

Two-Layer Planarization: Improving on Parameterized Algorithmics

Henning Fernau^{1,2}

¹ Universität Tübingen, WSI für Informatik, Sand 13,
72076 Tübingen, Germany, fernau@informatik.uni-tuebingen.de

² The University of Newcastle, School of Electr. Eng. and Computer Science,
University Drive, Callaghan, NSW 2308, Australia

Abstract. A bipartite graph is *biplanar* if the vertices can be placed on two parallel lines in the plane such that there are no edge crossings when edges are drawn as straight-line segments. We study two problems:

- 2-LAYER PLANARIZATION: can k edges be deleted from a given graph G so that the remaining graph is biplanar?
- 1-LAYER PLANARIZATION: same question, but the order of the vertices on one layer is fixed.

Improving on earlier works of Dujmović *et al.* [4], we solve the 2-LAYER PLANARIZATION problem in $\mathcal{O}(k^2 \cdot 5.1926^k + |G|)$ time and the 1-LAYER PLANARIZATION problem in $\mathcal{O}(k^3 \cdot 2.5616^k + |G|^2)$ time. Moreover, we derive a small problem kernel for 1-LAYER PLANARIZATION.

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 another line containing the vertices in B , and the edges are drawn as straight line-segments. Such drawings have various applications, see [4]. 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*. It has been argued that 2-layer drawings in which all the crossings occur in a few edges are more readable than drawings with fewer total crossings [7]—which gives the CROSSING MINIMIZATION problem(s) [5].

This naturally leads to the definition of the 2-LAYER PLANARIZATION problem (2-LP): given a graph G (not necessarily bipartite), and an integer k called *parameter*, can G be made biplanar by deleting at most k edges? Two-layer drawings are of fundamental importance in the “Sugiyama” approach to multi-layer graph drawing [10]. This method involves (repeatedly) solving the 1-LAYER PLANARIZATION problem (1-LP).

Fixed parameter tractability. We develop improved algorithms for 2-LP and for 1-LP that are exponential in the parameter k . This has the following justification: when the maximum number k of allowed edge deletions is small, an algorithm for 1- or 2-LP whose running time is exponential in k but polynomial in the size of the graph may be useful. We expect the parameter k to be small in

practice. Instances of the 1- and 2-LP for dense graphs are of little interest from a practical point of view, as the resulting drawing will be unreadable anyway.

This analysis hence fits into the framework of parameterized algorithmics. 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 (independent of k).

Our results. In this paper, we apply so-called kernelization and search tree methods to obtain algorithms for the 1- and 2-LP problems, this way improving earlier results [4] with exponential bases of 3 and 6, respectively. This leads to an $\mathcal{O}(k^2 \cdot 5.1926^k + |G|)$ time algorithm for 2-LP in a graph G . We present a similar second algorithm to solve the 1-LP problem in $\mathcal{O}(k^3 \cdot 2.5616^k + |G|^2)$ time. To this end, we draw connections to HITTING SET problems. The top-down analysis technique presented in [6] is applied to obtain the claimed running times in the analysis of the search tree algorithms. Detailed proofs can be found in <http://eccc.uni-trier.de/eccc-reports/2004/TR04-078/index.html>.

2 Preliminaries

In this section we introduce notation, recall a characterization of biplanar graphs and formalize the problem statements. All the mentioned results are from [4].

In this paper each graph $G = (V, E)$ is simple and undirected. The subgraph of G induced by a subset E' of edges is denoted by $G[E']$. A vertex with degree one 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 *non-leaf degree* of a vertex v in graph G is the number of non-leaf edges at v in G , and is denoted by $\deg'_G(v)$.

A graph is a *caterpillar* if deleting all the leaves produces a (possibly empty) path. This path is the *spine* of the caterpillar. A *2-claw* is a graph consisting of one degree-3 vertex, the *center*, which is adjacent to three degree-2 vertices, each of which is adjacent to the center and one leaf. A graph consisting of a cycle and possibly some leaf-edges attached to the cycle is a *wreath*. Notice that a connected graph that does not have a vertex v with $\deg'(v) \geq 3$ is either a caterpillar or a wreath.

To prove their kernelization result for 2-LP, Dujmović *et al.* introduced the following potential function. For a graph $G = (V, E)$, define

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

Lemma 1. $\Phi(G) = 0$ if and only if G is a collection of caterpillars and wreaths.

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

Lemma 2. *Let G be a graph. The following assertions are equivalent: (a) G is biplanar. (b) G is a forest of caterpillars. (c) G is acyclic and contains no 2-claw as a subgraph. (d) G is acyclic and $\Phi(G) = 0$ (with Lemma 1).*

Lemma 2 implies that any biplanarization algorithm must destroy all cycles and 2-claws. The next lemma gives a condition for this situation.

Lemma 3. *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 .*

A set T of edges of a graph G is called a *biplanarizing set* if $G \setminus T$ is biplanar. The *bipartite planarization number* of a graph G , denoted by $\text{bpr}(G)$, is the size of a minimum biplanarizing set for G . The 2-LP problem is: given a graph G and an integer k , is $\text{bpr}(G) \leq k$? For a given bipartite graph $G = (A, B; E)$ and permutation π of A , the *1-layer biplanarization number* of G and π , denoted $\text{bpr}(G, \pi)$, 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-LP problem asks if $\text{bpr}(G, \pi) \leq k$.

Lemma 4. *For graphs G with $\Phi(G) = 0$, a minimum biplanarizing set of G consists of one cycle edge from each component wreath.*

Lemma 5. *For every graph G , $\text{bpr}(G) \geq \frac{1}{2}\Phi(G)$.*

3 2-Layer Planarization: Bounded search tree

The basic approaches for producing *FPT* algorithms are *kernelization* and *bounded search trees* [2]. Based on the preceding lemmas, Dujmović *et al.* showed:

Theorem 1. *Given a graph G and integer k , there is an algorithm that determines if $\text{bpr}(G) \leq k$ in $\mathcal{O}(k \cdot 6^k + |G|)$ time.*

That algorithm consists of two parts: a kernelization algorithm and a subsequent search tree algorithm *2-Layer Bounded Search Tree*. The latter algorithm basically looks for a vertex v with $\deg'(v) \geq 3$: if found at most 6 recursive branches are triggered to destroy the forbidden structures described in Lemma 3. After branching, a graph G with $\Phi(G) = 0$ remains, solvable with Lemma 4.

Can we further improve on the running time of the search tree algorithm? Firstly, observe that whenever $\deg'_{G'}(v) \geq \ell$ for any G' obtained from G by edge deletion, then already $\deg'_G(v) \geq \ell$. This means that we can modify the sketched algorithm by collecting *all* vertices of non-leaf degree at least three and, based on this, all *forbidden structures* F , i.e., 2-claws, 3-cycles, or 4-cycles, according to Lemma 3 (which then might interact). For reasons of improved algorithm analysis, we also regard 5-cycles as forbidden structures. By re-interpreting the edges of G as the vertices of a hypergraph $H = (E, F)$, where the hyperedges correspond to the forbidden structures, a 2-LP instance (G, k) is translated into an instance (H, k) of 6-HITTING SET (6-HS).

If we delete all those edges in G that are elements in a hitting set as delivered by a 6-HS algorithm, we arrive at a graph G' which satisfies $\deg'_{G'}(v) < 3$ for all vertices v . Hence, $\Phi(G') = 0$, and Lemma 4 applies.

Unfortunately, we cannot simply take some 6-HS algorithms as described in [8]. Why? The problem is that there may be minimal hitting sets C which are “skipped” due to clever branching, since there exists another minimal solution C' with $|C| \geq |C'|$. However, if we translate back to the original 2-LP instance, we still have to resolve the wreath components, and it might be that we “skipped” the only solution that already upon solving the 6-HS instance was incidentally also destroying enough wreaths. To be more specific, the analysis in [8] is based on the validity of the vertex domination rule: A vertex x is *dominated* by a vertex y if, whenever x belongs to some hyperedge e , then y belongs to e , as well. Then, delete all occurrences of x , since taking y into the hitting set (instead of x) is never worse. As explained, the subsequent wreath analysis destroys the applicability of that rule.

If we insist on enumerating all minimal hitting sets no larger than the given k , this problem can be circumvented, since we can do the wreath component analysis in the leaves of the search tree, but it would gain nothing in terms of time complexity, since examples of hypergraphs having 6^k minimal hitting sets of size at most k can be easily found: just consider k disjoint hyperedges, each of size 6, see [1].

However, our experience with analyzing HITTING SET problems by the help of the so-called top-down approach as detailed in [6] gives the basic ideas for the claimed improvements. The main purpose of the vertex domination rule is that it guarantees the existence of vertices of “sufficiently” high degree in the HITTING SET instance. Our aim is now to provide a more problem-specific analysis which maintains exactly that property. To avoid confusion, in the following, we will stay within the 2-LP formulation and will not translate back and forth between the 2-LP instance and the corresponding 6-HS instance.

In order to maintain the structure of the original graph for the final wreath analysis, we will mark edges that won't be put into a solution during the recursive branching process as *virtual*, but we won't delete them. Hence, along the course of the algorithm we present, there will be built a set M of edges that are marked virtual. A *forbidden structure* f is a set of edges of the graph instance $G = (V, E)$ such that

- f describes a cycle of length up to five or a 2-claw, and
- $f \setminus M \neq \emptyset$.

$c(f) = f \setminus M$ is the *core* of f ; $s(f) = |c(f)|$ is the *size* of f .

We will use the following reduction rules:

1. structure domination: A forbidden structure f is *dominated* by another structure f' if $c(f') \subset c(f)$. Then, mark f as dominated.
2. small structures: If $s(f) = 1$, put the only non-virtual edge into the solution that is constructed.
- 3a isolates: If e is an edge of degree zero, then mark e virtual.

The number of non-dominated forbidden structures to which a specific edge e belongs is also called the *degree* of e . Can we also handle edges of degree one (to a certain extent) by reduction rules? We will discuss this point later on.

Let $C = \{c, w_1, w_2, w_3, x_1, x_2, x_3\}$ be a 2-claw centered at c , such that w_i is neighbored (at least) to c and x_i for $i = 1, 2, 3$. We will call $F_i = \{cw_i, w_ix_i\}$ also a *finger* of C , so that the forbidden structure f_C corresponding to C is partitioned into three disjoint fingers. A 2-claw where one or more edges are virtual is called *injured*. Clearly, in an injured 2-claw with five edges, only one of the fingers actually got injured and two fingers are still *pretty*. In an injured 2-claw with four edges, we still have at least one pretty finger left over.

The second ingredient in the approach to hitting set problems described in [6] are so-called *heuristic priorities*. More specifically, we use the following rules to select forbidden structures and edges to branch at in case of multiple possibilities:

1. Select a forbidden structure f of smallest size that if possible corresponds to a short cycle.
2. If $s(f) \leq 5$ and if there is another forbidden structure f' with $c(f) \cap c(f') \neq \emptyset$ and $s(f') \leq 5$, modify $f := c(f) \cap c(f')$.
3. Select an edge e of maximal degree within f , if possible incident to the center of the 2-claw f , such that e belongs to a pretty finger.

In the following analysis, assume that we have already branched on all cycles up to length five (see the first heuristic priority). Then, we can apply the following reduction rule for (injured) 2-claws:

- 3b (injured) 2-claws: If e is an edge of degree one in a forbidden structure of size four, five or six corresponding to an (injured) 2-claw, and if e is incident to the center of the corresponding 2-claw, then mark e virtual.

This allows us to state the whole procedure in Alg. 1, where *branch at e* means the following:

```

if TLP( $G - e, k - 1, M, D_F$ ) then
    return YES
else if  $G[M \cup \{e\}]$  is acyclic then
    return TLP( $G, k, M \cup \{e\}, D_F$ )
end if

```

To prove the soundness of rule 3b., we have to show that we will never miss out cycles this way. We therefore show the following assertions:

Proposition 1. *At most one edge per finger will turn virtual due to rule 3b.*

Proof. 3b. obviously only affects *one* 2-claw at a time, since only edges of size one are turned virtual. Per 2-claw, the rule triggers at most once per finger. \square

Proposition 2. *Cycles of length at least six that only consists of virtual edges can never be created by running Alg. 1.*

To prove Proposition 2, the following observation is crucial.

Property 1. Let $F = \{xy, yz\}$ be one pretty finger of a non-dominated (injured) 2-claw C with center x such that xy occurs only in one forbidden structure, i.e., C . Then, y has degree two.

Algorithm 1 A search tree algorithm for 2-LP, called TLP

Require: a graph $G = (V, E)$, a positive integer k , a set of virtual edges M , a list of dominated forbidden structures D_F

Ensure: YES if there is a biplanarization set $B \subseteq E$, $|B| \leq k$ (and it will implicitly produce such a small biplanarization set then) or NO if no such set exists.

Exhaustively apply the reduction rules 1., 2., and 3a.; the resulting instance is also called (G, k, M, D_F) .

if $\Phi(G[E \setminus M]) > 2k$ **then**

 return NO {Lemma 5}

else if $\Phi(G[E \setminus M]) = 0$ **then**

if $k \geq \#$ component wreaths of $G[E \setminus M]$ **then**

 return YES {Lemma 4}

else

 return NO

end if

else

$\{\exists v \in V$ such that $\deg'_{G[E \setminus M]}(v) \geq 3\}$

if possible then

 Find a non-dominated cycle C of length at most 5

 Select an edge $e \in C$ and branch at e

else

 Exhaustively apply all reduction rules

 Select 2-claw C and edge $e \in C$ according to heuristic priorities; branch at e

end if

end if

Proof. If the conclusion were false, there must be an edge yv in the given 2-LP instance. Hence, there is an (injured) 2-claw C' with center x which is like C , only having z replaced by v . This contradicts that xy has degree one, since xy participates both in C and in C' . \square

Now to the time analysis of Alg. 1, following the ideas explained in [6] for 3-HS. $T(k)$ denotes the number of leaves in a worst-case search tree for Alg. 1, which incidentally also is the worst-case for the number of returned solutions. More distinctly, let $T^\ell(k)$ denote the situation of a search tree assuming that at least ℓ forbidden structures in the given instance (with parameter k) have size five. Of course, $T(k) \leq T^0(k)$. We analyze the recurrences for T^0 , T^1 and T^2 .

Lemma 6. $T^0(k) \leq T^0(k-1) + T^2(k)$.

Proof. Due to the reduction rule 3b., the 2-LP instance G contains an edge e of degree 2 in a forbidden structure f of size 6, since f represents a 2-claw. Hence, there is another 2-claw corresponding to a forbidden structure f' with $e \in f \cap f'$. One branch is that e is put into the biplanarization set. The size of the corresponding subtree can be estimated by $T^0(k-1)$. If e is not put into the biplanarization set, then e is marked virtual and hence at least two forbidden

structures of size five are created: $f \setminus \{e\}$ and $f' \setminus \{e\}$. Therefore, the size of that subtree is upperbounded by $T^2(k)$. \square

Some more involved analysis of the T^1 - and T^2 -branches as well as some algebra for solving the recursions, shows:

Lemma 7. $T^1(k) \leq 2T^0(k-1) + 2T^1(k-1) + T^2(k-1)$.

Lemma 8. $T^2(k) \leq \max\{2T^1(k-1)+3T^2(k-1), T^0(k-1)+16T^0(k-2), 2T^0(k-1) + 9T^0(k-2), 3T^0(k-1) + 4T^0(k-2), 4T^0(k-1) + T^0(k-2)\}$.

Theorem 2. *Given a graph G and an integer k , Alg. 1 determines if $\text{bpr}(G) \leq k$ in $\mathcal{O}(k^2 \cdot 5.1926^k + |G|)$ time, when applied to the problem kernel derived in [4].*

To prove these results, the following lemma is important, which is also interesting from a structural point of view on its own account; this also explains why we considered 5-cycles as forbidden structures.

Lemma 9. *In a graph without cycles up to length five, each 2-claw is vertex-induced.*

4 1-Layer Planarization: Kernelization algorithm

The next two results from [4] give important properties for π -bipolar graphs.

Lemma 10. *A bipartite graph $G = (A, B; E)$ with a fixed permutation π of A is π -bipolar if and only if G is acyclic and the following condition holds.*

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 . (★)

Let $G = (A, B; E)$ be a bipartite graph with a fixed permutation of A that satisfies condition (★). Let $H = K_{2,p}$ be a complete bipartite subgraph of G with $H \cap A = \{x, y\}$, and $H \cap B = \{v \in B : vx \in E, vy \in E, \deg_G(v) = 2\}$, and $|H \cap B| = p$. Then H is called a p -diamond. Every cycle of G is in some p -diamond with $p \geq 2$.

Lemma 11. *If $G = (A, B; E)$ is a bipartite graph and π is a permutation of A satisfying condition (★) then $\text{bpr}(G, \pi) = \sum_{\text{maximal } p\text{-diamonds of } G} (p-1)$.*

We are now going to derive a kernelization algorithm for 1-LP. Let us say that an edge e of a bipartite graph G potentially violates condition (★) if, using the notation of condition (★), $e = e_i$ for $i = 1, 2, 3$, where $e_1 = xv$ or $e_2 = vy$ or $e_3 = uz$ for some u strictly between x and y in π such that $z \neq v$. We will also say that e_1, e_2, e_3 (together) violate condition (★).

According to Lemma 10 (as well as the proof of Lemma 11 for the last two rules), the following reduction rules are sound, given an instance $(G = (A, B; E), \pi, k)$ of 1-LP. Analogues to the first three rules are well-known from HITTING SET problems, see [6, 8].

1L-RR-edge: If $e \in E$ does not participate in any cycle and does not potentially violate condition (\star) , then remove e from the instance (keeping the same parameter k).

1L-RR-isolate: If $v \in A \cup B$ has degree zero, then remove v from the instance and modify π appropriately (keeping the same parameter k).

1L-RR-large: If $e \in E$ participates in more than k^2 situations that potentially violate condition (\star) , then put e into the biplanarization set and modify the instance appropriately (also decreasing the parameter).

Let $E_\star \subseteq E$ be all edges that potentially violate condition (\star) . Let $E_\circ \subseteq E$ be all edges that participate in cycles. Let G_{4c} be generated from those edges from $E_\star \setminus E_\circ$ that participate in 4-cycles. By construction, G_{4c} satisfies (\star) . Lemma 11 shows that the next reduction rule can be applied in polynomial time:

1L-RR-4C: If $\text{bpr}(G_{4c}, \pi) > k$, then NO.

Lemma 12. *Let $G = (A, B; E)$ be a bipartite graph and let π be a permutation of A . Let $v \in B$. Then, there is at most one edge e incident to v that does not potentially violate condition (\star) and participates in cycles of length > 4 .*

Theorem 3. *Let $G = (A, B; E)$ be a bipartite graph, π be a permutation of A and $k \geq 0$. Assume that none of the reduction rules applies to the 1-LP instance (G, π, k) . Then, $|E| \leq k^3$. The kernel can be found in time $\mathcal{O}(|G|^2)$.^{***}*

Proof. Now consider E_\star as vertex set V' of a hypergraph $G' = (V', E')$ and put $\{e_1, e_2, e_3\}$ into E' iff e_1, e_2, e_3 together violate condition (\star) . A subset of edges from E whose removal converts $(A, B; E)$ into a bipartite graph which satisfies condition (\star) is in obvious one-to-one correspondence with a hitting set of the hypergraph G' . Niedermeier and Rossmanith have shown [8, Proposition 1] a cubic kernel for 3-HITTING SET, so that at most k^3 edges are in E_\star (else NO). Their reduction rules correspond to our rules 1L-RR-edge and 1L-RR-large.

If $e = xy \in E_\circ \setminus E_\star$ with $y \in B$ does not belong to a 4-cycle, then Lemma 12 shows that there is no other edge $zy \in E_\circ \setminus E_\star$. But since $xy \in E_\circ$, there must be some “continuing edge” zy on the long circle xy belongs to, so that $zy \in E_\star$ follows. We can take zy as a *witness* for xy . By Lemma 12, zy can witness for at most one edge from $E_\circ \setminus E_\star$ incident to y and not participating in a 4-cycle.

This allows us to partition E_\circ into three disjoint subsets: (a) $E_\circ \cap E_\star$, (b) $E_{4c} = \{e \in E_\circ \setminus E_\star \mid e \text{ participates in a 4-cycle}\}$: there can be at most $4k$ such edges according to 1L-RR-4C and Lemma 11, and (c) $E_\circ \setminus E_{4c}$: according to our preceding reasoning, there are at most $|E_\star|$ many of these edges. \square

5 1-Layer Planarization: Bounded search tree

Theorem 4. *(Dujmović et al. [4]) Given a bipartite graph $G = (A, B; E)$, a fixed permutation π of A , and integer k , there is an algorithm that determines if $\text{bpr}(G, \pi) \leq k$ in $\mathcal{O}(3^k \cdot |G|)$ time.*

^{***} More recently, a quadratic kernel for 3-HITTING SET was derived [9] based on results by Nemhauser and Trotter. Translating the corresponding reduction rules shows that $|E_\star|$ and hence $|E|$ is in fact upperbounded by $\mathcal{O}(k^2)$.

Can we further improve on this algorithm? Firstly, it is clear that we can combine the search tree algorithm with the kernelization algorithm described above. But furthermore, observe that the search tree algorithm basically branches on all members of E_* , trying to destroy the corresponding triples of edges violating condition (\star) . This means that we again take ideas stemming from solutions of the naturally corresponding instance of 3-HITTING SET. Unfortunately again, we cannot simply “copy” the currently best search tree algorithm for 3-HITTING SET [6, 8], running in time $\mathcal{O}(k \cdot 2.179^k + |G|)$, since destroying triples of edges violating condition (\star) might incidentally also destroy more or less of the 4-cycles. As explained in the 2-LP case, the problem is again the vertex domination rule. In order to gain anything against the previously sketched algorithm 1-Layer Bounded Search Tree, we must somehow at least *avoid branching on vertices of degree one contained in hyperedges of size three*.

Firstly, we can prove a lemma that shows that, whenever we have branched on all hyperedges of size three in the 3-HITTING SET instance (that correspond to situations violating condition (\star) in the original 1-LP instance) that contain vertices of degree at least two, then we have already destroyed all “large” cycles. Then, we investigate the possible interaction between a cycle of length four and a structure violating (\star) , after having “destroyed” all “mutually interacting” structures violating (\star) .

Lemma 13. *Let $G = (A, B; E)$ be a bipartite graph and π be a fixed permutation of A . Assume that if $h = \{e_1, e_2, e_3\}$ and $h' = \{e'_1, e'_2, e'_3\}$ are two situations violating (\star) , then $h \cap h' = \emptyset$. Let $C = \{ab, bc, cd, da\}$ be a sequence of edges forming a 4-cycle. Then, there is at most one hyperedge h —among the hyperedges modeling situations violating (\star) —such that $C \cap h \neq \emptyset$.*

Hence, after the indicated branching, for each 4-cycle, at most one hyperedge of size three remains such that the corresponding edge sets have non-empty intersection. Since we have to destroy every 4-cycle, the best we then can obviously do is to take out an edge that takes part in the “accompanying” situation violating (\star) . This can be done completely deterministically due to the preceding lemma. Finally, the only remaining situations correspond to possibly interacting 4-cycles. These can be solved with Lemma 11.

Theorem 5. *1-LP can be solved in $\mathcal{O}(k^3 \cdot 2.5616^k + |G|^2)$ time.*

6 Conclusion

In this paper we have presented two methods for producing \mathcal{FPT} algorithms in the context of 2-layer and 1-layer planarization. In particular, for fixed k , we have polynomial time algorithms to determine if $\text{bpr}(G) \leq k$ and $\text{bpr}(G, \pi) \leq k$. The smaller exponential bases (in comparison with [4]) are due to the tight relations with HITTING SET, as we exhibited. For small values of k , our algorithms provide a feasible method for the solution of these \mathcal{NP} -complete problems.

With the results in [4, 5], we have now good kernelization and search tree algorithms for three types of “layered planarization” problems:

1. For 2-LP, we got an $\mathcal{O}(k^2 \cdot 5.1926^k + |G|)$ algorithm and a kernel size $\mathcal{O}(k)$.[†]
2. For 1-LP, we found an $\mathcal{O}(k^3 \cdot 2.5616^k + |G|^2)$ algorithm and a kernel size $\mathcal{O}(k^3)$.
3. For 1-LAYER CROSSING MINIMIZATION, we obtained an $\mathcal{O}(1.4656^k + k|G|^2)$ algorithm and a kernel size $\mathcal{O}(k^2)$, where k is now the number of crossings.

For 2-LAYER CROSSING MINIMIZATION, the (more general) results of [3] only give an $\mathcal{O}(2^{32(2+2k)^3} |G|)$ algorithm, which should be further improvable.

Acknowledgments We are grateful for discussion of this topic with V. Dujmović and for the very thoughtful comments of the reviewers.

References

1. P. Damaschke. Parameterized enumeration, transversals, and imperfect phylogeny reconstruction. In R. Downey, M. Fellows, and F. Dehne, editors, *Intern. Workshop on Parameterized and Exact Computation IWPEC 2004*, volume 3162 of *LNCS*, pages 1–12. Springer, 2004.
2. R. G. Downey and M. R. Fellows. *Parameterized complexity*. Springer, 1999.
3. V. Dujmović, M. Fellows, M. Hallett, M. Kitching, G. Liotta, C. McCartin, N. Nishimura, P. Ragde, F. Rosamond, M. Suderman, S. Whitesides, and D. R. Wood. On the parameterized complexity of layered graph drawing. In F. Meyer auf der Heide, editor, *European Symp. on Algorithms ESA*, volume 2161 of *LNCS*, pages 488–499. Springer, 2001.
4. 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. A fixed-parameter approach to two-layer planarization. In P. Mutzel, M. Jünger, and S. Leipert, editors, *Graph Drawing GD 2001*, volume 2265 of *LNCS*, pages 1–15. Springer, 2002.
5. V. Dujmović, H. Fernau, and M. Kaufmann. Fixed parameter algorithms for one-sided crossing minimization revisited. In G. Liotta, editor, *Graph Drawing GD 2003*, volume 2912 of *LNCS*, pages 332–344. Springer, 2004.
6. H. Fernau. A top-down approach to search-trees: Improved algorithmics for 3-hitting set. TR04-073, Electronic Colloquium on Computational Complexity ECCC, 2004.
7. P. Mutzel. An alternative method to crossing minimization on hierarchical graphs. *SIAM J. Optimization*, 11(4):1065–1080, 2001.
8. R. Niedermeier and P. Rossmanith. An efficient fixed-parameter algorithm for 3-hitting set. *Journal of Discrete Algorithms*, 1:89–102, 2003.
9. N. Nishimura, P. Ragde, and D. Thilikos. Smaller kernels for hitting set problems of constant arity. In R. Downey, M. Fellows, and F. Dehne, editors, *Intern. Workshop on Parameterized and Exact Computation IWPEC 2004*, volume 3162 of *LNCS*, pages 121–126. Springer, 2004.
10. 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.

[†] By changing the heuristic priorities in one case and by using a generalization of rule 3b., we can improve the base to 5.1844.