# An $O(2^{O(k)}n^3)$ 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] Griffith University, Brisbane QLD 4111, Australia
`frank@dehne.net`
[2] University of Newcastle, Callaghan NSW 2308, Australia
{`mfellows,fran`}`@cs.newcastle.edu.au`
[3] University of Tennessee, Knoxville TN 37996-3450, U.S.A.
`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 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 "annotated" form of the problem, where a subset of the vertices may be marked as *not* 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 a 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 the vertices of the input graph may be annotated according to whether or not they

[⋆] 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, by the Australian Research Council and by the Australian Centre for Bioinformatics.

are allowed to belong to a solution set. This generalized form of the problem is formally defined as follows:

FEEDBACK VERTEX SET (FVS)
*Instance*: An undirected graph $G = (V, E)$
(loops and multiple edges are allowed),
an *annotated* 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 $k$, $|S| \leq k$, such that $G - S$ is acyclic?

The FEEDBACK VERTEX SET problem is NP-complete for both directed and undirected graphs [GJ79]. There are numerous applications of the problem in areas such as circuit testing, deadlock resolution, analyzing manufacturing processes and computational biology [BGNR98,ENSS98,FHS03,FHPSS04,KW90]. The minimization version of the problem is approximable within a factor of 2 in polynomial time [BBF99].

The FVS problem has been extensively studied from the parameterized point of view [BBG00,Bod94,DF92,DF99,KPS04,RSS02,RSS05]. 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 [Woe03] 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. [BBG00] 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, Saurabh and Subramanian [RSS05], refining some ideas from [RSS02] and [KPS04], 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 [DF99,Nie02,Nie05] for surveys 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 [DF99,Nie02,Nie05] 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* [RSV04,DFRS04,Ma04,GGHNW05]. 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. [GGHNW05], 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 the upcoming conference paper [GGHNW05].

In the next section we provide a brief discussion of the iterative compression technique and its application to the FVS problem. In §3 we describe our FPT algorithm for the solution-compression form of the FVS problem. In §4 we prove an "optimality" result for our algorithm (giving a lower bound on the possibility of further qualitative improvements). In §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, Smith and Vetta for the problem of deleting $k$ vertices to render a graph bipartite [RSV04]. The approach was articulated as a general FPT design technique in [DFRS04]. Some applications of the method can be found in [RSV04,DFRS04,Ma04,GGHNW05].

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 graph $G = (V, E)$
(loops and multiple edges are allowed),
an annotated subset $U \subseteq V$ of vertices,
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 $G$.

## 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 annotated as *not* to belong to a solution set.

**Rule 1: The Degree One Rule.** If $v$ is a vertex (annotated or not) of degree 1 in $G$, then delete $v$ and adjust the rest of the input data accordingly.

**Rule 2: The Degree Two Rule.** If $v$ is a vertex (annotated 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 $a = b$) and adjust the rest of the input data accordingly.

**Rule 3: Annotation Contraction.** If $u$ and $v$ are adjacent annotated vertices (that is, $u, v \in U$) then contract one of the edges between $u$ and $v$ and adjust the rest of the input data accordingly.

**Rule 4: The Loop Rules.** If there is a loop on an annotated vertex $v$ then answer NO. If there is a loop on an unannotated 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 $v$ (annotated or not) then delete all but two of these.

**Rule 6: Multiedge Selection.** If there is an annotated vertex $u$ that is connected by two edges to an unannotated 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 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 $2^{k+1} - 1$ subsets of $S$ of size at most $k$. 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 annotated,
(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 the 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 unannotated 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

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

A more refined version of our algorithm, detailed in §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 unannotated 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) annotated 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 annotated.

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 $A'$.*

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

**Definition 1.** *Let $F$ be a forest with 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 $\rho_i$ in $F$ from $x_i$ to $y_i$, subject to the requirement that for $i \neq j$, the paths $\rho_i$ and $\rho_j$ 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.)*

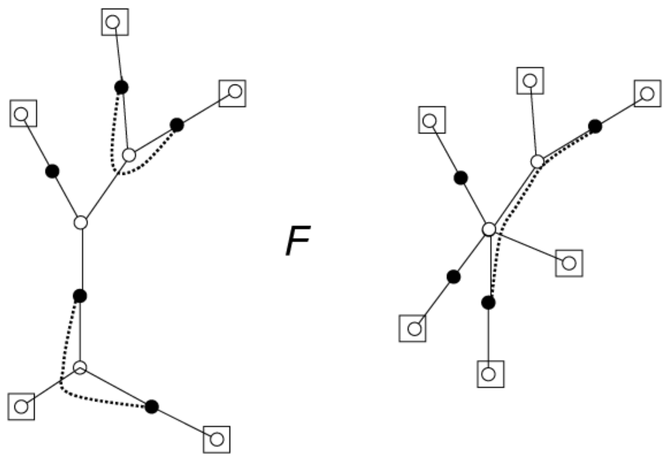

**Fig. 1.** A maximum path-matching of the subdivision vertices ("$J$ vertices") of the forest $F$, showing that $\pi(F) = 11 + 3 = 14$.

**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.* If it were a YES-instance (for $k'$) then there would be a feedback vertex set $S'$ consisting of at most $k'$ unannotated vertices. But then there would necessarily be at least $|A'|$ leaves and $J-matching$ paths $\rho_i$ in $F$ having empty intersection with $S'$. Since $S' \cap A' = \emptyset$ (because the vertices of $A'$ are annotated), there are at least $|A'|$ *virtual edges* or *virtual loops* connecting the vertices of $A'$ through $F - S'$. (For example, if a leaf $l$ of $F$ is not in $S'$, then by Lemma 2 it is adjacent to two vertices $a$ and $b$ in $A'$, which we consider here as a *virtual edge* between $a$ and $b$. If the path $\rho_i$ in $F$ from the $J$-vertex $x_i$ to the $J$-vertex $y_i$ does not contain any vertices in $S'$, then together with the connections of $x_i$ and $y_i$ to the set $A'$ guaranteed by Lemma 3, we have what can be considered either a *virtual edge* between $A'$ vertices — or a *virtual loop*, in case the $A'$-adjacencies guaranteed for $x_i$ and $y_i$ by Lemma 3 connect these vertices to the *same* vertex of $A'$.) Joining the vertices of $A'$ by $|A'|$ virtual edges or virtual loops necessarily implies that there is a cycle not including any vertices of $S'$, that is, that $S'$ is not a feedback vertex set, a contradiction.

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

The proof of Lemma 5 is intricate, and can be found in the full paper.

**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 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. Assuming that $k$ is large, this ratio is approximately:

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

This yields the estimates:
$x_1(k) = (27/283)k$ and
$x_2(k) = (28/283)k$.

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 §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.* Determining whether a graph on $n$ vertices has a vertex cover of size at most $k \log n$, where the parameter is $k$, is termed the $k \log n$ Vertex Cover Problem. This "renormalized" form of the well-known FPT Vertex Cover problem is complete for the parameterized complexity class $M[1]$ [DEFPR03,CF04]. The theorem follows because there is a linear-size and parameter-preserving (i.e., $k' = k$) polynomial-time reduction from Vertex Cover to Feedback Vertex Set, by simply replacing each edge of the Vertex Cover instance with a pair of parallel edges. Thus if there were an FPT algorithm for Feedback Vertex Set running in time $O^*(2^{o(s)})$ where $s$ is the size of the feedback vertex set, then we would have an algorithm for the $k \log n$ Vertex Cover Problem running in time $O^*(2^{o(k \log n)})$, but as shown in [CJ03], this is an FPT running time. By the completeness of the $k \log n$ Vertex Cover Problem for $M[1]$ we would have FPT $= M[1]$.

*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 [DEFPR03,CF04] for further information and discussion.)

*Remark 2.* A number of other FPT optimality results have been shown for various problems [DFR03,CJ03]. A notable example is the parameterized Planar Dominating Set problem, for which there is an FPT algorithm with a running time of $O^*(2^{O(\sqrt{k})})$ [ABFKN02]. 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 $= M[1]$ [CJ03].

## 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 open 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 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 has turned out to be usefully the case for Vertex Cover [ACFLSS04]. Such reduction rules could be important for addressing the very natural open problem concerning polynomial-size kernelization. Alternatively, perhaps some new lower bound techniques, such as those recently developed in [CFKX05], can be used to show that no polynomial-time polynomial-size many:1 kernelization for FVS is possible.

## References

[ABFKN02] 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.

[ACFLSS04] 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 6th Workshop on Algorithm Engineering and Experiments (ALENEX)*, New Orleans, January, 2004, ACM/SIAM, *Proc. Applied Mathematics 115*, L. Arge, G. Italiano and R. Sedgewick, eds.

[BBF99] 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.

[BBG00] A. Becker, R. Bar-Yehuda and D. Geiger. Random algorithms for the loop cutset problem. *Journal of Artificial Intelligence Research* 12 (2000), 219–234.

[BGNR98] 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.

[Bod94] H. Bodlaender. On disjoint cycles. *International Journal of Foundations of Computer Science* 5 (1994), 59–68.

[CF04] Y. Chen and J. Flum. On miniaturized problems in parameterized complexity theory. *Proceedings of the First International Workshop on Parameterized and Exact Computation*, Springer-Verlag, *Lecture Notes in Computer Science* vol. 3162 (2004), 108–120.

[CJ03] L. Cai and D. Juedes. On the existence of subexponential parameterized algorithms. *Journal of Computer and System Sciences* 67 (2003), 789–807.

[CFKX05] J. Chen, H. Fernau, I. Kanj and G. Xia. Parametric duality and kernelization: lower bounds and upper bounds on kernel size. *The 22nd Symposium on Theoretical Aspects on Computer Science* (STACS 2005), Springer-Verlag, *Lecture Notes in Computer Science* vol. 3404 (2005), 269-280.

[DEFPR03] 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.

[DF92] R. Downey and M. Fellows. Fixed-parameter tractability and completeness. *Congressus Numerantium* 87 (1992), 161–187.

[DF99] R. G. Downey and M. R. Fellows. *Parameterized Complexity.* Springer-Verlag, 1999.

[DFR03] F. Dehne, M. Fellows and F. Rosamond. An FPT algorithm for set splitting. *Proceedings of the 29th Workshop on Graph Theoretic Concepts in Computer Science (WG 2003)*, Springer-Verlag, *Lecture Notes in Computer Science* 2880 (2003), 180–191.

[DFRS04] 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*, Springer-Verlag, *Lecture Notes in Computer Science* vol. 3162 (2004), 271–280.

[ENSS98] G. Even, J. Naor, B. Scheiber and M. Sudan. Approximating minimum feedback sets and multicuts in directed graphs. *Algorithmica* 20 (1998), 151–174.

[FHPSS04] C. Fried, W. Hordijk, S.J. Prohaska, C.R. Stadler and P.F. Stadler. The footprint sorting problem. *J. Chem. Inf. Comput. Sci.* 44 (2004), 332 -338.

[FHS03] 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.

[GGHNW05] J. Guo, J. Gramm, F. Hueffner, R. Niedermeier, S. Wernicke. Improved fixed-parameter algorithms for two feedback set problems. *Proceedings of WADS 2005*, Springer-Verlag, Lecture Notes in Computer Science (2005), to appear.

[GJ79] M. R. Garey and D. S. Johnson. *Computers and Intractability: A Guide to the Theory of NP-Completeness.* W.H. Freeman, 1979.

[KPS04] I. Kanj, M. Pelsmajer and M. Schaefer. Parameterized algorithms for feedback vertex set. *Proceedings of the First International Workshop on Parameterized and Exact Computation*, Springer-Verlag, *Lecture Notes in Computer Science* vol. 3162 (2004), 235–247.

[KW90] A. Kunzmann and H. Wunderlich. An analytical approach to the partial scan problem. *Journal of Electronic Testing: Theory and Applications* 1 (1990), 163–174.

[Ma04] D. Marx. Chordal deletion is fixed-parameter tractable. Manuscript, 2004.

[Nie02] R. Niedermeier. *Invitation to fixed-parameter algorithms,* Habilitationschrift, University of Tubingen, 2002. (Electronic file available from R. Niedermeier.)

[Nie05] R. Niedermeier. *Invitation to Fixed-Parameter Algorithms*, Oxford University Press, forthcoming.

[RSS02] 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*, Springer, *Lecture Notes in Computer Science* vol. 2518 (2002), 241–248.

[RSS05] V. Raman, S. Saurabh and C.R. Subramanian. Faster algorithms for feedback vertex set. In: *Proceedings of the 2nd Brazilian Symposium on Graphs, Algorithms and Combinatorics*, GRACO 2005, April 27-29, 2005, Angra dos Reis (Rio de Janeiro), Brazil. Elsevier, *Electronic Notes in Discrete Mathematics* (2005), to appear.

[RSV04] B. Reed, K. Smith, and A. Vetta. Finding odd cycle transversals. *Operations Research Letters* 32 (2004), 299–301.

[Woe03] G. J. Woeginger. Exact algorithms for NP-hard problems: a survey. *Proceedings of 5th International Workshop on Combinatorial Optimization-Eureka, You Shrink! Papers dedicated to Jack Edmonds,* M. Junger, G. Reinelt, and G. Rinaldi (Festschrift Eds.) Springer-Verlag, *Lecture Notes in Computer Science* 2570 (2003), 184-207.