Parameterized Algorithms for Feedback Vertex Set

Iyad Kanj[1] Michael Pelsmajer[2] Marcus Schaefer[1]

[1] School of Computer Science, Telecommunications and Information Systems, DePaul University, 243 S. Wabash Avenue, Chicago, IL 60604-2301.
{ikanj,mschaefer}@cs.depaul.edu*

[2] Department of Applied Mathematics, Illinois Institute of Technology, Chicago, IL 60616.
pelsmajer@iit.edu

Abstract. We present an algorithm for the parameterized feedback vertex set problem that runs in time $O((2\lg k + 2\lg\lg k + 18)^k n^2)$. This improves the previous $O(max\{12^k, (4\lg k)^k\}n^\omega)$ algorithm by Raman et al. by roughly a 2^k factor ($n^w \in O(n^{2.376})$ is the time needed to multiply two $n \times n$ matrices). Our results are obtained by developing new combinatorial tools and employing results from extremal graph theory. We also show that for several special classes of graphs the feedback vertex set problem can be solved in time $c^k n^{O(1)}$ for some constant c. This includes, for example, graphs of genus $O(\lg n)$.

1 Introduction

Given an undirected graph G, a *feedback vertex set* in G is a subset of vertices F in G such that $G - F$ is acyclic. The *size* of a feedback vertex set F is $|F|$. The FEEDBACK VERTEX SET problem (FVS) is: given a graph G and a positive integer k, decide if G has a feedback vertex set of size at most k. It is well-known that the FVS problem is NP-complete on both directed and undirected graphs [17]. The minimization version of this problem has been studied intensively from the approximability point of view [1, 2, 13, 16, 24] due to its important applications in fields like circuit testing, deadlock resolution, and analyzing manufacturing processes [13, 18, 21]. For example, in the field of circuit testing, a small set of registers (vertices) needs to be identified in the circuit (graph) whose removal makes the circuit testable (i.e., the circuit needs to be acyclic) [18].

The FVS problem has also received considerable attention from the parameterized complexity point of view [3, 4, 10, 11, 23]. A parameterized problem is said to be *fixed-parameter tractable* if the problem can be solved in time $f(k)n^{O(1)}$ for some function f which is independent of n [11]. The class FPT denotes the class of all fixed-parameter tractable problems [11]. It was shown that the FVS problem on undirected graphs is in FPT [4, 10, 11], whereas it remains an open question whether the FVS problem on directed graphs is in FPT [11].

* The first author was supported in part by DePaul University Competitive Research Grant.

Once a problem has been shown to be in FPT, the search for better algorithms for the problem continues; that is, algorithms that remain practical for larger values of the parameter k. Successful examples of such developments include the VERTEX COVER and PLANAR DOMINATING SET problems (see [7, 15] and their references). The same holds true for the FVS problem on undirected graphs. Bodlaender [4], and Downey and Fellows [10], were the first to show that the problem is in FPT. In [11], Downey and Fellows presented an $O((2k+1)^k n^2)$ time algorithm for the problem. Becker et al. [3] gave a randomized algorithm running in time $O(4^k kn)$ that finds a minimum feedback vertex set of size k with probability at least $1-(1-4^{-k})^{c4^k}$ for an arbitrary constant c. By observing that every undirected graph on n vertices with minimum degree at least 3 has a cycle of length bounded by $2\lg n+1$, Raman presented a very simple algorithm for the problem running in time $O((6k\lg k)^{k+1}n^\omega)$ [22], where $n^\omega \in O(n^{2.376})$ is the running time of the best algorithm for multiplying two $n \times n$ matrices [8]. More recently, using some nice combinatorial techniques, Raman et al. [23] presented an algorithm for the problem running in time $O(max\{12^k, (4\lg k)^k\}n^\omega)$ improving significantly the $O((2k+1)^k n^2)$ time algorithm given in [11] (when k is sufficiently large).

In this paper we continue the efforts towards reducing the running time of the algorithms for the FVS problem. We develop new combinatorial tools and employ known results from extremal graph theory to show that the size of a minimum feedback vertex set is $\Omega(n/\lg n)$ in a graph with no cycles of length bounded by $\lg n$. This allows us to obtain an $O((2\lg k+2\lg\lg k+18)^k n^2)$ time algorithm for the FVS problem, improving the previous $O(max\{12^k, (4\lg k)^k\}n^\omega)$ time algorithm in [23] by roughly a 2^k factor. Obviously, the running time of this algorithm is still far from being practical, and the question of whether the FVS problem can be solved in time $c^k n^{O(1)}$ remains open.

We also consider the FVS problem on special classes of graphs. We show that the problem on graphs of genus $O(\lg n)$ and on K^r-minor free graphs is solvable in time $c^k n^{O(1)}$ for some constant c. We also show that the problem on bipartite graphs, graphs of genus $O(n^\epsilon)$ for any $\epsilon > 0$, and constant average degree graphs, can be solved in time $c^k n^{O(1)}$ for some constant c if and only if the FVS problem on general graphs can.

2 Preliminaries

Let $G=(V,E)$ be an undirected graph. For a set of vertices S in G we denote by $G-S$ the subgraph of G that results from removing the vertices in S, together with all edges incident to them. A *minimum feedback vertex set* is a feedback vertex set of minimum size. We denote by $\gamma(G)$ the size of a minimum feedback vertex set in G. For a vertex v, we denote by $deg(v)$ the degree of v in G.

Let v be a vertex in G such that $deg(v) \leq 2$. We define the following operation, which is standard in the literature. If $deg(v)=1$ then remove v (together with its incident edge) from G; if $deg(v)=2$ and the two neighbors x and y of v are not adjacent, then remove v and add an edge between x and y. Let us denote

this operation by **Short-Cut(v)**. We say that the operation **Short-Cut()** is *applicable* to a vertex v, if either $deg(v) = 1$, or $deg(v) = 2$ and the two neighbors of v are non-adjacent. A variation of the following proposition appears in [23] (see [2] for a proof).

Proposition 1. *(Lemma 1, [23]) Let G be an undirected graph and let v be a vertex in G to which the operation* **Short-Cut()** *is applicable. Let G' be the graph resulting from applying* **Short-Cut(v)**. *Then $\gamma(G') = \gamma(G)$.*

We assume that we have a subroutine **Clean(G)** which applies the operation **Short-Cut()** repeatedly to G until the operation is no longer applicable. It is clear from Proposition 1 that if G' is the resulting graph from applying **Clean(G)**, then $\gamma(G') = \gamma(G)$. The graph G is said to be *clean*, if **Clean(G)** is not applicable. Note that any degree-2 vertex in a clean graph must be on a cycle of length three.

An *almost shortest* cycle in G is a cycle whose length is at most the length of a shortest cycle in G plus one. It is well-known that an almost shortest cycle in an undirected graph with n vertices can be found in time $O(n^2)$ [19]. It is also well-known that any undirected graph with minimum degree at least 3 has a cycle of length at most $2 \lg n + 1$ [12].

3 The Algorithm

The basic idea behind most of the parameterized algorithms for the FVS problem presented so far has been to *branch* on short cycles and use the search tree method [11, 22, 23]. Suppose we are trying to determine if there exists a feedback vertex set in G of size bounded by k. Let C be a cycle in G of length l. Then every feedback vertex set of G must contain at least one vertex of C. For every vertex v on C, we can include v in the feedback vertex set, and then recurse to determine if $G - v$ has a feedback vertex set of size $k - 1$. Let us call such a process: *branching on* the cycle C. If we let $T(k)$ be the number of nodes in the search tree of such an algorithm that looks for a feedback vertex set of size bounded by k, then when the algorithm branches on a cycle of length l, $T(k)$ can be expressed using the recurrence relation $T(k) \leq l \cdot T(k-1) + 1$. The number of nodes in the search tree corresponding to the algorithm is $O((l_{max})^k)$, where l_{max} is the length of the longest cycle the algorithm branches on [11]. The running time of the algorithm is now proportional to the number of nodes in the search tree multiplied by the time we spend at every node of the search tree to find a cycle and process the graph. Thus, to reduce the running time of the algorithm, it is desirable to branch on short cycles. Most parameterized algorithms so far hinge on this approach [11, 22, 23].

In this section we develop another algorithm that uses this approach (based on the algorithm in [23]). We present the algorithm in Figure 1 below. We prove its correctness and analyze its running time in the next section.

FVS-solver

Input: an instance (G, k) of FVS
Output: a feedback vertex set F of G of size bounded by k in case it exists

0. $F = \emptyset$;
1. **if** G is acyclic then **return**(F);
2. **if** $k = 0$ and G contains a cycle then **return**('NO');
3. apply **Clean(G)**;
4. let C be an almost shortest cycle in G of length l;
5. **if** $l > 13$ and $k \leq 3\sqrt{n}$ **then return**('NO');
   else if $l > \lg n + 1$ and $(\lg n > \lg k + \lg\lg k + 13)$ **then return**('NO');
   else if $l > max\{2\lg k + 18, 2\lg n - 9\}$ **then return**('NO');
   else branch on C and update k, F, and G accordingly;

Fig. 1. The algorithm FVS-solver

4 Analysis of FVS-solver

The main idea behind the analysis of the algorithm presented in [23] is that if the graph does not contain a cycle of constant length, then the size of the feedback vertex set k must be large. In particular, the following structural result immediately follows from [23].

Lemma 1. *(Theorem 2, [23]) Let G be a graph on n vertices with minimum degree at least 3. If there is no cycle in G of length at most 12 then $\gamma(G) > 3\sqrt{n}$.*

The above result shows that when a clean graph has no cycle of length bounded by 12 (note that no degree-2 vertex exists in G at this point since G does not contain a cycle of length 3), $k > 3\sqrt{n}$, and hence, $\lg n < 2\lg k - 3$. Since every undirected graph with minimum degree at least 3 must have a cycle of length bounded by $2\lg n + 1$, G has a cycle of length bounded by $4\lg k$. An algorithm that branches on a shortest cycle will then either branch on a cycle of length at most 12, or of length at most $4\lg k$. According to the discussion in the previous section, this gives a search tree of size $O((max\{12, 4\lg k\})^k)$.

In this section we develop new combinatorial techniques and employ results from extremal graph theory to improve this analysis. The structural results obtained in this section will allow us to prove an upper bound of $O((2\lg k + 2\lg\lg k + 18)^k)$ on the size of the search tree of the algorithm **FVS-solver** presented in the previous section.

Let T be a tree. For a vertex v in T we denote by $d_T(v)$ the degree of v in T. A vertex $v \in T$ is said to be *good* if $d_T(v) \leq 2$. The statement of the following lemma can be easily proved.[3]

Lemma 2. *There are at least $\lfloor q/2 + 1 \rfloor$ good vertices in a tree on q vertices.*

[3] An easy way to see why the statement is true is to note that the average degree of a tree is bounded by 2.

For a good vertex u in T, $\{u\}$ is said to be a *nice* set if $d_T(u) \leq 1$, and for two good vertices u and v in T, the set $\{u, v\}$ is said to be a *nice* set if (u, v) is an edge in T.

Lemma 3. *Let T be a tree, and let n_g be the number of good vertices in T. Then there exists at least $\lfloor (n_g + 1)/2 \rfloor$ nice sets in T that are pairwise disjoint.*

Proof. Without loss of generality, we assume that T is rooted at a vertex r. We define the natural parent-child, and ancestor-descendent relationships, between the vertices in T. Note that each vertex in T is either the root vertex or has exactly one parent in T. For every vertex v in T, let T_v denote the subtree of T rooted at v and containing all its descendents. We will prove the following statement: if T_v contains q good vertices of T, then the number of pairwise disjoint nice sets in T_v is at least $\lfloor (q+1)/2 \rfloor$. It is clear that the previous statement will imply the statement of the lemma because $T = T_r$. (Observe that a good vertex in T_v might not be a good vertex in T, and this is why we require the q vertices to be good in T.) To prove the statement, let T_v be a rooted subtree of T, and proceed by induction on q. T_v must contain at least one leaf of T, so $q > 0$. If $q \leq 2$, then T_v must contain a leaf u in T, and $\{u\}$ is a nice set. Therefore the number of (pairwise disjoint) nice sets in this case is at least $1 \geq \lfloor (q + 1)/2 \rfloor$. Suppose now that the number of pairwise disjoint nice sets in any rooted tree containing p good vertices from T, for $2 \leq p < q$, is at least $\lfloor (p + 1)/2 \rfloor$. We distinguish two cases.

Case 1. v has at least two children. Let $v_1, \ldots, v_d$, be the children of v in T_v. Let q_i be the number of good vertices in T that are in T_{v_i}, $i = 1 \ldots d$, and note that $1 \leq q_i < q$, and that $q \leq q_1 + \ldots q_d + 1$ (v might be good, in which case it is the root of the tree and $d = 2$). By the inductive hypothesis, each T_{v_i} contains at least $\lfloor (q_i + 1)/2 \rfloor$ pairwise disjoint nice sets in T. Since every nice set in T_{v_i} is disjoint from every nice set in T_{v_j} for $1 \leq i \neq j \leq d$, it follows that T_v contains at least $\lfloor (q_1 + \ldots + q_d + d)/2 \rfloor = \lfloor (q_1 + \ldots q_d + 1 + d - 1)/2 \rfloor \geq \lfloor (q + 1)/2 \rfloor$ pairwise disjoint nice sets in T (note that $d \geq 2$).

Case 2. v has exactly one child v'. In this case v must be a good vertex in T. If v' is good, let v'' be the child of v' in T_v (note that v'' must exist since $q > 2$). Now $T_{v''}$ contains $q - 2$ good vertices in T. By induction, the number of pairwise disjoint nice sets in $T_{v''}$ is at least $\lfloor (q - 1)/2 \rfloor$. Since $\{v, v'\}$ is a nice set which is disjoint from all the nice sets in $T_{v''}$, it follows that the number of pairwise disjoint nice sets in T_v is at least $\lfloor (q + 1)/2 \rfloor$. If v' is bad, then v' must have at least two children in $T_{v'}$. The proof now is identical to that of **Case 1** by applying induction on the trees rooted at the children of v'.

This completes the induction and the proof. □

This lemma follows directly from Lemma 2 and Lemma 3.

Lemma 4. *Let T be a tree on q vertices. There are at least $q/4$ pairwise disjoint nice sets in T.*

Let (G, k) be an instance of FVS, where G is clean and has n vertices, and assume that G does not have a cycle of length bounded by 12. Since G is clean and has no cycles of length 3, G has minimum degree at least 3. Let F be a minimum feedback set of G, let $f = |F|$, and $\mathcal{F} = G - F$. Applying Lemma 4 to every tree in $\mathcal{F}$, we get that $\mathcal{F}$ (i.e., the trees in $\mathcal{F}$) contains at least $(n-f)/4$ pairwise disjoint nice sets. So if we let S be the set of pairwise disjoint nice sets in $\mathcal{F}$, then $|S| \geq (n-f)/4$. We construct a graph G_F as follows. The set of vertices of G_F is F. The edges of G_F are defined as follows. Let $\{a, b\}$ be a nice set in S. Since a and b are good in $\mathcal{F}$ (i.e., in the tree of $\mathcal{F}$ that they belong to) and both have degree greater or equal to 3 in G, a must have at least one neighbor in F, and b must have at least one neighbor F. Pick exactly one neighbor a_1 of a in F, and exactly one neighbor b_1 of b in F. Note that a_1 and b_1 must be distinct since G has no cycles of length 3. Add the edge (a_1, b_1) to G_F. Now let $\{a\}$ be a nice set in S, then a must have at least two neighbors in F. We pick exactly two neighbors a_1 and a_2 of a in F, and we add the edge (a_1, a_2) in G_F. This completes the construction of G_F. Since G has no cycles of length bounded by 6, for any two distinct nice sets in S, the edges associated with them in G_F are distinct. This means that G_F is a simple graph with at least $(n-f)/4$ edges. We have the following lemma.

Lemma 5. *If G_F has a cycle of length l then G has a cycle of length bounded by $3l$.*

Proof. Let $(v_1, \ldots, v_l, v_1)$ be a cycle in G_F. Since each (v_i, v_{i+1}) (the index arithmetic is taken modulo l) is an edge in G_F, (v_i, v_{i+1}) is associated with a nice set S_i in S. If $S_i = \{a\}$, let P_i be the path (v_i, a, v_{i+1}) in G; if $S_i = \{a, b\}$, let P_i be the path (v_i, a, b, v_{i+1}) in G. Since the nice sets in S are disjoint, the paths P_i, $i = 1, \ldots, l$, are internally vertex-disjoint in G, and they determine a cycle of length bounded by $3l$. □

The following result is known in extremal graph theory (see [5, 6, 14]).

Lemma 6. *Let G be a graph with n vertices where $n \geq 3$. If G does not contain a cycle of length at most $2l$, then the number of edges in G is bounded by $90ln^{1+1/l}$.*

Theorem 1. *Let G be a graph with $n \geq 3$ vertices and with no cycles of length bounded by $max\{12, \lg n\}$. Then $\gamma(G) \geq (n/(61 \lg n))^{1-6/(\lg n+6)}$.*

Proof. Suppose that G does not have a cycle of length bounded by $max\{12, \lg n\}$. Let F be a minimum feedback vertex set of G, and let $f = |F| = \gamma(G)$. Since G has no cycles of length bounded by 12, if we let $\mathcal{F}$ and G_F be as defined in the above discussion, then it follows from above that the number of edges in G_F is at least $(n-f)/4$. Since G does not have a cycle of length bounded by $\lg n$, by Lemma 5, G_F has no cycle of length bounded by $(\lg n)/3$. Applying Lemma 6 with $l = (\lg n)/6$, we get that the number of edges in G_F is bounded by $15(\lg n) f^{1+6/\lg n}$. Thus

$$
\begin{aligned}
(n-f)/4 &\leq 15(\lg n) f^{1+6/\lg n} \\
n &\leq 60(\lg n) f^{1+6/\lg n} + f \leq 60(\lg n) f^{1+6/\lg n} + f^{1+6/\lg n} \\
n &\leq 61(\lg n) f^{1+6/\lg n} \qquad (1)
\end{aligned}
$$

Manipulating (1) we get $f \geq (n/(61 \lg n))^{1-6/(\lg n+6)}$ completing the proof. □

Theorem 1 implies that the size of a minimum feedback vertex set in a graph with minimum degree at least 3 and no cycles of length bounded by $\lg n$ must be $\Omega(n/\lg n)$.

Corollary 1. *Let G be a graph with $n \geq 3$ vertices and with no cycles of length bounded by $max\{12, \lg n\}$. Then $\lg n \leq \lg \gamma(G) + \lg \lg \gamma(G) + 13$.*

Proof. Applying the $\lg()$ function on both sides of (1) in Theorem 1 we get

$$\lg n \leq \lg 61 + \lg \lg n + \lg f + 6 \lg f / \lg n \qquad (2)$$

Since G has no cycles of length bounded by 12, from Lemma 1 we get $\lg n < 2 \lg f$, and hence $\lg \lg n < 1 + \lg \lg f$. Now noting that $\lg 61 \leq 6$ and $\lg f \leq \lg n$, it follows from (2) that $\lg n \leq \lg f + \lg \lg f + 13$. □

Lemma 7. *Let G be a graph with n vertices and minimum degree at least 3. For any integer constant $d > 0$, there is a cycle in G of length at most $max\{2 \lg \gamma(G) + 2 \lg (d-1) + 3, 2 \lg n - 2 \lg (d+1) + 4\}$.*

Proof. Let $d > 0$ be given, and let Δ be the maximum degree of G. By Lemma 4 in [23], $\gamma(G) > (\delta - 2)n/(2(\Delta - 1))$, where δ is the minimum degree of the graph. Since $\delta \geq 3$, we get $n < 2(\Delta - 1)\gamma(G)$, and hence

$$
\begin{aligned}
\lg n &< \lg \gamma(G) + \lg (\Delta - 1) + 1 \\
2 \lg n + 1 &< 2 \lg \gamma(G) + 2 \lg (\Delta - 1) + 3 \qquad (3)
\end{aligned}
$$

Let r be a vertex in G of degree Δ. Perform a breadth-first search starting at r until a shortest cycle is first encountered in the graph. Let l be the length of the shortest cycle first encountered in this process. Since r has degree Δ and every other vertex has degree at least 3, it is not difficult to show, using a counting argument, that the number of vertices in the graph is at least $\Delta(2^{(l-2)/2} - 1) + 1$ if l is even, and $\Delta(2^{(l-1)/2} - 1) + 1$ if l is odd.

Suppose that $l \geq 4$. We have

$$
\begin{aligned}
n &\geq \Delta(2^{(l-2)/2} - 1) + 1 \\
n &> \Delta(2^{(l-2)/2} - 1) \\
\lg n &> \lg \Delta + \lg (2^{(l-2)/2} - 1) \qquad (4)
\end{aligned}
$$

Also $l \geq 4$ implies that $2^{(l-2)/2} \geq 2$. Using the inequality $\lg(x-1) \geq \lg x - 1$ for $x \geq 2$ in (4), and manipulating (4), we get

$$l \leq 2\lg n - 2\lg \Delta + 4 \tag{5}$$

If $l < 4$, then $l = 3$ and since $n > \Delta$, (5) is still true.

Now if $\Delta \leq d$, then from the fact that G has a cycle of length bounded by $2\lg n + 1$, and from (3), we conclude that there is a cycle in G of length at most $2\lg\gamma(G) + 2\lg(\Delta - 1) + 3 \leq 2\lg\gamma(G) + 2\lg(d-1) + 3$. Otherwise, $\Delta \geq d+1$, and by (5), there is a cycle in G of length at most $2\lg n - 2\lg\Delta + 4 \leq 2\lg n - 2\lg(d+1) + 4$. It follows that G has a cycle of length at most $max\{2\lg\gamma(G) + 2\lg(d-1) + 3, 2\lg n - 2\lg(d+1) + 4\}$. This completes the proof. □

Corollary 2. *Let G be a graph with minimum degree at least 3. There exists a cycle in G of length at most $max\{2\lg\gamma(G) + 17, 2\lg n - 10\}$*

Proof. Apply Lemma 7 with $d = 127$. □

Theorem 2. *Let G be a graph with n vertices. In time $O((2\lg k + 2\lg\lg k + 18)^k n^2)$ we can decide if G has a feedback vertex set of size bounded by k.*

Proof. It is not difficult to see that the algorithm **FVS-solver** solves the FVS problem. In particular, if **FVS-solver** returns a NO answer in step 5, then either (i) $l > 13$ and $k \leq 3\sqrt{n}$, or (ii) $l > \lg n + 1$ and $\lg n > \lg k + \lg\lg k + 13$, or (iii) $l > max\{2\lg k + 18, 2\lg n - 9\}$. Since l is the length of an almost shortest cycle, if (i) holds then G does not have a cycle of length bounded by 12, and hence no feedback vertex set of size bounded by k by Lemma 1. If (ii) holds, then G does not have a cycle of length bounded by $max\{12, \lg n\}$, and hence no feedback vertex set of size bounded by k by Corollary 1 (note that in this case $n \geq 3$, and l must also be greater than 13 since $\lg n > 13$). Finally if (iii) holds, then G has no cycle of length bounded by $\{2\lg k + 17, 2\lg n - 10\}$. From Corollary 2 (note that G has minimum degree at least three at this point since G is clean and has no cycles of length three) we conclude that k must be smaller than $\gamma(G)$, and hence G has no feedback vertex set of size bounded by k.

To analyze the running time of the algorithm, let l be the length of a cycle C that the algorithm branches on. By looking at step 5 in the algorithm, we see that if the algorithm branches on C then one of the following cases must hold: (a) $l \leq 13$, or (b) $l > 13$, $k > 3\sqrt{n}$, and $l \leq \lg n + 1$, or (c) $\lg n \leq \lg k + \lg\lg k + 13$ and $l \leq max\{2\lg k + 18, 2\lg n - 9\}$.

If (b) holds, then the conditions in (b) give that $l < 2\lg k$. If (c) holds, then combining the two inequalities in (c) we get $l \leq max\{2\lg k + 18, 2\lg k + 2\lg\lg k + 17\} \leq 2\lg k + 2\lg\lg k + 18$. It follows that in all cases (a), (b), and (c), the algorithm branches on a cycle of length at most $2\lg k + 2\lg\lg k + 18$.

Thus, according to the discussion at the beginning of this section, the size of the search tree corresponding to the algorithm is $O((2\lg k + 2\lg\lg k + 18)^k)$. Now at each node of the search tree the algorithm might need to find an almost shortest cycle, call **Clean()**, check if the graph is acyclic, and process the graph. Finding an almost shortest cycle takes $O(n^2)$ time. When **Clean()** is applied, since every vertex that **Clean()** removes has degree bounded by 2, **Clean()** can be implemented to run in linear time in the number of vertices it removes, and hence, in $O(n)$ time. Checking if the graph is acyclic and processing the graph takes no more than $O(n^2)$ time. It follows that the running time of the algorithm is $O((2\lg k + 2\lg\lg k + 18)^k n^2)$. □

According to the above theorem, the algorithm **FVS-solver** improves the algorithm in [23] by roughly a 2^k factor.

5 FVS On Special Classes of Graphs

In this section we consider the FVS problem on special classes of graphs. We look at the following classes: bipartite graphs, bounded genus graphs, and K^r-minor free graphs for fixed r.

Bipartite Graphs

Let G be a graph with n vertices and m edges. Consider the operation of subdividing an edge e in G by introducing a degree-2 vertex v. Then this operation is precisely the inverse operation of **Short-Cut(v)**. Therefore, if we let G' be the graph obtained from G by subdividing an edge $e \in G$, then it follows from Proposition 1 that $\gamma(G') = \gamma(G)$. Subdividing each edge in G yields a bipartite graph G', which according to the previous statement satisfies $\gamma(G') = \gamma(G)$. The graph G' has $n+m$ vertices and $2m$ edges. This shows that the FVS problem on general graphs can be solved in time $f(k)n^{O(1)}$ if and only if the FVS problem on bipartite graphs can be solved in $f(k)n^{O(1)}$ time. In particular, an algorithm of running time $c^k n^{O(1)}$ (for some constant c) for the FVS problem on bipartite graph implies an algorithm of running time $c^k n^{O(1)}$ for the FVS problem on general graphs.

Bounded Genus Graphs

The following lemma follows from a standard Euler-formula argument.

Lemma 8. *Let G be a graph with n vertices, minimum degree at least 3, and genus g. If $n \geq 8g$ then there is a cycle in G of length at most 12.*

Let G be a graph with n_0 vertices and genus g_0 satisfying $g_0 \leq c\lg n_0$ for some constant c. Note that if we branch on a cycle in G or process G as in the algorithm **FVS-solver**, the number of vertices and genus of G change, and the relationship $g \leq c\lg n$, where n and g are the number of vertices and genus, respectively, in the

resulting graph may not hold. However, since the operations in **FVS-solver** do not increase the genus of the graph, the genus g of the resulting graph satisfies $g \leq g_0 \leq c \lg n_0$. Consider the algorithm **BGFVS-solver** given in Figure 2, which is a modification of the algorithm **FVS-solver** presented in Section 1, that will solve the FVS problem on graphs of genus bounded by $c \lg n$.

BGFVS-solver

Input: an instance (G, k) of FVS where G has n_0 vertices and genus g_0
Output: a feedback vertex set F of G of size bounded by k in case it exists and g_0 satisfies $g_0 \leq c \lg n_0$

0. $F = \emptyset$;
1. **if** G is acyclic then **return**(F);
2. **if** $k = 0$ and G contains a cycle then **return** ('NO');
3. apply **Clean(G)**;
4. **if** the number of vertices n of G is bounded by $8c \lg n_0$ **then** solve the problem by brute force and **STOP**;
5. let C be an almost shortest cycle in G of length l;
6. **if** $l > 13$ **then return**('Invalid Instance');
   else branch on C and update k, F, and G accordingly;

Fig. 2. The algorithm BGFVS-solver

It is not difficult to see the correctness of the algorithm **BGFVS-solver**. The only step that needs clarification is when the algorithm returns in step 6 that the instance is invalid. If this happens, then the number of vertices n in the resulting graph G must satisfy $n > 8c \lg n_0$, where n_0 is the number of vertices in the original graph, and G does not have a cycle of length bounded by 12 (note that the algorithm finds an almost shortest cycle). Since the genus of the resulting graph g is bounded by the genus g_0 of the original graph, if g_0 satisfies the condition $g_0 \leq c \lg n_0$ then we would have $8g \leq 8g_0 \leq 8c \lg n_0 \leq n$, which according to Lemma 8, would imply the existence of a cycle of length at most 12 in G (since G has minimum degree at least 3 at this point). Since no such cycle exists in G, the genus g_0 in the original graph does not satisfy the condition $g_0 \leq c \lg n_0$, and hence the input instance does not satisfy the genus bound requirement. Therefore the algorithm rejects the instance in this case. This shows actually that **BGFVS-solver** does not need to know in advance if the input instance satisfies the genus bound requirement or not. As long as the input instance satisfies the genus bound requirement, the algorithm solves the problem.

Now to analyze the running time of the algorithm, we note that step 4 can be carried out in time $O(n_0^{8c+1})$ by enumerating every subset of vertices in the graph and checking whether it is a feedback vertex set of size bounded by k. The algorithm never branches on a cycle of length greater than 13. It follows that the running time of the algorithm is $O(13^k n_0^{8c+\omega+1})$.

Theorem 3. *The* FVS *problem on graphs of genus* $O(\lg n)$ *can be solved in time* $13^k n^{O(1)}$.

Let G be a graph with n_0 vertices and genus g_0, and let $\epsilon > 0$ be given. We construct a graph G' as follows. Let W be a wheel on $n_0^{2/\epsilon}$ vertices. Add W to G by linking a vertex in W other than the center to any vertex in G. It is easy to verify that $\gamma(G') = \gamma(G) + 2$, and that a feedback vertex set of G can be constructed from a feedback vertex set of G' easily. Since a wheel is planar, it follows that the genus g' of G is equal to g. The number of vertices n' of G' is $n' = n_0 + n_0^{2/\epsilon}$. Therefore $g' = g \leq n_0^2 \leq n'^{\epsilon}$. This shows that we can reduce the FVS problem on general graphs to the FVS problem on graphs of genus $O(n^\epsilon)$ (for any $\epsilon > 0$) in polynomial time such that if the FVS problem on graphs of genus $O(n^\epsilon)$ can be solved in $f(k)n^{O(1)}$ time, then so can the FVS problem on general graphs. In particular, if the FVS problem on graphs of genus $O(n^\epsilon)$ can be solved in time $c^k n^{O(1)}$ then so can the FVS problem on general graphs.[4]

K^r-Minor Free Graphs

Let r be a constant, and let $\mathcal{G}_r$ be the class of graphs with minimum degree at least 3 and no K^r-minor. Combining theorems from [20, 25] and [26] as described in [9, p180], we have the following: There exists a constant c such that, for all r, each $G \in \mathcal{G}_r$ has a cycle of length less than $cr\sqrt{\lg r}$.

Let G be a K^r-minor free graph with minimum degree at least 3. According to the result described above, there is a cycle in G of length bounded by $c_1 = cr\sqrt{\lg r}$, which is a constant. Therefore, if we modify the algorithm **FVS-solver** so that it branches on an almost shortest cycle after applying **Clean(G)**, then the algorithm always branches on a cycle of length at most $c' = 1 + max\{3, c_1\}$.[5] This shows that the running time of the modified algorithm is $O(c'^k n^2)$.

Theorem 4. *For any constant* r, *the* FVS *problem on* K^r*-minor free graphs can be solved in time* $O(c'^k n^2)$, *where* c' *is a constant.*

References

1. V. Bafna, P. Berman, and T. Fujito. A 2-approximation algorithm for the Undirected Feedback Vertex Set problem. *SIAM Journal on Discrete Mathematics*, 12(3):289–297, 1999.
2. Reuven Bar-Yehuda, Dan Geiger, Joseph Naor, and Ron M. Roth. Approximation algorithms for the feedback vertex set problem with applications to constraint satisfaction and bayesian inference. *SIAM Journal on Computing*, 27(4):942–959, 1998.

[4] Using the same technique, one can prove that if the FVS problem on graphs with average degree bounded by a constant, can be solved in time $f(k)n^{O(1)}$, then so can the FVS problem on general graphs.

[5] Note that if the length of the almost shortest cycle that the algorithm computes is not bounded by c', the algorithm rejects the instance.

3. A. Becker, R. Bar-Yehuda, and D. Geiger. Random algorithms for the Loop Cutset problem. *Journal of Artificial Intelligence Research*, 12:219–234, 2000.
4. H. Bodlaender. On disjoint cycles. *International Journal of Foundations of Computer Science*, 5:59–68, 1994.
5. B. Bollobás. *Extremal Graph Theory.* Academic Press, London, 1978.
6. J. A. Bondy and M. Simonovits. Cycles of even length in a graph. *J. Combinatorial Theory (B)*, 16:97–105, 1974.
7. J. Chen, I. Kanj, and W. Jia. Vertex cover: further observations and further improvements. *Journal of Algorithms*, 41:280–301, 2001.
8. D. Coppersmith and S. Winograd. Matrix multiplication via arithmetic progression. *Journal of Symbolic Computation*, 9:251–280, 1990.
9. R. Diestel. *Graph Theory (2nd Edition).* Springer-Verlag, New York, 2000.
10. R. Downey and M. Fellows. Fixed-parameter tractability and completeness. *Congressus Numerantium*, 87:161–187, 1992.
11. R. Downey and M. Fellows. *Parameterized Complexity.* Springer, New York, 1999.
12. P. Erdos and L. Posa. On the maximal number of disjoint circuits of a graph. *Pubbl. Math. Debrecen*, 9:3–12, 1962.
13. G. Even, J. Naor, B. Schieber, and M. Sudan. Approximating minimum feedback sets and multicuts in directed graphs. *Algorithmica*, 20(2):151–174, 1998.
14. R. J. Faudree and M. Simonovits. On a class of degenerate extremal graph problems. *Combinatorica*, 3 (1):83–93, 1983.
15. F. Fomin and D. Thilikos. Dominating sets in planar graphs: branch-width and exponential speed-up. In *Proceedings of the 14th ACM-SIAM Symposium on Discrete Algorithms*, pages 168–177, 2003.
16. T. Fujito. A note on approximation of the Vertex Cover and Feedback Vertex Set problems-unified approach. *Information Processing Letters*, 59(2):59–63, 1996.
17. M. Garey and D. Johnson. *Computers and Intractability: A Guide to the Theory of NP-Completeness.* W. H. Freeman, New York, 1979.
18. R. Gupta, R. Gupta, and M. Breuer. Ballast: A methodology for partial scan design. *IEEE Transactions on Computers*, 39(4):538–544, 1990.
19. A. Itai and M. Rodeh. Finding a minimum circuit in a graph. *SIAM Journal on Computing*, 7(4):413–423, 1978.
20. A. V. Kostochka. The minimum Hadwiger number for graphs with a given mean degree of vertices. *Metody Diskret. Analiz.*, 38:37–58, March 1982.
21. A. Kunzmann and H. Wunderlich. An analytical approach to the partial scan problem. *Journal of Electronic Testing: Theory and Applications*, 1:163–174, 1990.
22. V. Raman. Parameterized complexity. In *Proceedings of the 7th National Seminar on Theoretical Computer Science*, pages 1–18, 1997.
23. V. Raman, S. Saurabh, and C. Subramanian. Faster fixed-parameter tractable algorithms for Undirected Feedback Vertex Set. In *Proceedings of the 13th Annual International Symposium on Algorithms and Computation*, volume 2518 of *Lecture Notes in Computer Science*, pages 241–248, 2002.
24. P. Seymour. Packing directed circuits fractionally. *Combinatorica*, 15(2):281–288, 1995.
25. A. Thomason. An extremal function for contractions of graphs. *Math. Proc. Cambridge Philos. Soc.*, 95(2):261–265, 2001.
26. C. Thomassen. Girth in graphs. *J. Combin. Theory Ser. B*, 35:129–141, March 1983.