Dominating sets reconfiguration under token sliding

Marthe Bonamy\textsuperscript{1}, Paul Dorbec\textsuperscript{2}, and Paul Ouvrard\textsuperscript{1}

\textsuperscript{1}Univ. Bordeaux, Bordeaux INP, CNRS, LaBRI, UMR5800, F-33400 Talence, France \textsuperscript{*}
\textsuperscript{2}Normandie Univ, UNICAEN, ENSICAEN, CNRS, GREYC, 14000 Caen, France \textsuperscript{†}

\textsuperscript{*}{marthe.bonamy, paul.ouvrard}@u-bordeaux.fr
\textsuperscript{†}paul.dorbec@unicaen.fr

Abstract

Let $G$ be a graph and $D_s$ and $D_t$ be two dominating sets of $G$ of size $k$. Does there exist a sequence $(D_0 = D_s, D_1, \ldots, D_{\ell-1}, D_\ell = D_t)$ of dominating sets of $G$ such that $D_{i+1}$ can be obtained from $D_i$ by replacing one vertex with one of its neighbors? In this paper, we investigate the complexity of this decision problem. We first prove that this problem is PSPACE-complete, even when restricted to split, bipartite or bounded treewidth graphs. On the other hand, we prove that it can be solved in polynomial time on dually chordal graphs (a superclass of both trees and interval graphs) or cographs.

1 Introduction

General introduction. Reconfiguration problems arise when, given an instance of a problem, we want to find a step-by-step transformation (called a reconfiguration sequence) between two feasible solutions such that all intermediate solutions are also feasible. Unfortunately, such a transformation does not always exist and some solutions may even be frozen, i.e., they cannot be modified at all. In this context, two natural questions arise: (i) When can we ensure that there exists such a transformation? (ii) What is the complexity of deciding whether a reconfiguration sequence exists?

Interest in combinatorial reconfiguration steadily increased during the last decade. Reconfiguration of several problems, including COLORING \cite{4,10,12}, INDEPENDENT SET \cite{6,7,23}, DOMINATING SET \cite{17,24,27,32} and SATISFIABILITY \cite{15,26} have been studied. For an overview of recent results on reconfiguration problems, the reader is referred to the surveys of van den Heuvel \cite{33} and Nishimura \cite{30}. In this article, we focus on the reconfiguration of dominating sets.

A dominating set is a set of vertices such that every vertex not in the set has a neighbor in it. One can represent a dominating set as a set of tokens, where exactly one token is placed on each vertex that is part of the dominating set. Then, modifying a dominating set corresponds to shifting the tokens according to some rule, called a reconfiguration rule. In the literature, three kinds of operations have been mainly studied:

1. Token Addition and Removal (TAR): one can add or remove a token as long as the total number of tokens does not go beyond a given threshold;
2. Token Jumping (TJ): one can move a token to any vertex of the graph;
3. Token Sliding (TS): one can slide a token along an edge, i.e., one moves a token to a neighbor of its current location.

One can observe that in the last two models, the size of each solution remains constant at any time, as opposed to what happens in the TAR model. In this article, we are mostly interested in the Token Sliding model.

We define the reconfiguration graph for domination, denoted $R_k(G)$ as follows: the vertices of $R_k(G)$ are the dominating sets of size $k$ and there is an edge between two vertices if and only if one can...
go from the first to the second thanks to the reconfiguration rule that we consider (token sliding in our case). Three natural problems can be identified:

1. The **reachability** problem: given a graph $G$ and two dominating sets $D_s$ and $D_t$, is there a path between $D_s$ and $D_t$ in $R_k(G)$? In other words, does there exist a reconfiguration sequence between $D_s$ and $D_t$?

2. The **connectivity** problem: given a graph $G$, is the reconfiguration graph $R_k(G)$ connected?

3. The **shortest path** problem: given a graph $G$, two dominating sets $D_s$ and $D_t$ and an integer $\ell$, is the distance in $R_k(G)$ between $D_s$ and $D_t$ at most $\ell$? In other words, does there exist a reconfiguration sequence between $D_s$ and $D_t$ of length at most $\ell$?

In this article, we focus on the reachability version and we will denote this problem by DSR$_{TS}$ for short. We adopt the same notation for VCR$_{TS}$ and ISR$_{TS}$, respectively.

**Related results.** The reconfiguration of dominating sets has been mainly studied under the **Token Addition and Removal** model. Haas and Seyffarth gave sufficient conditions to guarantee the connectivity of the reconfiguration graph according to $k$, the cardinality threshold of dominating sets [16]. More precisely, they proved that $R_{n-1}(G)$ (where $n$ is the number of vertices of $G$) is connected if $G$ has at least two independent edges. This value can be lowered to $\Gamma+1$ (where $\Gamma$ is the maximum size of a minimal dominating set) if the input graph $G$ is chordal or bipartite. Suzuki et al. [32] showed that this result cannot be generalized to any graph since they constructed an infinite family of graphs for which $R_{\Gamma+1}(G)$ is not connected. On the positive side, they proved that $R_{n-\mu}(G)$ is connected if $G$ has a matching of size $\mu+1$.

Haddadan et al. [17] studied the complexity of the reconfiguration of dominating sets under the token addition and removal rule from a graph classes perspective. They proved that the reachability problem is PSPACE-complete, even if the input graph is a split graph, a bipartite graph, has bounded bandwidth or is planar with maximum degree six. On the other hand, they gave linear-time algorithms for trees, interval graphs or cographs.

Mouawad et al. [27] studied the parameterized complexity of DOMINATING SET RECONFIGURATION under token addition and removal. They proved that this problem is W[2]-hard when parameterized by $k+\ell$, where $k$ is the threshold and $\ell$ the length of the reconfiguration sequence. As a positive result, Lokshtanov et al. [24] gave a fixed-parameter algorithm with respect to $k$ for graphs excluding $K_{d,d}$ as a subgraph, for any constant $d$.

The third author also considered this problem (still in the TAR model) through the lens of an optimization variant (see Blanché et al. [3]) as recently introduced by Ito et al. for independent sets [20].

To the best of our knowledge, the reconfiguration of dominating sets under TS has only been studied from a structural perspective. Fricke et al. [13] introduced the concept of $\gamma$-graph which corresponds to the reconfiguration graph $R_{\gamma}(G)$ under the token sliding rule. In particular, they proved that $R_{\gamma}(G)$ is connected and bipartite if $G$ is a tree. For a more complete overview, the reader is referred to [28].

**Our contribution.** In this article, we are interested in the reachability question of dominating sets reconfiguration under token sliding. This reconfiguration rule has already been studied for various reconfiguration problems but not for dominating sets, to the best of the authors’ knowledge.

We tackle this problem with a complexity perspective according to several graph classes: in Section 3 we prove for instance that DSR$_{TS}$ is PSPACE-complete for split graphs or bipartite graphs. Note that the reductions used in the proofs of Theorems 5 and 6 are identical to the ones of [17], which are quite standard (see, e.g., [2]). Our reduction to prove the PSPACE-completeness of DSR$_{TS}$ for split graphs is similar but not identical to the one of [17], as we reduced from DSR$_{V}$ and not from VERTEX COVER CONFIGURATION.

In Section 4, we show that this problem can be solved in polynomial time on other graph classes such as cographs or dually chordal graphs (the formal definitions of these two graph classes is given in Subsections 4.1 and 4.2 respectively). Note that our result for cographs is a consequence of Theorem 7.
which is more general (see the discussion in Subsection 4.1). Note also that our result on dually chordal graphs generalizes the ones of [17] for trees and interval graphs since the class of dually chordal graphs is a superclass of both interval graphs and trees as discussed in Subsection 4.2.

Figure 1 gives an overview of our results where \( A \to B \) means that the class \( B \) is properly included in the class \( A \).

# 2 Preliminaries

This section is devoted to some basic definitions of graph theory used in this article, followed by a more formal introduction of the problem we are interested in.

Each graph \( G = (V, E) \) considered is simple (i.e., \( G \) is undirected and has no multiple edges or loops) where \( V \) represents the vertex set of \( G \) and \( E \) its edge set. We denote by \( n = |V| \) and \( m = |E| \) the number of vertices and edges of \( G \). The eccentricity of a vertex \( u \) denoted by \( \epsilon(u) \) is the maximum distance between \( u \) and any other vertex. For a subset of vertices \( S \subseteq V \), we denote by \( G[S] \) the subgraph induced by \( S \). For a vertex \( u \in V \), we denote by \( N_G(u) \) its open neighborhood, i.e., the set \( \{v \mid vw \in E\} \) and by \( N_G[u] \) its closed neighborhood, i.e. the set \( N_G(u) \cup \{u\} \). For a subset of vertices \( S \subseteq V \), we define the closed neighborhood of \( S \) as the union of the closed neighborhood of the vertices in \( S \), i.e., \( N_G[S] = \bigcup_{u \in S} N_G[u] \).

Let \( G_1 \) and \( G_2 \) be two graphs. We recall two basic binary operations on graphs: the disjoint union and the join operations. The disjoint union \( G_1 \cup G_2 \) of two graphs on disjoint vertex sets is the graph with vertex set \( V(G_1 \cup G_2) = V(G_1) \cup V(G_2) \) and edge set \( E(G_1 \cup G_2) = E(G_1) \cup E(G_2) \). The join operation can be obtained from the disjoint union by adding all possible edges between \( G_1 \) and \( G_2 \). More formally, the join of \( G_1 \) and \( G_2 \) denoted by \( G_1 + G_2 \) is the following graph:

- \( V(G_1 + G_2) = V(G_1) \cup V(G_2) \);
- \( E(G_1 + G_2) = E(G_1) \cup E(G_2) \cup \{uv \mid u \in V(G_1), v \in V(G_2)\} \).

A dominating set for a graph \( G = (V, E) \) is a subset of vertices \( D \subseteq V \) such that \( N[D] = V \), i.e., each vertex either belongs to \( D \) or has a neighbor in \( D \). For a graph \( G \), we denote by \( \gamma(G) \) the domination number of \( G \) defined as the minimum size of a dominating set. Let \( G \) be a graph and \( D \) a dominating set of \( G \). We say that \( u \) is a private neighbor of \( v \) (with respect to \( D \)) if \( u \notin D \) and \( v \) is the only neighbor of \( u \) in \( D \). Therefore, a dominating set is inclusion-wise minimal if and only if each of its vertices has a private neighbor.

**Our problem.** In the token sliding model, a natural question is whether we should authorize more than one token to be placed on a vertex during the reconfiguration sequence. Here is an example where
it makes a difference: consider the star graph $S_n$ on $n+1$ vertices and two dominating sets $D_1$ and $D_2$ of $S_n$ of size $k$, with $k \in [2, n-1]$. Any dominating set of that size necessarily contains the central vertex. To reconfigure $D_1$ into $D_2$, we are forced to move a token from one leaf to another, which can only be done by going through the central vertex which already contains a token. Given such artificially negative examples, we choose to allow the superposition of tokens on a vertex. Note that this question did not arise in previous papers considering the token sliding model, to the best of our knowledge.

Figure 2: Example of TS-sequence from $D_s$ to $D_t$.

We are now ready to define properly the DOMINATING SET RECONFIGURATION problem under token sliding.

**Instance:** A graph $G = (V, E)$ and two dominating sets $D_s$ and $D_t$ of cardinality $k$ of $G$.

**Question:** Is there a TS-sequence between $D_s$ and $D_t$, i.e., does $D_s \leftrightarrow^{TS} D_t$ hold?
Theorem 1. For every $\ell \geq 3$, there exists a graph $G_\ell$ where, for every $k < \ell$, every dominating set of size $k$ can be reconfigured into any other, while there are two dominating sets of size $\ell$ that cannot be reconfigured one into the other.

Proof. We first prove the statement for $k = 2$. For every integer $\ell > 2$, we define the graph $G_\ell$ such that $G_\ell$ contains exactly one dominating set of size $\gamma(G) = 2$ but for which the dominating sets of size $\ell$ are not reconfigurable. To construct $G_\ell$, we first create $\ell$ pairs of triangles $(G_1^1, G_1^2), \ldots, (G_\ell^1, G_\ell^2)$ such that $G_1^1$ and $G_1^2$ share exactly one vertex $w_i$. Moreover, let all the $G_i^1$'s share a vertex $u$ and all the $G_i^2$'s share a vertex $v$ (see Figure 3 for $G_3$ as an example). Note that we have $\gamma(G_\ell) = 2$ since $\{u, v\}$ is a dominating set and $G_\ell$ does not contain a universal vertex (i.e., a vertex adjacent to all the other vertices).

Consider the dominating set $D_s = \{w_1, \ldots, w_\ell\}$. It is a dominating set of $G_\ell$ of size $\ell$. By token sliding, $D_s$ cannot be reconfigured into any other dominating set of size $\ell$. Indeed, in $D_s$ we cannot move any $w_i$ in a triangle because it would leave the other triangle of the pair $(G_i^1, G_i^2)$ not dominated. Note that any set of $\ell$ vertices containing $u$ and $v$ is a dominating set of $G_\ell$, hence the existence of two dominating sets of size $\ell$ as desired.

Consider now $k < \ell$. Any dominating set of $G_\ell$ on fewer than $\ell$ vertices contains both $u$ and $v$. Indeed, if for instance $u$ is not in the dominating set, then $\ell$ extra vertices are necessary to dominate the triangles $G_i^1$. Therefore, any dominating set $D$ of $G_\ell$ on $k$ vertices contains both $u$ and $v$. The other vertices are therefore not necessary for domination purposes, and we can slide them around as desired, superposing them with $u$ and $v$ arbitrarily. There are many dominating sets on $k$ vertices, but they all contain $u$ and $v$ and can be trivially reconfigured one into another.

3 PSPACE-completeness

In this section, we study the complexity of DSR$_{TS}$ in the general case. We show that this problem is PSPACE-complete, even when restricted to split graphs, bipartite graphs or bounded treewidth graphs. Let us first recall the following result from Haddadan et al. [17], stating the complexity of the reconfiguration problem for the TAR model.

Theorem 2 ([17]). Let $G$ be a graph and $D_s, D_t$ be two dominating sets of $G$ of size $k$. Deciding whether $D_s \xrightarrow{TAR} D_t$ is PSPACE-complete.

Note that the problem remains PSPACE-complete, even if the input graph is a planar graph with maximum degree 6, has bounded bandwidth, is bipartite or is a split graph as pointed out previously. Let us now show that the TJ and TAR rules are equivalent under some constraints. Note that a similar proof can be found in [16].

Lemma 3. Let $G$ be a graph and $D_s$ and $D_t$ be two dominating sets of $G$ of size $k$. We have $D_s \xrightarrow{TAR} D_t$ if and only if $D_s \xrightarrow{TJ} D_t$.

Proof. The proof is an adaptation of the Theorem 1 of Kamiński et al. [21].
(⇐) Suppose first $D_s \xrightarrow{TJ} D_t$, and let $S$ be a TJ-sequence that reconfigures $D_s$ into $D_t$. This sequence corresponds to a sequence of moves $u \xrightarrow{TJ} v$. We construct a TAR-sequence by replacing each atomic move $u \xrightarrow{TJ} v$ with two moves of the TAR model: we first add $v$ and then delete $u$. By first adding $v$, we preserve the domination property. Besides, since we immediately delete $u$ after the addition of $v$, each intermediate solution is of size at most $k + 1$, as desired.

(⇒) For the other direction, let $S'$ be a TAR-sequence that reconfigures $D_s$ into $D_t$. Note that since $|D_s| = |D_t| = k$, $S'$ is of even length. Moreover, by hypothesis, $S'$ does not contain a configuration of size more than $k + 1$. If all the configurations of $S'$ are of size $k$ or $k + 1$, this means that $S'$ corresponds to an alternation of an addition of a token on some vertex $v$ immediately followed by the deletion of a token on a vertex $u$. Therefore, to get a TJ-sequence, we simply replace each of these subsequences by a move $u \xrightarrow{TJ} v$. Suppose now that $S'$ contains some configuration of size less than $k$ and consider a configuration, let us say $D_i$, of smallest size. Since $D_i$ is a configuration of smallest size, this means that it has been obtained from $D_{i-1}$ by the deletion of some vertex $x$. We also know that the configuration $D_{i+1}$ is obtained from $D_i$ by the addition of some vertex $y$. If $x = y$, then these two steps are redundant and can simply be ignored. Otherwise, observe that if we first add $y$ and then delete $x$, the new sequence is still valid. If all the configurations are of size $k$ or $k + 1$, we immediately obtain a TJ-sequence. Otherwise, we can repeat this process until this is the case.

As a corollary of Theorem 2 and Lemma 3, we obtain that deciding whether two dominating sets of size $k$ of a graph $G$ can be reconfigured under the token jumping model is a PSPACE-complete problem. We are now ready to prove Theorem 4.

**Theorem 4.** DSRTS is PSPACE-complete on split graphs.

**Proof.** First, note that the problem is in PSPACE [19]. We give a polynomial-time reduction from DSRTS, which is PSPACE-complete as discussed above. Let $G = (V, E)$ be a graph with $V(G) = \{v_1, \ldots, v_n\}$. We construct the corresponding split graph $G'$ as follows:

- $V(G') = V_1 \cup V_2$ where $V_1 = \{v_1, \ldots, v_n\}$ and $V_2 = \{w_1, \ldots, w_n\}$;

- $E(G') = \{uv \mid u, v \in V_1\} \cup \{v_iw_j \mid v_i \in N_G[v_j]\}$, i.e., we add all possible edges in $V_1$ so that $V_1$ forms a clique. We also add an edge between a vertex $v_i \in V_1$ and a vertex $w_j \in V_2$ if and only if the corresponding vertex $v_i$ in the original graph $G$ belongs to the closed neighborhood of $v_i$ in $G$.

Observe that $G'$ is a split graph since $V_1$ forms a clique and $V_2$ an independent set (see Figure 4 for an example). To a set of vertices of $G$, we associate the corresponding vertices of $V_1$ in $G'$. By definition of $G'$, any dominating set $D$ of $G$ is also a dominating set for $G'$; indeed, a vertex $v_i \in V_1$ dominates all the vertices in $V_1$ (since it is a clique) and all the vertices in $V_2$ that correspond to vertices in its closed neighborhood in $G$. That $D$ dominates $G$ allows us to conclude that the corresponding set also dominates $V_2$. Hence, $D$ is also a dominating set of $G'$.

![Figure 4: Example for the reduction of Theorem 4](image-url)
Let \((G, D_s, D_t)\) be an instance of DSR\(_{TS}\); we reduce this instance to the instance of DSR\(_{TS}\) \((G', D_s, D_t)\). This reduction can be done in quadratic time. Now, we need to prove that \(D_s \xrightarrow{TS} D_t\) if and only if there is a reconfiguration sequence between \(D_s\) and \(D_t\) in \(G'\) using the token jumping model.

\((\Rightarrow)\) Consider a TJ-sequence in \(G\), and translate it to \(G'\). All intermediate sets still are dominating sets, and since all pairs of vertices are joined by an edge in \(V_1\), this sequence is a valid TS-sequence in \(G'\).

\((\Leftarrow)\) We now prove the other direction. Let \((D_0 = D_s, \ldots, D_k = D_t)\) be a TS-sequence in \(G'\). First, observe that any dominating \(D'\) of \(G'\) such that \(D' \subseteq V_1\) corresponds to a dominating set of \(G\). Indeed, any vertex \(v_j \in V_2\) is dominated by a vertex \(v_i \in V_1\) and by construction of \(G', v_iv_j \in E(G)\). Hence, \(v_j\) is dominated by \(v_i\) and thus \(D'\) is also a dominating set of \(G\). Hence, if the sequence does not use vertices in \(V_2\), we immediately obtain a TJ-sequence in \(G\) from \(D_s\) to \(D_t\), as the token jumping model does not require adjacency. Suppose on the other hand that the sequence goes through some vertices in \(V_2\). Since all vertices are initially in \(V_1\), there is a subsequence that contains a move \(v_i \xrightarrow{TS} v_j\). Since \(w_j \notin V_1\), there exists a later step where the token on \(w_j\) is moved to an adjacent vertex \(v_k\) in \(V_1\) (since \(V_2\) is independent). However, \(w_j\) does not dominate any vertex in \(V_2\) (since \(V_2\) is a stable set) and thus \(N[w_j] \subseteq N[v_k]\). Therefore, we simply replace these two moves by a single move \(v_i \xrightarrow{TS} v_k\). We can thus assume that the reconfiguration sequence only uses vertices in \(V_1\). The conclusion follows.

Next, we prove that DSR\(_{TS}\) is PSPACE-complete on bipartite graphs. We use a reduction from the VERTEX COVER RECONFIGURATION problem under token sliding (or VCR\(_{TS}\) for short). Recall that a vertex cover is a set of vertices such that every edge has an endpoint in the set. The complement of a VERTEX COVER is a reconfiguration sequence between \(V\) and all the vertices in \(V\), so that \((1 - V)\) is a dominating set.

Theorem 5. DSR\(_{TS}\) is PSPACE-complete on bipartite graphs.

**Proof.** We give a polynomial-time reduction from VCR\(_{TS}\). This is an adaptation of the well-known reduction from VERTEX COVER to DOMINATING SET [14]. Let \(G = (V, E)\) be a graph. We construct the corresponding bipartite graph \(G' = (V_1 \uplus V_2, E')\) as follows: for each edge \(uv \in E\), add \(u\) and \(v\) to \(V_1\) and a new vertex \(z_{uv}\) of degree two to \(V_2\) that is adjacent to exactly \(u\) and \(v\). Note that \(E'\) does not contain the edge \(uv\) so that \(V_1\) induces an independent set. Finally, add to \(V_2\) a vertex \(x\) adjacent to all the vertices in \(V_1\) and attach to \(x\) a degree-one vertex \(y\) which is added to \(V_1\) (see Figure 5 for an example). Formally, the graph \(G'\) is the following:

- \(V(G') = V_1 \cup V_2\) where \(V_1 = V(G) \cup \{y\}\) and \(V_2 = \{z_{uv} \mid uv \in E\} \cup \{x\}\);
- \(E' = \{uz_{uv} \text{ and } z_{uv}v \mid u, v \in V_1\) and \(z_{uv} \in V_2\}\} \cup \{xv \mid v \in V_1\}) \cup \{xy\}\).

Observe that \(G'\) is bipartite and the reduction can be done in polynomial time. We now prove that the vertex covers of \(G\) of size \(k\) are reconfigurable if and only if the dominating sets of \(G'\) of size \(k + 1\) are. Let \((G, C_s, C_t)\) be an instance for the VCR\(_{TS}\) problem. We define the corresponding instance for the

![Figure 5: Example for the reduction of Theorem 5](image-url)
DSR_{TS} problem as \((G', C_s \cup \{x\}, C_t \cup \{x\})\). Since \(C_s\) is a vertex cover of \(G\), for every edge \(uv \in E(G)\) we have \(\{u, v\} \cap C_s \neq \emptyset\) and thus the vertices \(u, v, z_{uv}\) are dominated by \(C_s\) in \(G\). Now \(x\) dominates both \(x\) and \(y\), so \(D_0 = C_s \cup \{x\}\) is a dominating set of \(G'\), and by the same argument, so is \(D_1 = C_t \cup \{x\}\). Since VCR_{TS} and DSR_{TS} both employ the same reconfiguration rule, we simply denote by \(u \sim v\) (instead of \(TS\) a move of a reconfiguration sequence between \(C_0\) and \(C_t\) (respectively \(D_0\) and \(D_1\)).

(\Rightarrow) We start with the only if direction. First, it immediately follows from the definition of \(D_0\) and \(D_t\) that \(D_0 \setminus \{x\} = C_s\) and \(D_t \setminus \{x\} = C_t\). Let us assume that \((G, C_s, C_t)\) is a yes-instance for the VCR_{TS} problem. Then, there exists a reconfiguration sequence \(S\) using the token sliding model between \(C_s\) and \(C_t\). One can construct a sequence \(S'\) for \(G'\) by replacing a move \(u \sim v\) (where \(uv \in E(G)\)) of \(S\) into two moves: \(u \sim z_{uv}\) followed by \(z_{uv} \sim v\). We need to prove that the domination property is preserved at every step. First, observe that each intermediate solution contains \(x\), so each move of the form \(z_{uv} \sim v\) is safe because \(u\) is still dominated by \(x\) and \(z_{uv}\) by \(v\). Therefore, the only risk is to leave some vertex \(z_{uv}\) non dominated after a move \(u \sim z_{uv}\). In that case, this implies that \(w\) does not belong to the solution, which in turn means that the edges \(wu\) and \(uv\) are covered only by \(u\). Therefore, the move \(u \sim v\) of the sequence \(S\) is not valid (because the edge \(wu\) is no longer covered), a contradiction. Therefore, \((G', D_s, D_t)\) is a yes-instance for the DSR_{TS} problem.

(\Leftarrow) It remains to prove the if direction. Suppose that \((G', D_s, D_t)\) is a yes-instance for the DSR_{TS} problem. Then, there exists a reconfiguration sequence \(S' = (D_2, \ldots, D_k)\) in \(G'\). First, observe that at each step, \(y\) needs to be dominated and thus either \(x\) or \(y\) belongs to each solution. Moreover, initially, \(D_0\) does not contain \(y\). We can ignore all moves of the form \(x \sim y\) (each such move will be eventually followed by a move \(y \sim x\)), and assume that \(x\) contains at least one token in each solution. Therefore, the only vertices whose domination is not immediate by the existence of a token on \(x\) are the vertices of the form \(z_{uv}\), i.e., the vertices that correspond to the edges of \(G\). Recall that \(D_0 = C_s \cup \{x\}\), so every vertex in \(D_0 \setminus \{x\}\) belongs to \(V(G') \cap V(G)\). We consider in turn the two other possible moves \(u \sim v\), where \(u \in V(G') \cap V(G)\) (i.e., \(u\) corresponds to a vertex of the original graph \(G\)), and \(v\) either belongs to \(V(G') \setminus V(G)\) or \(v = x\). We focus on the next operation (which may not be consecutive) that touches the vertex \(v\). Suppose first that \(v \in V(G') \setminus V(G)\), i.e., \(v\) corresponds to a vertex \(z_{uv'}\) for some vertex \(uv' \in N_G[u]\). If the next move that touches \(z_{uv'}\) is \(z_{uv'} \sim u\), these two operations can be ignored. Otherwise, since \(z_{uv'}\) has degree two, the next operation that touches \(z_{uv'}\) is \(z_{uv'} \sim uv'\). Moreover, we claim that we can assume that \(z_{uv'} \sim v\) is the operation that immediately follows the move \(u \sim z_{uv}\). Indeed, \(N_{G'}[z_{uv}'] \subseteq N_{G'}[u]\) so if a dominating set \(D\) contains \(z_{uv}'\), \(D' = (D \setminus \{z_{uv}'\}) \cup \{uv'\}\) is also a dominating set of \(G'\). So one can assume that in \(S'\), if we have a move \(u \sim z_{uv}\), it will be immediately followed by a move \(z_{uv'} \sim uv'\). In that case, one can replace these two moves by \(u \sim u'\) in a reconfiguration sequence from \(C_0\) to \(C_t\). Let us now consider the other possible move: \(u \sim x\). If the next move that touches \(x\) is \(x \sim u\), we again simply ignore these two steps. Let \(D_t\) be the dominating set of \(S'\) to which the move \(u \sim x\) is applied. Recall that when a token is moved from a vertex \(a\) to a vertex \(z_{ab}\) (for some neighbor \(b\) of \(a\)), it is followed by \(z_{ab} \sim b\). Therefore, we know that \(D_t\) does not contain any vertex of the form \(z_{uv}\). So for every edge \(uv\) incident to \(u\), \(D_t + 1\) must contain \(u\) (this is possible if \(u\) has at least two tokens in \(D_t\)) or \(u'\) in order to dominate \(z_{uv}\). Hence, \(C_{t+1} = D_t + 1 \setminus \{x\}\) is a vertex cover of \(G\). If the next move that touches \(x\) is \(x \sim u'\), one can safely replace these two moves \(u \sim x\) and \(x \sim u'\) by \(d\) moves where \(d\) is the distance between \(u\) and \(u'\) in \(G\).

Therefore, one can obtain from \(S'\) a TS-sequence that reconfigures \(C_s\) into \(C_t\) and thus \((G, C_s, C_t)\) is a yes-instance for VCR_{TS}, as desired. This concludes the proof of Theorem 5.

Next, we prove that DSR_{TS} is PSPACE-complete on planar graphs of maximum degree 6 and bounded bandwidth graphs. Recall that a graph has bandwidth at most \(k\) if there exists a numbering \(\ell\) of the vertices with distinct integers between 1 and \(n\) (where \(n\) is the number of vertices of the graph) such that adjacent vertices must have labels at distance at most \(k\) (i.e., for every edge \(uv \in E_G\), \(|\ell(u) - \ell(v)| \leq k\)).

**Theorem 6.** DSR_{TS} is PSPACE-complete on planar graphs of maximum degree 6 and bounded bandwidth graphs.

**Proof.** First, recall that VCR_{TS} is PSPACE-complete on planar graphs of maximum degree 3 [18] and on bounded bandwidth graphs [34]. The proof for dominating sets reconfiguration under TAR on planar graphs from [17] works also here since VCR_{TS} is PSPACE-complete on planar graphs. We use the well-known reduction mentioned in Theorem 5, which is the following: start with a copy of the original
4 Polynomial-time algorithms

In this section, we prove the following theorem, that is of special interest for the case of cographs. Recall that the domination number of a join \( G_1 + G_2 \) is always at most two, since taking a vertex from each operand of the join dominates the whole graph.

**Theorem 7.** Let \( G_1 \) and \( G_2 \) be two graphs, and \( D_4 \) and \( D_t \) be two dominating sets of \( G_1 + G_2 \) of the same size. The dominating set \( D_4 \) can be reconfigured into \( D_t \) by token sliding if and only if one of the three following conditions holds:

(i) \(|D_4| = |D_t| \geq 3,

(ii) the domination number of \( G_1 \) or of \( G_2 \) is at most two,

(iii) both \( G_1 \) and \( G_2 \) are connected.

**Proof.** We first show that if none of these conditions hold, then \((G_1 + G_2, D_4, D_t)\) is a no-instance. Let \( G_1 \) and \( G_2 \) be two graphs with \( \gamma(G_1) > 2 \) and \( \gamma(G_2) > 2 \), and assume without loss of generality that \( G_1 \) is not connected, say with two components \( C_1 \) and \( C_2 \). Note that \( \gamma(G_1 + G_2) = 2 \) since neither \( G_1 \) nor \( G_2 \) has a universal vertex.

Let \( D_4 = \{u, v\} \) and \( D_t = \{w, v\} \) be two minimum dominating sets of \( G \) with \( u \in C_1 \), \( w \in C_2 \) and \( v \in V(G_2) \). We prove that \( D_4 \) can not be reconfigured into \( D_t \). Since \( G_1 \) is not connected, there is no
path between \( u \) and \( w \) in \( G[V_1] \). Therefore, the only way to reach \( w \) from \( u \) is to go through \( V(G_2) \). But since \( \gamma(G_2) > 2 \) no pair of vertices in \( G_2 \) can dominate \( G_2 \), and thus no move from \( V(G_1) \) to \( V(G_2) \) is possible.

We now prove that each of the above conditions is sufficient for the dominating sets to be reconfigured.

**Condition (i)** Suppose \( |D_k| = |D_t| \geq 3 \). Recall that picking a vertex of \( G_1 \) and one of \( G_2 \) always forms a dominating set of \( G_1 + G_2 \). We infer that it is always possible to make one move from \( D_k \) to reach a configuration with tokens in both \( G_1 \) and \( G_2 \), then from such position tokens can be slid freely in their part, until reaching \( D_t \) with a last move. 

We assume now that \( |D_k| = |D_t| \leq 2 \).

**Condition (ii)** For the case when \( G_1 \) or \( G_2 \) has domination number at most two, we consider different cases depending on whether a graph has domination number one or not.

**Case 1.** If \( \gamma(G_1) = 1 \) or \( \gamma(G_2) = 1 \): then \( G_1 + G_2 \) contains a universal vertex. Then, from \( D_k \) one can place a token on this vertex, reconfigure other possible tokens freely, then move that token to reach \( D_t \).

**Case 2.** If \( \gamma(G_1) = 2 \) and \( \gamma(G_2) = 2 \). Assume without loss of generality that \( \gamma(G_1) = 2 \). Note that in this case, \( \gamma(G_1 + G_2) = 2 \), let \( D_k = \{v_1, v_2\} \). We define an arbitrary canonical dominating set \( C \) by taking a vertex (e.g., of smallest index) in each of \( G_1 \) and \( G_2 \); we denote these vertices \( u_1 \in V(G_1) \) and \( u_2 \in V(G_2) \). Recall that each reconfiguration sequence is reversible. Hence, it is sufficient to prove that both \( D_k \) and \( D_t \) can be transformed into \( C \). We only show this statement for \( D_k \); the proof for \( D_t \) follows by symmetry.

Suppose first that \( v_1 \) and \( v_2 \) belong to the same original graph, say \( v_1, v_2 \in V(G_1) \). We show how to reconfigure \( D_k \) into \( C \) in at most two steps. First, observe that since \( C \) is a dominating set of \( G \), \( u_1 \in N[\{v_1, v_2\}] \), say \( u_1 \) belongs to \( N[v_1] \). Our first step is to slide the token from \( v_2 \) to \( u_2 \) along the corresponding edge of the join. Then, by our observation that \( u_1 \in N[v_1] \), we can slide if necessary the token from \( v_1 \) to \( u_1 \).

Suppose now that \( v_1 \) and \( v_2 \) belong respectively to \( V(G_1) \) and \( V(G_2) \). Since \( \gamma(G_1) = 2 \), let \( \{w_1, w_2\} \) be a dominating set of \( G_1 \) and thus of \( G_1 + G_2 \) (it can be computed naively in cubic time). It dominates \( v_1 \) so assume without loss of generality that \( v_1w_1 \) is an edge. First moving the token from \( v_1 \) to \( w_1 \) (if \( v_1 \neq w_1 \)) and then from \( v_2 \) to \( w_2 \), at most two steps permit us to reconfigure \( D_k \) into \( \{w_1, w_2\} \), which we can then reconfigure into \( C \) by the above argument.

**Condition (iii)** Suppose finally that \( \gamma(G_1) \geq 3 \) and \( \gamma(G_2) \geq 3 \) but both \( G_1 \) and \( G_2 \) are connected. Then \( \gamma(G_1 + G_2) = 2 \) and the minimum dominating sets of \( G_1 + G_2 \) are exactly the sets containing a vertex in \( G_1 \) and a vertex in \( G_2 \). Let \( D_k = \{v_1, v_2\} \) and \( D_t = \{w_1, w_2\} \) with \( w_1, w_2 \in V(G_1) \) and \( v_2, w_2 \in V(G_2) \). Since \( G_1 \) is connected, there exists a path from \( v_1 \) to \( w_1 \) in \( G[V(G_1)] \). Moving the token along this path, we always keep a dominating set by the above observation. Doing similarly along a path from \( v_2 \) to \( w_2 \), we have a reconfiguration sequence from \( D_k \) to \( D_t \).

We now consider the special case of cographs. Recall that the family of cographs can be defined as the family of graphs with no induced \( P_4 \), or equivalently by the following recursive definition:

- \( K_1 \) is a cograph;
- for \( G_1 \) and \( G_2 \) any two cographs, the disjoint union \( G_1 \cup G_2 \) is a cograph;
- for \( G_1 \) and \( G_2 \) any two cographs, the join \( G_1 \cup G_2 \) is a cograph.

Brandelt and Mulder gave in [1] an alternative characterization of cographs: \( G \) is a cograph if and only if \( G \) is the disjoint union of distance-hereditary graphs with diameter at most two. Note that computing a minimum dominating set in distance-hereditary graphs is linear-time solvable [29]. Hence, we can compute the domination number of a cograph in linear time as well.

By the previous theorem, we infer that if a cograph is constructed as a join of two cographs, the case is polynomial-time decidable. The case when \( G = K_1 \) is straightforward. If \( G = G_1 \cup G_2 \) is the
disjoint union of two cographs, then for two dominating sets $D_s$ and $D_t$, deciding whether $D_s \equiv_{TS} D_t$ is equivalent to deciding whether $D_s \cap V(G_1) \equiv_{TS} D_t \cap V(G_1)$ in $G_1$, and $D_s \cap V(G_2) \equiv_{TS} D_t \cap V(G_2)$ in $G_2$, which can be done inductively by induction. As a consequence, we obtain the following:

**Theorem 8.** There is a polynomial-time algorithm deciding $DSR_{TS}$ in cographs.

### 4.2 Dually chordal graphs

Let $G = (V, E)$ be a graph with $V = \{v_1, v_2, \ldots, v_n\}$. We denote by $G_i$ the graph $G[\{v_i, v_{i+1}, \ldots, v_n\}]$. A **maximum neighbor of a vertex** $u$ is a vertex $v \in N[u]$ such that we have $N[w] \subseteq N[v]$ for every vertex $w \in N[u]$. In other words, $v$ contains in its closed neighborhood every vertex at distance at most two from $u$. A **maximum neighbor ordering** (or $mno$ for short) is an ordering of the vertices in such a way that $v_i$ has a maximum neighbor in the graph $G_i$. A graph is **dually chordal** if it has a maximum neighbor ordering. This ordering can be computed in linear time \cite{K54}. Moreover, the $mno$ computed by this algorithm is such that for every vertex $v_i$ (with $i < n$), $v_i$’s maximum neighbor is different from $v_i$ (for connected graphs). An alternative proof of a similar statement for not necessarily connected graphs can be found in \cite{V08}. In the following, we always assume that an $mno$ is associated with a function $mn : V \rightarrow V$ that associates with each vertex a maximum neighbor.

Note that a dually chordal graph is not necessarily chordal. Figure \ref{fig:dually-chordal} gives an example of a graph which is dually chordal but not chordal, since it contains an induced cycle on four vertices. The label inside each vertex corresponds to its rank in the ordering, and its maximum neighbor is the endpoint of its single outgoing edge (note that $v_5$’s maximum neighbor is itself). Moreover, observe that any tree $T$ is a dually chordal graph: root the tree in some vertex and orient all edges toward the root; any numbering keeping all $G_i$ connected is an $mno$ where arcs point towards the vertex maximum neighbor.

**Link with interval graphs.** An interval graph is the intersection graph of a family of intervals on the real line. In other words, let $\{I_1, I_2, \ldots, I_n\}$ be a set of intervals. Each interval $I$ can be represented by its extremities $\ell(I), r(I)$ with $\ell(I) \leq r(I) \in \mathbb{R}$. We call these values respectively the $\ell$-value and $r$-value (for left and right). The corresponding interval graph $G = (V, E)$ is the following:

- $V = \{I_1, I_2, \ldots, I_n\};$
- $I_i I_j \in E \iff I_i \cap I_j \neq \emptyset$, i.e., $\ell(I_i) \leq r(I_i)$ and $\ell(I_j) \leq r(I_j)$.

Let $G = (V, E)$ be an interval graph. For convenience, we denote by $v_i$ the vertex related to the interval $I_i$. We now order the vertices of $G$ with respect to their $r$-values, i.e., $v_i < v_j$ if and only if $r(I_i) < r(I_j)$ (or $r(I_i) = r(I_j)$ and $\ell(I_i) < \ell(I_j)$). Then, we get the following useful property:

**Observation 9.** Let $v_i$ and $v_j$ be two vertices of $G$ such that $v_i < v_j$. If $v_i v_j \in E$, then for any $v_k$ such that $v_i < v_k < v_j$, we have $v_i v_k v_j \in E$.

**Proof.** Since $v_i < v_k < v_j$, we have $r(I_i) \leq r(I_k) \leq r(I_j)$. Since $v_i v_j$ is an edge, $\ell(I_j) \leq r(I_j)$. Thus, we get that $\ell(I_i) \leq r(I_k)$. Adding that $\ell(I_k) \leq r(I_k) \leq r(I_j)$, the conclusion follows.

**Observation 10.** Interval graphs are dually chordal graphs.
Moreover, any other neighbor \( v \) of \( v_i \) in \( G_i \) is adjacent to \( v_j \). Moreover, any other neighbor \( v_t > v_i \) of \( v_k \) either satisfies \( v_i < v_t < v_j \) or \( v_k < v_j < v_t \). In both cases, Observation \( 9 \) concludes the proof.

An example of the construction used above on an interval graph is given in Figure 7 where the maximum neighbor of \( I_1 \) is its only out-neighbor (the endpoint of the only directed edge incident to \( I_1 \)).

![Figure 7: Interval graph and its maximum neighborhood ordering.](image)

**Proof.** To see this observation, we prove that the ordering described above is an mno. For every vertex \( v_i \), we show that its neighbor of maximum index \( v_j \) in the ordering is a maximum neighbor. Indeed, consider any neighbor \( v_k \) of \( v_i \) in \( G_i \). By definition of \( v_j \), we have \( v_i < v_k \leq v_j \), and \( v_k \) is adjacent to \( v_j \). Moreover, any other neighbor \( v_t > v_i \) of \( v_k \) either satisfies \( v_i < v_t < v_j \) or \( v_k < v_j < v_t \). In both cases, Observation \( 9 \) concludes the proof.

**Computing the canonical dominating set.** Let \( G \) be a dually chordal graph, whose vertices are ordered by an mno. Let \( C = \{c_1, c_2, \ldots, c_k\} \) be a dominating set of \( G \) and \( T = \{t_1, t_2, \ldots, t_k\} \) a set of vertices, both sets in increasing order according to the mno. We say that \( C \) is a triggered dominating set with triggering vertices \( T \) if and only if:

(i) \( c_i = mn(t_i) \) for all \( 1 \leq i \leq k \),

(ii) following the mno, \( t_i \) is the least vertex not in \( N[c_1, \ldots, c_{i-1}] \), for all \( 1 \leq i \leq k \).

It is known that the MINIMUM DOMINATING SET problem is linear-time solvable on dually chordal graphs [8]. In our case, we give another algorithm to compute a triggered dominating set, that will serve as a canonical dominating set.

Observe that an mno is associated with exactly one triggered dominating set. The following algorithm, called MDS, is strongly inspired by the classical algorithm for computing minimum dominating sets in trees [25]. It takes as input a dually chordal graph \( G = (V, E) \) with an mno and computes a triggered dominating set \( C \) of size \( \gamma(G) \) and its corresponding set of triggering vertices \( T \) in running time \( O(|V| + |E|) \).

Lemma 11 is devoted to proving the correctness of the algorithm MDS.

**Lemma 11.** Given a dually chordal graph \( G = (V, E) \), the algorithm MDS computes a triggered dominating set \( C \) of \( G \) of order \( \gamma(G) \) in time \( O(|V| + |E|) \).

**Proof.** The fact that \( C \) is a triggered dominating set with triggering vertices \( T \) is a direct consequence of the construction of the algorithm. Statement \( [i] \) comes from line 5 of the algorithm, while statement \( [ii] \) simply comes from the fact that we deal with the vertices in increasing order in the loop of line 3. Still, we need to prove that this dominating set is of size \( \gamma(G) \).

A labeled graph is a graph whose vertices are labeled FREE, REQUIRED or BOUNDED, such that a vertex is labeled FREE if and only if it is adjacent to a vertex labeled REQUIRED and it is not labeled REQUIRED. Observe that the algorithm MDS maintains all along a labeled graph. In a labeled graph, we define a labeled dominating set a set of vertices containing all the vertices labeled REQUIRED and dominating all vertices labeled BOUNDED. The labeled domination number is the minimum size of a labeled dominating set. We show that the algorithm MDS keeps the labeled domination number of the graph invariant. At the beginning, when all the vertices are labeled BOUNDED, the labeled domination
Lemma 12. Given a dually chordal graph $G = (V, E)$ and a dominating set $D$, **Dually-Chordal-Reconf** modifies $D$ with respect to the token sliding model in such a way that $C \subseteq D$ in $O(|V|)$ time, where $C$ is the canonical triggered dominating set computed by the MDS algorithm.
**Algorithm 2** Dually-Chordal-Reconf

Require: A dually chordal graph $G = (V,E)$, a minimum dominating set $D$ of $G$

1. Compute an mno for $G$
2. $(C,T) \leftarrow$ MDS(G).
3. for $i$ from 1 to $\gamma(G)$ do
4.   Let $x_i$ be the least vertex of $D \cap N[t_i]$ 
5.   if $x_i c_i \in E$ then
6.     $x_i \rightsquigarrow c_i$
7.   else
8.     $y_i \leftarrow mn(x_i)$
9.     $x_i \rightsquigarrow y_i$
10.  $y_i \rightsquigarrow c_i$
11.  end if
12.  end for

**Proof.** Let $T = (t_1, t_2, \ldots, t_n)$ and $C = \{c_1, c_2, \ldots, c_n\}$ be the output of algorithm MDS, with $c_i = mn(t_i)$. We denote by $C_i = \{c_1, \ldots, c_i\}$ the set of $i$ first vertices of $C$ according to the mno.

In order to prove the correctness of the algorithm, we need to prove that the two following constraints are satisfied:

(i) each move is valid with respect to the token sliding model;

(ii) every intermediate set is a dominating set of $G$. (Note that this ensures the existence of the $x_i$ of line 4)

We prove these two properties by induction on the index $i$ ($0 < i \leq \gamma$). For some $i > 0$, assume that the algorithm reconfigured properly $D$ into $D_{i-1} = (D \setminus \{x_1, \ldots, x_{i-1}\}) \cup \{c_1, \ldots, c_{i-1}\}$. We explain how to extend this up to rank $i$. By definition, $t_i$ is the least vertex which is not dominated by $N[C_{i-1}] = N[\{c_1, \ldots, c_{i-1}\}]$. Let $x_i$ be the least vertex dominating $t_i$ in $D$. Observe that $x_i \notin \{c_1, \ldots, c_{i-1}\}$ since $t_i$ is the triggering vertex of $c_i$. To simplify notation, we denote by $G'$ the subgraph of $G$ induced by vertices larger than $t_i$ in the mno (i.e., the subgraph $G_j$ where $j$ is the index of $t_i$ in the mno). Note that since $C_{i-1} \subset D_{i-1}$, all vertices in $G \setminus G'$ are dominated. We consider two cases:

**Case 1.** $x_i$ is adjacent to $c_i$. Observe first that this case occurs whenever $x_i \geq t_i$ in the mno (where $x_i \in N_{G'}[t_i] \subseteq N_{G'}[c_i]$). In that case, the algorithm executes line 6 and the token sliding constraint (ii) is satisfied. Now, since $c_i = mn(t_i)$ and $x_i$ is adjacent to $t_i$, $N_{G'}[x_i] \subseteq N_{G'}[c_i]$, and constraint (ii) is also satisfied. The conclusion follows from the fact that all vertices in $G \setminus G'$ are dominated.

**Case 2.** $x_i$ is not adjacent to $c_i$. This is possibly the case when $x_i < t_i$ in the mno. The algorithm then first reconfigures $x_i$ into its maximum neighbor $y_i$, which is adjacent to $x_i$ and dominates all neighbors of $x_i$ that might not be dominated yet by $\{c_1, \ldots, c_{i-1}\}$, satisfying constraint (ii). Moreover, $x_i$ is adjacent to $t_i$ and $x_i \neq t_i$ in the mno, so $y_i$, as the maximum neighbor of $x_i$, must be adjacent to $t_i$ and all its neighbors in $G'$, among which there is $c_i$. So the next move to $c_i$ satisfies the token sliding constraint (ii). Also, since $y_i$ is adjacent to $t_i$ and $c_i$ is the maximum neighbor of $t_i$, $N_{G'}(y_i) \subseteq N_{G'}[c_i]$, and constraint (ii) is also satisfied. This concludes the proof.

**Theorem 13.** DSR$_{TS}$ can be solved in quadratic time on dually chordal graphs.

Proof. Let $G = (V,E)$ be a dually chordal graph and $D_s, D_t$ be two dominating sets of $G$ of size $k$ (i.e., $(G, D_s, D_t)$ is an instance of the DSR$_{TS}$ problem). Assume that $G$ is connected (otherwise we proceed independently for each connected component, checking first that the number of tokens on each component fits). We explain how to reconfigure $D_s$ into $D_t$ in at most quadratic time.

First, we compute in linear time the canonical dominating set $C$ of $G$ with the algorithm MDS. By Lemma 12 one can transform $D_s$ and $D_t$ in such a way that both contain $C$. This can be done in linear time (with respect to the order of $G$) since we move at most $\gamma(G)$ tokens and each move requires at most two steps. If $k = \gamma(G)$, we are done. Otherwise, choose a vertex $v$ of minimum eccentricity
and move all the remaining tokens by a shortest path to $v$. Therefore, the total time complexity is $O(|V| + |E|) + O(|V|) + O((k - \gamma(G)) \cdot \epsilon(v))$, which is at most quadratic (when $k = \Omega(n)$).

Observe that when $k$ is close to $\gamma$, the algorithm is linear. However, when the number of extra tokens is large (i.e., is linear in $n$), the quadratic overhead may be necessary. Indeed, consider a path on $n$ vertices $P_n$. The minimum eccentricity of $P_n$ is that of the middle vertex $v$ which is $\lfloor n/2 \rfloor$. Therefore, if all the extra tokens are on an extremity of the path, the time needed to move all of them to $v$ is quadratic. Since a path is a dually chordal graph, the conclusion follows.

5 Open questions

In all of our polynomial results presented in Section 4 computing a minimum dominating set can be done in polynomial or even linear time on the graph classes considered. Therefore, a challenging question is the following: does there exist a graph class for which computing a minimum dominating set is NP-complete but $\text{DSR}_{TS}$ can be solved in polynomial time? As a result of Theorem 13 we get that $\text{DSR}_{TS}$ is polynomial-time solvable on interval graphs. Recall that a circle graph is the intersection graph set of chords of a circle. The DOMINATING SET PROBLEM has been shown to be NP-complete on this graph class [22]. Hence, we ask the following question:

**Question 1.** What is the complexity of $\text{DSR}_{TS}$ on circle graphs?

If the answer is positive, note that it would generalize our result on cographs. A circular-arc graph is the intersection graph of a set of arcs on the circle. Even if computing a minimum dominating set can be computed in linear time on circular-arc graphs [21], we are interested in the complexity of the reconfiguration version under token sliding. More precisely, we ask for the following:

**Question 2.** Is $\text{DSR}_{TS}$ polynomial-time solvable on circular-arc graphs?

Besides, we found polynomial-time algorithms for cographs and dually chordal graphs but the underlying reconfiguration sequence is most likely not optimal. Indeed, it may be possible that the shortest path in $R_k(G)$ between the two given solutions does not go through the canonical dominating set. Therefore, can we bound the diameter of the reconfiguration graph? In other words, what is the maximum length of a shortest reconfiguration sequence between any pair of dominating sets? Moreover, what is the complexity of finding the optimal solution, i.e., the shortest reconfiguration sequence between two dominating sets on cographs or dually chordal graphs? Is it polynomial-time solvable, as the reachability variant studied here? Or does it become harder?

Acknowledgements We thank the anonymous referees for several remarks which helped improve the presentation of this paper. This work was supported by ANR project GraphEn (ANR-15-CE40-0009).

References

[1] Hans-Jürgen Bandelt and Henry Martyn Mulder. Distance-hereditary graphs. *Journal of Combinatorial Theory, Series B*, 41(2):182 – 208, 1986.

[2] Alan A. Bertossi. Dominating sets for split and bipartite graphs. *Information Processing Letters*, 19(1):37 – 40, 1984.

[3] Alexandre Blanché, Haruka Mizuta, Paul Ouvrard, and Akira Suzuki. Decremental optimization of dominating sets under the reconfiguration framework. In Leszek Gasieniec, Ralf Krasing, and Tomasz Radzik, editors, *Combinatorial Algorithms*, pages 69–82, Cham, 2020. Springer International Publishing.

[4] Marthe Bonamy, Nicolas Bousquet, Carl Feghali, and Matthew Johnson. On a conjecture of mohar concerning kempe equivalence of regular graphs. *Journal of Combinatorial Theory, Series B*, 135:179 – 199, 2019.
[5] Paul Bonsma and Luis Cereceda. Finding paths between graph colourings: Pspace-completeness and superpolynomial distances. *Theoretical Computer Science*, 410(50):5215 – 5226, 2009. Mathematical Foundations of Computer Science (MFCS 2007).

[6] Paul S. Bonsma. Independent set reconfiguration in cographs and their generalizations. *Journal of Graph Theory*, 83(2):164–195, 2016.

[7] Nicolas Bousquet, Arnaud Mary, and Aline Parreau. Token jumping in minor-closed classes. In Ralf Klasing and Marc Zeitoun, editors, *Fundamentals of Computation Theory*, pages 136–149, Berlin, Heidelberg, 2017. Springer Berlin Heidelberg.

[8] Andreas Brandstädt, Victor D. Chepoi, and Feodor F. Dragan. The algorithmic use of hypertree structure and maximum neighbourhood orderings. *Discrete Applied Mathematics*, 82(1):43 – 77, 1998.

[9] Julia Böttcher, Klaas P. Pruessmann, Anusch Taraz, and Andreas Würfl. Bandwidth, expansion, treewidth, separators and universality for bounded-degree graphs. *European Journal of Combinatorics*, 31(5):1217 – 1227, 2010.

[10] Luis Cereceda, Jan van den Heuvel, and Matthew Johnson. Connectedness of the graph of vertex-colourings. *Discrete Mathematics*, 308(5):913 – 919, 2008. Selected Papers from 20th British Combinatorial Conference.

[11] Paul Dorbec, Gašper Košmrlj, and Gabriel Renault. The domination game played on unions of graphs. *Discrete Math.*, 338(1):71–79, 2015.

[12] Carl Feghali. Paths between colourings of sparse graphs. *European Journal of Combinatorics*, 75:169 – 171, 2019.

[13] Gerd Fricke, Sandra Mitchell Hedetniemi, Stephen T. Hedetniemi, and Kevin R. Hutson. γ-graphs of graphs. *Discussiones Mathematicae Graph Theory*, 31(3):517–531, 2011.

[14] Michael R. Garey and David S. Johnson. *Computers and Intractability; A Guide to the Theory of NP-Completeness*. W. H. Freeman & Co., New York, NY, USA, 1990.

[15] Parikshit Gopalan, Phokion G. Kolaitis, Elitza Maneva, and Christos H. Papadimitriou. The connectivity of boolean satisfiability: Computational and structural dichotomies. *SIAM J. Comput.*, 38(6):2330–2355, March 2009.

[16] R. Haas and K. Seyfarth. The k-dominating graph. *Graphs and Combinatorics*, 30(3):609–617, May 2014.

[17] Arash Haddadan, Takehiro Ito, Amer E. Mouawad, Naomi Nishimura, Hirotaka Ono, Akira Suzuki, and Youcef Tebbal. The complexity of dominating set reconfiguration. *Theor. Comput. Sci.*, 651(C):37–49, October 2016.

[18] Robert A. Hearn and Erik D. Demaine. 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, October 2005.

[19] Takehiro Ito, Erik D. Demaine, Nicholas J.A. Harvey, Christos H. Papadimitriou, Martha Sideri, Ryuhei Uehara, and Yushi Uno. On the complexity of reconfiguration problems. *Theoretical Computer Science*, 412(12):1054 – 1065, 2011.

[20] Takehiro Ito, Haruka Mizuta, Naomi Nishimura, and Akira Suzuki. Incremental optimization of independent sets under the reconfiguration framework. In *Computing and Combinatorics - 25th International Conference, COCOON 2019, Xi’an, China, July 29-31, 2019, Proceedings*, pages 313–324, 2019.

[21] Marcin Kamiński, Paul Medvedev, and Martin Milanič. Complexity of independent set reconfigurability problems. *Theoretical Computer Science*, 439:9 – 15, 2012.
[22] J. Mark Keil. The complexity of domination problems in circle graphs. *Discrete Applied Mathematics*, 42(1):51 – 63, 1993.

[23] Daniel Lokshtanov and Amer E. Mouawad. The complexity of independent set reconfiguration on bipartite graphs. In *Proceedings of the Twenty-Ninth Annual ACM-SIAM Symposium on Discrete Algorithms*, SODA 2018, New Orleans, LA, USA, January 7-10, 2018, pages 185–195, 2018.

[24] Daniel Lokshtanov, Amer E. Mouawad, Fahad Panolan, M.S. Ramanujan, and Saket Saurabh. Reconfiguration on sparse graphs. *Journal of Computer and System Sciences*, 95:122 – 131, 2018.

[25] S.L. Mitchell, E.J. Cockayne, and S.T. Hedetniemi. Linear algorithms on recursive representations of trees. *Journal of Computer and System Sciences*, 18(1):76 – 85, 1979.

[26] A. Mouawad, N. Nishimura, V. Pathak, and V. Raman. Shortest reconfiguration paths in the solution space of boolean formulas. *SIAM Journal on Discrete Mathematics*, 31(3):2185–2200, 2017.

[27] Amer E. Mouawad, Naomi Nishimura, Venkatesh Raman, Narges Simjour, and Akira Suzuki. On the parameterized complexity of reconfiguration problems. *Algorithmica*, 78(1):274–297, May 2017.

[28] C. M. Mynhardt and S. Nasserasr. Reconfiguration of colourings and dominating sets in graphs: a survey, 2020.

[29] Falk Nicolai and Thomas Szymczak. Homogeneous sets and domination: A linear time algorithm for distance—hereditary graphs. *Networks*, 37(3):117–128, 2001.

[30] Naomi Nishimura. Introduction to reconfiguration. *Algorithms*, 11(4):52, 2018.

[31] Maw shang Chang. Efficient algorithms for the domination problems on interval and circular-arc graphs. *SIAM J. Comput*, 27:1671–1694, 1998.

[32] Akira Suzuki, Amer E. Mouawad, and Naomi Nishimura. Reconfiguration of dominating sets. In Zhipeng Cai, Alex Zelikovsky, and Anu Bourgeois, editors, *Computing and Combinatorics*, pages 405–416, Cham, 2014. Springer International Publishing.

[33] Jan van den Heuvel. The complexity of change. In Simon R. Blackburn, Stefanie Gerke, and Mark Wildon, editors, *Surveys in Combinatorics*, volume 409 of *London Mathematical Society Lecture Note Series*, pages 127–160. Cambridge University Press, 2013.

[34] Marcin Wrochna. Reconfiguration in bounded bandwidth and tree-depth. *Journal of Computer and System Sciences*, 93:1 – 10, 2018.