An Improved Kernel for the Undirected Planar Feedback Vertex Set Problem

Faisal N. Abu-Khzam[1] and Mazen Bou Khuzam[2]

[1] Department of Computer Science and Mathematics
Lebanese American University
Beirut, Lebanon
faisal.abukhzam@lau.edu.lb

[2] Department of Mathematics, Kuwait University
Al Safat, Kuwait
mazen@sci.kuniv.edu.kw

Abstract. We consider the parameterized Feedback Vertex Set problem on unweighted, undirected planar graphs. We present a kernelization algorithm that takes a planar graph G and an integer k as input and either decides that (G, k) is a no instance or produces an equivalent (kernel) instance (G', k') such that $k' \leq k$ and $|V(G')| < 97k$. In addition to the improved kernel bound (from $112k$ to $97k$), our algorithm features simple linear-time reduction procedures that can be applied to the general Feedback Vertex Set problem.

1 Introduction

For a given undirected graph G, a feedback vertex set is a set of vertices whose removal yields an induced forest. The problem of finding a feedback vertex set of minimum size has applications in several domains including (for example) constraint processing and Bayesian inference [1,7]. Formally, the Feedback Vertex Set problem, henceforth FVS, is defined as follows:

Given: A graph G and a positive integer k.
Question: Does G have a feedback vertex set of cardinality k or less?

FVS is NP-complete in general [11], and remains NP-complete when the input is restricted to planar graphs [15]. When parameterized by the solution size, FVS is fixed parameter tractable [9]. This has motivated many successful efforts to design parameterized FVS algorithms [6,8,12,13]. In particular, kernelization algorithms for FVS received some attention lately [2,5,14].

The Feedback Vertex Set problem has a kernelization algorithm that guarantees a quadratic-order kernel in general [14]. Linear kernels have been obtained on graphs of bounded genus and H-minor free graphs [3,10]. On planar graphs, a kernel of order $112k$ has been presented in [4]. In this paper, we present a linear-time kernelization algorithm for planar graphs that delivers kernels of order $97k-203$ (in the worst-case). The restriction to planar graphs allows us to use structural constraints that bound the order (and size) of our reduced instances. However, all the reduction procedures apply to general problem instances.

2 Preliminaries

Throughout this paper, the pair (G, k) denotes a given planar instance of Feedback Vertex Set. (G, k) will undergo a sequence of reduction operations that are based on a set of rules. Each reduction operation transforms (G, k) into a potentially smaller instance, (G', k'), such that $k' \leq k$ and G has a feedback vertex set of size k if and only if (G', k') has a solution of size k'.

We shall use common graph theoretic terminologies. In particular, we denote by $N_H(v)$ the set of neighbors of a vertex v in a graph (or subgraph) H. For $A \subset V$, $G[A]$ denotes the subgraph of G induced by A, and $N_H(A) = \cup_{v \in A} N_H(v) - A$. We denote by $V(G)$ and $E(G)$ the sets of vertices and edges of the graph G, respectively. A graph H is bipartite if $V(H) = A \cup B$ so that each of the subsets A and B induces an edge-less subgraph. We shall refer to such graph by $H = (A, B)$. The complete bipartite graph is denoted by $K_{a,b} = (A, B)$ where $a = |A|$ and $b = |B|$.

We assume the graph G is given with a particular planar embedding. We denote the number of vertices, edges and faces of G by n, e and f respectively. These three values are related via the invariance formula of Euler for connected planar graphs, which states that $n - e + f = 2$. Euler's formula leads to upper bounds on the number of edges for many sub-families of planar graphs. In particular, a simple bipartite planar graph has at most $2n - 4$ edges. In general, a planar graph satisfies: $e \leq 3n - 6$. These two well known formulae are used to compute the upper bound on our kernel size.

During the reduction process, it may be possible to determine that a vertex belongs to a solution of (G, k), if one exists. Such vertex will be deleted and placed in a set S that, henceforth, denotes a potential solution of (G, k). Moreover, the value of k will be decremented by one. This action is justified by the rather trivial fact that deleting a vertex v deletes all cycles containing it. Thus, for $v \in S$, $S \setminus \{v\}$ is a solution of $(G - v, k - 1)$.

Multiple edges between adjacent vertices are possible, as they may arise due to some operations performed on G. When this occurs, we assume that G is drawn (or re-drawn) in the plane in such a way that any cycle of length two is the boundary of a face (i.e., it has an empty interior). Intuitively speaking, this is always possible since a multi-edge can be envisioned as obtained from *thickening* an edge and (then) splitting it. With this in mind, we can always assume that a pair of vertices share at most two edges, since additional edges do not introduce more cycles. When a pair, $\{u, v\}$, of adjacent vertices share two edges, we say that edge uv is a *double-edge*. Otherwise it is a *single-edge*.

We shall also assume that G has no self loops. The existence of a self loop yields an easy detection of an element of S (the single vertex that forms the cycle), which results in deleting the corresponding vertex. Vertex deletion, on the other hand, are applied frequently by our algorithm and could result in disconnecting the graph in question. When this happens, we process the resulting connected components separately, in a sequential manner. So we may always assume that G is connected.

For a vertex $v \in V(G)$, we denote by $C(v)$ the set of all cycles of G that contain v. We extend this notation to subsets of $V(G)$ by setting $C(A) = \cup_{v \in A} C(v)$, where $A \subset V(G)$. We denote by P_i any (sub)graph isomorphic to the path of length i. As usual, the path length is the number of edges forming the path. Moreover, the set of interior vertices of a path P shall be denoted by $I(P)$.

Finally, the concept of cycle domination plays a role in simplifying our arguments. For $\{u, v\} \subset V$, we say that u is dominated by v when $C(u) \subset C(v)$ (every cycle passing through u passes also in v). In other words, a solution that contains v is at most as large as any solution that contains u. Detecting such cycle domination is easy: for each pair $\{u, v\} \subset V$, delete v and check whether u belongs to a cycle of the remaining graph. This checking can be performed via depth-first search in linear time since G has a linear number of edges.

3 Reduction Rules

In this section, we present the reduction rules that are adopted by our kernelization algorithm. Every such rule is stated as a lemma that describes a pair (*condition*, *action*) and asserts that the *action* can be performed if the corresponding *condition* holds. A reduction rule is safe (or sound) if its action transforms (G, k) into an equivalent instance, while updating the potential solution S. We shall assume that, for each i, rules 1 through i are applied exhaustively before applying rule $i + 1$. Some reductions include the addition of edges or double-edges. While this seems contrary to the concept of data reduction, it will be an intermediate step to further reduction in the number of vertices (see Rules 5 and 7 below).

Some of the following rules are folkloric, and are left without proof. In particular, Rules 1-3 have appeared in several FVS algorithms (see [5,8]). We also introduce a few rules that work for general FVS instances and allow us to use planarity to achieve a linear-bound on the kernel size.

Reduction Rule 1. Vertices whose degree is less than two can be deleted.

Reduction Rule 2. If a vertex u has degree two, with incident single-edges uv and uw, then we can delete u and add the edge vw.

Reduction Rule 2*. If a vertex u has two neighbors v and w such that uv is a single-edge and uw is a double-edge, then we can place w in S, decrement k by one, and delete u and w.

Proof. At least one of u and w is in S and $C(u) \subset C(w)$. So it is safe to place w in S and decrement k by one.

Note that Rules 2 and 2* are special cases of the following general reduction rule: if u and v are adjacent vertices and $C(u) \subset C(v)$, then edge uv can be contracted. This rule, however, is not needed to obtain the linear bound presented in this paper.

Reduction Rule 3. If there are more than three edges between two vertices, then we can safely remove all but two of these edges.

Reduction Rule 4. If a vertex u has exactly two neighbors v and w such that uv, uw and vw are double-edges, then we can add both v and w to S, decrement k by two, and delete u, v and w.

Reduction Rule 5. Let uv be a double-edge in G and let w be a degree-three vertex whose neighbors are u, v and w'. Then deleting w and adding the two edges uw' and vw' results in an equivalent instance (See figure 1). This action can be performed even if w' is a neighbor of u or v (or both).

Proof. At least one element of $\{u, v\}$ belongs to S. W.l.o.g., assume $u \in S$. Then the instance $(G-u, k-1)$, which results from deleting u is equivalent to (G, k). In this resulting instance, w can be bypassed, by Rule 2 (w is deleted and the edge vw' is added). Observe also that adding the two edges uw' and vw' introduces only one additional cycle, namely (u, v, w'). If (G, k) is a yes instance, then the new cycle will be covered by at least one element of $\{u, v\}$. Q.E.D.

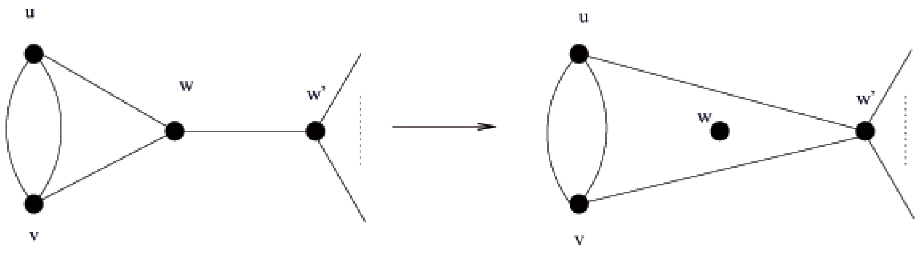

Fig. 1. Rule 5

Reduction Rule 6. Let P be an induced path of endpoints u and v such that $N(P \setminus \{u, v\}) = \{w\}$. If P has more than two interior vertices, then an equivalent instance can be obtained by placing w in S and deleting it. This action results in replacing P by the edge uv due to (the re-application of) Rule 2.

Proof. First note that S must cover the local cycles of $G[P \cup \{w\}]$ and any (global) cycles that pass through the endpoints of P. So it is possible that S contains an element of $I(P)$. If P has interior vertices $w_1, w_2, \cdots, w_s$ such that $s \geq 3$, then each w_i is dominated by $\{w, u\}$ (or $\{w, v\}$). If $s > 3$, and w does not belong to S, then at least two interior vertices of P are in S. So it is safe to add w to S. If $s = 3$ and the solution S has only one element from $I(P)$, then $w_2 \in S$ since neither w_1 nor w_3 covers the cycles through w in $G[P \cup \{w\}]$. However, deleting w_2 will not result in disconnecting $G[P \cup \{w\}]$. Therefore we can replace w_2 by w and obtain an equivalent (if not better) solution.

Reduction Rule 7. Let $(A, B) = K_{2,b}$ be a subgraph of G. If $b \geq 3$ and every vertex of B is of degree three, then we can add a double-edge between the two elements of A. By Rule 5, and (possibly) Rule 4, this leads to deleting all elements of B.

Proof. Note that every element of B is dominated by A (this being true since every element of B has only one neighbor outside A). Let $A = \{v, v'\}$ and $B = \{v_1, v_2, \cdots, v_b\}$. If none of the elements of A is in S, then at least two elements of B are needed to cover the cycles of the $K_{2,b}$-subgraph induced by $A \cup B$. It follows that adding the double-edge between the two elements of A is sound. This addition does not affect the planarity of G for the following reason: once the double-edge vv' is introduced, Rule 5 applies and every common neighbor of degree three of v and v' is deleted, except when two elements of B are adjacent, in which case we use Rule 5 to delete one of them and (then) Rule 4 to delete the other.

The last reduction rule deals with induced paths whose interior vertices have only two neighbors in addition to their endpoints. At this stage, we assume every vertex has degree three or more. We observe the following.

Observation 1. *Let P be an induced path satisfying $|N(I(P))| = 4$. In other words, the interior vertices of P have only two neighbors u and v in addition to the two endpoints of P. If $|I(P)| \geq 5$ and every element of $I(P)$ has at least one neighbor in $\{u, v\}$, then any solution S of (G, k) that does not contain an element of $\{u, v\}$ must contain at least two elements of $I(P)$.*

Proof. If any of u and v has 4 neighbors in $I(P)$, then the assertion holds (trivially). Since $|I(P)| \geq 5$, at least one element of $\{u, v\}$ (say v) has 3 neighbors in $I(P)$, while the other has at least two neighbors. Assume one element $x \in I(P)$ covers all cycles of $G[P \cup \{u, v\}]$. Obviously, x must be a neighbor of v that is interior to the path connecting v's neighbors in $I(P)$. Moreover, x must also be interior to the path connecting the neighbors of u in $I(P)$ (even when u has only two such neighbors, since in that case u and v cannot have common neighbors). If x is deleted, the remaining neighbors of u and v (which "surround" x in P) would be connected via both u and v, forming a cycle of length 6, which contradicts the assumption.

Reduction Rule 8. Let P be an induced path satisfying $|N(I(P))| = 4$. As in the above Observation, the interior vertices of P have only two neighbors u and v in addition to the two endpoints of P. If $|I(P)| = 6$, and $|N_{I(P)}(v)| \geq |N_{I(P)}(u)|$, then a solution of (G, k) exists if and only if (G, k) has a solution that contains v. So, in this case, we delete v and decrement k by one.

Proof. Let $V(P) = \{w_0, w_1, w_2, \cdots, w_s\}$ where $w_i w_{i+1}$ is an edge of $G[P]$ ($0 \leq i < s$; $s \geq 8$). Any solution of (G, k) must cover all the local cycles and possibly disconnect the two endpoints of P in $G[P \cup \{u, v\}]$. We show that two elements of $I(P)$ are either not enough to cover all local cycles or one of them can be replaced by v (safely). To see this, consider first the case where u has at most two neighbors in $I(P)$ and v has at least 4 such neighbors. If v is not in the solution, at least two elements of $I(P)$ are needed to disconnect the local cycles through v. An equivalent (if not better) solution would include v and one of the neighbors of u (to disconnect P and cover the local cycle through u, if any).

Now consider the case where each of u and v has more than two neighbors in $I(P)$. Assume there is a solution that contains only two elements w_i and w_j $(0 < i < j < s)$ and disconnets any local path between w_0 and w_s in $G[P \cup \{u, v\}]$. Then $\{w_i, w_j\}$ is a feedback vertex set for the graph H obtained from $G[P \cup \{u, v\}]$ by adding the edge w_0w_s. Let H' be the acyclic graph obtained by deleting w_i and w_j. The total number of edges incident on u and v in H' is at least 4 (since $|I(P)| \geq 6$), while the number of edges that are deleted from the cycle $\{w_0, w_1, \cdots, w_s\}$ is at most 4. It follows that H' has at least $s + 1$ edges and exactly $s + 1$ vertices, which contradicts the assumtpion that H' is acyclic. Therefore, any solution that contains only two elements of $I(P)$ can result in either not covering all the local cycles or not disconnecting all local paths between w_0 and w_s (in $G[P \cup \{u, v\}]$). In such case, it would be safe to take both u and v into the solution.

The instance (G, k) is said to be reduced with respect to the above reduction rules if none of them applies to (G, k). We shall prove that such a reduced instance has a solution only if $|V(G)| < 97k$.

4 A Linear Kernel

Our reduction process consists of applying the rules of the previous section exhaustively until none of them applies. Applying a rule starts by a search for a graph structure that makes its condition holds. If this search is successful, it will be followed by the action described by the rule. The resulting instance, (G, k), is called a reduced instance.

Note that every reduction rule can be applied in linear time. In fact, it should be clear that applying each of the Rules 1-4 takes linear time, especially if an updated degrees-array is used. Moreover, it takes linear-time to check if the condition of Rule 5 holds (select a degree-three vertex and check whether two of its neighbors share a double-edge). Each of the conditions of the remaining rules can be checked via depth-first search (DFS). Starting with any vertex of G, DFS takes $O(E(G))$ (thus, linear-time) to find a path or a $K_{2,b}$ that satisfies the required condition. Note that we require the "successive" application of all the rules until they cannot be applied. The process of applying the rules is repeated whenever k is decremented. It follows that the total reduction time is in $O(kn)$.

Throughout the rest of this section we assume that (G, k) is a reduced **yes** instance. Again, let S be a corresponding (complete) solution, and assume G is a plane graph. By Rule 2, the degree of every vertex of G is bounded below by three.

The complement of S in $V(G)$ induces a forest F. Let L be the set of tree-leaves in F. Each such leaf must have at least two neighbors in S. This is obvious since each leaf has only one neighbor in F and at least three neighbors in G. Note the use of Rule 2* here: the unique edge that is incident on a leaf of F is a single-edge. So a leaf node with only one neighbor in S will be deleted, even if its degree is three. Our objective is to compute an upper bound on the cardinality of L, and use it to bound the number of elements in F. The following observation is needed in the sequal.

Observation 2. *Let u and v be non-adjacent vertices of a planar graph H. If u and v have a common neighbor w whose degree is two in H, then the graph obtained from H by adding the edge uv is also planar.*

Proof. The following operations preserve planarity: contract edge uw, duplicate the (resulting) edge uv, then subdivide one of the edges that connect u and v (to get w back).

Our main result depends on some counting arguments that make use of the following lemmas. Hereafter, a $(c, 2)$-chain of a graph G is an induced path on c vertices (and length $c - 1 \geq 0$), each of which is of degree two in G. When the length of the induced path is not important (or arbitrary), such a path is dubbed a degree-two chain. A degree-two chain is maximal if it is not contained in a larger degree-two chain.

Consider a tree T and let C be a collection of disjoint $(c, 2)$-chains of interior vertices in T. We say that C is maximal if $T[V(T) \setminus V(C)]$ has no $(c, 2)$-chains.

Lemma 1. *Let T be a tree with at most l leaves. If, for $c \geq 1$, every maximal collection of disjoint $(c, 2)$-chains has at most n_c internal vertices, then T has at most $n_c + 2cl - 2c + 1$ vertices.*

Proof. The number of degree-three vertices of T is less than the number of its leaves. Let T' be the tree obtained from T by replacing every maximal degree-two chain by a single edge (by a sequence of edge-contractions). Then T' has at most $l - 1$ internal vertices and at most $2l - 2$ edges. Consequently, the number of maximal degree-two chains in T is bounded above by $2l - 2$. The length of any degree-two chain is either smaller than c or consists of (or can be decomposed into) one or more $(c, 2)$-chains and at most one $(b, 2)$-chain, $b \leq c-1$. It follows that the number of internal degree-two vertices is bounded above by $n_c + (c - 1)(2l - 2)$. Q.E.D.

We now show that n_6 is bounded by a linear function of k and that F has a linear number of leaves.

Lemma 2. *Let $H = (A, B)$ be a simple bipartite planar graph. If $|A| = k$ and every vertex of B has at least three neighbors in A, then $|B| \leq 2k - 4$.*

Proof. Recall that, due to a corollary of Euler's formula, H has at most $2n_h - 4$ edges, where $n_h = |V(H)| = k + |B|$. Since every vertex of B is of degree at least three, we have $2n_h - 4 \geq |E(H)| \geq 3|B|$. Replacing n_h by $k + |B|$ gives the desired inequality.

Lemma 3. *If C is a maximal collection of $(6, 2)$-chains of internal vertices in F, then $|C| \leq 2k - 4$.*

Proof. Let P be a $(6, 2)$-chain of internal vertices in F. By Rules 6 and 8, $N(P)$ has at least three elements from S (since P is $I(P')$ for some path P' of length 8 in F). Let C' be the set of vertices obtained by contracting every chain in C (thus replacing every $(6, 2)$-chain by a single vertex of degree ≥ 3). Ignoring the edges between elements of S, and replacing multiple edges by simple edges, we consider the resulting simple bipartite planar graph (S, C'). The result now follows from Lemma 2.

Lemma 4. *Let L_2 be the subset of L consisting of leaves that have exactly two neighbors in S. Then $|L_2| \leq 6k - 12$.*

Proof. For each pair $\{u, v\} \subset S$, the number of elements of L_2 that are common neighbors of u and v is at most 2. This follows easily from Rule 7. Consider the planar graph H induced by $S \cup L_2$. And let H' be a graph obtained from H by adding an edge between non-adjacent elements of S that share a common neighbor. Then, by Observation 2, H' is also planar. Moreover, the number of pairs of S that can share a common neighbor (from L) is bounded above by the number of edges in H', which is at most $3k - 6$. The proof is now complete (knowing that each such pair has at most 2 common neighbors in L).

We now consider the leaves of F, if any, that have at least three neighbors in S.

Lemma 5. *Let $L_3 = L \backslash L_2$ be the set of leaves of F that have more than two neighbors in S. Then $|L_3| \leq 2k - 4$.*

Proof. Consider the planar bipartite subgraph H whose vertex set is $S \cup L_3$ and whose edges are elements of $E(G)$ that connect elements of S to those of L_3. The proof follows from Lemma 2 by letting l be the number of leaves in L_3 and n_h the number of vertices in H.

The above two lemmas guarantee that our reduced instance has a solution S of size k only if the corresponding induced forest F has at most $8k - 16$ leaves and at most $2k - 4$ $(6, 2)$-chains. In fact, the number of elements of F that have three neighbors in S and the number of $(6, 2)$-chains of F (which is $n_6/6$) sum up to at most $2k - 4$. This can be obtained by considering (again) a bipartite plane graph $H = (A, B)$ where $A = S$ and B consists of "contracted" $(6, 2)$-chains and all elements of F that have three neighbiors in S. Therefore $\frac{n_6}{6} + |L_3| \leq 2k - 4$ (or $n_6 \leq 12k - 6|L_3| - 24$). Moreover, $l \leq 6k - 12 + |L_3|$ (again l is the total number of leaves in F).

It follows, by Lemmas 1, 4 and 5 that $|F|$ is bounded above by:
$n_6 + 2(6)l - 2(6) + 1 \leq (12k - 6|L_3| - 24) + 12(6k + |L_3| - 12) - 11 = 84k + 6|L_3| - 179 \leq 84k + 6(2k - 4) - 179 = 96k - 203$.

This proves our claimed linear kernel bound, which we state in the following theorem.

Theorem 1. *There is a linear-time algorithm that, given an arbitrary planar instance (G, k) of Feedback Vertex Set, either decides that no solution exists or produces an equivalent instance whose size is bounded above by $97k - 203$.*

5 Conclusion

We showed how to reduce any planar instance of Feedback Vertex Set into one whose order is bounded by $97k - 203$, where k is the input parameter. Our reduction rules apply to general FVS instances and might be useful on their own, as preprocessing steps.

Our kernel bound is obtained by counting arguments that made heavy use of planarity. This bound can be improved further via reductions that consider induced paths as in Rule 8, and by more detailed counting arguments. We continue to investigate the potential use of our techniques to obtain a much smaller kernel bound without loosing the simplicity and efficiency of the corresponding reduction procedures.

References

1. Bar-Yehuda, R., Geiger, D., Naor, J.(S.), Roth, R.M.: Approximation algorithms for the vertex feedback set problem with applications to constraint satisfaction and bayesian inference. In: SODA 1994: Proceedings of the Fifth Annual ACM-SIAM Symposium on Discrete Algorithms, Philadelphia, PA, USA, pp. 344–354. Society for Industrial and Applied Mathematics (1994)
2. Bodlaender, H.L.: A Cubic Kernel for Feedback Vertex Set. In: Thomas, W., Weil, P. (eds.) STACS 2007. LNCS, vol. 4393, pp. 320–331. Springer, Heidelberg (2007)
3. Bodlaender, H.L., Fomin, F.V., Lokshtanov, D., Penninkx, E., Saurabh, S., Thilikos, D.M.: (meta) kernelization. In: Proceedings of the 2009 50th Annual IEEE Symposium on Foundations of Computer Science, FOCS 2009, pp. 629–638. IEEE Computer Society, Washington, DC (2009)
4. Bodlaender, H.L., Penninkx, E.: A Linear Kernel for Planar Feedback Vertex Set. In: Grohe, M., Niedermeier, R. (eds.) IWPEC 2008. LNCS, vol. 5018, pp. 160–171. Springer, Heidelberg (2008)
5. Burrage, K., Estivill-Castro, V., Fellows, M.R., Langston, M.A., Mac, S., Rosamond, F.A.: The Undirected Feedback Vertex Set Problem Has a Poly(k) Kernel. In: Bodlaender, H.L., Langston, M.A. (eds.) IWPEC 2006. LNCS, vol. 4169, pp. 192–202. Springer, Heidelberg (2006)
6. Chen, J., Fomin, F.V., Liu, Y., Lu, S., Villanger, Y.: Improved Algorithms for the Feedback Vertex Set Problems. In: Dehne, F., Sack, J.-R., Zeh, N. (eds.) WADS 2007. LNCS, vol. 4619, pp. 422–433. Springer, Heidelberg (2007)

7. Dechter, R.: Enhancement schemes for constraint processing: backjumping, learning, and cutset decomposition. Artificial Intelligence 41(3), 273–312 (1990)
8. Dehne, F., Fellows, M.R., Langston, M.A., Rosamond, F.A., Stevens, K.: An $o^*(2^{O(k)})$ FPT Algorithm for the Undirected Feedback Vertex Set Problem. In: Wang, L. (ed.) COCOON 2005. LNCS, vol. 3595, pp. 859–869. Springer, Heidelberg (2005)
9. Downey, R.G., Fellows, M.R.: Parameterized Complexity. Springer (1999)
10. Fomin, F.V., Lokshtanov, D., Saurabh, S., Thilikos, D.M.: Bidimensionality and kernels. In: Proceedings of the Twenty-First Annual ACM-SIAM Symposium on Discrete Algorithms, SODA 2010, Philadelphia, PA, USA, pp. 503–510. Society for Industrial and Applied Mathematics (2010)
11. Garey, M.R., Johnson, D.S.: Computers and Intractability. W. H. Freeman, New York (1979)
12. Guo, J., Gramm, J., Hüffner, F., Niedermeier, R., Wernicke, S.: Improved Fixed-Parameter Algorithms for Two Feedback Set Problems. In: Dehne, F., López-Ortiz, A., Sack, J.-R. (eds.) WADS 2005. LNCS, vol. 3608, pp. 158–168. Springer, Heidelberg (2005)
13. Raman, V., Saurabh, S., Subramanian, C.R.: Faster fixed parameter tractable algorithms for finding feedback vertex sets. ACM Trans. Algorithms 2(3), 403–415 (2006)
14. Thomassé, S.: A $4k^2$ kernel for feedback vertex set. ACM Transactions on Algorithms TALG 6(2), 1–8 (2010)
15. Yannakakis, M.: Node-and edge-deletion np-complete problems. In: STOC 1978: Proceedings of the Tenth Annual ACM Symposium on Theory of Computing, pp. 253–264. ACM, New York (1978)