-cycle (C, e) is called a *stable fixed-edge cycle* or short *SFE-cycle* if there is no other FE-cycle (C', e) with $C' \neq C$ and $V(C') = V(C)$.

We call a graph containing an SFE-cycle an *SFE-graph.*

The following proposition transforms a graph containing a hamiltonian SFE-cycle into a uniquely hamiltonian graph in a planarity and minimum degree preserving way. This implies that we can search for planar graphs containing a hamiltonian SFE-cycle instead of uniquely hamiltonian planar graphs. A similar transformation was first mentioned by Leder in his master thesis [13].

Proposition 2.2. *Let G be a graph containing a hamiltonian SFE-cycle. Then there exists a uniquely hamiltonian graph G' with $|V(G')| = 2|V(G)| - 2$ and $\delta(G') \geq \delta(G)$. Furthermore, if G is planar then G' is also planar.*

Proof. Let (C, e) with $e = (u, v)$ be the hamiltonian SFE-cycle of G. We construct G' by starting with two copies G^* and G^{**} of $G \setminus e$. We then identify the vertices $u^* \in V(G^*)$ with $u^{**} \in V(G^{**})$ and $v^* \in V(G^*)$ with $v^{**} \in V(G^{**})$ to glue the two copies together. The hamiltonian cycle C naturally implies a hamiltonian cycle C' in G' by using two copies of $C \setminus e$ and connecting them. It is also stable since if there would be another hamiltonian cycle C'' in G' then it must be different in one of the two copies. Considering the edges of the cycle in this copy together with the edge e gives us then a cycle in G that is different to C and also a hamiltonian SFE-cycle of G, which is a contradiction.

Clearly, $|V(G')| = 2|V(G)| - 2$ holds. All vertices in the two copies have the same degrees as in the original graph except for the identified vertices u^* and v^*, but the degrees of those two can only increase and therefore we have $\delta(G') \geq \delta(G)$. Furthermore, if G is planar we can find an embedding such that e is on the outer face and then removing e, adding the copy of the embedding and identifying u^* and v^* with their copies can be done without any crossings. Therefore, G' is also planar.

Note that Leder connected the vertices u^* with u^{**} and v^* with v^{**} using new edges instead of identifying them, which results in a slightly larger graph with the advantage that all vertices in the copies have the same degrees as in the original graph. This is especially important for regular graphs, since it is regularity-preserving.

In the next result we present the transformation by Leder [13] that transforms a graph containing a dominating SFE-cycle into a graph containing a hamiltonian SFE-cycle preserving planarity.

Theorem 2.3. *Let G be a graph with minimum degree three containing a dominating SFE-cycle (C, e). Then there exists a graph G' with minimum degree three that contains a hamiltonian SFE-cycle with $|V(G')| \leq 2|V(G)| - |V(C)|$. Furthermore, if G is planar, then G' is also planar.*

Proof. We apply iteratively a planarity preserving transformation that constructs from a graph G with a dominating SFE-cycle (C, e) a new graph G' with a dominating SFE-cycle (C', e') such that the number of unvisited vertices by the cycle is reduced by one. The order of the new graph is at most increased by one

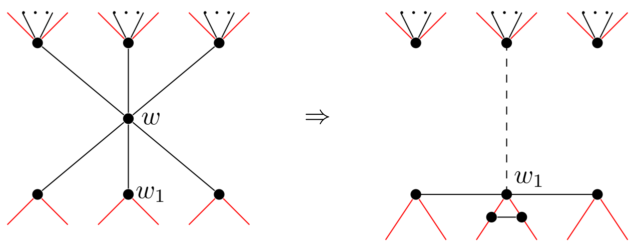

Figure 1: Transformation used for removing unvisited vertices. The dominating cycle is visualized by the red edges. The dashed edge is used if there is only one degree three neighbor and the two new vertices are only inserted if there are three or more degree three neighbors.

per iteration. Let $w \in V(G) \setminus V(C)$ be an unvisited vertex. If w has no neighbors of degree three, we can just remove the vertex w and define $G' = G \setminus v$ and $(C', e') = (C, e)$.

In the other case let w_1 be a neighbor of degree three. If w_1 is the only neighbor of degree three we connect w_1 with any other neighbor w_2 of w and define $G' = G \setminus v \cup (w_1, w_2)$ and $(C', e') = (C, e)$. Any cycle in G' that contains the edge (w_1, w_2) can be translated to a cycle in G containing the vertex w. Therefore, (C, e) being stable implies that (C', e') is stable.

If w has exactly two neighbors of degree three w_1 and w_2 we can connect those two and remove the vertex w, i.e. $G' = G \setminus v \cup (w_1, w_2)$, and $(C', e') = (C, e)$. Again any cycle in G' that contains the edge (w_1, w_2) corresponds to a cycle in G containing w and therefore this implies that (C', e') is stable.

The most interesting case is if w has at least three neighbors of degree three. Let e_1 and e_2 be the other two edges incident to w_1 that are not incident to w. Let $x_1 \neq w$ and $x_2 \neq w$ be the other end vertex of e_1 and e_2. We remove now the vertex w and connect all other neighbors of w with degree three to w_1. To preserve the stability of the cycle we need to add two more vertices v_1 and v_2 placed on the edges e_1 and e_2 and connect them, see Figure 1. Formally we define

$$G' = G \setminus \{w, e_1, e_2\} \cup \{v_1, v_2, (v_1, v_2), (w_1, v_1), (w_1, v_2), (v_1, x_1), (v_2, x_2)\} \bigcup_{w' \in N(w) : w' \neq w_1, \deg(w') = 3} (w', w_1)$$

and $C' = C \setminus \{(w_1, x_1), (w_1, x_2)\} \cup \{(w_1, v_1), (w_1, v_2), (v_1, x_1), (v_2, x_2)\}$. If the fixed edge was e_1 or e_2 we define as fixed edge $e' = (w_1, v_1)$ respectively $e' = (w_1, v_2)$, otherwise $e' = e$.

A cycle in G' that uses exactly one edge between two neighbors of w can be transformed to a cycle in G that visits w. Furthermore, a cycle in G' that uses two edges between two neighbors of w, i.e. visits w_1 in between those two edges, and still visits the new vertices v_1 and v_2 can again be transformed to a cycle in G that visits w. Therefore, one can verify that the stability of (C, e) in G and its implied maximality imply that the new FE-cycle (C', e') is also stable.

Note, that the transformations in all three cases above are planarity preserving, see also Figure 1. Furthermore, the first cases only remove one vertex and only the last case removes one vertex and adds two new vertices which results in $|V(G')| \leq |V(G)| + 1$. Applying this transformation in sequence to all unvisited vertices $V(G) \setminus V(C)$ in G gives us then a graph G' and an SFE-cycle (C', e') that satisfy all needed properties.

Proposition 2.2 and Theorem 2.3 together give us that if we find a planar graph with minimum degree three having a dominating SFE-cycle this would disprove the conjecture by Bondy and Jackson. To search for such graphs we want to develop algorithms that can check for a graph if it contains a dominating SFE-cycle, i.e. we want to solve the following problem.

Problem 1. Given a graph G, does G contain a dominating SFE-cycle?

But we can go one step further and even relax the dominating condition. To do that we need to restrict us to 2-connected graphs, which is not really a restriction since every graph with minimum degree 3 with a dominating cycle must be 2-connected. The next theorem presents a planarity-preserving transformation from a 2-connected SFE-graph with minimum degree three to a graph with minimum degree three that contains a dominating SFE-cycle.

Proposition 2.4. *Given a 2-connected graph G with minimum degree three that contains an SFE-cycle C. Then there exists a 2-connected graph G' with minimum degree three that contains a dominating SFE-cycle and $|V(G')| \leq |V(G)|$. If C is not dominating then $|V(G')| < |V(G)|$. Furthermore, if G is planar then G' is planar.*

Proof. If C is dominating we can choose $G' = G$ and are done. Otherwise, let H be a non-trivial connected component of $G \setminus V(C)$. We contract now H in G to one vertex w. The 2-connectedness of G implies now that w has degree at least two. If the degree is two we can replace w and its two incident edges by one edge and get again a graph with minimum degree three. Furthermore, every cycle in this new graph that contains this new edge can be transformed to a cycle in G that contains a path through the connected component of H. If the degree of w is larger than three we get also a graph with minimum degree three and again every cycle that contains w in the new graph can be transformed to a cycle in the original graph containing a path through the connected component of H. This implies that the FE-cycle (C, e) is in both cases also stable in the new graph.

Applying the above procedure for every non-trivial connected component of $G \setminus V(C)$ we get a graph where (C, e) is dominating and still stable. Since we only contracted connected subgraphs in the above procedure, we get that $V(G') < V(G)$ and that G' is planar if G is planar.

Let us consider now the following conjecture.

Conjecture 1. There does not exist a 2-connected planar SFE-graph with minimum degree three.

Proposition 2.2, Theorem 2.3, and Proposition 2.4 imply the following theorem.

Theorem 2.5. *Conjecture 1 is equivalent to Bondy and Jackson's conjecture.*

To search for such graphs we need to check if a given graph is an SFE-graph, i.e. solve the following problem.

Problem 2. Given a 2-connected graph G, is G an SFE-graph?

3. Algorithmic Approaches

In this section we present two approaches, one based on integer linear programming and one based on cycle enumeration. For the ILP based approach we will present a basic method and some algorithmic improvements for solving it. The algorithmic improvements and also the cycle enumeration approach will use a data structure for storing and querying the found cycles, which we will also present in this section. Nevertheless, the problem can also be solved with the basic ILP approach, which does not use the data structure. This is important for verifying the results independently.

3.1. Basic ILP Approach

We simply check for each edge $e \in E(G)$ if there is any SFE-cycle with the fixed edge e. To do that we search for a maximal cycle containing the edge e, avoiding all cycles found until now. Whenever we find an FE-cycle we check if it is stable or not. This procedure is repeated until no new FE-cycle is found.

An ILP model is used for finding new cycles and another one for checking if a cycle is stable. We denote the set of all already found cycles for the edge e by $\mathcal{C}$. Furthermore, for a set of vertices $V' \subseteq V(G)$ we denote by $\delta(V')$ the set of all edges that are incident with one vertex in V' and one vertex outside V'. Similarly, for $v \in V$ we denote by $\delta(v)$ the set of incident edges of v. Moreover, let w be a fixed end vertex of e. The following ILP model is for finding new cycles.

$$\max \sum_{v \in V(G)} x_v \tag{1}$$

$$\text{s.t.} \sum_{e \in \delta(v)} y_e = 2x_v \qquad \forall v \in V(G) \tag{2}$$

$$\sum_{e \in \delta(V')} y_e \geq 2x_v \qquad \forall \emptyset \neq V' \subseteq V(G) \setminus \{w\}, v \in V' \tag{3}$$

$$y_e = 1 \tag{4}$$

$$\sum_{v \in V(G) \setminus V(C)} x_v \geq 1 \qquad \forall C \in \mathcal{C} \tag{5}$$

$$y_e, x_v \in \{0,1\} \qquad \forall e \in E(G), \forall v \in V(G) \tag{6}$$

The boolean x-variables indicate if a vertex is part of the cycle or not and the boolean y-variables if an edge is part of the cycle or not. To ensure that each found cycle is maximal we maximize the number of vertices in the cycle, see (1). We use here a classical approach for modeling cycles in an undirected version: For eliminating subtours we use cut constraints originally proposed for the TSP by Dantzig et al. [14], see (3). As it is standard with this approach the exponential many subtour elimination constraints get added to the program dynamically in a branch-and-cut manner using a fast max-flow algorithm for separating cuts in the LP-relaxation. Constraint (4) ensures that e is part of the cycle and constraints (5) guarantee that we find a new maximal cycle that was not found before.

Note that for similar problems models based on directed edge variables frequently result in better performance than the undirected variant, since the directed variant has a tighter LP-relaxation [12]. However, as we will see in Section 6 the graph sizes we are mainly concerned with are so small that the ILP models get solved fast, and our experiments indicated that the overhead of a directed model does not pay off in our case.

For each cycle C we find with this model we solve the following ILP model to check if the cycle is stable. This time we do not need the x-variables since we already know which vertices must be part of the cycle. Therefore, we can use any model for finding hamiltonian cycles, such as the following. Let G' be the graph induced by the vertex set $V(C)$.

$$\max 0 \tag{7}$$

$$\text{s.t.} \sum_{e \in \delta(v) \cap E(G')} y_e = 2 \qquad \forall v \in V(C) \tag{8}$$

$$y_e = 1 \tag{9}$$

$$\sum_{e \in \delta(V')} y_e \geq 2 \qquad \forall \emptyset \neq V' \subseteq V(C) \setminus \{w\} \tag{10}$$

$$\sum_{e \in E(G') \setminus E(C)} y_e \geq 2 \tag{11}$$

$$y_e \in \{0,1\} \qquad \forall e \in E(G) \tag{12}$$

Note that this model has no meaningful objective function as we are interested just in the fact whether or not a feasible solution exists. We use again cut constraints for eliminating subtours, see (10). To ensure that the cycle is different from the already found cycle C we ensure with Constraint (11) that the new cycle contains edges that do not appear in the old cycle.

This basic approach needs to solve many ILP models to check if a given graph contains an SFE-cycle. Although the basic approach has an overhead for each ILP model it needs to solve, it is still important since we can use it to check our results and especially to verify the correctness of the enumeration approach.

To improve this basic approach we want to add in an advanced approach constraints like (5) dynamically. This results in cycles that may not be maximal. The following data structure helps us to store cycles and detect non-maximal cycles.

3.2. Data Structure For Storing Cycles

To be able to check if a new found cycle C is a potential maximal cycle for some fixed edge we need to check if there exists for each edge $e \in E(C)$ an already found cycle C' that contains e and $V(C) \subsetneq V(C')$. To that end we will store each cycle we find in an appropriate data structure. Furthermore, the data structure will help us to search for SFE-cycles in the end.

The above query problem is an extension of the containment query problem, which is a widely studied problem [15, 16, 17]. For every known data structure for these problems it holds that either the query time is in $\Omega(N)$ or storage size is in $2^{\Omega(n)}$ where N is the number of stored sets and n is the alphabet size, which is in our case the number of vertices of the graph. If this must hold is an open question, although some theoretical lower bounds for computation times or storage size got already established [18].

Since we consider an extension of this problem, we will focus on a simple solution based on the Hasse diagram [19]. Furthermore, since we build up the data structure during the execution of our algorithm we have to balance the query times and the size of the data structure. Because we are searching for maximal cycles and in the case of Problem 1 for dominating cycles the vertex complements of such cycles tend to be small sets. Therefore, we focus on the vertex complements $V(G) \setminus V(C)$ in our data structure. The worst case size of the Hasse diagram is $\mathcal{O}(N^2)$, which is in our case normally much smaller than $2^{\Omega(n)}$, especially if we only consider dominating cycles.

Additionally, to the Hasse diagram of all vertex complements of found cycles we store a hash map mapping vertex complements X and edges e to the number of found cycles $m(X, e)$ with vertex complement X containing the edge e. If $m(X, e) \geq 2$ we know that cycles with the vertex complement X cannot be SFE-cycles for the edge e. Furthermore, we use a special value of $m(X, e) = \infty$ if there exists a cycle with a smaller vertex complement $Y \subsetneq X$ that contains e, in this case no cycle with the vertex complement X can be a maximal FE-cycle for the edge e.

For each new cycle C we first check if its vertex complement X is already in the Hasse diagram, if not we add it. Then we adapt $m(X, e)$ for all edges e in the cycle. Furthermore, we need to search for all $Y \subsetneq X$ inside the Hasse diagram and set $m(X, e) = \infty$ for all e where $m(Y, e) \neq 0$. Note that we only need to search for direct neighbors in the Hasse diagram since if $Y_1 \subsetneq Y_2 \subsetneq X$ we know that $m(Y_1, e) \neq 0$ implies $m(Y_2, e) = \infty \neq 0$. Last but not least we need to search for all $Z \supsetneq X$ inside the Hasse diagram and set $m(Z, e) = \infty$ for all edges $e \in E(C)$.

Whenever we have a vertex complement X for which all values of $m(X, e)$ for all edges e are either 0 or ∞ we can remove it, since the corresponding cycles are all not relevant anymore. This helps us to keep the size of the data structure small, which is relevant if we find many not maximal cycles. After we added all found cycles we can use this data structure to check if there is any SFE-cycle by simply checking if for any vertex complement X and edge e we have $m(X, e) = 1$.

3.3. Advanced ILP Approach

We will use now the previously presented data structure to improve the performance of our ILP approach. First of all we do not want to search for cycles for each edge e separately, since we can reuse a cycle for all its edges. To that end we split the approach into two phases, the first phase searches for maximal cycles and the second phase checks if one of them is an FE-cycle for some fixed edge e. All cycles we find in the first phase get added to the data structure. The second phase searches then the data structure for cycles C with $m(X(C), e) = 1$ and checks if (C, e) is an SFE-cycle in the same way as we presented for the basic ILP approach in Subsection 3.1.

To search for new maximal cycles for arbitrary fixed edges we add new boolean variables z_e for $e \in E(G)$ to our model which describe that the edge e is a potential fixed edge. We then replace the constraints (4) by the following constraints where $\mathcal{C}$ represents the set of all so four found cycles:

$$z_e \leq y_e \qquad \forall e \in E(G) \tag{13}$$

$$\sum_{e \in E} z_e \geq 1 \tag{14}$$

$$\sum_{v \notin V(C)} x_v \geq z_e \qquad \forall C \in \mathcal{C}, \forall e \in E(C) : m(X(C), e) \geq 1. \tag{15}$$

Constraints (13) ensure that a fixed edge must always be part of the cycle and constraint (14) enforces that there is at least one fixed edge. Furthermore, constraints (15) guarantee that the newly found cycle is a new maximal cycle for the fixed edge e if z_e is active.

To further speed up the approach we treat constraints (15) as lazy constraints and solve the model only once instead of restarting the solver after each found cycle for the updated model. That means whenever we find a new cycle, i.e., an integer solution to the current model, we add for all edges e with $m(X(C), e) = 1$ the respective constraint (15) dynamically. Formally this results in the end in an infeasible model but by collecting all the cycles gathered during the execution we get a complete collection of all potential maximal cycles. We can then execute phase two as explained above to check if any of the gathered cycles in the data structure is an SFE-cycle for some edge.

Note that since we do not have a fixed edge e in the model, we have to specify what we use as vertex w in constraints (3). This vertex should always be part of the cycle. To that end we enumerate over all vertices in $V(G)$, starting with the first as w. Then we apply the first phase and afterwards we forbid this vertex in all following runs by forcing $x_v = 0$ and chose the next vertex as w. The successive runs will be much faster than the first ones since the model gets easier to solve and fewer cycles will be found with each forbidden vertex. After all this we apply phase two to the current data structure.

If we are searching for dominating cycles we can improve this by only looking at an edge $e \in E(G)$ and using the fact that one of the ends of this edge must be part of the cycle. Therefore, we only need to apply the first phase two times. Furthermore, we will see in Section 4 that if we search for a minimum counterexample we can identify vertices that must always be part of the cycle, see Proposition 4.4, and therefore can use such a vertex as w and have to apply the first phase only once.

3.4. Cycle Enumeration Approach

In the above ILP approach we directly searched for maximal cycles. The idea of the alternative cycle enumeration approach is to enumerate all cycles in the graph and let our data structure do the work of filtering out the maximal ones. To speed up the algorithm we try to identify as early as possible if a cycle may be maximal and do not add it to the data structure if this is not the case.

The basic idea is to build a path recursively and try all possibilities to extend it until we can close the path to a cycle. We keep track of the two ends of the path and all the edges in the graph that are only incident to unvisited vertices or ends. Then for the end vertex with fewer remaining incident edges we try every such edge as an extension of the path. We call an edge a closing edge if it connects the two end vertices, and we call the resulting cycle the corresponding cycle to this closing edge. Whenever we find a closing edge, we can add the corresponding cycle to our data structure. However, before we close a cycle we try all non-closing edges and check if at least one of them leads to at least one cycle. In this case we know that the corresponding cycle of the current closing edge cannot be maximal.

To further speed up this algorithm we use five different states for vertices: `unvisited`, `visited`, `end`, `fixed`, and `dropped`. The vertices within the path are `visited` and the two end vertices have state `end`. For a vertex that is not in the path and has at some point only one incident edge left we know that it cannot be part of the cycle. We set the state to `dropped` and remove its incident edge.

If we search for dominating cycles we know that one of the two end vertices of each edge must be part of the cycle. Whenever we drop a vertex we can mark its only remaining unvisited adjacent vertex as `fixed`. Furthermore, if at some point a fixed vertex gets dropped we know that this path can never lead to a dominating cycle. To check for a closing edge if the corresponding cycle is dominating we keep during the whole algorithm track of the number of edges for which none of the two end vertices is part of the current path. A closing edge corresponds to a dominating cycle if and only if this number is zero at this point.

What remains is to specify with which vertex we start the algorithm. To do that we can iterate over all possible starting vertices in the same way as we described at the end of Subsection 3.3 and apply the algorithm to each of them. Whenever we used a start vertex and applied the algorithm we forbid it by setting its status to `dropped` in the following runs.

4. Properties of a Minimal Counter Example

In this section we will discuss some properties a minimum planar, 2-connected SFE-cycle graph with minimum degree 3, i.e. a minimum counterexample to Conjecture 1, must have. By minimum we mean w.r.t. the following relation:

$$G = (V, E) \leq G' = (V', E') \Leftrightarrow |V| < |V'| \vee (|V| = |V'| \wedge |E| \leq |E'|)$$

For the whole section let $G = (V, E)$ be a minimum counterexample and let (C, e) be the SFE-cycle. The following notations will be useful.

Definition 4.1. A vertex v of G is called a *small vertex* if its degree $d(v) \leq 3$ and if it is not incident to the fixed edge e. Otherwise, a vertex v is called a *large vertex.* If a large vertex has a degree larger than three it is called a *really large vertex* and otherwise it is called a *fixed large vertex.*

A vertex in $V(C)$ is called a *cycle vertex* and a vertex in $V \setminus V(C)$ is called an *outer vertex.*

We write $V^{\mathrm{l}}(G)$ for the set of all large vertices in G and $V^{\mathrm{s}}(G)$ for the set of all small vertices in G. Furthermore, we write $V^{\mathrm{lc}}(G)$, $V^{\mathrm{lo}}(G)$, $V^{\mathrm{sc}}(G)$, $V^{\mathrm{so}}(G)$ for the set of all large cycle vertices, large outer vertices, small cycle vertices, and small outer vertices, respectively.

The following statement about the maximal FE-cycle C follows directly from Proposition 2.4.

Corollary 4.2. *C is dominating.*

Furthermore, we can prove the following connectedness result.

Proposition 4.3. *G is 3-connected.*

Proof. Assume G is not 3-connected. Hence, there are two vertices v and w such that $G \setminus \{v, w\}$ is not connected. Let V_1 be the vertex set of a component of $G \setminus \{v, w\}$ that does not contain the fixed edge. Let H be the subgraph of G induced by $V_1 \cup \{v, w\}$. We consider the graph $G' = H \cup vw$. Since G is 2-connected we know that the degree of v and w in G' is at least two. If the degree of v in G' is two we can consider the other neighbor $x \neq w$ of v in G'. We get that $V_1 \setminus x$ is a nonempty component of $G \setminus \{x, w\}$. It is nonempty since otherwise x would have only the neighbors v and w in G, which contradicts the minimum degree of 3.

Therefore, we can choose w.l.o.g. the vertices v and w in such a way that the vertex set V_1 is minimal and therefore v and w have degree 3 or higher in G'.

Now, C together with the edge vw induce a cycle C' in G'. The stability of the FE-cycle (C, e) implies the stability of the FE-cycle (vw, C') in G'. But this is a contradiction to G being a minimum counterexample.

Proposition 4.4. *Every neighbor of a large vertex is in $V(C)$.*

Proof. Assume a neighbor w of a large vertex v is not in $V(C)$. Then we can remove the edge wv and still have the same SFE-cycle. If w has now degree 2 we remove it and replace it with an edge. Furthermore, if v was a fixed large vertex and has degree two now, we remove it and replace it with an edge, which is now the new fixed edge. This leads to a contradiction of G being a minimum counterexample.

Proposition 4.5. *Every edge between large vertices is in $E(C)$.*

Proof. An edge between large vertices that is not in $E(C)$ could be removed, which would result in a smaller graph that is planar, 2-connected, still has minimum degree 3, and still contains an SFE-cycle. This, however, contradicts the assumption that G is a minimum counterexample.

Corollary 4.6. *There is no large vertex with at least three large vertex neighbors in G and there is no cycle consisting only of large vertices in G.*

Proof. If there would be a large vertex with at least three large vertex neighbors, then there is at least one large neighbor whose connecting edge is not part of the cycle C, which is a contradiction to Proposition 4.5.

Furthermore, if there would be a cycle only of large vertices in G, then by Proposition 4.5 all its edges must be part of C, which means that it equals C. Let $v \in V(C)$ be a large vertex in this case. Then there exists one edge $e = vw$ incident to v that is not in $E(C)$. Since v is large we know by Proposition 4.4 that $w \in V(C)$ and therefore by our assumption that w is a large vertex. But this is a contradiction to Proposition 4.5.

Lemma 4.7. *The number of small vertices $|V^{\mathrm{s}}(G)|$ is equal to*

$$\sum_{v \in V^{\mathrm{lc}}(G)} (d(v) - 2) + \sum_{v \in V^{\mathrm{lo}}} d(v) + 2\,|\{vw \in E(G) \setminus E(C) : v, w \in V^{\mathrm{sc}}(G)\}| + 4|V^{\mathrm{so}}(G)|.$$

Proof. Each small cycle vertex w has exactly one incident edge vw that is in $E(G) \setminus E(C)$. This means that counting the small cycle vertices is the same as counting the half edges that are in $E(G) \setminus E(C)$ and are connected to a small cycle vertex. There are now three possibilities for v. Either it is in $V^{\mathrm{o}}(G)$, in $V^{\mathrm{lc}}(G)$, or in $V^{\mathrm{sc}}(G)$.

A vertex v in $V^{\mathrm{lc}}(G)$ has $d(v) - 2$ incident edges that are not part of the cycle. By Proposition 4.4 and Proposition 4.5 those edges must be incident to a small cycle vertex. Therefore, each large cycle vertex v is connected to $d(v) - 2$ small cycle vertices via edges in $E(G) \setminus E(C)$.

Moreover, an outer vertex v must be connected to $d(v)$ small cycle vertices, clearly via edges in $E(G) \setminus E(C)$, by Proposition 4.4 and Corollary 4.2.

Adding this up over all outer vertices and large cycle vertices gives us the number of small cycle vertices that are either connected to a large cycle vertex or to an outer vertex via an edge in $E(G) \setminus E(C)$. The remaining small cycle vertices must now all be connected to exactly one other small cycle vertex via an edge in $E(G) \setminus E(C)$. Therefore, the number of such remaining cycle vertices equals

$$2\,|\{vw \in E(G) \setminus E(C) : v, w \in V^{\mathrm{sc}}(G)\}|\,.$$

What remains is now to add $|V^{\mathrm{so}}|$ to the sum to get the number of all small vertices, which results in the theorem's statement.

For the next statement we introduce the notion of very small vertices and pseudo-matchings.

Definition 4.8. A small vertex is called *very small* if all its neighbors are small.

Definition 4.9. A *pseudo-matching* in a graph G is a subgraph where each connected component is either a K_2 or a $K_{1,3}$. A *maximum pseudo-matching* in a graph G is a pseudo-matching M in G that maximizes the number of vertices $|V(M)|$.

Lemma 4.10. *Let G^{vs} be the graph induced by all very small vertices in G and M^{vs} a maximum pseudo-matching in G^{vs}. Then it holds that*

$$2\,|\{vw \in E(G) \setminus E(C) : v, w \in V^{\mathrm{sc}}(G)\}| + 4|V^{\mathrm{so}}(G)| \geq 2|V(G^{\mathrm{vs}})| - |V(M^{\mathrm{vs}})|.$$

Proof. We begin by defining a pseudo-matching M in G^{vs}. For each very small outer vertex v for which all three neighbors are also very small we add the claw with center v to the pseudo-matching M. Furthermore, for each very small outer vertex v that has at least one very small neighbor and at least one not very small neighbor we add an edge between v and one of its very small neighbors to M. Last but not least, we add for all very small cycle vertices that are connected to other very small cycle vertices via an edge e in $E(G) \setminus E(C)$ this edge e to M.

Note now that

$$X := 2\,|\{vw \in E(G) \setminus E(C) : v, w \in V^{\mathrm{sc}}(G)\}| + 4|V^{\mathrm{so}}(G)|$$

is equal to the number of small vertices that are not connected to a large vertex by any edge in $E(G) \setminus E(C)$. By the construction of our pseudo-matching M there are three different types of very small vertices in $V(G^{\mathrm{vs}}) \setminus V(M) = V_1 \cup V_2 \cup V_3$. Let V_1 be all very small cycle vertices that are connected to another small but not very small cycle vertex via an edge in $E(G) \setminus E(C)$. Furthermore, let V_2 be all very small outer vertices that are connected to three small but not very small cycle vertices. Finally, let V_3 be all very small cycle vertices that are connected to a very small outer vertex that has exactly two very small neighbors and the edge connecting the other very small neighbor is part of M.

We define now for each vertex in $v \in V(G^{\mathrm{vs}}) \setminus V(M)$ an additional small cycle vertex w_v that is not very small but also not connected to a large vertex by any edge in $E(G) \setminus E(C)$. Furthermore, those vertices w_v are all different for different vertices $v \in V(G^{\mathrm{vs}}) \setminus V(M)$.

For $v \in V_1$ we define w_v to be the small neighbor of v via an edge in $E(G) \setminus E(C)$. For $v \in V_2$ we define w_v to be any neighbor of v and for $v \in V_3$ we define w_v to be the third neighbor of the outer vertex that is not a very small vertex.

With that we get that

$$\begin{aligned} X &\geq |V(G^{\mathrm{vs}}) \cup \{w_v : v \in V(G^{\mathrm{vs}}) \setminus V(M)\}| \\ &= |V(G^{\mathrm{vs}})| + |V(G^{\mathrm{vs}} \setminus V(M)\}| = 2|V(G^{\mathrm{vs}})| - |V(M)|. \end{aligned}$$

The Lemma follows now from the fact that M^{vs} is a maximum pseudo-matching and therefore by definition $|V(M)| \leq |V(M^{\mathrm{vs}})|$.

Theorem 4.11. *Let G^{vs} be defined as in Lemma 4.10 and $C^{\mathrm{vs}}_{\mathrm{odd}}$ be the number of connected components in G^{vs} with an odd number of vertices. Then*

$$|V^{\mathrm{s}}(G)| \geq \sum_{v \in V^1(G)} (d(v) - 2) + |V(G^{\mathrm{vs}})| + C^{\mathrm{vs}}_{\mathrm{odd}},$$

holds, which is equivalent to

$$|E(G)| \leq |V^{\mathrm{s}}(G)| + |V(G)| - \frac{1}{2}(|V(G^{\mathrm{vs}})| + C^{\mathrm{vs}}_{\mathrm{odd}}).$$

Proof. The first statement follows from Lemma 4.7, Lemma 4.10, and the fact that the connected components of a pseudo-matching always have an even number of vertices and therefore

$$|V(M^{\mathrm{vs}})| \leq |V(G^{\mathrm{vs}})| - C^{\mathrm{vs}}_{\mathrm{odd}}.$$

To see the equivalence with the second statement we do the following equivalence transformations.

$$\begin{aligned} & |V^{\mathrm{s}}(G)| \geq \sum_{v \in V^1(G)} (d(v) - 2) + |V(G^{\mathrm{vs}})| + C^{\mathrm{vs}}_{\mathrm{odd}} \\ \Leftrightarrow\ & |V^{\mathrm{s}}(G)| \geq \sum_{v \in V(G)} (d(v) - 2) - |V^{\mathrm{s}}(G)| + |V(G^{\mathrm{vs}})| + C^{\mathrm{vs}}_{\mathrm{odd}} \\ \Leftrightarrow\ & 2|V^{\mathrm{s}}(G)| \geq 2|E(G)| - 2|V(G)| + |V(G^{\mathrm{vs}})| + C^{\mathrm{vs}}_{\mathrm{odd}} \\ \Leftrightarrow\ & |E(G)| \leq |V^{\mathrm{s}}(G)| + |V(G)| - \frac{1}{2}(|V(G^{\mathrm{vs}})| + C^{\mathrm{vs}}_{\mathrm{odd}}). \end{aligned}$$

An important property of a minimum counterexample, which helps much in searching for counterexamples, is the following.

Theorem 4.12. *G is triangle-free.*

Proof. Assume G contains a triangle T. We distinguish whether G can get embedded into the plane such that T is a 3-face in this embedding or not. If there is no such embedding, we know that the removal of T would split G into two components. Since G is 3-connected, there must be an edge from every vertex of T to each component. This implies that all vertices of T are large, which is a contradiction to Corollary 4.6.

In the other case we know that there is an embedding of G such that T is a 3-face. Let v, w, x be the three vertices of the 3-face T. We distinguish now two cases.

1. T does not contain the fixed edge e. We distinguish again three cases.
   (a) If all three vertices are small, we can contract them all into one vertex of degree three. Every maximal FE-cycle in the original graph corresponds now to a maximal FE-cycle in the new graph and vice versa. This implies that the corresponding cycle $(\tilde{C}, e)$ in the new graph is again an SFE-cycle with the same fixed edge e as in the original graph. This is a contradiction to G being a minimum counterexample.
   (b) If one vertex is large and the other two are small, we say w.l.o.g. the large vertex is v and we can shrink the two small vertices w and x together. There are now again two cases. If the cycle C in the original graph contains the path w, v, x or x, v, w, then the new graph has an SFE-cycle that visits all vertices of C (including the new merged vertex instead of w and x) except v. If the cycle C contains the path w, x, then (C, e) transforms into an SFE-cycle containing the same vertices as C (except the new merged vertex instead of w, x).
   (c) If two vertices are large and one is small, we get by Proposition 4.5 that the edge between the two large vertices is used by the cycle C. If we remove now this edge there is clearly no cycle containing all vertices of C anymore. Moreover, C transforms into a cycle containing all vertices except one of the large vertices, call it v. This cycle must be stable since every cycle that contains all vertices of C except v can be transformed into a cycle containing all vertices in the original graph.
   (d) If all vertices of the triangle are large, we get by Corollary 4.6 a direct contradiction.
2. If T contains the fixed edge, w.l.o.g. let vw be the fixed edge. In this case v and w are large vertices by definition and therefore by Corollary 4.6 we get that x is a small vertex and by Proposition 4.4 that x is in C. Let u be the third neighbor of x, then the edge xu must be part of C. We merge now the triangle to one new vertex z and consider the to C corresponding cycle $\tilde{C}$ together with the fixed edge zu. Every cycle containing the edge zu corresponds to a cycle in the original graph containing all three vertices of T and the edge vw. This implies that $(\tilde{C}, \tilde{e} = zu)$ is an SFE-cycle in the new graph, which is a contradiction to G being a minimum counterexample.

The next theorem shows an even stronger connectedness of a minimum counterexample.

Theorem 4.13. *G is essentially 4-edge connected.*

Proof. Assume G contains three edges e_1, e_2, e_3 whose removal disconnect G into two non-trivial components G_1, G_2. Assume w.l.o.g. that the fixed edge e is not in G_1. The three vertices v_1, v_2, and v_3 in G_1 with v_i incident to e_i have degree at least two in G_1.

Since C is dominating and since G_1 and G_2 are both non-trivial we know that C must use two of the three cut edges. W.l.o.g. let C use the edges e_1 and e_2. The cycle C corresponds to a path P in G_1 going from v_1 to v_2.

If we consider the graph G' consisting of G_1 together with an edge connecting v_1 with v_2 (if they are not already connected) we can extend P with this new edge to an FE-cycle C' in G' by fixing the edge between v_1 and v_2. We distinguish now two cases, if v_3 is in $V(C')$ or not.

If v_3 is not in $V(C')$ we know that C' is stable since, if there would exist another FE-cycle with the same fixed edge that contains the same or more vertices we could remove the fixed edge from it and get a different path P' in G_1. We could then replace the path P in C with P' and get a contradiction to the stability of C. The graph G' may contain vertices of degree two. If v_1 or v_2 have degree two we can replace them in G' with an edge, which will then be the new fixed edge. Furthermore, if v_3 has degree two we can simply replace it with an edge since v_3 is not used in the cycle C'. Note that G' must contain at least 4 vertices since otherwise we could easily find a cycle that also contains v_3, which would be a contradiction to the maximality of C'. Therefore, after the removals described above the remaining graph is still not empty, it has minimum degree three, is planar, is smaller than G, and has an SFE-cycle, which is a contradiction.

The remaining case is that v_3 is in $V(C')$. Here G_1 must contain a path P_1 from v_2 to v_3 containing at least all the vertices of P. If there would not exist such a path we could contract G_2 in G to a vertex x and fix the edge from x to v_2. Then the cycle consisting of P and the edges v_1x and v_2x is again stable. This can be seen since the only other possibility would be to use the edges v_2x and v_3x but then the remaining path after removing those two edges would have the properties of P_1, which we assumed does not exist.

With the same argumentation we can prove the existence of a path P_2 from v_1 to v_3 in G_1 that contains at least all the vertices of P. We contract now the component G_1 in G to a vertex x and get a new graph G'. Clearly, the cycle C corresponds to a new cycle C' using the edges v_1x and v_2x. The fixed edge of C' is still the same fixed edge as the one of C (which was not in the contracted G_1). If there would exist now a second cycle containing at least the same vertices as C' in G' this second cycle must use either v_1x and v_3x or it uses v_2x and v_3x, since otherwise it would contradict the stability of C. But if such a cycle exists we could replace the two edges with either P_1 or P_2 and would get a cycle in the original graph G, which would again contradict the stability of C.

The following corollary summarizes now all properties for a minimum counterexample G if we do not know the FE-cycle (C, e). Since we do not know the fixed edge we define for the corollary that the small vertices are exactly the vertices of degree 3 and the large vertices are the vertices of degree larger than 3.

Corollary 4.14. *Let $G = (V, E)$ be a minimum planar, 2-connected SFE-graph with minimum degree 3. Furthermore, let G^{vs} be the graph induced by all very small vertices in G. Then the following properties hold.*

- *G is 3-connected.*
- *G is essentially 4-edge connected.*
- *G contains no triangles.*
- *There is no large vertex in G with at least three large vertex neighbors.*
- *There is no cycle of large vertices in G.*
- *Let $C^{\mathrm{vs}}_{\mathrm{odd}}$ be the number of connected components in G^{vs} with an odd number of vertices, then it holds that*
$$|E(G)| \leq |V^{\mathrm{s}}(G)| + |V(G)| - \frac{1}{2}(|V(G^{\mathrm{vs}})| + C^{\mathrm{vs}}_{\mathrm{odd}}).$$

Corollary 4.14 summarizes all the properties a graph must satisfy, but we can also consider the properties a potential SFE-cycle must satisfy such as Corollary 4.2 and Proposition 4.4. To search for a minimum SFE-graph it is enough to solve for each graph satisfying the properties from Corollary 4.14 the following problem.

Problem 3. Given a graph G, does G contain a dominating SFE-cycle (C, e) containing all neighbors of large vertices of G?

Note, that the algorithms presented in Section 3 can all also solve Problem 3. For the ILP based approaches we can fix neighbors of large vertices by forcing the corresponding x-variables to one and for the enumeration approach we can set the status of neighbors of large vertices to `fixed` at the beginning of the algorithm.

5. Systematic Search for a Minimal Counter Example

In this section we describe how we use the results from Corollary 4.14 to do a computationally effective search for a minimum counterexample. We use plantri [11], a tool for generating exactly one representant of all isomorphism classes of planar graphs of a given order. It furthermore can efficiently create only 3-connected graphs with a given minimum degree.

Our first approach was to directly generate representants of 3-connected planar graphs and then filter them by the properties from Corollary 4.14. This results in a bad performance since the filtering will be the bottleneck due to the restrictive condition of triangle freeness. To be able to incorporate triangle freeness into the generation process we generate representants of the dual graphs instead of the original graphs.

Note, that for 3-regular planar gaphs the dual is unique and again 3-regular. We can then translate the triangle freeness of the original graphs to minimum degree four in the dual graph, which can be handled by plantri in a performant way. Note that there might still be triangles in the resulting graph that are not faces, but those would lead to a triangle of large vertices, which is also forbidden by the conditions in Corollary 4.14 and therefore get filtered out afterwards.

In order to generate all candidate graphs with up to n vertices we need to generate all dual graphs with up to $n-2$ vertices. This can be seen by the fact that $|E(G)| \geq 2|F(G)|$ in triangle free planar graphs and therefore

$$|F(G)| = |E(G)| - |V(G)| + 2 \geq 2|F(G)| - |V(G)| + 2 \Rightarrow |F(G)| \leq |V(G)| - 2.$$

Furthermore, we get an upper bound for the number of edges by

$$|E(G)| = |F(G)| + |V(G)| - 2 \leq |F(G)| + n - 2.$$

To summarize the total process given a maximum target graph size n, we let n^* iterate from 8 to $n-2$ and use plantri to generate all 3-connected planar graphs with minimum degree 4, n^* vertices, and at most $n^* + n - 2$ edges. Then we build the duals of all generated representants and filter them by the conditions in Corollary 4.14. The resulting graphs are all possible candidate graphs with size up to n. Then we can use any of our three approaches to check if one of them contains a candidate SFE-cycle by solving Problem 3.

Note the step of building the duals can be parallelized in plantri by splitting the search space into junks, which are of reasonable similar sizes. Therefore, we can parallelize the whole process, which we will heavily use to get our computational results.

6. Computational Results

In this section we present computational tests and results of our algorithms in a two-folded manner. On the one hand we will use three structurally different classes of graphs and test and compare our approaches on them. On the other hand, we will use the best performing approach to systematically search for a minimum counterexample, which will computationally prove the main result of this work, Theorem 6.2.

All tests and computations are performed on machines with Intel Xeon E5-2640 v4 processors with 2.4GHz using at most 8GB RAM per thread. The tests in Subsection 6.2 are each done on a single core and for the results in Subsection 6.3 we made use of up to 320 cores splitted over 16 machines in a parallelized fashion. Furthermore, the implementations are done in C++ and for the ILP approaches we use Gurobi 8.1 to solve.

6.1. Instance Sets

We describe here three different instance sets I_1, I_2, and I_3 which we will use for testing our algorithms. The instance sets I_1 and I_2 are for the problems 1 and 2. The third instance set is for Problem 3.

The first set I_1 consists of up to 5000 random 3-regular planar graphs for each graph order from 6 to 59. Note, that for order 6 to 9 there are fewer than 5000 isomorphism classes and therefore we simply use a representant of each of them. For all other orders we sample 5000 different representants.

The second set I_2 gets constructed in the following way. For each order from 6 to 59 take up to 5000 different representants of random 3-regular planar graphs as before, then randomly insert into every edge of the graph a vertex of degree 2 with 2% probability. The probability was chosen in such a way that the resulting graphs with around 20 vertices contain SFE-cycles with a probability of around 50% which was evaluated experimentally. For larger graphs up to 59 vertices the probability of containing an SFE-cycle increases but the probability of containing a dominating SFE-cycle decreases, see also Figure 3.

The third set I_3 consists for each n^* from 8 to 59 of up to 5000 random graphs that satisfy all properties of Corollary 4.14, i.e. are possible candidates for a minimum counterexample, and have n^* faces.

To generate the instance sets we used plantri for creating graphs with the desired properties and selected with a given probability the graphs. For the sets of I_3 we used plantri on the dual graphs with minimum degree four. For the orders 6–14 in the case of I_1 and I_2 and for the orders 8–17 in the case of I_3 we enumerated all representants and randomly chose 5000 of them. For all other orders we used a feature from plantri that can partition the search space into sets of similar size and enumerate through one of those sets efficiently. Then we randomly select a set, generate graphs from this set and keep a graph with a small probability of $1/2\,000\,000$ in the case of I_1 and I_2 and $1/1\,000\,000$ in the case of I_3 until we have two graphs, or we enumerated the whole set. Then we start with another randomly selected set (same sets can get selected multiple times) until we have 5000 graphs not allowing duplicates.

Note that this procedure selects not uniform from the whole search space since the splitting of the search space by plantri is limited and there might be large sets in this partition. The probability of a graph depends on the order of enumeration of those sets, the graphs enumerated first have the highest selection probabilities and the graphs at the end the lowest. But for our purposes this biased selection is sufficient. To select truly uniformly for the three different search spaces would be non-trivial and presumably computationally far too expensive.

The instance sets and the source code of the implementations of our algorithms can be downloaded from `https://www.ac.tuwien.ac.at/research/problem-instances`.

6.2. Testing the Three Approaches on the Three Instance Sets

To verify the correctness of our approaches and to compare their performance we applied all three approaches on our instance sets. For instance sets I_1 and I_2 we test two problem variants, Problem 1, which searches for dominating SFE-cycles and Problem 2, which searches for any SFE-cycle. For graphs of the instance set I_3 we only consider the variant of Problem 3, which searches for potential SFE-cycles in a minimum counterexample. For each instance set, problem variant, and order n we use a running time limit of one hour to solve all the up to 5000 graphs of order n.

Figure 2 compares the average running times per graph of the three algorithms for the different graph orders and problem variants in seconds. Note that the graph orders in instance set I_3 denote the order of the dual graph and therefore the number of faces of the original graph. We only display data points for orders for which at least 30 graphs could get solved within the one hour time limit.

As we can see, our enumeration algorithm performs best on small graphs and does not scale well for larger graphs. Furthermore, for small graphs the advanced ILP approach performs better than the basic ILP approach. For larger graphs of the instance set I_1 and I_2 the basic ILP approach performs better than the advanced ILP approach. This can be explained by the fact that the basic ILP approach has to solve many more ILPs than the advanced ILP approach. For small graphs those ILP models are so easy to solve that the overhead of starting the solver outweighs the actual solving time. On the other hand, for larger graphs the models get harder to solve and the simplicity of the models in the basic ILP approach compared to the advanced ILP approach compensates the overhead of solving more models.

For the instance set I_3 the models stay easy to solve also for larger graphs since a lot of vertices can be fixed in the cycle due to the definition of Problem 3 and therefore the advanced ILP approach performs better than the simple ILP approach over all orders. Our enumeration performs better than the ILP approaches for searching for a minimum counterexample up to a dual graph order of 19. This will be important, when we want to search for a smallest counterexample.

Instance set I_2 is important to also test positive instances, i.e. graphs that contain SFE-cycles. Although the three algorithms could not solve all graphs in the given time limit, we compared the results of the solved graphs between the three approaches and can conclude that for each of the tested graphs the approaches

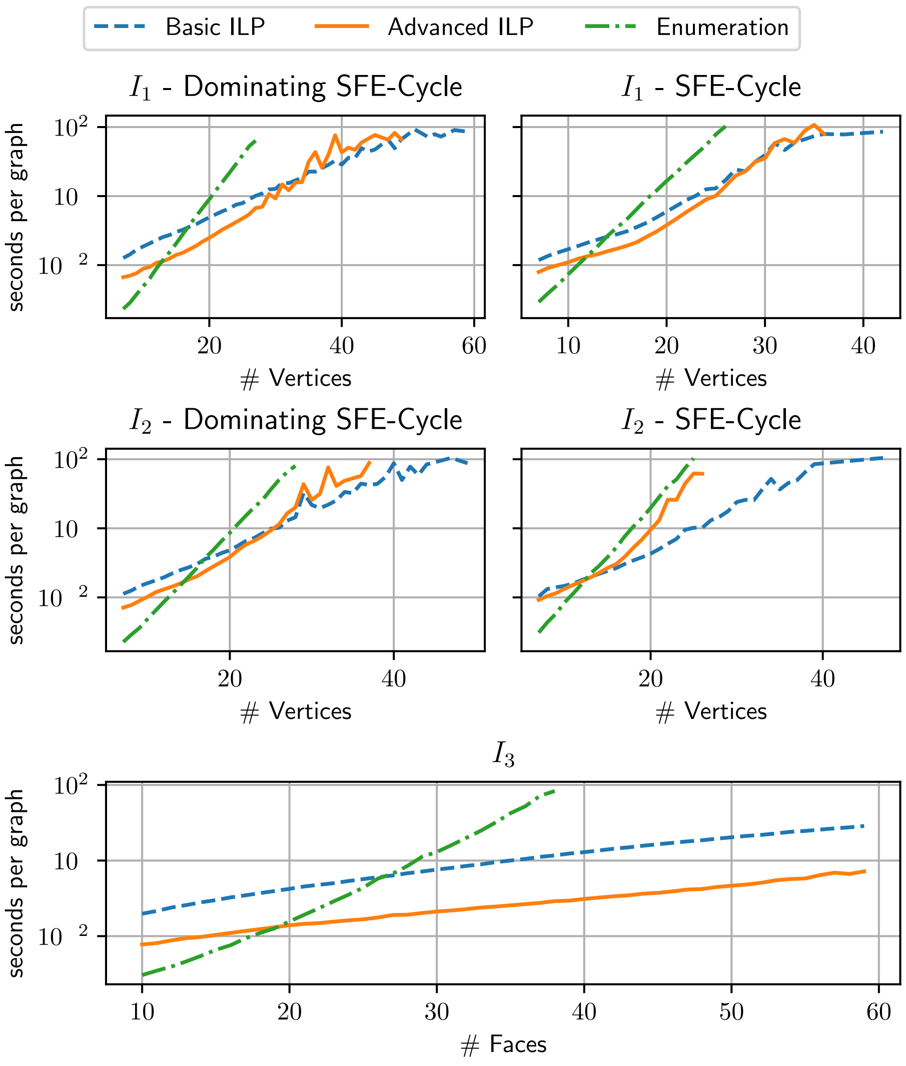

Figure 2: Average running time comparison of the three different algorithms on the three instance sets.

coincide in their decision between containing an SFE-cycle or not. It may be that the approaches find different SFE-cycles if they exist. In Figure 3 we see for the two different variants for instance set I_2 for each graph order how many graphs could get checked within the time limit and how many SFE-graphs were found by the basic ILP approach. The other two approaches were able to check only the same amount of graphs or less for each graph order.

To better understand the difference between the basic ILP approach and the advanced ILP approach we analyze the number of ILPs solved for each single graph. Figure 4 shows the number of ILPs solved per graph for each graph order and problem variant for the two different ILP approaches.

As we can see the basic ILP approach needs to solve many more models than the advanced approach. The difference is especially big for the instance set I_3 where the advanced approach needs to solve only up to 5 models per graph on average. The basic model needs to solve on average multiple hundred models per graph for the larger I_3 instances.

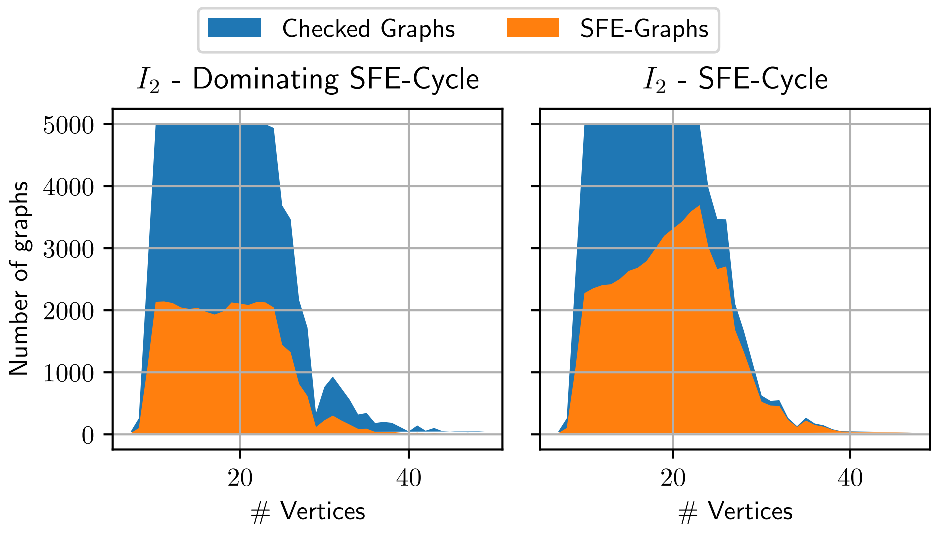

Figure 3: Number of tested graphs in instance set I_2 compared to number of found SFE-graphs.

6.3. Searching for Minimal Counter Examples with up to 24 Vertices

In this subsection we describe the results of applying the approach described in Section 5 to search for a minimum counterexample. As we saw in Subsection 6.2 the enumeration approach performs best for Problem 3 on graphs with up to 19 faces. We will see that the vast majority of candidate graphs with up to 24 vertices have 19 or fewer vertices and therefore we will use the enumeration approach for checking if any of the candidate graphs contain an SFE-cycle. Note that we also use the enumeration approach for graphs with more than 19 faces for simplicity. One could instead use the advanced ILP-approach for all graphs with 20 or more faces, which may be important if we want to go further and check graphs with more than 24 vertices.

Table 1 shows the results of our search. Each row lists the results for the generated graphs with a given number of vertices, which is displayed in the first column. Furthermore, the column *part.* gives the number of parallel threads used to generate the graphs, filter them and check them for SFE-cycles. The third and fourth column show the number of graphs that got checked for SFE-cycles by solving Problem 3 and the number of graphs that were filtered out by the properties of Corollary 4.14.

Since we are interested in the running time of the slowest thread, this is given in seconds by column *max t[s]*. The next three columns list summed running times of all threads in hours. Column *total* gives the total CPU time of everything together, generation, filtering, and checking. Moreover, columns *gen.* and *check* give the total CPU times for generation and checking respectively. As we can see the highest running time is used to generate graphs with 22 faces although only 600 000 such graphs get generated in the end and only 5 of them get really checked. This shows that the bottleneck of the current approach is not solving Problem 3 but to efficiently generate graphs satisfying the properties of Corollary 4.14.

The column *C/G* lists the average number of cycles the algorithm found per graph. Note that those cycles may not even be maximal and are simply added to the data structure for storing cycles to check at the end if some of them are SFE-cycles. Finally, the column *SFE* shows that none of the generated graphs contain any SFE-cycles.

Those computational results prove the following lemma.

Lemma 6.1. *There does not exist any planar graph with minimum degree three and at most 24 vertices that contains an SFE-cycle.*

With that we can finally prove our main theorem.

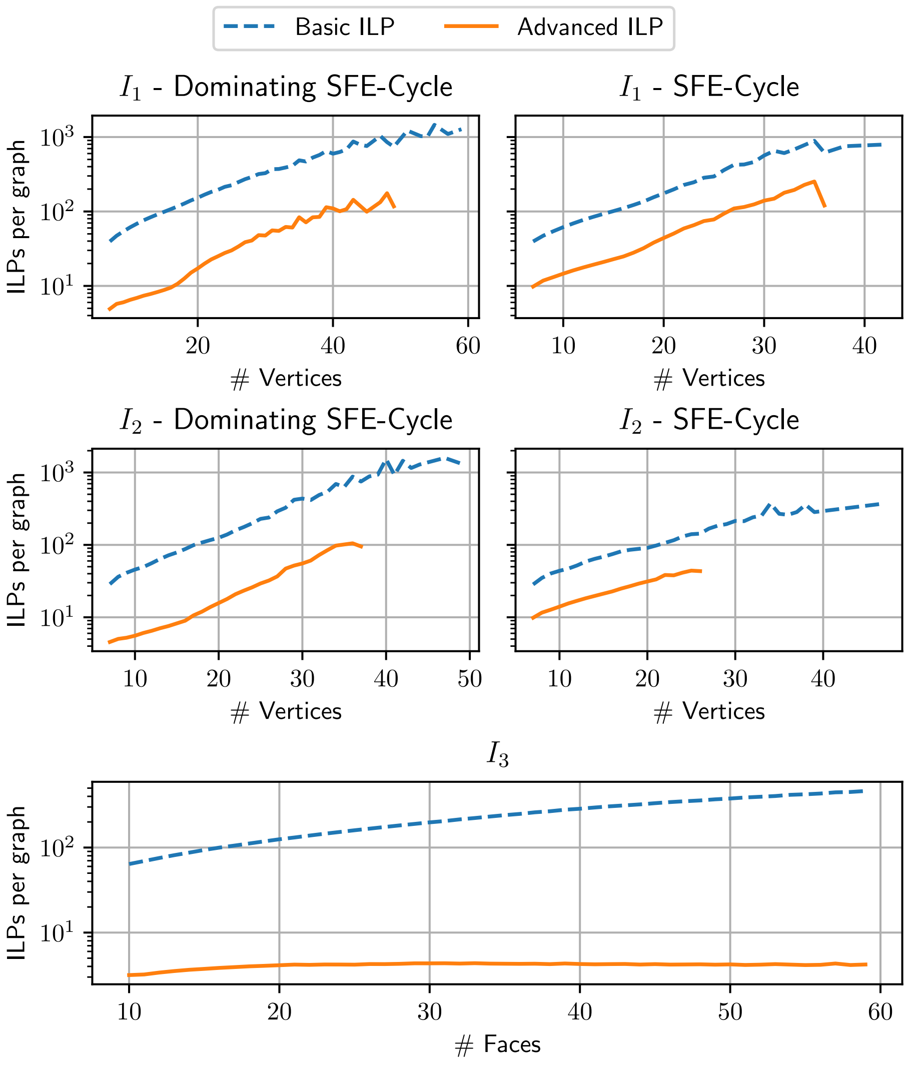

Figure 4: Number of ILPs solved per graph of the two different ILP approaches on the three instance sets.

Theorem 6.2. *There does not exist any planar graph with minimum degree three having at most 25 vertices that contains a stable cycle. This implies that Bondy and Jackson's conjecture holds for graphs with up to 25 vertices.*

Proof. Clearly, every uniquely hamiltonian cycle is a stable cycle and therefore we only need to prove the first statement. Assume that there exists a planar graph G with minimum degree three having at most 25 vertices that contains a stable cycle C. W.l.o.g. let G be a minimum by the relation defined at the beginning of Section 4. There are now two cases, either G is 3-regular or it contains a vertex v with degree at least four.

In the first case we know that C cannot be uniquely hamiltonian and therefore there must exist a vertex u which is not part of C. W.l.o.g. we can assume that all neighbors of u are part of C since otherwise we can contract the whole component of vertices which are not part of C. Note that this implies that the neighbors of u are not connected by an edge since this would imply that C is not maximal. What we do now is remove the vertex u and connect two of its neighbors with a new edge. What remains is a planar

Table 1: Results for Candidate Graphs with up to 24 Vertices.

		# Graphs			Total CPU times [h]				
#F	part.	checked	filtered	max t[s]	total	gen.	check	C/G	SFE
6	1	1	0	<1	<1	<1	<1	8.0	0
7	1	1	0	<1	<1	<1	<1	15.0	0
8	1	4	0	<1	<1	<1	<1	14.8	0
9	1	12	2	<1	<1	<1	<1	15.9	0
10	1	61	6	<1	<1	<1	<1	17.6	0
11	1	371	57	1	<1	<1	<1	18.6	0
12	1	2927	588	8	<1	<1	<1	21.0	0
13	1	24 957	6806	50	<1	<1	<1	23.9	0
14	32	228 793	78 750	184	<1	<1	<1	27.7	0
15	128	2 090 930	862 784	644	3	<1	3	29.2	0
16	512	14 750 689	7 746 354	1316	25	<1	24	25.1	0
17	512	45 728 911	57 554 063	2160	79	1	76	19.8	0
18	512	36 650 395	211 986 918	1486	71	2	67	16.2	0
19	512	6 061 407	283 793 146	328	27	13	14	14.2	0
20	512	201 991	145 836 208	1370	106	106	<1	13.2	0
21	512	1309	24 624 586	9464	1100	1100	<1	13.0	0
22	512	5	598 097	151 153	12 544	12 544	<1	20.8	0

graph containing a stable cycle with up to 24 vertices and one vertex w of degree two.

If, on the other hand, C is hamiltonian and G contains a vertex v with degree at least four we know that at least two of the incident edges to v are not part of the cycle C, let $e = vw$ be one of them. Because G is a minimum we know that w must have degree three since otherwise we could remove e and get a smaller graph with the same properties. We remove now the edge e from G and get a graph with up to 25 vertices and one vertex w of degree two containing a stable cycle.

We got in both cases a planar graph containing a stable cycle with exactly one vertex w of degree two having at most 25 vertices. In both cases we replace w and its two remaining incident edges by one new edge e' and get a new graph G', which has at most 24 vertices and minimum degree three. If w is not in $V(C)$ then C is also a stable cycle in G' and therefore also a SFE-cycle in G'. Otherwise, we can replace the edges e_1 and e_2 in C by the new edge e' and get a new cycle C' in G'. The stability of C in G implies now that (C', e') is an SFE-cycle in G'. Therefore, we get in both cases a contradiction to Lemma 6.1.

Another interesting implication of Lemma 6.1 concerns uniquely hamiltonian planar graphs with vertex connectivity two.

Theorem 6.3. *There does not exist any planar graph with minimum degree three and vertex connectivity two having at most 46 vertices that contains a stable dominating cycle.*

Proof. We assume there exists such a graph G with a stable dominating cycle C. Let v, w be a 2-vertex cut of G and C be the smallest component of $G \setminus \{v, w\}$. Furthermore, let v, w be in such a way that $|V(C)|$ is as small as possible. Clearly, $|V(C)| \leq 22$. We define now G' as the graph induced by the vertex set $V(C) \cup \{v, w\}$ and add the edge $e' = vw$ if it is not already part of G'.

Since C is dominating we know that C traverses v and w. We define C' by restricting C to G' and adding the edge vw to C', which gives us a dominating cycle in G'. The stability of C in G implies now the stability of the fixed-edge cycle (C', e') in G'. Furthermore, since we chose v, w in such a way that $|V(C)|$ is as small as possible we know that the degree of v and w in G' is at least three. Therefore, G' has minimum degree three, which is a contradiction to Lemma 6.1.

7. Conclusion

In this work we focused on systematically searching for uniquely hamiltonian planar graphs with minimum degree three. To this end we reduced the search to looking for planar graphs with minimum degree three that contain a stable fixed-edge cycle. The fundamental approach was to generate candidate graphs using the external tool plantri and then check for each of the generated graphs if they contain a stable fixed-edge cycle (SFE-cycle).

We formulated three different problem variants. They ask if a given graph contains an SFE-cycle, a dominating SFE-cycle, or a dominating SFE-cycle with some additional properites, respectively. We proved additional properties an SFE-cycle of a minimum counterexample must satisfy and use them in the problem variant three. To solve the problem variants we proposed three different approaches, two based on integer linear programming and one based on cycle enumeration.

To efficiently search for a minimum planar SFE-graph with minimum degree three we proved several properties which a minimum counterexample must satisfy, see Corollary 4.14 for a summary. The property which helps the most in reducing the search space is triangle freeness.

We tested our three algorithms on a selection of random instances for the three different problem variants. Most of the time the enumeration approach performs better for small graphs and the ILP based approaches for larger graphs. Finally, we generated all planar 3-regular triangle-free graphs with minimum degree three and up to 24 vertices using plantri to create the dual graphs. Then, we filtered the graphs using the properties of Corollary 4.14 and applied to the enumeration approach to the remaining graphs to check if they contain an SFE-graph. The computations for searching for a minimum counterexample were heavily parallelized and used a total of 1.59 CPU years with a bottleneck in generating all planar 3-regular triangle-free graphs with minimum degree three and up to 24 vertices.

The computational results show that no planar SFE-graph with minimum degree three with up to 24 vertices exist. This implies the main result of this work that no uniquely hamiltonian planar graph with minimum degree three with up to 25 vertices exist and no uniquely hamiltonian planar graph with minimum degree three with connectivity two with up to 46 vertices exist.

To be able to further increase the lower bound of 25 vertices in the future we have to focus on the fast generation of planar 3-connected triangle-free graphs with minimum degree three as this was the bottleneck in our search for a minimum counterexample. Furthermore, if we can solve this bottleneck, the usage of the presented ILP based approaches instead of the enumeration approach for the larger graphs will be crucial. If also those approaches are too slow, one could try to use constraint programming or SAT-based approaches and compare them to the ILP based approaches. Last but not least, there might be additional properties that we can prove for a minimum counterexample leading to a more effective search. This might also help to solve the bottleneck of generating candidate graphs.

Acknowledgments

This work was supported by the Austrian Science Fund (FWF) under grant P27615 and the Vienna Graduate School on Computational Optimization, grant W1260.

References

[1] W. T. Tutte, On hamiltonian circuits, Journal of the London Mathematical Society s1-21 (2) (1946) 98–101. `doi:10.1112/jlms/s1-21.2.98`.

[2] A. G. Thomason, Hamiltonian cycles and uniquely edge colourable graphs, Advances in Graph Theory 3 (1978) 259–268. `doi:10.1016/S0167-5060(08)70511-9`.

[3] J. A. Bondy, B. Jackson, Vertices of Small Degree in Uniquely Hamiltonian Graphs, Journal of Combinatorial Theory, Series B 74 (2) (1998) 265–275. `doi:10.1006/jctb.1998.1845`.

[4] S. Abbasi, A. Jamshed, A Degree Constraint for Uniquely Hamiltonian Graphs, Graphs and Combinatorics 22 (4) (2006) 433–442. `doi:10.1007/s00373-006-0666-z`.

[5] R. C. Entringer, H. Swart, Spanning cycles of nearly cubic graphs, Journal of Combinatorial Theory, Series B 29 (3) (1980) 303–309. `doi:10.1016/0095-8956(80)90087-8`.

[6] H. Fleischner, Uniqueness of maximal dominating cycles in 3-regular graphs and of hamiltonian cycles in 4-regular graphs, Journal of Graph Theory 18 (5) (1994) 449–459. doi:10.1002/jgt.3190180503.

[7] J. A. Bondy, Basic graph theory: paths and circuits, Handbook of combinatorics 1 (1995) 3–110.

[8] H. Fleischner, Uniquely hamiltonian graphs of minimum degree 4, Journal of Graph Theory 75 (2) (2014) 167–177. doi:10.1002/jgt.21729.

[9] J. Goedgebeur, B. Meersman, C. T. Zamfirescu, Graphs with few hamiltonian cycles, Mathematics of Computation. To appear (2019). doi:10.1090/mcom/3465.

[10] B. Klocker, H. Fleischner, G. R. Raidl, Finding Uniquely Hamiltonian Graphs of Minimum Degree Three with Small Crossing Numbers, in: Hybrid Metaheuristics: 10th International Workshop, HM 2016, Vol. 9668 of Lecture Notes in Computer Science, Springer, Plymouth, UK, 2016, pp. 1–16. doi:10.1007/978-3-319-39636-1_1.

[11] G. Brinkmann, B. D. McKay, Fast generation of planar graphs, MATCH Communications in Mathematical and in Computer Chemistry 58 (2) (2007) 323–357.

[12] L. A. Wolsey, Integer programming, Wiley, 1998.

[13] F. R. Leder, Uniquely hamiltonian graphs, Master's thesis, TU Wien (2012).

[14] G. Dantzig, R. Fulkerson, S. Johnson, Solution of a Large-Scale Traveling-Salesman Problem, Journal of the Operations Research Society of America 2 (4) (1954) 393–410. doi:10.1287/opre.2.4.393.

[15] M. Charikar, P. Indyk, R. Panigrahy, New algorithms for subset query, partial match, orthogonal range searching, and related problems, in: Automata, Languages and Programming, Springer, 2002, pp. 451–462. doi:10.1007/3-540-45465-9_39.

[16] I. E. Shanthi, Y. Izaaz, R. Nadarajan, On the SD-tree Construction for Optimal Signature Operations, in: Proceedings of the 1st Bangalore Annual Compute Conference, COMPUTE '08, ACM, New York, NY, USA, 2008, pp. 14:1–14:8. doi:10.1145/1341771.1341786.

[17] S. Bevc, I. Savnik, Using tries for subset and superset queries, in: Proceedings of the 31st International Conference on Information Technology Interfaces, 2009. ITI '09, 2009, pp. 147–152. doi:10.1109/ITI.2009.5196071.

[18] T. S. Jayram, S. Khot, R. Kumar, Y. Rabani, Cell-probe Lower Bounds for the Partial Match Problem, in: Proceedings of the Thirty-fifth Annual ACM Symposium on Theory of Computing, STOC '03, ACM, New York, NY, USA, 2003, pp. 667–672. doi:10.1145/780542.780639.

[19] K. H. Rosen, Discrete mathematics and its applications, McGraw-Hill, 2019.