Decremental Optimization of Dominating Sets Under the Reconfiguration Framework

Alexandre Blanché[1], Haruka Mizuta[2], Paul Ouvrard[1(✉)], and Akira Suzuki[2]

[1] Univ. Bordeaux, Bordeaux INP, CNRS, LaBRI, UMR5800, 33400 Talence, France
{alexandre.blanche,paul.ouvrard}@u-bordeaux.fr
[2] Graduate School of Information Sciences, Tohoku University, Aoba 6-6-05, Aramaki-aza, Aoba-ku, Sendai, Miyagi 980-8579, Japan
haruka.mizuta.s4@dc.tohoku.ac.jp, a.suzuki@ecei.tohoku.ac.jp

Abstract. Given a dominating set, how much smaller a dominating set can we find through elementary operations? Here, we proceed by iterative vertex addition and removal while maintaining the property that the set forms a dominating set of bounded size. This can be seen as the optimization variant of the dominating set reconfiguration problem, where two dominating sets are given and the question is merely whether they can be reached from one another through elementary operations. We show that this problem is PSPACE-complete, even if the input graph is a bipartite graph, a split graph, or has bounded pathwidth. On the positive side, we give linear-time algorithms for cographs, trees and interval graphs. We also study the parameterized complexity of this problem. More precisely, we show that the problem is W[2]-hard when parameterized by the upper bound on the size of an intermediary dominating set. On the other hand, we give fixed-parameter algorithms with respect to the minimum size of a vertex cover, or $d + s$ where d is the degeneracy and s is the upper bound of the output solution.

Keywords: Combinatorial reconfiguration · Dominating set · Parameterized complexity

1 Introduction

Recently, *Combinatorial reconfiguration* [11] has been extensively studied in the field of theoretical computer science (See, e.g., surveys [10,18]). A reconfiguration problem is generally defined as follows: we are given two feasible solutions of

Partially supported by JSPS and MAEDI under the Japan-France Integrated Action Program (SAKURA). The first and third author is partially supported by ANR project GrR (ANR-18-CE40-0032). The second author is partially supported by JSPS KAKENHI Grant Number JP19J10042, Japan. The third author is partially supported by ANR project GraphEn (ANR-15-CE40-0009). The fourth author is partially supported by JST CREST Grant Number JPMJCR1402, and JSPS KAKENHI Grant Numbers JP17K12636 and JP18H04091, Japan.
Full version available at https://arxiv.org/abs/1906.05163.

L. Gąsieniec et al. (Eds.): IWOCA 2020, LNCS 12126, pp. 69–82, 2020.
https://doi.org/10.1007/978-3-030-48966-3_6

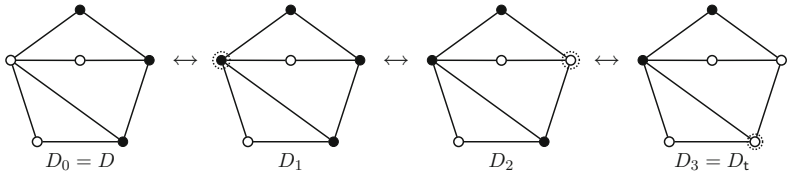

Fig. 1. Reconfiguration sequence between D_0 and D_3 via dominating sets D_1, D_2 with upper bound $k = 4$, where vertices contained in a dominating set are depicted by black circles, and added or removed vertices are surrounded by dotted circles.

a combinatorial search problem, and asked to determine whether we can transform one into the other via feasible solutions so that all intermediate solutions are obtained from the previous one by applying the specified reconfiguration rule. This framework is applied to several well-studied combinatorial search problems; for example, Independent Set [3,9,13,14], Vertex Cover [16,17], Dominating Set [8,15,17,19], and so on.

The Dominating Set Reconfiguration problem is one of the well-studied reconfiguration problems. For a graph $G = (V, E)$, a vertex subset $D \subseteq V$ is called a *dominating set* of G if D contains at least one vertex in the closed neighborhood of each vertex in V. Figure 1 illustrates four dominating sets of the same graph. Suppose that we are given two dominating sets D_0 and D_t of a graph whose cardinalities are at most a given upper bound k. Then the Dominating Set Reconfiguration problem asks to determine whether we can transform D_0 into D_t via dominating sets of cardinalities at most k such that all intermediate ones are obtained from the previous one by adding or removing exactly one vertex. Note that this reconfiguration rule, i.e. adding or removing exactly one vertex while keeping the cardinality constraint, is called *the token addition and removal* (TAR) rule. Figure 1 illustrates an example of transformation between two dominating sets D_0 and D_3 for an upper bound $k = 4$.

Combinatorial reconfiguration models "dynamic" transformations of systems, where we wish to transform the current configuration of a system into a more desirable one by a step-by-step transformation. In the current framework of combinatorial reconfiguration, we need to have in advance a target (a more desirable) configuration. However, it is sometimes hard to decide a target configuration, because there may exist exponentially many desirable configurations. Based on this situation, Ito *et al.* introduced the new framework of reconfiguration problems, called *optimization variant* [12]. In this variant, we are given a single solution as a current configuration, and asked for a more "desirable" solution reachable from the given one. This variant was introduced very recently, hence it has only been applied to Independent Set Reconfiguration to the best of our knowledge. Therefore and since Dominating Set Reconfiguration is one of the well-studied reconfiguration problems as we already said, we focus on this problem and study it under this framework.

1.1 Our Problem

In this paper, we study the optimization variant of Dominating Set Reconfiguration, denoted by OPT-DSR. To avoid confusion, we call the original Dominating Set Reconfiguration the *reachability variant*, and we denote it by REACH-DSR. Suppose that we are given a graph G, two integers k, s, and a dominating set D of G whose cardinality is at most k; we call k an *upper bound* and s a *solution size*. Then OPT-DSR asks for a dominating set D_{t} satisfying the following two conditions: (a) the cardinality of D_{t} is at most s, and (b) D_{t} can be transformed from D under the TAR rule with upper bound k. For example, if we are given a dominating set D_0 in Fig. 1 and two integers $k = 4$ and $s = 2$, then one of the solutions is D_3, because D_3 can be transformed from D_0 and $|D_3| \leq 2$ holds.

1.2 Related Results

Although OPT-DSR is being introduced in this paper, some results for REACH-DSR relate to OPT-DSR in the sense that the techniques to show the computational hardness or construct an algorithm will be used in our proof for OPT-DSR. We thus list such results for REACH-DSR in the following.

There are several results for the polynomial-time solvability of REACH-DSR. Haddadan *et al.* [8] showed that REACH-DSR under TAR rule is PSPACE-complete for split graphs, for bipartite graphs, and for planar graphs, while linear-time solvable for interval graphs, for cographs, and for forests. REACH-DSR is also studied well from the viewpoint of fixed-parameter (in)tractability. Mouawad *et al.* [17] showed that REACH-DSR under TAR is W[2]-hard when parameterized by an upper bound k. As a positive result, Lokshtanov *et al.* [15] gave a fixed-parameter algorithm with respect to $k + d$ for graphs that exclude $K_{d,d}$ as a subgraph.

Ito *et al.* studied an optimization variant of Independent Set Reconfiguration (denoted OPT-ISR) [12]. More precisely, they proved that this problem is PSPACE-hard on bounded pathwidth, NP-hard on planar graphs, while linear-time solvable on chordal graphs. They also gave an XP-algorithm with respect to the solution size, and a fixed-parameter algorithm with respect to both solution size and degeneracy.

1.3 Our Results

In this paper, we study OPT-DSR from the viewpoint of the polynomial-time (in)tractability and fixed-parameter (in)tractability.

We first study the polynomial-time solvability of OPT-DSR with respect to graph classes (See Fig. 2). Specifically, we show that the problem is PSPACE-complete even for split graphs, for bipartite graphs, and for bounded pathwidth graphs, and NP-hard for planar graphs with bounded maximum degree. On the other hand, the problem is linear-time solvable for cographs, trees and interval graphs. The inclusions of these graph classes are represented in Fig. 2.

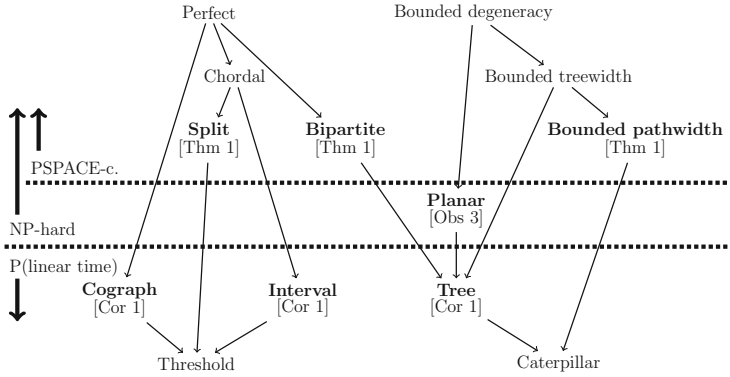

Fig. 2. Our results for polynomial-time solvability with respect to graph classes, where A → B means that the class A contains the class B.

We then study the fixed-parameter (in)tractability of OPT-DSR. We first focus on the following four graph parameters: the degeneracy d, the maximum degree Δ, the pathwidth pw, and the vertex cover number τ (that is the size of a minimum vertex cover). Figure 3(a) illustrates the relationship between these parameters, where A → B means that the parameter A is bounded by some function of B. This relation implies that if we have a result stating that OPT-DSR is fixed-parameter tractable for A then the tractability for B follows, while if we have a negative (i.e. intractability) result for B then it extends to A. From results for polynomial-time solvability, we show the PSPACE-completeness for fixed pw and NP-hardness for fixed Δ, and hence the problem is fixed-parameter intractable for each parameter pw, Δ and d under P ≠ PSPACE or P ≠ NP. As a positive result, we give an FPT algorithm for τ. We then consider two input parameters: the solution size s and the upper bound k. (See Fig. 3(c).) We show that OPT-DSR is W[2]-hard when parameterized by k. We note that we can assume without loss of generality that $s < k$ holds, as explained in Sect. 2. Therefore, it immediately implies W[2]-hardness for s. Most single parameters (except for τ) cause a negative (intractability) result. We thus finally consider combinations of one graph parameter and one input parameter. We give an FPT algorithm with respect to $s+d$. (See Fig. 3(b).) In the end, we can conclude from the discussion above that for any combination of a graph parameter $p \in \{d, \Delta, pw, \tau\}$ and an input parameter $q \in \{s, k\}$, OPT-DSR is fixed-parameter tractable when parameterized by $p + q$. Due to space limitations, proofs of statements marked with (*) have been omitted (see the arXiv version).

2 Preliminaries

For a graph G, we denote by $V(G)$ and $E(G)$ the vertex set of G and edge set of G, respectively. For a vertex $v \in V(G)$, we let $N_G(v) = \{w \mid vw \in E(G)\}$ and $N_G[v] = N_G(v) \cup \{v\}$; we call a vertex in $N_G(v)$ a *neighbor* of v in G. For a

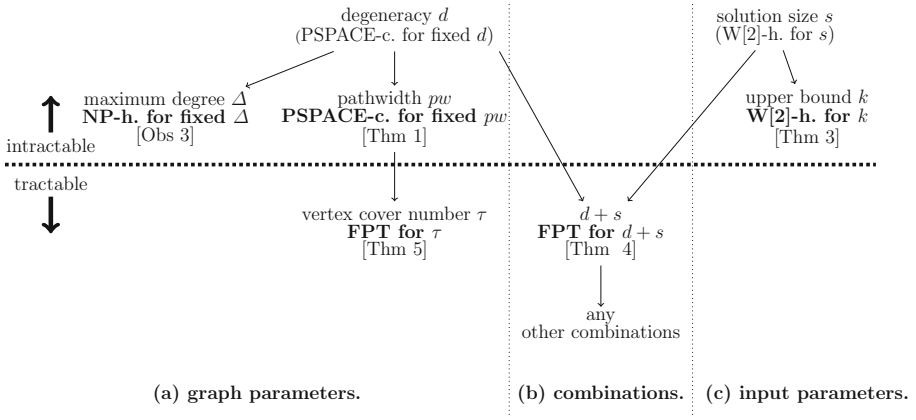

Fig. 3. Our results for fixed-parameter tractability, where A → B means that the parameter A is bounded on some function of B.

vertex subset $S \subseteq V(G)$, we let $N_G[S] = \bigcup_{v \in S} N_G[v]$. If there is no confusion, we sometimes omit G from the notation.

2.1 Optimization Variant of Dominating Set Reconfiguration

For a graph $G = (V, E)$, a vertex subset $D \subseteq V$ is a *dominating set* of G if $N[D] = V(G)$. For a dominating set D, we say that $u \in D$ *dominates* $v \in V$ if $v \in N[u]$ holds. We say that a vertex $v \in D$ has a *private neighbor* in D if there exists a vertex $u \in N[v]$ such that $N[u] \cap D = \{v\}$. In other words, the vertex u is dominated only by v in D. Note that the private neighbor of a vertex can be itself. A dominating set is (inclusion-wise) *minimal* if and only if each of its vertices has a private neighbor, and *minimum* if and only if the cardinality is minimum among all dominating sets. Notice that any minimum dominating set is minimal.

Let D and D' be two dominating sets of G. We say that D and D' are *adjacent* if $|D \Delta D'| = 1$, where $D \Delta D' = (D \setminus D') \cup (D' \setminus D)$ and we denote this by $D \leftrightarrow D'$. Let us now assume that D and D' are both of size at most k, for some given $k \geq 0$. Then, a *reconfiguration sequence* between D and D' under the TAR rule (or sometimes called a TAR-sequence) is a sequence $\langle D = D_0, D_1, \ldots, D_\ell = D' \rangle$ of dominating sets of G such that:

- for each $i \in \{0, 1, \ldots, \ell\}$, D_i is a dominating set of G such that $|D_i| \leq k$; and
- for each $i \in \{0, 1, \ldots, \ell - 1\}$, $D_i \leftrightarrow D_{i+1}$ holds.

Considering a reconfiguration sequence under the TAR rule, we sometimes write TAR(k) instead of TAR to emphasize the upper bound k on the size of a solution. We say that D' is *reachable from* D if there exists a reconfiguration sequence between D and D'; since a reconfiguration sequence is reversible, if D' is reachable from D, then D is also reachable from D'. We write $D \overset{k}{\leftrightsquigarrow} D'$ if D' (resp.

D) is reachable from D (resp. D'). Then, the *optimization variant* of the Dominating Set Reconfiguration problem (OPT-DSR) is defined as follows:

OPT-DSR

Instance: A graph G, two integers $k, s \geq 0$, a dominating set D of G such that $|D| \leq k$.

Question: A dominating set D_t of G such that $|D_\mathsf{t}| \leq s$ and $D \overset{k}{\leftrightsquigarrow} D_\mathsf{t}$ if it exists, no-instance otherwise.

We denote by a 4-tuple (G, k, s, D) an instance of OPT-DSR.

2.2 Useful Observations

From the definition of OPT-DSR, we have the following observations.

Observation 1. *Let (G, k, s, D) be an instance of OPT-DSR. If k, s and $|D|$ violate the inequality $s < |D| \leq k$, then D is a solution of the instance.*

Proof. By the definition of D, we know $|D| \leq k$. Therefore if the inequality is violated, we have $|D| \leq s \leq k$ or $|D| \leq k \leq s$. In both cases, $|D| \leq s$ holds, and hence D is a solution. □

It is observed that the condition in Observation 1 can be checked in linear time. Therefore, we sometimes assume without loss of generality that $s < |D| \leq k$ holds. Then, another observation follows.

Observation 2. *Let (G, k, s, D) be an instance of OPT-DSR such that $s < |D|$ holds. If D is minimal and $|D| = k$ holds, then the instance has no solution.*

Proof. Since $|D| = k$, we cannot add any vertex to D without exceeding the threshold k. Besides, since D is minimal, we cannot remove any vertex while maintaining the domination property. As a result, there is no dominating set D_t of size at most s reachable from D i.e. $D \overset{k}{\leftrightsquigarrow} D_\mathsf{t}$ does not hold for any dominating set D_t such that $|D_\mathsf{t}| \leq s$. □

Again, the conditions in Observation 2 can be checked in linear time, and hence we can assume without loss of generality that D is not minimal or $|D| < k$ holds. Suppose that D is not minimal. Then we can always obtain a dominating set of size less than k by removing some vertex without private neighbor from D, that is, we have a dominating set D' with $D \overset{k}{\leftrightsquigarrow} D'$ and $|D'| < k$. Note that (G, k, s, D) has a solution if and only if (G, k, s, D') does. Therefore, it suffices to consider the case where $|D| < k$ holds. Combining it with Observation 1, we sometimes assume without loss of generality that $s < |D| < k$ holds.

Finally, we have the following observation which states that OPT-DSR is a generalization of the Dominating Set Problem:

Observation 3. *Let $G = (V, E)$ be a graph and s be an integer. Then the instance $(G, |V|, s, V)$ of OPT-DSR is equivalent to finding a dominating set of G of size at most s.*

Proof. Let D_t be a dominating set of G of size at most s. Since we started from a dominating set containing all the vertices of G, it is sufficient to remove one by one each vertex in $V \setminus D_t$ to reach D_t. □

Observation 3 implies that hardness results for the original Dominating Set problem extend to OPT-DSR. In particular, we get that OPT-DSR is NP-hard even for the case where the input graph has maximum degree 3, or is planar with maximum degree 4 [7]. However, we will show in Sect. 3.1 that this problem is actually PSPACE-complete.

3 Polynomial-Time (In)tractability

3.1 PSPACE-Completeness for Several Graph Classes

Theorem 1. OPT-DSR *is PSPACE-complete even when restricted to bounded pathwidth graphs, for split graphs, and for bipartite graphs.*

First, observe that OPT-DSR is in PSPACE. Indeed, when we are given a dominating set D_t as a solution for some instance of OPT-DSR, we can check in polynomial time whether it has size at most s or not. Furthermore, since REACH-DSR is in PSPACE, we can check in polynomial space whether it is reachable from the original dominating set D. Therefore, we can conclude that OPT-DSR is in PSPACE.

We now give three reductions to show the PSPACE-hardness for split graphs, bipartite graphs and bounded pathwidth graphs, respectively. These reductions are slight adaptations of the ones of PSPACE-hardness for REACH-DSR developed in [8]. We only give the hardness proof for split graphs (see the arXiv version for the two other proofs). To this end, we use a polynomial-time reduction from the optimization variant of Vertex Cover Reconfiguration, denoted by OPT-VCR.

Given a graph $G = (V, E)$, a *vertex cover* is a subset of vertices that contains at least one endpoint of each edge in E. We now give the formal definition of OPT-VCR. Suppose that we are given a graph G, two integers $k, s \geq 0$, and a vertex cover C of G whose cardinality is at most k. Then OPT-VCR asks for a vertex cover C_t of size at most s reachable from C under the $\mathsf{TAR}(k)$ rule. This problem is known to be PSPACE-complete even for bounded pathwidth graphs[1] [12].

Lemma 1. OPT-DSR *is PSPACE-hard even for split graphs.*

[1] In [12], Ito *et al.* actually showed the PSPACE-completeness for the optimization variant of Independent Set Reconfiguration. However, the result can easily be converted to OPT-VCR from the observation that any vertex cover of a graph is the complement of an independent set.

(a) Original graph G' with vertex cover $\{v_2, v_4\}$. (b) Corresponding split graph G.

Fig. 4. Reduction for Lemma 1. Note that $\{v_2, v_4\}$ is a dominating set of G.

Proof. As we said, we give a polynomial-time reduction from OPT-VCR. More precisely, we extend the idea developed for the NP-hardness proof of Dominating Set problem on split graphs [2].

Let (G', k', s', C) be an instance of OPT-VCR with $V(G') = \{v_1, v_2, \ldots, v_n\}$ and $E(G') = \{e_1, e_2, \ldots, e_m\}$. We construct the corresponding split graph G as follows (see also Fig. 4). Let $V(G) = A \cup B$, where $A = V(G')$ and $B = \{w_1, w_2, \ldots, w_m\}$; the vertex $w_i \in B$ corresponds to the edge $e_i \in E(G')$. We join all pairs of vertices in A so that A forms a clique in G. In addition, for each edge $e_i = v_p v_q$ in $E(G')$, we join $w_i \in B$ with each of v_p and v_q. Let G be the resulting graph, and let $(G, k = k', s = s', D = C)$ be the corresponding instance of OPT-DSR (we will prove later that D is a dominating set of G). Clearly, this instance can be constructed in polynomial time. It remains to prove that (G', k', s', C) is a yes-instance if and only if (G, k, s, D) is a yes-instance.

We start by the only-if direction. Suppose that (G', k', s', C) is a yes-instance. Then, there exists a vertex cover C_t of size at most s' reachable from C under the TAR(k') rule. Since $k' = k$, $s = s'$ and both problems employ the same reconfiguration rule, it suffices to prove that any vertex cover of G' is a dominating set of G. Since $C \subseteq V(G') = A$ and A is a clique, all vertices in $A \setminus C$ are dominated by the vertices in C. Thus, consider a vertex $w_i \in B$, which corresponds to the edge $e_i = v_p v_q$ in $E(G')$. Then, since C is a vertex cover of G', at least one of v_p and v_q must be contained in C. This means that w_i is dominated by the endpoint v_p or v_q in G. Therefore, each vertex cover in the reconfiguration sequence between C and C_t is a dominating set of G (including $D = C$ and $D_\mathsf{t} = C_\mathsf{t}$) and thus, (G, k, s, D) is a yes-instance.

We now focus on the if direction. Suppose that (G, k, s, D) is a yes-instance. Then, there exists a dominating set D_t of G of size at most s reachable under the TAR(k) rule by a sequence $\mathcal{R} = \langle D_0, D_1, \ldots, D_\mathsf{t} \rangle$, with $D = D_0$. Recall that $D = C$ and thus D is a vertex cover of G'. We want to produce a sequence of dominating sets that are subsets of A. To this end, we proceed by eliminating the vertices of B that appears in $\mathcal{R}$ one by one from the sequence. Let i be the smallest index such that $D_i \in \mathcal{R}$ contains a vertex $w \in B$ associated to the edge $v_a v_b \in E(G)$. Let $j \geq i$ be the largest index such that every dominating set $D_l \in \mathcal{R}$ $(i \leq l \leq j)$ contains w. Now we show that D_{j+1} is reachable from D_{i-1} under TAR(k) rule without touching w, that is, there is a sequence where each

dominating set in the sequence does not contain w. For every $D_l \in \mathcal{R}$ $(i \leq l \leq j)$ we instead consider the set $D'_l = (D_l \setminus w) \cup \{v_a\}$. Note that $v_a \in N_G(w)$, and $|D'_l| \leq |D_l| \leq k$. Observe that each D'_l is a dominating set since $N_G[w] \subseteq N_G[v_a]$. If $v_a \in D_{i-1}$, then $D_{i-1} = D'_i$. Otherwise, D'_i is obtainable from D_{i-1} in one step since we just replace the addition of w by the one of v_a. Moreover, due to the choice of j, $D_{j+1} = D_j \setminus \{w\}$. Hence, D_{j+1} contains a vertex in A adjacent to w. If this vertex is v_a, $D'_j = D_{j+1}$. Otherwise, $D_{j+1} = D'_j \setminus \{v_a\}$, which corresponds to a valid TAR move. Finally, since we ensure that each dominating set D'_l with $i \leq l \leq j$ contains v_a, we can ignore each move in the subsequence of $\mathcal{R}$ that touches v_a. Hence, either $D'_l = D'_{l+1}$ or $D'_l \leftrightarrow D'_{l+1}$ holds, for every $i \leq l < j$. By ignoring duplicates from the sequence $\langle D_{i-1}, D'_i, \ldots, D'_j, D_{j+1} \rangle$, we obtain a desired subsequence which does not touch w. Therefore, we can eliminate w in the subsequence $\langle D_{i-1}, D_i, \ldots, D_j, D_{j+1} \rangle$ of $\mathcal{R}$ by replacing it with the desired subsequence. Hence by repeating this process for each subsequence containing w we get a new sequence that does not touch w at all. We then repeat this process for every vertex of B that appears in $\mathcal{R}$ and we obtain a sequence $\mathcal{R}'$ where each dominating set is a subset of A. Finally, observe that any dominating set D of G such that $D \subseteq A = V(G')$ forms a vertex cover of G', because each vertex $w_i \in B$ is dominated by at least one vertex in $D \subseteq V(G')$. Therefore, (G', s', k', C) is a yes-instance. □

Finally, the two following lemmas complete the proof of Theorem 1.

Lemma 2 (*). OPT-DSR *is PSPACE-hard even for bounded pathwidth graphs.*

Lemma 3 (*). OPT-DSR *is PSPACE-hard even for bipartite graphs.*

3.2 Linear-Time Algorithms

We now explain how OPT-DSR can be solved in linear time for several graph classes. To this end, we deal with the concept of a canonical dominating set. A dominating set D_c is *canonical* if D_c is a minimum dominating set which is reachable from any dominating set D under the TAR$(|D|+1)$ rule. Then we have the following theorem.

Theorem 2. *Let $\mathcal{G}$ be a class of graphs such that any graph $G \in \mathcal{G}$ has a canonical dominating set and we can compute it in linear time. Then OPT-DSR can be solved in linear time on $\mathcal{G}$.*

Proof. Let (G, k, s, D) be an instance of OPT-DSR, where $G \in \mathcal{G}$. Recall that we can assume without loss of generality that $s < |D| < k$; we can check in linear time whether the inequality is satisfied or not, and if it is violated, then we know from Observation 1 and 2 that it is a trivial instance. Since $G \in \mathcal{G}$, G admits a canonical dominating set and we can compute in linear time an actual one. Let D_c be such a canonical dominating set. Then it follows from the definition that D_c is reachable from D under the TAR(k) rule since $k \geq |D| + 1$.

Since D_c is a minimum dominating set, we can output it if $|D_c| \leq s$ holds, and no-instance otherwise. All processes can be done in linear time, and hence the theorem follows. □

Haddadan *et al.* showed in [8] that cographs, trees (actually, forests), and interval graphs admit a canonical dominating set. Their proofs are constructive, and hence we can find an actual canonical dominating set. It is observed that the constructions on cographs and trees can be done in linear time. The construction on interval graphs can also be done in linear time with a nontrivial adaptation by using an appropriate data structure. Therefore, we have the following linear-time solvability of OPT-DSR.

Corollary 1. *OPT-DSR can be solved in linear time on cographs, trees, and interval graphs.*

4 Fixed-Parameter (In)tractability

In this section, we study the fixed-parameter complexity of OPT-DSR with respect to several graph parameters. More precisely, we first show that OPT-DSR is W[2]-hard when parameterized by the upper bound k. To prove it, we use the idea of the reduction constructed by Mouawad *et al.* to show the W[2]-hardness of REACH-DSR [17].

Theorem 3 (*). *OPT-DSR is W[2]-hard when parameterized by the upper bound k.*

On the other hand, we give FPT algorithms with respect to the combination of the solution size s and the degeneracy d in Subsect. 4.1 and the vertex cover number τ in Subsect. 4.2.

4.1 FPT Algorithm for Degeneracy and Solution Size

The following is the main theorem in this subsection.

Theorem 4. *OPT-DSR is fixed-parameter tractable when parameterized by $d + s$, where d is the degeneracy and s the solution size.*

To prove the theorem, we give an FPT algorithm with respect to $d + s$. Note that our algorithm uses the idea of an FPT algorithm solving the reachability variant of Dominating Set Reconfiguration, developed by Lokshtanov *et al.* [15]. Their algorithm uses the concept of domination core; for a graph G, a *domination core* of G is a vertex subset $C \subseteq V(G)$ such that any vertex subset $D \subseteq V(G)$ is a dominating set of G if and only if $C \subseteq N_G[D]$ [6].

Suppose that we are given an instance (G, k, s, D) of OPT-DSR where G is a d-degenerate graph. By Observation 2, we can assume without loss of generality that $|D| < k$. We first check whether G has a dominating set of size at most s: this can be done in FPT$(d + s)$ time for d-degenerate graphs [1]. If G does not have it, then we can instantly conclude that this is a no-instance.

In the remainder of this subsection, we assume that G has a dominating set of size at most s. In this case, we kernelize the instance: we shrink G by removing some vertices while keeping the existence of a solution until the size of the graph only depends on d and s. To this end, we use the concept of domination core.

Lemma 4 (Lokshtanov *et al.* [15]). *If G is a d-degenerate graph and G has a dominating set of size at most s, then G has a domination core of size at most ds^d and we can find it in FPT$(d+s)$ time.*

Therefore, one can compute a domination core of G of size at most ds^d in FPT$(d+s)$ time by Lemma 4. In order to shrink G, we use the reduction rule **R1:** if there is a domination core C and two vertices $v_r, v_l \in V(G) \setminus C$ such that $N_G(v_r) \cap C \subseteq N_G(v_l) \cap C$, we remove v_r. We need to prove that **R1** is "safe", that is, we can remove v_r from G without changing the existence of a solution. However, if the input dominating set D contains v_r, we cannot do it immediately. Therefore, we first remove v_r from D.

Lemma 5 (*). *Let D be a dominating set such that both $|D| < k$ and $v_r \in D$ hold. Then there exists D' such that $v_r \notin D'$ and $D \overset{k}{\rightsquigarrow} D'$, and D' can be computed in linear time.*

We can now redefine D as a dominating set which does not contain v_r. We then consider removing v_r from G. Let $G' = G[V(G) \setminus \{v_r\}]$. The following lemma ensures that removing v_r keeps the existence of a solution.

Lemma 6 (*). *Let (G, k, s, D) be an instance where $v_r \notin D$. Then, (G, k, s, D) has a solution if and only if (G', k, s, D) has a solution.*

We exhaustively apply the reduction rule **R1** to shrink G. Let G_k and D_k be the resulting graph and dominating set, respectively. Then, any two vertices $u, v \in V(G_k) \setminus C$ satisfy $N_{G_k}(u) \cap C \neq N_{G_k}(v) \cap C$ (more precisely, $N_{G_k}(u) \cap C \not\subseteq N_{G_k}(v) \cap C$). Then the following lemma completes the proof of Theorem 4.

Lemma 7. *(G_k, k, s, D_k) can be solved in FPT$(d+s)$ time.*

Proof. We first show that the size of the vertex set of G_k is at most $f(d,s) = ds^d + 2^{ds^d}$. Since $|C| \leq ds^d$, it suffices to show that $|V(G_k) \setminus C| \leq 2^{ds^d}$ holds. Recall that any two vertices $u, v \in V(G_k) \setminus C$ satisfy $N_{G_k}(u) \cap C \neq N_{G_k}(v) \cap C$. Then since the number of combination of vertices in C is at most $2^{|C|} \leq 2^{ds^d}$, we have the desired upper bound $|V(G_k) \setminus C| \leq 2^{ds^d}$.

We now prove that (G_k, k, s, D_k) can be solved in FPT$(d+s)$ time. To this end, we construct an *auxiliary graph* G_A, where the vertex set of G_A is the set of all dominating sets of G_k, and any two nodes (that correspond to dominating sets of G_k) D and D' in G_A are adjacent if and only if $|D \Delta D'| = 1$ holds. Let $n = |V(G_k)|$ and $m = |E(G_k)|$. Then the number of candidate nodes in G_A (vertex subsets of G_k) is bounded by $O(2^n)$. For each candidate, we can check in

$O(n+m)$ time if it forms a dominating set. Thus we can construct the vertex set of G_A in $O(2^n(n+m))$ time. We then construct the edge set of G_A. There are at most $O(|V(G_A)|^2) = O(4^n)$ pairs of nodes in G_A. For each pair of nodes, we can check in $O(n)$ time if their corresponding dominating sets differ in exactly one vertex. Therefore we can construct the edge set of G_A in $O(4^n n)$ time, and hence the total time to construct G_A is $O(4^n n + 2^n(n+m))$ time. We finally search a solution by running a breadth-first search algorithm from D_k on G_A in $O(|V(G_A)| + |E(G_A)|) = O(4^n)$ time.

We can conclude that our algorithm runs in time $O(4^n n + 2^n(n+m))$ in total. Since $n \leq f(d,s)$ and $m \leq n^2 \leq (f(d,s))^2$, this is an FPT time algorithm. □

4.2 FPT Algorithm for Vertex Cover Number

Let (G,k,s,D) be an instance of OPT-DSR. As in the previous section, we may first assume by Observation 2 that $|D| < k$. We recall that $\tau(G)$ is the size of a minimum vertex cover of G. In order to lighten notations, we simply denote by τ the vertex cover number of the input graph. Then, we have the following:

Theorem 5. *OPT-DSR is fixed-parameter tractable when parameterized by τ.*

Observation 4 (*). *If G is d-degenerate, then $d \leq \tau$.*

We are now able to get down to the proof of Theorem 5, by providing an algorithm that solves OPT-DSR and runs in time FPT(τ). We first compute a minimum vertex cover $X \subseteq V(G)$ of G in time FPT(τ) [4]. We partition the vertices of G into two components, the vertex cover X and the remaining vertices I. By definition of vertex cover, no edge can have both endpoints outside X, therefore I is an independent set. Note that if $s \leq \tau$, then by Observation 4 we have $d + s \leq 2\tau$, where d is the degeneracy of G. In this case we are able to use the algorithm of the last section, that runs in time FPT($d+s$). We may therefore assume $\tau < s$. In that case, we have the following lemma:

Lemma 8 (*). *If $\tau < s$, then (G,k,s,D) is a* yes*-instance.*

It remains to discuss the complexity of this algorithm. As we already said, we first compute a minimum vertex cover X of G in time FPT(τ). If $s \leq \tau$, we run the FPT algorithm of Sect. 4.1. Otherwise, we first compute the set T and then run the subroutine which are both described in the proof of Lemma 8. The two rules used in this subroutine only apply to vertices that belong to the set I and whenever one is applied, exactly one vertex in I is removed (and none is added). Hence, they are applied at most $|I \cap D|$ times. Therefore, the subroutine runs in polynomial time and produces the desired dominating set D_{t}. As a result, this algorithm is FPT with respect to τ. This concludes the proof.

Concluding Remarks. In this paper, we showed that OPT-DSR is PSPACE-complete even if restricted to some graph classes. However, we only know that it is NP-hard for bounded maximum degree graphs or planar graphs, as an

immediate corollary of Observation 3. Hence, it would be interesting to determine whether OPT-DSR is NP-complete or PSPACE-complete on these two graph classes. Note that the complexity on planar graphs remains open for OPT-ISR.

We also proved that OPT-DSR is W[2]-hard for parameter k but the question remains as to whether there exists an XP algorithm for upper bound k.

References

1. Alon, N., Gutner, S.: Linear time algorithms for finding a dominating set of fixed size in degenerated graphs. Algorithmica **54**(4), 544 (2008). https://doi.org/10.1007/s00453-008-9204-0
2. Bertossi, A.A.: Dominating sets for split and bipartite graphs. Inf. Process. Lett. **19**(1), 37–40 (1984)
3. Bonamy, M., Bousquet, N.: Token sliding on chordal graphs. In: Bodlaender, H.L., Woeginger, G.J. (eds.) WG 2017. LNCS, vol. 10520, pp. 127–139. Springer, Cham (2017). https://doi.org/10.1007/978-3-319-68705-6_10
4. Chen, J., Kanj, I.A., Xia, G.: Improved upper bounds for vertex cover. Theor. Comput. Sci. **411**(40), 3736–3756 (2010)
5. Downey, R.G., Fellows, M.R.: Parameterized Complexity. Springer, New York (1999). https://doi.org/10.1007/978-1-4612-0515-9
6. Drange, P., et al.: Kernelization and sparseness: the case of dominating set. In: 33rd Symposium on Theoretical Aspects of Computer Science (STACS 2016), pp. 31:1–31:14 (2016)
7. Garey, M.R., Johnson, D.S.: Computers and Intractability: A Guide to the Theory of NP-Completeness. Freeman, San Francisco (1979)
8. Haddadan, A., et al.: The complexity of dominating set reconfiguration. Theor. Comput. Sci. **651**, 37–49 (2016)
9. Hearn, R.A., Demaine, E.D.: PSPACE-completeness of sliding-block puzzles and other problems through the nondeterministic constraint logic model of computation. Theor. Comput. Sci. **343**(1–2), 72–96 (2005)
10. van den Heuvel, J.: The complexity of change. In: Surveys in Combinatorics 2013. London Mathematical Society Lecture Note Series, vol. 409, pp. 127–160. Cambridge University Press (2013)
11. Ito, T., et al.: On the complexity of reconfiguration problems. Theor. Comput. Sci. **412**(12–14), 1054–1065 (2011)
12. Ito, T., Mizuta, H., Nishimura, N., Suzuki, A.: Incremental optimization of independent sets under reachability constraints. In: Proceedings of the 25th International Computing and Combinatorics Conference (COCOON 2019), pp. 313–324 (2019)
13. Kamiński, M., Medvedev, P., Milanič, M.: Complexity of independent set reconfigurability problems. Theor. Comput. Sci. **439**, 9–15 (2012)
14. Lokshtanov, D., Mouawad, A.E.: The complexity of independent set reconfiguration on bipartite graphs. In: Proceedings of the 29th Annual ACM-SIAM Symposium on Discrete Algorithms (SODA 2018), pp. 7:1–7:19 (2019)
15. Lokshtanov, D., Mouawad, A.E., Panolan, F., Ramanujan, M.S., Saurabh, S.: Reconfiguration on sparse graphs. J. Comput. Syst. Sci. **95**, 122–131 (2018)
16. Mouawad, A.E., Nishimura, N., Raman, V.: Vertex cover reconfiguration and beyond. In: Ahn, H.-K., Shin, C.-S. (eds.) ISAAC 2014. LNCS, vol. 8889, pp. 452–463. Springer, Cham (2014). https://doi.org/10.1007/978-3-319-13075-0_36

17. Mouawad, A.E., Nishimura, N., Raman, V., Simjour, N., Suzuki, A.: On the parameterized complexity of reconfiguration problems. Algorithmica **78**(1), 274–297 (2016). https://doi.org/10.1007/s00453-016-0159-2
18. Nishimura, N.: Introduction to reconfiguration. Algorithms **11**(4), 52 (2018)
19. Suzuki, A., Mouawad, A.E., Nishimura, N.: Reconfiguration of dominating sets. J. Comb. Optim. **32**(4), 1182–1195 (2015). https://doi.org/10.1007/s10878-015-9947-x