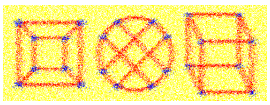

Journal of Graph Algorithms and Applications
http://jgaa.info/ vol. 28, no. 3, pp. 31–48 (2024)
DOI: 10.7155/jgaa.v28i3.2970

Splitting Plane Graphs to Outerplanarity

Martin Gronemann[1] *Martin Nöllenburg*[1] *Anaïs Villedieu*[1]

[1]Algorithms and Complexity Group,
TU Wien, Vienna, Austria

Submitted: June 2023	Accepted: May 2024	Published: September 2024
Article type: Regular paper	Communicated by: C.-C. Lin, B. M.-T. Lin, G. Liotta	

Abstract. Vertex splitting replaces a vertex by two copies and partitions its incident edges amongst the copies. This problem has been studied as a graph editing operation to achieve desired properties with as few splits as possible, most often planarity, for which the problem is NP-hard. Here we study how to minimize the number of splits to turn a plane graph into an outerplane one. We tackle this problem by establishing a direct connection between splitting a plane graph to outerplanarity, finding a connected face cover, and finding a feedback vertex set in its dual. We prove NP-completeness for plane biconnected graphs, while we show that a polynomial-time algorithm exists for maximal planar graphs. Additionally, we show upper and lower bounds for certain families of maximal planar graphs. Finally, we provide a SAT formulation for the problem, and evaluate it on a small benchmark.

1 Introduction

Graph editing problems are fundamental problems in graph theory. They define a set of basic operations on a graph G and ask for the minimum number of these operations necessary in order to turn G into a graph of a desired target graph class $\mathcal{G}$ [29, 34, 39, 47]. For instance, in the Cluster Editing problem [43] the operations are insertions or deletions of individual edges and the target graph class are cluster graphs, i.e., unions of vertex-disjoint cliques. In graph drawing, a particularly interesting graph class are planar graphs, for which several related graph editing problems have been studied, e.g., how many vertex deletions are needed to turn an arbitrary graph into a planar one [37] or how many vertex splits are needed to obtain a planar graph [19, 28]. In this paper, we are interested in the latter operation: vertex splitting. A *vertex split* creates two copies of a vertex v, distributes its edges among these two copies and then deletes v from G.

Further, we are translating the graph editing problem into a more geometric or topological drawing editing problem. This means that we apply the splitting operations not to the vertices of

Anaïs Villedieu is supported by the Austrian Science Fund (FWF) under grant P31119.

E-mail addresses: mgronemann@ac.tuwien.ac.at (Martin Gronemann) noellenburg@ac.tuwien.ac.at (Martin Nöllenburg) avilledieu@ac.tuwien.ac.at (Anaïs Villedieu)

(a) (b)

Figure 1: (a) An instance of Outerplane Splitting Number, where the colored vertices will be split; (b) resulting outerplane graph after the minimum 3 splits.

an abstract graph, but to the vertices of a planar graph drawing, or more generally to a planar embedded (or *plane*) graph. In a plane graph, each vertex has an induced cyclic counterclockwise order of its incident edges, which needs to be respected by any vertex split in the sense that we must split its cyclic order into two contiguous intervals, one for each of the two copies. From a different perspective, the two faces that serve as the separators of these two edge intervals are actually merged into a single face by the vertex split.

Finally, we consider outerplanar graphs as the target graph class. Thus, we want to apply a minimum number of vertex splits to a plane graph G, which merge a minimum number of faces in order to obtain an outerplanar embedded graph G', where all vertices are incident to a single face, called the *outer face*. We denote this minimum number of splits as the *outerplane splitting number* $\mathrm{osn}(G)$ of G (see Figure 1). Outerplanar graphs are a prominent graph class in graph drawing (see, e.g., [10, 22, 32, 33]) as well as in graph theory and graph algorithms more generally (e.g., [13, 23, 36]). For instance, outerplanar graphs admit planar circular layouts or 1-page book embeddings [8]. Additionally, outerplanar graphs often serve as a simpler subclass of planar graphs with good algorithmic properties. For instance, they have treewidth 2 and their generalizations to k-outerplanar graphs still have bounded treewidth [9, 12], which allows for polynomial-time algorithms for NP-complete problems that are tractable for such bounded-treewidth graphs. This, in turn, can be used to obtain a PTAS for these problems on planar graphs [5].

We are now ready to define our main computational problem as follows.

Problem 1 (Outerplane Splitting Number) *Given a plane biconnected graph $G = (V, E)$ and an integer k, can we transform G into an outerplane graph G' by applying at most k vertex splits to G?*

Contributions. In this paper, we introduce the above problem Outerplane Splitting Number. We start by showing the key property for our subsequent results, namely that (minimum) sets of vertex splits to turn a plane biconnected graph G into an outerplane one correspond to (minimum) connected face covers in G (Section 2), which in turn are equivalent to (minimum) feedback vertex sets in the dual graph of G. Using this tool we then show that for general plane biconnected graphs Outerplane Splitting Number is NP-complete (Section 3), whereas for maximal planar graphs we can solve it in polynomial time (Section 4). We also provide upper and lower bounds on the outerplane splitting number for maximal planar graphs (Section 5). Finally, we propose a SAT formulation to split arbitrary simple graphs to outerplanarity (Section 6). We evaluate it on a collection of small sparse graphs and show that it performs well for these instances.

Related Work. Splitting numbers have been studied mostly for abstract (non-planar) graphs with the goal of turning them into planar graphs. The Planar Splitting Number problem is NP-complete in general [19], but exact splitting numbers are known for complete and complete bipartite graphs [26, 28], as well as for the 4-cube [18]. For two-layer drawings of general bipartite graphs, the problem is still NP-complete in general with some restrictions that are polynomial-time solvable [2], but FPT [3] when parametrized by the number of split vertices. It has also been studied for other surfaces such as the torus [24] and the projective plane [25]. Another related concept is the split thickness of a graph G (or its folded covering number [31]), which is the smallest k such that G can be transformed into a planar graph by applying at most k splits per vertex. Recognizing graphs with split thickness 2 is NP-hard, but there is a constant-factor approximation algorithm and a fixed-parameter algorithm for graphs of bounded treewidth [17]. Recently, the complexity of the embedded splitting number problem of transforming non-planar graph drawings into plane ones has been investigated [40]. The authors showed that the problem is NP-hard, and presented an FPT algorithm for the case, where the set of vertices that should be split is given. Beyond the theoretical investigations of splitting numbers and planarity, there are also applied works in graph drawing making use of vertex splitting to untangle edges [46] or to improve layout quality for community exploration [4, 27]. Our investigations on the Outerplane Splitting Number problem could be of practical interest in the sense that outerplane drawings allow for adding crossing-free links to all vertices of a particular component from the outside, which is required, for instance, in the ChordLink model [4].

Regarding vertex splitting for achieving graph properties other than planarity, Trotter and Harary [44] studied vertex splitting to turn a graph into an interval graph. Paik et al. [41] considered vertex splitting to remove long paths in directed acyclic graphs and Abu-Khzam et al. [1] studied heuristics using vertex splitting for a cluster editing problem. Baumann et al. [6] showed that vertex splitting to a graph of pathwidth 1 admits a linear kernel when parameterized by the solution size; they further showed that testing if k vertex splits can turn an input graph into one of a graph class Π that can be expressed in monadic second order logic and has bounded treewidth is FPT in k. This includes, e.g., outerplanar graphs, Halin graphs, and generally graphs of bounded treewidth or pathwidth. Firbas and Sorge [21] studied the classical and parameterized complexity of vertex splitting to obtain hereditary graph properties and Firbas et al. [20] considered vertex splitting to obtain a cluster graph, i.e., a disjoint union of cliques, and showed that this problem is NP-complete, but admits a linear kernel.

Preliminaries. The key concept of our approach is to merge a set of faces of a given plane graph $G = (V, E)$ with vertex set $V = V(G)$ and edge set $E = E(G)$ into one big face which is incident to all vertices of G. Hence, the result is outerplanar. The idea is that if two faces f_1 and f_2 share a vertex v on their boundary (we say f_1 and f_2 *touch*, see Figure 2a), then we can split v into two new vertices v_1, v_2. In this way, we are able to create a narrow gap, which merges f_1, f_2 into a bigger face f (see Figure 2b). With this in mind, we formally define an *embedding-preserving split* of a vertex v w.r.t. two incident faces f_1 and f_2. We construct a new plane graph $G' = (V', E')$ with $V' = V \setminus \{v\} \cup \{v_1, v_2\}$. Consider the two neighbors of v both incident to f_1 and let w_1 be the second neighbor in clockwise order. Similarly, let w_i be the second vertex adjacent to v and incident to f_2. We call w_d the vertex preceding w_1 in the cyclic ordering of the neighbors, with d being the degree of v, see Figure 2a. Note that while $w_1 = w_{i-1}$ and $w_i = w_d$ is possible, $w_d \neq w_1$ and $w_{i-1} \neq w_i$. For the set of edges, we now set $E' = E \setminus \{(v, w_1), \ldots, (v, w_d)\} \cup \{(v_2, w_1), \ldots, (v_2, w_{i-1})\} \cup \{(v_1, w_i), \ldots, (v_1, w_d)\}$ and assume that they inherit their embedding from G. From now on we refer to this operation simply as a *split* or

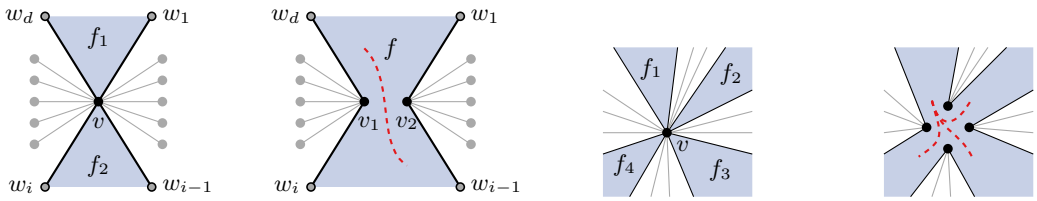


Figure 2: (a) Two touching faces f_1, f_2 with a common vertex v on their boundary. (b) Result of the split of v with respect to f_1, f_2 joining them into a new face f. (c-d) Merging 4 faces $f_1, \ldots, f_4$ covering a single vertex v with 3 splits.

when f_1, f_2 are clear from the context, we may refer to *merging* the two faces at v. The vertices v_1, v_2 introduced in place of v are called *copies* of v. If a copy v_i of a vertex v is split again, then any copy of v_i is also called a copy of the original vertex v.

We can now reformulate the task of using as few splits as possible. Our objective is to find a set of faces S that satisfies two conditions. (1) Every vertex in G has to be on the boundary of at least one face $f \in S$, that is, the faces in S *cover* all vertices in V.[1] And (2) for every two faces $f, f' \in S$ there exists a set of faces $\{f_1, \ldots, f_k\} \subseteq S$ such that $f = f_1, \ldots, f_k = f'$, and f_i touches f_{i+1} for $1 \leq i < k$. In other words, a solution S must be connected in terms of touching faces.[2]

We now introduce the main tool in our constructions that formalizes this concept.

2 Face-Vertex Incidence Graph

Let $G = (V, E)$ be a plane biconnected graph and F its set of faces. The *face-vertex incidence graph* is defined as $H = (V \cup F, E_H)$ and contains the edges $E_H = \{(v, f) \in V \times F : v \text{ is on the boundary of } f\}$. Graph H is by construction bipartite and we assume that it is plane by placing each vertex $f \in F$ into its corresponding face in G.

Definition 1 *Let G be a plane biconnected graph, let F be the set of faces of G, and let H be its face-vertex incidence graph. A* face cover *of G is a set $S \subseteq F$ of faces such that every vertex $v \in V$ is incident to at least one face in S. A face cover S of G is a* connected *face cover if the induced subgraph $H[S \cup V]$ of $S \cup V$ in H is connected.*

We point out that the problem of finding a connected face cover is not equivalent to the Connected Face Hitting Set Problem [42], where a connected set of vertices incident to every face is computed. We continue with two lemmas that are concerned with merging multiple faces at the same vertex (Figure 2c).

Lemma 1 *Let G be a plane biconnected graph, $v \in V$ a vertex of G, and $S \subseteq F$ a subset of the faces F of G, where each face $f \in S$ has the vertex v on its boundary. Then $|S| - 1$ splits are sufficient to merge the faces in S into one.*

[1] Testing whether such S with $|S| \leq k$ exists, is the NP-complete problem Face Cover [11].

[2] Notice that the conditions (1) and (2) are slightly stronger than the definition of connected face cover in Definition 1 below.

Proof: Let $f_1, \ldots, f_k$ with $k = |S|$ be the faces of S in the clockwise order as they appear around v (f_1 chosen arbitrarily). We start by merging f_1 with f_2, and iteratively merge the resulting face with f_i for $i = 3, \ldots, k$, which requires in total $|S| - 1$ splits (see Figure 2c and Figure 2d). □

Lemma 2 *Let G be a plane biconnected graph and let S be a connected face cover of G. Then $|S| - 1$ splits are sufficient to merge the faces of S into one.*

Proof: Let $H' = H[S \cup V]$ and compute a spanning tree T in H'. For every vertex $v \in V(T) \cap V(G)$, we apply Section 2 with the face set $F'(v) = \{f \in S \cap V(T) \mid (v, f) \in E(T)\}$. We root the tree at an arbitrary face $f' \in S$, which provides a hierarchy on the vertices and faces in T. Every vertex $v \in V(T) \cap V(G)$ requires by Section 2 $|F'(v)| - 1$ splits. Note that for all leaf vertices in T, $|F'(v)| = 1$, i.e., they will not be split. Each split is charged to the children of v in T. Since H is bipartite, so is T. It follows that every face $f \in S \setminus \{f'\}$ is charged exactly once by its parent, thus $|S| - 1$ splits suffice. □

Lemma 3 *Let G be a plane biconnected graph and σ a sequence of k splits to make G outerplane. Then G has a connected face cover of size $k + 1$.*

Proof: Since by definition applying σ to G creates a single big face that is incident to all vertices in $V(G)$ by iteratively merging pairs of original faces defining a set $S \subseteq F$, it is clear that S is a face cover of G and since the result of the vertex splits and face merges creates a single face, set S must also be connected. □

As a consequence of Section 2 we obtain that Outerplane Splitting Number and computing a minimum connected face cover are equivalent.

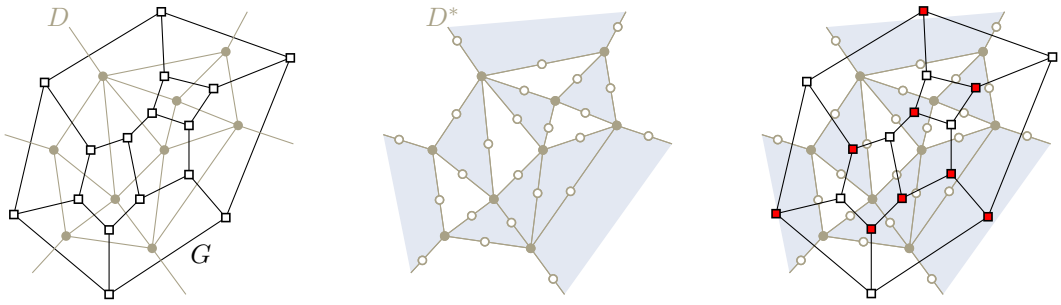
Theorem 1 *Let G be a plane biconnected graph. Then G has outerplane splitting number k if and only if it has a connected face cover of size $k + 1$.*

3 NP-completeness

In this section, we prove that finding a connected face cover of size k (and thus Outerplane Splitting Number) is NP-complete. The idea is to take the dual of a planar biconnected Vertex Cover instance and subdivide every edge once (we call this an *all-1-subdivision*). Note that the all-1-subdivision of a graph G corresponds to its vertex-edge incidence graph and the all-1-subdivision of the dual of G corresponds to the face-edge incidence graph of G. A connected face cover then corresponds to a vertex cover in the original graph, and vice versa. The following property greatly simplifies the arguments regarding Section 2.

Property 1 *Let G' be an all-1-subdivision of a biconnected planar graph G and S a set of faces that cover $V(G')$. Then S is a connected face cover of G'.*

Proof: Let H be the face-vertex incidence graph of G', and assume to the contrary that the induced subgraph $H' = H[S \cup V(G')]$ is not connected. Then there exists an edge $(u, v) \in E(G)$ such that u and v are in different connected components in H'. Let w be the subdivision vertex of (u, v) in G'. As a subdivision vertex, w is incident to only two faces, one of which, say f, must be contained in S. But f is also incident to u and v and hence u and v are in the same component of H' via face f, a contradiction. Hence H' is connected and S is a connected face cover of G'. □



(a) primal (black) and dual graph (gray)

(b) subdivision and connected face cover of the dual graph

(c) vertex cover (red) in the primal graph

Figure 3: Link between the primal graph G, its vertex cover, the dual D and its subdivision D^*.

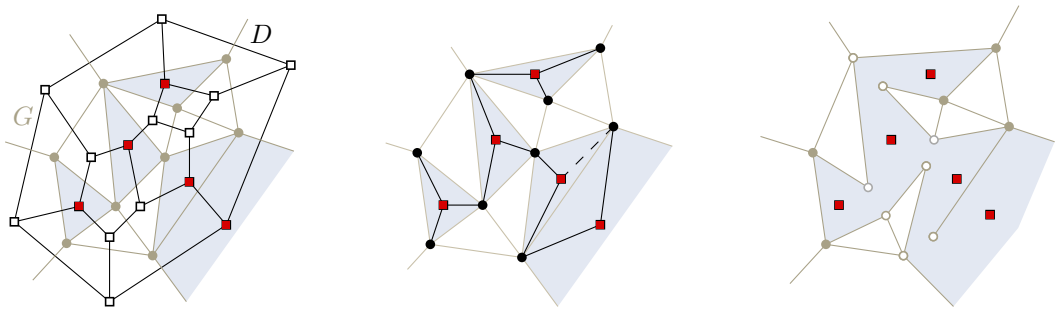
The proof of the next theorem is very similar to the reduction of Bienstock and Monma to show NP-completeness of Face Cover [11]; due to differences in the problem definitions, such as the connectivity of the face cover and whether the input graph is plane or not, we provide the full reduction for the sake of completeness.

Theorem 2 *Deciding whether a plane biconnected graph G has a connected face cover of size at most k is* NP*-complete.*

Proof: Clearly the problem is in NP. To prove hardness, we first introduce some notation. Let G be a plane biconnected graph and D the corresponding dual graph. Furthermore, let D^* be the all-1-subdivision of D. We prove now that a connected face cover S^* of size k in D^* is in a one-to-one correspondence with a vertex cover S of size k in G (see Figure 3). More specifically, we show that the dual vertices of the faces of S^* that form a connected face cover in D^*, are a vertex cover for G and vice versa. The reduction is from the NP-complete Vertex Cover problem in biconnected planar graphs in which all vertices have degree 3 (cubic graphs) [38].

Connected Face Cover ⇒ Vertex Cover: Let G be such a biconnected plane Vertex Cover instance. Assume we have a connected face cover S^* with $|S^*| = k$ for D^*. Note that the faces of D^* correspond to the vertices in G. We claim that the faces S^*, when mapped to the corresponding vertices $S \subseteq V(G)$ are a vertex cover for G. Assume otherwise, that is, there exists an edge $e^* \in E(G)$ that has no endpoints in S. However, e^* has a dual edge $e \in E(D)$ and therefore a subdivision vertex $v_e \in V(D^*)$. Hence, there is a face $f \in S^*$ that has v_e on its boundary by definition of connected face cover. And when mapped to D, f has e on its boundary, which implies that the primal edge e^* has at least one endpoint in S^*; a contradiction.

Vertex Cover ⇒ Connected Face Cover: To prove that a vertex cover S induces a connected face cover S^* in D^*, we have to prove that S^* covers all vertices and the induced subgraph in the face-vertex incidence graph H is connected. We proceed as in the other direction. S covers all edges in $E(G)$, thus every edge $e \in E(D)$ is bounded by at least one face of S^*. Hence, every subdivision vertex in $V(D^*)$ is covered by a face of S^*. Furthermore, every vertex in D^* is adjacent to a subdivision vertex, thus, also covered by a face in S^*. Since S^* is covering all vertices, we obtain from Section 3 that S^* is a connected face cover. □



(a) primal (gray) and dual graph (black)

(b) spanning tree (black) of the connected face cover

(c) outerplanar graph after splitting along the tree

Figure 4: The connected face cover (blue) is a feedback vertex set (red) in the dual.

4 Feedback Vertex Set Approach

A *feedback vertex set* $S^\circ \subset V(G)$ of a graph G is a vertex subset such that the induced subgraph $G[V(G) \setminus S^\circ]$ is acyclic. We show here that finding a connected face cover S of size k for a plane biconnected graph G is equivalent to finding a feedback vertex set $S^\circ \subset V(D)$ of size k in the dual graph D of G. The *weak dual*, i.e., the dual without a vertex for the outer face, of an outerplanar graph is a forest. Thus we must find the smallest number of splits in G which transform D into a forest. In other words, we must break all the cycles in D, and hence all of the vertices in the feedback vertex set S° of D correspond to the faces of G that should be merged together (see Figure 4).

Property 2 *Let H be the face-vertex incidence graph of a plane biconnected graph G and let S° be a feedback vertex set in the dual D of G. Then S° induces a connected face cover S in G.*

Proof: We need to show that S° is a face cover and that it is connected. First, assume there is a vertex $v \in V(G)$ of degree $\deg(v) = d$ that is not incident to a vertex in S°, i.e., a face of G. Since G is biconnected, v is incident to d faces $f_1, \ldots, f_d$, none of which is contained in S°. But then $D[V(D) \setminus S^\circ]$ has a cycle $(f_1, \ldots, f_d)$, a contradiction.

Next, we define $\overline{S^\circ} = V(D) \setminus S^\circ$ as the complement of the feedback vertex set S° in D. Assume that $H[V \cup S^\circ]$ has at least two separate connected components C_1, C_2. Then there must exist a closed curve in the plane separating C_1 from C_2, which avoids the faces in S° and instead passes through a sequence $(f_1, \ldots, f_\ell)$ of faces in $\overline{S^\circ}$, where each pair (f_i, f_{i+1}) for $i \in \{1, \ldots, \ell - 1\}$ as well as (f_ℓ, f_1) are adjacent in the dual D. Again this implies that there is a cycle in $D[V(D) \setminus S^\circ]$, a contradiction. Thus S° is a connected face cover. □

Theorem 3 *A plane biconnected graph G has outerplane splitting number k if and only if its dual D has a minimum feedback vertex set of size $k + 1$.*

Proof: Let S° be a minimum feedback vertex set of the dual D of G with cardinality $|S^\circ| = k+1$ and let H be the face-vertex incidence graph of G. We know from Section 4 that $H' = H[V(G) \cup S^\circ]$ is connected and hence S° induces a connected face cover S with $|S| = k + 1$. Then by Section 2 G has $\mathrm{osn}(G) \leq k$.

Let conversely σ be a sequence of k vertex splits that turn G into an outerplane graph G' and let F be the set of faces of G. By Section 2 we obtain a connected face cover S of size $k + 1$

consisting of all faces that are merged by σ. The complement $\overline{S} = F \setminus S$ consists of all faces of G that are not merged by the splits in σ and thus are the remaining (inner) faces of the outerplane graph G'. Since G' is outerplane and biconnected, $\overline{S}$ is the vertex set of the weak dual of G', which must be a tree. Hence S is a feedback vertex set in D of size $k+1$ and the minimum feedback vertex set in D has size at most $k+1$. □

Since all faces in a maximal planar graph are triangles, the maximum vertex degree of its dual is 3. Thus, we can apply the polynomial-time algorithm of Ueno et al. [45] to this dual, which computes the minimum feedback vertex set in graphs of maximum degree 3 by reducing the instance to polynomial-solvable matroid parity problem instance, and obtain

Corollary 1 *We can solve* Outerplane Splitting Number *for maximal planar graphs in polynomial time.*

Many other existing results for feedback vertex set extend to Outerplane Splitting Number, e.g., it has a kernel of size $13k$ [14] and admits a PTAS [16].

5 Upper and Lower Bounds

In this section we provide some upper and lower bounds on the outerplane splitting number in certain maximal planar graphs.

5.1 Upper Bounds

Based on the equivalence of Section 4 we obtain upper bounds on the outerplane splitting number from suitable upper bounds on the feedback vertex set problem, which has been studied for many graph classes, among them cubic graphs [15]. Liu and Zhao [35] showed that cubic graphs $G = (V, E)$ of girth at least four (resp., three) have a minimum feedback vertex set of size at most $\frac{|V|}{3}$ (resp., $\frac{3|V|}{8}$). Kelly and Liu [30] showed that connected planar subcubic graphs of girth at least five have a minimum feedback vertex set of size at most $\frac{2|V|+2}{7}$. Recall that the girth of a graph is the length of its shortest cycle.

Proposition 1 *The outerplane splitting number of a maximal planar graph $G = (V, E)$ of minimum degree (i) 3, (ii) 4, and (iii) 5, respectively, is at most (i) $\frac{3|V|-10}{4}$, (ii) $\frac{2|V|-7}{3}$, and (iii) $\frac{4|V|-13}{7}$, respectively.*

Proof: Maximal planar graphs with $n = |V|$ vertices have $2n-4$ faces. So the corresponding dual graphs have $2n-4$ vertices. Moreover, since the degree of a vertex in G corresponds to the length of a facial cycle in the dual, graphs with minimum vertex degree 3, 4, or 5 have duals with girth 3, 4, or 5, respectively. So if the minimum degree in G is 3, we obtain an upper bound on the feedback vertex set of $(3n-6)/4$; if the minimum degree is 4, the bound is $(2n-4)/3$; and if the minimum degree is 5, the bound is $(4n-6)/7$. The claim then follows from Section 4. □

5.2 Lower Bounds

We first provide a generic lower bound for the outerplane splitting number of maximal planar graphs. Let G be an n-vertex maximal planar graph with $2n-4$ faces. Each face is a triangle

incident to three vertices. In a minimum-size connected face cover S^*, at least one face covers three vertices. Due to the connectivity requirement, all other faces can add at most two newly covered vertices. Hence we need at least $\frac{n-1}{2}$ faces in any connected face cover. By Section 2 this implies that $\text{osn}(G) \geq \frac{n-3}{2}$.

Proposition 2 *Any maximal planar graph G has outerplane splitting number at least $\frac{|V(G)|-3}{2}$.*

Next, towards a better bound, we define a family of maximal planar graphs $T_d = (V_d, E_d)$ of girth 3 for $d \geq 0$ that have outerplane splitting number at least $\frac{2|V_d|-8}{3}$. The family are the complete planar 3-trees of depth d, which are defined recursively as follows. The graph T_0 is the 4-clique K_4. To obtain T_d from T_{d-1} for $d \geq 1$ we subdivide each inner triangular face of T_{d-1} into three triangles by inserting a new vertex and connecting it to the three vertices on the boundary of the face.

Proposition 3 *The complete planar 3-tree T_d of depth d has outerplane splitting number at least $\frac{2|V_d|-8}{3}$.*

Proof: Each T_d is a maximal planar graph with $n_d = 3 + \sum_{i=0}^{d} 3^i = \frac{3^{d+1}+5}{2}$ vertices. All 3^d leaf-level vertices added into the triangular faces of T_{d-1} in the last step of the construction have degree 3 and are incident to three exclusive faces, i.e., there is no face that covers more than one of these leaf-level vertices. This immediately implies that any face cover of T_d, connected or not, has size at least 3^d. From $n_d = \frac{3^{d+1}+5}{2}$ we obtain $d = \log_3 \frac{2n_d-5}{3}$ and $3^d = \frac{2n_d-5}{3}$. Section 2 then implies that $\text{osn}(T_d) \geq \frac{2n_d-8}{3}$. □

6 SAT Formulation

We showed above that Outerplane Splitting Number can be solved in polynomial time for maximal planar graphs and that it is already NP-complete for plane biconnected graphs. In this section, we propose a SAT formulation for a variant of the problem, that can compute an outerplane embedding, if one exists, of any given simple input graph (not necessarily a planar biconnected one) using up to k classical splitting operations, which do not need to respect any particular embedding. Specifically, given a simple graph G and an integer k, the SAT formula decides if G can be transformed into an outerplane graph G^* with at most k vertex splits.

Our formulation is based on a SAT formulation for book embeddings, similar to the one proposed by Bekos et al. [7], specifically to model crossings and transitivity. As outerplanar graphs are exactly those graphs that admit 1-page book embeddings, our formulation aims, for the given input graph, to find a set of splits which would allow for the graph to be embedded on a single page. We use a weighted Max-SAT formulation, meaning each clause is weighted (and a specific weight is used to describe hard clauses that must be satisfied), and the goal is to maximize the sum of the weights of the satisfied clauses. To the best of our knowledge, vertex splitting problems have not yet been modeled using SAT formulations or integer linear programming. As these are common techniques when working with hard graph drawing and other combinatorial problems, we hope that this initial effort can serve as a baseline to provide practical solutions to a larger set of vertex splitting problems in the future.

Intuitively, we build a graph $G' = (V', E')$, where every vertex of V is represented $k+1$ times, once as the input vertex, and k times as its possible copies. For each edge (u, v), we create a complete bipartite graph with one bipartition being u and all the new vertices representing potential

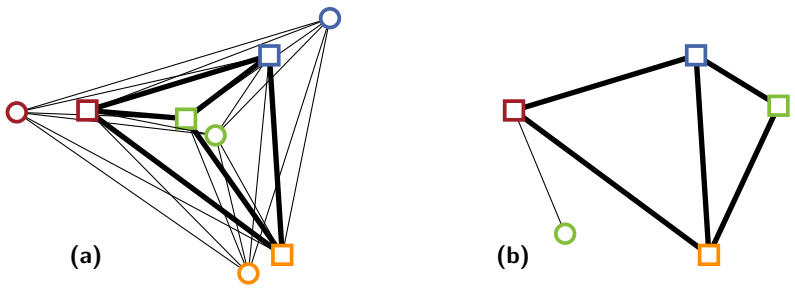


Figure 5: **(a)** An example of the graph G' computed for the formula from a K_4, with a budget of one split. The squares represent the vertices of the input graph, and the circles represent their possible copies, the thick edges are the edges of the input and the thin edges are alternative edges that can be used if the copy is selected, **(b)** the green vertex has been split, thus the copy (green circle) is retained as well as some of its incident edges to other non-split vertices.

copies of u and the second bipartition being v and the new vertices that represent potential copies of v. We then find a smallest outerplanar subgraph G^* in G' such that every edge and every vertex is represented at least once in the subgraph, as shown in Figure 5.

The formula $\mathcal{F}(G,k)$ is written using the conjunctive normal form, and we now introduce the different variables needed to build our constraints. For each vertex $v \in V$ and each of the k global copies denoted by $c_1, \ldots, c_k$, the variable $\texttt{vertex}(v,i)$ has value 1 if the ith copy c_i is a copy of vertex v and appears in G^*. For all $v \in V$, $\texttt{vertex}(v,0)$ corresponds to the input vertex and necessarily, $\texttt{vertex}(v,0) = 1$. Since edge $(u,v) \in E$ can be represented in the final drawing by any combination of copies of u and v, we create an edge from every copy of u to every copy of v in G', thus for $0 \leq i,j \leq k$, we create a variable $\texttt{edge}((u,i),(v,j))$, and the value is set to 1 if the edge between the ith copy of u and the jth copy of v is in G^*. For two edges e and e', we create the variable $\texttt{coexist}(e,e')$, such that $\texttt{coexist}(e,e') = 1$ if both e and e' are in G^*. Lastly, we need a variable to represent the vertex ordering along the outerface (the spine of the book embedding). More specifically, for every $u,v \in V$ and $0 \leq i,j \leq k$, we create $\texttt{before}((u,i),(v,j))$ such that the value 1 implies that the ith copy of u appears before the jth copy of v on the spine when embedding G^*.

We now introduce the set of constraints of $\mathcal{F}(G,k)$. For clarity, we simplify where possible the notation of vertices, and use v to refer to a copy of a vertex in V (instead of (v,i)). This set of vertices and their copies is called V'. Similarly, we sometimes call e an edge between two vertices of V', and denote this edge set with E'.

To fix the linear ordering of the vertices on the outerface, we require transitivity constraints. Therefore, for all triples of distinct vertices $u,v,w \in V'$, we create the two following clauses:

$$(\neg\texttt{before}(u,v) \vee \neg\texttt{before}(v,w) \vee \texttt{before}(u,w)) \tag{1}$$

$$(\texttt{before}(u,v) \vee \texttt{before}(v,w) \vee \neg\texttt{before}(u,w)) \tag{2}$$

which respectively ensure that if u is before v and v is before w, then u is before w, and that if u is after v and v after w, than u is after w (as shown in Figure 6(a)).

To minimize the number of copies used, for each vertex $v \in V$ and each $1 \leq i \leq k$ we create the following soft constraint (note that for $i = 0$ this constraint does not exist as the input vertices

are kept "for free"):

$$(\neg \texttt{vertex}(v, i)) \tag{3}$$

meaning that for every input vertex v that is not split, this clause is satisfied and its weight is added to the objective function.

As our budget allows for k splits, every splitting operation is done on a single vertex, thus for every value of $1 \leq i \leq k$, only one vertex can correspond to the ith split, hence for all pairs of distinct vertices $u, v \in V$:

$$(\neg \texttt{vertex}(u, i) \vee \neg \texttt{vertex}(v, i)) \tag{4}$$

which ensures that for at most one $v \in V$, $\texttt{vertex}(v, i) = 1$.

Since for $(u, v) \in E$ we create an edge between all the possible copies of u and v, we have to ensure that at least one edge is present in G^* between u or a copy of u, and v or a copy of v, thus:

$$\begin{gathered}(\texttt{edge}((u,0),(v,0)) \vee \cdots \vee \texttt{edge}((u,0),(v,k)) \vee \\ \cdots \\ \vee (\texttt{edge}((u,k),(v,0)) \vee \cdots \vee \texttt{edge}((u,k),(v,k))\end{gathered} \tag{5}$$

hence, for at least one value of i and j with $0 \leq i, j \leq k$, $\texttt{edge}((u, i), (v, j)) = 1$ (see Figure 6(b)).

Most importantly, we need to ensure no two pairs of active edges are crossing. To obtain this we first need to know which edge pairs are coexisting in G^* and can thus potentially intersect, for $e, e' \in E'$:

$$(\texttt{coexist}(e, e') \vee \neg \texttt{edge}(e) \vee \neg \texttt{edge}(e')) \tag{6}$$

where $\texttt{coexist}(e, e') = 1$ if $\texttt{edge}(e) = \texttt{edge}(e') = 1$.

This allows us to build the crossing constraints, thus for $e, e' \in E'$ such that $e = (u, v)$ and $e' = (s, t)$, we need to characterize all possible vertex orderings that can induce a crossing, which forbids the two edges from coexisting, we set $L = \neg \texttt{coexist}(e, e')$:

$$(L \vee \neg \texttt{before}(s, u) \vee \neg \texttt{before}(u, t) \vee \neg \texttt{before}(s, v) \vee \texttt{before}(v, t)) \tag{7}$$
$$(L \vee \neg \texttt{before}(s, u) \vee \neg \texttt{before}(u, t) \vee \texttt{before}(s, v) \vee \neg \texttt{before}(v, t)) \tag{8}$$
$$(L \vee \neg \texttt{before}(s, u) \vee \texttt{before}(u, t) \vee \neg \texttt{before}(s, v) \vee \neg \texttt{before}(v, t)) \tag{9}$$
$$(L \vee \texttt{before}(s, u) \vee \neg \texttt{before}(u, t) \vee \neg \texttt{before}(s, v) \vee \neg \texttt{before}(v, t)) \tag{10}$$
$$(L \vee \neg \texttt{before}(s, u) \vee \texttt{before}(u, t) \vee \texttt{before}(s, v) \vee \texttt{before}(v, t)) \tag{11}$$
$$(L \vee \texttt{before}(s, u) \vee \neg \texttt{before}(u, t) \vee \texttt{before}(s, v) \vee \texttt{before}(v, t)) \tag{12}$$
$$(L \vee \texttt{before}(s, u) \vee \texttt{before}(u, t) \vee \neg \texttt{before}(s, v) \vee \texttt{before}(v, t)) \tag{13}$$
$$(L \vee \texttt{before}(s, u) \vee \texttt{before}(u, t) \vee \texttt{before}(s, v) \vee \neg \texttt{before}(v, t)). \tag{14}$$

Consider as an example Eq. 14. If $\texttt{before}(s, u) = \texttt{before}(u, t) = \texttt{before}(s, v) = \neg \texttt{before}(v, t) = 0$, then the corresponding vertex ordering on the spine is in order v, t, u and s which induces a crossing, thus, we must have $L = 1$ and consequently $\texttt{coexist}(e, e') = 0$ as these two edges cannot coexist in a planar drawing (see Figure 6(c)). The above clauses cover all eight possibilities of orderings of u, v, s, t which would induce a crossing.

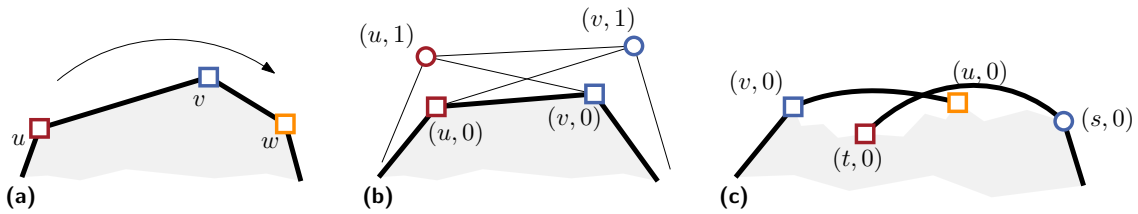


Figure 6: **(a)** Example of the `before` constraint, since u is before v, and v before w, then $\texttt{before}(u, w) = 1$ to satisfy Equation (1), **(b)** if $k = 1$, for one edge, Equation (5) represents a choice between four edges: $\texttt{edge}((u, 0), (v, 0))$, $\texttt{edge}((u, 0), (v, 1))$, $\texttt{edge}((u, 1), (v, 0))$, $\texttt{edge}((u, 1), (v, 1))$, **(c)** the situation described by Equation (14), where the two edges $((u, 0), (v, 0))$ and $((s, 0), (t, 0))$ cannot coexist if the vertices are ordered $(v, 0), (t, 0), (u, 0), (s, 0)$.

Lastly, we can only allow edges in G^* if their endpoints are also in G^*, thus for $(u, v) \in E'$,

$$(\neg\texttt{edge}(u, v) \vee \texttt{vertex}(u)) \wedge (\neg\texttt{edge}(u, v) \vee \texttt{vertex}(v)) \tag{15}$$

which ensures that $\texttt{edge}(u, v) = 1$ only if $\texttt{vertex}(u) = \texttt{vertex}(v) = 1$.

Only those clauses that track active copies are weighted (here soft) clauses, every other clause is a hard clause. Since every edge $e \in E$ has $(k + 1)^2$ representatives in E' we have $|E'| = |E|(k + 1)^2$, and since the coexistence variables are created for each pair of edges in E', there are $\mathcal{O}(|E'|^2) = \mathcal{O}(|E|^2k^4)$ constraints. As this formula does not model the splitting operation of the Outerplane Splitting Number problem, a natural question remains whether it can be adapted to handle embedded splitting operations. While a subordering of the vertices, e.g., on the outer face, can be fixed by setting the values of some variables, it is difficult to encode how copies should stay within specified faces during a split.

Experiments. We performed a small evaluation of the SAT formulation with a Python implementation using the MaxHS solver[3] on a subset of graphs from the Rome graph collection[4], which contains 11,534 planar or almost planar graphs on 10 to 100 vertices. The experiments were run on an Intel Core i7-9700 CPU with 16 GB of RAM. We sampled graphs on $n = 10, 20, 30, 40$ vertices, as representatives of the small to medium-sized graphs in the benchmark set. For each value of n we randomly chose ten graphs with increasing number of edges, from the smallest graph that was not yet outerplanar to the densest graph available for the chosen value of n. We note that even the densest of these graphs were sparse enough so that they are not yet trivially excluded from being 3-splittable to an outerplanar graph (which can have at most $2n - 3$ edges). For each graph we attempted to solve the SAT formula corresponding to the outerplanar splitting problem for $k = 1, 2, 3$ splits. The selected parameters almost always allowed us to solve each instance in under a minute, which $k = 4$ exceeded significantly with up to 15 minutes. This increase in runtime and the fact that each vertex split increases the visual complexity of keeping track of all the vertex copies in the resulting drawings made us stop the experiment at $k = 3$ splits, which is a suitable parameter for the chosen graph sizes. In fact, 29 out of the 40 graph instances in the experiment could be successfully turned into an outerplanar graph with at most three splits. The full results of the evaluation can be found in Table 1.

[3] http://www.maxhs.org/
[4] http://www.graphdrawing.org/data.html

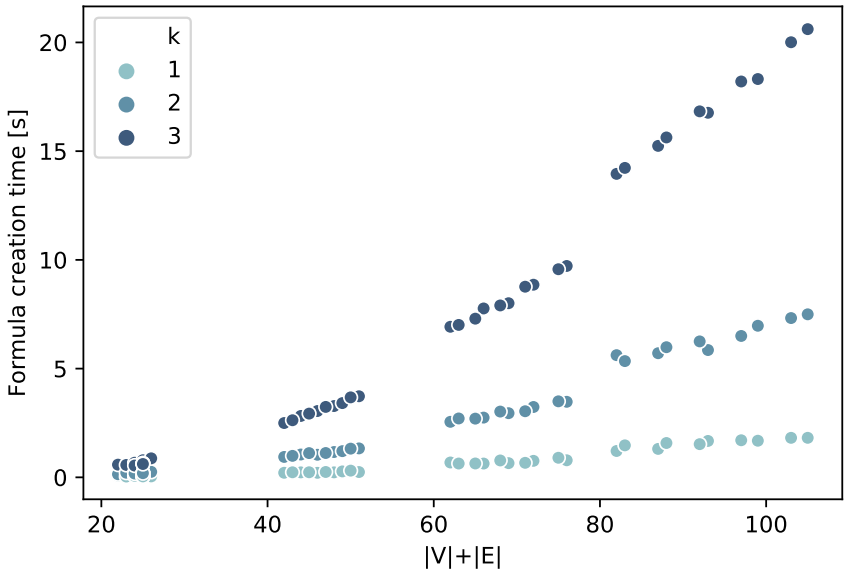


Figure 7: Time required to create the formula in seconds, by the size of the graph (number of vertices plus number of edges). Each color corresponds to a different number of allowed splits.

As expected from the number of constraints above, the time required to generate the formula is largely influenced by the number k of splits and the size of the graph, see Figure 7. We note that even for the largest of our instances, the formula was generated relatively quickly, e.g., it took about 20 seconds for a graph with $|V| = 40$ vertices, about 60 edges, and $k = 3$ splits.

Interestingly, we see in Figure 8 that solving smaller instances is significantly faster than the time required to generate their SAT formula. This trend is reversed for relatively higher values of $k = 3$ on denser graphs. Specifically, note that larger instances that still admitted a solution take significantly more time to be solved (dark blue outlier dots in Figure 8). In general, we can also see that for instances that did not admit a solution for the given value of k, the solver relatively quickly returned "UNSAT".

Overall, in conclusion of our small experiment, we found that our SAT formulation is indeed able to solve the outerplanar splitting problem for a small budget of k splits and graphs up to 40 vertices and 60 edges. For denser instances, that are expected to require a larger number of splits, an outerplanar embedding cannot generally be efficiently computed with this approach, though it appears that clear no-instances are still possible to detect within a reasonable amount of time.

7 Conclusion

We have introduced the Outerplane Splitting Number problem and established its complexity for plane biconnected graphs. The most important open question revolves around the embedding requirement. Splitting operations can be defined more loosely and allow for any new embedding and neighborhood of the split vertices. In general, it is also of interest to understand how the problem differs when the input graph does not have an embedding at all, as in the original splitting number

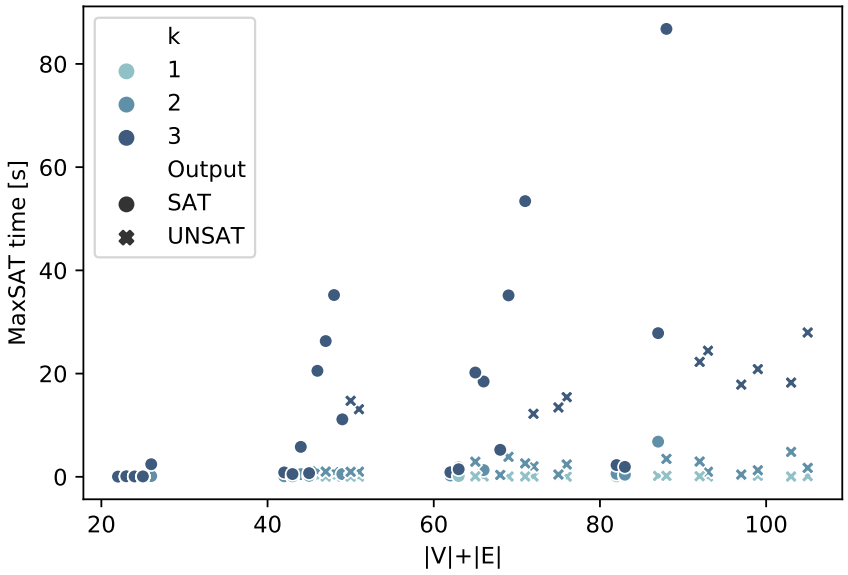


Figure 8: Time required to compute a solution for the formula in seconds, by the size of the graph (number of vertices plus number of edges). Each color corresponds to a different number of allowed splits, dots correspond to formulas that were solved, and crosses to formulas that do not admit a solution.

problem. Since Outerplane Splitting Number can be solved in polynomial time for maximal planar graphs but is hard for plane biconnected graphs, there is a complexity gap to be closed when faces of degree more than three are involved. Vertex splitting in graph drawings has so far been studied to achieve planarity and outerplanarity. A natural extension is to study it for other graph classes or graph properties.

Another direction for future work is to investigate algorithmic approaches for vertex splitting of embedded graphs in practical scenarios. Our initial experiments with the SAT formulation indicate that for small graphs with up to 40 vertices and a budget with up to three splits, computing exact solutions is computationally feasible in a practical sense. Designing practically fast and effective algorithms for larger graphs and more splits is an interesting algorithm engineering challenge.

References

[1] F. N. Abu-Khzam, J. R. Barr, A. Fakhereldine, and P. Shaw. A greedy heuristic for cluster editing with vertex splitting. In *Artificial Intelligence for Industries (AI4I'21)*, pages 38–41. IEEE, 2021. doi:10.1109/AI4I51902.2021.00017.

[2] A. R. Ahmed, P. Angelini, M. A. Bekos, G. D. Battista, M. Kaufmann, P. Kindermann, S. G. Kobourov, M. Nöllenburg, A. Symvonis, A. Villedieu, and M. Wallinger. Splitting vertices in 2-layer graph drawings. *IEEE Computer Graphics and Applications*, 43(3):24–35, 2023. doi:10.1109/MCG.2023.3264244.

[3] R. Ahmed, S. G. Kobourov, and M. Kryven. An FPT algorithm for bipartite vertex splitting. In P. Angelini and R. von Hanxleden, editors, *Graph Drawing and Network Visualization (GD'22)*, volume 13764 of *LNCS*, pages 261–268. Springer, 2022. doi:10.1007/978-3-031-22203-0_19.

[4] L. Angori, W. Didimo, F. Montecchiani, D. Pagliuca, and A. Tappini. Hybrid graph visualizations with chordlink: Algorithms, experiments, and applications. *IEEE Trans. Vis. and Comput. Graph.*, 28(2):1288–1300, 2022. doi:10.1109/TVCG.2020.3016055.

[5] B. S. Baker. Approximation algorithms for NP-complete problems on planar graphs. *J. ACM*, 41(1):153–180, 1994. doi:10.1145/174644.174650.

[6] J. Baumann, M. Pfretzschner, and I. Rutter. Parameterized complexity of vertex splitting to pathwidth at most 1. In D. Paulusma and B. Ries, editors, *Graph-Theoretic Concepts in Computer Science (WG'23)*, volume 14093 of *LNCS*, pages 30–43. Springer, 2023. doi:10.1007/978-3-031-43380-1_3.

[7] M. A. Bekos, M. Kaufmann, and C. Zielke. The book embedding problem from a SAT-solving perspective. In E. D. Giacomo and A. Lubiw, editors, *Graph Drawing and Network Visualization (GD'15)*, volume 9411 of *LNCS*, pages 125–138. Springer, 2015. doi:10.1007/978-3-319-27261-0_11.

[8] F. Bernhart and P. C. Kainen. The book thickness of a graph. *J. Comb. Theory, Ser. B*, 27(3):320–331, 1979. doi:10.1016/0095-8956(79)90021-2.

[9] T. Biedl. On triangulating k-outerplanar graphs. *Discret. Appl. Math.*, 181:275–279, 2015. doi:10.1016/j.dam.2014.10.017.

[10] T. C. Biedl. Small drawings of outerplanar graphs, series-parallel graphs, and other planar graphs. *Discret. Comput. Geom.*, 45(1):141–160, 2011. doi:10.1007/s00454-010-9310-z.

[11] D. Bienstock and C. L. Monma. On the complexity of covering vertices by faces in a planar graph. *SIAM J. Comput.*, 17(1):53–76, 1988. doi:10.1137/0217004.

[12] H. L. Bodlaender. A partial k-arboretum of graphs with bounded treewidth. *Theor. Comput. Sci.*, 209(1-2):1–45, 1998. doi:10.1016/S0304-3975(97)00228-4.

[13] H. L. Bodlaender and F. V. Fomin. Approximation of pathwidth of outerplanar graphs. *J. Algorithms*, 43(2):190–200, 2002. doi:10.1016/S0196-6774(02)00001-9.

[14] M. Bonamy and L. Kowalik. A 13k-kernel for planar feedback vertex set via region decomposition. *Theor. Comput. Sci.*, 645:25–40, 2016. doi:10.1016/j.tcs.2016.05.031.

[15] J. A. Bondy, G. Hopkins, and W. Staton. Lower bounds for induced forests in cubic graphs. *Can. Math. Bull.*, 30(2):193–199, 1987. doi:10.4153/CMB-1987-028-5.

[16] E. Demaine and M. Hajiaghayi. Bidimensionality: New connections between FPT algorithms and PTASs. In *Symposium on Discrete Algorithms (SODA'05)*, pages 590–601, 2005. URL: https://dl.acm.org/doi/10.5555/1070432.1070514.

[17] D. Eppstein, P. Kindermann, S. G. Kobourov, G. Liotta, A. Lubiw, A. Maignan, D. Mondal, H. Vosoughpour, S. Whitesides, and S. K. Wismath. On the planar split thickness of graphs. *Algorithmica*, 80(3):977–994, 2018. doi:10.1007/s00453-017-0328-y.

[18] L. Faria, C. M. H. de Figueiredo, and C. F. X. de Mendonça N. The splitting number of the 4-cube. In C. L. Lucchesi and A. V. Moura, editors, *Latin American Symposium on Theoretical Informatics (LATIN'98)*, volume 1380 of *LNCS*, pages 141–150. Springer, 1998. doi:10.1007/BFb0054317.

[19] L. Faria, C. M. H. de Figueiredo, and C. F. X. de Mendonça N. Splitting number is NP-complete. *Discret. Appl. Math.*, 108(1):65–83, 2001. doi:10.1016/S0166-218X(00)00220-1.

[20] A. Firbas, A. Dobler, F. Holzer, J. Schafellner, M. Sorge, A. Villedieu, and M. Wißmann. The complexity of cluster vertex splitting and company. In H. Fernau, S. Gaspers, and R. Klasing, editors, *Theory and Practice of Computer Science (SOFSEM'24)*, volume 14519 of *LNCS*, pages 226–239. Springer, 2024. doi:10.1007/978-3-031-52113-3_16.

[21] A. Firbas and M. Sorge. On the complexity of establishing hereditary graph properties via vertex splitting. *CoRR*, abs/2401.16296, 2024. arXiv:2401.16296.

[22] F. Frati. Planar rectilinear drawings of outerplanar graphs in linear time. *Computat. Geom.*, 103:101854, 2022. doi:10.1016/j.comgeo.2021.101854.

[23] G. N. Frederickson. Searching among intervals and compact routing tables. *Algorithmica*, 15(5):448–466, 1996. doi:10.1007/BF01955044.

[24] N. Hartsfield. The toroidal splitting number of the complete graph K_n. *Discret. Math.*, 62(1):35–47, 1986. doi:10.1016/0012-365X(86)90039-7.

[25] N. Hartsfield. The splitting number of the complete graph in the projective plane. *Graphs Comb.*, 3(1):349–356, 1987. doi:10.1007/BF01788557.

[26] N. Hartsfield, B. Jackson, and G. Ringel. The splitting number of the complete graph. *Graphs Comb.*, 1(1):311–329, 1985. doi:10.1007/BF02582960.

[27] N. Henry, A. Bezerianos, and J. Fekete. Improving the readability of clustered social networks using node duplication. *IEEE Trans. Vis. Comput. Graph.*, 14(6):1317–1324, 2008. doi:10.1109/TVCG.2008.141.

[28] B. Jackson and G. Ringel. The splitting number of complete bipartite graphs. *Arch. Math.*, 42(2):178–184, 1984. doi:10.1007/BF01772941.

[29] G. Kant. Augmenting outerplanar graphs. *J. Algorithms*, 21(1):1–25, 1996. doi:10.1006/jagm.1996.0034.

[30] T. Kelly and C. Liu. Minimum size of feedback vertex sets of planar graphs of girth at least five. *Eur. J. Comb.*, 61:138–150, 2017. doi:10.1016/j.ejc.2016.10.009.

[31] K. B. Knauer and T. Ueckerdt. Three ways to cover a graph. *Discret. Math.*, 339(2):745–758, 2016. doi:10.1016/j.disc.2015.10.023.

[32] S. Lazard, W. J. Lenhart, and G. Liotta. On the edge-length ratio of outerplanar graphs. *Theor. Comput. Sci.*, 770:88–94, 2019. doi:10.1016/j.tcs.2018.10.002.

[33] W. Lenhart and G. Liotta. Proximity drawings of outerplanar graphs. In S. C. North, editor, *Graph Drawing (GD'96)*, volume 1190 of *LNCS*, pages 286–302. Springer, 1996. doi:10.1007/3-540-62495-3_55.

[34] J. M. Lewis and M. Yannakakis. The node-deletion problem for hereditary properties is NP-complete. *J. Comput. Syst. Sci.*, 20(2):219–230, 1980. doi:10.1016/0022-0000(80)90060-4.

[35] J. Liu and C. Zhao. A new bound on the feedback vertex sets in cubic graphs. *Discret. Math.*, 148(1-3):119–131, 1996. doi:10.1016/0012-365X(94)00268-N.

[36] A. Maheshwari and N. Zeh. External memory algorithms for outerplanar graphs. In A. Aggarwal and C. P. Rangan, editors, *Algorithms and Computation (ISAAC'99)*, volume 1741 of *LNCS*, pages 307–316. Springer, 1999. doi:10.1007/3-540-46632-0_31.

[37] D. Marx and I. Schlotter. Obtaining a planar graph by vertex deletion. *Algorithmica*, 62(3-4):807–822, 2012. doi:10.1007/s00453-010-9484-z.

[38] B. Mohar. Face covers and the genus problem for apex graphs. *J. Comb. Theory, Ser. B*, 82(1):102–117, 2001. doi:10.1006/jctb.2000.2026.

[39] A. Natanzon, R. Shamir, and R. Sharan. Complexity classification of some edge modification problems. *Discret. Appl. Math.*, 113(1):109–128, 2001. doi:10.1016/S0166-218X(00)00391-7.

[40] M. Nöllenburg, M. Sorge, S. Terziadis, A. Villedieu, H. Wu, and J. Wulms. Planarizing graphs and their drawings by vertex splitting. In P. Angelini and R. von Hanxleden, editors, *Graph Drawing and Network Visualization (GD'22)*, volume 13764 of *LNCS*, pages 232–246. Springer, 2022. doi:10.1007/978-3-031-22203-0_17.

[41] D. Paik, S. M. Reddy, and S. Sahni. Vertex splitting in dags and applications to partial scan designs and lossy circuits. *Int. J. Found. Comput. Sci.*, 9(4):377–398, 1998. doi:10.1142/S0129054198000301.

[42] P. Schweitzer and P. Schweitzer. Connecting face hitting sets in planar graphs. *Inf. Process. Lett.*, 111(1):11–15, 2010. doi:10.1016/j.ipl.2010.10.008.

[43] R. Shamir, R. Sharan, and D. Tsur. Cluster graph modification problems. *Discret. Appl. Math.*, 144(1-2):173–182, 2004. doi:10.1016/j.dam.2004.01.007.

[44] W. T. Trotter and F. Harary. On double and multiple interval graphs. *J. Graph Theory*, 3(3):205–211, 1979. doi:10.1002/jgt.3190030302.

[45] S. Ueno, Y. Kajitani, and S. Gotoh. On the nonseparating independent set problem and feedback set problem for graphs with no vertex degree exceeding three. *Discret. Math.*, 72(1-3):355–360, 1988. doi:10.1016/0012-365X(88)90226-9.

[46] H.-Y. Wu, M. Nöllenburg, and I. Viola. Multi-level area balancing of clustered graphs. *IEEE Trans. Vis. Comput. Graph.*, 28(7):2682–2696, 2022. doi:10.1109/TVCG.2020.3038154.

[47] M. Yannakakis. Node-and edge-deletion NP-complete problems. In *Symposium on Theory of Computing (STOC'78)*, pages 253–264. ACM, 1978. doi:10.1145/800133.804355.

Table 1: Results from the experiment, the time to create the formula is indicated under gen_t and the time to solve the formula under sol_t, both in seconds, the Y/N columns indicates Y if a solution exists and was computed and N otherwise.

$\|V\|$	$\|E\|$	$k=1$			$k=2$			$k=3$		
		gen_t [s]	sol_t [s]	Y/N	gen_t [s]	sol_t [s]	Y/N	gen_t [s]	sol_t[s]	Y/N
10	12	0.08	<0.01	Y	0.14	0.01	Y	0.58	0.01	Y
	13	0.04	<0.01	Y	0.21	0.02	Y	0.57	0.11	Y
	14	0.07	<0.01	Y	0.22	0.01	Y	0.27	0.08	Y
	14	0.07	0.01	Y	0.18	0.02	Y	0.54	0.07	Y
	15	0.04	0.02	Y	0.20	0.02	Y	0.64	0.08	Y
	15	0.05	<0.01	Y	0.23	0.02	Y	0.79	0.07	Y
	15	0.04	<0.01	Y	0.20	0.01	Y	0.64	0.08	Y
	15	0.04	<0.01	Y	0.20	0.02	Y	0.61	0.08	Y
	15	0.04	<0.01	Y	0.19	0.01	Y	0.61	0.08	Y
	16	0.04	0.01	N	0.25	0.14	Y	0.27	2.44	Y
20	22	0.21	0.05	Y	0.94	0.05	Y	2.50	0.85	Y
	23	0.23	0.02	Y	0.99	0.10	Y	2.63	0.54	Y
	24	0.23	0.02	N	1.05	0.52	Y	2.82	5.80	Y
	25	0.23	0.02	Y	1.11	0.19	Y	2.93	0.71	Y
	26	0.21	0.03	N	1.04	1.05	N	3.05	20.54	Y
	27	0.25	0.04	N	1.12	1.01	N	3.24	26.30	Y
	28	0.23	0.01	N	1.16	0.88	N	3.28	35.22	Y
	29	0.27	0.04	N	1.20	0.53	Y	3.41	11.13	Y
	30	0.31	0.02	N	1.32	0.96	N	3.68	14.73	N
	31	0.24	0.04	N	1.33	0.99	N	3.73	13.10	N
30	32	0.68	0.04	Y	2.55	0.26	Y	6.92	0.88	Y
	33	0.64	0.15	Y	2.71	1.85	Y	7.01	1.46	Y
	35	0.64	0.09	N	2.70	2.93	N	9.30	20.20	Y
	36	0.63	0.08	N	2.75	1.31	Y	7.77	18.46	Y
	38	0.78	0.02	N	3.02	0.36	N	7.90	5.23	Y
	39	0.65	0.06	N	2.95	3.87	N	8.01	35.15	Y
	41	0.67	0.07	N	3.04	2.60	N	8.77	53.42	Y
	42	0.76	0.06	N	3.24	2.02	N	8.86	12.20	N
	45	0.90	0.05	N	3.50	0.44	N	9.57	13.44	N
	46	0.78	0.03	N	3.47	2.37	N	9.72	15.44	N
40	42	1.21	0.09	Y	5.62	0.58	Y	13.96	2.28	Y
	43	1.47	0.11	Y	5.35	0.41	Y	14.23	1.92	Y
	47	1.31	0.17	N	5.71	6.81	Y	15.24	27.84	Y
	48	1.58	0.18	N	5.98	3.47	N	15.63	86.79	Y
	52	1.53	0.12	N	6.25	2.96	N	16.83	22.27	N
	53	1.67	0.06	N	5.85	0.98	N	16.76	24.43	N
	57	1.71	0.04	N	6.50	0.43	N	18.20	17.86	N
	59	1.68	0.20	N	6.97	1.27	N	18.32	20.87	N
	63	1.82	0.06	N	7.32	4.82	N	20.01	18.25	N
	65	1.82	0.13	N	7.50	1.73	N	20.61	27.96	N