Algorithmica (2006) 45: 159–182
DOI: 10.1007/s00453-005-1181-y

Algorithmica

A Fixed-Parameter Approach to 2-Layer Planarization*

Vida Dujmović,[1,2] Michael Fellows,[3] Michael Hallett,[1] Matthew Kitching,[1] Giuseppe Liotta,[4] Catherine McCartin,[5] Naomi Nishimura,[6] Prabhakar Ragde,[6] Fran Rosamond,[3] Matthew Suderman,[1] Sue Whitesides,[1] and David R. Wood[1]

Abstract. A bipartite graph is *biplanar* if the vertices can be placed on two parallel lines (*layers*) in the plane such that there are no edge crossings when edges are drawn as line segments between the layers. In this paper we study the 2-LAYER PLANARIZATION problem: Can k edges be deleted from a given graph G so that the remaining graph is biplanar? This problem is $\mathcal{NP}$-complete, and remains so if the permutation of the vertices in one layer is fixed (the 1-LAYER PLANARIZATION problem). We prove that these problems are fixed-parameter tractable by giving linear-time algorithms for their solution (for fixed k). In particular, we solve the 2-LAYER PLANARIZATION problem in $\mathcal{O}(k \cdot 6^k + |G|)$ time and the 1-LAYER PLANARIZATION problem in $\mathcal{O}(3^k \cdot |G|)$ time. We also show that there are polynomial-time constant-approximation algorithms for both problems.

Key Words. Graph drawing, Planarization, Crossing minimization, Sugiyama approach, Fixed-parameter tractability, NP-complete, Graph algorithms.

1. Introduction. In a 2-*layer drawing* of a bipartite graph $G = (A, B; E)$, the vertices in A are positioned on a line in the plane, which is parallel to a different line containing the vertices in B, and the edges are drawn as line-segments. Such drawings have applications in visualization [3], [14], DNA mapping [27], and VLSI layouts [15]; a recent survey [19] gives more details.

A *biplanar graph* is a bipartite graph that admits a 2-layer drawing with no edge crossings; we call such a drawing a *biplanar drawing*. Consider a 2-layer drawing of a bipartite graph produced by first drawing a maximum biplanar subgraph with no crossings and then drawing all the remaining edges. Although such a drawing is unlikely to minimize the number of crossings, there is some experimental evidence to suggest that

---

* A preliminary version of this paper was presented at the 9th International Symposium on Graph Drawing (GD 2001), Vienna, Austria, September 23–26, 2001.

[1] School of Computer Science, McGill University, Montréal, Canada. {vida,kitching,msuder,sue,wood}@cs.mcgill.ca, hallett@mcb.mcgill.ca. Research supported by NSERC and FCAR.
[2] School of Computer Science, Carleton University, Ottawa, Canada.
[3] School of Electrical Engineering and Computer Science, The University of Newcastle, Newcastle, Australia. {mfellows,fran}@cs.newcastle.edu.au.
[4] Dipartimento di Ingegneria Elettronica e dell'Informazione, Università di Perugia, Perugia, Italy. liotta@diei.unipg.it. Research supported by CNR and MURST.
[5] School of Mathematical and Computing Sciences, Victoria University of Wellington, Wellington, New Zealand. Catherine.McCartin@mcs.vuw.ac.nz.
[6] School of Computer Science, University of Waterloo, Waterloo, Canada. {nishi,plragde}@uwaterloo.ca. Research supported by NSERC.

Received April 27, 2004; revised January 25, 2005. Communicated by T. Nishizeki.
Online publication March 20, 2006.

2-layer drawings in which all the crossings occur in a few edges are more readable than drawings with fewer total crossings [18]. Maximizing the size of a biplanar subgraph is equivalent to minimizing the number of edges not in it. This leads naturally to the definition of the 2-LAYER PLANARIZATION problem: Given a graph G (not necessarily bipartite) and an integer k, can G be made biplanar by deleting at most k edges? This problem is the focus of this paper.

2-Layer drawings are of fundamental importance in the "Sugiyama" approach to multilayer graph drawing [24]. This method first assigns vertices to layers, then makes repeated sweeps up and down the layers to determine an ordering of the vertices in one layer with respect to the ordering of the preceding layer. This involves solving the 1-LAYER PLANARIZATION problem: Given a bipartite graph $G = (A, B; E)$, a permutation π of A, and an integer k, can at most k edges be deleted to permit G to be drawn without crossings with π as the ordering of vertices in A? In this paper we present results on this problem as well.

Crossing minimization background. Instead of deleting edges, one can seek to minimize the number of crossings in a 2-layer drawing. Since graphs that admit 2-layer drawings are necessarily bipartite, the input graph here must be bipartite as well. The problems associated with one or zero permutations as part of the input are called 1- and 2-LAYER CROSSING MINIMIZATION, respectively. Both of these well-studied problems are $\mathcal{NP}$-complete [10], [11]. The 1-LAYER CROSSING MINIMIZATION problem is $\mathcal{NP}$-complete even for graphs with only degree-1 vertices in the fixed layer and vertices of degree at most 4 in the other layer [17]; that is, for a forest of 4-stars.

Integer linear programming algorithms have been presented for 1- and 2-LAYER CROSSING MINIMIZATION [13], [26]. Jünger and Mutzel [13] surveyed numerous heuristics proposed for both problems, and experimentally compared their performance with the optimal solutions. They reported that the *iterated barycentre* method of Sugiyama et al. [24] performed best in practice. However, from a theoretical point of view the *median heuristic* of Eades and Wormald [10] is a better approach for 1-LAYER CROSSING MINIMIZATION. In particular, the median heuristic is a linear-time 3-approximation algorithm, whereas the barycentre heuristic is a $\Theta(\sqrt{n})$-approximation algorithm [10]. Furthermore, for graphs with maximum degree 3 in the free layer, the median heuristic is a 2-approximation algorithm for this problem [3]. A more general approximation result with respect to the maximum degree in the free layer is obtained by Yamaguchi and Sugimoto [28]. Recently, Nagamochi [21] devised a 1.47-approximation algorithm for the problem. Shahrokhi et al. [22] gave a polynomial-time algorithm that approximates 2-LAYER CROSSING MINIMIZATION within a factor of $\mathcal{O}(\log n)$ for a wide class of n-vertex graphs. Demetrescu and Finocchi [2] presented a heuristic for 1-LAYER CROSSING MINIMIZATION based on feedback arc sets.

Planarization background. Despite the practical significance of the problems, 1- and 2-LAYER PLANARIZATION have received less attention in the graph drawing literature than their crossing minimization counterparts. The 2-LAYER PLANARIZATION problem is $\mathcal{NP}$-complete [9], [25] even for planar biconnected bipartite graphs with vertices in the respective bipartitions having degree 2 and 3 [9]. The 1-LAYER PLANARIZATION problem is also $\mathcal{NP}$-complete, even for graphs with only degree-1 vertices in the fixed layer and vertices of degree at most 2 in the other layer [9]; that is, for collections of 1- and

2-paths. With the order of the vertices in both layers fixed the problem can be solved in polynomial time [9], [20].

Integer linear programming algorithms have been developed for these problems [18], [20]. Shahrokhi et al. [22] present an $\mathcal{O}(n)$ time dynamic programming algorithm for 2-LAYER PLANARIZATION of weighted acyclic graphs, for which the objective is to minimize the total weight of deleted edges. Although Tomii et al. [25] claim an $\mathcal{O}(n^3)$ time algorithm for the 2-LAYER PLANARIZATION problem on acyclic graphs, Mutzel [18] demonstrates a tree for which their algorithm is not optimal.

Fixed parameter tractability. When the maximum number k of allowed edge deletions is small, it may be useful to have an algorithm for 1- or 2-LAYER PLANARIZATION whose running time is exponential in k but polynomial in the size of the graph. The theory of parameterized complexity [4] addresses complexity issues of this nature, in which a problem is specified in terms of one or more parameters. A parameterized problem with input size n and parameter size k is *fixed parameter tractable*, or in the class $\mathcal{FPT}$, if there is an algorithm to solve the problem in $f(k) \cdot n^{\alpha}$ time, for some function f and constant α. A problem in $\mathcal{FPT}$ is thus solvable in polynomial time for fixed k where the degree of the polynomial is independent of k.

In this paper we consider the parameterized analogues of the 1- and 2-LAYER PLANARIZATION problems, where k is the fixed parameter, not included in the input. In a companion paper [5], we prove using bounded pathwidth techniques that the h-layer generalizations of the 2-LAYER CROSSING MINIMIZATION and 2-LAYER PLANARIZATION problems are in $\mathcal{FPT}$, where h is also considered a parameter of the problem. The 1-layer versions of these problems can also be solved using this approach. Unfortunately, a pathwidth-based approach is not particularly practical, since the running time of the algorithms is $\mathcal{O}(2^{32(h+2k)^3} n)$. Recently, Dujmović and Whitesides [7] developed a more practical FPT algorithm for the 1-LAYER CROSSING MINIMIZATION problem, which was further improved in [6].

We expect the parameter k to be small in practice. Instances of the 1- and 2-LAYER PLANARIZATION for dense graphs are of little interest from a practical point of view, since such instances have a high number of crossings in any 2-layer drawing [3]. Therefore, they are hardly worth optimizing, as the resulting drawing will be unreadable anyway.

Note that there can be many ways of formulating a parameterized version of an optimization problem. For example, 2-layer planarization is closely related to the SPANNING CATERPILLAR FOREST problem, where a caterpillar (as defined formally in Section 2.1) is a graph the removal of whose leaves results in a path. This problem asks if a graph $G = (V, E)$ has a spanning forest consisting of ℓ component caterpillars. G has a biplanarizing set with k edges if and only if G has a spanning forest with $\ell = k - (|E| - |V|)$ component caterpillars. From a traditional point of view, the SPANNING CATERPILLAR FOREST problem (that of determining whether or not G has a spanning forest with at most ℓ component caterpillars) is equivalent to the 2-LAYER PLANARIZATION problem. In particular, both are $\mathcal{NP}$-complete. In fact, the SPANNING CATERPILLAR FOREST problem with $\ell = 1$ is $\mathcal{NP}$-complete by a simple reduction from HAMILTONIAN PATH. Therefore, unless $\mathcal{P} = \mathcal{NP}$, SPANNING CATERPILLAR FOREST is not in $\mathcal{FPT}$, as a polynomial-time algorithm for $\ell = 1$ would imply $\mathcal{P} = \mathcal{NP}$. Thus, from the perspective of parameterized complexity, unless $\mathcal{P} = \mathcal{NP}$, the complexities of the SPANNING CATERPILLAR FOREST and 2-LAYER PLANARIZATION problems are different. In this sense the parameterized complexity provides a more fine-grained classification of the complexity of problems compared with the traditional complexity approach.

Our results. In this paper we apply more practical methods from the theory of fixed parameter tractability to obtain algorithms for the 1- and 2-LAYER PLANARIZATION problems. In particular, using a "kernelization" approach we obtain an $\mathcal{O}(\sqrt{k}\cdot 17^k+|G|)$ time algorithm for 2-LAYER PLANARIZATION in a graph G, which we improve to $\mathcal{O}(k\cdot 6^k+|G|)$ using a "bounded search tree" approach combined with kernelization. Here $|G| = |V| + |E|$ for a graph $G = (V, E)$. For small values of k, the 2-LAYER PLANARIZATION problem is thus solvable optimally in a reasonable amount of time. We solve the 1-LAYER PLANARIZATION problem in $\mathcal{O}(3^k \cdot |G|)$ time using the bounded search tree approach. As a by-product of our study and the fact that 2-layer planarization can be solved optimally for trees [22], we derive a linear-time 2-approximation algorithm for the optimization version of the 2-LAYER PLANARIZATION problem, and a polynomial-time 3-approximation algorithm for the optimization version of the 1-LAYER PLANARIZATION.

This paper is organized as follows. After definitions and preliminary results in Section 2, we present the kernelization approach for 2-LAYER PLANARIZATION in Section 3. Section 4 describes our bounded search tree algorithm for the same problem. In Section 5 we consider the 1-LAYER PLANARIZATION problem, and present a bounded search tree algorithm for its solution. Section 6 describes constant approximation algorithms for the optimization versions of the 1- and 2-LAYER PLANARIZATION problems. We conclude in Section 7.

2. Preliminaries. In this section we introduce notation, recall a characterization of biplanar graphs, formalize the problem statements, and give an overview of the kernelization method.

In this paper each graph $G = (V, E)$ is simple and undirected. The set of vertices of G is sometimes denoted by $V(G)$ and the set of edges by $E(G)$. The following terms are defined for an arbitrary set of edges $S \subseteq E(G) \cup E(\overline{G})$ and an arbitrary edge $vw \in E(G) \cup E(\overline{G})$, where $\overline{G}$ is the complement of G. $G\backslash S$ denotes the graph with vertex set $V(G)$ and edge set $E(G)\backslash S$, and $G\backslash vw$ denotes $G\backslash\{vw\}$. $G \cup S$ denotes the graph with vertex set $V(G)$ and edge set $E(G) \cup S$, and $G \cup vw$ denotes $G \cup \{vw\}$. $S\backslash vw$ denotes $S\backslash\{vw\}$, and $S \cup vw$ denotes $S \cup \{vw\}$.

A vertex with degree 1 is a *leaf*. If vw is the edge incident to a leaf w, then we say w is a *leaf at* v and vw is a *leaf edge at* v. The *degree* of a vertex v in graph G is the number of edges incident to v in G, and is denoted by $\deg_G(v)$, or $\deg(v)$ if the graph G is clear from the context. The *non-leaf degree* of a vertex v in G is the number of non-leaf edges incident to v in G, and is denoted by $\deg'_G(v)$, or $\deg'(v)$ if the graph G is clear from the context.

2.1. *Biplanar Graphs*. A bipartite graph is *biplanar* if it admits a biplanar drawing, that is, a 2-layer drawing with no edge crossings. For example, a path is biplanar, but not all trees are biplanar. For example, the 2-*claw* shown in Figure 1(b) is not biplanar. Also, graphs containing cycles are clearly not biplanar.

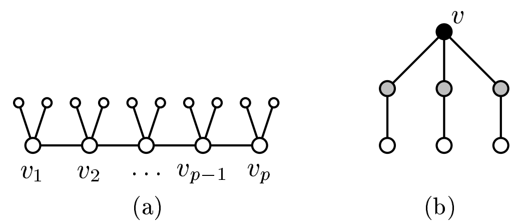


Fig. 1. (a) A caterpillar with spine $v_1, \dots, v_p$. (b) A 2-claw centred at v.

To state the characterization for biplanar graphs we first formalize this terminology. A graph is a *caterpillar* if deleting all the leaves produces a (possibly empty) path (see Figure 1(a)). This path is the *spine* of the caterpillar. A 2-*claw* is a graph consisting of one degree-3 vertex, the *centre*, coloured black in Figure 1(b), which is adjacent to three degree-2 vertices, coloured gray in Figure 1(b), each of which is adjacent to the centre and one leaf. If v is the centre of a 2-claw C, then C is *centred at* v. Edges of C incident to v are called *primary* edges of C. Edges of C incident to the leaves of C are called *secondary* edges of C.

Biplanar graphs are easily characterized, and there is a simple linear-time algorithm to recognize biplanar graphs, as the next lemma makes clear.

LEMMA 1 [8], [12], [25]. *Let G be a graph. The following are equivalent*:

(a) *G is biplanar.*
(b) *G is a forest of caterpillars* (*see Figure* 2).
(c) *G is acyclic and contains no* 2-*claw.*
(d) *The graph obtained from G by deleting all leaves is a forest and contains no vertex of degree* 3 *or greater.*

Lemma 1 implies that any planarization algorithm must destroy all cycles and 2-claws. Hence the vertices with non-leaf degree at least 3 are of particular interest since each such vertex lies on a cycle or a 2-claw, as demonstrated in the next lemma.

LEMMA 2. *If there exists a vertex v in a graph G such that* $\deg'_G(v) \geq 3$ *then G contains a* 2-*claw or a* 3- *or* 4-*cycle containing* v.

PROOF. Let w_1, w_2, w_3 be three distinct non-leaf neighbours of v. If some pair of these neighbours is adjacent then there is a 3-cycle containing v. Otherwise, let x_i be a

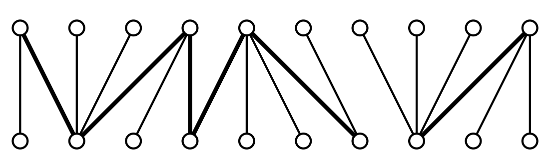


Fig. 2. A biplanar graph is a forest of caterpillars. Spine edges are thick.

neighbour of w_i such that $x_i \neq v$, $1 \leq i \leq 3$. Such an x_i exists since w_i is not a leaf. If all x_i are distinct then $\{v, w_1, w_2, w_3, x_1, x_2, x_3\}$ forms a 2-claw; otherwise G contains a 4-cycle through v. □

We define $V_3 = \{v \in V\colon \deg'(v) \geq 3\}$ and $V_3' = \{w \in V \backslash V_3\colon \deg(w) \geq 2, \exists v \in V_3 \text{ s.t. } vw \in E\}$. That is, V_3 is the set of vertices with at least three non-leaf neighbours, and V_3' is the set of non-leaf neighbours of vertices in V_3 that are not themselves in V_3. Observe that the centre of a 2-claw is in V_3. In Figure 1 and subsequent illustrations, vertices in V_3 are black and vertices in V_3' are gray. In fact, when convenient, we sometimes refer to vertices of G in V_3 as *black*, those in V_3' as *gray*, and vertices that are neither in V_3 nor in V_3' as *white*.

2.2. *Problem Statements*. A set T of edges of a (not necessarily bipartite) graph G is called a *biplanarizing set* if $G \backslash T$ is biplanar. The *bipartite planarization number* of a graph G, denoted by $\mathsf{bpr}(G)$, is the size of a minimum biplanarizing set for G. Thus the 2-LAYER PLANARIZATION problem is: Given a graph G and an integer k, is $\mathsf{bpr}(\mathrm{G}) \leq k$? For a given bipartite graph $G = (A, B; E)$ and permutation π of A, the 1-*layer biplanarization number* of G and π, denoted $\mathsf{bpr}(G, \pi)$, is the minimum number of edges in G whose deletion produces a graph that admits a biplanar drawing with π as the ordering of the vertices in A. The 1-LAYER PLANARIZATION problem asks if $\mathsf{bpr}(\mathrm{G}, \pi) \leq k$.

2.3. *Terminology*. To describe our kernelization algorithm we introduce some terminology. A *component caterpillar* of a graph is a connected component that is a caterpillar. Let $P = (v_1, v_2, \ldots, v_p)$ be a path in G with $p \geq 3$ vertices. If $\deg'_G(v_1) \geq 3$, $\deg'_G(v_i) = 2$ for all i, $1 < i < p$, and $\deg'_G(v_p) = 1$, then P together with all the leaves at vertices $v_2, \ldots, v_p$ comprises a *pendant caterpillar*, as illustrated in Figure 3(a). A pendant caterpillar is said to be *connected* at v_1, its *connection point*. If $\deg'_G(v_1) \geq 3$, $\deg'_G(v_i) = 2$ for all i, $1 < i < p$, and $\deg'_G(v_p) = 3$, then P together with all the leaves at vertices $v_2, \ldots, v_{p-1}$ comprises an *internal caterpillar*, as illustrated in Figure 3(b). An internal caterpillar is said to be *connected* at v_1 and v_p, its *connection points*. An internal caterpillar where $p = 4$, $\deg_G(v_2) = 2$, and $\deg_G(v_3) = 2$ is called an *internal 3-path*. Edge v_2v_3 in an internal 3-path is called its *middle edge*. The number of edges in a pendant (or internal) caterpillar is called its *size*.

Consider a p-cycle $C = (v_1, v_2, \ldots, v_p)$ in G. If $\deg'_G(v_i) = 2$ for all i, $1 \leq i \leq p$, then C together with all the leaves at $v_1, \ldots, v_p$ comprises a *component wreath*, as illustrated in Figure 4(a). A graph consisting of one component wreath is a *wreath*. If $\deg'_G(v_1) \geq 3$, and $\deg'_G(v_i) = 2$ for all i, $2 \leq i \leq p$, then C together with all the leaves

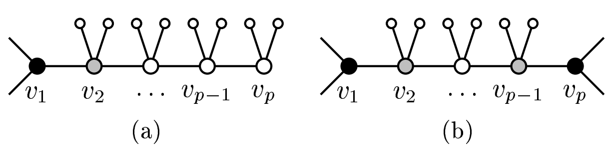


Fig. 3. (a) A pendant and (b) an internal caterpillar.

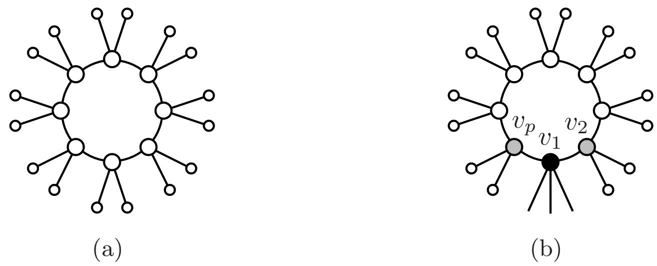


Fig. 4. (a) A wreath, and (b) a pendant wreath.

at $v_2, \dots, v_p$ comprises a *pendant wreath*, as illustrated in Figure 4(b). A pendant wreath is said to be *connected* at v_1, its *connection point*. The number of edges in a pendant wreath is called its *size*. A pendant wreath of size three is called a *pendant triangle*. Edge v_2v_3 in a pendant triangle is called its *middle edge*. Edges that lie on C are called *cycle edges* of a (pendant, component) wreath, and C is called a *wreath cycle*.

Notice that a connected graph that does not have a vertex v with $\deg'(v) \geq 3$ is either a caterpillar or a wreath, and that any two of the structures defined above are edge-disjoint. For example, an edge of G cannot belong to two internal caterpillars, or to a pendant caterpillar and an internal caterpillar. In particular, these structures are maximal: for example an internal caterpillar cannot contain another internal caterpillar.

2.4. *Regular Biplanarizing Sets*. A biplanarizing set T of G is *regular* if T contains no leaf edge, and each edge of T with both endpoints white is in a component wreath of G. The following lemma shows that to solve our planarization problem, it suffices to look for regular biplanarizing sets.

LEMMA 3. *Let T be a biplanarizing set of G. There exists a regular biplanarizing set T' of G such that $|T'| \leq |T|$.*

PROOF. We first show that there exists a biplanarizing set T'' that contains no leaf edge and has $|T''| \leq |T|$. Let F be the forest of caterpillars $G \backslash T$. To prove that T'' exists, it suffices to show that for any leaf edge $vw \in T$, either $F \cup vw$ is biplanar or there exists a non-leaf edge $e \in E(G)$ such that $(F \cup vw) \backslash e$ is biplanar. F contains a component $F_w = \{w\}$ and a component F_v such that $v \in F_v$. $F \cup vw$ is biplanar if and only if $F' = F_v \cup F_w \cup vw$ is biplanar. Since $\deg_G(w) = 1$, F' is acyclic. Now suppose that there is a 2-claw C in F' centred at x. Since w is a leaf, $v \neq x$ and $w \neq x$. Since each 2-claw in F' contains vw, edge vw is a secondary edge of C. Since F_v is a caterpillar and thus has no 2-claws, it follows that v has no neighbours other than x and w in F'. Therefore $F' \backslash xv$ is biplanar since it consists of one component that is an edge vw and a second component that is a subgraph of F_v. Since xv is not a leaf edge in G, this completes the proof that T'' exists.

We now show that T'' can be transformed into a regular biplanarizing set T' such that $|T'| \leq |T''|$. Let H denote $G \backslash T''$. Suppose that T'' contains an edge vw, where both v and w are white and vw does not belong to a component wreath. To complete the proof, it suffices to show that if $H' = H \cup vw$ is not biplanar then there exists a non-leaf edge xy with x or y non-white such that $H' \backslash xy$ is biplanar. Assume H' is not biplanar. H' does not contain a 2-claw, since vw, having both endpoints white, does not belong to any 2-claw in G. Therefore, H' is comprised of component caterpillars and exactly one component wreath W that contains vw. By the assumption, vw does not belong to a component wreath of G, thus W is a subgraph of a component in G that is not a wreath. Therefore, the wreath cycle of W contains edge xy such that x or y is black in G. This completes the proof since $H' \backslash xy$ is clearly biplanar. □

3. Kernelization Algorithm. A basic method for developing FPT algorithms is to reduce a parameterized problem instance I to an "equivalent" instance I_{sf} (the *kernel*), where the size of I_{sf} is bounded by some function of the parameter. Then the instance I_{sf} is solved using an exhaustive search method, and its solution is then used to determine a solution to the original instance I. Downey and Fellows [4, Chapter 3.2] survey this general approach, which is known as *kernelization*.

We now give an overview of our kernelization algorithm for the 2-LAYER PLANARIZATION problem in a graph G. The construction of a kernelized instance has two phases. In the first phase we identify a set of edges $\mathcal{S}_G$ of G that may be assumed without loss of generality to be in a biplanarizing set. More precisely, there exists a minimum biplanarizing set that contains $\mathcal{S}_G$.

In the second phase, loosely speaking, we contract the edges of G with both endpoints white. The intuition behind this is that an edge with both endpoints white does not belong to any 2-claw. Furthermore, contracting such edges preserves cycles. We obtain a graph G_{sf} such that $\mathsf{bpr}(G) = \mathsf{bpr}(G_{\mathsf{sf}}) + |\mathcal{S}_G|$. We then use exhaustive search to determine if G_{sf} has a biplanarizing set of size at most $k - |\mathcal{S}_G|$. If it does, we use this to produce a biplanarizing set for G with k edges in $\mathcal{O}(|G|)$ time.

To obtain a total running time of the form $f(k) + \mathcal{O}(|G|)$, we prove that if a certain condition is satisfied, then $|G_{\mathsf{sf}}| \in \mathcal{O}(k)$. In fact this condition is necessary for $\mathsf{bpr}(G_{\mathsf{sf}}) \leq k$ and can be tested in $\mathcal{O}(|G_{\mathsf{sf}}|)$ time.

3.1. *First Phase*. We now describe the first phase of the kernelization. For a given graph G, construct $\mathcal{S}_G$ as follows:

- Add a cycle edge from each component wreath to $\mathcal{S}_G$.
- For each pendant wreath, if it is a pendant triangle, add its middle edge to $\mathcal{S}_G$; otherwise, add a cycle edge incident to the connection point to $\mathcal{S}_G$.

The next two technical lemmas will enable us to prove, in Lemma 6, that $\mathcal{S}_G$ belongs to a minimum biplanarizing set for G.

LEMMA 4. *Let vw be the middle edge of a pendant triangle of G. If* $\mathsf{bpr}(G) \leq k$ *then* $\mathsf{bpr}(G \backslash vw) \leq k - 1$.

PROOF. Let T be a biplanarizing set for G with $|T| = k$ and let (v, w, x) be a pendant triangle connected at x. Then $vw \in \mathcal{S}_G$. Let $G' = G \backslash vw$.

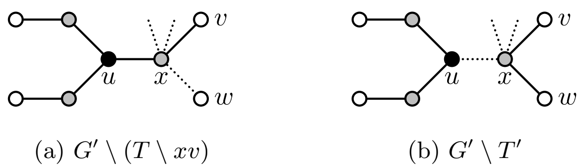


Fig. 5. Replacing xv and xw by xu in a biplanarizing set.

Since biplanar graphs are acyclic, T must contain some edge from the triangle. If $vw \in T$ then $T \backslash vw$ is a biplanarizing set for G' with at most $k-1$ edges; thus $\mathsf{bpr}(G') \leq k-1$. Now assume $vw \notin T$. Hence, at least one of xv or xw, say xv, belongs to T. If $T \backslash xv$ is a biplanarizing set for G' then $\mathsf{bpr}(G') \leq k-1$. Now assume $T \backslash xv$ is not a biplanarizing set for G'. By Lemma 1(c), $G' \backslash (T \backslash xv)$ contains a subgraph C that is a cycle or 2-claw. Furthermore, since T is a biplanarizing set for G', edge xv belongs to C. Since xv is not in a cycle of G', C is a 2-claw. C must be centred at some vertex $u \notin \{v, w\}$ that is adjacent to x, as illustrated in Figure 5(a). All edges xy ($y \neq \{v, u\}$), including xw, must be in T, as otherwise $G \backslash T$ contains a 2-claw. Thus all the 2-claws in $G' \backslash (T \backslash xv)$ must be centred at u and contain xv. Now replace xv and xw in T by xu to obtain $T' = (T \backslash \{xv, xw\}) \cup xu$, as illustrated in Figure 5(b). Since T' is a biplanarizing set for G' with $|T'| = |T| - 1 \leq k-1$, $\mathsf{bpr}(G') \leq k-1$. □

LEMMA 5. *Let e be a cycle edge incident to the connection point of a pendant wreath in a graph G that is not a pendant triangle. If* $\mathsf{bpr}(G) \leq k$ *then* $\mathsf{bpr}(G \backslash e) \leq k-1$.

PROOF. Let T be a biplanarizing set for G with $|T| = k$, and let W be a pendant wreath connected at v_1 with cycle $(v_1, \ldots, v_p)$ where $e = v_1v_2 \in \mathcal{S}_G$. Let $G' = G \backslash e$.

If $v_1v_2 \in T$ then $T \backslash v_1v_2$ is a biplanarizing set of G' with at most $k-1$ edges; thus $\mathsf{bpr}(G') \leq k-1$. Now assume v_1v_2 does not belong to T. Hence T must contain some other edge of the wreath cycle of W.

Consider first the case that T contains two edges e_1 and e_2 of W neither of which is v_1v_2. By the definition of a component wreath, $G \backslash \{v_1v_2, v_pv_1\}$ is comprised of a component H' and a component caterpillar. Furthermore, $G \backslash \{e_1, e_2\}$ has a component H'' such that H' is a subgraph of H''. Since $T \backslash \{e_1, e_2\}$ is a biplanarizing set for H'', $T \backslash \{e_1, e_2\}$ is also a biplanarizing set for H'. Thus $(T \backslash \{e_1, e_2\}) \cup \{v_1v_2, v_pv_1\}$ is a biplanarizing set for G; and, therefore, $(T \backslash \{e_1, e_2\}) \cup v_pv_1$ is a biplanarizing set for G' with at most $k-1$ edges, and hence $\mathsf{bpr}(G') \leq k-1$.

Now assume T contains exactly one edge of W. This edge must be a cycle edge v_iv_{i+1}, for some $2 \leq i \leq p$ where $v_{p+1} = v_1$. We claim that $T \backslash v_iv_{i+1}$ is a biplanarizing set of G'. Otherwise, by Lemma 1(c), $G' \backslash (T \backslash v_iv_{i+1})$ contains a subgraph C containing v_iv_{i+1} that is either a cycle or a 2-claw. Since the only cycle in G containing v_iv_{i+1} is the cycle of W, and this cycle does not occur in G', C is not a cycle. Thus C is a 2-claw, and its centre is either v_1 or some neighbour of v_1. We now show that in either case there is a 2-claw in $G \backslash T$, which contradicts the fact that T is a biplanarizing set for G.

If C is centred at some neighbour u of v_1 then $i = p$ and $(C \backslash v_pv_1) \cup v_1v_2$ is a 2-claw in $G \backslash T$.

Suppose now that C is centred at v_1. Then $i \in \{p-1, p\}$. First suppose $i = p-1$. If there is a leaf y at v_p then $(C \backslash v_{p-1}v_p) \cup v_p y$ is a 2-claw in $G \backslash T$. (Recall that $v_p y \notin T$ since $v_i v_{i+1}$ is the only edge in $T \cap W$.) Suppose now there is no leaf at v_p. Since W is not a pendant triangle, there is a 2-path (v_1, v_2, z) in G not containing the edge $v_{p-1}v_p$; here the vertex z is either v_3 or some leaf at v_2. Then $(C \backslash \{v_{p-1}v_p, v_p v_1\}) \cup \{v_1 v_2, v_2 z\}$ is a 2-claw in $G \backslash T$. If $i = p$ then $(C \backslash v_p v_1) \cup \{v_1 v_2, v_2 v_3\}$ contains a 2-claw in $G \backslash T$.

No matter where C is centred there is a 2-claw in $G \backslash T$, which is a contradiction. Thus our claim is proved; that is, $T \backslash v_i v_{i+1}$ is a biplanarizing set for G' with $|T| - 1 \leq k - 1$ edges. Thus, $\mathsf{bpr}(G') \leq k - 1$. $\square$

Now we prove that $\mathcal{S}_G$ belongs to a minimum biplanarizing set for G.

LEMMA 6. *For every graph G, $\mathsf{bpr}(G) \leq k$ if and only if $\mathsf{bpr}(G \backslash \mathcal{S}_G) \leq k - |\mathcal{S}_G|$.*

PROOF. Denote the subgraph $G \backslash \mathcal{S}_G$ by G'. We first prove that if $\mathsf{bpr}(G') \leq k - |\mathcal{S}_G|$ then $\mathsf{bpr}(G) \leq k$. Let T' be a biplanarizing set for G' with $|T'| \leq k - |\mathcal{S}_G|$. Then $T = T' \cup \mathcal{S}_G$ is a biplanarizing set for G with $|T| \leq k$. Hence $\mathsf{bpr}(G) \leq k$.

We now prove that if $\mathsf{bpr}(G) \leq k$ then $\mathsf{bpr}(G') \leq k - |\mathcal{S}_G|$. It suffices to show that if G has a pendant or component wreath W, then $\mathsf{bpr}(G \backslash (\mathcal{S}_G \cap W)) \leq k - 1$. Lemmas 4 and 5 prove this in the case that W is a pendant wreath.

Now consider the remaining case, in which W is a component wreath. Let T be a biplanarizing set for G with $|T| = k$. Since biplanar graphs are acyclic, T must contain some edge vw from the cycle of W. Then $T \backslash vw$ is a biplanarizing set for $G \backslash (\mathcal{S}_G \cap W)$ since $W \backslash (\mathcal{S}_G \cap W)$ is a caterpillar. Thus $\mathsf{bpr}(G \backslash (\mathcal{S}_G \cap W)) \leq k - 1$. $\square$

This completes the first phase of the kernelization. We now describe the second phase.

3.2. *Second Phase*. By Lemma 6 we may now assume that the input graph contains neither pendant nor component wreaths. Instead of working with a problem instance G' with parameter k', we can work with $G = G' \backslash \mathcal{S}_{G'}$ and parameter $k = k' - |\mathcal{S}_{G'}|$.

The graph induced by the white vertices in any graph is comprised of component wreaths and a forest of caterpillars. Since G has no wreaths by the above assumption, the induced graph is a forest of caterpillars. This motivates the second phase of our algorithm. Specifically, in the second phase we construct a *kernel graph* $G_{\mathsf{sf}} = (V_{\mathsf{sf}}, E_{\mathsf{sf}})$ from G by performing the following reduction operations on G in the order given below.

Reduction operations

1. For each vertex v of G, replace a set of leaf edges at v by a single leaf edge at v.
2. While there is an edge vw of G with both v and w white in G, contract vw.
3. Delete isolated vertices.

Since the graph induced by the white vertices of G is a forest of caterpillars, the above operations create neither loops nor multiple edges. Therefore, the cycle structure of G is not affected by the reduction operations as there is a bijection between the cycles of G

before and after these operations. Similarly, the set of non-leaf edges of G with at least one non-white endpoint is also preserved by the reduction operations.

LEMMA 7. *For every graph G with $\mathcal{S}_G = \emptyset$, $\mathsf{bpr}(\mathrm{G}) \le k$ if and only if $\mathsf{bpr}(\mathrm{G}_{\mathsf{sf}}) \le k$.*

PROOF. Let $\mathcal{S} \subseteq E(G)$ be the set of non-leaf edges of G with at least one non-white endpoint. The reduction operations do not affect the edges in $\mathcal{S}$, thus $\mathcal{S} \subseteq E_{\mathsf{sf}}$. By Lemma 3, there exists a minimum biplanarizing set of G that is a subset of $\mathcal{S}$, and equivalently there exists a minimum biplanarizing set of G_{sf} that is a subset of $\mathcal{S}$. To prove the lemma it suffices to show that $T \subseteq \mathcal{S}$ is a biplanarizing set of G if and only if T is a biplanarizing set of G_{sf}.

Consider the graph G' obtained from G after completing reduction operation 1. Since T contains no leaves, T is a biplanarizing set of G if and only if T is a biplanarizing set of G'. Therefore, we need only prove that T is a biplanarizing set of G' if and only if T is a biplanarizing set of G_{sf}.

Since, by the assumption, G' has no component wreaths, and as previously pointed out, the cycle structure of G' is not affected by the reduction operations. Thus the existence of a cycle in $G' \backslash T$ implies that there is a cycle in $G_{\mathsf{sf}} \backslash T$ and, equivalently, the existence of a cycle in $G_{\mathsf{sf}} \backslash T$ implies that there is a cycle in $G' \backslash T$. Furthermore, since no 2-claw contains an edge with both endpoints white, there is a bijection between the 2-claws in G' and the 2-claws in G_{sf}. Therefore, the existence of a 2-claw in $G' \backslash T$ implies that there is a 2-claw in $G_{\mathsf{sf}} \backslash T$ and, equivalently, the existence of a 2-claw in $G_{\mathsf{sf}} \backslash T$ implies that there is a 2-claw in $G \backslash T$. $\square$

To prove that $|G_{\mathsf{sf}}| \in \mathcal{O}(k)$ (assuming $\mathsf{bpr}(\mathrm{G}_{\mathsf{sf}}) \le k$ and $\mathcal{S}_G = \emptyset$), we introduce the following potential function, whose definition is suggested by Lemma 1(d). For a graph $G = (V, E)$, define

$$\forall v \in V, \qquad \Phi_G(v) = \max\{\deg'_G(v) - 2, 0\} \quad \text{and} \quad \Phi(G) = \sum_{v \in V} \Phi_G(v).$$

Intuitively, $\Phi_G(v)$ approximates the number of edges in the distance-2 neighbourhood of v that must be included in a biplanarizing set for G.

LEMMA 8. *$\Phi(G) = 0$ if and only if G is a collection of caterpillars and wreaths.*

PROOF. Since neither caterpillars nor wreaths have vertices with non-leaf degree greater than 2, their potential function is clearly equal to zero. For the other direction, suppose $\Phi(G) = 0$ and consider a graph G' obtained from G by deleting all its leaves. G' does not have a vertex of degree 3 or more, so G' is a collection of paths and cycles. Therefore, G is a collection of caterpillars and wreaths. $\square$

Notice that Lemma 8 proves another characterization of biplanar graphs. Namely, G is biplanar if and only if G is acyclic and $\Phi(G) = 0$. Thus for graphs G with $\Phi(G) = 0$, a minimum biplanarizing set of G consists of one cycle edge from each component wreath. For graphs with $\Phi(G) > 0$ the following observation will be useful.

LEMMA 9. *Let G be a graph with $\Phi(G) > 0$ (that is, $V_3 \neq \emptyset$), and let d denote the average non-leaf degree of vertices in V_3. Then $|V_3| = \Phi(G)/(d-2)$.*

PROOF. By definition,

$$d|V_3| = \sum_{v \in V_3} \deg'(v) = \sum_{v \in V_3} (\Phi_G(v) + 2) = \Phi(G) + 2|V_3|.$$

Thus, $(d-2)|V_3| = \Phi(G)$, and the result follows. □

We now prove that $\Phi(G)$ provides a lower bound for bpr(G).

LEMMA 10. *For every graph G, $\mathsf{bpr}(G) \geq \frac{1}{2}\Phi(G)$.*

PROOF. The result follows from Lemma 8 if we prove that deleting one edge vw from G with $\Phi(G) > 0$ reduces $\Phi(G)$ by at most two; that is, $\Phi(G) - \Phi(G \backslash vw) \leq 2$. For every vertex $u \in V(G)$, let $\Phi^*(u) = \Phi_G(u) - \Phi_{G\backslash vw}(u)$.

First suppose that at least one of v and w (say v) is a leaf in G. Then $\Phi^*(v) = 0$ and $\Phi^*(w) = 0$. If w becomes a leaf by deleting vw, then w has one neighbour x for which $\Phi^*(x) \leq 1$. Thus $\Phi(G) - \Phi(G \backslash vw) \leq 2$. Now suppose that neither v nor w are leaves in G then there are three possible outcomes when the edge vw is deleted.

Case 1: $\Phi^*(v) > 0$ *and* $\Phi^*(w) > 0$. Then $\deg'_G(v) \geq 3$ and $\deg'_G(w) \geq 3$. Thus $\Phi^*(v) = 1$ and $\Phi^*(w) = 1$. Furthermore, v and w do not become leaves by deleting vw, and $\Phi^*(u) = 0$ for all vertices $u \notin \{v, w\}$. Therefore, $\Phi(G) - \Phi(G \backslash vw) \leq 2$.

Case 2: *Exactly one of* $\Phi^*(v)$ *and* $\Phi^*(w)$ *is positive, say* $\Phi^*(v)$. Then $\deg'_G(v) \geq 3$ and $\deg'_G(w) \leq 2$. Thus, w has at most one neighbour x ($\neq v$) such that $\Phi^*(x) > 0$. Furthermore $\Phi^*(x) \leq 1$. Thus for each neighbour $u \neq x$ of v, $\Phi^*(u) = 0$. Therefore, $\Phi(G) - \Phi(G \backslash vw) \leq 2$.

Case 3: $\Phi^*(v) = 0$ *and* $\Phi^*(w) = 0$. Thus, $\deg'_G(v) \leq 2$ and $\deg'_G(w) \leq 2$. Each of v and w has at most one neighbour x and y, respectively, for which $\Phi^*(x) > 0$ and $\Phi^*(y) > 0$. If $x \neq y$, then $\Phi^*(x) \leq 1$ and $\Phi^*(y) \leq 1$; if $x = y$ then $\Phi^*(x = y) \leq 2$. Therefore, $\Phi(G) - \Phi(G \backslash vw) \leq 2$. □

Now consider an instance (G, k) of the 2-LAYER PLANARIZATION problem such that $\mathcal{S}_G = \emptyset$. If $2k < \Phi(G_{\mathsf{sf}})$ then we can immediately conclude from Lemma 10 that $\mathsf{bpr}(G_{\mathsf{sf}}) > k$ and hence $\mathsf{bpr}(G) > k$. On the other hand, if $2k \geq \Phi(G_{\mathsf{sf}})$ then, as we now prove, the size of the kernel graph is bounded by a function solely of k.

LEMMA 11. *For every graph G and integer k, if $\mathcal{S}_G = \emptyset$ and $2k \geq \Phi(G_{\mathsf{sf}})$, then the kernel has at most $20k$ edges and $|G_{\mathsf{sf}}| \in \mathcal{O}(k)$.*

PROOF. If $|V_3| = 0$, then $\Phi(G) = 0$. In that case, since G has no component wreaths, G is biplanar and $\mathsf{bpr}(G) = 0$. Furthermore, since all the vertices of G are white, G_{sf} is the empty graph and thus $|G_{\mathsf{sf}}| = 0 \leq 20k$.

Consider now the case that $|V_3| > 0$. We count the edges in G_{sf} with respect to the black vertices; that is, the vertices in V_3. Since each gray vertex in G_{sf} has at most two non-leaf neighbours, at least one of which is black, the number of non-leaf edges in G_{sf} is at most $2\sum_{v\in V_3}\deg'_{G_{\mathsf{sf}}}(v)$. Furthermore, since every leaf vertex is white, only black and gray vertices may be incident to leaf edges. Therefore the total number of leaf edges in G_{sf} is at most $|V_3|+|V'_3|$. Since the number of vertices in V'_3 is at most $\sum_{v\in V_3}\deg'_{G_{\mathsf{sf}}}(v)$, we have

$$|E_{\mathsf{sf}}| \le |V_3| + \sum_{v\in V_3} 3\deg'_{G_{\mathsf{sf}}}(v) = |V_3| + \sum_{v\in V_3}(3\Phi_{G_{\mathsf{sf}}}(v)+6) = 3\Phi(G_{\mathsf{sf}}) + 7|V_3|.$$

By Lemma 9 applied to G_{sf} and since $d \ge 3$, we have $|V_3| \le \Phi(G_{\mathsf{sf}})$. Since by assumption $\Phi(G_{\mathsf{sf}}) \le 2k$, we have $|E_{\mathsf{sf}}| \le 20k$, and since G_{sf} has no isolated vertices, $|G_{\mathsf{sf}}| \in \mathcal{O}(k)$. □

One solution to the 2-LAYER PLANARIZATION problem is to search through all sets of edges in G_{sf} with k edges. We obtain an algorithm whose running time is $\mathcal{O}(k\cdot\binom{20k}{k}+|G|)$. However, we now prove that we need only search through a subset $\mathcal{K}$ of the edges in G_{sf}. Let $\mathcal{K}$ contain all the non-leaf edges vw of G_{sf} such that if v is black and w is gray then w is not an endpoint of a middle edge in G_{sf}. The set $\mathcal{K}$ is called the *subkernel* of G_{sf}.

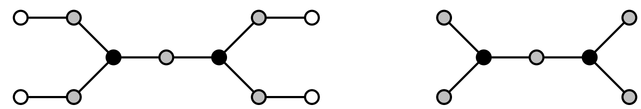
It is convenient to introduce the idea of a minimal biplanarizing set. A biplanarizing set T for a graph G is *minimal* if $T\setminus vw$ is not a biplanarizing set for any edge $vw \in T$.

LEMMA 12. *Let T be a biplanarizing set for G_{sf}. Then there exists a biplanarizing set for G_{sf} contained in $\mathcal{K}$ with at most $|T|$ edges.*

PROOF. By Lemma 3, we may assume that T is a minimal regular biplanarizing set for G_{sf}, and hence T contains no leaf edges of G_{sf}. Thus to prove the lemma, it suffices to show that for every non-leaf edge $e \in T$ such that $e \notin \mathcal{K}$ there is an edge $e' \in \mathcal{K}$ such that $(T\setminus e)\cup e'$ is a biplanarizing set for G_{sf}.

Suppose T contains a non-leaf edge $vw \notin \mathcal{K}$. Then vw is a non-middle edge in an internal 3-path; let $wx \in \mathcal{K}$ be the middle edge of that internal 3-path. Let $G'_{\mathsf{sf}} = (G_{\mathsf{sf}}\setminus T)\cup vw$. By minimality of T, G'_{sf} is not biplanar. We prove the lemma by demonstrating that either $G'_{\mathsf{sf}}\setminus wx$ is biplanar or there is a neighbour $u \ne w$ of v such that $uv \in \mathcal{K}$ and $G'_{\mathsf{sf}}\setminus uv$ is biplanar.

Every cycle and 2-claw in G'_{sf} must contain vw. Since w has degree 2, every cycle in G'_{sf} also contains wx. Thus $G'_{\mathsf{sf}}\setminus wx$ is a forest. If $G'_{\mathsf{sf}}\setminus wx$ is biplanar we are done; otherwise, there is at least one 2-claw, C, in $G'_{\mathsf{sf}}\setminus wx$. Since w is a leaf in $G'_{\mathsf{sf}}\setminus wx$, vw must be a secondary edge of C. Thus C is centred at some neighbour $u \ne w$ of v, and therefore $u \in V_3$ and $uv \in \mathcal{K}$. Thus $\deg_{G'_{\mathsf{sf}}}(v) = 2$ (as otherwise $G_{\mathsf{sf}}\setminus T$ has a 2-claw) and therefore we know that in G'_{sf} vertices v, w, and x have degree at most 2. Since every cycle in G'_{sf} must contain vw, these vertex degrees imply that every cycle in G'_{sf} must contain uv. Hence $G'_{\mathsf{sf}}\setminus uv$ is a forest. Furthermore, $\deg_{G'_{\mathsf{sf}}\setminus uv}(v) = 1$ and all the vertices in the distance two neighbourhood of v in $G'_{\mathsf{sf}}\setminus uv$ have degree in G'_{sf} at most 2. Thus $G'_{\mathsf{sf}}\setminus uv$ is biplanar, which completes the proof since $uv \in \mathcal{K}$. □



(a) (b)

Fig. 6. A graph with biplanarization number one and with a subkernel of six edges.

LEMMA 13. *Let (G, k) be an instance of the* 2-LAYER PLANARIZATION *problem such that $\mathcal{S}_G = \emptyset$ and $2k \geq \Phi(G_{\mathsf{sf}}) > 0$. If $\mathsf{bpr}(\mathsf{G}_{\mathsf{sf}}) \leq k$ then $|\mathcal{K}| \leq 2kd/(d-2)$.*

PROOF. By definition of $\mathcal{K}$, each edge in $\mathcal{K}$ is not a leaf edge and at least one of its endpoints is in V_3. Therefore when counting the number of edges in $\mathcal{K}$ with respect to the vertices in V_3, we have

$$|\mathcal{K}| \leq \sum_{v \in V_3} \deg'_{G_{\mathsf{sf}}}(v) = \sum_{v \in V_3} (\Phi_{G_{\mathsf{sf}}}(v) + 2) = 2|V_3| + \Phi(G_{\mathsf{sf}}).$$

By Lemma 9, $|\mathcal{K}| \leq \Phi(G_{\mathsf{sf}})(1 + 2/(d-2)) = \Phi(G_{\mathsf{sf}})\, d/(d-2)$. Since $\Phi(G_{\mathsf{sf}}) \leq 2k$, $|\mathcal{K}| \leq 2k\, d/(d-2)$. □

Since $d \geq 3$, the size of the subkernel $|\mathcal{K}| \leq 6k$. The graph illustrated in Figure 6(a) has biplanarization number one, and its subkernel of six edges is shown in Figure 6(b). Thus our analysis for the size of the subkernel is tight.

Recall the definition of pendant caterpillar from Section 2.3. An internal caterpillar where $p = 3$ and $\deg_G(v_2) = 2$ is called a *pendant* 2*-path*. In the example of Figure 6, it is not necessary to include the edges contained in the pendant 2-paths in any biplanarizing set. This observation suggests the following methods for further reducing the size of the subkernel.

OBSERVATION 1. *For each vertex $v \in V_3$, let $\alpha(v)$ be the number of pendant 2-paths connected at v in G_{sf}.*

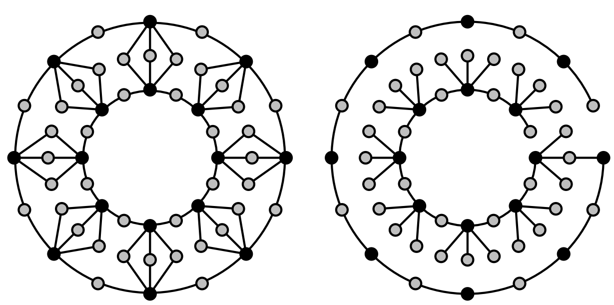
1. *Every biplanarizing set must contain* $\max\{\alpha(v) - 2, 0\}$ *edges from the $\alpha(v)$ 2-paths connected at v. We may as well put* $\max\{\alpha(v) - 2, 0\}$ *of the edges incident to v in these pendant 2-paths in a biplanarizing set. Thus we can assume that $\alpha(v) \leq 2$.*
2. *If $\alpha(v) = 2$ and $\deg'_{G_{\mathsf{sf}}}(v) = 3$ then none of the edges in these two 2-paths need be in $\mathcal{K}$.*
3. *If $\alpha(v) = 1$, then none of the edges in the pendant 2-path need be in $\mathcal{K}$.*

PROOF. The correctness of the first observation is obvious. Now consider the second observation. Let $T \subseteq \mathcal{K}$ be a biplanarizing set for G_{sf}. Let x, y, and z be the non-leaf neighbours of v where vx and vy belong to the two pendant 2-paths connected at v. Since $T \subseteq \mathcal{K}$, no edge of the two pendant 2-paths other than vx and vy may belong to T. If z is an endpoint of a middle edge in G_{sf}, let $w \neq v$ be the neighbour of z and let $T' = (T \backslash \{vx, vy\}) \cup zw$. Otherwise, let $T' = (T \backslash \{vx, vy\}) \cup vz$. $G_{\mathsf{sf}} \backslash T'$ is clearly biplanar. Since the edge (zw or vz) added to the biplanarizing set belongs to $\mathcal{K}$, and since neither vz nor zw belongs to any pendant 2-path in G_{sf}, the correctness of the second observation follows.

Consider now the third observation. Let x denote the non-leaf neighbour of v that belongs to the pendant 2-path connected at v. Let $T \subseteq \mathcal{K}$ be a minimum biplanarizing set for G_{sf}. If T contains no edge of the pendant 2-path we are done; otherwise, since $T \subseteq \mathcal{K}$, we may assume that T contains vx and no other edge of the pendant 2-path. $(G \backslash T) \cup vx$ is acyclic and by minimality of T it contains a 2-claw C. C must contain vx. Thus v has a non-leaf neighbour $z \neq x$. If z is an endpoint of a middle edge in G_{sf} let $w \neq v$ be the neighbour of z and let $T' = (T \backslash vx) \cup zw$. Otherwise, let $T' = (T \backslash vx) \cup vz$. $G_{\mathsf{sf}} \backslash T'$ is clearly biplanar, and since the edge (zw or vz) added to the biplanarizing set belongs to $\mathcal{K}$, and since neither vz nor zw belong to any pendant 2-path in G_{sf}, the correctness of the third observation follows. □

While in many cases arising in practice the above observations could lead to improved running time for our algorithm, we now describe a pathological family of graphs for which our analysis in Lemma 12 for the size of the subkernel is tight even with the above improvements. Consider the graph $G_{p,q}$ $(p, q \in \mathbb{N})$ consisting of an *inner* cycle $(v_1, \ldots, v_{2p})$ and an *outer* cycle $(w_1, \ldots, w_{2p})$ with v_{2i} connected by q 2-paths to w_{2i} for all i, $1 \leq i \leq p$, as illustrated in Figure 7(a) in the case of $G_{8,3}$. All vertices in V_3 have non-leaf degree $d = q + 2$. $G_{p,q}$ has $(d+2)p$ vertices and $2dp$ edges. It is easily verified that the subkernel of $G_{p,q}$ is the whole graph. As shown in Figure 7(b), $G_{p,q}$ has a spanning caterpillar with $p(d+2) - 1$ edges. There is no larger biplanar subgraph than a spanning caterpillar. Thus $\mathsf{bpr}(G_{p,q}) = 2dp - (p(d+2) - 1) = p(d-2) + 1$. The ratio of the number of edges in the subkernel of $G_{p,q}$ to $\mathsf{bpr}(G_{p,q})$ is $2dp/(p(d-2)+1) \to 2d/(d-2)$ as $p \to \infty$. Thus the analysis of the size of the subkernel in Lemma 12 is tight for all d.

3.3. *Algorithm.* We now present our algorithm for the 2-LAYER PLANARIZATION problem based solely on kernelization.



(a) (b)

Fig. 7. (a) The graph $G_{8,3}$ and (b) a spanning caterpillar of $G_{8,3}$.

Algorithm Kernelization

Input: graph $G = (V, E)$
Parameter: non-negative integer k
Output: NO if $\mathsf{bpr}(G) > k$; otherwise, YES and a biplanarizing set for G

1. determine $\mathcal{S}_G$ and let $k' = k - |\mathcal{S}_G|$
2. determine the kernel $G_{\mathsf{sf}} = (V_{\mathsf{sf}}, E_{\mathsf{sf}})$ of $G \backslash \mathcal{S}_G$
3. **if** $\Phi(G_{\mathsf{sf}}) > 2k'$ **then** return NO
4. determine the subkernel $\mathcal{K} \subseteq E_{\mathsf{sf}}$ of G_{sf}
5. **if** $k' \geq |\mathcal{K}|$ **then** return YES and $\mathcal{K} \cup \mathcal{S}_G$
   else if $\exists\, T \subseteq \mathcal{K}$ such that $|T| = k'$, $G_{\mathsf{sf}} \backslash T$ is acyclic, and $\Phi(G_{\mathsf{sf}} \backslash T) = 0$ **then**
   return YES and $T \cup \mathcal{S}_G$
   else return NO

THEOREM 1. *Given a graph $G = (V, E)$ and integer k, the algorithm* Kernelization (G, k) *determines if* $\mathsf{bpr}(G) \leq k$ *and, if so, returns a biplanarizing set of size at most k. The running time is $\mathcal{O}(\sqrt{k} \cdot (2\mathbf{e}\, d/(d-2))^k + |G|)$, where d is the average non-leaf degree of vertices in V_3, and $\mathbf{e}$ is the base of the natural logarithm.*

PROOF. By Lemma 6, $\mathsf{bpr}(G) \leq k$ if and only if $\mathsf{bpr}(G \backslash \mathcal{S}_G) \leq k'$. By Lemma 7, $\mathsf{bpr}(G \backslash \mathcal{S}_G) \leq k'$ if and only if $\mathsf{bpr}(G_{\mathsf{sf}}) \leq k'$. Thus, $\mathsf{bpr}(G) \leq k$ if and only if $\mathsf{bpr}(G_{\mathsf{sf}}) \leq k'$. By Lemma 10, if $\Phi(G_{\mathsf{sf}}) > 2k'$ then $\mathsf{bpr}(G_{\mathsf{sf}}) > k'$; thus Step 3 is valid. By Lemma 12, the entire subkernel $\mathcal{K}$ is a biplanarizing set for G_{sf}. Therefore if $k' \geq |\mathcal{K}|$, G_{sf} has a biplanarizing set of size at most k', in which case, by Lemma 7, $\mathcal{K} \cup \mathcal{S}_G$ is a biplanarizing set for G of size at most k. Otherwise if $k' < |\mathcal{K}|$, then, by Lemma 12, to determine if $\mathsf{bpr}(G_{\mathsf{sf}}) \leq k'$ it suffices to test if G_{sf} has a biplanarizing set comprised of k' edges contained in $\mathcal{K}$. To do this, the algorithm simply searches through every subset T of k' edges in $\mathcal{K}$, and tests if T is a biplanarizing set for G_{sf}. If a set $T \subset \mathcal{K}$ is found to be biplanarizing set for G_{sf}, then by Lemma 7, $T \cup \mathcal{S}_G$ is a biplanarizing set for G. Thus Step 5 of algorithm Kernelization is valid.

We now analyze the running time of the algorithm. By Lemma 1(d), testing whether $T \subset \mathcal{K}$ is a biplanarizing set for G_{sf} can be carried out in $\mathcal{O}(|G_{\mathsf{sf}}|)$ time, which by Lemma 11 is $\mathcal{O}(k')$ time. Since $k' \leq k$, $\mathcal{O}(k') \in \mathcal{O}(k)$. If $k' < |\mathcal{K}|$ then by Lemma 13 $|\mathcal{K}| \leq 2k'\, d/(d-2)$. The number of k'-edge subsets of $\mathcal{K}$ is at most

$$\binom{2k'\, d/(d-2)}{k'} < \frac{(2k'\, d/(d-2))^{k'}}{k'!} < \frac{(2\mathbf{e}\, d/(d-2))^{k'}}{\sqrt{k'}},$$

by Stirling's formula. Thus Step 5 of the algorithm runs in $\mathcal{O}(\sqrt{k'} \cdot (2\mathbf{e}\, d/(d-2))^{k'})$ time. Since Steps 1–4, determining G_{sf} and $\mathcal{K}$, take $\mathcal{O}(|G|)$ time, the total running time of the algorithm is $\mathcal{O}(\sqrt{k'} \cdot (2\mathbf{e}\, d/(d-2))^{k'} + |G|)$. Since $k' \leq k$ this gives $\mathcal{O}(\sqrt{k} \cdot (2\mathbf{e}\, d/(d-2))^{k} + |G|)$ running time. □

Since $d \geq 3$, in the worst case the running time of algorithm Kernelization is $\mathcal{O}(\sqrt{k} \cdot (6\mathbf{e})^k + |G|) \in \mathcal{O}(\sqrt{k} \cdot 17^k + |G|)$. Notice that if the parameter k is big; that is, $k \geq |\mathcal{K}|$ the running time of the algorithm is $\mathcal{O}(|G|)$. That is also the case if $\Phi(G) = 0$.

4. Bounded Search Tree. A second approach for producing FPT algorithms is called the method of *bounded search trees* [4, Chapter 3.1]. Here one uses exhaustive search in a tree whose size is bounded by a function of the parameter. In this section we present an algorithm for the 2-LAYER PLANARIZATION problem based on a bounded search tree approach. Each node of the search tree corresponds to a subproblem (G', k'), where $G' \subseteq G$ and $k' \leq k$. At each node we find, if possible, a subgraph C that is a 2-claw or a small cycle. Since every biplanarizing set must contain at least one of the edges in C, our algorithm recursively solves $|E(C)|$ subproblems with one of the edges in C deleted from the graph in each subproblem. Recall that Lemma 2 provided a sufficient condition for the existence of such a set C.

Algorithm 2-Layer Bounded Search Tree

Input: graph $G_0 = (V_0, E_0)$
Parameter: non-negative integer k_0
Output: YES if and only if $\mathsf{bpr}(G_0) \leq k_0$

1. **if** $\Phi(G_0) > 2k_0$ **then** return NO.
2. **else if** $\Phi(G_0) = 0$ **then**
   if $k_0 \geq$ # component wreaths of G_0 **then** return YES.
   else return NO.
3. **else** ($\exists\, v \in V_0$ such that $\deg'_{G_0}(v) \geq 3$)
   if $k_0 > 0$ **then**
   (a) find a 2-claw, 3-cycle or 4-cycle C in G_0 containing v as described in Lemma 2
   (b) **for each** edge $xy \in C$ **do**
   if 2-Layer Bounded Search Tree($G_0 \backslash xy, k_0 - 1$) returns YES **then**
   return YES.
   return NO.

Note that the algorithm can be easily modified to return a biplanarizing set for YES instances of the 2-LAYER PLANARIZATION problem. We could solve 2-LAYER PLANARIZATION by running 2-Layer Bounded Search Tree (G, k). Instead, we apply 2-Layer Bounded Search Tree to the kernel of G so that the running time at each node of the search tree is $\mathcal{O}(k)$ rather than $\mathcal{O}(|G|)$.

The above description of our algorithm is recursive and we do not explicitly build a search tree. However, as is standard, associated with our recursive algorithm is a recursion tree [1] or search tree.

THEOREM 2. *Given a graph G and integer k, let G_{sf} be the kernel of $G \backslash \mathcal{S}_G$. The algorithm* 2-Layer Bounded Search Tree $(G_{\mathsf{sf}}, k - |\mathcal{S}_G|)$ *determines if* $\mathsf{bpr}(G) \leq k$ *in* $\mathcal{O}(k \cdot 6^k + |G|)$ *time.*

PROOF. We prove the correctness of the algorithm by induction on k_0 with the following induction hypothesis: "2-Layer Bounded Search Tree (G_0, k_0) returns YES if and only if $\mathsf{bpr}(G_0) \leq k_0$". The base case, $k_0 = 0$, (Step 1, Step 2, and the last line of Step 3) follows immediately from Lemmas 8 and 10. Assume $k_0 > 0$ and the induction hypothesis holds for $k_0 - 1$. By Lemma 10, if $\Phi(G_0) > 2k_0$ then $\mathsf{bpr}(G_0) > k_0$; thus Step 1 is valid. If

$\Phi(G_0) = 0$ (as in Step 2) then by Lemma 8, every connected component is a caterpillar or a wreath. Caterpillars and wreaths have biplanarization numbers of 0 and 1, respectively. Thus $\mathsf{bpr}(G_0)$ is the number of component wreaths of G_0, and hence Step 2 of the algorithm is valid.

Now assume $k_0 > 0$ and $\Phi(G_0) > 0$; that is, there exists a vertex $v \in V$ such that $\deg'_{G_0}(v) \geq 3$. By Lemma 2, G_0 contains a 2-claw or a 3- or 4-cycle C. Every biplanarizing set for G_0 must contain an edge in C. Thus $\mathsf{bpr}(G_0) \leq k_0$ if and only if there exists an edge $xy \in C$ such that $\mathsf{bpr}(G_0 \backslash xy) \leq k_0 - 1$. By induction, 2-Layer Bounded Search Tree $(G_0 \backslash xy, k_0 - 1)$ correctly determines if $\mathsf{bpr}(G_0 \backslash xy) \leq k_0 - 1$. Therefore the algorithm correctly determines if $\mathsf{bpr}(G_0) \leq k_0$. In particular, 2-Layer Bounded Search Tree $(G_{\mathsf{sf}}, k - |\mathcal{S}_G|)$ correctly determines if $\mathsf{bpr}(G_{\mathsf{sf}}) \leq k - |\mathcal{S}_G|$, which holds if and only if $\mathsf{bpr}(G) \leq k$ by Lemmas 6 and 7.

In each recursive call, k is reduced by one. Thus the height of the search tree is at most k. At each node of the search tree, there are $|E(C)|$ branches. Since $|E(C)| \leq 6$, the search tree has at most 6^k nodes. At any given node of the search tree, the algorithm takes $\mathcal{O}(|G_0|)$ time. Each G_0 is a subgraph of G_{sf}. Since the algorithm immediately terminates if $\Phi(G_0) > 2k_0$, $|G_{\mathsf{sf}}| \in \mathcal{O}(k)$ by Lemma 11. Hence the time taken at each node of the search tree is $\mathcal{O}(k)$. Therefore the running time of the algorithm is $\mathcal{O}(k \cdot 6^k + |G|)$. $\square$

We now compare the exponential terms of the time bounds for the Kernelization and 2-Layer Bounded Search Tree algorithms. The exponential term for Kernelization is $(2\mathbf{e}\,d/(d-2))^k$ while the exponential term for 2-Layer Bounded Search Tree is 6^k. In the worst case, when $d = 3$, the Kernelization term is approximately 17^k, which is considerably more than 6^k. However, for $d \geq 22$, $2\mathbf{e}\,d/(d-2) < 6$, and the Kernelization algorithm provides an exponential term with a smaller base than the 2-Layer Bounded Search Tree algorithm.

5. 1-Layer Planarization. We now consider the 1-LAYER PLANARIZATION problem defined in Section 2.2: Given a bipartite graph $G = (A, B; E)$ and permutation π of A, is $\mathsf{bpr}(G, \pi) \leq k$? If $\mathsf{bpr}(G, \pi) = 0$ we say that G is *π-biplanar*. The figures in this section show vertices in A as gray and vertices in B as white. We found it elusive to design an algorithm for this problem based on the kernelization method. However, we did find an algorithm based on the bounded search tree method. The following result characterizes π-biplanar graphs.

LEMMA 14. *A bipartite graph $G = (A, B; E)$ with a fixed permutation π of A is π-biplanar if and only if G is acyclic and the following condition holds*:

($\star$) *For every path (x, v, y) of G with $x, y \in A$, and for every vertex $u \in A$ between x and y in π, the only edge incident to u (if any) is uv.*

PROOF. We first show that if a bipartite graph $G = (A, B; E)$ with a fixed permutation π of A is π-biplanar, then G is acyclic and ($\star$) holds. The fact that every biplanar drawing is a forest of caterpillars implies the necessity for G to be acyclic. The necessity of condition ($\star$) is also easily verified by observing that if ($\star$) does not hold for some

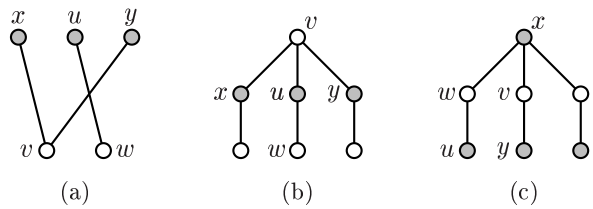


Fig. 8. Forbidden structures for π-biplanarity.

path (x, v, y) and vertex u, then u has a neighbour $w \neq v$. Regardless of the relative positions of w and v in the permutation of B, uw must cross xv or yv, as illustrated in Figure 8(a). This observation was also made by Mutzel and Weiskircher [20].

We now show that if a bipartite graph $G = (A, B; E)$ with a fixed permutation π of A is acyclic and $(\star)$ holds, then G is π-biplanar. Suppose G is acyclic and condition $(\star)$ holds. To construct a 2-layer drawing of G, we describe the permutation of B. Let $(1, 2, \ldots, |A|)$ be the left to right ordering of vertices in A defined by π. For each vertex $v \in B$, define $L(v) = \min\{i : iv \in E\}$; that is, $L(v)$ is the leftmost neighbour of v in the fixed permutation of A. We say a vertex $v \in B$ *belongs* to i if $L(v) = i$. Order the vertices $v \in B$ by increasing value of $L(v)$, breaking ties as follows. For each i, $1 \leq i \leq |A|$, break ties between the leaf neighbours of i arbitrarily. There is at most one non-leaf vertex belonging to i, as otherwise condition $(\star)$ is violated, as illustrated in Figure 9(a). Therefore, if i has a non-leaf neighbour, place all the leaf neighbours of i to the left of its non-leaf neighbour. This defines a 2-layer drawing.

Suppose there is a crossing between some edges iw and jv with $i, j \in A$ $(i < j)$ and $v, w \in B$. Then v is to the left of w in the permutation of B, and thus $L(v) \leq L(w) \leq i$. If $L(v) < i$ then condition $(\star)$ is violated for the path $(L(v), v, j)$ and vertex i, as illustrated in Figure 9(b). Otherwise, if $L(v) = i$ then vertex w cannot be a leaf as otherwise w would be to the left of v. If w is not a leaf, then let l be another neighbour of w. We know that $l \neq j$ as otherwise there would be a cycle in G. Then condition $(\star)$ is violated either for the path (i, w, l) and vertex j, or for the path $(L(v), v, j)$ and vertex l. Thus there is no crossing in the 2-layer drawing of G. □

LEMMA 15. *If $G = (A, B; E)$ is a bipartite graph and π is a permutation of A that satisfies condition $(\star)$, then all the cycles of G are 4-cycles and any two non-edge-disjoint cycles share exactly two edges. Moreover, the degree of any vertex in B that appears in a cycle is exactly 2.*

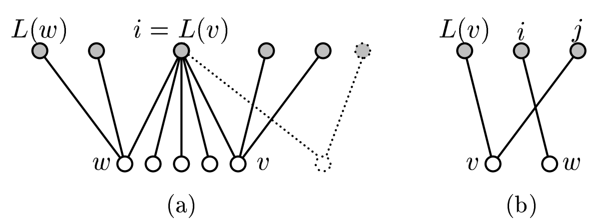


Fig. 9. Construction of the permutation of B.

PROOF. Suppose G contains a cycle C with $2k$ edges with $k \geq 3$. Let $C = (v_1, v_2, \dots, v_{2k}, v_{2k+1})$ with $v_1 = v_{2k+1} \in A$. Suppose without loss of generality that v_1 is to the left of v_3 in π. If v_5 is between v_1 and v_3 then condition $(\star)$ is not satisfied for the path (v_1, v_2, v_3) and vertex v_5. If v_5 is to the left of v_1, then condition $(\star)$ is not satisfied for the path (v_3, v_4, v_5) and vertex v_1. Thus v_5 is to the right of v_3. Continuing this argument, v_{2i+1} is to the right of v_{2i-1} for all i, $1 \leq i \leq k$. Thus $v_{2k+1} (= v_1)$ is to the right of v_1, which is a contradiction. Thus every cycle in G has four edges.

If G contains two distinct 4-cycles C_1 and C_2 that share exactly one edge vw, then $(C_1 \cup C_2) \backslash vw$ is a 6-cycle, which is a contradiction. No two distinct 4-cycles in a simple graph can share more than two edges. Thus, any two non-edge-disjoint cycles share exactly two edges.

Let (x, a, y, b) be a 4-cycle of G with x to the left of y in π. Suppose there is an edge aw in G with $x \neq w \neq y$. If w is between x and y in π, then condition $(\star)$ is not satisfied for the path (x, b, y) and vertex w. Otherwise, without loss of generality, say y is between x and w in π. Then condition $(\star)$ is not satisfied for the path (x, a, w) and vertex y. Thus there is no such edge aw. Hence, the degree of all vertices in B that appear in a cycle is exactly 2. □

Let $G = (A, B; E)$ be a bipartite graph with a fixed permutation of A that satisfies condition $(\star)$. Let $H = K_{2,p}$ be a complete bipartite subgraph of G with $H \cap A = \{x, y\}$, $H \cap B = \{v \in B\colon\ vx \in E, vy \in E, \deg_G(v) = 2\}$, and $|H \cap B| = p$. Then H is called a *p-diamond*. A 5-diamond is illustrated in Figure 10.

It follows from Lemma 15 that every cycle of G is in some p-diamond with $p \geq 2$. The next lemma gives the 1-layer biplanarization number $\mathsf{bpr}(\mathrm{G}, \pi)$ of G in terms of its p-diamonds, where G is a graph with permutation π satisfying condition $(\star)$.

LEMMA 16. *If $G = (A, B; E)$ is a bipartite graph and π is a permutation of A satisfying condition $(\star)$ then*

$$\mathsf{bpr}(\mathrm{G}, \pi) = \sum_{\text{maximal } p\text{-diamonds of } G} (p - 1).$$

PROOF. For each maximal p-diamond H of G, delete $p - 1$ of the edges incident to one of the vertices in $H \cap A$. The resulting graph is acyclic and satisfies condition $(\star)$, and thus, by Lemma 14, is π-biplanar. To remove all cycles from G requires the deletion of at least $p - 1$ edges from each maximal p-diamond since maximal p-diamonds are edge-disjoint. The result follows. □

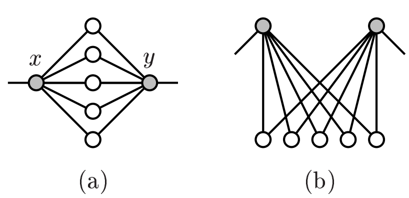


Fig. 10. (a) A 5-diamond. (b) A 2-layer drawing of a 5-diamond.

We now have the following bounded search tree algorithm for the 1-LAYER PLANARIZATION problem. Our recursive description of the algorithm assumes that a bipartite graph $G = (A, B; E)$ and permutation π of A are given.

Algorithm 1-Layer Bounded Search Tree

Input: graph $G_0 = (A_0, B_0, E_0)$; permutation π_0 of A_0
Parameter: non-negative integer k_0
Output: NO if $\text{bpr}(G_0, \pi_0) > k$; otherwise, YES.

1. **if** $(\star)$ fails for some path (x, v, y) and vertex u of G_0 **then**
   if $k_0 > 0$ **then**
      for each edge $e \in \{xv, yv, uw\}$ **do**
         if 1-Layer Bounded Search Tree $(G_0 \backslash e, \pi_0, k - 1)$ returns YES **then**
            return YES.
   return NO.
2. **else if** $k \geq \sum_{\text{maximal } p\text{-diamonds of } G_0} (p - 1)$ **then** return YES.
3. **else** return NO.

As in Section 4 we associate the search (recursion) tree with the recursive description of our algorithm.

THEOREM 3. *Given a bipartite graph $G = (A, B; E)$, a fixed permutation π of A, and integer k, the algorithm* 1-Layer Bounded Search Tree (G, π, k) *determines if* $\text{bpr}(G, \pi) \leq k$ *in* $\mathcal{O}(3^k \cdot |G|)$ *time.*

PROOF. The correctness of the algorithm follows from Lemmas 14 and 16. We now analyze the running time of the algorithm. First we order the adjacency lists of vertices in B according to π in $\mathcal{O}(|G|)$ time. For each vertex $v \in B$, let $L(v) = \min\{i : iv \in E\}$; and $R(v) = \max\{i : iv \in E\}$; that is, $L(v)$ and $R(v)$ are the leftmost and rightmost neighbours of v in the fixed permutation of A. We now check if condition $(\star)$ holds in $\mathcal{O}(|A|)$ time as follows. For every non-leaf vertex $v \in B$ we test if $(\star)$ is satisfied for a 2-path $L(v), v, R(v)$ and all the vertices of A in the open interval $(L(v), R(v))$. This procedure stops when a 2-path and a vertex are found that violate condition $(\star)$ or when all non-leaf vertices $v \in B$ are considered. The procedure runs in $\mathcal{O}(|A|)$ time since it stops the first time it encounters two intervals $(L(v), R(v))$ and $(L(w), R(w))$ for $v \neq w$ with non-empty intersection; otherwise all the intervals $(L(v), R(v))$ and $(L(w), R(w))$ for $v \neq w$ have empty intersection. To count the number and size of the diamonds in G takes $\mathcal{O}(|G|)$ time. Thus, the algorithm takes $\mathcal{O}(|G|)$ time at each node of the search tree. Since each node of the search tree has three children, and the height of the tree is at most k, the algorithm runs in $\mathcal{O}(3^k \cdot |G|)$ time. □

6. Approximations. It is simple to verify that Lemmas 10 and 13 imply that there is a linear-time $(2d/(d - 2))$-approximation algorithm for the 2-LAYER PLANARIZATION problem where $d \geq 3$ is the average non-leaf degree of vertices in V_3. However, it is easy to do better. The following observation seems to have gone unnoticed in the literature.

LEMMA 17. *There is a linear-time* 2*-approximation algorithm for the optimization version of the* 2-LAYER PLANARIZATION *problem.*

PROOF. Let $G = (V, E)$ be a connected graph with n vertices and m edges. Let $r = m-(n-1)$. Then $\mathsf{bpr}(G) \geq r$, since a biplanar graph is a forest of caterpillars with at most $n-1$ edges. Consider the following algorithm. Let S be a set of edges of G such that $G\backslash S$ is a spanning tree T of G. Then $|S| = r$. Apply the linear-time algorithm of Shahrokhi et al. [22] to obtain a minimum set of edges $S_T \subseteq E(T)$ such that $T\backslash S_T$ is biplanar. Then $S \cup S_T$ is a biplanarizing set of G with $r + |S_T| = r + \mathsf{bpr}(T) \leq r + \mathsf{bpr}(G) \leq 2\,\mathsf{bpr}(G)$ edges. Thus this algorithm is a 2-approximation, and it clearly runs in $\mathcal{O}(n+m)$ time. □

LEMMA 18. *There is a polynomial-time* 3*-approximation algorithm for the optimization version of the* 1-LAYER PLANARIZATION *problem.*

PROOF. Consider an instance (G, π) of the 1-LAYER PLANARIZATION problem with a bipartite graph $G = (A, B; E)$ and a fixed permutation π of A. Motivated by Lemma 14, we define the following forbidden structure. A path (x, v, y) with $x, y \in A$ and an edge uw with $w \neq v$, $u \in A$, and $x < u < y$ is called a *forbidden structure* in (G, π), as illustrated in Figure 8.

Consider the following algorithm. While condition $(\star)$ in Lemma 14 is violated by some forbidden structure $(x, v, y), uw$ delete all three edges xv, vy, and uw. Let S denote the set of deleted edges. The instance $(G\backslash S, \pi)$ satisfies the constraints imposed by Lemma 16. Therefore, $(G\backslash S, \pi)$ can be solved optimally.

The number of edge-disjoint forbidden structures in the instance (G, π) is at least $|S|/3$. By Lemma 14, at least one of the edges from each of the forbidden structures has to be deleted. Thus any optimal solution contains a set of edges $R \subset S$ such that R contains exactly one edge from each of the $|S|/3$ edge-disjoint forbidden structures. Therefore, $\mathsf{bpr}(G, \pi) = |S|/3 + \mathsf{bpr}(G\backslash R, \pi)$. The number of edges deleted from G by the algorithm is $|S| + \mathsf{bpr}(G\backslash S, \pi)$. Since $G\backslash S$ is a spanning subgraph of $G\backslash R$, $\mathsf{bpr}(G\backslash R, \pi) \geq \mathsf{bpr}(G\backslash S, \pi)$. Therefore $|S| + \mathsf{bpr}(G\backslash S, \pi) \leq 3(|S|/3 + \mathsf{bpr}(G\backslash R, \pi)) = 3\,\mathsf{bpr}(G, \pi)$ and thus the algorithm is a 3-approximation. A running time analysis, similar to the one presented in the proof of Theorem 3, reveals that the algorithm can be implemented to run in $\mathcal{O}(|A||B|^2)$ time. □

7. Conclusion. In this paper we have presented two methods for producing FPT algorithms in the context of 2- and 1-layer planarization. In particular, for fixed k, we have linear-time algorithms to determine if $\mathsf{bpr}(G) \leq k$ and $\mathsf{bpr}(G, \pi) \leq k$. For small values of k our algorithms provide a feasible method for the solution of these $\mathcal{NP}$-complete problems.

The results presented in this paper suggest the following open problems.

OPEN PROBLEM 1. Is there a c-approximation algorithm for the optimization version of the 2-LAYER PLANARIZATION problem with $c < 2$? Is there an FPT algorithm for the 2-LAYER PLANARIZATION problem parameterized by the number of edge deletions k, with the exponential part of the running time less than 6^k?

OPEN PROBLEM 2. Is there a c-approximation algorithm for the 1-LAYER PLANARIZATION problem with $c < 3$? Is there an FPT algorithm for the optimization version of the 1-LAYER PLANARIZATION problem parameterized by the number of edge deletions k, with the exponential part of the running time less than 3^k?

In this paper we have not emphasized implementation details that may in practice reduce the running time. For example, the algorithm 2-Layer Bounded Search Tree could be modified to recompute the kernel and the subkernel at each (or selected) node(s) of the search tree. Furthermore, the recursive calls should be made only on those edges of 2-claws, 3- or 4-cycles that are in the subkernel.

Notice that the exact values for $\mathsf{bpr}(G)$ or $\mathsf{bpr}(G, \pi)$ can be determined by running our algorithms for each $k = 0, 1, 2, \ldots$ until the first value of k is reached that gives the "YES" instance. Clearly that value is equal to $\mathsf{bpr}(G)$ (or $\mathsf{bpr}(G, \pi)$). Therefore, our algorithms can be used to compute optimal solutions for 1- and 2-LAYER PLANARIZATION in time $\mathcal{O}(3^{\mathsf{bpr}(G,\pi)} \cdot |G|)$ and $\mathcal{O}(\mathsf{bpr}(G) \cdot 6^{\mathsf{bpr}(G)} + |G|)$, respectively. Initial experiments [23] comparing these algorithms with the other known method for optimal 1- and 2-layer planarization, namely, integer linear programming [18], [20] suggest that our approach may be worthwhile in practice.

Acknowledgments. This research was initiated at the International Workshop on Fixed Parameter Tractability in Graph Drawing, Bellairs Research Institute of McGill University, Holetown, Barbados, Feb. 9–16, 2001, organized by Sue Whitesides. We thank the institute for making this collaboration possible.

References

[1] T. Cormen, C. Leiserson, and R. Rivest, *Introduction to Algorithms*, MIT Press, Cambridge, MA, 1990.

[2] C. Demetrescu and I. Finocchi, Breaking cycles for minimizing crossings, *ACM J. Exp. Algorithmics*, **6**(2):1–39, 2001. Also appeared as an extended abstract, Break the "right" cycles and get the "best" drawing, in B. Moret and A. V. Goldberg, eds., *Proc. 2nd Workshop on Algorithm Engineering and Experiments* (*ALENEX '00*), San Francisco, CA, pp. 171–182, 2000.

[3] G. Di Battista, P. Eades, R. Tamassia, and I. G. Tollis, *Graph Drawing: Algorithms for the Visualization of Graphs*, Prentice-Hall, Englewood Cliffs, NJ, 1999.

[4] R. G. Downey and M. R. Fellows, *Parameterized Complexity*, Springer, New York, 1999.

[5] V. Dujmović, M. Fellows, M. Hallett, M. Kitching, G. Liotta, C. McCartin, N. Nishimura, P. Ragde, F. Rosemand, M. Suderman, S. Whitesides, and D. R. Wood, On the parameterized complexity of layered graph drawing, in F. Meyer auf der Heide, ed., *Proc. 5th Annual European Symposium on Algorithms* (*ESA '01*), vol. 2161 of Lecture Notes in Computer Science, pp. 488–499, Springer, Berlin, 2001.

[6] V. Dujmović, H. Fernau, and M. Kaufmann, Fixed parameter algorithms for 1-sided crossing minimization revisited, in [16], pp. 332–344.

[7] V. Dujmović and S. Whitesides, An efficient fixed parameter tractable algorithm for 1-sided crossing minimization, *Algorithmica*, **40**(1):15–31, 2004.

[8] P. Eades, B. D. McKay, and N. C. Wormald, On an edge crossing problem, in *Proc. 9th Australian Computer Science Conference*, pp. 327–334, Australian National University, Canberra, 1986.

[9] P. Eades and S. Whitesides, Drawing graphs in two layers, *Theoret. Comput. Sci.*, **131**(2):361–374, 1994.

[10] P. Eades and N. C. Wormald, Edge crossings in drawings of bipartite graphs, *Algorithmica*, **11**(4):379–403, 1994.

[11] M. R. Garey and D. S. Johnson, Crossing number is NP-complete, *SIAM J. Algebraic Discrete Methods*, **4**(3):312–316, 1983.

[12] F. Harary and A. Schwenk, A new crossing number for bipartite graphs, *Util. Math.*, **1**:203–209, 1972.

[13] M. Jünger and P. Mutzel, 2-Layer straightline crossing minimization: performance of exact and heuristic algorithms, *J. Graph Algorithms Appl.*, **1**(1):1–25, 1997.

[14] M. Kaufmann and D. Wagner, eds., *Drawing Graphs*: *Methods and Models*, vol. 2025 of Lecture Notes in Computer Science, Springer, Berlin, 2001.

[15] T. Lengauer, *Combinatorial Algorithms for Integrated Circuit Layout*, Wiley, New York, 1990.

[16] G. Liotta, ed., *Proc*. 11*th International Symposium on Graph Drawing* (*GD* '03), vol. 2912 of Lecture Notes in Computer Science, Springer, Berlin, 2004.

[17] X. Muñoz, U. Unger, and I. Vrťo, One sided crossing minimization is NP-hard for sparse graphs, in P. Mutzel, M. Jünger, and S. Leipert, eds., *Proc*. 9*th International Symposium on Graph Drawing* (*GD* '01), vol. 2265 of Lecture Notes in Computer Science, pp. 115–123, Springer, Berlin, 2002.

[18] P. Mutzel, An alternative method to crossing minimization on hierarchical graphs, *SIAM J. Optim.*, **11**(4):1065–1080, 2001a.

[19] P. Mutzel, Optimization in leveled graphs, in C. A. Floudas and P. M. Pardalos, eds., *Encyclopedia of Optimization*, vol. 4, pp. 189–196, Kluwer, Dordrecht, 2001b.

[20] P. Mutzel and R. Weiskircher, Two-layer planarization in graph drawing. In K. Y. Chwa and O. H. Ibarra, eds., *Proc*. 9*th Internat*. *Symposium on Algorithms and Computation* (*ISAAC* '98), vol. 1533, pp. 69–78, Springer, Berlin, 1998.

[21] H. Nagamochi, An improved approximation to the one-sided bilayer drawing, in [16], pp. 406–418.

[22] F. Shahrokhi, O. Sýkora, L. A. Székely, and I. Vrťo, On bipartite drawings and the linear arrangement problem, *SIAM J. Comput.*, **30**(6):1773–1789, 2001.

[23] M. Suderman and S. Whitesides, Experiments with the fixed-parameter approach for two-layer planarization, in [16], pp. 345–356.

[24] K. Sugiyama, S. Tagawa, and M. Toda, Methods for visual understanding of hierarchical system structures, *IEEE Trans*. *Systems Man Cybernet*., **11**(2):109–125, 1981.

[25] N. Tomii, Y. Kambayashi, and S. Yajima, On planarization algorithms of 2-level graphs, *IECEJ*, **EC77-38**:1–12, 1977.

[26] V. Valls, R. Marti, and P. Lino, A branch and bound algorithm for minimizing the number of crossing arcs in bipartite graphs, *J. Oper. Res.*, **90**:303–319, 1996.

[27] M. S. Waterman and J. R. Griggs, Interval graphs and maps of DNA, *Bull. Math. Biol.*, **48**(2):189–195, 1986.

[28] A. Yamaguchi and A. Sugimoto, An approximation algorithm for the two-layered graph drawing problem, in T. Asano, H. Imai, D. T. Lee, S. Nakano, and T. Tokuyama, eds., *Proc*. *Computing and Combinatorics* (*COCOON* '99), vol. 1627 of *Lecture Notes in Computer Science*, pp. 81–91, Springer, Berlin, 1999.