ECAI 2023
K. Gal et al. (Eds.)

doi:10.3233/FAIA230334

On Solution Discovery via Reconfiguration

Michael R. Fellows [a], Mario Grobler [b], Nicole Megow [b], Amer E. Mouawad [c], Vijayaragunathan Ramamoorthi [b], Frances A. Rosamond [a], Daniel Schmand [b] and Sebastian Siebertz [b]

[a]University of Bergen, Norway
[b]University of Bremen, Germany
[c]American University of Beirut, Lebanon
ORCiD ID: Michael R. Fellows https://orcid.org/0000-0002-6148-9212, Mario Grobler https://orcid.org/0000-0001-8103-6440, Nicole Megow https://orcid.org/0000-0002-3531-7644, Amer E. Mouawad https://orcid.org/0000-0003-2481-4968, Vijayaragunathan Ramamoorthi https://orcid.org/0000-0001-8554-6392, Frances A. Rosamond https://orcid.org/0000-0002-5097-9929, Daniel Schmand https://orcid.org/0000-0001-7776-3426, Sebastian Siebertz https://orcid.org/0000-0002-6347-1198

Abstract. The dynamics of real-world applications and systems require efficient methods for improving infeasible solutions or restoring corrupted ones by making modifications to the current state of a system in a restricted way. We propose a new framework of *solution discovery via reconfiguration* for constructing a feasible solution for a given problem by executing a sequence of small modifications starting from a given state. Our framework integrates different aspects of classical local search, reoptimization, and combinatorial reconfiguration. We exemplify our framework on a multitude of fundamental combinatorial problems, namely Vertex Cover, Independent Set, Dominating Set, and Coloring. We study the classical as well as the parameterized complexity of the solution discovery variants of those problems and explore the boundary between tractable and intractable instances.

1 Introduction

In many dynamic real-world applications of decision-making problems, feasible solutions must be found starting from a certain predetermined system state. This is in contrast to the classical academic problems where we are allowed to compute a feasible solution from scratch. However, when constructing a new solution from scratch, we have no control over the difference between the current state and the new target one. In this work, we develop a new framework, where we aim for modifying some, possibly infeasible, solution to a feasible one via a bounded number of "small" modification steps.

As an example, consider a frequency assignment problem [1, 24] where a number of agents communicate via wireless message transmissions. Each agent can broadcast over a fixed frequency. Any two nearby agents are required to operate over different frequencies, as otherwise their signals would interfere, which must be avoided. The objective is to assign a given number of frequencies such that no interference occurs. This problem can conveniently be modeled as a graph coloring problem. The vertices of the graph represent the agents and two vertices are connected by an edge if they are close to one another and frequencies of the corresponding agents could interfere. Then, the frequencies correspond to colors assigned to the vertices, as adjacent vertices must receive distinct colors.

Now, consider the situation where agents have already been assigned frequencies that cause some interference. This might happen to a previously correct (non-interfering) system, e.g., because new agents were added, locations changed, or the broadcasting range increased. Recomputing a valid coloring from scratch might be undesirable as this may induce many changes in the system. Depending on the given infeasible assignment, it might be possible to satisfy the feasibility constraints by just a few controlled changes to the coloring. A change however, may trigger other changes, and the question is whether a "cheap" reconfiguration of the system into a valid state is possible. In terms of graph coloring, the solution discovery variant of the problem has as input a non-proper coloring of a graph using a fixed number of colors and a fixed budget. The goal is to transform the given non-proper coloring into a proper one such that the number of recoloring steps does not exceed the budget (various definitions of a recoloring step exist; we discuss the most common ones later).

As a second motivation for our framework, consider a group of mobile agents tasked with monitoring (parts of) a country (represented by a graph) for security threats, such as natural disasters. Each agent is responsible for monitoring a certain subset of the country (a few cities), and the cities are connected in a way that reflects the adjacencies between the different parts of the network. The agents in a dominating set would represent agents that are able to monitor the entirety of the parts of interest. By identifying a small dominating set in this graph, it is possible to identify/place a small number of agents that are responsible for maintaining the security of the country. In a real-world application, agents can be retired from or added to the system or the areas of interest to monitor (i.e. cities) can also change over time. Given that it is most likely expensive to move the agents around (think of agents as being mobile monitoring vehicles), it is desirable to be able to compute a new dominating set in the modified system while accounting for the cost of transformation steps.

In this work, we propose a new framework that models such solution discovery problems. We focus on fundamental graph problems, where we are given a graph G and an integer k and the goal is to find a feasible solution of size (at least/most) k, e.g., finding a vertex subset with certain properties (such as an independent set, vertex cover, dominating set) of size k or a proper vertex coloring using at most k colors. We introduce the *solution discovery* variant of such problems, where we are given a graph G, a starting configuration of size k (which is not necessarily a feasible solution), and a budget b. The goal is to transform the starting configuration into a feasible solution using at most b modification steps.

We restrict the modification steps to changes along edges of the graph that can be described as *token sliding*. We borrow this notion from the area of *combinatorial reconfiguration* [40, 37] and we may also discuss related notions such as *token addition/removal resp. swapping* and *jumping*. The restriction of modification steps in this way does not only give a precise characterization of a "small" change in the system state, but it also appropriately captures situations in which a solution can be modified by moving entities along edges in a graph such as in the two examples presented above. More generally, we model applications where items/agents can be moved only along links in a given physical network such as road networks or computer networks, or whenever there are restrictions on the movement trajectories.

The solution discovery framework is related to other approaches transforming one solution into another such as *local search* [2], *reoptimization* [5], *dynamic algorithms* [27], and *combinatorial reconfiguration* [37, 40]. The key characteristics of our framework are: we ask for finding a feasible solution starting from a *predefined system state* (as in local search, dynamic algorithms, reoptimization), but we restrict the *local modification steps* as is in combinatorial reconfiguration.

We demonstrate our framework by applying it to the Vertex Cover, Independent Set, Dominating Set, and Coloring problems. These are prominent combinatorial problems in artificial intelligence, see e.g. [11, 26, 28, 35, 39, 41], with plenty of applications, including feature selection [43], scheduling, planning, resource allocation [4], frequency assignment [1], network security [12], sensor systems [32], etc. These are central and well-studied problems also from the perspective of local search, reoptimization, and combinatorial reconfiguration.

We initiate the study of solution discovery problems. We identify polynomial-time solvable cases and show that most solution discovery problems are NP-hard already on quite restricted classes of graphs and hence considered intractable from a classical complexity-theoretic point of view. Therefore, we also apply the theory of *parameterized complexity* [18], which provides a powerful framework to overcome this obstacle. The key goal in this theory is to find one or more additional dimensions by which to measure the inputs to (NP-hard) computational problems, called the parameter(s), and to provide algorithms whose running time restricts the combinatorial explosion to the parameter(s). On instances where the parameter values are relatively small, parameterized algorithms are efficient. For our applications, we are naturally concerned with finding small sequences from initial to target configurations. Therefore, the number of reconfiguration steps, i.e., the budget, is one ideal candidate for parameterization. Other natural candidate parameters are the size of configurations as well as structural graph parameters.

1.1 Solution discovery via reconfiguration

We formally define the solution discovery variant of graph vertex-subset problems as follows. Let Π be a vertex subset problem, i.e., a problem defined on undirected graphs such that a solution consists of a subset of vertices. The Π-Discovery problem is defined as follows. We are given a graph G, a subset $S \subseteq V(G)$ of size k (which at this point is not necessarily a solution for Π), and a budget b (as a positive integer). We assume that each vertex in S contains a token (which corresponds to an agent). The goal is to decide if we can move the tokens on S using at most b moves such that the resulting set is a solution for Π.

Depending on the underlying problem, different notions of token moves have been established in the reconfiguration literature. That is, for token configurations $Q, Q' \subseteq V(G)$, we have the following models. In the *token sliding model*, we say a token *slides* from $u \in V(G)$ to $v \in V(G)$ if $u \in Q$, $v \notin Q$, $v \in Q'$, $u \notin Q'$, and $\{u, v\} \in E(G)$. In the *token jumping model*, a token *jumps* from $u \in V(G)$ to $v \in V(G)$ if $u \in Q$, $v \notin Q$, $v \in Q'$, and $u \notin Q'$. Finally, in the *token addition/removal model*, a token is *added* to vertex $v \in V(G)$ if $v \notin Q$ and $v \in Q'$. Similarly, a token is *removed* from vertex $v \in V(G)$ if $v \in Q$ and $v \notin Q'$.

We note that we focus on the sliding model for vertex subset problems as most of our results for Π-Discovery are based on it. By instantiating Π with Vertex Cover, Independent Set, or Dominating Set, we obtain the Vertex Cover Discovery, Independent Set Discovery, or Dominating Set Discovery problem, respectively.

Similar to the vertex subset discovery problems, we define the Coloring Discovery problem and specify the reconfiguration moves in Section 5. Coloring is a very fundamental combinatorial problem and does not fall into the class of vertex subset problems. We selected it to exemplify the richness of our solution discovery framework beyond problems in which the solution is a single vertex set satisfying certain properties.

1.2 Our results

We study the classical as well as the parameterized complexity of solution discovery problems and prove the following results. All proofs that are missing due to space constraints can be found in the appended full version. To formally state our results, we assume some familiarity with graph theory, the considered problems, and (parameterized) complexity.

Vertex Cover Discovery: We show that the problem is polynomial-time solvable on every fixed graph class of bounded treewidth (that is, XP for parameter treewidth), FPT for parameter k on general graphs and FPT for parameter b when restricted to structurally nowhere dense classes of graphs. On the negative side, we show that the problem is NP-complete even on planar graphs of maximum degree four and W[1]-hard for parameter b, even on 2-degenerate bipartite graphs.

Independent Set Discovery: We show that the problem is XP for parameter treewidth, FPT for parameter k on d-degenerate and nowhere dense classes as well as FPT for parameter b on structurally nowhere dense classes of graphs. On the negative side, we show that the problem is NP-complete even on planar graphs of maximum degree four, W[1]-hard for parameter $k+b$ even on graphs excluding $\{C_4, \ldots, C_p\}$ as induced subgraphs (for any constant p), and W[1]-hard for parameter b even on 2-degenerate bipartite graphs.

Dominating Set Discovery: We show that the problem is XP for parameter treewidth, FPT for parameter k on biclique-free and semi-ladder-free graphs as well as FPT for parameter b on structurally nowhere dense classes of graphs. On the negative side, we show that the problem is NP-complete even on planar graphs of maximum degree five, W[1]-hard for parameter $k+b$ even on bipartite graphs, and W[1]-hard for parameter b even on 2-degenerate graphs.

Coloring Discovery: We show that the problem is polynomial-time solvable for $k=2$ and FPT parameterized by $k+b$ on structurally nowhere dense classes of graphs. On the negative side, we show that the problem is NP-complete for every $k \geq 3$ even on planar bipartite graphs (that is, para-NP-hard for parameter k), W[1]-hard for parameter treewidth, and W[1]-hard for parameter $k+b$ on general graphs.

Our results provide an almost complete classification of the complexity of the problems on minor-closed, resp. monotone (i.e. subgraph-closed) classes of graphs. Concerning the polynomial-time solvable cases, observe that a class of graphs has bounded treewidth if and only if it excludes a planar graph as a minor [38]. Hence, for the vertex subset discovery problems, hardness on planar graphs implies that for minor-closed classes of graphs efficient algorithms can only be expected for classes with bounded treewidth, which is exactly what we establish. Coloring Discovery is NP-complete both on planar graphs and on classes with bounded treewidth, for any $k \geq 3$, hence, our tractability result for $k=2$ is the best we can hope for. It remains open whether Coloring Discovery is tractable for parameter $k+b$ on d-degenerate graphs, parameter b on planar graphs, and parameter k on classes with bounded treewidth.

The notions of degeneracy and nowhere density are very general notions of graph sparsity. Besides the theoretical interest in these classes, many real-world networks, including several social networks (such as physical disease propagation networks), biological networks (such as gene interactions or brain networks), and informatics networks (such as autonomous systems), turn out to be degenerate or nowhere dense, see e.g. [10, 16, 21]. The further considered classes, like biclique-free classes, semi-ladder-free classes and structurally nowhere dense classes are dense generalizations of these classes that preserve some of their good algorithmic properties.

We provide an almost complete classification on monotone classes of graphs for the vertex-subset discovery problems and the parameter b; we establish hardness on degenerate classes and tractability on (structurally) nowhere dense classes. For parameter k, Vertex Cover Discovery is tractable on all graphs. Similarly, for parameter k, Independent Set Discovery is tractable both on degenerate and on nowhere dense classes, and it remains open whether it is tractable on biclique-free classes. Dominating Set Discovery is tractable even on semi-ladder-free, and hence on biclique-free classes.

Organization. In the presentation of our results, we mainly focus on the Independent Set Discovery problem. We selected it as a representative for vertex subset problems and demonstrate the techniques and the kind of theoretical results that will be obtained in our framework. We further discuss results for the Coloring Discovery problem, which does not fall into the class of vertex subset problems, to reflect the broad applicability of our framework. Due to space constraints, all of our remaining results are presented in the appended version.

2 Preliminaries

We denote the set of natural numbers by $\mathbb{N}$. For $k \in \mathbb{N}$ we define $[k] = \{1, 2, \ldots, k\}$.

Graphs. We consider finite, simple, loopless, and undirected graphs. For a graph G, we denote by $V(G)$ and $E(G)$ the vertex set and edge set of G, respectively. Two vertices $u, v \in V(G)$ with $\{u, v\} \in E(G)$ are called adjacent or neighbors. The vertices u and v are called the endpoints of the edge $\{u, v\}$. The degree of a vertex v is the number of neighbors of v. A sequence $v_1, \ldots, v_q$ of pairwise distinct vertices is a path of length $q-1$ if $\{v_i, v_{i+1}\} \in E(G)$ for all $1 \leq i < q$. A sequence $v_1, \ldots, v_q$ of pairwise distinct vertices is a cycle of length q if $\{v_i, v_{i+1}\} \in E(G)$, for all $1 \leq i < q$, and $\{v_q, v_1\} \in E(G)$. We write P_q to denote a path of length q and C_q to denote a cycle of length q.

Parameterized complexity. An instance of a *parameterized problem* $L \subseteq \Sigma^* \times \mathbb{N}$, where Σ is a fixed finite alphabet, is a tuple $(x, \kappa) \in \Sigma^* \times \mathbb{N}$. The number κ is called the *parameter* of the instance. The problem L is called *fixed-parameter tractable*, FPT for short, if there exists an algorithm that on input (x, κ) decides in time $f(\kappa) \cdot |(x, \kappa)|^c$ whether $(x, \kappa) \in L$, for a computable function f and constant c. Likewise, the problem belongs to para-NP if it can be solved within the same time bound by a nondeterministic algorithm. L is *slice-wise polynomial*, XP for short, if there is an algorithm deciding whether (x, κ) belongs to L in time $f(\kappa) \cdot |(x, \kappa)|^{g(\kappa)}$, for computable functions f, g.

The *W-hierarchy* is a collection of parameterized complexity classes $\mathsf{FPT} \subseteq \mathsf{W[1]} \subseteq \ldots \subseteq \mathsf{W[t]} \subseteq \ldots \subseteq \mathsf{para\text{-}NP} \cap \mathsf{XP}$. We have $\mathsf{FPT} \neq \mathsf{para\text{-}NP}$ if and only if $\mathsf{P} \neq \mathsf{NP}$, which is a standard assumption. Also the inclusion $\mathsf{FPT} \subseteq \mathsf{W[1]}$ is conjectured to be strict (and this is known to be true when assuming the exponential-time hypothesis). Therefore, showing intractability in the parameterized setting is usually accomplished by establishing an FPT-reduction from a W-hard problem.

Let $L, L' \subseteq \Sigma^* \times \mathbb{N}$ be parameterized problems. A *parameterized reduction* from L to L' is an algorithm that, given an instance (x, κ) of L, outputs an instance (x', κ') of L' such that $(x, \kappa) \in L \Leftrightarrow (x', \kappa') \in L'$, $\kappa' \leq g(\kappa)$ for some computable function g, and the running time of the algorithm is bounded by $f(\kappa) \cdot |(x, \kappa)|^c$ for some computable function f and constant c.

3 Independent set discovery

In the Independent Set (IS) problem, we are given a graph G and an integer k and the problem is to decide whether G

contains an independent set of size at least k, where an independent set is a set of pairwise non-adjacent vertices.

In the Independent Set Discovery (ISD) problem, we are given a graph G, a starting configuration S given by k tokens, and a budget $b \in \mathbb{N}$. The goal is to decide whether we can reach an independent set of G (of size k) starting from S using at most b token slides (we do not allow two or more tokens to occupy the same vertex). We denote an instance of ISD by (G, S, b); the considered parameter will be explicit in the text.

Note that, for Independent Set Discovery, the token jumping model and token addition/removal model boil down to the following problems. If $k \leq b$ for token jumping or $k \leq 2b$ for token/addition removal, then the question is simply whether there exists a solution of size k, as in this case we can simply move the tokens to this solution one by one. In other words, the problem boils down to the classical Independent Set problem. If $b \leq k$ (for token jumping) or $2b \leq k$ (for token addition/removal), then the question is whether there exists a solution whose symmetric difference with the initial configuration is at most $2b$. This question has been studied in the local search version of Independent Set; see Section 3.1. We therefore focus on the token sliding model.

3.1 Related work

The Independent Set problem is NP-complete [31], and even NP-complete to approximate within a factor of $n^{1-\varepsilon}$, for any $\varepsilon > 0$ [44]. The Independent Set problem parameterized by solution size k is W[1]-complete and therefore assumed to not be fixed-parameter tractable [17].

Hence, on general graphs, local search approaches cannot be expected to improve the above stated approximation factor. However, in practice we are often dealing with graphs belonging to special graph classes, e.g. planar graphs and, more generally, classes with subexponential expansion, where local search leads to much better approximation algorithms, and even to polynomial-time approximation schemes (PTAS), see e.g. [25].

The Independent Set problem is one of the most studied problems under the combinatorial reconfiguration framework [37, 40]. Recall that in the reconfiguration variant of a (graph vertex subset) problem, we are given a graph G and two feasible solutions S (source) and T (target) and the goal is to decide whether we can transform S to T via "small" reconfiguration steps while maintaining feasibility (sometimes bounding the number of allowed reconfiguration steps).

In contrast to the decision variant, Independent Set Reconfiguration (ISR) is known to be PSPACE-complete on general graphs for the token sliding, token jumping, and token addition/removal models [29, 30]. This remains true even for very restricted graph classes such as graphs of bounded bandwidth/pathwidth/treewidth [42]. On the positive side, polynomial-time algorithms are known only for very simple graph classes such as trees [15]. More positive results (which vary depending on the model) are possible if we consider the parameterized complexity of the problem [9].

3.2 Tractability

We first show that Independent Set Discovery is polynomial time solvable on every graph class of bounded treewidth. Our proof is based on dynamic programming techniques on graphs of bounded treewidth. Note that k in the theorem is in $\mathcal{O}(n)$ so that $2^{\mathcal{O}(t \log k)} \in n^{\mathcal{O}(t)}$. Hence, the problem is XP when parameterized by the treewidth of the input graph.

Theorem 1 *The Independent Set Discovery problem can be solved in time $2^{\mathcal{O}(t \log k)} \cdot n^{\mathcal{O}(1)}$, where t denotes the treewidth of the input graph.*

Our positive results for Independent Set Discovery parameterized by k make use of the notion of independence covering families introduced in [34]. Intuitively, such families cover all independent sets of a fixed size k. Formally, for a graph G and $k \geq 1$, a family of independent sets of G is called an *independence covering family for* (G, k), denoted by $\mathcal{F}(G, k)$, if for every independent set I in G of size at most k, there exists $J \in \mathcal{F}(G, k)$ such that $I \subseteq J$.

Theorem 2 ([34]) *Let $\mathscr{C}$ be a d-degenerate or nowhere dense class of graphs. For every graph $G \in \mathscr{C}$, and $k \geq 1$, we can compute in time $f(k) \cdot n^{\mathcal{O}(1)}$ an independence covering family for (G, k) of size at most $g(k) \cdot n^{\mathcal{O}(1)}$, where $f(k)$ and $g(k)$ are functions depending only on k and the class $\mathscr{C}$ but are independent of the size of the graph.*

We are now ready to prove the following theorem:

Theorem 3 *The Independent Set Discovery problem is fixed-parameter tractable when parameterized by k for every class $\mathscr{C}$ of graphs that admits independence covering families of size $g(k) \cdot n^{\mathcal{O}(1)}$ computable in time $f(k) \cdot n^{\mathcal{O}(1)}$, where f and g are computable functions. In particular, the problem is fixed-parameter tractable on d-degenerate and nowhere dense classes of graphs.*

Proof. Given an instance (G, S, b) of ISD where $G \in \mathscr{C}$, we start by computing an independence covering family $\mathcal{F}(G, k)$ of size $g(k) \cdot n^{\mathcal{O}(1)}$ in time $f(k) \cdot n^{\mathcal{O}(1)}$, which is possible by assumption (or by Theorem 2 for d-degenerate and nowhere dense classes of graphs). Let $\mathcal{F}(G, k) = \{J_1, J_2, \ldots\}$ denote the resulting family. Let $J \in \mathcal{F}(G, k)$. We construct a complete weighted bipartite graph $H_{S,J}$ as follows. Let S be the vertices S on one side and J be the vertices on the other side. We set the weight of each edge $\{u, v\}$ to be the number of edges along a shortest path from u to v in G (we set the weight to $m + b + 1$ whenever u and v belong to different components).

It remains to show that (G, S, b) is a yes-instance if and only if there exists a $J \in \mathcal{F}(G, k)$ such that $|J| \geq k$ and the minimum weight perfect matching in $H_{S,J}$ has weight at most b. Assuming the previous claim, the algorithm then follows by simply iterating over each J of size at least k, constructing the graph $H_{S,J}$, and then computing a minimum weight perfect matching in $H_{S,J}$. If we find a matching of weight at most b then we have a yes-instance; otherwise we have a no-instance.

Assume that (G, S, b) is a yes-instance. Then, there exists an independent set I of size k that can be reached from S by at most b token slides. By the definition of independence covering families, there exists $J \in \mathcal{F}(G, k)$ such that $I \subseteq J$. Moreover, since I is reachable from S in at most b slides, it must be the case that the weight of a perfect matching in $H_{S,I}$ is at most b. Hence, the minimum weight perfect matching in $H_{S,J}$ has weight at most b, as needed.

Now assume that there exists a $J \in \mathcal{F}(G,k)$ of size at least k such that the minimum weight perfect matching in $H_{S,J}$ has weight at most b. Recall that, by the definition of independence covering families, J is an independent set in G. Hence, any subset I of J of size k is an independent set of size k in G. Let I denote the set of vertices that are matched to some vertex in S in the minimum weight perfect matching. As we just described, I is an independent set of size k in G. Hence, it remains to show that we can reach I from S using at most b slides. Since we do not allow two tokens to lie on the same vertex at any time, we can resolve conflicts as follows. Assume the path that a token t_1 takes to reach its destination has another token t_2 on it. Then we switch their destinations and thereby resolve the conflict. One can check that the number of moves does not exceed the weight of the perfect matching.

It is well-known that Minimum Weight Perfect Matching can be solved in $\mathcal{O}(n^3)$ time using either the blossom algorithm [20] or the Hungarian algorithm [33]. ☐

By a reduction to first-order model checking on structurally nowhere dense classes of graphs (which is fixed-parameter tractable parameterized by formula length [19]), we can show the following result.

Theorem 4 *The Independent Set Discovery problem is fixed-parameter tractable when parameterized by the budget b and restricted to structurally nowhere dense classes of graphs.*

3.3 Intractability

We next establish that our results are essentially optimal by proving hardness results on more general classes of graphs. First, note that for all solution discovery variants of (graph) vertex subset problems, we can always assume that $b \leq n^2$, where n is the number of vertices in the input graph. This follows from the fact that each token will have to traverse a path of length at most n. Hence, all the solution discovery variants of such problems are indeed in NP and it remains to prove NP-hardness.

Theorem 5 *The Independent Set Discovery problem is NP-complete on planar graphs of maximum degree four.*

Proof. We give a reduction from IS on planar graphs of maximum degree three, which is known to be NP-complete [36]. Given an instance (G,κ) of IS, where G is a planar graph of maximum degree three, we construct an instance of ISD as follows. We create a new graph H that initially consists of a copy of G. Then, for each vertex $v \in V(H)$, we create a new path on five vertices w_v, x_v, c_v, y_v, z_v and we connect v to c_v (this will be called a path gadget). We choose $S = \{c_v, x_v, y_v \mid v \in V(G)\}$ and we set the budget $b = 2n - \kappa$, where $n = |V(G)|$. Note that $k = |S| = 3|V(G)|$. This completes the construction of the instance (H,S,b). It is easy to observe that the graph H is planar and of maximum degree four. We prove that (G,κ) is a yes-instance of IS if and only if (H,S,b) is a yes-instance of ISD.

First assume that G has an independent set I of size at least κ. Then, in H we can slide every token on c_v to v, where $v \in I$. For all other vertices $v \notin I$ we slide every token on x_v to w_v and every token on y_v to z_v. Observe that we need a budget of 2 to repair the path on every vertex $v \notin I$, while we need only a budget of 1 to repair the paths on vertices $v \in I$. Since I is of size at least κ, we need no more than $2n - \kappa = b$ slides. To see that the resulting set is an independent set of H, note that for every path on a vertex $v \in I$ we have moved the token from c_v to v itself. As I is an independent set, and the only conflicting neighbor of x_v resp. y_v is c_v, the tokens from these paths form an independent set. The tokens on paths of vertices $v \notin I$ also form an independent set. As the only neighbor of w_v is x_v and the token has been moved from x_v to w_v, hence there is no conflict. This is also true for y_v and z_v. As the neighbors x_v and y_v of c_v have been freed, and there is no token on v itself, i.e., these paths form an independent set.

For the reverse direction, assume that (H,S,b) is a yes-instance of ISD. Let I be the resulting independent set. We need to show that $|I \cap V(G)| \geq \kappa$, which then corresponds to an independent set in G. Assume towards a contradiction that $|I \cap V(G)| = \ell < \kappa$. This implies that $3n - \ell$ tokens are still on the path gadgets. Since every path gadget can contain at most 3 independent vertices and I is an independent set, at least $n - \ell$ path gadgets contain 3 tokens. It takes at least 2 slides to keep the 3 tokens independent while not moving them out of the path. Hence, we require a budget of at least $2n - 2\ell$ for these slides. Moreover, each of the ℓ tokens on $V(G)$ require at least one slide. In total, we require a budget of $2n - \ell > 2n - \kappa$, a contradiction. ☐

We next show that the problem is also hard from a parameterized perspective when considering the parameter $k + b$.

Theorem 6 *The Independent Set Discovery problem is W[1]-hard when parameterized by $k+b$ even on graphs excluding $\{C_4, \ldots, C_p\}$ as induced subgraphs, for any constant p.*

Proof. We present a parameterized reduction from the Multicolored Independent Set (MIS) problem, which is known to be W[1]-hard on graphs excluding $\{C_4, \ldots, C_p\}$ as induced subgraphs, for any constant p [6]. Recall that in the MIS problem we are given a graph G and an integer κ, where $V(G)$ is partitioned into κ cliques $V_1, V_2, \ldots, V_\kappa$, and the goal is to find a multicolored independent set of size κ, i.e., an independent set containing one vertex from each set V_i, for $i \in [\kappa]$. Given an instance (G,κ) of MIS, we construct an instance (H,S,b) of ISD as follows. First, let H be a copy of G. Then, for each $i \in [\kappa]$, we add an edge on two new vertices $\{u_i, w_i\}$ and we make u_i adjacent to all vertices in V_i. Finally, we choose $S = \{u_i \mid i \in [\kappa]\} \cup \{w_i \mid i \in [\kappa]\}$ and we set $b = \kappa$. Note that $k = |S| = 2\kappa$.

Assume that G has a multicolored independent set of size κ. Let $I = \{v_1, \ldots, v_\kappa\}$ denote such a set, where $v_i \in V_i$. Then we can solve the discovery instance by sliding each token on u_i to the vertex v_i, as needed. For the reverse direction, since we need to slide all the tokens on vertices u_i and each set V_i can contain only one token, it follows that this is only possible if G has a multicolored independent set of size κ. ☐

In what follows, we further investigate the complexity of the Independent Set Discovery problem when parameterized only by b or k instead of $k + b$ on even more restricted graph classes. It turns out that the problem remains W[1]-hard when parameterized by b and restricted to the class of 2-degenerate bipartite graphs.

Theorem 7 *The* Independent Set Discovery *problem is* W[1]*-hard when parameterized by the budget b even on the class of 2-degenerate bipartite graphs.*

Proof. We present a parameterized reduction from the Multicolored Clique problem, which is a well-known W[1]-hard problem [14]. Recall that in the Multicolored Clique problem we are given a graph G and an integer κ, where $V(G)$ is partitioned into κ independent sets $V_1, V_2, \ldots, V_\kappa$, and the goal is to find a multicolored clique of size κ, i.e., a clique containing one vertex from each set V_i, for $i \in [\kappa]$. Given an instance (G, κ) of Multicolored Clique, we construct an instance (H, S, b) of ISD as follows. We first describe the graph H. For each vertex set V_i, we create a new set X_i, where each vertex $v \in V_i$ is replaced by a path on 3 vertices which we denote by x_v, y_v, and z_v. Moreover, we add a vertex u_i that is connected to all vertices z_v, for $v \in V_i$. Finally, we add a vertex w_i that is only adjacent to vertex u_i. For $i < j \in [\kappa]$, we use $E_{i,j}$ to denote the set of edges between vertices in V_i and vertices in V_j (in the graph G). For each $E_{i,j}$, we create a new set of vertices, which we denote by $Y_{i,j}$, that contains one vertex e_{uv} for each edge $\{u, v\} \in E_{i,j}$. We additionally add an edge (consisting of two new vertices) $\{w_{i,j}, u_{i,j}\}$, where $u_{i,j}$ is also adjacent to every vertex in $Y_{i,j}$. For each vertex $e_{uv} \in Y_{i,j}$, $i < j \in [\kappa]$, we connect e_{uv} to y_u via a path consisting of two new vertices and we connect e_{uv} to y_v via a path consisting of two new vertices. Let $X = \bigcup_{i \in [\kappa]} X_i$ and $Y = \bigcup_{i,j \in [\kappa]} Y_{i,j}$. We use W to denote the set of all vertices along paths from Y to X that are at distance one from some vertex in Y and we use Z to denote the set of all vertices along such paths that are at distance two from some vertex in Y. Finally, let $S = W \cup \{u_i \mid i \in [\kappa]\} \cup \{w_i \mid i \in [\kappa]\} \cup \{u_{i,j} \mid i < j \in [\kappa]\} \cup \{w_{i,j} \mid i < j \in [\kappa]\} \cup \{y_v \mid v \in V(G)\}$ and let $b = 3\binom{\kappa}{2} + 2\kappa$ and $k = 2\kappa + 2\binom{\kappa}{2} + n + 2m$ (see Figure 1).

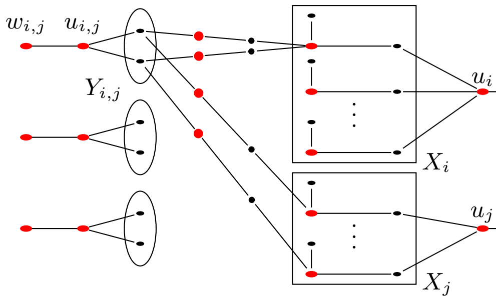

Figure 1. An illustration of the W[1]-hardness reduction for Independent Set Discovery on 2-degenerate bipartite graphs.

It is not hard to see that the graph H is indeed bipartite by construction. To see that the graph H is 2-degenerate it suffices to note that after deleting all y_v vertices, u_i vertices, and $u_{i,j}$ vertices, we get a graph of maximum degree two and none of the deleted vertices are adjacent. Hence every subgraph of H either has a vertex of the form w_i, $w_{i,j}$, x_v, z_v, or a vertex in $W \cup Y$ which all have degree at most 2, or no such vertex implying that every other vertex has degree at most 2. We claim that (G, κ) is a yes-instance of Multicolored Clique if and only if (H, S, b) is a yes-instance of ISD.

First assume that (G, k) is a yes-instance and let $C = \{v_1, v_2, \ldots v_\kappa\}$ denote a multicolored clique in G, where each vertex v_i belongs to V_i, for $i \in [\kappa]$. We construct a sequence of slides transforming S to an independent set as follows. For each i, we slide the token on y_{v_i} to x_{v_i} and then slide the token on u_i to z_{v_i}. This requires a total of 2κ slides. Next, for each pair i, j with $i < j$, we slide the token on $u_{i,j}$ to the vertex $e_{v_i v_j}$ and then slide the two tokens in W to their neighbors in Z. This requires a total of $3\binom{\kappa}{2}$ slides which gives us a total of $b = 3\binom{\kappa}{2} + 2\kappa$ slides. Since C is a multicolored clique in G, it follows that the resulting configuration is indeed an independent set of H.

For the reverse direction, assume that (H, S, b) is a yes-instance of ISD. Since we have two adjacent tokens on u_i and w_i, for each $i \in [\kappa]$, we know that we need at least one move for each i. Moreover, since every vertex y_v contains a token, we know that we need an extra slide for each i. Hence, we need a minimum of 2κ slides for the edges of the form $\{u_i, w_i\}$. Similarly, for each pair i, j with $i < j$, we have two adjacent tokens on $w_{i,j}$ and $u_{i,j}$. Moreover, all vertices in W contain tokens. Hence, for each pair i, j with $i < j$ we need at least three slides for a total of $3\binom{\kappa}{2}$ slides for the edges of the form $\{u_{i,j}, w_{i,j}\}$. Hence, there must exist κ vertices y_{v_i} and $\binom{\kappa}{2}$ vertices $e_{v_i v_j}$ adjacent to those vertices in order to successfully slide the tokens on the u_i and $u_{i,j}$ vertices away from their neighbors w_i and $w_{i,j}$ that contain tokens. □

4 Vertex cover and dominating set discovery

In what follows, we summarize our results for Vertex Cover Discovery and Dominating Set Discovery.

Theorem 8 *The* Vertex Cover Discovery *problem can be solved in time $\mathcal{O}(2^k \cdot n^3)$ and in time $2^{\mathcal{O}(t \log k)} \cdot n^{\mathcal{O}(1)}$, where k denotes the number of tokens and t denotes the treewidth of the input graph.*

Theorem 9 *The* Vertex Cover Discovery *problem is fixed-parameter tractable when parameterized by b and restricted to structurally nowhere dense classes of graphs.*

Theorem 10 *The* Vertex Cover Discovery *problem is* NP*-complete on planar graphs of maximum degree four and* W[1]*-hard when parameterized by the budget b even on 2-degenerate bipartite graphs.*

Theorem 11 *The* Dominating Set Discovery *problem can be solved in time $2^{\mathcal{O}(t \log k)} \cdot n^{\mathcal{O}(1)}$, where t denotes the treewidth of the input graph.*

Theorem 12 *The* Dominating Set Discovery *problem is fixed-parameter tractable for parameter k and restricted to semi-ladder-free graphs and when parameterized by b and restricted to structurally nowhere dense classes of graphs.*

Theorem 13 *The* Dominating Set Discovery *problem is* NP*-complete on planar graphs of maximum degree five,* W[2]*-hard when parameterized by $k + b$ on the class of bipartite graphs, and* W[1]*-hard when parameterized by the budget b even on the class of 2-degenerate graphs.*

5 Coloring discovery

A *k-coloring* of a graph G is a mapping $\varphi\colon V(G) \to [k]$. A k-coloring is said to be *proper* if whenever $\{u, v\} \in E(G)$ then $\varphi(u) \neq \varphi(v)$. In the COLORING problem, we are given a graph G and an integer k and the goal is to decide whether G admits a proper k-coloring.

In the COLORING DISCOVERY (CD) problem, we are given a (non-proper) k-coloring and a budget $b \in \mathbb{N}$, and the task is to decide whether there is a transformation of the given coloring into a proper k-coloring by using at most b recoloring steps.

There are many possible definitions of adjacency relations between feasible (and infeasible) colorings. We consider the following reconfiguration steps proposed in the reconfiguration literature. In the *color flipping model*, in each step, we can change the color of any vertex to any color in the color set $[k] = \{1, \ldots, k\}$. In the *color swapping model*, a (u, v)-color-swap allows for the swap of colors between two arbitrary vertices $u \in V(G)$ and $v \in V(G)$, where $u \neq v$. Finally, in the *color sliding model*, a (u, v)-color-slide is a (u, v)-color swap along an edge of the graph. In other words, in color sliding we can only perform a (u, v)-color-swap if $\{u, v\} \in E(G)$.

The reconfiguration variant of the NP-complete COLORING problem [22] is a central problem in combinatorial reconfiguration and has been studied in the color flipping model [8, 13] and the color sliding model [7].

The COLORING DISCOVERY problem has already been studied in the color flipping model and the color swapping model under the names k-FIX [23] and k-SWAP [3], respectively. Garnero et al. [23] show that, for color flipping, the discovery variant is NP-complete, even for bipartite planar graphs. Moreover, they show that the problem is W[1]-hard when parameterized by b, even for bipartite graphs, whereas it is fixed-parameter tractable when parameterized by $k+b$. Interestingly, the latter is not true in the color swapping model, where the problem is W[1]-hard for any fixed $k \geq 3$ when parameterized by b [3]. It is also shown that both problems k-FIX and k-SWAP are W[1]-hard when parameterized by the treewidth of the input graph. Finally, it is shown that the k-FIX problem, i.e., discovery in the color flipping model, is polynomial-time solvable for $k \leq 2$ colors but this question was left open for the color swapping and color sliding models [23] and we answer it below.

Theorem 14 *COLORING DISCOVERY with $k = 2$ is solvable in polynomial time in the color swapping and sliding models.*

Proof sketch. We may restrict ourselves to bipartite graphs. (A graph is 2-colorable if and only if it is bipartite, and a graph can be verified to be bipartite by a graph search in polynomial time.) For each connected component G_i of the bipartite input graph G we find a bipartition (L_i, R_i) of $V(G_i)$. Any proper coloring of G_i colors all vertices of L_i in color red and all vertices of R_i in color green, or vice versa.

Both the needed budget for recoloring as well as the number of vertices colored in red and green in a proper coloring might be different for the two choices. For a given choice, both the needed number of swaps and number of colors can be calculated straightforwardly. As a swap always swaps colors red and green, the overall needed budget is given by the number of vertices colored in red that have to be recolored to green. Dependent on the two possibilities, we denote the needed budget by b_1^i or b_2^i to recolor the connected component G_i properly. Additionally, for each choice we define the excess of a connected component G_i by the number of vertices that are initially colored in color red minus those that have to be colored in color red, i.e.,

$$e_1(G_i) = |\{v \in V(G_i) \mid \varphi(v) = \mathsf{red}\}| - |L_i| \text{ , and}$$
$$e_2(G_i) = |\{v \in V(G_i) \mid \varphi(v) = \mathsf{red}\}| - |R_i| \text{ .}$$

Thus, dependent on the two possible choices we either need a budget b_1^i and realize an excess $e_1(G_i)$, or a budget of b_2^i with excess $e_2(G_i)$. Overall, we can recolor the whole graph with the given budget if and only if there is a set of choices such that the overall excess is 0 and the needed budget is at most b. We finish the proof by presenting a dynamic program for the problem. □

Theorem 15 *COLORING DISCOVERY with $k \geq 3$ colors is NP-complete under all three models, even when restricted to planar bipartite graphs.*

Due to this hardness, we consider the parameterized complexity of the problem with respect to the parameters k, b, and treewidth; On restricted classes of graphs the COLORING DISCOVERY problem becomes tractable. For general graphs, the problem is intractable for the same parameters.

Theorem 16 *COLORING DISCOVERY is fixed-parameter tractable when parameterized by $k + b$ and restricted to structurally nowhere dense classes of graphs under all three models.*

Theorem 17 *COLORING DISCOVERY parameterized by $k + b$ and COLORING DISCOVERY parameterized by treewidth are W[1]-hard in the color sliding model.*

6 Conclusion and future work

We have proved many positive and hardness results concerning solution discovery variants of fundamental graph problems. While some problems resulting from our framework have already been considered in the literature (albeit under different names), we believe that viewing such problems from a unified perspective can lead to more global insights and, hopefully, more unifying results.

We hope to foster further research on discovery variants of other problems, in graphs and beyond, as well as other relevant model variations, e.g., different reconfiguration steps, which model other restrictions on the changes of a system state.

Our model captures static decision-making settings, where an arbitrary infeasible solution shall be turned into a feasible one using only few well-defined modification steps. In some dynamic settings, the starting state may not be completely arbitrary, but is due to predictable or controlled changes in the system. It would be interesting to explore a generalization of our model in which the initial state can be assumed to satisfy certain properties. Besides the practical relevance, this also increases the degrees of freedom in which we can analyze the problems and pushes towards multivariate analyses, where the changes in the graph can now also be part of the parameter.

Acknowledgements

This research was supported by the research cluster *Minds, Media, Machines* (MMM) at the University of Bremen.

References

[1] K. Aardal, S. P. M. van Hoesel, A. M. C. A. Koster, C. Mannino, and A. Sassano, 'Models and solution techniques for frequency assignment problems', *Annals of Operations Research*, **153**(1), 79–129, (2007).

[2] E. Aarts and J. K. Lenstra, *Local search in combinatorial optimization*, Chichester: Wiley, 1997.

[3] M. De Biasi and J. Lauri, 'On the complexity of restoring corrupted colorings', *Journal of Combinatorial Optimization*, **37**(4), 1150–1169, (2019).

[4] Y. Blum and J. S. Rosenschein, 'Multiagent graph coloring: Pareto efficiency, fairness and individual rationality', in *AAAI 2008*, pp. 24–29. AAAI Press, (2008).

[5] H.-J. Böckenhauer, J. Hromkovic, and D. Komm, 'Reoptimization of hard optimization problems', in *Handbook of Approximation Algorithms and Metaheuristics, Second Edition, Volume 1: Methologies and Traditional Applications*, 427–454, Chapman and Hall/CRC, (2018).

[6] É. Bonnet, N. Bousquet, P. Charbit, S. Thomassé, and R. Watrigant, 'Parameterized complexity of independent set in h-free graphs', *Algorithmica*, **82**(8), 2360–2394, (2020).

[7] É. Bonnet, T. Miltzow, and P. Rzazewski, 'Complexity of token swapping and its variants', *Algorithmica*, **80**(9), 2656–2682, (2018).

[8] P. Bonsma and L. Cereceda, 'Finding paths between graph colourings: Pspace-completeness and superpolynomial distances', *Theoretical Computer Science*, **410**(50), 5215–5226, (2009).

[9] N. Bousquet, A. E. Mouawad, N. Nishimura, and S. Siebertz, 'A survey on the parameterized complexity of the independent set and (connected) dominating set reconfiguration problems', *arXiv preprint arXiv:2204.10526*, (2022).

[10] C. Titus Brown, D. Moritz, M. P. O'Brien, F. Reidl, T. Reiter, and B. D. Sullivan, 'Exploring neighborhoods in large metagenome assembly graphs using spacegraphcats reveals hidden sequence diversity', *Genome Biology*, **21**, 1–16, (2020).

[11] S. Cai, W. Hou, J. Lin, and Y. Li, 'Improving local search for minimum weight vertex cover by dynamic strategies', in *IJCAI 2018*, pp. 1412–1418. ijcai.org, (2018).

[12] S. Cai, J. Lin, and C. Luo, 'Finding A small vertex cover in massive sparse graphs: Construct, local search, and preprocess', *Journal of Artificial Intelligence Research*, **59**, 463–494, (2017).

[13] L. Cereceda, J. van den Heuvel, and M. Johnson, 'Finding paths between 3-colorings', *Journal of Graph Theory*, **67**(1), 69–82, (2011).

[14] M. Cygan, F. V. Fomin, L. Kowalik, D. Lokshtanov, D. Marx, M. Pilipczuk, M. Pilipczuk, and S. Saurabh, *Parameterized Algorithms*, Springer, 2015.

[15] E. D. Demaine, M. L. Demaine, E. Fox-Epstein, D. A. Hoang, T. Ito, H. Ono, Y. Otachi, R. Uehara, and T. Yamada, 'Polynomial-time algorithm for sliding tokens on trees', in *ISAAC 2014*, pp. 389–400. Springer, (2014).

[16] E. D. Demaine, F. Reidl, P. Rossmanith, F. S. Villaamil, S. Sikdar, and B. D. Sullivan, 'Structural sparsity of complex networks: Bounded expansion in random models and real-world graphs', *Journal of Computer and System Sciences*, **105**, 199–241, (2019).

[17] R. G. Downey and M. R. Fellows, 'Fixed-parameter tractability and completeness II: On completeness for W[1]', *Theoretical Computer Science*, **141**(1-2), 109–131, (1995).

[18] R. G. Downey and M. R. Fellows, *Parameterized complexity*, Springer New York, 2012.

[19] J. Dreier, N. Mählmann, and S. Siebertz, 'First-order model checking on structurally sparse graph classes', *arXiv preprint arXiv:2302.03527*, (2023).

[20] J. Edmonds and R. M. Karp, 'Theoretical improvements in algorithmic efficiency for network flow problems', *Journal of the ACM*, **19**(2), 248–264, (1972).

[21] M. Farrell, T. D. Goodrich, N. Lemons, F. Reidl, F. S. Villaamil, and B. D. Sullivan, 'Hyperbolicity, degeneracy, and expansion of random intersection graphs', in *WAW 2015*, pp. 29–41. Springer, (2015).

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

[23] V. Garnero, K. Junosza-Szaniawski, M. Liedloff, P. Montealegre, and P. Rzazewski, 'Fixing improper colorings of graphs', *Theoretical Computer Science*, **711**, 66–78, (2018).

[24] W. K. Hale, 'Frequency assignment: Theory and applications', *Proceedings of the IEEE*, **68**, 1497–1514, (1980).

[25] S. Har-Peled and K. Quanrud, 'Approximation algorithms for polynomial-expansion and low-density graphs', *SIAM Journal on Computing*, **46**(6), 1712–1744, (2017).

[26] E. Hebrard and G. Katsirelos, 'Clause learning and new bounds for graph coloring', in *IJCAI 2019*, pp. 6166–6170. ijcai.org, (2019).

[27] M. Henzinger, 'The state of the art in dynamic graph algorithms', in *SOFSEM*, volume 10706 of *Lecture Notes in Computer Science*, pp. 40–44, (2018).

[28] P. Huang, M. Liu, P. Wang, W. Zhang, F. Ma, and J. Zhang, 'Solving the satisfiability problem of modal logic S5 guided by graph coloring', in *IJCAI 2019*, pp. 1093–1100. ijcai.org, (2019).

[29] T. Ito, E. D. Demaine, N. J. A. Harvey, C. H. Papadimitriou, M. Sideri, R. Uehara, and Y. Uno, 'On the complexity of reconfiguration problems', *Theoretical Computer Science*, **412**(12-14), 1054–1065, (2011).

[30] M. Kaminski, P. Medvedev, and M. Milanic, 'Complexity of independent set reconfigurability problems', *Theoretical Computer Science*, **439**, 9–15, (2012).

[31] R. M. Karp, 'Reducibility among combinatorial problems', in *Complexity of Computer Computations*, eds., Raymond E. Miller, James W. Thatcher, and Jean D. Bohlinger, 85–103, Springer, (1972).

[32] V. Kavalci, A. Ural, and O. Dagdeviren, 'Distributed vertex cover algorithms for wireless sensor networks', *International Journal of Computer Networks & Communications*, **6**, 95–110, (2014).

[33] H. W. Kuhn, 'The Hungarian method for the assignment problem', *Naval Research Logistics Quarterly*, **2**(1-2), 83–97, (1955).

[34] D. Lokshtanov, F. Panolan, S. Saurabh, R. Sharma, and M. Zehavi, 'Covering small independent sets and separators with applications to parameterized algorithms', *ACM Transactions on Algorithms*, **16**(3), 32:1–32:31, (2020).

[35] C. Luo, H. H. Hoos, S. Cai, Q. Lin, H. Zhang, and D. Zhang, 'Local search with efficient automatic configuration for minimum vertex cover', in *IJCAI 2019*, pp. 1297–1304. ijcai.org, (2019).

[36] B. Mohar, 'Face covers and the genus problem for apex graphs', *Journal of Combinatorial Theory, Series B*, **82**(1), 102–117, (2001).

[37] N. Nishimura, 'Introduction to reconfiguration', *Algorithms*, **11**(4), 52, (2018).

[38] N. Robertson and P. D. Seymour, 'Graph minors. v. excluding a planar graph', *Journal of Combinatorial Theory, Series B*, **41**(1), 92–114, (1986).

[39] Y. Shen, Y. Sun, X. Li, A. S. Eberhard, and A. T. Ernst, 'Enhancing column generation by a machine-learning-based pricing heuristic for graph coloring', in *AAAI 2022*, pp. 9926–9934. AAAI Press, (2022).

[40] J. van den Heuvel, 'The complexity of change.', *Surveys in Combinatorics*, **409**(2013), 127–160, (2013).

[41] Y. Wang, S. Cai, S. Pan, X. Li, and M. Yin, 'Reduction and local search for weighted graph coloring problem', in *AAAI 2020*, pp. 2433–2441. AAAI Press, (2020).

[42] M. Wrochna, 'Reconfiguration in bounded bandwidth and tree-depth', *Journal of Computer and System Sciences*, **93**, 1–10, (2018).

[43] X. Xie and X. Qin, 'Dynamic feature selection algorithm based on minimum vertex cover of hypergraph', in *PAKDD 2018*, pp. 40–51. Springer, (2018).

[44] D. Zuckerman, 'Linear degree extractors and the inapproximability of max clique and chromatic number', in *STOC 2006*, pp. 681–690. ACM, (2006).