Dynamic Parameterized Problems

R. Krithika[1], Abhishek Sahu[2], and Prafullkumar Tale[3]

1 **The Institute of Mathematical Sciences, HBNI, Chennai, India**
`rkrithika@imsc.res.in`
2 **The Institute of Mathematical Sciences, HBNI, Chennai, India**
`asahu@imsc.res.in`
3 **The Institute of Mathematical Sciences, HBNI, Chennai, India**
`pptale@imsc.res.in`

Abstract

In this work, we study the parameterized complexity of various classical graph-theoretic problems in the dynamic framework where the input graph is being updated by a sequence of edge additions and deletions. Vertex subset problems on graphs typically deal with finding a subset of vertices having certain properties that are of interest to us. In real-world applications, the graph under consideration often changes over time and due to this dynamics, the solution at hand might lose the desired properties. The goal in the area of dynamic graph algorithms is to efficiently maintain a solution under these changes. Recomputing a new solution on the new graph is an expensive task especially when the number of modifications made to the graph is significantly smaller than the size of the graph. In the context of parameterized algorithms, two natural parameters are the size k of the symmetric difference of the edge sets of the two graphs (on n vertices) and the size r of the symmetric difference of the two solutions. We study the Dynamic Π-Deletion problem which is the dynamic variant of the Π-Deletion problem and show NP-hardness, fixed-parameter tractability and kernelization results. For specific cases of Dynamic Π-Deletion such as Dynamic Vertex Cover and Dynamic Feedback Vertex Set, we describe improved FPT algorithms and give linear kernels. Specifically, we show that Dynamic Vertex Cover admits algorithms with running times $1.1740^k n^{\mathcal{O}(1)}$ (polynomial space) and $1.1277^k n^{\mathcal{O}(1)}$ (exponential space). Then, we show that Dynamic Feedback Vertex Set admits a randomized algorithm with $1.6667^k n^{\mathcal{O}(1)}$ running time. Finally, we consider Dynamic Connected Vertex Cover, Dynamic Dominating Set and Dynamic Connected Dominating Set and describe algorithms with $2^k n^{\mathcal{O}(1)}$ running time improving over the known running time bounds for these problems. Additionally, for Dynamic Dominating Set and Dynamic Connected Dominating Set, we show that this is the optimal running time (up to polynomial factors) assuming the Set Cover Conjecture.

1998 ACM Subject Classification F.2 Analysis of Algorithms and Problem Complexity

Keywords and phrases dynamic problems, fixed-parameter tractability, kernelization

Digital Object Identifier 10.4230/LIPIcs.IPEC.2016.19

1 Introduction

Graphs are discrete mathematical structures that represent binary relations between objects. Due to their tremendous power to model real-world systems, many problems of practical interest can be represented as problems on graphs. Consequently, the design of algorithms on graphs is of major importance in computer science. Applications that employ graph algorithms typically involve large graphs that change over time and a natural goal in this setting is to design algorithms that efficiently maintain a solution under these updates.

Table 1 Summary of known and new results for different dynamic parameterized problems. All running time bounds suppress polynomial factors.

Dynamic Problem	Parameter k	Parameter r
Vertex Cover	1.1740^k, $1.1277^{k\ \dagger}$ $\mathcal{O}(k)$ kernel	1.2738^r $\mathcal{O}(r^2)$ kernel
Connected Vertex Cover	4^k [1], 2^k	W[2]-hard [1]
Feedback Vertex Set	$1.6667^{k\ \$}$ $\mathcal{O}(k)$ kernel	3.592^r, $3^{r\ \$}$ $\mathcal{O}(r^2)$ kernel
Dominating Set	$2^{\mathcal{O}(k^2)}$ [9], $2^{k\ \ddagger}$	W[2]-hard [9]
Connected Dominating Set	4^k [1], $2^{k\ \ddagger}$	W[2]-hard [1]
Π-Deletion	See Section 2 for hardness and tractability results	

† exponential-space algorithm $^\$$ randomized algorithm ‡ optimal under Set Cover Conjecture

In this work, we only consider instances where the possible changes to a graph are edge additions and deletions. Formally, a dynamic version of a graph-theoretic problem is a quintuple (G, G', S, k, r) where G and G' are graphs on the same vertex set and the size of the symmetric difference of their edge sets is upper bounded by k. The set S is a subset of vertices of G satisfying certain property and the task is to determine whether G' has a set S' of vertices satisfying the same property such that the symmetric difference of S and S' is at most r.

Dynamic problems have been recently studied in the parameterized complexity framework [1, 9, 15]. Two relevant parameters for dynamic problem instances are the edit parameter k and the distance parameter r. In this work, we revisit the dynamic versions of several classical problems with these parameterizations and describe parameterized complexity results. For a fixed collection of graphs Π, given a graph G and an integer l, the Π-Deletion problem is to determine if G has a set $S \subseteq V(G)$ of vertices with $|S| \leq l$ such that $G - S \in \Pi$. Π-Deletion is an abstraction of various problems in the graph-theoretic framework and it is known that finding a minimum solution to Π-Deletion is NP-hard for most choices of Π [19]. Hence, it has been extensively studied in various algorithmic realms. We define the dynamic version of this problem referred to as Dynamic Π-Deletion and show NP-hardness, fixed-parameter tractability and kernelization results. Then, for the specific cases of Π-Deletion such as Dynamic Vertex Cover and Dynamic Feedback Vertex Set, we describe improved FPT algorithms with respect to the edit parameter and give linear kernels. Then, for the same parameterization, we describe algorithms for Dynamic Connected Vertex Cover, Dynamic Dominating Set and Dynamic Connected Dominating Set. We also show running time lower bounds for algorithms solving Dynamic Dominating Set and Dynamic Connected Dominating Set assuming the Set Cover Conjecture [5]. Table 1 summarizes our results along with the running time bounds known for these problems.

All graphs considered in this paper are finite, undirected, unweighted and simple. Notation/definitions not given explicitly here can be found in [8]. For a graph G, $V(G)$ and $E(G)$ denote the sets of vertices and edges respectively. The symmetric difference of two subsets $S, S' \subseteq V(G)$, denoted by $d_v(S, S')$, is defined as the size of the set $(S \setminus S') \cup (S' \setminus S)$. For two graphs G and G' on the same vertex set, $d_e(G, G')$ denotes the size of the symmetric difference of $E(G)$ and $E(G')$. For a vertex $u \in V(G)$, its neighbourhood $N_G(u)$ is set of all the vertices adjacent to it and its closed neighbourhood $N_G[u]$ is the set $N_G(u) \cup \{u\}$. This notation is extended to subsets of vertices as $N_G[S] = \bigcup_{v \in S} N_G[v]$ and $N_G(S) = N_G[S] \setminus S$ where $S \subseteq V(G)$. The subscript in the notation for neighbourhood is omitted if the graph under consideration is clear from the context. The degree of a vertex is the size of its neighbourhood. For a set E' of edges, $V(E')$ denotes the union of the endpoints of the edges in E'. For a set $S \subseteq V(G)$, $G[S]$ and $G - S$ denote the subgraphs of G induced on sets S and $V(G) \setminus S$ respectively. The contraction operation of an edge $e = uv$ in G results in the addition of a new vertex w adjacent to $N_G(u) \cup N_G(v)$ and the deletion of u and v. An independent set is a set of pairwise non-adjacent vertices and a clique is a set of pairwise adjacent vertices. A triangle is a clique of size 3. We also refer to an edgeless graph as an independent set and a complete graph as a clique. A forest is a graph with no cycles and a tree is a connected forest.

In parameterized complexity, every instance of a problem is associated with an non-negative integer called as the parameter that often measures some structural property of the instance. A common parameter is a bound l on the size of an optimum solution to the problem instance I. A problem is fixed-parameter tractable (FPT) with respect to the parameter l if it has an algorithm with $f(l)|I|^{\mathcal{O}(1)}$ running time for some computable function f. The running time $f(l)|I|^{\mathcal{O}(1)}$ where f is an exponential function is also specified as $\mathcal{O}^*(f(l))$ suppressing the polynomial factors. In order to classify parameterized problems as being FPT or not, the W-hierarchy is defined as $\mathsf{FPT} \subseteq \mathsf{W}[1] \subseteq \mathsf{W}[2] \subseteq \cdots \subseteq \mathsf{XP}$ using Boolean circuits and parameterized reductions. It is believed that the subset relations in this sequence are all strict and a parameterized problem that is hard for some complexity class above FPT in this hierarchy is unlikely to be FPT. A kernelization algorithm is a polynomial-time algorithm that transforms an instance (I, l) of the problem to an equivalent instance (I', l') of the same problem such that $|I'| + l' = g(l)$ for some computable function g. The instance (I', l') is called a kernel and if $g(l) = l^{\mathcal{O}(1)}$, then it is called a polynomial kernel and we say that the problem admits a polynomial kernelization (or kernel). We refer the reader to [6, 10, 11] for an introduction to parameterized complexity and kernelization.

2 Dynamic Π-Deletion

A graph property Π is a collection of graphs. Π is said to be (induced) hereditary if for any graph in Π, all of its (induced) subgraphs are in Π. The membership testing problem for Π is the task of determining if a graph is in Π or not. Let I_n denote the graph on n vertices with no edges and K_n denote the complete graph on n vertices. For most natural choices of Π, the Π-Deletion problem is NP-complete [19] and interesting dichotomy results are known in the parameterized complexity framework [2, 16]. We formally define its dynamic variant referred to as Dynamic Π-Deletion as follows.

Dynamic Π-Deletion **Parameter:** k, r
Input: Graphs G, G' on the same vertex set, a set $S \subseteq V(G)$ such that $G - S \in \Pi$ and integers k, r with $d_e(G, G') \leq k$.
Question: Does there exist $S' \subseteq V(G')$ with $d_v(S, S') \leq r$ such that $G' - S' \in \Pi$?

Observe that if Π-Deletion is in NP then so is Dynamic Π-Deletion. We are now ready to state our first result.

▶ **Theorem 2.1.** *Let Π be a graph property that includes all independent sets or all cliques. Then, Π-Deletion reduces to Dynamic Π-Deletion in polynomial time.*

Proof. Let (H, l) be an instance of Π-Deletion. We reduce (H, l) to the instance $(G, G' = H, S = \emptyset, k, r = l)$ of Dynamic Π-Deletion as follows. If Π includes all independent sets then $G = I_{|V(H)|}$ and $k = |E(H)|$. Otherwise, $G = K_{|V(H)|}$ and $k = \binom{|V(H)|}{2} - |E(H)|$. In both the cases, by the property of Π, $G - S \in \Pi$. Also, the vertex sets of H, G and G' are the same. Then, for a set $S' \subseteq V(H)$, we have $H - S' \in \Pi$ if and only if $G' - S' \in \Pi$ such that $d_v(S, S') = |S'|$. ◀

As a consequence of Theorem 2.1, we have the following hardness result.

▶ **Corollary 2.2.** *Let Π be a property that includes all independent sets or all cliques.*

- *If Π-*Deletion *is* NP*-hard, then* Dynamic Π-Deletion *is* NP*-hard.*
- *If Π-*Deletion *parameterized by solution size is fixed-parameter intractable then* Dynamic Π-Deletion *parameterized by r is fixed-parameter intractable.*
- *If Π-*Deletion *is* NP*-complete and does not admit a polynomial kernel when parameterized by solution size then* Dynamic Π-Deletion *parameterized by r does not admit a polynomial kernel.*

Proof. The NP-hardness and the fixed-parameter intractability results follow straightaway from Theorem 2.1. If Π-Deletion is NP-complete, then Dynamic Π-Deletion (which is in NP) reduces to Π-Deletion in polynomial time. Therefore, if Dynamic Π-Deletion parameterized by r admits a polynomial kernel, such a kernel can be transformed to a polynomial (in solution size) kernel for Π-Deletion using this reduction and the reduction described in Theorem 2.1.Thus, the claimed kernelization hardness follows too. ◀

The following lemma shows that for many choices of Π, to solve Dynamic Π-Deletion, it suffices to look for a solution that contains the current solution.

▶ **Lemma 2.3.** *Let Π be an induced hereditary property. If S' is a solution to the* Dynamic Π-Deletion *instance (G, G', S, k, r) with $d_v(S, S') = r'$, then there is another solution S'' with $d_v(S, S'') \leq r'$ and $S \subseteq S''$.*

Proof. We have $d_v(S, S') = |S \setminus S'| + |S' \setminus S| = r'$. Let S'' be the set $S \cup S'$. Then, $d_v(S, S'') = |S \setminus S''| + |S'' \setminus S| = |S'' \setminus S| = |S' \setminus S| \leq r'$. Now, as $G' - S' \in \Pi$ and Π is hereditary, it follows that $G' - S'' \in \Pi$. ◀

Now, we proceed to show that Dynamic Π-Deletion reduces to Π-Deletion for many choices of Π.

▶ **Theorem 2.4.** *Let Π be an induced hereditary property whose membership testing problem is polynomial-time solvable. Then,* Dynamic Π-Deletion *reduces to* Π-Deletion *in polynomial time.*

Proof. Consider an instance (G, G', S, k, r) of Dynamic Π-Deletion. The task is to determine if G' has a solution S' with $d_v(S, S') \leq r$. If $G' - S \in \Pi$, then S is the required solution S'. Otherwise, from Lemma 2.3, assume that the required set S' contains S. Let H denote the graph $G' - S$. Then, $H - (S' \setminus S) \in \Pi$. Therefore, for a set $T \subseteq V(H)$, we have $H - T \in \Pi$ if and only if $G' - (S \cup T) \in \Pi$ such that $d_v(S, S') = |T|$. ◀

Then, the following claim holds.

▶ **Corollary 2.5.** *Let Π be a hereditary property whose membership testing problem is polynomial-time solvable. If Π-*Deletion *is* FPT *with respect to the solution size l as the parameter, then* Dynamic Π-Deletion *is* FPT *with respect to both r and k as parameters.*

Proof. Consider an instance (G, G', S, k, r) of Dynamic Π-Deletion. Suppose Π-Deletion admits an algorithm with $\mathcal{O}^*(f(l))$ running time. Then, from Theorem 2.4, there is an algorithm $\mathcal{A}$ solving (G, G', S, k, r) in $\mathcal{O}^*(f(r))$ time. Thus, the problem is FPT when parameterized by r. Let $\tilde{E}$ denote the set $E(G') \setminus E(G)$. Let T be a set of vertices of G' of size at most k that contains at least one endpoint of each edge in $\tilde{E}$. As Π is hereditary and $G - S \in \Pi$, it follows that $G' - (S \cup T) \in \Pi$. Now, if $r \geq k$, then $S \cup T$ is the required solution S'. Otherwise, the algorithm $\mathcal{A}$ solving Dynamic Π-Deletion runs in $\mathcal{O}^*(f(k))$ time. Hence, the problem is FPT when parameterized by k. ◀

Finally, we move on to kernelization results.

▶ **Corollary 2.6.** *Let Π be a hereditary property whose membership testing problem is polynomial-time solvable. Suppose* Π-Deletion *parameterized by the solution size l admits a kernel with $p(l)$ vertices and $q(l)$ edges.*

- *If Π includes all independent sets, then* Dynamic Π-Deletion *admits a kernel with $2p(r) \leq 2p(k)$ vertices and $q(r) \leq q(k)$ edges.*
- *If Π includes all cliques, then* Dynamic Π-Deletion *admits a kernel with $2p(r) \leq 2p(k)$ vertices and $q(r) + p(r)^2 \leq q(k) + p(k)^2$ edges.*

Proof. Consider an instance (G, G', S, k, r) of Dynamic Π-Deletion. If $G' - S \in \Pi$ or $r \geq k$, the output of the kernelization algorithm is $(K_1, \emptyset, K_1, 0, 0)$ which is a trivial YES instance of Dynamic Π-Deletion. Suppose $G' - S \notin \Pi$ and $r < k$. Let (H', r') be the kernel of the instance (H, r) of Π-Deletion obtained from Theorem 2.4. Then, $(H'', H', \emptyset, |E(H')|, r')$ is the kernel of (G, G', S, k, r) where $H'' = I_{|V(H')|}$ if Π includes all independent sets and $H'' = K_{|V(H')|}$ if Π includes all cliques. Hence, the claimed bounds on the kernel size follow. ◀

A property Π is interesting if the number of graphs in Π and the number of graphs not in Π are unbounded. Any induced hereditary property that is interesting either contains all independent sets or contains all cliques. Thus, all the above results hold for such properties. In particular, the results of this section hold for the dynamic variants of classical problems like Vertex Cover and Feedback Vertex Set.

3 Dynamic Vertex Cover

A vertex cover is a set of vertices that has at least one endpoint from every edge. Given a graph G and an integer l, Vertex Cover is the problem of determining if G has a vertex cover of size l. Its dynamic variant, Dynamic Vertex Cover, is defined as follows.

Dynamic Vertex Cover **Parameter:** k, r
Input: Graphs G, G' on the same vertex set, a vertex cover S of G and integers k, r such that $d_e(G, G') \leq k$.
Question: Does there exist a vertex cover S' of G' such that $d_v(S, S') \leq r$?

In [1] it is claimed that Dynamic Vertex Cover is W[1]-hard with respect to $k + r$ as the parameter by a reduction from Independent Set parameterized by the solution size. However, the reduction is incorrect and the fixed-parameter intractability does not follow. Clearly, Dynamic Vertex Cover is Dynamic Π-Deletion where Π is the set of all independent sets. Vertex Cover parameterized by the solution size l admits a kernel with at most $2l$ vertices [4] and the current best FPT algorithm runs in $\mathcal{O}^*(1.2738^l)$ time [3].

By Theorem 2.4 and Corollaries 2.5 and 2.6, these results extend to Dynamic Vertex Cover as well. Also, as Vertex Cover is NP-hard, its dynamic version is NP-hard too by Theorem 2.1 and Corollary 2.2. In particular, the following results hold.

- Dynamic Vertex Cover is NP-complete and admits algorithms with $\mathcal{O}^*(1.2738^r)$ and $\mathcal{O}^*(1.2738^k)$ running times.
- Dynamic Vertex Cover admits an $\mathcal{O}(r^2)$ kernel and an $\mathcal{O}(k^2)$ kernel.

We now improve over these results by describing a linear kernel and a faster FPT algorithm with respect to k as the parameter. First, we describe the linear kernelization.

▶ **Theorem 3.1.** Dynamic Vertex Cover *admits a kernel with at most* $2k$ *vertices and at most* k *edges.*

Proof. Consider an instance (G, G', S, k, r) of Dynamic Vertex Cover. By Lemma 2.3, it suffices to search for a solution S' that contains S. As $d_e(G, G') \leq k$, we have $|E(G')\backslash E(G)| \leq k$. From Theorem 2.4 and Corollary 2.6, it suffices to output a linear kernel of the Vertex Cover instance $(G' - S, r)$. We apply the following standard preprocessing on $G' - S$.

▶ **Reduction Rule 3.2.** *Delete isolated vertices.*

▶ **Reduction Rule 3.3.** *If there is a vertex* v *of degree* 1 *add* $N(v)$ *into the solution, decrease* r *by 1 and delete* $N[v]$ *from the graph.*

Let H' denote the resultant graph on which these rules are no longer applicable and r' denote the corresponding budget. Then, $(G' - S, r)$ is a YES instance of Vertex Cover if and only if (H', r') is a YES instance of Vertex Cover. As the minimum degree of H' is at least 2, we have $2|E(H')| \geq 2|V(H')|$. As $G' - S$ has at most k edges, it follows that $|E(H')| \leq k$. Thus, $|V(H')| \leq k$ and from Corollary 2.6 the kernel of (G, G', S, k, r) is $(I_{|V(H')|}, H', \emptyset, k = |E(H')|, r')$. ◀

Next, we describe an algorithm faster than $\mathcal{O}^*(1.2738^k)$.

▶ **Theorem 3.4.** Dynamic Vertex Cover *can be solved in* $\mathcal{O}^*(1.1740^k)$ *time.*

Proof. Consider an instance (G, G', S, k, r) of Dynamic Vertex Cover. By Lemma 2.3, it suffices to search for a solution S' that contains S. From Theorem 2.4 and Corollary 2.6, it suffices to solve the Vertex Cover instance $(G' - S, r)$. We first apply Reduction Rules 3.2 and 3.3 on $H = G' - S$ as long as they are applicable. Then, $|V(H)| \leq |E(H) \leq k$. A minimum vertex cover of a graph on n vertices can be found in $\mathcal{O}^*(1.2002^n)$ time [22]. Thus, an $\mathcal{O}^*(1.2002^k)$ algorithm follows. We will describe a faster branching algorithm where the measure used to bound the number of nodes of the search tree is the number of edges in H and the leaves of the tree are instances corresponding to the empty graph or a graph with maximum degree at most 2. To this end, we apply the following additional rule exhaustively.

▶ **Reduction Rule 3.5.** *If there is a triangle on vertices* u, v, w *such that* $|N(u)| = 2$ *then include* v, w *into the solution and delete* u, v, w *from the graph.*

We eliminate all other triangles in the graph by applying following branching strategy.

▶ **Branching Rule 3.6.** *Let* u, v, w *be the vertices of a triangle.*

- *Branch 1: Include* u *into the solution and delete it from the graph.*
- *Branch 2: Include* $N(u)$ *into the solution and delete* $N[u]$ *from the graph.*

As the degree of a vertex in a triangle is at least 3, the measure drops by at least 3 in the first branch and by at least 6 in the second. When this rule is no longer applicable, we have a triangle-free graph. Now, we state our final branching rule.

▶ **Branching Rule 3.7.** *Let u be a vertex of degree at least three.*
- *Branch 1: Include vertex u into the solution and delete it from the graph.*
- *Branch 2: Include $N(u)$ into the solution and delete $N[u]$ from the graph.*

In the first branch, the measure drops by at least 3. As the graph is triangle-free, no two neighbours of u are adjacent. Further, all vertices have degree at least 2. Therefore, the measure drops by at least $2|N(u)| \geq 6$ in the second branch. As no new edges are added to the graph in any rule, the measure never increases after the application of a reduction or branching rule. All reduction rules can be applied in polynomial time and at each branching rule, we only spend polynomial time to find a vertex to branch on. When k is zero or the maximum degree of the graph is at most 2, finding a minimum vertex cover is polynomial-time solvable. If the vertex cover budget exceeds the permissible value at a branch, that branch is aborted. The initial measure is upper bounded by k and the worst case branching vector is $(3, 6)$. This leads to the recurrence $T(k) \leq T(k-3) + T(k-6)$ whose solution is 1.1740^k. Thus, the algorithm runs in $\mathcal{O}^*(1.1740^k)$ time. ◀

The treewidth of a graph is a parameter that quantifies the closeness of the graph to a tree (see [6] for the precise definition). If the treewidth of the input graph is upper bounded by tw, then a minimum vertex cover can be obtained in $\mathcal{O}^*(2^{tw})$ time [6]. The following result relates the treewidth of a graph to the number of its vertices and edges.

▶ **Lemma 3.8** ([17]). *If G is a graph on n vertices and m edges, then the treewidth of G is upper bounded by $\frac{m}{5.769} + \mathcal{O}(\log n)$. Moreover, a tree decomposition of the corresponding width can be constructed in polynomial time.*

Since the graph H on which a minimum vertex cover is desired has at most k edges and k vertices, its treewidth tw is bounded by $\frac{k}{5.769} + \mathcal{O}(\log k)$. Then, we have the following result.

▶ **Theorem 3.9.** Dynamic Vertex Cover *can be solved in $\mathcal{O}^*(1.1277^k)$ time.*

Though this algorithm is faster than the branching algorithm described earlier, it requires exponential space while the algorithm in Theorem 3.4 requires only polynomial space.

4 Dynamic Connected Vertex Cover

A connected vertex cover of a graph is a vertex cover that induces a connected subgraph and Dynamic Connected Vertex Cover is defined as follows.

Dynamic Connected Vertex Cover **Parameter:** k, r
Input: Graphs G, G' on the same vertex set, a connected vertex cover S of G and integers k, r such that $d_e(G, G') \leq k$.
Question: Does there exist a connected vertex cover S' of G' such that $d_v(S, S') \leq r$?

The problem is NP-complete, W[2]-hard when parameterized by r and admits an $\mathcal{O}^*(4^k)$ algorithm by a reduction to finding a minimum weight Steiner tree [1]. We describe an $\mathcal{O}^*(2^k)$ algorithm by a reduction to finding a group Steiner tree. Given a graph G, an integer p and a family $\mathcal{F}$ of subsets of $V(G)$, the Group Steiner Tree problem is the task of determining whether G contains a tree on at most p vertices that contains at least one vertex from each set in $\mathcal{F}$. This problem is known to admit an algorithm with $\mathcal{O}^*(2^{|\mathcal{F}|})$ running time [20]. We use this algorithm as a subroutine in our algorithm for Dynamic Connected Vertex Cover. First, we show a lemma on the property of a solution to an instance of Dynamic Connected Vertex Cover analogous to Lemma 2.3.

▶ **Lemma 4.1.** *Consider an instance (G, G', S, k, r) of* Dynamic Connected Vertex Cover. *If S' is a connected vertex cover of G' with $d_v(S, S') = r'$, then $S' \cup S$ is also a connected vertex cover of G' with $d_v(S, S' \cup S) \leq r'$.*

Proof. Assume G' is connected, otherwise the given instance is a NO instance. As a set that contains a vertex cover is also a vertex cover, it follows that $T = S' \cup S$ is a vertex cover of G'. As S' is a vertex cover of G', $S \setminus S'$ is an independent set in G'. As G' is connected, every vertex in $S \setminus S'$ is adjacent to some vertex in S'. Then, as $G'[S']$ is connected and $S' \subseteq T$, it follows that $G'[T]$ is connected too. Further, as $T \setminus S = S' \setminus S$ it follows that $d_v(S, T) = |T \setminus S| + |S \setminus T| = |T \setminus S| = |S' \setminus S| \leq r'$. ◀

Now, we prove the main result of this section.

▶ **Theorem 4.2.** Dynamic Connected Vertex Cover *can be solved in $\mathcal{O}^*(2^k)$ time.*

Proof. Consider an instance (G, G', S, k, r) of Dynamic Connected Vertex Cover. By Lemma 4.1, we can assume that the required solution S' contains S. Observe that $G'[S]$ is not necessarily connected and the edges in G' that are not covered by S are those edges in $E' = (E(G') \setminus E(G)) \cap E(G' - S)$. Now, we show a reduction to finding a group Steiner tree. Contract each connected component of $G'[S]$ to a single vertex. Let H denote the resulting graph and let $X = V(H) \setminus V(G')$. Construct an instance $(H, |X| + r, \mathcal{F})$ of Group Steiner Tree where $\mathcal{F} = \{\{u, v\} \mid uv \in E'\} \cup \{\{x\} \mid x \in X\}$. We claim that (G, G', S, k, r) is a YES instance of Dynamic Connected Vertex Cover if and only if $(H, |X| + r, \mathcal{F})$ is a YES instance of Group Steiner Tree.

Suppose G' has a connected vertex cover S' such that $d_v(S, S') \leq r$ and $S \subseteq S'$. As $G'[S']$ is connected, it follows that $H[X \cup (S' \cap V(G' - S))]$ is also connected. Moreover, as $|S' \cap V(G' - S)| \leq r$, it follows that the spanning tree of $H[X \cup (S' \cap V(G' - S))]$ is of size at most $|X| + r$. Hence $(H, |X| + r, \mathcal{F})$ is a YES instance of Group Steiner Tree. Conversely, suppose $(H, |X| + r, \mathcal{F})$ is a YES instance of Group Steiner Tree. Let T denote the solution tree of H. Then, $X \subseteq V(T)$ and $|V(T) \setminus X| = |V(T) \cap V(G' - S)| \leq r$. Define $S' = S \cup (V(T) \cap V(G' - S))$. The size of S' is at most $|S| + r$. Further, $G'[S']$ is connected as S' is obtained from the vertices of T. Also, for every edge in E', T contains at least one of its endpoints. Thus, S' is the desired connected vertex cover of G'. As the sum of the number of connected components of $G'[S]$ and the size of E' is upper bounded by $k + 1$, it follows that $|\mathcal{F}| \leq k + 1$. Thus, the Group Steiner Tree algorithm of [20] runs in $\mathcal{O}^*(2^k)$ time. ◀

5 Dynamic Feedback Vertex Set

A feedback vertex set is a set of vertices whose deletion results in a forest and Dynamic Feedback Vertex Set is defined as follows.

Dynamic Feedback Vertex Set **Parameter:** k, r
Input: Graphs G, G' on the same vertex set, a feedback vertex set X of G and integers k, r such that $d_e(G, G') \leq k$.
Question: Does there exist a feedback vertex set X' of G' such that $d_v(X, X') \leq r$?

Clearly, Dynamic Feedback Vertex Set is Dynamic Π-Deletion where Π is the set of all forests. Given a graph G and an integer l, the Feedback Vertex Set problem is the task of determining if G has a feedback vertex set of at most l vertices. As it is a classical NP-complete problem, its dynamic variant is NP-complete too by Theorem 2.1 and Corollary 2.2. Feedback Vertex Set is known to admit an $\mathcal{O}^*(3.592^l)$ algorithm [18] and a kernel with $\mathcal{O}(l^2)$ vertices [21]. Also, a randomized algorithm that solves the problem in $\mathcal{O}^*(3^l)$ time is known [7]. By Theorem 2.4 and Corollaries 2.5 and 2.6, all these results extend to Dynamic Feedback Vertex Set. In particular, the following results hold.

- Dynamic Feedback Vertex Set can be solved in $\mathcal{O}^*(3.592^r)$ time and in $\mathcal{O}^*(3.592^k)$ time.
- Dynamic Feedback Vertex Set admits randomized algorithms with $\mathcal{O}^*(3^r)$ and $\mathcal{O}^*(3^k)$ running times.
- Dynamic Feedback Vertex Set admits an $\mathcal{O}(r^2)$ kernel and an $\mathcal{O}(k^2)$ kernel.

We now improve these bounds by describing a linear kernel and a faster randomized FPT algorithm with respect to k as the parameter. First, we describe the linear kernelization.

▶ **Theorem 5.1.** Dynamic Feedback Vertex Set *admits a kernel with at most* $4k$ *vertices and at most* $3k$ *edges.*

Proof. Consider an instance (G, G', X, k, r) of Dynamic Feedback Vertex Set. Observe that if G' is obtained from G by only deleting edges, then X is feedback vertex set of G' too. Also, edges in $E(G') \setminus E(G)$ that have an endpoint in X do not affect the solution. Moreover, from Lemma 2.3, it suffices to search for a feedback vertex set of G' that contains X. Let H be the subgraph of G' induced on $V(G') \setminus X$. From Theorem 2.4, (G, G', X, k, r) is a YES instance of Dynamic Feedback Vertex Set if and only if (H, r) is a YES instance of Feedback Vertex Set. From Corollary 2.6, it suffices to output a linear kernel of the instance (H, r). We primarily exploit the fact that H is obtained by adding at most k edges to a forest. This implies that $|E(H)| \leq |V(H)| + k - 1$. Let $\tilde{E}$ be the set of edges in G' whose both endpoints are in $V(G) \setminus X$ and $U = V(\tilde{E})$. We apply the following reduction rule to $G - X$. Note that $G - X$ is a subgraph of H as $E(H) = E(G - X) \cup \tilde{E}$.

▶ **Reduction Rule 5.2.** *If there is a vertex v of degree at most 1 such that $v \notin U$, then delete v from the graph.*

This rule preserves the size of a minimum feedback vertex set as v has degree at most 1 in H too and no minimal feedback vertex set of H contains it. As the number of vertices with degree at least 3 is upper bounded by the number of leaves in a forest, it follows that the number of vertices of degree at least 3 is at most $2k$ on the resulting graph G'' on which this rule is not applicable. Consider the graph H'' obtained from G'' by adding $\tilde{E}$. We once again delete vertices of degree at most 1 (if any) and then apply following reduction rule exhaustively.

▶ **Reduction Rule 5.3.** *If there is a vertex v of degree 2, then delete v and add an edge between its two neighbours.*

Once again this rule preserves the size of a minimum feedback vertex set as any minimal feedback vertex set of H'' that contains v can be modified into another minimal feedback vertex set of at least the same size that does not contain v. Note that the application of Reduction Rules 5.2 and 5.3 ensure that $|E(H'')| \leq |V(H'')| + k - 1$ is satisfied. When neither of the reduction rules are applicable on H'', the minimum degree of H'' is at least 3 and $|E(H'')| \leq |V(H'')| + k - 1$. This implies that $1.5|V(H'')| \leq |E(H'')| \leq |V(H'')| + k - 1$

and hence $|V(H'')| \leq 2k-2$, $|E(H'')| \leq 3k-3$. Also, (H,r) is a YES instance of Feedback Vertex Set if and only if (H'',r) is a YES instance of Feedback Vertex Set. Thus, from Corollary 2.6, the kernel of (G,G',S,k,r) is $(I_{|V(H'')|}, H'', \emptyset, k = |E(H'')|, r)$. ◀

Next, we proceed to describe a randomized FPT algorithm for the problem. If the treewidth of the input graph is upper bounded by tw then there is a randomized $\mathcal{O}^*(3^{tw})$ time algorithm that computes a minimum feedback vertex set [7]. Further, for finding a minimum feedback vertex set of a graph on n vertices, there is a randomized algorithm running in $\mathcal{O}(1.6667^n)$ time [12]. The following result relates the treewidth of a graph to the number of its vertices and edges.

▶ **Lemma 5.4** ([13]). *For any $\epsilon > 0$, there exists an integer n_ϵ such that for every connected graph G on n vertices and m edges with $n > n_\epsilon$ and $1.5n \leq m \leq 2n$, the treewidth of G is upper bounded by $\frac{m-n}{3} + \epsilon n$. Moreover, a tree decomposition of the corresponding width can be constructed in polynomial time.*

This theorem along with the described linear kernelization leads to the following result.

▶ **Theorem 5.5.** Dynamic Feedback Vertex Set *admits a randomized algorithm running in $\mathcal{O}^*(1.6667^k)$ time.*

Proof. Consider an instance (G,G',X,k,r) of Dynamic Feedback Vertex Set. Let (H'',r) be the corresponding instance of Feedback Vertex Set obtained from the linear kernelization of Theorem 5.1. That is, H'' is a graph (not necessarily simple) on n vertices and m edges such that $m \leq n+k-1$ and $n \leq 2k-2$. Further, every vertex of H'' has degree at least 3 and hence $m \geq 1.5n$. If $m > 2n$, then as $m \leq n+k-1$, we have $n < k-1$. Then, a minimum feedback vertex set of H'' can be obtained in $\mathcal{O}(1.6667^k)$ using the randomized exact exponential-time algorithm described in [12]. Otherwise, $1.5n \leq m \leq 2n$. Let ϵ be a constant (to be chosen subsequently). Then, let n_ϵ be the integer obtained from Lemma 5.4 satisfying the required properties. If $n \leq n_\epsilon$, then a minimum feedback vertex set of H'' can be obtained in constant time as n_ϵ is a constant depending only on ϵ. Otherwise, the treewidth of H'' is at most $t = \frac{m-n}{3} + \epsilon n = \frac{m}{3} + n(\epsilon - \frac{1}{3})$. Then, using the randomized algorithm described in [7], a minimum feedback vertex set of H'' can be obtained in $\mathcal{O}^*(3^t)$ time. Now, by choosing ϵ to be a sufficiently small constant, t can be made arbitrarily close to $\frac{m-n}{3}$. For instance, if $\epsilon = 10^{-10}$, then t is $.\overline{3}m - .333333333\overline{2}3n$. As $\frac{m-n}{3} \leq \frac{n+k-1-n}{3} = \frac{k}{3}$, the algorithm in [7] runs in $\mathcal{O}^*(1.443^k)$ time. ◀

6 Dynamic Dominating Set

A dominating set is a set of vertices that has a non-empty intersection with the closed neighbourhood of every vertex and a set $S \subseteq V(G)$ is said to dominate another set $T \subseteq V(G)$ if $T \subseteq N[S]$. The Dynamic Dominating Set problem is defined as follows.

Dynamic Dominating Set	**Parameter:** k, r
Input: Graphs G, G' on the same vertex set, a dominating set D of G and integers k, r such that $d_e(G,G') \leq k$.	
Question: Does there exist a dominating set D' of G' such that $d_v(D,D') \leq r$?	

The problem is NP-complete and W[2]-hard when parameterized by r [9]. Also, it is FPT when parameterized by k via an $2^{\mathcal{O}(k^2)}$ algorithm but admits no polynomial kernel unless NP $\subseteq$ coNP/poly [9]. We describe a faster FPT algorithm for this parameterization. First, we show that it suffices to look for a dominating set with a specific property.

▶ **Lemma 6.1.** *Consider an instance (G, G', D, k, r) of* Dynamic Dominating Set*. If D' is a dominating set of G' with $d_v(D, D') = r'$, then $D' \cup D$ is also a dominating set of G' with $d_v(D, D' \cup D) \leq r'$.*

Proof. As a set that contains a dominating set is also a dominating set, it follows that $D'' = D' \cup D$ is a dominating set of G'. Further, $d_v(D, D'') = |D'' \setminus D| + |D \setminus D''| = |D'' \setminus D| = |D' \setminus D| \leq r'$. ◀

Now, we solve Dynamic Dominating Set by reducing it to an instance of Set Cover. In the Set Cover problem, we are given a family $\mathcal{F}$ of subsets of a universe U and a positive integer ℓ. The problem is to determine whether there exists a subfamily $\mathcal{F}' \subseteq \mathcal{F}$ of size at most ℓ such that $U = \bigcup_{X \in \mathcal{F}'} X$.

▶ **Theorem 6.2.** Dynamic Dominating Set *admits an algorithm that runs in $\mathcal{O}^*(2^k)$ time.*

Proof. Consider an instance (G, G', D, k, r) of Dynamic Dominating Set. If G' is obtained from G by only adding edges, then D is dominating set of G'. The only kind of edge deletions that could possibly affect the solution are those that have one endpoint in D and the other endpoint in $V(G') \setminus D$. Further, as $d_e(G, G') \leq k$, $|V(G') \setminus N_{G'}[D]| \leq k$. That is, there are at most k vertices in G' that are not dominated by D. Let H be the subgraph of G' induced on $V(G') \setminus D$. Partition $V(H)$ into two sets $C = N_{G'}(D)$ and $B = V(H') \setminus C$.

We claim that (G, G', D, k, r) is a YES instance of Dynamic Dominating Set if and only if there exists a set $P \subseteq V(H)$ of cardinality at most r such that $B \subseteq N_H[P]$. If there is a set P of size at most r in $V(H)$ that dominates B, then $D' = D \cup P$ is a dominating set of G' with $d_v(D, D') \leq r$. Hence, (G, G', D, k, r) is a YES instance of Dynamic Dominating Set. Conversely, suppose there is a dominating set D' of G' with $d_v(D, D') \leq r$. Define D'' as $D' \setminus D$. Note that $|D''| \leq r$. By construction of H, B is not dominated by D and hence $B \subseteq N_H[D'']$. This implies that D'' is the required set of vertices of H that dominates B.

The problem now reduces to finding a set of at most r vertices from $B \cup C$ that dominates B in H. We construct an instance of Set Cover with $U = B$, $\mathcal{F} = \{N_H(u) \cap B \mid u \in C\} \cup \{N_H[w] \cap B \mid w \in B\}$ and $\ell = r$. Then, there exists a set P of size at most r in H which dominates B if and only if $(U, \mathcal{F}, \ell)$ is a YES instance of Set Cover. A set $X \in \mathcal{F}$ is said to be associated with a vertex v in C if $X = N_H(v) \cap B$ or with a vertex v in B if $X = N_H[v] \cap B$. If there exists a set P with desired property, then every vertex w in B is contained in open or closed neighbourhood of some vertex in P. Consider the subfamily $\mathcal{F}'$ of $\mathcal{F}$ that are associated with vertices in P. Every element of U is contained in at least one of these sets. Thus, $\mathcal{F}'$ is the required set cover. Conversely, if there exists a set cover $\mathcal{F}'$ of size at most $\ell = r$, then let P' be the set of vertices which are associated with sets in $\mathcal{F}'$. Then, $|P'| = |\mathcal{F}'| \leq r$ and every vertex in B is either in P' or is adjacent to some vertex in P'. Hence, P' is the desired set.

As any instance $(U, \mathcal{F}, \ell)$ of Set Cover can be solved in $\mathcal{O}^*(2^{|U|})$ [14] and $|U| = |B| \leq k$, the claimed running time bound follows. ◀

Finally, we show a lower bound on the running time of an algorithm that solves Dynamic Dominating Set assuming the Set Cover Conjecture which states that Set Cover cannot be solved in $\mathcal{O}^*((2-\epsilon)^{|U|})$ for any $\epsilon > 0$ [5]. We do so by a reduction from Set Cover to Dynamic Dominating Set.

▶ **Theorem 6.3.** Dynamic Dominating Set *does not admit an algorithm with $\mathcal{O}^*((2-\epsilon)^k)$ running time for any $\epsilon > 0$ assuming the Set Cover Conjecture.*

Proof. Consider an instance $(U, \mathcal{F}, \ell)$ of Set Cover where $U = \{u_1, \cdots, u_n\}$ and $\mathcal{F} = \{S_1, \cdots, S_m\}$. Without loss of generality, assume that every u_i is in at least one set S_j. Let G be the graph with vertex set $U \cup V \cup \{x\}$ where $U = \{u_1, \cdots, u_n\}$ and $V = \{s_1, \cdots, s_m\}$. The set V is a clique and the set U is an independent set in G. Further, a vertex u_i is adjacent to s_j if and only if $u_i \in S_j$ and x is adjacent to every vertex in $U \cup V$. Clearly, $D = \{x\}$ is a dominating set of G. Let G' be the graph obtained from G by deleting edges between x and U. We claim that $(U, \mathcal{F}, \ell)$ is a YES instance of Set Cover if and only if $(G, G', D = \{x\}, k = n, r = \ell)$ is a YES instance of Dynamic Dominating Set. Suppose $\mathcal{F}'$ is a set cover of size at most ℓ. Then, $D' = D \cup \{s_i \mid S_i \in \mathcal{F}'\}$ is a dominating set of G' with $d_v(D, D') \leq \ell$. Conversely, suppose G' has a dominating set D' with $d_v(D, D') \leq \ell$. From Lemma 6.1, assume that $D \subseteq D'$ and so $|D' \setminus D| \leq \ell$. For every vertex $u \in U \cap D'$, replace u by one of its neighbours in V. The resultant dominating set D'' contains D and satisfies $D'' \setminus D \subseteq V$. Now, $\{S_i \in \mathcal{F} \mid v_i \in D'' \cap V\}$ is a set cover of size at most ℓ. This leads to the claimed lower bound under the Set Cover Conjecture. ◀

7 Dynamic Connected Dominating Set

A connected dominating set is a dominating set that induces a connected subgraph and the Dynamic Connected Dominating Set problem is defined as follows.

Dynamic Connected Dominating Set **Parameter:** k, r
Input: Graphs G, G' on the same vertex set, a connected dominating set D of G and integers k, r such that $d_e(G, G') \leq k$.
Question: Does there exist a connected dominating set D' of G' such that $d_v(D, D') \leq r$?

The problem is NP-complete and admits an $\mathcal{O}^*(4^k)$ algorithm by a reduction to finding a minimum weight Steiner tree [1]. We now show that it admits an $\mathcal{O}^*(2^k)$ algorithm by a reduction to finding a group Steiner tree. Analogous to the problems considered earlier, we first prove a property on the required solution.

▶ **Lemma 7.1.** *Consider an instance (G, G', D, k, r) of* Dynamic Connected Dominating Set*. If D' is a connected dominating set of G' with $d_v(D, D') = r'$, then $D' \cup D$ is also a connected dominating set of G' with $d_v(D, D' \cup D) \leq r'$.*

Proof. Assume G' is connected, otherwise the given instance is a NO instance. As a set that contains a dominating set is also a dominating set, it follows that $D'' = D' \cup D$ is a dominating set of G'. Now, $D'' \setminus D$ is $D' \setminus D$. As D' is a dominating set of G', every vertex in $D \setminus D'$ is adjacent to some vertex in D'. Then, as $G'[D']$ is connected and $D' \subseteq D''$, it follows that $G'[D'']$ is connected too. Further, $d_v(D, D'') = |D'' \setminus D| + |D \setminus D''| = |D'' \setminus D| = |D' \setminus D| \leq r' \leq r$. ◀

Now, we describe an algorithm by reducing the problem to finding a Group Steiner tree.

▶ **Theorem 7.2.** Dynamic Connected Dominating Set *admits an algorithm that runs in $\mathcal{O}^*(2^k)$ time.*

Proof. Consider an instance (G, G', D, k, r) of Dynamic Connected Dominating Set. Assume G' is connected. Clearly, the edges in $E(G') \setminus E(G)$ do not affect the solution. Partition $V(G') \setminus D$ into two sets $C = N_{G'}(D)$ and $B = V(G') \setminus C$. Contract each connected component of $G'[D]$ to a single vertex. Let H denote the resulting graph and let $X = V(H) \setminus V(G')$. Construct an instance $(H, |X| + r, \mathcal{F})$ of Group Steiner Tree where

$\mathcal{F} = \{N_{G'}[v] \mid v \in B\} \cup \{\{x\} \mid x \in X\}$. We claim that (G, G', D, k, r) is a YES instance of Dynamic Connected Dominating Set if and only if $(H, |X| + r, \mathcal{F})$ is a YES instance of Group Steiner Tree.

Suppose there exists a connected dominating set D' of G' such that $d_v(D, D') \leq r$ and $D \subseteq D'$. For every vertex u in B, there is a vertex x in $D' \cap (B \cup C)$ that is adjacent to u. As $G'[D']$ is connected, it follows that $H[X \cup (D' \cap (B \cup C))]$ is also connected. Moreover, as $|D' \cap (C \cup B)| \leq |D'| - |D| \leq r$, it follows that the spanning tree of $H[X \cup (D' \cap (C \cup B))]$ is of size at most $|X| + r$. Hence $(H, |X| + r, \mathcal{F})$ is a YES instance of Group Steiner Tree. Suppose $(H, |X| + r, \mathcal{F})$ is a YES instance of Group Steiner Tree. Let T denote the solution tree of H. Then, $X \subseteq V(T)$ and $|V(T) \setminus X| = |V(T) \cap (C \cup B)| \leq r$. Define $D' = D \cup (V(T) \cap (B \cup C))$. The size of D' is at most $|D| + r$. Now, $G'[D']$ is connected as D' is obtained from the vertices of T. Also, for every vertex u in B, T contains at least one vertex in $N_{G'}[v]$. Thus, D' is the desired connected dominating set of G'.

As $E(G) \setminus E(G')$, the sum of the number of connected components of $G'[D]$ and the size of B is upper bounded by $k + 1$. That is, $|\mathcal{F}| \leq k + 1$ and the Group Steiner Tree algorithm of [20] runs in $\mathcal{O}^*(2^k)$ time. ◀

Finally, by a reduction from Set Cover to Dynamic Connected Dominating Set, we show the following result.

▶ **Theorem 7.3.** Dynamic Connected Dominating Set *does not admit an algorithm with* $\mathcal{O}^*((2 - \epsilon)^k)$ *running time for any* $\epsilon > 0$ *assuming the Set Cover Conjecture.*

Proof. We observe that the reduction described in Theorem 6.3 reduces instances of Set Cover to instances of Dynamic Connected Dominating Set. Thus, the claimed lower bound holds assuming the Set Cover Conjecture. ◀

8 Conclusion

We described FPT algorithms for the dynamic variants of several classical parameterized problems with respect to the edit parameter. The role of structural parameters like treewidth and pathwidth in this setting remains to be explored. Also, further exploration of the contrast between the parameterized complexity of a problem and its dynamic version is an interesting direction of research.

Acknowledgments. We are grateful to Saket Saurabh for the invaluable discussions and for providing several useful pointers that led to the writing of this paper. We are also thankful to the reviewers for their suggestions that strengthened the results for Dynamic Π-Deletion.

References

1 F. N. Abu-Khzam, J. Egan, M. R. Fellows, F. A. Rosamond, and P. Shaw. On the Parameterized Complexity of Dynamic Problems. *Theoretical Computer Science*, 607 (3):426–434, 2015.

2 L. Cai. Fixed-Parameter Tractability of Graph Modification Problems for Hereditary Properties. *Information Processing Letters*, 58(4):171–176, 1996.

3 J. Chen, I. A. Kanj, and W. Jia. Vertex Cover: Further Observations and Further Improvements. *Journal of Algorithms*, 41(2):280–301, 2001.

4 J. Chen, I. A. Kanj, and G. Xia. Improved Upper Bounds for Vertex Cover. *Theoretical Computer Science*, 411(40-42):3736–3756, 2010.

5 M. Cygan, H. Dell, D. Lokshtanov, D. Marx, J. Nederlof, Y. Okamoto, R. Paturi, S. Saurabh, and M. Wahlstrom. On Problems As Hard As CNF-SAT. In *Proceedings of the IEEE Conference on Computational Complexity*, CCC'12, pages 74–84. IEEE Computer Society, 2012.

6 M. Cygan, F. V. Fomin, K. Lukasz, D. Lokshtanov, D. Marx, M. Pilipczuk, M. Pilipczuk, and S. Saurabh. *Parameterized Algorithms.* Springer, 2015.

7 M. Cygan, J. Nederlof, M. Pilipczuk, M. Pilipczuk, J. M. M. van Rooij, and J. O. Wojtaszczyk. Solving Connectivity Problems Parameterized by Treewidth in Single Exponential Time. In *Proceedings of the IEEE* 52^{nd} *Annual Symposium on Foundations of Computer Science*, FOCS'11, pages 150–159, 2011.

8 R. Diestel. *Graph Theory.* Springer-Verlag Berlin Heidelberg, 2006.

9 R. G. Downey, J. Egan, M. R. Fellows, F. A. Rosamond, and P. Shaw. Dynamic Dominating Set and Turbo-Charging Greedy Heuristics. *Tsinghua Science and Technology*, 19(4):329–337, 2014.

10 R. G. Downey and M. R. Fellows. *Fundamentals of Parameterized Complexity.* Springer-Verlag London, 2013.

11 J. Flum and M. Grohe. *Parameterized Complexity Theory.* Springer, 2006.

12 F. V. Fomin, S. Gaspers, D. Lokshtanov, and S. Saurabh. Exact Algorithms via Monotone Local Search. In *Proceedings of the* 48^{th} *Annual ACM Symposium on Theory of Computing*, STOC'16. ACM, 2016.

13 F. V. Fomin, S. Gaspers, S. Saurabh, and A. A. Stepanov. On Two Techniques of Combining Branching and Treewidth. *Algorithmica*, 54(2):181–207, 2009.

14 F. V. Fomin, D. Kratsch, and G. J. Woeginger. Exact (exponential) Algorithms for the Dominating Set Problem. In *Proceedings of the* 30^{th} *International Workshop on Graph-Theoretic Concepts in Computer Science*, WG'04, pages 245–256. Springer Berlin Heidelberg, 2004.

15 S. Hartung and R. Niedermeier. Incremental List Coloring of Graphs, Parameterized by Conservation. *Theoretical Computer Science*, 494:86–98, 2013.

16 S. Khot and V. Raman. Parameterized Complexity of Finding Subgraphs with Hereditary Properties. *Theoretical Computer Science*, 289(2):997–1008, 2002.

17 J. Kneis, D. Mölle, S. Richter, and P. Rossmanith. A Bound on the Pathwidth of Sparse Graphs with Applications to Exact Algorithms. *SIAM Journal on Discrete Mathematics*, 23(1):407–427, 2009.

18 T. Kociumaka and M. Pilipczuk. Faster Deterministic Feedback Vertex Set. *Information Processing Letters*, 114(10):556–560, 2014.

19 J. M. Lewis and M. Yannakakis. The Node-Deletion Problem for Hereditary Properties is NP-Complete. *Journal of Computer and System Sciences*, 20(2):219–230, 1980.

20 N. Misra, G. Philip, V. Raman, S. Saurabh, and S. Sikdar. FPT Algorithms for Connected Feedback Vertex Set. *Journal of Combinatorial Optimization*, 24(2):131–146, 2012.

21 S. Thomassé. A Quadratic Kernel for Feedback Vertex Set. In *Proceedings of the* 12^{th} *Annual ACM-SIAM Symposium on Discrete Algorithms*, SODA'09, pages 115–119, 2009.

22 M. Xiao and H. Nagamochi. Exact Algorithms for Maximum Independent Set. In *Proceedings of the* 24^{th} *International Symposium on Algorithms and Computation*, ISAAC'13, pages 328–338. Springer Berlin Heidelberg, 2013.