Theory Comput. Systems **41**, 479–492 (2007)
DOI: 10.1007/s00224-007-1345-z

Theory of Computing Systems

An $\mathcal{O}(2^{O(k)}n^3)$ FPT Algorithm for the Undirected Feedback Vertex Set Problem*

Frank Dehne,[1] Michael Fellows,[2] Michael Langston,[3] Frances Rosamond,[2] and Kim Stevens[4]

[1]Carleton University,
Ottawa, Ontario, Canada K1S 5B6
frank@dehne.net

[2]University of Newcastle,
Callaghan, NSW 2308, Australia
michael.fellows@newcastle.edu.au
frances.rosamond@newcastle.edu.au

[3]University of Tennessee,
Knoxville, TN 37996-3450, USA
langston@cs.utk.edu

[4]The Mechanics Institute,
Bob's Farm, NSW 2316, Australia
wonganellawines@telstra.com

Abstract. We describe an algorithm for the FEEDBACK VERTEX SET problem on undirected graphs, parameterized by the size k of the feedback vertex set, that runs in time $O(c^k n^3)$ where $c = 10.567$ and n is the number of vertices in the graph. The best previous algorithms were based on the method of bounded search trees, branching on short cycles. The best previous running time of an FPT algorithm for this problem, due to Raman, Saurabh and Subramanian, has a parameter function of the form $2^{O(k \log k/\log\log k)}$. Whether an exponentially linear in k FPT algorithm for

* This research has been supported in part by the U.S. National Science Foundation under Grant CCR-0075792, by the U.S. Office of Naval Research under Grant N00014-01-1-0608, by the U.S. Department of Energy under Contract DE-AC05-00OR22725, and by the Australian Research Council.

this problem is possible has been previously noted as a significant challenge. Our algorithm is based on the new FPT technique of iterative compression. Our result holds for a more general form of the problem, where a subset of the vertices may be marked as *forbidden* to belong to the feedback set. We also establish "exponential optimality" for our algorithm by proving that no FPT algorithm with a parameter function of the form $O(2^{o(k)})$ is possible, unless there is an unlikely collapse of parameterized complexity classes, namely FPT $= M[1]$.

1. Introduction

The FEEDBACK VERTEX SET problem for undirected graphs can be informally described as the problem of finding a set of vertices that "covers all the cycles" in the graph in the sense that every cycle in the graph includes at least one vertex of a solution set. We consider here a generalization of the problem, where a subset of the vertices of the input graph may be *forbidden* to belong to a solution set. This generalized form of the problem is formally defined as follows:

FEEDBACK VERTEX SET (FVS)

Instance:	An undirected multigraph $G = (V, E)$ (i.e., loops and multiple edges are allowed), a *forbidden* subset $U \subseteq V$ of vertices, and a positive integer k.
Parameter:	k
Question:	Is there a subset S of the vertices not in U, $S \subseteq V - U$, of size at most k, $\lvert S\rvert \leq k$, such that $G - S$ is acyclic?

In an instance of the problem as defined above, we will refer to the vertices of U as *forbidden* vertices, and we will refer to the vertices of $V - U$ as *unforbidden* or *normal* vertices. We will also refer to G as simply a *graph*, although loops and multiple edges are allowed. This generalized form of the FEEDBACK VERTEX SET problem was first considered by Bar-Yehuda et al. [BGNR].

The FEEDBACK VERTEX SET problem (in its usual, ungeneralized form) is NP-complete for both directed and undirected graphs [GJ]. There are numerous applications of the problem in areas such as circuit testing, deadlock resolution, analyzing manufacturing processes and in various contexts in computational biology [BGNR], [ENSS], [FHS], [FHP$^+$], [KW]. The minimization version of the problem is approximable within a factor of 2 in polynomial time [BBF]. See Festa et al., for a 1999 survey [FPR].

The FVS problem has been extensively studied from the parameterized point of view [BBG], [B], [DF1], [DF2], [KPS], [RSS1], [RSS2]. A parameterized problem is said to be *fixed-parameter tractable* (FPT) if it can be solved in time $f(k)n^c$ for some function f (unrestricted), where n is the total input size, k is the declared parameter and c is a constant independent of k and n. This running time may be written as $O^*(f(k))$

in the notation introduced by Woeginger [W] that focuses attention on the exponential-time costs due to the parameter and ignores the polynomial-time costs due to the overall input size. Highlights of previous research on the FVS problem in the parameterized framework include:

- A randomized FPT algorithm due to Becker et al. [BBG] running in time $O^*(4^k)$ finds a minimum feedback vertex set of size k with probability at least $1-(1-4^{-k})^{c4^k}$ for an arbitrary constant c.
- After several rounds of improvement, the best previous deterministic FPT algorithm, due to Raman et al. [RSS2], using some ideas from [RSS1] and [KPS], has a running time of $O^*(2^{O(k \lg k/\lg\lg k)})$. The basic idea for this and most previous algorithms is to branch on short cycles in a bounded search tree approach. See [DF2], [N1], and [N2] for a survey of this and other FPT techniques.

A number of problems concerning FVS have notably remained open:

(1) Is there an $O^*(2^{O(k)})$ FPT algorithm for FVS on undirected graphs?
(2) Is there a polynomial-time algorithm that kernelizes FVS on undirected graphs to a kernel of size polynomial in k? See [DF2], [N1], and [N2] for a discussion of kernelization and FPT.
(3) Is the FVS problem in FPT for directed graphs?

In this paper we answer the first of these significant open problems by an approach based on the relatively new technique of *iterative compression* [RSV], [DFRS], [M], [GGH$^+$]. As we prepare the final version of this paper, we have become aware that independently a solution to (1) has been described by Guo et al. [GGH$^+$], also based on iterative compression. Our algorithm differs in some details, and has a run-time analysis that is superior to the apparently slightly earlier solution to question (1) described in [GGH$^+$].

In the next section we provide a brief discussion of the iterative compression technique and its application to the FVS problem. In Section 3 we describe our FPT algorithm for the solution-compression form of the FVS problem. In Section 4 we prove an "optimality" result for our algorithm (giving a lower bound on the possibility of further qualitative improvements). In Section 5 we conclude with a review of open problems.

2. Iterative Compression Applied to FVS

The FPT technique of *iterative compression* seems first to have appeared in an FPT algorithm devised by Reed et al. for the problem of deleting k vertices to render a graph bipartite [RSV]. The approach was articulated as a general FPT design technique in [DFRS]. Some applications of the method can be found in [RSV], [DFRS], [M], and [GGH$^+$].

Here we use this approach to solve the FVS decision problem by recursively solving the following constructive *solution-compression* form of the problem:

SOLUTION COMPRESSION FOR FEEDBACK VERTEX SET

Instance: An undirected multigraph $G = (V, E)$
(loops and multiple edges are allowed),
a forbidden subset $U \subseteq V$ of vertices and
a *solution set* $S \subseteq V - U$ such that $G - S$ is acyclic, where $|S| = k + 1$.

Parameter: k

Output: Either: (1) a solution set S' of size k, or
(2) NO (i.e., no solution of size k is possible).

We employ an FPT algorithm for the above compression form of the FVS problem in the following way. We recursively solve a constructive form of the problem of deciding whether a graph $G = (V, E)$ admits a feedback vertex set of size k with vertices to be chosen from $V - U$. In this constructive form of the decision problem we are required either to produce a solution of size k, if one exists, or to return NO otherwise.

Given an instance $(G = (V, E), U \subseteq V, k)$, we recursively address the constructive decision problem for the instance $(G - v, U, k)$ where v is an arbitrarily chosen vertex in $V - U$. If this recursive call on $G - v$ returns NO, that is, no k-vertex solution for $G - v$ is possible, then clearly the correct answer for G is NO as well.

Alternatively, if the recursive call on the instance $(G - v, U, k)$ returns a k-element solution $S \subseteq V - U$, then $S \cup \{v\}$ is a solution of size $k + 1$ for G. We now employ as a subroutine the FPT algorithm for the solution compression problem. If $f(k)n^c$ is the running time for SOLUTION COMPRESSION FOR FVS, then our recursive solution to the constructive decision problem runs in time $f(k)n^{c+1}$, where n is the number of vertices in the graph G.

In the next section we describe our FPT algorithm for the problem of SOLUTION COMPRESSION FOR FVS.

3. An FPT Algorithm for FVS Solution Compression

We will use the following *reduction rules* that can be easily applied to simplify (or summarily decide) an instance of the problem. Recall that some vertices (the vertices in U in the problem definition) may be *forbidden* to belong to a solution set.

Rule 1: The Degree 1 Rule. If v is a vertex (forbidden or not) of degree 1 in G, then delete v. The parameter k is unchanged.

Rule 2: The Degree 2 Rule. If v is a vertex (forbidden or not) of degree 2 in G, with neighbors a and b (allowing possibly $a = b$), then modify G by replacing v and its two incident edges with a single edge between a and b (or a loop on $a = b$). The parameter k is unchanged.

Rule 3: Annotation Contraction. If u and v are adjacent forbidden vertices (that is, $u, v \in U$) then contract one of the edges between u and v. The parameter k is unchanged.

Rule 4: The Loop Rules. If there is a loop on an forbidden vertex v then answer NO. If there is a loop on an unforbidden vertex $v \in V - U$ then take v into the solution set, and reduce to the instance $(G - v, U, k - 1)$.

Rule 5: Multiedge Reduction. If there are more than two edges between u and v (forbidden or not) then delete all but two of these. The parameter k is unchanged.

Rule 6: Multiedge Selection. If there is an forbidden vertex u that is connected by two edges to an unforbidden vertex v, then take v into the solution set, that is, reduce to the instance $(G - v, U, k - 1)$.

The soundness of all these reduction rules is self-evident. In time $O(n)$ we can determine if any of the above reduction rules can be applied to a problem instance. Note that applications of the rules may cascade. We say that an instance is *reduced* if none of the reduction rules can be applied.

Note that if we reduce an instance (G, U, k) to an instance (G', U', k') by a series of applications of the above reduction rules, then given a solution S' of size k' for G', we can in time $O(n)$ recover a solution S of size k for G. We will always harmlessly assume that the instance we are working with is reduced.

Algorithm for SOLUTION COMPRESSION FOR FVS

Input: A reduced instance $(G = (V, E), U \subseteq V, k)$, and a solution $S \subseteq V - U$ of size $k + 1$.
Output: Either a solution of size at most k, or NO if none exists.

Step 1: Branch on all 2^{k+1} subsets of S. The branch corresponding to a subset $A \subseteq S$ represents the search for a size k solution S' that includes the vertices of A, that is, $A \subseteq S'$, and that does not include any of the vertices of $S - A = A'$.

Thus, in the instance (G', U', k') that represents this branch of Step 1:

(1) the vertices of A are deleted,
(2) the vertices of A' are forbidden,
(3) $k' = k - |A|$, and
(4) the instance is further reduced according to Reduction Rules 1–6.

We will argue below that for the reduced instance $(G' = (V', E'), U', k')$ considered on any of these 2^{k+1} branches of Step 1, we have either:

(i) $|V' - U'| \leq 4k$, or
(ii) we can immediately determine that the answer is NO.

Step 2: On each branch of Step 1, exhaustively analyze the resulting reduced instance by checking each k'-element subset of the unforbidden vertices to see if any provides a solution.

Step 2 requires checking at most $\binom{4k}{k}$ subsets. A simple bound on the running time of our algorithm is $O(c^k n^2)$ where $c = 18.963$, since

$$\binom{4k}{k} \approx (9.4815)^k$$

by Stirling's approximation of n factorial. A more refined version of our algorithm, detailed in Section 3.3, runs in time $O^*(10.567^k)$.

3.1. *The Reduced Instance Bound for Step* 1

The correctness of the algorithm is obvious because of its extreme simplicity. What is less obvious is the claimed bound of $4k$ on the number of unforbidden vertices in the reduced instance generated on a branch of Step 1 that need to be considered further.

Let $A \subseteq S$ and $A' = S - A$ as in the description of Step 1. The immediate instance graph G' on the A-branch of Step 1 consists of two sets of vertices:

(1) The (now) forbidden vertices of A', where we have the bound $|A'| \leq k + 1$.
(2) The other vertices, which we denote F. Some of these may be forbidden.

This immediate branch instance is further reduced, and this reduction process may result in some modification of the above picture. For example, connected components of the subgraph generated by A' would be contracted to a single vertex, by repeated applications of Rule 3. To simplify the argument, we will assume that the immediate branch instance is already reduced so that our description of the vertices of G' as partitioned into A' and F is accurate (these sets would be modified by further reduction, but a bipartition with the same properties we make use of below would result in any case). The following structural claims hold.

Lemma 1. *The subgraph $\langle F \rangle$ induced by F is acyclic.*

Proof. Otherwise S would not be a solution for G. □

Henceforth we may use F (for convenience) to denote also the forest induced by the vertices in the vertex set F.

Lemma 2. *Each leaf l of the forest F is adjacent to at least two distinct vertices in A'.*

Proof. In view of Lemma 1 and Reduction Rules 1 and 2, there must be at least two edges connecting l to vertices in A'. Reduction Rule 6 would apply if l were connected to only one vertex of A'. □

The vertices in the forest F can be partitioned into three sets. Let L denote the leaves of F, let J be the vertices that have degree 2 in the forest subgraph $\langle F \rangle$. We will refer to the vertices of J as the *subdivision vertices* of F. Let B, the *branch vertices* of F, be the vertices of degree at least 3 in the subgraph $\langle F \rangle$.

Lemma 3. *Each vertex $j \in J$ is connected to at least one vertex of A'.*

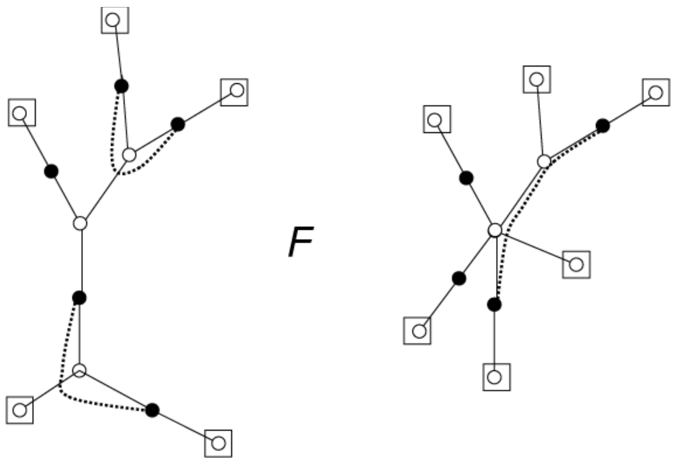

Fig. 1. A maximum path-matching of the subdivision vertices ("J vertices") of the forest F, showing that $\pi(F) = 11 + 3 = 14$.

Proof. Otherwise, in view of Lemma 1, Reduction Rule 2 would apply. □

Definition 1. Let F be a forest with the vertex set partitioned into the three sets: (1) the leaves L, (2) the subdivision vertices J and (3) the branch vertices B of F. A *path-matching* of the J-vertices of F of size r consists of:

(1) r mutually disjoint 2-element subsets $\{x_i, y_i\} \subseteq J$, $1 \leq i \leq r$,
(2) for each i, $1 \leq i \leq r$, a path ρ_i in F from x_i to y_i, subject to the requirement that for $i \neq j$, the paths ρ_i and ρ_j are vertex disjoint.

Definition 2. The *potential* $\pi(F)$ of the forest F is defined to be the sum of the number of leaves $|L|$ of F and the size of a maximum path-matching of the J-vertices. (See Figure 1 for an example.)

Lemma 4. *Suppose that for the reduced instance (G', U', k') with vertex set partitioned into A' and F as above we have $\pi(F) \geq k' + |A'|$. Then the answer for this instance is NO.*

Proof. Since $\pi(F) \geq k' + |A'|$, there must be a set $\mathcal{P}$ of at least $k' + |A'|$ paths in G' that begin and end at vertices of A', and that are pairwise internally vertex-disjoint. This is ensured by the definition of $\pi(F)$ and by Lemmas 2 and 3. If (G', U', k') were a YES-instance, then there would be a feedback vertex set S' consisting of at most k' unforbidden vertices. In particular, S' could not contain any vertices of A'. Let $\mathcal{P}'$ denote the subset of paths in $\mathcal{P}$ that are disjoint from S'. Since the paths in $\mathcal{P}$ are pairwise internally vertex disjoint, so that, intuitively speaking, any vertex of S' can hit at most one of them, $|\mathcal{P}'| \geq |A'|$. However, here we reach a contradiction, since the vertices of A' together with the paths in $\mathcal{P}'$ must form a cycle disjoint from S'. □

Lemma 5. *For any forest F on m vertices*, $\pi(F) \geq (m+1)/2$.

The proof of Lemma 5 is somewhat involved, and we defer the discussion to the next subsection.

Lemma 6. *If on the branch of Step* 1 *corresponding to $A \subseteq S$ we have a reduced instance (G', U', k') where the vertices of G' are partitioned into A' and F as in the discussion above*, *and where* $|F| \geq 4k+1$, *then this is a NO-instance*.

Proof. By Lemma 5, $\pi(F) \geq 2k+1$. The rest follows by Lemma 4, since $|A'| \leq k+1$ and $k' \leq k$. □

3.2. *The Proof of Lemma* 5

Lemma 5 states that any forest F on m vertices has potential $\pi(F) \geq (m+1)/2$.

Proof. There are two parts to the argument:

(1) We prove the lemma for trees of maximum degree 3. The proof is by structural induction.
(2) We then prove the lemma for arbitrary trees by minimum counterexample, using (1) essentially as the base case. The lemma for arbitrary forests follows almost trivially.

As it is simpler, we treat the second step first, assuming (1) for the moment. Let T be a counterexample tree having a minimum number of vertices, $|T| = m$. By (1), T must have at least one vertex v of degree 4 or more. We consider *breaking* T into two trees T_1 and T_2 as illustrated in Figure 2. The vertex v is "broken" into two copies by choosing an incident edge e and "detaching" T_1 as the subtree joined to the rest of T at v by the edge e, and by making one of the copies of v a leaf in T_1. The tree T_2 consists

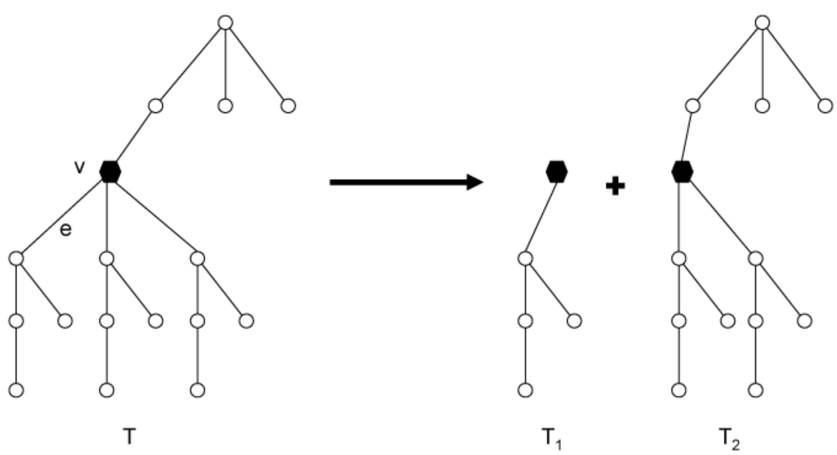

Fig. 2. Breaking T into T_1 and T_2 at v.

of T with e and the subtree (T_1) attached by e removed. Thus in T_2, the degree of (the other copy of) v is decreased by 1, and we have $|T_1| + |T_2| = |T| + 1$. See Figure 2.

Let $m_i = |T_i|$ for $i = 1, 2$. Thus $m_1 + m_2 = m + 1$. The lemma must hold for each of the trees T_i, since T is presumed to be a minimum counterexample. Therefore $\pi(T_i) = (m_i + 1)/2$ for $i = 1, 2$. Choose suitable path-matchings of the sets of subdivision vertices J_i in the T_i that witness this. Combining these witness structures in T ("putting T back together") gives

$$\pi(T) \geq (m_1 + 1)/2 + (m_2 + 1)/2 - 1$$

with the -1 term because a leaf is lost when the two copies of v are fused back together. (Note that the copy of v in T_2 has degree at least 3 in T_2 and therefore does not belong to J_2, so that there are no other losses in combining the two witness structures.) This gives

$$\pi(T) \geq (m + 3)/2 - 1 = (m + 1)/2$$

and the lemma is proved, assuming (1).

To prove the lemma for trees of maximum degree 3, we induct on the structure of such trees. Each such tree T is considered to be rooted at a vertex r, where either: (1) r is a leaf of T, or (2) r has degree 2 in T. We will refer to (1) and (2) as the *types* of the rooted trees we discuss.

Trees of maximum degree 3 are generated by two operations on these rooted trees:

(i) A unary operation $x(T)$ (*extension* of T) that can be applied to rooted trees of type either (1) or (2) and that consists in adding a new vertex r' connected to r, with r' becoming the root of the resulting "extended" tree.

(ii) A binary operation $T_1 \oplus T_2$ (*join* of T_1 and T_2) that applies only when both T_1 and T_2 are of type (1), that is, have roots of degree 1. In this operation, the roots of the two trees are identified, resulting in a rooted tree of type (2).

The two operations are illustrated in Figure 3.

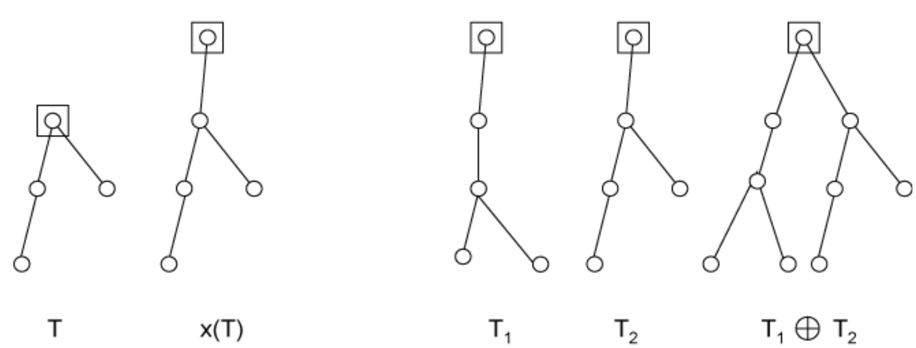

Fig. 3. The parsing operations for trees of maximum degree 3.

An elementary induction shows that all trees of maximum degree 3 can be parsed in terms of these two operations on (smaller) rooted trees. For a rooted tree of type (1) or type (2) our induction hypothesis is as follows. Here we consider that the vertices of T are partitioned into the four sets: $\{r\}$, L, J and B, of the root, the leaves, the subdivision vertices and the branch vertices, respectively, as in earlier discussions, but with the exception of the root. In particular, here we do not consider that the root belongs to J, even for rooted trees of type (2).

Induction Hypothesis. One of the following claims holds:

(1) $|J|$ is even and the J-vertices of T admit a perfect path-matching in the sense defining $\pi(T)$.
(2) $|J|$ is odd and the J-vertices can be path-matched in T with the exception of one vertex $u \in J$, and furthermore the path-matching can be accomplished so that there is a path from u to the root r that is disjoint from the paths in T that realize the path-matching of J.

The induction hypothesis is illustrated in Figure 4.

It is straightforward to verify the several cases of the induction step for the two parsing operations. For example, we can verify that for the operation $T_1 \oplus T_2$ where both T_1 and T_2 satisfy case (2) of the induction hypothesis with non-path-matched J-vertices, respectively, u_1 and u_2, the outcome T of the operation satisfies case (1) of the induction hypothesis. In this outcome, the paths from the non-path-matched vertices u_i to the joined roots r_i are combined to form a path matching u_1 to u_2 in T. We leave the other cases to the reader.

By the above inductive argument, it follows that there can be at most one unmatched J-vertex in a maximum path-matching of J in an (unrooted) tree T of maximum degree 3, where the vertices of T are partitioned into the three sets L, J and B. Since $|B| = |L| - 2$ and therefore

$$m = |L| + |J| + |B| = 2|L| + |J| - 2$$

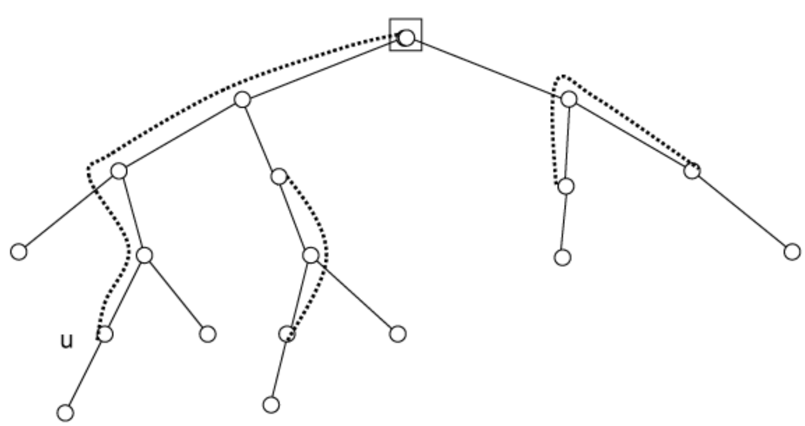

Fig. 4. An example of the induction hypothesis for $|J|$ odd.

we have

$$\pi(T) \geq |L| + (|J| - 1)/2 = (m + 1)/2,$$

which proves the lemma for trees of maximum degree at most 3. □

3.3. *A More Efficient Version*

Lemma 4 shows that there is a simple way to improve the efficiency of our algorithm. On the branch of Step 1 corresponding to a subset A of the $(k + 1)$-sized solution S, we can answer NO if for the reduced instance we have $\pi(F) \geq k' + |A'|$. Since $k' = k - |A|$ and $|A'| = k + 1 - |A|$, and using Lemma 5, the total bound on the number of possible solutions explored in Steps 1 and 2 is

$$\sum_{i=0}^{k} \binom{k+1}{i} \binom{2((k+1-i) + (k-i) - 1) - 1}{k-i} = \sum_{i=0}^{k} \binom{k+1}{i} \binom{4k - 4i - 1}{k-i}.$$

Define

$$f(x, k) = \binom{k}{x} \binom{4(k-x)}{k-x}$$

and suppose $f(x, k)$ is maximized for $x^* = x(k)$. Then our sum above is bounded by $(k + 1) \cdot f(x^*, k + 1)$.

We next work out two estimates $x_1(k)$ and $x_2(k)$ such that

$$x_1(k) \leq x^*(k) \leq x_2(k)$$

and we will therefore have a bound on our sum of

$$(k+1) \cdot \binom{k+1}{x_2(k+1)} \binom{4((k+1) - x_1(k+1))}{(k+1) - x_1(k+1)}.$$

(The reason for the two estimates is that the first part of $f(x, k)$ increases with x, and the second part decreases with x.)

We study the ratio $f(x, k)/f(x+1, k)$. The maximizing value x^* is located (essentially) at the point where this ratio is equal to 1. Considered as real functions, the partial derivative of $f'(x, k) = f(x, k)/f(x + 1, k)$ with respect to x is positive in the range $[0, k)$. Hence, $f'(x, k)$ is an increasing function of x in that range. It follows that, over the reals, there is a unique x^* such that $f'(x^*, k) = 1$.

Assuming that k is large, the ratio is approximately:

$$\frac{f(x, k)}{f(x+1, k)} \approx \left(\frac{x+1}{k-x}\right)(4)\left(\frac{4}{3}\right)^3.$$

This yields the estimates:

$$x_1(k) = (27/283)k \quad \text{and} \quad x_2(k) = (28/283)k.$$

Using the bound (based on Stirling's approximation) that

$$\binom{ak}{bk} \leq \left(\frac{a^a}{b^b(a-b)^{a-b}}\right)^k$$

for constants $a > b$, we obtain the bound on our total cost sum of $(k + 1)(10.567)^k$.

4. Optimality

Our FPT algorithm for the problem of SOLUTION COMPRESSION FOR FVS yields, by the approach of Section 2, an FPT algorithm for the parameterized FEEDBACK VERTEX SET problem that runs in time $O(c^k n^3)$ where $c = 10.567$. In qualitative terms, we have given an algorithm with a running time of the form $O^*(2^{O(k)})$. We next show that this is, in a qualitative sense, "optimal" for the problem.

Theorem 1. *There can be no FPT algorithm for* FEEDBACK VERTEX SET *with a running time of the form* $O^*(2^{o(k)})$ *unless* $FPT = M[1]$.

Proof. The standard NP-completeness reduction from VERTEX COVER to FEEDBACK VERTEX SET does not change the value of the parameter. Hence, if FEEDBACK VERTEX SET has a $2^{o(k)} p(n)$ algorithm, for some polynomial p, then the same holds for VERTEX COVER. The existence of such an algorithm for VERTEX COVER implies FPT $= M[1]$ and has other consequences considered unlikely [IPZ], [CJ], [DEF$^+$], [CF]. □

Remark 1. The consequence FPT $= M[1]$ is highly unlikely, since it is known that FPT $= M[1]$ if and only if satisfiability of 3SAT instances on n variables can be decided in time $O^*(2^{o(n)})$. (See [DEF$^+$] and [CF] for further information and discussion.)

Remark 2. A number of other FPT optimality results have been shown for various problems [DFR] and [CJ]. A notable example is the parameterized PLANAR DOMINATING SET problem, for which there is an FPT algorithm with a running time of $O^*(2^{O(\sqrt{k})})$ [ABF$^+$]. It has been shown that there can be no FPT algorithm for this problem with a running time of the form $O^*(2^{o(\sqrt{k})})$ unless FPT $= M[1]$ [CJ].

5. Open Problems

There are two compelling problems concerning FVS that remain unresolved.

- Is the FEEDBACK VERTEX SET problem for directed graphs in FPT? This is currently an open problem even for the restriction to planar digraphs.
- Is there a polynomial-time kernelization algorithm for FVS on undirected graphs that reduces an instance (G, k) to (G', k') where $k' \leq k$ and the size of G' is bounded by a polynomial in k?

Perhaps an iterative compression approach similar to the one employed in our main result here might be of use in addressing the FVS problem for digraphs.

The potential practical significance of our algorithm should also be investigated. Our approach to the FVS problem here is a new one. The "flat" parallelism of Step 1 (where there are many branches of the algorithm created "all at once," as contrasted with many branches created by repeated binary branching, as is more typically the case for FPT algorithms) could conceivably be significant for highly parallel implementations.

The reduction rules that we have employed here are all local and elementary in character. It could be productive to explore if global "crown type" reduction rules for the problem might be possible, as in the case of VERTEX COVER and other problems [ACF+], [CFJ], [DFRS]. Such reduction rules might be of use in addressing the very natural open problem concerning polynomial-size kernelization. Alternatively, perhaps some new lower bound techniques such as those recently developed in [CFKX] can be used to show that no polynomial-size many-one kernelization for FVS is likely to be possible.

References

[ABF+] J. Alber, H. L. Bodlaender, H. Fernau, T. Kloks and R. Niedermeier. Fixed parameter algorithms for Dominating Set and related problems on planar graphs. *Algorithmica* **33** (2002), 461–493.

[ACF+] F. N. Abu-Khzam, R. L. Collins, M. R. Fellows, M. A. Langston, W. H. Suters and C. T. Symons. Kernelization algorithms for the vertex cover problem: theory and experiments. *Proceedings of the* 6*th Workshop on Algorithm Engineering and Experiments* (*ALENEX*), New Orleans, January, 2004, pp. 62–69. ACM/SIAM, *Proc. Applied Mathematics* 115, L. Arge, G. Italiano and R. Sedgewick, eds.

[B] H. Bodlaender. On disjoint cycles. *International Journal of Foundations of Computer Science* **5** (1994), 59–68.

[BBF] V. Bafna, P. Berman and T. Fujito. A 2-approximation algorithm for the undirected feedback vertex set problem. *SIAM Journal on Discrete Mathematics* **12** (1999), 289–297.

[BBG] A. Becker, R. Bar-Yehuda and D. Geiger. Random algorithms for the loop cutset problem. *Journal of Artificial Intelligence Research* **12** (2000), 219–234.

[BGNR] R. Bar-Yehuda, D. Geiger, J. Naor and R. Roth. Approximation algorithms for the feedback vertex set problem with applications to constraint satisfaction and Bayesian inference. *SIAM Journal on Computing* **27** (1998), 942–959.

[CF] Y. Chen and J. Flum. On miniaturized problems in parameterized complexity theory. *Proceedings of the First International Workshop on Parameterized and Exact Computation*, pp. 108–120. Lecture Notes in Computer Science, vol. 3162. Springer-Verlag, Berlin, 2004.

[CFJ] B. Chor, M. Fellows and D. Juedes. Linear kernels in linear time, or how to save k colors in $O(n^2)$ steps. *Proceedings WG* 2004, pp. 257–269. Lecture Notes in Computer Science, vol. 3353. Springer-Verlag, Berlin, 2004.

[CFKX] J. Chen, H. Fernau, I. Kanj and G. Xia. Parametric duality and kernelization: lower bounds and upper bounds on kernel size. *Proceedings of the* 22*nd Symposium on Theoretical Aspects on Computer Science* (*STACS* 2005), pp. 269–280. Lecture Notes in Computer Science, vol. 3404. Springer-Verlag, Berlin, 2005.

[CJ] L. Cai and D. Juedes. On the existence of subexponential parameterized algorithms. *Journal of Computer and System Sciences* **67** (2003), 789–807.

[DEF+] R. Downey, V. Estivill-Castro, M. Fellows, E. Prieto-Rodriguez and F. Rosamond. Cutting up is hard to do: the complexity of k-cut and related problems. *Electronic Notes in Theoretical Computer Science* **78** (2003), 205–218.

[DF1] R. Downey and M. Fellows. Fixed-parameter tractability and completeness. *Congressus Numerantium* **87** (1992), 161–187.

[DF2] R. G. Downey and M. R. Fellows. *Parameterized Complexity*. Springer-Verlag, New York, 1999.

[DFR] F. Dehne, M. Fellows and F. Rosamond. An FPT algorithm for set splitting. *Proceedings of the* 29*th Workshop on Graph Theoretic Concepts in Computer Science* (*WG* 2003), pp. 180–191. Lecture Notes in Computer Science, vol. 2880. Springer-Verlag, Berlin, 2003.

[DFRS] F. Dehne, M. Fellows, F. Rosamond and P. Shaw. Greedy localization, iterative compression and modeled crown reductions: new FPT techniques, an improved algorithm for set splitting and a novel 2k kernelization for vertex cover. *Proceedings of the First International Workshop on Parameterized and Exact Computation*, pp. 271–280. Lecture Notes in Computer Science, vol. 3162. Springer-Verlag, Berlin, 2004.

[ENSS] G. Even, J. Naor, B. Scheiber and M. Sudan. Approximating minimum feedback sets and multicuts in directed graphs. *Algorithmica* **20** (1998), 151–174.

[FHP+] C. Fried, W. Hordijk, S. J. Prohaska, C. R. Stadler and P. F. Stadler. The footprint sorting problem. *Journal of Chemical Information and Computer Science* **44** (2004), 332–338.

[FHS] M. Fellows, M. Hallett and U. Stege. Analogs and duals of the MAST problem for sequences and trees. *Journal of Algorithms* **49** (2003), 192–216.

[FPR] P. Festa, P. M. Pardalos and M. G. C. Resende. Feedback set problems. In D. Z. Du, P. M. Pardalos (eds.), *Handbook of Combinatorial Optimization*, Vol. A, pp. 209–258. Kluwer, Dordrecht, 1999.

[GGH+] J. Guo, J. Gramm, F. Hueffner, R. Niedermeier and S. Wernicke. Improved fixed-parameter algorithms for two feedback set problems. *Proceedings of WADS* 2005, pp. 158–169. Lecture Notes in Computer Science, vol. 3608. Springer-Verlag, Berlin, 2005.

[GJ] M. R. Garey and D. S. Johnson. *Computers and Intractability*: *A Guide to the Theory of NP-Completeness*. Freeman, San Francisco, CA, 1979.

[IPZ] R. Impagliazzo, R. Paturi and F. Zane. Which problems have strongly exponential complexity? *Journal of Computer and System Sciences* **63** (2001), 512–530.

[KPS] I. Kanj, M. Pelsmajer and M. Schaefer. Parameterized algorithms for feedback vertex set. *Proceedings of the First International Workshop on Parameterized and Exact Computation*, pp. 235–247. Lecture Notes in Computer Science, vol. 3162. Springer-Verlag, Berlin, 2004.

[KW] A. Kunzmann and H. Wunderlich. An analytical approach to the partial scan problem. *Journal of Electronic Testing*: *Theory and Applications* **1** (1990), 163–174.

[M] D. Marx. Chordal deletion is fixed-parameter tractable. Manuscript, 2004.

[N1] R. Niedermeier. Invitation to Fixed-Parameter Algorithms. Habilitationschrift, University of Tubingen, 2002. (Electronic file available from R. Niedermeier.)

[N2] R. Niedermeier. *Invitation to Fixed Parameter Algorithms*. Oxford University Press, Oxford, 2006.

[RSS1] V. Raman, S. Saurabh and C. Subramanian. Faster fixed-parameter tractable algorithms for undirected feedback vertex set. In *Proceedings of the* 13*th Annual International Symposium on Algorithms and Computation*, pp. 241–248. Lecture Notes in Computer Science, vol. 2518. Springer-Verlag, Berlin, 2002.

[RSS2] V. Raman, S. Saurabh and C.R. Subramanian. Faster algorithms for feedback vertex set. In *Proceedings of the* 2*nd Brazilian Symposium on Graphs*, *Algorithms and Combinatorics*, *GRACO* 2005, April 27–29, 2005, Angra dos Reis (Rio de Janeiro), Brazil. Electronic Notes in Discrete Mathematics. Elsevier, Amsterdam, 2005.

[RSV] B. Reed, K. Smith, and A. Vetta. Finding odd cycle transversals. *Operations Research Letters* **32** (2004), 299–301.

[W] G. J. Woeginger. Exact algorithms for NP-hard problems: a survey. *Proceedings of* 5*th International Workshop on Combinatorial Optimization—Eureka*, *You Shrink*! *Papers dedicated to Jack Edmonds*, M. Junger, G. Reinelt, and G. Rinaldi (Festschrift eds.), pp. 184–207. Lecture Notes in Computer Science, vol. 2570. Springer-Verlag, Berlin, 2003.

Received November 12, 2005, *and in final form February* 28, 2006. *Online publication August* 14, 2007.