UNIVERSITÉ LYON 1
École doctorale InfoMaths (ED 512)

HABILITATION À DIRIGER DES RECHERCHES

Présentée par
Nicolas Bousquet

préparée au laboratoire LIRIS, soutenance prévue le 03/07/2024.

A Journey on Configuration Graphs

Coloring and Independent Set Reconfiguration

Composition du jury :

Rapporteurs:	Maria Chudnovsky	Professeure, Princeton University
	Jean Cardinal	Professeur, Université Libre de Bruxelles
	Mathieu Liedloff	Professeur, Université d'Orléans, LIFO
Examintateurs:	Marie Albenque	Directrice de Recherche CNRS, IRIF
	Laurent Beaudou	Maître de conférences HDR, Université Clermont-Auvergne, LIMOS
	Eric Duchêne	Professeur, Université Claude Bernard Lyon 1, LIRIS
	Daniel Marx	Professeur, CISPA Helmholtz Center
	Marie France Sagot	Directrice de Recherche INRIA, LBBE

Chapter 1

Introduction

Warm-up

As a researcher, we have only twice a total freedom on what to write: for the PhD thesis and the habilitation. It is also an opportunity to make a short break in our crazy life of researchers and ask ourselves: \why did I choose to do that ? Are we happy with it ? What would I have done di erently ? What did I like the most in what I have done ?

In this manuscript, I have decided to focus on recon guration problems and even more speci cally on graph recoloring and recon guration of independent sets. I have studied many other types of problems since I defended my PhD thesis: distributed algorithms and graph certi cation, parameterized algorithms, structural graph theory, economic game theory and many other types of recon guration problems to name a few. But for a matter of time devoted to the writing of this manuscript (and of coherence) I have decided to focus more speci cally on these two problems. I will nevertheless brie y present some other recon guration problems in a nal chapter of this manuscript.

What to expect from this manuscript?

All the habilitations are di erent. Let me explain what this one is not:

- It is not a \stapler habilitation". All the material included in this manuscript is completely new[1] except for a few pages corresponding to an extension of a survey paper [48] we wrote together with Amer Mouawad, Naomi Nishimura and Sebastian Siebertz. (It corresponds to a part of Section 4.5 and Section 5.2.2).
- As I said, it is not an overview of all my research. I decided to exclusively focus on one part of my research on which I spent an important

[1] Even if, I must confess, I sometimes a copy-pasted a few lines of some introductions of papers of my own.

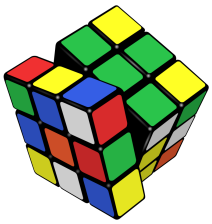

Figure 1.1: A Rubik's cube. (source Wikipedia)

part of my research time in these last 10 years.

- The manuscript does not include any detailed full proof of any of my results. According to me, ooding reviewers with technical details is not necessarily the best way to give a nice overview of my area of research.
- This manuscript is not a non-technical document either. I have decided to not fully include any (complicated) proof but have instead prefered giving sketchs of the proofs of most of the results covered in this manuscript (some of them are mine, some are not). Hopefully, the reader interested in graph recon guration can extract from this thesis some information on the classical proof techniques and ideas used in this area. But that also permits me to give the scienti c context of all my results and explain which gain it gives to the community.
- I assumed that the reader is familiar with basic notions of graph theory. More involved notions will be de ned either in the Preliminary chapter or when used in the manuscript. A non-specialist of structural or algorithmic graph theory may nd some parts a bit technical but I tried to give motivations that go far beyond graph theory as much as I could.

All these choices are personal and have a ected the structure and the content of this manuscript. I hope that you, as a reader, will like it and could take home some information from it.

1.1 Examples of reconfiguration problems

Before formally de ning what is a recon guration problem in Section 1.2, let us rst give some examples that illustrate the variety of these problems.

Puzzles.

When I was young, my brother used to play with Rubik's cube (see Figure 1.1). In this game (called 1-player game or puzzle in the rest of the

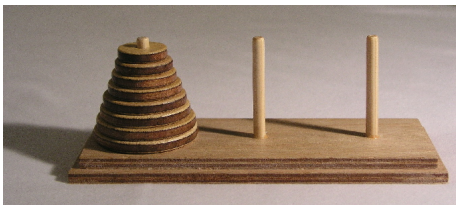

Figure 1.2: Hanoi tower. (source Wikipedia)

manuscript to avoid confusion with all the other types of games studied in computer science like economic game theory, combinatorial games, cops and robber games, positional games for logic...etc...), we have a cube with 9 small colored squares in each face. In total there are 6 colors, appearing 9 exactly times. The goal consists in transforming a con guration of the cube in the con guration where each face has exactly 9 squares of the same color. The hardness of this puzzle comes from its combinatorics: despite its reasonable size at rst glance, there are $43,252,003,274,489,860,000$ possible con gurations in a Rubik's cube ! To be honest, I never succeeded in solving a Rubik's cube. My brother used to solve it e ciently while I never succeeded to nish it[2].

I was not much better when I played the 15-puzzle (or \Taquin" in french). In this game, the board is a 4 4-grid consisting of 15 parts of a picture and an empty slot. The goal consists in moving the empty slot[3] in order to reconstruct the picture. The rules of many other famous games consist in trying to reach some desirable position starting from an arbitrary initial con guration: Rush hour (where a car wants to escape a tra c jam), Cache-Noisettes (where squirrels want to hide nuts).

One can wonder why my brother succeeded in solving the Rubik's cube while I didn't. The answer is simple, he learned some generic methods that permit to approach the target solution little by little. In a more mathematical language: he learned and applied e ciently an algorithm that allowed him to reduce the symmetric di erence between the current solution and the target solution. What is di cult with these methods is that, in order to approach the target solution (and decrease the di erence between the current solution and the target solution), you might have to increase (temporarily) the symmetric di erence. In other words, you might have to temporarily \destroy" your partial solution to build a better one. I learned, many years later, that it is the main reason why recon guration problems are hard (but also exciting).

Algorithms for Rubik's cube are not that simple to describe. It is easier

[2] But I usually succeeded to nish one face, so, I turned it in the right direction so that I could pretend to have succeeded.

[3] That is to permute in the representation the empty slot with a piece that is adjacent to it.

for the so-called Hanoi tower puzzle. In the Hanoi tower puzzle, we have three rods and a number of disks of pairwise distinct diameters (see Figure 1.2). At each step, we can slide exactly one disk from a rod to another one as long as we are not putting a disk of large diameter on a disk of small diameter. Usually we are starting from a position where all the disks are on the first rod and we want to move them on the last one. As for all the other puzzles mentioned in this manuscript, its number of configurations is exponential but there is a simple algorithm to solve it. Imagine that we have n disks. Then we simply have to pretend that the larger disk does not exist and move the $n - 1$ other disks on the second rod (which we can do by induction). We then move the larger disk to the last rod and finally move the $n - 1$ disks to the last rod.

Note that this algorithm provides a transformation whose length is exponential in the number of disks. Indeed, the length $\ell(n)$ of the transformation for n disks is $2\ell(n - 1) + 1$. So the length of an optimal transformation is not always polynomial and then reconfiguration problems do not naturally belong to NP. We will see that they \typically" belong to PSPACE which is assumed to be a class that strictly contains NP. In other words, reconfiguration problems are presumably much harder than \classical" optimization problems. A second consequence is that, there does not necesarily exist a polynomial (in terms of the size of the instance) transformation from an initial to a target position.

Finally note that induction based proofs are standard in reconfiguration and we will see many of them in this manuscript. They are unfortunately often not optimal in general[4]. One of the goals of this manuscript is to describe and survey algorithmic techniques developed for reconfiguration problems.

Pancake flipping and sorting by reversals.

When I was young, I clearly preferred eating my mother's crepes (which are thin pancakes from French Brittany) rather than solving Rubik's cubes. Imagine that we have a pile of pancakes on a plate. Each of the pancakes have a face that is more toasted than the other. As any reasonable person, we would like to present the pile of pancakes to guests so that all the pancakes have their toasted part below. Unfortunately, all the other plates are already used or are in the dishwasher. So we have to flip pancakes, little by little, using a spatula, as follows: we can put the spatula below a pancake and flip all the pancakes above it upside-down. More mathematically, we can select an integer i and return all the pancakes between the positions 1 and i in such a way, after the flip, the j-th crepe is at position $i - j$ and it has been flipped. One can wonder: is it always possible to present all the pancakes

[4] It is optimal for Hanoi tower though.

on their nicest side? If yes, how many steps are needed ? The answer to the first question is positive. While it is possible to find some upper bounds for the second question, the existence of a shortest transformation is not always simple to find. Depending on models (there are slight variations on the problem), deciding the existence of a transformation of length k is either in P or NP-complete[5].

This problem, which might seem artificial except if you like pancakes, is heavily related to the so-called sorting by reversals problem which is of importance in bio-informatics. In the problem we have two words of length n on an alphabet (typically 4 letters, A, T, C, G, corresponding to nucleotides of the DNA) with the same number of occurrences of the different letters. The goal is to transform the first into the second with the following operation: at each step, we can select a prefix P of the word and reverse this word. That is, if the word is PQ then we can transform it into $\overline{P}Q$ in one step. (This operation that looks strange at first glance just means that a portion of DNA glued the other part in the wrong direction which sometimes happens in practice).

Computing this distance permits to evaluate the distance between genes. Indeed, given two genes, one can wonder how many modifications have to be performed in order to transform the first into the second. The less modifications are needed, the closest the species are. One of the possible modifications during cell divisions is the one described above (even if, in reality, the set of possible modifications of the DNA is indeed more complex).

Many problems in bio-informatics consists in trying to find a shortest flipping sequence between combinatorial objects and sorting by reversal is only one of these numerous examples.

Motion of robots and warehouse's man problem.

Many problems in robotics can also be seen as reconfiguration problems. In warehouses, there are more and more robots and the goal of these robots is to move in order to perform some tasks. In other words, robots have an initial position and want to reach a target position. When the factory is small and the number of robots big, deciding the existence of such transformations is complicated. And even when solutions exist, robots might have to move in the wrong direction for some time in order to free some space for others (like in Rush hour). Hearn and Demaine proved that deciding the existence of a transformation between two configurations is PSPACE-complete even when robots are of bounded size. The complexity study of this problem, called the warehouse's man problem, can be considered as the genesis of research in algorithmic aspects of combinatorial reconfiguration.

[5]Note that the problem here belongs to NP. For many reconfiguration problems in biology and discrete geometry, proving the existence of a polynomial transformation is often simple, finding one of minimum length is the hard and exciting task.

Note that this problem is slightly di erent from the problems already described. Indeed, in all the previous problems above, the question was: how can I nd a transformation to my target position? How many steps do I need? So in particular, we implicitly assume that the target con guration can be reached[6]. For many recon guration problems considered in this manuscript, the existence of a transformation is not guaranteed and deciding it is often hard.

Flip distance between geometric objects.

Flipping between combinatorial objects in discrete geometry has been widely studied for decades. As one of the (many other) objects, one can mention Schnyder woods for triangulations of planar graphs which has been proven to have many applications (for the representation of planar graphs) and generalizations to larger surfaces.

Many other types of ips have been studied in the literature including ip distance between triangulations (where studies focus on the diameter of the con guration as long as trying to determine the length of a shortest sequence), between non-crossing perfect matchings or non-crossing spanning trees or quadrangulations.

1.2 Reconfiguration

In theory, solving a Rubik's cube is incredibly simple. It su ces to rotate the faces. The number of possible positions is bounded so, at some point, we will reach the target position if it is possible. The hardness indeed comes from the fact that the number of con gurations of the Rubik's cube is insane ! This is the rst curse of recon guration: the recon guration rules are usually simple, the instances are of small size, the problems we want to solve can be formulated easily but the underlying structure is huge.

1.2.1 Configuration graphs

Let be a problem and I be an instance of (one can typically imagine that we are given a Rubik's cube). Let us also assume that we are given a *recon guration rule* (that is a way to modify a solution into another, for Rubik's cube, it consists in a rotation of a face of the cube).

The *con guration graph* $G(I)$ is the (meta)-graph whose vertex set consists of all the solutions of I and there is an edge between two vertices S_1, S_2 if and only if it is possible to transform the rst solution into the second in one recon guration step. (For Rubik's cube it corresponds to a face rota-

[6] which is not necessarily the case for Rubik's cube for instance.

tion)[7]. The *diameter* of the con guration graph is the maximum distance between two vertices of the con guration graph (or + *1* if two vertices do not belong to the same connected component).

As we have already seen for the Rubik's cube, the main reason why recon guration problems are di cult does not come from the problem by themselves but from their size. Even if the initial objects are small, the con guration is rapidly extremely large. We have already seen that for the Rubik's cube, the number of con gurations is larger than several billions. If we consider k-independent sets of a graph, we end up with a graph with possibly $\binom{n}{k}$ vertices and the number of colorings of a graph can be exponential in n.

1.2.2 Standard question in the reconfiguration field

Depending on the motivation of the underlying recon guration problem, several types of problems have been studied in the literature:

- **Reachability.** Let I be an instance of a problem and R be a recon guration rule and S_1, S_2 be two solutions of I. The goal is to determine if there exists a transformation from S_1 to S_2 via a sequence of recon guration steps keeping a solution all along the transformation.

  If we see this problem from a con guration graph perspective, the question becomes: does there exist a path from S_1 to S_2 in the con guration graph? This problem can be easily solved in graphs of decent size using a simple BFS. However, since the size of con guration graphs might be exponential in the size of the instance, computing such a BFS might be too long. So we have to develop new techniques to nd paths without a complete exploration of the con guration graph.

- **Connectivity.** Let I be an instance of a problem and R be a recon guration rule. The goal is to determine if, for every pair of solutions S_1, S_2, there exists a transformation from S_1 to S_2 via a sequence of recon guration steps keeping a solution all along the transformation.

  From a con guration graph perspective, the question becomes: is the con guration graph $G(I)$ connected? Again, this question can be easily solved using a BFS when con guration graphs are of decent size. So the hardness of this problem again comes from the fact that the con guration may be very large.

  Determining the connectivity of the con guration graph is, for instance, interesting for random sampling. Indeed, one can try to sample solutions as follows: we start from a solution we know and apply to

[7] Note that, all along the manuscript, recon guration steps can be undone meaning that the con guration graph is an undirected graph.

it iteratively a random small modi cation until we sample a solution almost at random. In other words, we perform a random walk in the con guration graph starting from a prescribed initial solution. To be sure that all the solutions can be sampled, we need to be sure that we can reach all the solutions from the initial prescribed solution, in other words, when the con guration graph is connected.

More generally, the connectivity of the con guration graph is heavily related to the mixing time of the underlying Markov chain (the more connected the con guration graph is the faster the Markov chain converges to its stationnary distribution).

- **Diameter.** Let I be an instance of a problem and R be a reconguration rule which ensures that $G(I)$ is connected. The goal is to determine the diameter of $G(I)$. Again, using a classical approach (like computing all pairs of shortest paths), we can compute the diameter in polynomial time in $|G(I)|$. So again, the hardness of the problem is due to the size of the con guration graph. An important line of research consists in determining if the diameter is polynomial (or even linear) in $|I|$ and does not depend on the size of $G(I)$.

  Such questions naturally arise in bioinformatics (genomic distance, sorting by reversals, ip distance between phylogenetic trees...etc...), discrete geometry (ip distance between triangulations or trees) or in combinatorial optimization. For instance, the famous Hirsch conjecture (now refuted) stating that the diameter of a polytope with n faces in a d-dimensional Euclidean space has diameter at most $n \quad d$, can be rephrased as a recon guration problem.

- **Algorithmic aspects of reconfiguration.** The community working in recon guration is interested in both purely theoretical results (motivated by applications in random sampling) but also in more algorithmic results (motivated for instance for applications in puzzles or bio-informatics).

  Algorithmically, recon guration problems are usually much harder than their optimization counterparts. We will, for instance, see that while the existence of an independent set of size at least k is in NP, nding a transformation between two independent sets of size k is PSPACE-complete. Many recon guration problems considered in this manuscript are actually PSPACE-complete. Some recon guration problems however lie in simpler graph classes (pancake ippings, ips between triangulations...etc...). But these problems are usually simpler since one can usually easily show that a transformation exists and we simply want to determine a shortest transformation (whose length is often known to be polynomial with simple arguments). How-

ever these problems are hard and designing e cient approximation algorithms is very challenging.

What, according to me, makes recon guration problems algorithmically interesting is that most of the classical techniques developed in the last decades do not naturally generalize to recon guration problems. It is for instance, very complicated to design dynamic programming algorithms for recon guration problems which makes trees or bounded treewidth graphs already challenging and exciting graph classes to study (we will discuss further this point in Chapter 4).

1.2.3 Relations with other fields of research

Random sampling and Markov chains.

A Markov chain or Markov process is a stochastic model describing a sequence of possible events in which the probability of each event only depends on the current state and not on the previous ones. In other words, it is a memoryless process where the future does not depend on history but only on the current state. There are many ways to see a nite discrete Markov chain. When I imagine one, I usually imagine a *graph*. A graph G is a pair (V, E) consisting of a set of points V called *vertices* and where E consists of pairs of elements of V called *edges*.

A Markov chain with nitely many states simply is a directed graph $D = (V, E)$ (that is we put a direction on edges, i.e. pairs of elements of V are oriented), possibly with self loops. The arcs are given a non-negative weight corresponding to a probability distribution on the arcs. In other words, for all the vertices v of D, the sum of the weights of the arcs leaving v equals one. That gives, when we are in some con guration, a distribution of probabilities on what will be the next position.

Markov chains are widely studied in mathematics and have many applications, for instance for random sampling. When one wants to sample a solution at random of a problem one can do it naturally using a Markov chain. To do so, we de ne an adjacency relation between solutions and simply perform a random walk using this adjacency relation. In other words, we simply perform a random walk in the con guration graph of the solutions.

The two mains questions studied in the random sampling community are the following:

- Is the Markov chain ergodic ? Which can be restated in terms of graphs as: is the con guration graph connected [8]?
- What is the mixing time of the Markov chain? That is, how much time do we have to wait to be sure that a solution is sampled almost

[8] To be ergodic, we need slightly more than the connectivity of the con guration graph but since we usually consider lazy Markov chains the two notions are equivalent.

at random. This question is, for instance, related to the diameter and the connectivity of the con guration graph.

Enumeration problems.

In the last decades, a line of research has been devoted to enumerate at random or count the solutions of a problem (motivated for instance by problems for cheminformatics or database theory). Let us mention that some methods related with con guration graphs. Assume that one can prove that the configuration graph of the solutions of a problem are connected. Then, in order to generate all the solutions, one simply has to explore the con guration graph, using a BFS or a DFS.

Using a small trick[9], one can prove that, if the degree of the con guration graph is polynomial, one can enumerate all the solutions with a *polynomial delay* (that is there is a polynomial delay between the enumeration of two consecutive solutions). Unfortunately such enumeration algorithms have an important drawback: they might need an exponential space. Indeed, when one performs a BFS/DFS in a graph, we need a polynomial space in the size of the input graph. Since the con guration graph may have exponential size, this implies an exponential running time. Finding enumeration algorithms which only need a polynomial space is usually a challenging problem. One of the methods used to do so consists in nding an (implicit) ordering of the solutions which permits to know, when we explore an edge of the configuration graph, if the con guration we reach has already been explored or not.

Correction of databases and re-optimization.

Another related eld, slightly further from recon guration though, is re-optimization or correction of databases. In this setting, we are given a database with errors or an outdated solution of a problem. And the goal is to reach a correct database or an updated optimal solution in the minimum number of steps.

While these problems seem very close at rst glance, the techniques are quite di erent since, in these problems, we do not need to keep a solution all along the transformation but simply reach one of them. In other words, we can travel in a \supergraph" of the con guration graphs, making these problems easier (from a computational perspective). Note that in recent years, some researchers studied optimization recon guration problems, whose goal is to nd, given a solution S, the best (for some optimisation function) solution in the connected component of S in the con guration graph.

[9] The trick consists in outputting the solutions of even layers when we go down on the tree and vertices of odd layers when we go up.

1.3 Brief overview of the manuscript

Existing surveys on combinatorial reconfiguration and difference with them.

In this manuscript, we will survey results and proof techniques in graph recoloring and independent set recon guration. In the past 15 years, several works already summarize results in combinatorial recon guration. A rst survey, written by Jan van den Heuvel, surveys in particular (but not only), existing results on graph recoloring very nicely [131]. But most of the results I will mention in this manuscript are more recent that the ones covered in this survey, which proves the important vitality of this research eld in the last 15 years.

A second survey was written in 2017 by Naomi Nishimura [119] who proposed an excellent survey summarizing all the results on combinatorial recon guration at that time. This survey is the bible for whoever wants to know if a problem has already been studied in the past. The important width of this survey comes with a choice of not going too deep in proof technique descriptions. That is why I have decided to focus on a subset of recon guration problems to present in more detail techniques used in recon guration.

I wrote in 2022 [48] a survey with Naomi Nishimura, Amer Mouawad and Sebastian Siebertz which focuses on parameterized aspects of dominating set and independent set recon guration. This manuscript has the same avor but is more general (and then much longer).

Introductions to recon guration are proposed in many PhD thesis, including the ones of my former PhD students (I was very lucky to co-advise) Marc Heinrich [91], Alice Jo ard [100] and Valentin Bartier [3]. In my habilitation, I will cover a material of a wider range and describe proof techniques with more details. However I will sometimes refer to their PhD for some speci c problems.

Let us now quickly describe the content of the three main chapters of this manuscript.

1.3.1 Graph recoloring

The rst part of the manuscript will be devoted to graph recoloring. Let $G = (V, E)$ be a graph. A *(proper) k-coloring*[10] α of G is a function α from V to $\{1, \dots, k\}$ such that, for every pair of adjacent vertices u, v, $\alpha(u) \neq \alpha(v)$.

Two types of adjacencies between colorings have been studied: single vertex recolorings and Kempe-recolorings. In what follows, we will almost exclusively study the rst one. Two colorings α, β are *adjacent* if they di er

[10] Unless otherwise stated, all the colorings will be proper and we will omit the proper for brevity.

on exactly one vertex. The *con guration graph* $G(G, k)$ admits as vertices all the proper k-colorings with the adjacency described above.

In this chapter we will mainly cover three topics that received a considerable attention:

1. What is the evolution of $G(G, k)$ when k is a function of the degeneracy of the graph? This line of research, initiated by Cereceda during his PhD thesis, has attracted a lot of attention, not only in general but in restricted graph classes such as planar graphs.

2. What is the evolution of $G(G, k)$ when k is a function of the maximum degree of the graph? This question, motivated by problems from random sampling, has also been studied quite a lot.

3. Algorithmic aspects of graph recoloring. Deciding the reachability problem is known to be PSPACE-complete and deciding this problem on restricted graph classes has been widely studied.

We will also survey some results on distributed recoloring, a recent topic of research. I will explain the links between distributed recoloring and locality in recon guration, which is, according to me, an important research direction to prove linear upper bounds on the diameter of con guration graphs.

We will complete this chapter by having a quick look at two related problems: recon guration via Kempe changes and graph homomorphisms recon guration (a coloring being a particular case of graph homomorphism). And we will nally provide a last but nice application of graph recoloring due to Wrochna.

1.3.2 Independent Set Reconfiguration

The second main chapter of this manuscript will be devoted to the study of independent set recon guration. An *independent set* (seen as a set of *tokens*) is a subset of vertices which are pairwise non-adjacent. We will mainly focus on the di erences between the two models of independent set recon guration, namely Token Sliding and Token Jumping.

In the Token Jumping model, we can change a vertex of the independent set into any possible other vertex while in the Token Sliding model we can only move a token along an edge of the graph. We will in particular see that:

1. These two problems behave completely di erently even on restricted graph classes such as chordal graphs on which one problem is trivial while the other is PSPACE-complete (and designing polynomial time algorithms on restricted subclasses is highly non-trivial).

2. They also behave differently from a parameterized point of view. While for the Token Jumping model, the behavior of the Independent Set Reconfiguration (ISR) is quite well understood for sparse graph classes, we will see that it is not the case for Token Sliding (and we will explain the first existing results towards this direction).

3. We will finally mention some results related to the diameter of the configuration graph at the end of the chapter, a topic that -surprisingly- did not receive a lot of attention for ISR so far.

1.3.3 Other reconfiguration problems

We will complete this manuscript with a short chapter on two other reconfiguration problems: dominating set reconfiguration and matroid reconfiguration. I mention them for two different reasons: the first one because it has also been widely studied and I would like to give the reader a rapid overview of what is known as well as interesting questions. The second, matroids, are deeply related to reconfiguration problems and many questions related to generalizations of matroids reconfiguration are widely open and, I think, very exciting.

We will conclude this manuscript with a couple of research directions for the future. (Note moreover that many open problems and questions are included all along the manuscript and will not be repeated in the conclusion).

Chapter 2

Preliminaries

2.1 Introduction to Graph Theory

2.1.1 Graphs

We assume that each graph G is nite, simple, and undirected. We let $V(G)$ and $E(G)$ denote the vertex set and edge set of G, respectively. The *open neighborhood* of a vertex v is denoted by $N_G(v) = fu \; j \; (u,v) \; 2 \; E(G)g$ and the *closed neighborhood* by $N_G[v] = N_G(v) \; [\; fvg$. For a set of vertices $Q \quad V(G)$, we de ne $N_G(Q) = \bigcup_{v2Q} N_G(v) \; n \; Q$ and $N_G[Q] = N_G(Q) \; [\; Q$. The subgraph of G *induced* by Q is denoted by $G[Q]$, where $G[Q]$ has vertex set Q and edge set $f(u,v) \; 2 \; E(G) \; j \; u,v \; 2 \; Qg$. We let $G \quad Q = G[V(G) \; n \; Q]$.

A *walk* of length ℓ from v_0 to v_ℓ in G is a vertex sequence $v_0, \ldots, v_\ell$, such that for all $i \; 2 \; f0, \ldots, \ell \quad 1g$, $(v_i, v_{i+1}) \; 2 \; E(G)$. It is a *path* if all vertices are distinct. It is a *cycle* if $\ell \quad 3$, $v_0 = v_\ell$, and $v_0, \ldots, v_{\ell \quad 1}$ is a path. For a pair of vertices u and v in $V(G)$, by $\text{dist}_G(u,v)$ we denote the *distance* between u and v in G (which is the length of a shortest path between u and v measured in number of edges and set to 1 if u and v belong to di erent connected components). The *girth* of G, girth(G), is the length of a shortest cycle contained in G. The girth of an acyclic graph (i.e. a forest) is de ned to be in nite.

A class C of graphs is *monotone* if it is closed under taking subgraphs, i.e., if $G \; 2 \; C$ and $H \quad G$, then also $H \; 2 \; C$. It is *hereditary* if it is closed under taking induced subgraphs. We do not distinguish between isomorphic graphs.

For $A, B \quad V(G)$ we write $E(A,B)$ for the set of edges with one endpoint in A and one endpoint in B. For disjoint subsets A, B of $V(G)$, we write $G[A,B]$ for the subgraph of G *semi-induced* by A and B, that is, the subgraph with vertex set $A \; [\; B$ and all the edges with one endpoint in A and one endpoint in B. A bipartite graph H is a semi-induced subgraph of G if $H = G[A,B]$ for some disjoint subsets A and B of $V(G)$. Let H be a bipartite graph with vertices $a_1, \ldots, a_n$ (in one part) and $b_1, \ldots, b_n$ (in the

other part). Then H is:

- a *biclique* $K_{n,n}$ of size n if we have $(a_i, b_j) \in E(H)$ for all $i, j \in [n]$,
- a *co-matching* of size n if we have $(a_i, b_j) \in E(H) \iff i \neq j$ for all $i, j \in [n]$,
- a *ladder* of size n if $\{a_i, b_j\} \in E(H) \iff i \in \{j, j+1\}$ for all $i, j \in [n]$, and
- a *semi-ladder* of size n if $\{a_i, b_j\} \in E(H)$ for all $i, j \in [n]$ with $i > j$, and $\{a_i, b_i\} \notin E(H)$ for all $i \in [n]$.

The *biclique index*, *co-matching index*, *ladder index*, or *semi-ladder index* of a graph G is the largest n such that G contains a biclique, co-matching, ladder, or semi-ladder of size n as a semi-induced subgraph, respectively. A clique of size n, denote by K_n, is a set of n pairwise adjacent vertices. A class of graphs is *biclique free*, *co-matching free*, *ladder free*, or *semi-ladder free* if the supremum of the biclique indices, co-matching indices, ladder indices, or semi-ladder indices of its members is finite, respectively.

A *hypergraph* is a pair (V, E) where V is a set of vertices and E is a set of subsets of V called hyperedges. Let $H = (V, E)$ be a hypergraph. A set X of vertices of H is *shattered* if for every subset Y of X there exists a hyperedge e such that $e \setminus X = Y$. An intersection between X and a hyperedge e of E is called a *trace (on X)*. Equivalently, a set X is shattered if all its $2^{|X|}$ traces exist (in H). The *Vapnik-Chervonenkis dimension* (*VC-dimension*, for short) of a hypergraph is the maximum size of a shattered set.

Let $G = (V, E)$ be a graph. The *closed neighborhood hypergraph of G* is the hypergraph with vertex set $V(G)$ and where $X \subseteq V(G)$ is a hyperedge if and only if $X = N[v]$ for some vertex $v \in V$. The VC-dimension of a graph is the VC-dimension of its closed neighborhood hypergraph. The VC-dimension of a class of graphs $\mathcal{C}$ is the maximum VC-dimension of a graph of $\mathcal{C}$.

For a graph G and a set $X \subseteq V(G)$, we often partition the vertices of $V(G) \setminus X$ into what we call *X-projection classes* (we write *projection classes* when X is clear from context). That is, all vertices $u, v \in V(G) \setminus X$ of one class satisfy $N(u) \setminus X = N(v) \setminus X$. For $Y \subseteq X$, we let C_Y denote the set of vertices of $V(G) \setminus X$ whose neighborhood in X is exactly Y.

2.1.2 Width of graphs

We will define most of the classical graph classes on the fly in the manuscript when we see them for the first time. We however want to discuss some width graph parameters. The most classical width on graph is indeed treewidth.

Let $G = (V, E)$ be a graph. A *tree decomposition T of G* is a tree together with an assignment function such that:

- We associate to every vertex u of G a non-empty subtree T_u of T and,

- For every edge $e = (u, v)$ the subtrees T_u and T_v intersect.

One can de ne equivalently a tree decomposition as a pair (T, B) such that:

- For every node[1] u of T, $B(u)$ is a subset of vertices of G called the *bag* of u,
- For every edge vw *2* E, there exists at least one node u such that both v and w are in $B(u)$ and,
- For every vertex v *2* V, the set of nodes T_v containing v in their bags is a subtree of T.

The *width* of a tree-decomposition is the maximum size of a bag minus one. The *treewidth* of a graph G, denoted by $\mathbf{tw}(G)$ is the minimum width of a tree-decomposition of G. One can easily remark that forests are exactly graphs of treewidth at most 1. Graphs of treewidth at most 2 are series-parallel graphs, and contain important classes such as outerplanar graphs. The *pathwidth of* G, denoted by $pw(G)$, is de ned similarly except that the tree T has to be a path. We call such a tree-decomposition a *path decomposition*. One can easily remark that the pathwidth of a graph G is at least the treewidth of G and that the two parameters are not necessarily comparable. Indeed, a complete binary tree for instance has treewidth one but arbitrarily large pathwidth. A graph has *bandwidth* at most k if there exists an ordering $v_1, \ldots, v_n$ of G such that, for every i, all the neighbors of v_i in G are included in $\{v_{i-k}, \ldots, v_{i+k}\}$. It is an easy exercise to prove that a graph of bounded bandwidth has bounded pathwidth.

When we observe leaves of the tree T, one can remark that either their bag is included in the bag of their parent (and the node can be removed while still giving a tree decomposition of T) or there exists a vertex that only belongs to that bag. This ensures that every graph of treewidth at most r has a vertex of degree at most r (actually we can even prove that the graph is r-degenerate).

A *chordal graph* is a graph G that admits a tree-decomposition such that all the bags induce cliques. In other words, it is the intersection graph of subtrees of a tree. This tree decomposition is called a *clique tree* of G. Note that the above remark together with the fact that bags induce cliques ensures that all the chordal graphs contain *simplicial vertices*, that is, vertices whose neighborhood is a clique. Indeed, one can easily prove that all the vertices that only appear in bags of leaves of the clique tree are simplicial.

It is well known that chordal graphs are perfect. A graph is *perfect* if all its induced subgraphs satisfy $\chi(G) = \omega(G)$. Equivalently, it has been proven that a graph is perfect if it does not contain any induced odd cycle of length at least 5 nor its complement. There is a natural relation between chordal graphs and graphs of bounded treewidth which is the following:

[1] To avoid confusion, we will call nodes the vertices of the tree.

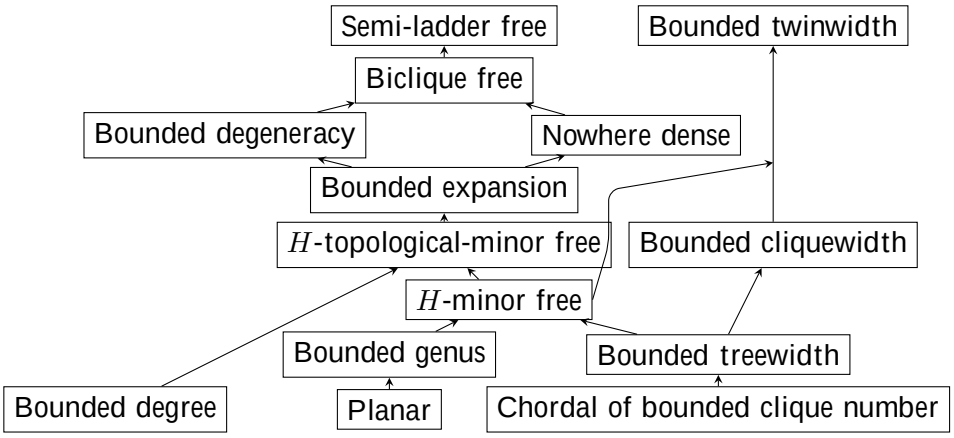

Figure 2.1: Inclusion diagram of several graph classes mentioned all along the manuscript. Arrows indicate inclusion.

Remark 1. *For every graph G of treewidth t, there exists a chordal graph H which is a supergraph of G and which has clique number at most t + 1.*

The same relation holds between bounded pathwidth graphs and interval graphs. In some sense, chordal graphs can be seen as completion of bounded treewidth graphs. This completion is often easier to manipulate since the structure of a bag in a chordal graph is a clique.

Hierarchy of graph classes Let us simply give the hierarchy of graph classes of Figure 2.1. We will not de ne them all here but they will be useful in the rest of the manuscript. The reader interested in the di erence between these graph classes can refer to this gure while reading the manuscript.

2.2 Markov chains

Let us give some basic de nitions on nite Markov chains. The reader interested in Markov chains can refer to the numerous books or lecture notes dealing with the topic online. Some de nitions will not be completely formal or not as general as what they can be. It is done on purpose in order to (try to) nd the best balance between intuition and full generality. Let us rst de ne the Markov chains before explaining how they can be used to sample solutions of a problem.

Markov chains. The *con guration graph* (or *transition matrix*) of a nite discrete Markov chain is a directed graph $G = (V, E)$ with possible loops together with a weight function ω on the arcs such that, for every vertex v, $\sum_{w \text{ s.t. } vw \in E} \omega(vw) = 1$. In other words, for every vertex, the arcs leaving v

de ne a probability distribution over the set of vertices of the graph. The vertices of G are called the *con gurations*[2] of G.

A Markov chain[3] is *lazy* if $\omega(vv) > 0$ for some v. A Markov chain is said to be *irreducible* if the graph G is strongly connected. A Markov chain is said to be *aperiodic* if the least common divisor of all the closed walks of the graph G is 1. From a graph point of view it means for instance that the graph is not bipartite (graphs for which all the cycles have length 0 modulo 2) but more generally it does not belong to the class where all the cycles have length 0 mod k. Note that if the chain is lazy then it is indeed aperiodic since, if we are given a cycle of length r we can easily construct closed walks of length of length r + ℓ for any integer ℓ 0. A chain that is both irreducible and aperiodic is said to be *ergodic*.

A Markov chain with initial state[4] v 2 V is a sequence of vertices $v_1, \ldots, v_r$ on G such that, at every step, v_{i+1} is chosen with probability $\omega(v_i v_{i+1})$. In other words, a Markov chain is a random walk in the con- guration graph. More formally, a Markov chain with initial vertex v is a stochastic process (i.e. a sequence of random variables $x_1, \ldots, x_r$) such that:

- $x_1 = v$
- P(x_{i+1} = w_{i+1}jx_i = $w_i, x_{i\ 1}$ = $w_{i\ 1}, \ldots, x_1$ = v_1) = P(x_{i+1} = w_{i+1}jx_i = w_i) = $\omega(w_i w_{i+1})$

The rst point of the de nition ensures that we are starting from v while the second point ensures that the next step of the Markov chain only depends on the last vertex of the path already visited in the con guration graph.

In the rest of the manuscript, our con guration graph will have the additional property that for every pair of vertices u, v, $\omega(u\ !\ v) = \omega(v\ !\ u)$. Such con guration graphs are called *reversible*. In particular, if the con guration graph is reversible, the graph G actually is an undirected graph.

Let M be a Markov chain with a set of states V. Let ν be a distribution on V. We say that ν is *stationary* if, starting with probability distribution ν on V we end up on ν after applying one step of the Markov chain. In other words, $M\nu = \nu$. A stationary distribution is then a xed point of the Markov chain. One of the main results in Markov chain theory is the following:

Theorem 2. *Every ergodic Markov chain converges to its unique stationary distribution.*

[2] It is often called the set of states for general Markov chains.

[3] It is actually the con guration graph of the Markov chain, but by abuse of notation we say that a Markov chain has some property if its con guration graph has it.

[4] We can more generally start from an arbitrary distribution over the con gurations, but we will only start from a single con guration in the rest of the manuscript.

The *uniform distribution* is the distribution ν where $\nu(v) = \nu(w)$ for every pair $v, w \in V$ with $v \neq w$. In other words, $\nu(v) = 1/|V|$ for every v. One can moreover easily remark that the following holds:

Remark 3. *If the configuration graph is reversible then the uniform distribution is a stationary distribution.*

Note that if we have a full access to the graph G then it is indeed simple to check that all the properties of the Markov chain we previously mentioned indeed holds. We can also find the stationary distribution using basic linear algebra methods since, if we see G as a transition matrix, we simply have to solve the equation $G\mathbf{x} = G$, where $\mathbf{x}$ is a vector of length $|V|$. Unfortunately in many practical cases, we cannot directly perform this computation for two reasons: (i) either we only have stochastic information about the transition matrix or, (ii) the graph G is actually too large to be computed and stored. The cases we will need to cover actually lie in the second case since the graph G will be actually exponential in the size of our initial instance in most of the cases. In order to explain why, let us illustrate the random sampling algorithm using Markov chains on graph colorings.

Mixing time We will not formally define the mixing time of a Markov chain to avoid cumbersome notations. However, informally speaking, the mixing time of the Markov chain is the number of steps needed to ensure that, if we start from an arbitrary distribution, we are close to the stationary distribution. In other words, the mixing time gives the speed of convergence of the chain towards the stationary distribution. Usually, the faster, the better since we want to sample a solution (almost) at random as fast as possible.

2.3 Parameterized complexity

We assume here that classical complexity classes (P, NP, PSPACE, L and NL) are known and do not define them. The interested reader can refer to any book or advanced complexity course to get the definition of these classes (and more).

A problem is *fixed-parameter tractable*, FPT for short, on a class C of graphs with respect to a parameter k, if there is an algorithm deciding whether a given graph $G \in C$ admits a solution of size k in time $f(k)\,|V(G)|^c$, for a computable function f and constant c. A *kernelization algorithm* is a polynomial-time algorithm that reduces an input instance to an equivalent instance of size bounded in the parameter only (independent of the input graph size), known as a *kernel*; we will say that two instances are *equivalent* if they are both yes-instances or both no-instances. Every fixed-parameter tractable problem admits a kernel, however, possibly of exponential or worse size. For efficient algorithms it is therefore most desirable to

obtain polynomial, or even linear, kernels. The *W-hierarchy* is a collection of parameterized complexity classes FPT W[1] ... W[t]. The conjecture FPT (W[1] can be seen as the analogue of the conjecture that P (NP. Therefore, showing hardness in the parameterized setting is usually accomplished by establishing an FPT-reduction from a W[i]-hard problem for some i. We refer to the textbook [61] for extensive background on parameterized complexity.

Hence, for a recon guration problem, a kernelization algorithm is a polynomial-time algorithm that transforms every input instance (G, k, S_s, S_t) into an instance (G^0, k^0, S_s^0, S_t^0) with $jG^0j + k^0$ $f(k)$ for some function f and such that (G, k, S_s, S_t) is a yes-instance if and only if (G^0, k^0, S_s^0, S_t^0) is a yes-instance. On the reduced instance (G^0, k^0, S_s^0, S_t^0) one can then run a brute-force algorithm to decide whether the initial instance was a yes-instance.

2.4 Distributed algorithms

Let us rst introduce the LOCAL model of distributed computing, which is one of the most studied models in distributed algorithms. This model has been introduced by Linial [110].

Contrary to centralized algorithms where a central entity performs all the computations, in the distributed setting, the nodes perform the computation but only have access to partial information and must make a choice with that information. We consider a graph G (often called network in that setting) whose vertices have unbounded computational power, and whose edges are communication links between the corresponding vertices. We are given a combinatorial problem that we need to solve in the graph G. In the case of deterministic algorithms, each vertex of G starts with an arbitrary identi er (an integer between 1 and n^c, for some constant c 1). The vertices then exchange messages (of unbounded size) with their neighbors in synchronous rounds, and after a xed number of rounds (the *round complexity of the algorithm*), each vertex outputs its local \part" of the global solution of the problem. This could for instance be the color of the vertex in a proper k-coloring. In Locally Checkable Labeling (LCL) problems, this output has to be of constant size, and should be checkable locally, in the sense that the solution is correct globally if and only if it is correct in all neighborhoods of some (constant) radius. LCL problems include problems like k-coloring (with constant k), or maximal independent set, but not maximum independent set (for instance), and are central in the eld of distributed algorithms.

It turns out that with the assumption that messages have unbounded size, vertices can just send to their neighbors at each round all the information that they have received so far, and in t rounds each vertex v \knows" its neighborhood $B_t(v)$ at distance t (the set of all vertices at distance at most t from v). More speci cally v knows the labeled subgraph of G induced

by $B_t(v)$ (where the labels are the identifiers of the vertices), and nothing more, and the output of v is based solely on this information.

Usually the goal is to minimize the round complexity. Since in t rounds each vertex sees its neighborhood at distance t, after a number rounds equal to the diameter of G, each vertex sees the entire graph. Since each vertex has unbounded computational power, a distinguished vertex (the vertex with the smallest identifier, say) can compute an optimal solution of the problem and communicate this solution to all the vertices of the graph. This shows that any problem can be solved in a number of rounds equal to the diameter of the graph, which is at most n when G is connected. The goal is to obtain algorithms that are significantly more efficient, i.e. of round complexity $O(\log n)$, or even $O(\log^* n)$, where $\log^* n$ is the number of times we have to iterate the logarithm, starting with n, to reach a value in $(0, 1]$.

Chapter 3

Graph Recoloring

The goal of this chapter is to overview (most of) the existing results on graph recoloring proven in the past twenty years and highlight some of my results. My goal is to try to give an idea of what has been done and how and which challenges still remain. I will sketch the proof techniques of most of the results in order to give to the reader an idea of the methods that are usually used to tackle the problems in the community. This will allow me to explain my contributions with respect to the state of the art at the time of the publication.

3.1 Recoloring and palette size

In this chapter, we will mainly focus on graph recoloring via single vertex recolorings. More general recon guration steps have also been studied. We will brie y survey recolorings with Kempe changes in Section 3.8.1 or distributed recoloring in Section 3.5. However, in the remainder of this chapter, unless otherwise speci ed, we will assume that one vertex is recolored at each step.

For completeness, let us recall some de nitions given in Chapter 1. Let k be an integer and G be a graph. Two k-colorings are said to be *adjacent* if they di er on exactly one vertex. Given two k-colorings α, β of G (referred to as the *source* and *target colorings*), a *recoloring sequence* between α and β consists in changing the color of one vertex at a time, transforming the source coloring α into the target coloring β, with the property that the coloring is proper at every step. In other words, a recoloring sequence between α and β is a sequence of adjacent (proper) colorings starting on α and ending on β where consecutive colorings di er on exactly one vertex.

Given a graph G and an integer k, the three classic questions considered in recoloring are:

Question 4 (Connectivity)**.** *Is it possible to nd a recoloring sequence between any pair of k-colorings of G?*

The decision associated problem will be de ned as follows:

k-MIXING
Input: A graph G.
Output: YES i there exists a recoloring sequence between any pair of k-colorings of G (i.e. is $G(G,k)$ connected?).

Question 5 (Diameter)**.** *When it is possible, how many steps are needed to ensure the existence of a transformation?*

Question 6 (Reachability)**.** *Let α, β be two k-colorings of G. Is it possible to transform α into β?*

Note that Question 6 is only interesting when the answer to Question 4 is negative. On the contrary, Question 5 only makes sense when the answer to Question 4 is positive (or we should restrict to pairs of colorings between which a transformation exists).

We can restate these questions from an alternative point of view using the con guration graph. Recall that the *k-con guration graph* $G(G,k)$ of G has a vertex for each (proper) k-coloring of G, and an edge between every pair of k-colorings that di er on exactly one vertex of G. Finding a recoloring sequence between two colorings α and β is equivalent to nding a path in $G(G,k)$ between α and β. Such a path is called a *recon guration sequence* from α to β. The *k-recoloring diameter*, denoted by $\mathrm{diam}(G,k)$, is the diameter of $G(G,k)$ [1]. With this point of view, Question 4 becomes: is $G(G,k)$ connected? And Question 5 becomes: what is $\mathrm{diam}(G,k)$?

We can de ne similarly the con guration graph of list colorings. Even if it will not be the main topic of interest in this chapter, in several proofs, we will need to consider list coloring versions. Let G be a graph and $\mathbf{L}$ be a function that assigns to each vertex v a subset of integers L_v. A *list coloring* α *of* $(G,\mathbf{L})$ is a proper coloring of G such that $\alpha(v)$ *2* L_v for every v. Two proper list-colorings are adjacent if they di er on exactly one vertex. As for standard coloring, the *list-con guration graph* $G(G,\mathbf{L})$ of G has a vertex for each list coloring of G, and an edge between every pair of list colorings of G that di er on exactly one vertex of G. We can similarly wonder when the con guration is connected and, when it is, what is its diameter. We can now de ne formally the k-(LIST) REACHABILITY problem:

k-(LIST) REACHABILITY
Input: A graph G, an integer k, (resp. lists of size k assigned to each vertex), two (resp. two list) k-colorings α, β of G.
Output: YES i there exists a (list-)recoloring sequence between α and β (i.e. do α, β belong to the same connected component of $G(G,k)$ (resp. $G(G,\mathbf{L})$)?

[1] By convention, the diameter is + *1* when $G(G;k)$ is disconnected.

Typical behavior when the number of colors varies.
The answer to Questions 4 and 5 can be modi ed when the number of colors k (called the *palette size*) is modi ed. The typical behavior can be described by a series of regimes:

1. When k is too small, proper k-colorings simply do not exist, hence we cannot discuss recoloring.
2. With few colors, proper k-colorings exist, but there is no recon guration sequence between many pairs of colorings. In other words the con guration graph has several connected components.
3. With a larger palette, we reach a regime where recoloring is feasible in general, but recon guration sequences might be long. In other words, the k-con guration graph is connected, or not far from being connected, but has a large diameter.
4. By increasing the number of colors, the recon guration sequences are shorter and shorter.
5. When k is really large, there always exists a recon guration sequence recoloring every vertex a constant number of times.

Note that the description above is a bit simplistic and reality might be a bit more complicated. For instance, for a co-matching of size n, the k-con guration graph is connected when k = 3 but not when $k = n$ as we will see in Section 3.3 (see Figure 3.6 for an illustration).

Most of the research in graph recoloring consists in trying to characterize when there is a phase transition between the regimes described above.

3.2 Recoloring with the lens of degeneracy: Cereceda's conjecture

One of the most widely studied questions is the behavior of the diameter of the k-con guration graph when k is a function of the degeneracy of the graph. A graph G is d-*degenerate* if, for every subgraph G' of G, there exists a vertex of degree at most d. Equivalently, G is d-degenerate if there exists an ordering $v_1, \ldots, v_n$ of the vertices of G such that, for every i, v_i has at most d neighbors in $v_{i+1}, \ldots, v_n$. The ordering $v_1, \ldots, v_n$ is called a d-*degeneracy ordering of* G. Note that a graph G of maximum degree[2] (G) is indeed -degenerate. But the gap between and the degeneracy can be arbitrarily large[3].

The following was observed independently by both Bonsma and Cereceda [31] and Dyer et al. [68]:

[2] (*G*), or when *G* is clear from context, will always denote the maximum degree *G* in the rest of the manuscript.

[3] A star on *n* vertices is 1-degenerate but has maximum degree *n* 1.

Theorem 7 (Dyer et al. [68], Bonsma and Cereceda [31]). *Let d be an integer, $k \geq d+2$ and G be a d-degenerate graph. The graph $\mathcal{G}(G,k)$ is connected and* $\mathrm{diam}(G,k) \leq 2^n - 1$.

Sketch of the proof. By induction on n. The result indeed holds when $n = 1$. Let $v_1, \ldots, v_n$ be a d-degeneracy ordering of G. By induction, there exists a recoloring sequence R from[4] $\alpha_{|G-v_1}$ into $\beta_{|G-v_1}$ of length at most $2^{n-1} - 1$. When we perform the same recoloring sequence in G, the only possible conflicts[5] happen when a vertex v_i has to be recolored with color c but (i) v_i is adjacent to v_1 and (ii) v_1 is currently colored c. In that case, we can first modify the color of v_1 with a color that does not appear in $N[v_1]$ and then recolor v_i. It is possible since v_1 has degree at most d and $k \geq d+2$.

So there exists a transformation between α and β. The vertex v_1 is recolored at most $|R|$ times (since we recolor it at most once for every step in R) plus one last time at the final step to give v_1 its target color. So the total number of recolorings is at most $2|R|+1$, which completes the proof. □

Let α be a coloring of G. A color c is *free* for v in α if c does not appear in $\alpha(N[v])$. When there is no free color for v, we say that v is *frozen*. A coloring is *frozen* if all the vertices are frozen.

Obviously, the analysis of Theorem 7 is far from optimal in practice since (i) and (ii) are indeed usually not satisfied at each step. Moreover, we choose to recolor v_1 with an arbitrary free color. When there are several free colors, some choices might be better than others. The *local best choice* consists in recoloring v_1 with the free color that will appear the latest in $N(v_1)$ in the rest of the transformation[6]. For a d-degenerate graph and $k \geq d+2$, we can define the Best Choice Algorithm as the algorithm given in the proof of Theorem 7 which makes the local best choice each time v_1 has to be recolored. While, in most of the cases, this modification does not affect the worst case analysis of Theorem 7, it sometimes permits to obtain linear bounds. In particular, one can easily prove that the following holds:

Lemma 8 (Bousquet, Perarnau [49]). *Let d, k be two integers such that $k \geq 2d+2$ and G be a d-degenerate graph.*
There exists a reconfiguration sequence between any pair of k-colorings recoloring each vertex at most $d+1$ times. In particular:

$$\mathrm{diam}(G, 2d+2) \leq (d+1)n.$$

[4] For a subset S of vertices and α a coloring, we denote by $\alpha_{|S}$ the restriction of α to the vertices of S.

[5] We say that there is a *conflict* in a reconfiguration sequence if the resulting coloring after the recoloring step is not proper.

[6] The best choice is local in the sense that it might not be optimal in the shortest reconfiguration sequence of the whole graph because of future vertices that have to be included. It can be seen as a greedy choice that minimizes the number of recolorings.

Sketch of the proof. The proof follows the steps of the proof of Theorem 7 except that we keep track of the number of recolorings of each vertex. Since, at each step where v_1 has to be recolored, there are d + 1 free colors for v_1 and the neighborhood of v_1 is recolored at most d (d + 1) times in R, one can easily prove that the BEST CHOICE ALGORITHM recolors v_1 at most d + 1 times. □

Several variants of Theorem 7 have been used as subroutines of recoloring algorithms (e.g. in [39] where it is used with a partition of $V(G)$ into independent sets). Using the BEST CHOICE ALGORITHM (with a much more involved analysis this time), we proved with Valentin Bartier and Marc Heinrich that the con guration graph of 5-colorings of graphs of treewidth 2 have linear diameter [7] (see Section 3.3.1 for more details).

Cereceda's conjecture. For every d-degenerate graph G, Theorem 7 ensures that $G(G,k)$ is connected if k d + 2 but the proof only provides an exponential upper bound on the diameter. And Lemma 8 ensures that $G(G,k)$ has linear diameter if k 2d + 2. This raises two questions:

1. Can diam(G,k) be exponential when $k = d + 2$?
2. What is the behavior of the diameter when the number of colors increases? In particular, when does diam(G,k) become linear?

Both questions are extensively studied. Cereceda proposed in 2009 [53] the following conjecture:

Conjecture 9 (Cereceda [53])**.** *For every d-degenerate graph G, if k d+2 then* diam$(G,k) = O_d(n^2)$.[7]

Cereceda's conjecture is only known to be true for d = 1 [27] (forests). The quadratic function cannot be improved since the diameter of the con guration graph of 3-colorings of paths[8] has quadratic diameter. The lower bound for this proof follows from the following simple but extremely nice argument, which is the only non-trivial lower bound technique known in graph recoloring:

Theorem 10 (Bonamy et al. [27])**.** *The path P_n on n vertices satis es:*

$$\text{diam}(P_n, 3) = \quad (n^2).$$

Sketch of the proof. Let us sketch the main ideas of the proof of the lower bound. The upper bound will follow from Theorem 20 in Section 3.3.1. Let

[7] The notation $f(n) = O_d(g(n))$ means that for every d 0, there exists a constant C_d such that $f(n)$ C_d $g(n)$.

[8] Paths being indeed 1-degenerate.

Figure 3.1: Two 3-colorings of a path at quadratic distance. In the representation of colorings, I adopt the Lucas de Meyer's convention where the initial coloring is the inner color while the target one is the outcolor for each vertex. In other words a vertex colored red in the middle and green outside is colored red in the initial coloring and green in the target one.

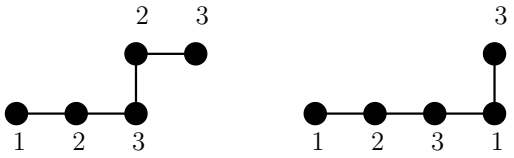

Figure 3.2: Two paths associated to 3-colorings of a path. The path at the right is obtained from the path at the left after the recoloring of the 4-th vertex from 3 to 1.

us denote by $v_1, \ldots, v_n$ the vertices of P_n and α be a coloring of P_n. In the rest of the proof, we consider the colors modulo 3.

For every i n 1, the di erence $\alpha(v_{i+1})$ $\alpha(v_i)$ is either +1 or 1. So we can associate to α an oriented path Q in the n n-grid as follows. We start from the vertex $(0,0)$ in the grid and, for every i, if $\alpha(v_{i+1})$ $\alpha(v_i)$ = +1, then Q goes to the vertex at its right. Otherwise $\alpha(v_{i+1})$ $\alpha(v_i)$ = 1 and then the path Q goes to the vertex above (see Figure 3.2 for an illustration).

Assume that we can recolor v_i from color a to color b in α. Since there are exactly 3 colors, the two neighbors of v_i are colored the same in α (with the third color c). So the recoloring consists in replacing in Q an edge right followed by an edge up into an edge up followed by an edge right (or the converse) (see Figure 3.2). In particular, this single vertex recoloring only modi es by one the area under the curve Q .

Since we can easily construct two colorings whose area under the curve di er by (n^2), the lower bound of Theorem 10 follows: 123123123... and 132132132132... (see Figure 3.1). □

The lower bound of Theorem 10 can be easily extended to chordal graphs G with $(\omega(G) + 1)$ colors [27], where $\omega(G)$ denotes the size of a maximum clique of G. Indeed, we can simply add to the path P_n ω 2 new vertices connected to all the vertices of the path (and all connected together). Note however that it increases arbitrarily the maximum degree of the graph (see Figure 3.3 for an illustration).

Lemma 11 (Bonamy et al. [27]). *For every* ω 2*, there exist chordal graphs* G *such that:*

$$\text{diam}(G, \omega(G) + 1) = (n^2).$$

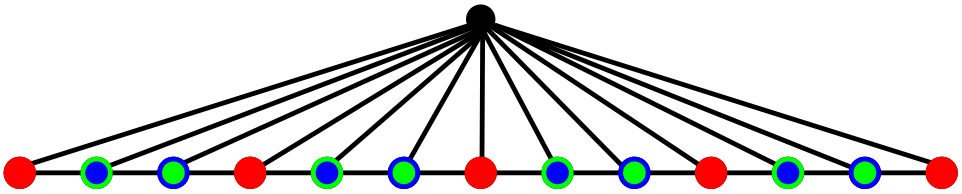

Figure 3.3: Two 4-colorings of a chordal graph G with $\omega(G) = 3$ at quadratic distance.

The tightness of this bound will be proved in Section 3.3.1. Since in all the constructions where the degeneracy is large and the diameter is quadratic we are aware of, there exist large degree vertices. This raises the following question:

Question 12. *Does there exist a function f such that, for every d, there exists a d-degenerate graph G_d of maximum degree $f(d)$ such that* diam$(G_d, d+2) = (n^2)$ *?*

Cereceda's proof only ensures that the con guration graph has exponential diameter. One can wonder if we can at least ensure that the diameter of the con guration graph is polynomial. Together with Marc Heinrich we proved in [41] the rst polynomial upper bound (depending on d) on the diameter of the con guration graph. In particular, it provides the only proof that diam$(G, 4)$ is polynomial for 2-degenerate graphs G.

Theorem 13 (Bousquet, Heinrich [41])**.** *Let d, k be two integers and G be a d-degenerate graph.*

- *if* $k \quad \frac{3}{2}(d+1)$ *then* diam$(G,k) = O(n^2)$*;*
- *if* $k \quad (1+\epsilon)(d+1)$ *and* $0 < \epsilon < 1$ *then* diam$(G,k) = O(n^{d1=e})$*;*
- *if* $k \quad d+2$ *then* diam$(G,k) = O_d(n^{d+1})$*.*

Sketch of the proof. We will not give a complete proof of Theorem 13 but simply its avor for $k = d+2$. Let us start with a simple remark. In order to prove that the diameter is $O(n^{d+1})$, instead of proving that we can transform any coloring α into any coloring β, we will prove that we can transform α into a $(d+1)$-coloring in $O(n^{d+1})$ steps[9]. The conclusion then follows from the following very simple claim whose proof is left as an exercise.

Claim 14. *Let d, k be two integers such that $k \quad d+2$. If any k-coloring of a d-degenerate graph G can be transformed into a $(k \quad 1)$-coloring in N steps then the diameter of* G(G,d) *is at most* $2dN$*.*

[9] This technique is pretty standard in recon guration as we will see all along this chapter.

Let us now explain how we can transform α into a (k 1)-coloring. In the proof of Theorem 7, we peeled the graph by removing small degree vertices one after another. The rst idea of the proof consists in taking the opposite point of view: we will start with v_1 and add the vertices in the reverse ordering. In other words, we will consider the subgraph induced by $v_1, \ldots, v_i$ for increasing i.

When we do so, we cannot simply remove the other vertices of the graph as we did in Theorem 7. Indeed, if we recolor the vertex v_1 with the color of v_j for some j i, we are not sure to be able to recolor v_j with another color to authorize this recoloring. Indeed, the vertex v_j might have a large degree and then v_j might be frozen in the current coloring. So instead of deleting vertices $v_{i+1}, \ldots, v_n$, we will x their color[10]. These vertices will be called the *blocked vertices*.

The blocked vertices $v_{i+1}, \ldots, v_n$ set some constraints on the vertices $v_1, \ldots, v_i$, and we will represent these constraints with list colorings. Formally, let α be a coloring of G and i n. For every $j < i$, the list $L_i(v_j)$ of v_j (for α) is $[1,k]$ n $\alpha(N[v_j] \setminus \{v_{i+1}, \ldots, v_n\})$ [11]. Since k d + 2, one can easily remark that, for every i, j, the number of neighbors N_j of v_j in $v_{j+1}, \ldots, v_i$ satis es $|L_i(v_j)|$ $|N_j|$ + 2 (called property ($\star$)).

Let us denote by G_i the subgraph of G induced by $\{v_1, \ldots, v_i\}$ and $d_i^+(j) = |N(v_j) \setminus \{v_{j+1}, \ldots, v_i\}|$. Property ($\star$) can be rephrased as follows: the order $v_1, \ldots, v_i$ satis es that, for every j i, $|L_i(v_j)|$ $d_i^+(v_j)$ + 2. Note that for $i = n$, it simply means that $v_1, \ldots, v_n$ is a d-degenerate ordering of G and k d + 2.

The second idea of the proof consists in introducing the notion of full colors. A color c is *full* for $v_1 \ldots, v_i$ (for a coloring α) if, for every j i, one of the following holds:

1. $\alpha(v_j) = c$ or,
2. there exists $j' > j$ such that $v_{j'}$ is colored c in G_i or,
3. c 2 $L_i(v_j)$.

Note that if a color c is full for $v_1, \ldots, v_n$ then, every vertex is either colored c or has a neighbor after it in the ordering colored c. So the removal of all the vertices colored c (and of the color c from the set of colors) leaves a instance where:

- The degeneracy has decreased by one and,
- The number of colors has decreased by one.

[10] In the sense that we will not modify their colors.

[11] For every subset of vertices S, we denote by (S) the set of colors in S.

So if we can reach a coloring where a color c is full for $v_1, \ldots, v_n$ we can apply induction on k. Indeed, we remove the color c and all the vertices X colored c and apply induction on $G^0 = G[V \setminus X]$. So to conclude we simply have to prove that we can transform α into a coloring with a full color in a polynomial number of steps.

One can easily prove that $\alpha(v_1)$ is full for v_1. The rest of the proof, not detailed here, consists in proving by induction that if we have a coloring such that a color c is full for $v_1, \ldots, v_i$, then we can find a (short) recoloring sequence of G_i to a coloring such that a color c' is full for $v_1, \ldots, v_{i+1}$. We repeat this operation at most n times to reach a coloring where a color c is full for $v_1, \ldots, v_n$.

The proofs of the other upper bounds follow the same scheme except that: (i) since there are more colors, we can find colorings where several colors are full (which allow us to reduce faster the degeneracy of the graph) and, (ii) we can stop when the number of colors is at least $2d + 2$ rather than when $d = 0$ since, in that case, Lemma 8 ensures that there exists a reconfiguration sequence of linear length. □

Note that the first item improves a result of Cereceda [53] who showed that diam$(G, 2d + 1)$ is quadratic for any d-degenerate graph G.

We have decided to state Conjecture 9 in its weakest version since we allowed the constant in front of the quadratic function to depend on d. We actually have no proof that it is necessary. This question is somehow related to Question 12 since we cannot obtain a quadratic diameter without "cheating" by arbitrarily increasing the maximum degree of the graph.

Conjecture 15 (Strong Cereceda's conjecture)**.** *There exists a constant C such that, for every d-degenerate graph G and every $k \quad d+2$,* diam(G, k) $C \quad n^2$.

Note that in most of the cases where Cereceda's conjecture has been proven, the constant does not depend on d (see e.g. Theorem 20 and 21 in Section 3.3.1).

Let us end this part with some open problems:

Conjecture 16 (Bartier, Bousquet [36])**.**

- *There exist $c < \frac{3}{2}$ and r 2* N *such that* diam$(G, cd + r) = \quad (n^2)$ *for every d-degenerate graph G.*
- *There exist $c <$ 2 and r 2* N *such that* diam$(G, cd + r) = \quad (n)$ *for every d-degenerate graph G?*

Note that a positive answer to the second question immediately implies a positive answer to the first one as observed in [41] (it follows directly from the proof of Theorem 13).

The particular case of Cereceda's conjecture for d = 2 is, so far, still open.

Conjecture 17. *There exists a constant c such that* diam$(G, 4)$ *is quadratic for every 2-degenerate graph G.*

Recall that Cereceda's conjecture is only known to be true when d = 1 [27] and d = 2 when = 3 [78][12]. For d = 2, the best known general upper bound is $O(n^3)$ by Theorem 13.

Maximum average degree versus degeneracy. The *average degree* of G is $2\frac{|E(G)|}{|V(G)|}$. The *maximum average degree* of G, denoted by $mad(G)$ is the maximum, over all the induced subgraphs H of G, of the average degree of H. There are natural relations between maximum average degree and degeneracy. Indeed, a graph of maximum average degree d has degeneracy at most d and a graph of degeneracy d has maximum average degree at most $2d$ (since every subgraph on r vertices has at most dr edges).

By adapting the ideas of the proof of Theorem 13 together with a partition of the graph into independent sets of linear size, Carl Feghali proved that the following holds:

Theorem 18 (Feghali [76])**.** *Let d, k be two integers such that k d + 1 and $\epsilon > 0$. Every graph G with maximum average degree d ϵ satisfies*[13]

$$\text{diam}(G, k) = O\left(n(\log n)^{d\ 1}\right).$$

Theorem 18 strengthens a result we obtained with Guillem Perarnau [49] which only provides a polynomial upper bound on the diameter. For d-regular graphs, Theorem 13 only ensures that the diameter is at most $n^{d\ 1}$ when k = d + 2 while Theorem 18 ensures that the diameter is almost linear. Theorem 18 also gives the best known upper bounds on the recoloring diameter for planar graphs (for k 8) and many of its restricted classes. However, in planar graphs (and several other graph classes), the maximum average degree tends to 6. So, Theorem 13 gives the best upper bound for k = 7 while Theorem 18 is better as long as k 8. So, in general, the two results are incomparable (see Section 3.3.2 for a more complete discussion on planar graphs).

Note that the number of colors cannot be improved in Theorem 18 since paths P have maximum average degree (tending to) 2 and $G(P, 3)$ has quadratic diameter. Similarly, as we observed before, it cannot be improved in Theorem 13 either since the k-con guration graphs of cliques are not connected when k = d + 1.

[12] The recent result of [39] even ensures that the diameter is linear in that case.

[13] The notation O means that the constant might depend on .

3.3 Cereceda's conjecture on restricted graph classes

3.3.1 Chordal graphs and bounded treewidth graphs.

Before showing that Conjecture 9 holds on chordal graphs, let us first make the following remark:

Lemma 19 (Cereceda [53]). *Let K_n be the clique on n vertices and $k \geq n+1$. Then there exists a transformation between any pair of k-colorings α, β of K_n recoloring every vertex at most twice.*

Sketch of the proof. Let D be the digraph on n vertices with an arc from x to y if y is colored in $\beta(x)$ in α. Informally, xy is an arc if the current color (in α) of y prevents the recoloring of x into its final color (in β). Since the graph is a clique and α, β are proper colorings, $d^+(x) \leq 1$ and $d^-(x) \leq 1$ in D for any $x \in V$. Hence D is the disjoint union of directed paths and circuits.

One can easily remark that we can directly reconfigure directed paths by recoloring every vertex of the path at most once. Indeed, we first recolor the vertex of out-degree 0 with its target color and repeat until the path is empty.

Since $k \geq n + 1$, every vertex can be recolored with a free color. So, for circuits, we first recolor any vertex of the circuit with a free color. The circuit is then transformed into a directed path and the conclusion follows. □

Bonamy et al. proved in [27] that Conjecture 9 holds for chordal graphs (they even prove the strongest version of the conjecture, Conjecture 15 holds). Note that the quadratic function cannot be improved for any value of d when $k = d + 2$ as we have seen in Lemma 11.

Theorem 20 (Bonamy et al. [27]). *Let d, k be two integers such that $k \geq d+2$. For every d-degenerate chordal graph G, there exists a reconfiguration sequence between any pair of k-colorings recoloring every vertex at most n times.*

In particular $\mathrm{diam}(G, k) \leq n^2$ *for every chordal graph when* $k \geq \omega(G)+1$.

Sketch of the proof. The second point simply follows from the fact that $\omega(G) = d + 1$ for chordal graphs (chordal graphs being perfect). The idea of the proof of the first point is based on what became a classical argument on reconfiguration proofs: the *vertex identification technique*.

Let us prove by induction on the number of vertices of G that it is possible to transform a coloring α into a coloring β by recoloring every vertex at most n times. If the graph G is a clique, the result holds by Lemma 19 (the proof is illustrated on Figure 3.4). Let T be a clique tree of G where no bag is included in any other and v be a vertex that only belongs to a leaf of the clique tree. Such a vertex is called a *simplicial vertex*. (It is well-known that

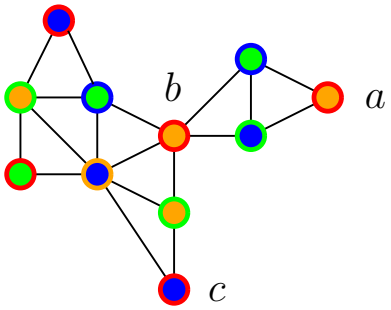

Figure 3.4: Initial coloring is on the outside while target coloring is on the inside. The vertices a and b are colored the same in the initial and target coloring so they can be merged into a single vertex. On the contrary, c has to be recolored before being merged with b. It is possible since c is simplicial.

every chordal graph admits a simplicial vertex.) Note that the neighborhood of a simplicial vertex is a clique. Let X_v be the bag of v in T and X be the bag of the neighbor of X_v in T. The bag X contains a vertex w that is not in X_v (otherwise X_v can be contracted with X in the clique tree).

Let us prove that we can recolor in α (and similarly in β) the vertex v with the color of w by recoloring vertices of X_v at most once. If v is not colored with the color c of w and has no neighbor colored c in α, then we can directly recolor it with c. Otherwise v has a neighbor y colored c. Since y is not adjacent to w (the coloring is proper), y only belongs to X_v in T. In particular y is simplicial. Since $k \quad \omega + 1$, we can thus recolor y with a new color and then color v with c. So, by recoloring every vertex of X_v at most once, we obtain two colorings α', β' where v, w are colored the same.

The contraction of v and w gives a new chordal graph with clique number ω and $n \quad 1$ vertices. By induction, we can recolor α' into β' by recoloring every vertex at most $n \quad 1$ times. This recoloring can be extended to G by performing the same sequence (and recoloring both v and w when the contracted vertex is recolored). □

Theorem 20 has been extended by Bonamy and Bousquet to bounded treewidth graphs [19]. We propose here a very short proof of that result by Feghali [74]:

Theorem 21 (Bonamy, Bousquet [19], Feghali [74])**.** *Let t, k be two integers such that $k \quad t + 2$. Every graph G of treewidth at most t satisfies* $\mathrm{diam}(G, k) \quad n^2$.

Proof. Let us sketch the short proof of Feghali [74] which reduces this statement to Theorem 20 (with a slightly worse bound on the diameter). As in the proof of Theorem 13, the idea is to prove that the following holds:

Claim 22. *Let G be a graph of treewidth t and k $t+2$. Then we can reach from any k-coloring γ of G a (k 1)-coloring of G in at most n^2 steps.*

Before sketching the proof of this claim, let us explain how we can use it to derive Theorem 21. We can transform α (resp. β) into a coloring α' (resp. β') using only k 1 colors in at most n^2 steps. One can easily show that every graph contains an independent set I whose removal reduces the treewidth of the graph by (at least) one. We color in α' and β' all the vertices of I with color k. So the colorings now agree on I. We conclude by recoloring by induction $\alpha'(G \setminus I)$ into $\beta'(G \setminus I)$ (with at most k 1 colors) in $O(n^2)$ steps.

Let us now explain how to prove Claim 22 whose idea has been reused in several other proofs.

Proof of Claim 22. Let γ be a coloring of G. If a bag X of a tree decomposition T of G contains two vertices u, v colored the same in γ, then we contract them (and the width of the resulting graph has not increased).

We repeat this operation until, in all the bags, all the vertices are colored di erently. Let us denote by G' the resulting graph and γ' the coloring of G'. If we consider the chordalization H of G (that is the graph obtained from G' by adding all the edges between any pair of vertices belonging to the same bag of T), then γ' is a proper coloring of the chordal graph H and k $\omega(H)+1$. Thus, by Theorem 20, we can transform γ' into an ω-coloring of H by recoloring every vertex at most n times. To conclude, one can easily remark that we can extend this recoloring sequence into a recoloring sequence from γ to an $(t+1)$-coloring of G in G (since $\omega(H)$ $t+1$ and identi ed vertices form an independent set). □

□

One can wonder what happens when the number of colors increases. Lemma 8 ensures that the diameter is linear when the number of colors is twice the treewidth plus 2. In particular, for graphs of treewidth 2, the diameter is linear when k 6. We improve this result with Valentin Bartier and March Heinrich and settle the whole picture for graphs of treewidth 2:

Theorem 23 (Bartier, Bousquet, Heinrich [7])**.** *Let G be a graph of treewidth at most 2 and k 5. Then* diam$(G, 5)$ C n *for some integer C.*

Sketch of the proof. We rst use the Feghali's trick described in the proof of Theorem 21 to transform G into a chordal graph of clique number at most three. By abuse of notations, we still denote by G the resulting graph.

Let α, β be two 5-colorings of G. The proof then consists in proving that the BEST CHOICE ALGORITHM (BCA) described just after Theorem 7 recolors every vertex a bounded number of times. We refer to the proof of Lemma 8 for notations. Let v_1 be a vertex of degree 2 (which exists since the

treewidth is 2) and let S be the recon guration sequence given by the BCA between α and β in $G \quad v$. When we try to adapt the proof of Lemma 8, v_1 can be recolored at most C + 1 times. We assume by contradiction that it is indeed the case and show that many strong conditions must be satis ed on the sequence of recolorings.

We can (for instance) prove that:

- Both neighbors of v_1 are recolored exactly C times in S.
- v_1 is recolored once every two recolorings of its neighbors in S.
- the neighbors of v_1 are recolored alternatively in the sequence S and follow a precise recoloring sequence.

Indeed, since the two neighbors of v_1 are adjacent (v_1 is simplicial), we can derive a lot of information on the structure of the recolorings of $N(v_1)$ in S. And since they are themselves recolored C times, there is a lot of structure on how their own neighborhoods are recolored. Finally, we prove that if C is large enough, it gives too many constraints on the sequence of recolorings, which leads to a contradiction with the choices performed by the BCA algorithm. □

One can note that the number of colors in Theorem 23 is the best we can hope for by Lemma 11.

We were not able to generalize this statement to larger values of k. In particular, we were not able to slightly improve the bound of Lemma 8 for bounded treewidth graphs and show that the diameter is linear even when the number of colors is at most 2 $\mathbf{tw}(G)$ + 1. As for Theorem 23, we simply have to prove that we can save one recoloring in the Best Choice Algorithm to obtain this bound but we did not succeed to prove it. We left the following as open problems:

Conjecture 24 (Bartier, Bousquet, Heinrich [7]). • *There exists a constant C such that* diam$(G, \mathbf{tw}(G) + C) = O(n)$ *for every graph G.*

- *There exists a constant C such that* diam$(G, \omega(G) + C) = O(n)$ *for every graph chordal graph G.*

A weakening of this question was asked in [36] for chordal graphs: Even the existence of a real $r < 2$ such that diam$(G, r \ \mathbf{tw}(G) + C)$ is linear is still open. Also note that this question is the restriction in the case for bounded treewidth graphs of Conjecture 16.

With Valentin Bartier, we proved that Conjecture 24 holds for chordal graphs whose maximum degree is bounded by a function of the maximum clique. Namely we showed that the following holds:

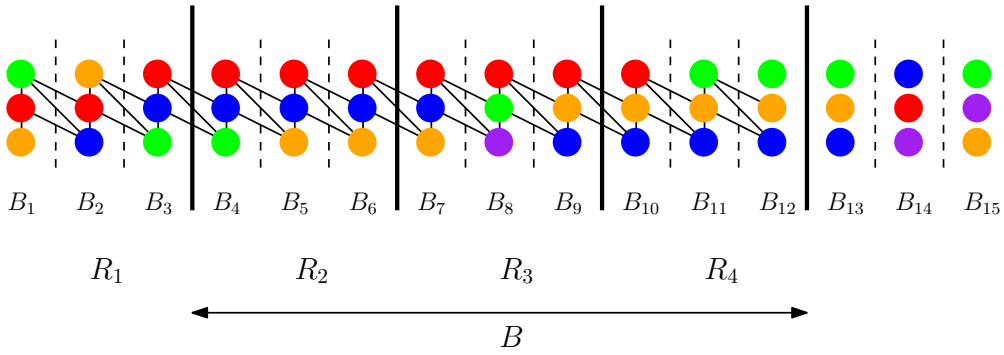

Figure 3.5: There is no edge between non consecutive blocks. A region is a subset of three blocks. The second and fourth regions are color change regions and the third is a permutation region. The bu er here is the second, third and fourth region. While we have a control on what happens in the bu er, what is at the left is the nal coloring and what is at the right is the initial coloring.

Theorem 25 (Bartier, Bousquet [36])**.** *Let G be a d-degenerate chordal graph of maximum degree . For every k $d+4$,* diam$(G,k) = O(f( ) n)$*.*

Moreover, given two colorings α, β of G, a transformation of length $O(f( )n)$ can be found in linear time.

Proof. The proof of this result is quite technical and we will try to give the main intuition of it in a very particular case: power of paths. (The notions of the proof are illustrated on Figure 3.5).

Let P be a path on n vertices on vertex set $v_1, \ldots, v_n$ and let us denote by G the graph where v_i, v_j is an edge if and only if $ji \quad jj \quad \omega$. Powers of paths are indeed chordal and one can remark that ω consecutive vertices form a clique. There is a natural partition of the vertex set into blocks of ω consecutive vertices. Also remark that there is a very simple coloring of the graph with ω colors where the i-th vertex of each block is colored with color i. Indeed, for every $i \quad \omega$, the union of the i-th vertices over all the blocks form an independent set. (In other words, in Figure 3.5, the vertices on the same row form an independent set).

In what follows, we will see colorings of G with a slightly di erent perspective: A coloring will be considered as a sequence of n/ω (ordered) vectors of ω colors. The rst vector of colors corresponds to the colors of the vertices $v_1, \ldots, v_!$; the second vector corresponds to the vertices $v_{!+1}, \ldots, v_{2!}$ and so on.

Imagine now that we have two colorings α, β and that we want to recolor from α to β. Our goal will just consist in trying to recolor one sequence of ordered vectors of colors into the other. Until now, we have simply rede ned the notion of colorings as a sequence of ordered vectors.

The key idea of the proof now consists in proving that, if we have a certain amount of consecutive ordered vectors (whose size is bounded in terms of k and), called a *bu er* such that (i) the coloring of the bu er satis es some good properties and such that, (ii) at the left of the bu er the coloring already is the target coloring and, (iii) at the right of the bu er, the coloring is the initial coloring then we can slide by one vector at the right the bu er by only recoloring vertices of the bu er in such a way (i), (ii) and (iii) still holds at the end of the recoloring process.

A coloring satisfying (i), (ii), (iii) is called a good coloring. What is a good coloring of a bu er? It should have several properties. The rst one is the following:

- Between two consecutive blocks B, B', at most one color changes[14]. In other words, there is at most one integer i such that the color of the i-th vertex of B di ers from the color of the i-th vertex of B'.

This property allows us to be sure that we can recolor any vertex in the bu er since the total number of colors in the neighborhood of a vertex consists of at most $\omega + 2$ colors (every vertex is only adjacent to its block and the blocks before and after it). But knowing that we can recolor every vertex is indeed not enough to nd an e cient recoloring algorithm.

The second idea of the proof consists in partitioning the blocks of the bu er three by three. A set of three consecutive blocks is called a *region*. A good coloring of the bu er should satisfy the following: for every region, the three consecutive blocks should be of one of the three following types (see Figure 3.5 for an illustration):

- An identity region: the (ordered) vector of colors of the three blocks is the same.
- A color change region: there is at most one color di erence between the rst and the last block of the region; that is there is exactly one index i such that the color of the i-th vertex has changed (and all the other colors remain the same).
- A permutation region: there exist two integers i, j such that, for every $r \neq i, j$, the color of the r-th vertex is the same in the rst and the last block. And the colors of the i-th and j-th vertex of the rst and last blocks are permuted.

One can prove that, if we take a long enough bu er in terms of k, we can indeed nd a proper coloring where the rst block corresponds to the target coloring and the last block is the initial coloring and where all the regions are identi ed, color change or permutation regions.

[14] Actually there will be some small exceptions, but let us assume it for simplicity.

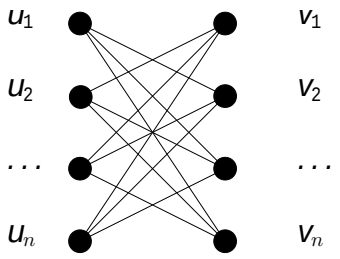

Figure 3.6: The complete bipartite graph $K_{n;n}$ minus a matching is 2-chromatic but admits frozen n colorings.

The tour de force consists in proving that if we have a bu er for a set of consecutive blocks, then we can move the bu er at the right of one step by simply recoloring vertices of the bu er. The idea consists in adding color changes and permutation regions to t the coloring of the new block that is now at the right of the bu er. However, we cannot add such regions forever (the bu er being of bounded size). We prove that if we apply twice a permutation region on the same pair i, j, then we can remove both permutations and replace both regions with identity regions. And the same holds for color change regions (which is much easier to prove). □

Recall that for chordal graphs, we have $\omega(G) = d+1$ so the result ensures that the diameter has size at most $O( ) n$) when $k \quad \omega + 3$. We end this part with the following question:

Conjecture 26. *If G is a d-degenerate chordal graph of maximum degree then* diam$(G, d + 3)$ *is at most $O(f( )n)$.*

In some very restricted cases (such as powers of paths), our proof technique can be extended to $k = d + 3$ but this is mainly due to the very strong structure of these graphs.

Beyond chordal graphs. One can wonder if all these results on chordal graphs can be generalized beyond. A rst natural generalization of chordal graphs are perfect graphs (graphs with no odd holes[15] and no antiholes). One can ask if, for perfect graphs, it is always true that $G(G, k)$ is connected as long as $k \quad \chi(G) + 1$. The answer is negative even on bipartite graphs, as shown by the following example.

Consider a complete bipartite graph $G = K_{n;n}$ where the set of vertices is $U \, [\, V$. We now remove from G a perfect matching between U and V (see Figure 3.6 for an illustration). One can easily remark that:

- The graph is perfect and 2-colorable (as it is bipartite).

[15] A *hole* being an induced cycle of length at least 4. An antihole is the complement of a hole.

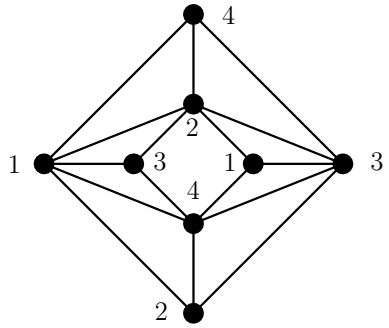

Figure 3.7: A 3-colorable weakly chordal graph with a frozen 4-coloring.

- $G(G, k)$ is connected for every 3 k n 1 (a color must belong only to one part and we can then reach a 2-coloring in $O(n)$ steps).
- $G(G, n)$ contains isolated vertices (by coloring each edge of the anti-matching with a new color).

A second natural generalization of chordal graphs are weakly chordal graphs. A graph is *weakly chordal* if it does not contain any hole nor anti-hole of length at least 5. Weakly chordal graphs are not perfect but their structure has been widely studied. In particular, an equivalent characterization of weakly chordal graphs in terms of 2-pairs exists. Again, one can naturally wonder if, for every weakly chordal graph G, $G(G, k)$ is connected as long as k $\chi(G) + 1$. Feghali and Fiala proved in [77] that this is not the case. The result was improved by Merkel in [115] who showed that, for every c, there exists a weakly chordal graph G_c such that $G(G_c, \chi(G_c) + c)$ is not connected. To do so Merkel started with the triangle-free weakly chordal graph of Figure 3.7 which admits a frozen 4-coloring. Then he proved with an inductive construction based on blow-ups that there exist weakly chordal graphs G_c with frozen k-colorings where k is large compared to $\chi(G_c)$.

Nevertheless, for both graph classes, the following is still open:

Question 27. *Do perfect graphs satisfy the Cereceda's conjecture? What about weakly chordal graphs?*

3.3.2 Planar graphs and related classes.

Planar graphs. In the last few years, planar graphs and related (sub)classes received considerable attention. Since planar graphs are 5-degenerate, it is possible to recolor any 7-coloring into any other in at most $O(n^6)$ steps by Theorem 13. Moreover, by Theorem 18, it is possible to nd a transformation of length $O(n$ polylog$(n))$ when k 8.

One can wonder if the value of 7 can be improved. In other words, is it possible to transform any 6-coloring of a planar graph into any other? The

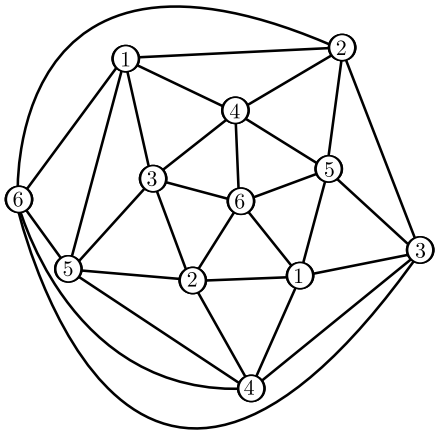

Figure 3.8: A frozen 6-coloring of a planar graph.

answer is negative as mentioned in [19]. There exists a frozen 6-coloring of a planar graph represented in Figure 3.3.2. (A coloring is said to be *frozen* when the color of no vertex can be modi ed.)

The second question that attracted a lot of attention in the last few years is the following: when can we ensure that there exists a linear transformation between any pair of k-colorings of a planar graph? Lemma 8 ensures that $G(G, 12)$ has linear diameter for every planar graph G. Dvorak and Feghali proved that this bound can be improved to 10:

Theorem 28 (Dvorak, Feghali [65, 67])**.** *For every planar graph G and every k 10, the diameter of $G(G, k)$ is at most $7n$.*

Sketch of the proof. This result has been proven twice. The authors rst proposed a proof using a discharging argument in [66]. The idea consists in transforming the proof into a 9-coloring by recoloring every vertex at most once or at most twice but then the rst recoloring consists in changing the color for color 10.

But we will focus here on another proof of that result proposed (again) by Dvorak and Feghali a few months later using a *Thomassen type approach* [67]. The high level idea of the proof consists in adapting the proof that planar graphs are 5-choosable[16] (due to Thomassen) [129].

Let us rst brie y re-explain the proof of [129] which is by induction on the number of vertices. The key ingredient of the proof is the induction hypothesis. Instead of assuming that all the vertices have lists of size 5, we assume that all the vertices have lists of size 5 but the vertices of the external face that have lists of size 3. Moreover an edge uv of the outerface has a xed coloring. Let us now explain how we can reduce the size of the

[16] That is, we can list color every planar as long as every vertex receives a list of size at least 5.

graph to apply induction. Assume first that the outerface has a chord. This chord cuts the graph (and the outerface) into two parts of stricly smaller size which both contains this chord. We first color the part containing the edge uv (whose color is fixed) by induction. This coloring then selects a color for the chord, which fixes the color of an edge of the outerface of the other part on which we apply induction. If the outerface contains no chord, we select a vertex w of the outerface adjacent to v and we (i) remove it from the graph, (ii) remove two colors of the list of w (different from the colors of u and v) from the lists of the neighbors of w. We can easily prove that the induction applies and that a coloring of the remaining graph can be extended to v.

Let us now come back to the proof of Theorem 28. In the rest of this proof, we will only very briefly sketch the argument that consists in proving that we can transform any 10-coloring into a 9-recoloring by recoloring every vertex at most twice with a similar technique. (One can easily derive from that statement that the configuration graph has linear diameter).

The general idea is the same. Let us first propose a model (which will not work and be adapted later). All the vertices which are not internal have lists of size 9 (the color 10 is forbidden)[17]. The vertices of the outerface have lists of size 5 and we fix the recoloring of an edge uv of the outerface. Note that some vertices might be colored with colors that are not in their lists (e.g. vertices colored 10) but we want to be guarantee that (i) if we recolor a vertex, its new color is in its list, (ii) at the end all the vertices are colored with colors in their lists and (iii) all the vertices are colored at most once.

Unfortunately, this approach can hardly work if we only fix the recoloring of an edge and try to remove only one vertex of the outerface at each step as in the Thomassen proof. So the idea of Dvorak and Feghali consists in removing all the vertices of the outerface at the same time instead of simply one vertex.

This raises a serious issue since one internal vertex might be adjacent to all (or at least many) vertices of the outerface. Indeed, if we remove the color of the vertices of the outerfaces to their neighbors, we may end up with a vertex with an empty list of colors after this operation. To avoid this problem, we will fix the recoloring of a P_3 (or a triangle) instead of an edge. For some reasons we will not detail here, this P_3 plays the same role as the chord in the Thomassen proof.

Let us now sum up our assumptions. There are three vertices of the outerface for which the recoloring sequence is completely fixed and we know that their recolorings have to be performed at the very last steps of the transformation. All the other vertices of the outerface have lists of size 5 and the internal vertices have lists of size 9. Moreover, if we remove the external face, vertices still have lists of large size. So we would like to apply

[17] Actually in the model all the vertices do not necessarily have colors of their own lists but if they are recolored, they have to be with a color of their lists.

induction as in the Thomassen proof.

Unfortunately, there is a hidden pathological case. In order to patch this problem, the idea of Dvorak and Feghali consists in saying that they authorize to recolor the P_3 in such a way the intermediate color might be 10. □

Subclasses of planar graphs Let us now brie y discuss the behavior of the diameter of the con guration graph on subclasses of planar graphs. For outerplanar graphs[18], the picture is complete. Indeed, a triangle being outerplanar, $G(G,3)$ is not necessarily connected[19]. The example of Lemma 11 with clique number 3 is chordal and ensures that $G(G,4)$ can have quadratic diameter. And Theorem 23 ensures that diam$(G,5)$ is linear since outerplanar graphs have treewidth at most 2.

With Marc Heinrich, we proved in [41] that every bipartite planar graph[20] G satis es that diam$(G,5)$ is at most quadratic. The proof of this result is based on a discharging argument (as many results discussed in the rest of this section). This result has been strengthened by Cranston and Mahmoud (again with a discharging proof) who proved that the same holds if we consider planar graphs with no triangle nor cycles of length 5 [60]. Many results have been obtained on the diameter of the con guration graph depending on the girth of the planar graph. As far as we know, they are all summarized on Table 3.3.2.

Girth/colours	4	5	6	7
3	$+1$	$+1$	$+1$	$O(n^6)$ [41]
4	$+1$	$O(n^4)$ [41]	$O(n\log^3(n))$ [76]	$O(n)$ [67]
5	$<+1$ [5]	$O(n\log^2 n)$ [76]	-	-
6	$O(n^3)$ [41]	$O(n)$ [5]	-	-
7+	$O(n\log n)$ [76]	-	-	-

Table 3.1: Existing and open cases for the diameter of the k-con guration graph of planar graphs with girth g for some combinations of values of k and g. Note that any bound at a given position in the table implies the same bound at its right and below it.

Recently, list recoloring of (subclasses of) planar graphs has been studied. Cranston proved in [59] that the (list)-con guration graph has linear diameter when (i) G is a triangle-free planar graphs and lists have size at least 7 or (ii) $mad(G) < 17/5$ and lists have size at least 6 or (iii) $mad(G) < 22/9$ and list have length at least 4. Results of the same avor (depending on the

[18] A graph is *outerplanar* if it can be drawn in the plane in such a way all its vertices belong to the external face.

[19] Recall that all outerplanar graphs are 3 colorable since they are 2-degenerate.

[20] Planar graphs that are also bipartite.

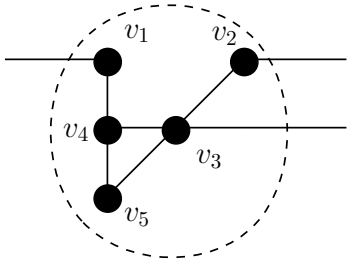

Figure 3.9: A 2-degenerate 1-island with the ordering $v_1, \ldots, v_6$. Assume that all the vertices of $V \setminus \{v_1, \ldots, v_6\}$ are recolored c times. Since v_1, v_2 only have one neighbor on the outside. So they are recolored at most $c/2+1$ times with the BCA. v_3 is recolored at most $\frac{1}{2}(c/2+1) + \frac{1}{2}c + 1 = 3c/4 + 2$ times; v_4 is recolored at most $\lceil \frac{1}{2}(c/2+1) + \frac{1}{2}(3c/4+2) + 1 \rceil = 5c/8 + 3$ times; and finally v_5 is recolored at most $7/8c + 4$ times. So if c is large enough, all the vertices are recolored less than c times.

sizes of faces adjacent to triangles) have been recently proven by Chandran, Gupta and Pradhan [57].

I would only like to end this section with the sketch of a very simple proof of a result of Table 3.3.2 which uses the concept islands of planar graphs. (The proof is not based on a discharging argument but the existence of such islands is proven with a discharging argument).

Theorem 29 (Bartier et al. [5]). *The configuration graph of* 5*-colorings of planar graphs of girth at least* 6 *has linear diameter.*

Sketch of the proof. The proof is based on a result of Esperet and Ochem [71] stating that every planar graph of girth at least 6 contains a 2-degenerate 1-island of bounded size. We say that H is a 2-*degenerate* 1-*island* if the following holds:

- each vertex $v \in V(H)$ has at most one neighbor in $G \setminus H$, and
- there exists an ordering $v_1, \ldots, v_{|V(H)|}$ of the vertices of H such that for each $i \in \{2, \ldots, |V(H)|\}$, the vertex v_i has at most two neighbors in the graph $G \quad H + \{v_1, \ldots, v_{i-1}\}$.

Since every large enough planar graph of girth at least 6 contains a 2-degenerate 1-island H of bounded size, the proof simply consists in proving that, if we can recolor $G \setminus H$ without recoloring too much every vertex, then the same holds for G. To do so, we use the BEST CHOICE ALGORITHM and prove that, since every vertex of H only has 1-neighbor which is not in the island, we save some recolorings (compared to 2-degenerate graphs), which permits to conclude. (See Figure 3.9 for the counting on a very simple example which gives the idea of the proof). □

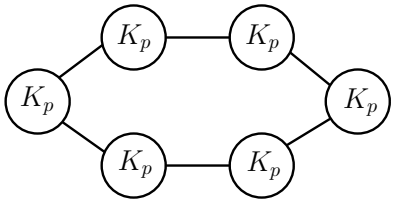

Figure 3.10: A blow-up of C_6 with cliques of size p. The chromatic number is $2p$ but the graph admits a $3p$ frozen coloring by coloring the i-th set with colors the colors $f3 \pmod 3), \ldots, 3 \pmod 3) + p \quad 1g$.

The proof of Theorem 29 is interesting since it removes at each step a subset of vertices instead of a single vertex or an independent set as it is usually the case. It would be interesting to see if similar larger removal subsets can be applied in other settings to prove or improve upper bounds on diameter of con guration graphs. Note that other types of islands exist in various subclasses of planar graphs but, as far as we tried, none gave us any relevant upper bound on the diameter of the con guration graph.

3.3.3 H-free graphs

Let H be a graph. A graph G is *H-free* if G does not contain H as an induced subgraph. One can wonder for which graphs H we can guarantee that $G(G, k)$ is connected or has a small diameter for every H-free graph G. The main topic of study in H-free graphs focuses on the case $k = \chi(G) + 1$.

Let $k \quad \chi(G) + 1$. Lemma 19 ensures that $\text{diam}(G, k) \quad 2n$ for every P_3-graph G. Bonamy and Bousquet proved that $G(G, k + 1)$ is connected and has linear diameter for every P_4-free graph G [19]. The diameter of the con guration graph has been improved to $4n$ in [15]. Merkel proved in [115] that $\text{diam}(G, k)$ is linear when G is a $3K_1$-free graph. On the negative side, since bipartite graphs might admit frozen colorings with an arbitrarily large number of colors (as we observed in Figure 3.6), $G(G, k)$ is not necessarily connected when H is a K_3-free graph. Feghali and Merkel also proposed in [80] a $2K_2$-graph (and then P_5-free) graph G which is 7-chromatic but admits a frozen 8-coloring.

A complete dichotomy theorem was recently obtained by Belavadi, Cameron, Merkel in [12] who characterized exactly for which graphs H, $G(G, k)$ is connected for every H-free graph G.

Theorem 30 (Belavadi, Cameron, Merkel [12])**.** *For every H. $G(G, \chi(G) + 1)$ is connected for all the H-free graphs G if and only if H is an induced subgraph of $P_3 + P_1$.*

Sketch of the proof. Their result is based on the results mentioned above and:

- The description of a simple collection $(4K_1, C_4, claw)$-free[21] graphs G such that $G(G, \frac{3}{2}\chi(G))$ is not connected (see Figure 3.10).
- A recoloring algorithm for (P_3 + P_1)-free graphs based on the simple structure of the anti-components of these graphs.

□

As far as I know, in the case of the $2K_2$-free graphs, we do not know if the following holds:

Conjecture 31. *For every c 2, there exists a $2K_2$-free graph such that, $G(G, \chi(G) + c)$ is not connected.*

The example for c = 1 in [80] has been proven to be exact using a computer assisted proof. Note that the question is also open for P_5-free graphs. The authors of [12] also started the study of similar questions when two graphs are forbidden.

We would like to end this part on H-free graphs with two questions. The rst is related to Question 31.

Question 32. *Does there exist a natural graph class G such that there exists G 2 G such that $G(G, \chi(G) + 1)$ is not connected but there exists c such that, for every G 2 G, $G(G, \chi(G) + c)$ is connected?*

In most of the cases, when $G(G, \chi(G) + 1)$ is not connected, it is because G admits a frozen coloring. But it is not always the case, for instance for every r 1, $G(C_{3r+2}, 3)$ is not connected but does not admit any frozen coloring. When the number of colors increases, it is easy to nd graphs whose con guration graph is not connected with no frozen coloring but they usually contain frozen subgraphs. This raises the following natural question.

Question 33 (Belavadi et al. [12])**.** *Let k 4. Does there exist a graph G whose k-con guration graph is not connected but G does not contain an induced subgraph with a frozen k-coloring.*

A positive answer to that question would give more examples of bad cases for recon guration which may be helpful to construct graphs whose con guration graphs are not connected or have large diameter.

3.4 Recoloring with the lens of maximum degree: Random sampling

3.4.1 Random sampling

In this part, we will brie y discuss relations between graph recoloring and Glauber dynamics. Even if I tried to make this section as self-contained as

[21] A *claw* is a star with three leaves, that is $K_{1;3}$.

possible, the reader with no prior knowledge on Markov chains can safely skip this subsection without any impact on his/her understanding of the rest of the manuscript. (For a very rapid introduction to Markov chains, one is referred to Chapter 2).

Let G be a graph and c_0 be a (proper) k-coloring of G. We de ne a Markov chain $X_0, X_1, \ldots, X_r$ over the set of k-colorings of G by de ning the coloring X_{i+1} from X_i as follows:

- Select a vertex v at random and a color c at random.
- Recolor v with color c in the coloring X_i if the resulting coloring is proper.

Note that this Markov chain is *lazy*, which means that, for any coloring α, if $X_i = \alpha$ for some i, the probability that the coloring remains α at step $i + 1$ is positive. Indeed, if the pair (v, c) that is drawn at random satis es that $\alpha(v) = c$ then the coloring remains the same. It is a well-known fact that, for lazy Markov chains, ergodicity and irreducibility are the same. In other words, this Markov chain is ergodic if and only if it is irreducible.

This Markov chain has been widely studied in the random sampling community as well as in statistical physics[22] under the name of *Glauber dynamics for graph colorings* (abbreviated into Glauber dynamics in the rest of the thesis). As we have seen in Section 3.2, an important line of research in graph recoloring has been devoted to studying con guration graphs when the number of colors is related to the degeneracy. The main parameter of interest considered in random sampling has been the maximum degree of the graph.

The aim of this chapter is not devoted to giving a full overview of the results related to the ergodicity or the mixing time of Glauber dynamics. The reader with a background on graph theory interested in Glauber dynamics can for instance read the excellent introduction of Marc Heinrich to the topic in his PhD [91].

Mixing time of Glauber dynamic. By Theorem 7, as long as k is at least the degeneracy plus 2, the con guration graph is connected and then the chain is indeed ergodic. One can nevertheless easily prove that the mixing time of the Glauber dynamic can be exponential when the number of colors is bounded, even on stars (which have degeneracy one).

Lemma 34. *The Glauber dynamics has a non-polynomial mixing time on stars when* $k = o(\frac{n}{\log n})$.

[22] The recon guration with Kempe chains has also been studied a lot in statistical physics, in particular the so-called WSK algorithm (see [122] for instance for more information about it).

Sketch of the proof. Consider a star with n 1 leaves colored with k + 1 colors colored initially at random. One can wonder how many steps are needed to change the color of the center of the star (in order to sample a coloring at random, we should be able to change the color of the center of the star). First, the center of the star has to be selected (which happens with probability $1/n$ at each step). But, above all, the selected color c for the center has to be available (i.e. it should not appear in the neighborhood of the center). One can easily prove that the probability that no leaf is colored c is $(1 \quad \frac{1}{k})^n$. A simple calculation ensures that this probability is smaller than $\frac{1}{\text{poly}n}$ when $k = o(\frac{n}{\log n})$. □

So the mixing time cannot be polynomial when k is too small (even on 1-degenerate graphs). But how much can it decrease when k increases? Let us explain why $(n \log n)$ is usually a lower bound on the mixing time.

We consider the following problem. In each cereal box, there is a small gift called a coupon. Assume that a collector wants to collect all the gifts in the breakfast cereal boxes and that, in total, there are n of them. How many boxes does the collector have to buy before getting all the coupons? One can prove that the total number of boxes that have to be bought by the collector in order to get all the coupons is, with high probability, $O(n \log n)$ (assuming that all the coupons appear with the same probability). To see this, one can simply remark that, if we denote by τ_i the number of steps we need to collect i coupons, we have:

$$\mathrm{E}(\tau_n) = \mathrm{E}(\sum_{i=1}^{n} \tau_i \quad \tau_{i\ 1}) = \sum_{i=1}^{n} (\mathrm{E}(\tau_i \quad \tau_{i\ 1}))$$

where τ_0 = 0. Each variable $\tau_i \quad \tau_{i\ 1}$ is a geometric random variable with success probability $(n \quad i + 1)/n$. A simple calculation ensures that this expectation is of order $\theta(n \log n)$.

Let us now explain why the coupon collector problem is related to our problem. If we start from a coloring, we (informally) need to change the colors of all the vertices of the graph to be sure that we can reach any possible coloring. So in order to be sure that we need to sample a coloring almost at random, we need to select at least once every vertex vertex, which, by the coupon collector problem happens after $(n \log n)$ steps[23]. One can then wonder what is the number of colors that ensures that the mixing time is $(n \log n)$ or at least polynomial in n. The central conjecture is the following:

Conjecture 35. *The mixing time of the Glauber dynamics of the* (+ 2)*-colorings of a graph of maximum degree is* $O(n \log n)$*.*

[23] The argument here is informal and actually slightly incorrect.

One can easily remark that, if correct, + 2 cannot be improved since (+ 1)-colorings of a clique are frozen (and then the con guration graph is not connected). Jerrum [99] and Salas and Sokal [121] independently proved that the mixing time is $O(n \log n)$ when k 2 + 1 using a path coupling argument.

Theorem 36 (Jerrum [99], Salas and Sokal [121])**.** *For every $k > 2$, the Glauber dynamics is mixing in $O(n \log n)$ steps.*

Proof. Let us rst sketch the idea of the proof for $k > 3$. We will not formally de ne all the notions needed for the proof, we will simply try to give informal (and then slightly incorrect sometimes) de nitions and intuitions.

Consider two chains X_t, Y_t which are Glauber dynamics on the same graph G with k colors. A *coupling* (X_t, Y_t) of these two Markov chains is a Markov chain such that if we consider independently the rst or the second coordinate everything that happens is as if we were running the Glauber dynamics on G with k colors. For instance, we can decide to choose a vertex v_1 at random and a color c_1 at random to de ne X_{t+1} and a vertex v_2 at random and a color c_2 at random to de ne Y_{t+1}. Each chain then indeed behaves as the Glauber dynamics.

However, we can also decide to correlate the moves of X_t with the moves of Y_t. The easiest (non independent) coupling consists in saying that, in both chains, we apply the same rule at the same time (let us call this coupling the *identity coupling*). In other words, from (X_t, Y_t), we select a vertex v at random and a color c at random and de ne (X_{t+1}, Y_{t+1}) as the pair of colorings obtained in the Glauber dynamics after applying the recoloring of v with c. The behavior of X_t and Y_t are related but if we simply look at the rst (or the second) coordinate we simply observe the classical behavior of the Glauber dynamics.

The magic result about path coupling is due to Bubley and Dyer [51] which ensures that if we can nd a path coupling which reduces (with a decent probability) the distance between X_t and Y_t when X_t and Y_t only di er on one vertex, then the Glauber dynamics is mixing in $O(n \log n)$ [24].

Let us now explain how we can use this result to obtain bounds on the mixing time of the Glauber dynamics when $k > 3$ (the coupling is described on Table 3.2). Let us consider the identity coupling. Let X_t, Y_t be two colorings that di er on exactly one vertex v. Let us now evaluate the expected distance[25] between the colorings (X_{t+1}, Y_{t+1}).

- If we select a vertex w that is not in $N[v]$ then w is recolored in X_t if and only if w is recolored in Y_t. So the distance between the colorings

[24] The result of Bubley and Dyer is widely more general but I tried to make it as simple as possible.

[25] We de ne the (Hamming) *distance* between two colorings as the number of vertices on which the two vertices di er.

Pair (w, c)	3	2
$w \notin N(v), c$	$(w, c) \to 0$	$(w, c) \to 0$
$v, c \neq c_1, c_2$	$(v, c) \to 0$	$(v, c) \to 0$
$v, c = c_1, c_2$	$(v, c) \to 1$	$(v, c) \to 1$
$w \in N(v), c \neq c_1, c_2$	$(w, c) \to 0$	$(w, c) \to 0$
$w \in N(v), c = c_1$	$(w, c) \to +1$	$(w, c_2(v)) \to 0$
$w \in N(v), c \neq c_2$	$(w, c) \to +1$	$(w, c_1(v)) \to +1$

Table 3.2: Description of the coupling to obtain a polynomial mixing time for > 3 and > 2 colors. For the pair in X_t in the leftmost column, the other columns give the pair selected in Y_t for the coupling (the term at the right of the $\to$ describes the evolution of the distance for that pair). We denote by c_1 and c_2 the colors of v on X_t and Y_t respectively.

is not modified in these cases.

- If we select v, then the distance decreases as long as we select a color that does not appear in $N(u)$ in X_t and Y_t. (Note that these colors are the same since X_t and Y_t only differ on v). Since there are at most colors in $N(v)$ and $k > 3$, the number of cases where the distance decreases is at least 2 + 1. Such an event is called a *success*.

- If we select a vertex w in $N(v)$ then the distance between X_{t+1} and Y_{t+1} might increase by one. The distance can increase when w is recolored with the color c of v in X_t or with the color c' of v in Y_t. (Indeed, in the first case for instance, w might be recolored with c in Y_{t+1} while w will remain of its initial color in X_{t+1}). Such an event is called a *fail* and there are at most 2 pairs (w, c) that might lead to a failure.

So in total, the number of pairs leading to a success is larger than the number of failures. So the expected distance between the colorings decreases (and one can easily compute by which amount). And the result of Bubley and Dyer ensures that the chain is mixing in time $O(n \log n)$ when $k > 3k$.

In order to prove the result when $k > 2$ instead of 3 one has to choose slightly more carefully the coupling between the two Markov chains. We perform the identity coupling except that we flip two types of recolorings.

- The recoloring, in X_t, of a neighbor w of v with the color of v in X_t is coupled, in Y_t, with the recoloring of w with the color of v in Y_t.

- The recoloring, in X_t, of a neighbor w of v with the color of v in Y_t is coupled, in Y_t, with the recoloring of w with the color of v in X_t.

In all other cases, X_t and Y_t attempt to modify the same vertex to the same color (that is, we follow the identity coupling).

The number of successes is, following the previous analysis, at least +1 since $k > 2$. The magic comes from the fact that the number of fails is at most . Indeed, the identity coupling has a drawback: when we try to recolor in X_t with the color c of v in Y_t we perform the same recoloring in Y_t. The color of w in X_{t+1} might change while we know that the color of w will not change in Y_{t+1}. So we couple an event where we know that there is no change with an event where there might be a change. And there are such events where there might be a change in X_t but not in Y_t. And symmetrically there are events where there might be a change in Y_t but there is no change in X_t.

The new coupling described above permits to couple the events where there is no change in X_t with events where there is no change in Y_t (rst item). And the events where there might be a change in X_t or Y_t are coupled together (second item). So the number of fails is at most and, which completes the proof. □

The ratio of 2 has been improved by Vigoda [132] who proved that the chain is rapidly mixing as long as $k > \frac{11}{6}$. The proof is widely more complicated and in particular consists in adapting the Glauber dynamics by recoloring short Kempe chains (and proving that this new Markov chain has a similar mixing time). The question of improving this bound remained open for almost 20 years until Chen et al [58] proved that the chain is rapidly mixing if the number of colors is at most ($\frac{11}{6}$ ϵ) for some (small) positive real ϵ. Conjecture 35 has been studied in various graph classes such as bounded treewidth graphs [92] (see e.g. [91] for more details).

3.4.2 The special case $k = \Delta + 1$

When k = + 1, the k-con guration graph is not necessarily connected as we already mentioned, for instance for cliques (the coloring of a clique being frozen). The same holds even restricted to triangle-free graphs [22]. In all these examples, the fact that the con guration graph is not connected is ensured by the existence of frozen colorings.

But, in general, the k-con guration graph might not be connected even if there is no frozen coloring. For instance, $G(C_5, 3)$ is not connected even if C_5 has no frozen 3-coloring[26]. One can for instance remark that it is not possible to transform the coloring 12312 into 13232.

Surprisingly, Feghali et al. proved that, as long as 3, it is always possible to transform a non-frozen (+1)-coloring into any other non-frozen (+ 1)-coloring. Note that the same does not hold for = 2 for instance for C_5 [27]. So the case of cycles described above is really speci c. More

[26] We will prove in Section 3.6 that the con guration graph has two connected components.

[27] The case of = 2 is often particular in graph (re)coloring; see e.g. Gallai's theorem.

formally, Feghali, Johnson and Paulusma proved in [78] that, for any 3, one can transform any non-frozen (+ 1)-coloring of a connected graph into any other in at most $O(n^2)$ steps. In other words, they prove that, for every connected graph G, the graph $G(G,$ + 1) consists of a (possibly empty) connected component containing all the non-frozen colorings and isolated vertices.

It naturally raises several questions:

1. What is the proportion of frozen colorings with respect to the total number of colorings? In particular, if we sample a coloring at random in the connected component of $G(G,$ + 1) composed of all the non-frozen colorings using the Glauber dynamics, do we sample a coloring at random in a set of colorings containing almost all the colorings?
2. Is the upper bound of $O(n^2)$ proposed in [78] tight?
3. What is the mixing time of the non-frozen component of $G(G,$ + 1) ?

With Marthe Bonamy and Guillem Perarnau, we characterize the maximum number of frozen colorings with respect to the total number of colorings:

Theorem 37 (Bonamy, Bousquet, Perarnau [22])**.** *Let G be a graph and α be a (+ 1)-coloring of G. Then*

$$\mathrm{P}(\alpha \textit{ is frozen}) \quad (9/10)^{n=(\ +1)\ 1}.$$

Sketch of the proof. The very high level idea of the proof is not that complicated (even if the proof techniques are quite involved). Let α be a partial coloring of a graph H and x be a vertex such that no vertex of $N(x)$ is colored in α. We say that a partial coloring of H is *frugal* if no vertex has two neighbors colored the same in α. Note that any partial coloring of a frozen (+ 1)-coloring of a graph G is frugal.

The main step of the proof consists in showing that the number of extensions of a frugal partial coloring α to $N(x)$ which are still frugal colorings is at most $\frac{9}{10}$ of the number of possible extensions of α to $N(x)$. The proof is based on a case distinction on the number of colors that appear in the second neighborhood of x in the partial coloring.

When we have this lemma, we can easily conclude by remarking that there exists a color class containing $r := n/($ + 1) vertices $v_1, \ldots, v_r$. We can then apply iteratively the result on $H_i = V \setminus N[\bigcup_{j\ i} v_i]$ to get the result. □

So if the number of vertices is large enough compared to the , the number of frozen colorings is exponentially small which answers the rst

question. Note that the division by in the exponent is indeed needed since all the colorings of a clique are frozen.

We also partially answered the second question with Laurent Feuilloley, Marc Heinrich and Mikael Rabie by proving that the quadratic upper bound obtained by Feghali, Johnson and Paulusma in [78] can actually be improved into (n) when the maximum degree is bounded. Namely, we proved the following:

Theorem 38 (Bousquet, Feuilloley, Heinrich, Rabie [39])**.** *Let G be a connected graph with 3. The diameter of the non-frozen component of $G(G,$ + 1) is at most $O($ ^{c} $n)$, where c is a constant.*

Sketch of the proof. One of the key ingredients of the proof consists in proving that, if we have a non-frozen vertex, then we can "duplicate" it. More formally, we can prove that, if x is non frozen in a (+ 1)-coloring α of G and y a vertex at distance 7 from x, then we can obtain, by only recoloring vertices of $B(x, 6)$, a coloring where both x and y are non-frozen. Let us call this property ($\star$). Note that since we only recolor vertices in $B(x, 6)$, the number of steps needed to obtain this coloring is bounded (in terms of)[28].

Let us explain how we can use the property ($\star$) to prove Theorem 38. Let α, β be two non-frozen (+ 1)-colorings of G.

- We rst select a maximal independent set S at distance c (for a large enough constant c) containing a non-frozen vertex x.
- Using ($\star$), we can unfreeze all the vertices of S one after another by duplicating non-frozen vertices and propagating them towards other vertices of S. We can do so by remarking that, if we recolor a vertex then its frozen neighbors become non-frozen. Indeed, since $k =$ + 1, a vertex is frozen if all the colors appear exactly once in its closed neighborhood. So, if a neighbor of a frozen vertex is recolored, then it becomes non-frozen. So if we have a path of frozen vertices, we can \propagate" the vertex that is not frozen along the path.
- We can prove (using a variant of Theorem 7) that $\alpha_{|V \cap B(S,7)}$ can be transformed in $\beta_{|V \cap B(S,7)}$ in O $(|V \cap B(S, 7)|)$ steps. Note however that this sequence cannot immediately be used in V since a recoloring might be in con ict with a vertex of $B(S, 7)$. However, we can prove (again using ($\star$)) that this recoloring sequence can be extended to V in O (n) steps. In other words, we can transform α into β' where β' matches with β on $V \cap B(S, 7)$ in O (n) steps.

[28] Actually it can even be obtained with a constant number of single vertex recolorings.

- We nally prove that we can independently recolor each ball[29] $B(x, 7)$ to reach the target coloring β for every x 2 S.

□

As for Theorem 25, the diameter here depends on (exponentially in that case) which makes it less interesting when is not a constant. It could be interesting to see if the dependency on can be avoided[30].

Question 39. *Let G be a connected graph of maximum 3. Is it true that:*

- *Every connected component of G(G, + 1) has diameter O(poly( )n)?*
- *Every connected component of G(G, + 1) has diameter at most Cn, for a constant C independent of ?*

Let us conclude this section with the following question that is a natural generalization of Conjecture 35.

Question 40. *Does every connected component of G(G, + 1) has polynomial mixing time?*

3.4.3 Above $\Delta + 1$ colors

When k +2, the con guration graph is always connected and the mixing time is conjectured to be $O(n \log n)$. One can wonder what is the diameter of the con guration graph when k + 2. Is it necessarily linear? One can easily prove that the diameter is $O(\ n)$. Actually, we can prove that the con guration diameter is $O(\chi_g(G)n)$ when $\chi_g(G)$ is the *grundy number* of G as long as k $\chi_g(G)$ + 1. The *greedy number* of G for the order $v_1, \ldots, v_n$ is the largest possible color used in the greedy coloring of G for the order $v_1, \ldots, v_n$ [31]. Note that, if we consider an optimal coloring of a graph G = (V, E) with color classes $V_1, \ldots, V_{\chi(G)}$ then the greedy number for the order $V_1, \ldots, V_{\chi(G)}$ is $\chi(G)$. In other words, there exists an order that gives an optimal coloring. The *grundy number of G* is the maximum, over all the orders O, of the greedy number of G for O. In other words, the grundy number is the worst possible number of colors used in a greedy coloring. Note that $\chi_g(G)$ + 1 for every graph G (but it can be arbitrarily smaller in some cases).

With Marthe Bonamy, we proved that the following holds:

[29] Actually we sometimes have to recolor a slightly larger set but, since c has been chosen large enough, the balls remain independent.

[30] Note that in [78], the diameter is quadratic but does not depend on .

[31] The *greedy coloring* of G consists for an order O in giving to every vertex the smallest possible color when it is considered.

Lemma 41 (Bonamy, Bousquet [19]). *If* $k \quad \chi_g(G) + 1$ *then* $\mathrm{diam}(G,k)$ $2\chi \quad n$.

Sketch of the proof. Let α, β be two colorings. The first step consists in considering vertices one by one to recolor them in α and β with a color (possibly different) in $\{1,\dots,\chi_g(G)\}$. This is indeed possible by definition of grundy number. Note that, at the end of this step, the color $\chi_g(G) + 1$ is not used. By abuse of notation, we still denote by α and β the resulting colorings.

The second step consists in considering the color classes of β one by one to obtain a coloring α_i such that the vertices of β colored j with $j \quad i$ are also colored with j in α_i. To transform $\alpha_{i\ 1}$ into α_i, we recolor all the vertices colored i in $\alpha_{i\ 1}$ with $\chi_g(G) + 1$, then recolor all the vertices colored i in β with i and finally change the color of the remaining vertices colored $\chi_g(G) + 1$ with another color, which completes the proof. □

One can wonder if the dependency on χ can be removed. Cambie et al. recently answered in the positive in [52] with a very simple argument:

Lemma 42 (Cambie, Cames van Batenburg, Cranston [52]). *For every graph* G, $\mathrm{diam}(G, \quad (G) + 2) \quad 2n$.
Even stronger, the diameter of $G(G,\mathbf{L})$ *is at most* $2n$ *as long as* $|L_v| \quad d(v) + 2$ *for every vertex* v.

Sketch of the proof. Let α, β be two list colorings of $(G,\mathbf{L})$. There exists in β a color class c whose size is not smaller than in α. We recolor all the vertices colored with c in α with another color and then color all the vertices colored c in β with c. The conclusion follows by induction. □

They then proved a finer upper bound depending on $\nu(G)$ where $\nu(G)$ is the maximum size of a matching (a set of pairwise extremity disjoint edges) of G.

Theorem 43 (Cambie, Cames van Batenburg, Cranston [52]). *For every graph* G *and list assignment* $\mathbf{L}$*:*

1. *If* $|L_v| \quad d(v) + 2$ *for every vertex* v *then the diameter of* $G(G,\mathbf{L})$ *is at most* $n + 2\nu(G)$.
2. *If* $|L_v| \quad 2d(v) + 1$ *for every vertex* v *then the diameter of* $G(G,\mathbf{L})$ *is at most* $n + \nu(G)$.

Sketch of the proof.
(1) If $\nu(G \quad v) = \nu(G)$ for all $v \in V$, then a classical result of Gallai ensures that G is factor-critical, i.e. G admits a matching of size $(n \quad 1)/2$. In that case $n + 2\nu(G) = 2n \quad 1$ and the conclusion follows from Lemma 42. Otherwise, there exists $v \in V$ that is in all the maximum matchings of G.

In that case, they prove that v can be colored similarly in both colorings in at most 3 steps[32]. Deleting v reduces n by 1 and $\nu(G)$ by 1. The proof follows by induction.

(**2**)The proof of the second point follows from the Edmonds{Gallai Decomposition theorem and a more subtle recoloring argument. □

They conjectured that the following holds:

Conjecture 44 (Cambie, Cames van Batenburg, Cranston [52])**.** *For every graph G and list assignment $\mathbf{L}$ such that jL_vj $d(v)$ + 2 then diameter G$(G, \mathbf{L})$ is at most n + $\nu(G)$.*

They show that, if true, this bound is optimal for the list version. Note however that, as far as I know, no non-trivial lower bound (larger than n) nor better upper bound is known for classical recoloring. In particular, the two following questions are open:

Question 45.
- *Is it true that the diameter of G$(G,$ (G) + 2) is at most n + $\nu(G)$?*
- *Does there exist a collection of graphs $(G_r)_{r\ 1}$ such that $\nu(G_r)$ = r and the diameter of G$(G,$ (G) + 2) is at least n + $\nu(G)$? At least n + $\epsilon\nu(G)$?*

Line graphs. The *line graph* H of a graph G is a graph whose vertex set is $E(G)$ and there is an edge between two vertices of H if the two corresponding edges in G share an endpoint. A *line graph* is a graph H that is the line graph of some graph G. It is well-known that the line graph H of a graph G of maximum degree is either or (+ 1)-colorable. It cannot be colored with less colors since edges of G incident to a vertex of maximum degree create a clique of size . And a classical argument using Kempe chains due to Vizing ensures that line graphs are (+ 1)-colorable.

One can wonder what is the minimum value k_{min} of k, as a function of , from which we can guarantee that the con guration graph of proper k-edge colorings is connected. This question has been answered by Aline Parreau, Jonathan Noel, Marc Heinrich and Alice Jo ard[33]. The coloring of the Petersen graph depicted in Figure 3.11 with 5 colors is frozen. And we can easily generalize this example to the Kneser graph $KG_{2\ 1;\ 1}$ of degree , colored with 2 1 colors such that every edge is assigned the color corresponding to the missing number in the union of the sets of its two vertices. This proves that k_{min} = 2 .

[32] That is actually true for every vertex and not simply for vertices belonging to all the maximum matchings.

[33] During a Combinatorial Recon guration workshop at Aussois. Private communication.

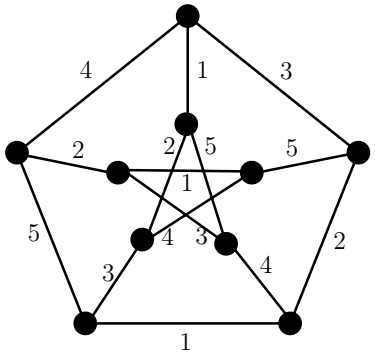

Figure 3.11: A 5-coloring of the Petersen graph that is frozen. (To see that the coloring is frozen, note that, for every vertex, the set of colors not incident to that vertex is disjoint to the one of its neighbors.

3.5 Recoloring with the lens of distributed computing: local and parallel recoloring

3.5.1 Linear diameter and locality

Most of the proof techniques described so far provide quadratic or exponential upper bounds on the diameter of the con guration graph. These proofs are typically based on vertex identi cation techniques that consist in identifying together vertices or groups of vertices (e.g. Theorems 20 or 21) or are based on vertex deletion techniques (e.g. Theorems 7, 23 or 13). The problem of the identi cation technique is that a linear number of vertices might be merged together and then a single vertex recoloring in the contracted graph might already imply a linear number of recolorings in the original one. This usually prevents us from obtaining a linear diameter. For the vertex deletion technique, in order to get a linear diameter, we have to prove that the new vertex can be added in the existing partial sequence by simply recoloring this additional vertex (or vertices) with a constant number of recolorings. This can be done in some particular cases (e.g. Theorem 23 or 29) but it often needs a very careful analysis and, at the end, we often have the feeling that the assumption is much stronger than what it should be in practice to guarantee a linear diameter.

If we see these techniques (and in particular the vertex identi cation technique) slightly di erently and look at the recon guration sequence between two colorings α and β, it means that when we decide to change the color of a (contracted) vertex, then we might have to change the color of many vertices in the original graph that are possibly far apart in the graph. For instance in the proof of Theorem 20, we nally end up on a clique where vertices correspond (roughly) to maximal independent sets in the original graph ! This fact is highly non-desirable if we want to nd short recon g-

uration sequences and is probably useless when we have some slack on the number of colors. Indeed, since we have some slack we should be able to only recolor vertices locally. Again in the case of chordal graph, we should be able to \cut" the maximal independent set and only recolor a small part of it when the number of colors is large enough.

In order to obtain linear upper bounds on the diameter of con guration graphs, we need to nd other techniques. The goal of this section is to discuss a bit further the two techniques already described in this manuscript, namely: Theorems 25 and 38 in order to explain their similarities and di erences. In my opinion, both techniques can be seen as local transformations. But how can we de ne the locality of a transformation? It is extremely hard to nd a good answer to that question. Very informally speaking, we would like to de ne it as: *the recoloring of some vertex v only has a local impact and does not globally a ect the graph*". Or in a converse way *the recoloring sequence for v will only depend on some ball of small radius centered on v and not on the whole graph G*". Another way of de ning it would be that these techniques can be adapted in the LOCAL model of distributed computing. We will see that the techniques I introduced satisfy (in a sense) both statements.

3.5.2 Parallel recoloring and sliding buffer technique

Recoloring in parallel and dependencies between recoloring steps. Consider a recoloring instance where the source and the target colorings di er only on an independent set. In this instance, any sequence created by iteratively assigning its target color to a vertex that does not have it already, is valid. The order can be chosen arbitrarily, because there is no dependency between the color changes: a vertex does not need one of its neighbors to rst change its color in order to be able to change its own.

A way to capture this absence of dependency is to note that we can parallelize the recoloring: we can just take all the vertices that do not have their target colors and recolor them in parallel[34]. We can now de ne a parallel recoloring sequence as a sequence of recolorings of independent sets while keeping a proper coloring all along the transformation. From such a parallel sequence, it is easy to derive a sequential recoloring sequence: decompose any parallel step by performing all the individual vertex recolorings one after the other.

Now, one might wonder if, in general, allowing parallel recoloring dramatically reduces the number of steps or not. Let us consider two examples with very di erent behaviors.

Consider the case of paths colored with 3 colors (note that = 2 for paths). And consider a source coloring of the form 1,2,3,1,2,3... and a target

[34] Recoloring all the vertices of some desirable independent set \at the same time" was already used, e.g. in [75].

coloring of the form 2,3,1,2,3,1... In this case, even if we allow parallelization, at step i, only the vertices at distance at most i from an extremity can change color. Therefore, the recoloring must be very sequential, and we will use at least (n) parallel steps. In other words, there are strong dependencies between color changes[35]. This example can be easily generalized to any power of paths to any possible number of colors. One can easily remark that the r-th power of a path admits a $(2r + 1)$-coloring where all the vertices but the first or last ones are frozen. Indeed one can remark, by simply coloring x_i with color i mod $2r + 1$, we obtain a proper coloring where all the vertices but the r first and last are frozen. So in order to simply permute the colors for this coloring, we need to modify the colors of vertices from the left (or from the right) before being able to recolor vertices in the middle of the graph. So in this example, we cannot expect to only modify colors locally to reach the target coloring. This example is interesting since $G_{r;n}$ (i) is a graph of maximum degree $2r$ with a non-frozen $(2r + 1)$-coloring, (ii) is a chordal graph of clique number $r + 1$ colored with $2r + 1$ colors. So there are some cases where we cannot expect a transformation with a bounded number of parallel recolorings even for chordal graphs where the number of colors is essentially twice the degeneracy[36].

As a second example, consider graphs of maximum degree with at least 2 + 2 colors. Let us explain how we can reach a (+ 1)-coloring in $O($) parallel recolorings. One can first, for every i + 1, recolor in parallel all the vertices colored i with a color in $\{ + 2, \ldots, 2 + 2\}$. After + 1 parallel recolorings, there is no vertex colored with a color in $\{1, \ldots, + 1\}$ anymore. And we can then finally obtain any (+ 1)-coloring with + 1 additional steps.

This raises the following natural question.

Question 46. *When recoloring is possible, how many* parallel steps *are needed?*

Note that when the number of parallel steps is bounded then we have a linear reconfiguration sequence. One can argue that, in that setting, we recolor a set that is possibly a maximal independent set and then that there is no difference with Theorem 20. But there is one big difference: we do not ask here the recolorings to be synchronized. Every vertex of the independent set makes its own choice that is completely independent from the other choices. In the sequence described above, the vertices initially colored i do not care of the colors of the other vertices colored i.

Most of the cases, the answer to Question 46 is large. Indeed, when we have a coloring that is almost frozen (like in the path example above but also

[35] Actually in that case, one can prove that a recoloring sequence needs (n^2) single vertex recolorings and that we can recolor it with (n) parallel steps.

[36] Even if Theorem 25 ensures that there exists a linear transformation in that case.

for (+ 1)-colorings where almost all vertices might be frozen, a bounded number of parallel steps cannot be reached. However, in the second case, we prove that a bounded number of parallel steps is possible after a *linear pre-processing*[37] while it is not possible for paths. In other words, we ask the following question:

Question 47. *When recoloring is possible, can we bound the number of* parallel steps *after a pre-processing of linear size?*

The drawback of that technique is that we need to nd a way to properly de ne an ordering on the vertices \from left to right" to make this method work. Moreover, the technicity of the proof is higher since we need to keep properties on the coloring of the bu er. The advantage though is that we do not need any pre-processing phase.

3.5.3 Distributed recoloring

Let us now formally de ne what we mean by distributed computing. In distributing recoloring (in the LOCAL model), every vertex v outputs a sequence $c_1, \dots, c_r$ of colors, called the *schedule of* v, where c_1 is the initial color of v, c_r is the target color of v and, at each step i, the set of vertices X such that c_i 6 c_{i+1} should be an independent set of the graph. While in an optimization problem only the number of rounds is important, in distributed recoloring, the length of the schedule is also of importance. So the goal is to nd the best possible balance between the number of rounds of the algorithm and length of the schedule.

Distributed recoloring has been introduced in [28] by Bonamy et al. One can notice immediately that, with the classical de nition of recoloring, we cannot expect that much in terms of number of rounds. Indeed, if we consider a coloring that is *locally frozen* in the sense that all the vertices but only a small number are frozen, one cannot expect a LOCAL algorithm with a sublinear number of rounds (for instance in the case of the power the path discussed in Section 3.5.1). So we have to relax a bit the conditions on the number of colors or on the colorings. The authors decide in [28] to consider the following variant: we are given two k-colorings of a graph G and a set of ℓ additional colors (not used in α nor β), how many steps are needed to transform α into β using the colors between 1 and $k + \ell$? We call this problem by the (k, ℓ)-*distributed recoloring problem*. We will see that this condition of additional color can actually be drastically weakened in some cases since we can prove that we only need to assume that our colorings are locally non-frozen (in the sense that in every ball must contain a non-frozen vertex).

The key results of [28] are the following:

[37] That is a linear number of single vertex recolorings.

Theorem 48 (Bonamy, Ouvrard, Rabie, Suomela, Uitto [28]). *The* $(3,1)$*-distributed recoloring problem on trees can be solved in* $O(\log n)$ *steps with a schedule of length* $O(1)$*.*

Sketch of the proof. The key ingredient of the proof is an algorithm that solves the following sub-problem in $O(\log n)$ rounds: given a tree, the goal is to find an independent set I such that the removal of I splits the tree in components of size 1 or 2. When this algorithm is given, we can simply give to all the vertices of I the additional color at the first round. Then, we can simply recolor locally and independently each connected component with the 3 initial colors and finally give to the vertices of I their final color. □

A natural extension of trees are chordal graphs. One can then naturally wonder if these results can be extended to chordal graphs. We answer this question positively in [38] by proving that the following holds:

Theorem 49 (Bousquet, Feuilloley, Heinrich, Rabie [38]). *Let* G *be a chordal graph and* α, β *be two proper* k*-colorings of* G*. It is possible to find a schedule of length* $n^{O(\log \ )}$ *to transform* α *into* β *in* $O(\omega^{2 \ 2} \log n)$ *rounds in the LOCAL model using at most:*

- c *additional colors, with* $c = \omega \quad k + 4$*, if* $k \quad \omega + 2$*,*
- 1 *additional color if* $k \quad \omega + 3$*.*

One can remark that for trees it gives a slightly weaker result since (i) it only ensures that one additional color is needed when we have 4-coloring of a tree (instead of a 3-coloring as in Theorem 48 and (ii) we have a schedule whose length depends on . However, the graph class covered by this result is much wider than the one covered by Theorem 48.

We will not cover the proof technique since we would need to introduce a bit too much material from distributed algorithms. However, the key ingredient of this proof consists in applying Theorem 25 in parallel to several independent sections of the graph. We obtain these different sections using a \Rake and Compress" method on the tree decomposition of the chordal graph. And in order to be sure that we can apply the machinery of Theorem 25, we have to use Kempe chains that are appropriately cut using additional colors if they are too long (to be sure that the recolorings remain local).

The second main positive result of [28] consists in giving $(3,1)$-distributed recoloring algorithm for graphs of maximum degree 3 (with sequences and schedule of size $O(\text{polylog}(n))$). Note that it consists of a graph of maximum degree with a total number of colors equal to + 1. One can then naturally wonder if Theorem 38 can be extended into a distributed algorithm. The answer is positive. Namely we showed that the following (informally stated theorem) holds:

Theorem 50 (Bousquet, Feuilloley, Heinrich, Rabie [38]). *Let d be a constant. Consider two colorings where in every ball of radius d there is a non-frozen vertex. Suppose we are given a set of selected vertices that are at distance at least d and at most $2d$ one from another. Then, for every vertex, we can compute its recoloring schedule by looking only at its neighborhood at distance $O(d)$ as long as the total number of colors is at least + 1.*

3.6 Algorithmic aspects graph recoloring

In this section, we will overview the most important algorithmic results related to graph recoloring. In a series of articles, Cereceda together with Bonsma, van den Heuvel and Johnson proved that the following holds:

Theorem 51. *The following holds:*

- 2-MIXING *and* 2-REACHABILITY *are in P.*
- 3-REACHABILITY *is in P [55] while* 3-MIXING *is coNP-complete [54].*
- 3-MIXING *is always negative for* 3*-chromatic graphs [54].*
- *For every k* 4, k-REACHABILITY *is PSPACE-complete [31].*

Let us brie y sketch the proof that 3-REACHABILITY can be decided in polynomial time. Let α be a 3-coloring of a graph G. Given a cycle C of G (together with a cyclic ordering of the vertices), we can assign a weight to every (oriented) edge uv of C which is $\alpha(v)$ $\alpha(u)$ mod 3. Note that the weight of an edge is either +1 or 1. The *winding number* of a cycle is the sum of the weights of C divided by 3. (One can easily remark that the winding number is an integer). One can remark that the following holds:

Lemma 52. *Any recoloring of a vertex of C does not modify the winding number.*

Sketch of the proof. The avor of the proof is similar to the one of Theorem 10. If we can modify the color of a vertex v, then the two neighbors of v on C are colored the same. So the two edges incident to v are weighted +1 and 1. The modi cation of the color of v simply replaces the pair of weights (+1, 1) by (1, +1) (or the converse). □

So, if there is a transformation between α and β then all the cycles must have the same winding number in α and β. This necessary condition is however not su cient.

A vertex v is *xed* (for α) if, for every coloring in the connected component of α in $G(G, 3)$, v is colored $\alpha(v)$ [38]. They proved that one can nd

[38] Fixed vertices are for instance vertices that belong to *xed cycles*, that are cycles of length 0 mod 3 where all the vertices *i* mod 3 are colored *i*.

in polynomial time all the xed vertices. They prove that xed vertices are either xed vertices of cycles or vertices that belong to frozen paths (of the type 123123...) between two xed vertices. And they proved that the "winding number" between xed vertices (de ned as for cycles) cannot be modi ed (with a proof following the scheme of Lemma 52).

They nally prove that the following holds which is enough to conclude

Theorem 53. *Given two* 3*-colorings* α, β *of a graph* G*, there exists a transformation between* α *and* β *if and only if:*

- *the xed vertices of* α *and* β *are the same and,*
- *the xed vertices are colored the same in* α, β *and,*
- *for every cycle and every path between xed vertices, the winding number is the same.*

Moreover, all these conditions can be checked in polynomial time.

We will not give the hardness proofs of Theorem 51 but let us explain why 3-MIXING belongs to co-NP. Assume that $G(G, 3)$ is not connected. Then there exist two colorings α and β that do not belong to the same component of $G(G, 3)$. And Theorem 53 ensures that if we are given these colorings, we can determine in polynomial time that they are not in the same connected component of $G(G, 3)$. So if an oracle can guess α, β we are done and then 3-MIXING is in co-NP.

Note that Lemma 52 is a very nice tool to decide the existence of a transformation. We have said before that $G(C_5, 3)$ is not connected. This can be seen as an easy corollary of Lemma 52. Indeed, one can remark that the winding number of 12312 is 1 while the winding number of 13213 has winding number 1. More generally, we can derive the number of connected components of $G(C_n, 3)$ quite easily.

Note that if we consider the list coloring problem instead, the LIST REACHABILITY problem becomes PSPACE-complete even for $k = 3$ [31].

Before going into results on restricted graph classes, let us nish this part with the following natural question left opened in several articles for more than 15 years:

Question 54 (Cereceda, van den Heuvel, Johnson [54])**.** *Is* k-MIXING *PSPACE-complete for* $k \geq 4$ *?*

Even if this question is still open, I recently proved with a very simple proof that the problem is NP-hard.

Theorem 55 (Bousquet [35])**.** *For every* $k \geq 4$*,* k-MIXING *is co-NP-hard.*

Sketch of the proof. The main idea of the proof of Theorem 55 consists in considering the 3-To-2 problem, asking whether, giving a 3-coloring of a bipartite graph if it can be transformed into some 2-coloring. Since 3-colorable graphs are never 3-mixing by Theorem 51 and since it is easy to prove that a bipartite graph B is 3-mixing if and only if any 3-coloring can be transformed into a 2-coloring, the 3-To-2 problem is co-NP-complete

To complete the proof, we make the following reduction. Let k 4. Let us provide a reduction from 3-To-2 to k-Mixing. Consider the graph G consisting of B plus k 3 vertices X inducing a clique which are complete to B. The proof follows from the fact that G is k-mixing if and only if every 3-coloring of B can be transformed into a 2-coloring. □

Restricted graph classes. The 4-Reachability problem is known to be PSPACE-complete even for bipartite graphs and any xed constant k 4. Wrochna proved in [134] that k-Reachability also is PSPACE-complete even restricted to bounded bandwidth graphs (we will give a formal de nition of bandwidth in Chapter 4). Since graphs of bounded bandwidth have bounded pathwidth and bounded cliquewidth, k-Reachability remains PSPACE-complete even restricted to these classes.

Theorem 56 (Wrochna [134])**.** k-Reachability *is PSPACE-complete on bounded bandwidth graphs.*

Let us brie y describe the proof of Wrochna[39]. To prove Theorem 56, he proved that the so-called H-Word Reachability problem is PSPACE-complete. He obtained the hardness of this problem with a reduction from the hardness of 2-balanced symmetric string rewriting. Since then, H-Word Reachability became one of the main tools used to prove hardness for recon guration problems.

Let be an alphabet and H be a directed graph (possibly with self-loops and digons) on vertex set H. We say that W is an H-word if, for every pair of consecutive letters ab of W, a ! b is an arc of H. Two H-words are *adjacent* if they di er on exactly one letter.

H-word Reachability
Input: Two H-words W_1, W_2 of the same length.
Output: YES if and only if there exists a recon guration sequence of H-words from W_1 to W_2 such that every pair of consecutive H-words in the sequence are adjacent.

The main part of the proof Theorem 56 consists in proving that the following holds:

[39] We will again discuss a similar proof in the next chapter for Independent Set Reconfiguration.

Theorem 57 (Wrochna [134]). *There is a digraph H for which H-Word Reachability is PSPACE-complete.*

Note that H is universal and then can be seen as a bounded graph (even if no bound on its size is provided in the proof). The following questions are, as far as I know, open:

Question 58.
- *Can we find an explicit upper bound on the size of H such that H-Word Reachability is PSPACE-complete?*
- *Is H-Word Reachability decidable in polynomial time when $|H| = 3$?*

The second part of the proof of Theorem 56 consists in proving that k-Palette List Reachability is PSPACE-complete even on a very restricted class of graphs[40]. Using this fact, Wrochna proved that k-Reachability is PSPACE-complete. To prove the latter point, one can easily remark that, if we create a new clique K of size k such that its i-th vertex k_i is colored with i then we can easily force the colors of a vertex u to belong to $S \subseteq \{1, \dots, k\}$ by simply connecting u to each vertex k_i such that $i \notin S$. If we create such a clique for every vertex, one can easily remark that we do not increase by more than k the pathwidth of G (and the same holds for bandwidth).

Let us complete this part with some additional algorithmic results on graph recoloring. On the positive side, Hanataka, Ito and Zhou proved in [89] that k-Reachability is polynomial on split graphs and trivially perfect graphs. They also show that this result cannot be extended to chordal graphs since deciding k-Reachability is PSPACE-complete even on chordal graphs of bounded clique number [87].

Wrochna [134] gave an FPT algorithm for k-Reachability (and even for more general problems) parameterized by k plus the tree-depth of a graph. Hatanaka et al. [88] also proved that List Coloring Reachability remains PSPACE-complete even for threshold graphs, whose modular-width is bounded. Ito, Kaminski, Demaine proved in [97] that List Coloring Reachability also is PSPACE-complete even restricted to planar graphs of maximum degree 3 with 6 colors.

The intriguing case of interval graphs. One particular subclass of chordal graphs received considerable attention in the literature: interval graphs. Indeed, if we consider two $(\omega + 1)$-colorings, then there exists a transformation between them by Theorem 7 (since they are $\omega - 1$ degenerate). On the other hand, no coloring exists if $k < \omega$. Despite all our efforts, we were not able to solve the following question:

[40] In an instance of k-Palette List Reachability, the vertices have constraints on their colors given by lists which are subsets of $\{1, \dots, k\}$.

Question 59. *Can* ω-REACHABILITY *be decided in polynomial time on interval graphs?*

One can naturally remark that cliques on ω vertices are completely frozen. So if the two colorings do not agree on cliques of size ω then we can answer negatively. So we can assume that cliques of size ω are colored the same. If we decide to remove vertices of these clique, we obtain an instance of the list reachability problem to the case where we want to nd a recoloring sequence between to ω colorings in an interval graph of clique number at most ω 1 where list constraints are "decreasing" in the sense that if a color c if forbidden for a vertex v then all the vertices at its left (or right) also have this constraint.

A particular case of that question has been solved by Bonsma and Paulusma in [34] who proved that if the interval graph is (ω 1)-connected then the question can be decided in polynomial time[41].

3.7 A last application of graph recoloring

Marcin Wrochna proposed the following result using Theorem 53.

Theorem 60 (Wrochna)**.** *Every graph without any cycle of length* 0 *mod* 3 *(non-necessarily induced) is* 3*-colorable.*

Sketch of the proof. Let us prove it by induction on the number of edges. If there is no edge, the conclusion indeed holds. Let uv be an edge of G. When deleting uv, the resulting graph admits a 3-coloring α by induction. Let X_1, X_2, X_3 be the vertices of V colored respectively 1, 2 and 3 in α. Let β be the 3-coloring where the vertices of X_1 are colored 2, the vertices of X_2 are colored 3 and the vertices of X_3 are colored 1.

Now, we apply Theorem 53 to prove that there exists a recon guration sequence from α to β in $G^{\emptyset}$ = G n uv. If there is no recoloring from α to β, either there is a cycle C that is frozen or there is a cycle C which does not admit the same winding number in α and β. A frozen cycle should have length 0 mod 3, which is impossible. And since we just took the permutation 1 *!* 2 *!* 3 *!* 1 on the colors, the winding number of every cycle has not been modi ed.

So there exists a recon guration sequence from α to β. Since u, v have the same colors in α and β but these colors are not the same and only one vertex is recolored at each step, there exists a step where u and v are not colored the same. Let us denote by γ this coloring. Since u, v are not colored the same in γ, the addition of the edge uv still gives a proper coloring of G, which completes the proof. □

[41] Their result actually holds more generally on chordal graphs.

One can wonder if this result can be extended to triangle-free graphs with no induced cycles of length 0 mod 3. Bonamy et al. proved in [23] that these graphs are c-colorable for some constant c. One can wonder if they are 3-colorable. The winding number assumption in the proof still holds. So the only problem that might happen is that the deletion of any edge might create an induced cycle of length 0 mod 3, which might happen as observed by Wrochna. So the following is still open:

Question 61. *Are all the graphs with no induced cycle of length* 0 *mod* 3 3*-colorable?*

3.8 Kempe chains and homomorphisms

3.8.1 A brief overview on reconfiguration via Kempe changes

Until now, we only considered single vertex recolorings. Note however that another important recoloring rule has been studied in the last decades: recoloring via Kempe changes. Let $G = (V, E)$ be a graph and α be a coloring of G. We denote by $G\ (a, b)$ the subgraph of G induced by vertices with colour a or b. A connected component of $G\ (a, b)$ is known as an (a, b)-*component* of G and α. These components are also referred to as *Kempe chains*. If a coloring β is obtained from a coloring α by exchanging the colours a and b on the vertices of an (a, b)-component of G, then β is said to have been obtained from α by a *Kempe change*. A pair of colorings are *Kempe equivalent* if one can be obtained from the other by a sequence of Kempe changes. A set of Kempe equivalent colorings is called a *Kempe class*.

Kempe changes where introduced in the nineteenth century in an attempt to prove the 4-color theorem [42]. Even if the proof ended up to be false, the idea has been successfully used many times, for instance to prove the 5-color theorem or for the proof of Vizing's theorem[43].

Meyniel rst proved that all the 5-colorings of a planar graph are all Kempe equivalent. Las Vergnas and Meyniel [107] proved that all the k-colorings of a graph of maximum degree are Kempe equivalent as long as k + 1. Mohar conjectured in [116] that the same holds as long as k . Jan van den Heuvel disproved this conjecture in [131] showing that the prism (see Figure 3.12) is a counterexample.

Feghali, Johnson and Paulusma proved in [79] that it is the only cubic counterexample to the conjecture of Mohar. We generalized this result to all values of with Bonamy Feghali and Johnson in [20].

Theorem 62 (Bonamy, Bousquet, Feghali and Johnson [20])**.** *Let* k 3 *and*

[42] All planar graphs are 4-colorable.

[43] Every graph is (+ 1)-edge colorable.

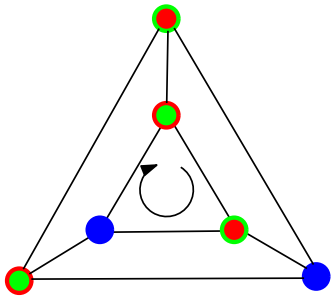

Figure 3.12: A prism. Every Kempe change modifies two colors on the inner and the outer cycles. In particular the orientation of the coloring of the cycle (Blue Red Green or Blue Green Red) is permuted in both cycles. So one cannot transform a coloring where both cycles are oriented the same into a coloring where the orientation of both cycles are reversed.

G be a graph distinct from the 3-prism. Any k-coloring can be transformed into any other via a sequence of at most $O(n^2)$ Kempe changes.

Recently, Bonamy, Delecroix and Legrand-Duchesne [24] improved this result by providing a polynomial transformation (and with a stronger statement).

One can wonder how many colors are needed to ensure the existence of a Kempe transformation. A well-known conjecture of Hadwiger states that, for every t, every K_t-minor-free graph is $(t \quad 1)$-colorable. Las Vergnas and Meyniel asked in 1981 if the $(t \quad 1)$-colorings of these graphs are all Kempe equivalent. It was recently disproved by Bonamy, Heinrich, Legrand-Duchesne and Narboni who showed in [26] that there exist K_t-minor free graphs with frozen $(\frac{3}{2} \quad \epsilon)t$-colorings for every $\epsilon > 0$. This beautiful proof is based on an extremely simple construction.

3.8.2 Graph homomorphisms

A natural generalization of colorings is graph homomorphisms. An *homomorphism* from G to H is a function that assigns to every vertex of G a vertex of H in such a way every edge uv of G is mapped on an edge of H. One can simply prove that G has an homomorphism to K_k if and only if G is k-colorable. An important line of research in the last few years has been devoted to homomorphism reconfiguration. The main problem studied is the H-Homomorphism Reconfiguration problem that, given two homomorphisms from G to H, tries to determine if it is possible to transform one into the other via a sequence of single modifications (that is we can change the function on one vertex at a time). H-Homomorphism Reconfiguration is PSPACE-complete even for $H = K_k$ as the problem is hard for coloring.

The seminal work in this community is a result of Wrochna who proves the following polynomial result

Theorem 63 (Wrochna [135])**.** *H-Homomorphism Reconfiguration can be decided in polynomial time when H does not contain any (non necessarily induced) copy of C_4.*

The proof is based on an homotopy argument and somehow generalizes the winding number argument used to prove that we can decide the existence of transformations between 3-colorings. In particular, one can easily prove that the number of times a cycle C of G turns around a cycle of H cannot be modified.

Many recent results extended the results of Wrochna by generalizing his arguments for (directed) homomorphisms. Many hardness results have also been obtained, even for very simple graphs H such as reflexive graphs. A graph is *reflexive* if every vertex has a self-loop. The existence of a homomorphism to a reflexive graph is trivial since one simply has to map all the vertices of G to the same vertex in H. Surprisingly, the picture is really different for homomorphisms since it has been proven in [109] that Homomorphism Reconfiguration is hard even restricted to reflexive graphs.

Chapter 4

Independent set Reconfiguration

4.1 Token Sliding and Token Jumping models

In this chapter, we focus on the recon guration of independent sets. Given a simple undirected graph G, a set of vertices $S \subseteq V(G)$ is an *independent set* if the vertices of this set are pairwise non-adjacent. Finding an independent set of size k, i.e., the Independent Set problem, is known to be NP-hard, but also W[1]-hard[1] parameterized by solution size k and not approximable within $O(n^{1-\epsilon})$, for any $\epsilon > 0$, unless P = NP. Moreover, Independent Set remains W[1]-hard on graphs excluding C_4 (the cycle on four vertices) as an induced subgraph [29].

In this manuscript, we view an independent set as a collection of tokens placed on the vertices of a graph such that no two tokens are placed on adjacent vertices. This gives rise to three natural adjacency relations between independent sets (or token con gurations), also called *recon guration steps*.

Token Sliding. In the Token Sliding model, introduced by Hearn and Demaine [90], two independent sets are *TS-adjacent* (or *adjacent* when the variant is clear from context) if one can be obtained from the other by removing a token from a vertex u and immediately placing it on another vertex v with the requirement that uv must be an edge of the graph. The token is then said to *slide* from vertex u to vertex v along the edge uv. A *(TS)-recon guration sequence* from I to J, denoted by $I \leftrightsquigarrow_{TS} J$, is a sequence of independent sets starting on I and ending on J such that consecutive independent sets in the sequence are TS-adjacent. The *k-TS-con guration graph of* G, denoted by $G_{TS}(G,k)$, is the graph whose vertices are the k-independent sets of G and there is an edge between two independent sets if they are TS-adjacent. Note that there exists a TS-recon guration sequence

[1] Informally, this means that it is unlikely to be xed-parameter tractable.

from I to J if I and J are in the same connected component of $G_{TS}(G,k)$. The main algorithmic problem that has been considered in the literature for this model is the following problem:

TOKEN SLIDING INDEPENDENT SET REACHABILITY (TS-IRS)
Input: A graph G, an integer k, two independent sets I, J of size k.
Output: YES if and only if $I \quad_{TS} J$.

Hearn and Demaine [90] introduced this problem in a reduction devoted to prove that the so-called warehouse man's problem is PSPACE-complete, even restricted to robots of bounded size. Without giving the details of that problem, it belongs to a wide class of problems in robotics where we want to transform a solution (consisting of robots in some position) into a target position by moving robots one by one while satisfying conditions all along the transformation (in order to avoid collisions between robots and unauthorized moves of robots).

If we see the possible sets of positions as the vertices of a graph and we put an edge between two positions if two robots cannot be in these two positions because they are too close or collapsing (the *con ict graph*), then a set of possible positions for robots corresponds to an independent set of the con ict graph. In all these problems, the robots cannot jump from a position to another but can simply move to close positions which corresponds to moving along edges of the con ict graph.

Another motivation behind the token sliding model (not only for independent sets but more globally for recon guration problems) is its relation with *puzzles*. A puzzle is a one-player game where, given some con guration of the puzzle, one wants to transform this con guration into a desirable position. As we have already seen in the introduction, the Rubik's cube and the 15-puzzle for instance lie in this category. For the example of the 15-puzzle for instance, the way we are transforming a con guration into another consists in sliding the empty slot along an edge of the board (consisting of a 4 4 grid). The warehouse's man problem, as well as the *Rush Hour* game, can be seen as recon guration problems close to the TS-IRS problem where we want to allow some space for the robots to move in a con ict graph.

Token Jumping. In the TOKEN JUMPING model, introduced by Kaminski et al. [101], we drop the restriction that the token should move along an edge of G and instead we allow it to move to any vertex of G provided it does not break the independence of the set of tokens. That is, a single recon guration step consists of rst removing a token on some vertex u and then immediately adding it back on any other vertex v, as long as no two tokens become adjacent. The token is said to *jump* from vertex u to vertex v. Two independent sets are *TJ-adjacent* (or *adjacent* when the variant is clear from context) if one can be obtained from the other by removing a token from a vertex u and immediately placing it on any other vertex v. A *(TJ)*-

recon guration sequence from I to J, denoted by $I \quad_{TJ} J$, is a sequence of independent sets starting on I and ending on J such that consecutive independent sets in the sequence are TJ-adjacent. The *k-TJ-con guration graph of* G, denoted by $G_{TJ}(G,k)$, is the graph whose vertices are the k-independent sets of G and there is an edge between two independent sets if they are TJ-adjacent. Note that there exists a TJ-recon guration sequence from I to J if I and J are in the same connected component of $G_{TJ}(G,k)$. Again, the most studied algorithmic problem for that model is the following:

Token Jumping Independent Set Reachability (TJ-IRS)
Input: A graph G, an integer k, two independent sets I, J of size k.
Output: YES if and only if $I \quad_{TJ} J$.

When we restrict to maximum independent sets of the graph, the token sliding and token jumping models are equivalent (and then TJ-ISR and TS-ISR are equivalent). Indeed, if we have a maximum independent set I of a graph G and we want to remove a vertex u from I and replace it by a vertex v, the vertex v has to be in the neighborhood of u since otherwise $I \cup \{v\}$ would be an independent set, which contradicts the fact that I is maximum. So, in what follows, when we say that some independent set recon guration problem is hard (or simple) for maximum independent sets, the result holds for both the TS and TJ models.

In general, however, TS-ISR and TJ-ISR have di erent behaviors as we will see later. In most of the cases, constructing algorithms in the token jumping model is easier than the token sliding model. We will explain why in Section 4.5 but the idea is that, for token jumping, we just have to keep an independent set while in the token sliding model, we also have to slide along edges. In some sense, the sliding version adds some connectivity condition on the transformation. This additional condition might force us, in order to simply move a token from u to a non-adjacent vertex v, to perform a lot of slides of other tokens to allow this move. Indeed, we would like to slide the token on a path from u to v but are not sure that it keeps a solution all along. So we might need to move back and forth a lot of tokens to allow the moves along this path. In particular this additional condition is really helpful in order to build hardness proofs

One of the motivations to study this variant comes from statistical physics. As for recoloring, one can sample independent sets at random using a Markov chain as follows: at every step, select a vertex v of the independent set I at random and a vertex w of the graph at random and replace v by w in I if the resulting set is an independent set. This hard-core model is studied in various elds such as statistical physics, operational research or random sampling.

Note that recently, the puzzle community tried to construct hard puzzles based on \jumping moves" rather than sliding moves. One of the motivations

to introduce and study these puzzles come from the existing hardness results on TJ-ISR. The reader interested in jumping puzzles is, for instance, referred to [103].

Token Addition Removal. In the TOKEN ADDITION REMOVAL model, introduced by Ito et al. in [96], we are allowed to add or remove a vertex from an independent set at each turn. One can indeed find a reconfiguration sequence between any pair of independent sets in this model by removing all the vertices of the first independent set and then adding all the vertices of the other. What makes the problem hard is that we add a threshold value t and ask for a transformation where all the independent sets in the sequence have size at least t. We will not really discuss this model in the rest of the manuscript since the Token Jumping and the Token Addition Removal models have been proven to be equivalent in [101].

4.2 Algorithmic aspects of Independent Set Reconfiguration

4.2.1 Hardness results

NCL and hardness on planar graphs. TS-ISR and TJ-ISR has been proven to be PSPACE-complete even restricted to maximum independent sets for planar graphs of maximum degree 3 in [90]. The initial motivation of the Hearn and Demaine paper was not to prove that TS-IRS and TJ-IRS are PSPACE-complete but to prove that some sliding problems are. To prove their results, they introduce a new problem, called *Nondeterministic Constraint Logic Reconfiguration* (or NCL RECONFIGURATION for short). They then prove that this problem is PSPACE-complete and finally give a simple reduction to prove that TS-ISR and TJ-ISR are PSPACE-complete.

Before explaining a bit what NCL RECONFIGURATION is, let us first explain why these problems lie in PSPACE (and more globally why most of the reconfiguration problems lie in PSPACE). Given a graph G and two independent sets I, J, one can indeed compute the configuration graph in exponential time and then determine if the two independent sets are in the same connected component of the configuration graph. This very simple algorithm ensures that the problems TS-ISR and TJ-IRS lie in EXPTIME. But they actually belong to PSPACE since it is simply a reachability problem in an exponential graph. Indeed, since there are less than n^k independent sets of size k, we can label each independent set with at most $\log(n^k)$ bits. Since the accessibility problem in a graph G can be solved in logarithmic space. Now, we have a simple non-deterministic algorithm to decide if I, J belong to the same connected component by simply \guessing" the next independent set in the sequence. This algorithm only needs a polynomial

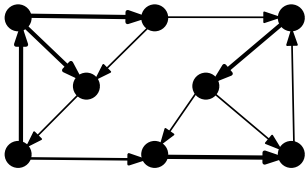

Figure 4.1: An AND/OR *graph with an orientation. Bold edges have weight 2 and light edges have weight one. All the vertices either have three or one bold edges. The con guration represented here is not valid since the bottom left* AND *vertex only have one incoming arc of weight one. All the other vertices satisfy the condition that the in-degree is at least two.*

number of bits of memory since it only needs to check that the current and next independent sets are indeed adjacent in the con guration graph which can be done in polynomial time. Since, by Savitch's theorem, we have NPSPACE=PSPACE, the problems TS-ISR and TJ-IRS are in PSPACE.

Let us now explain what is NCL (see Figure 4.1 for an illustration). Suppose that we are given a cubic graph with edge-weights in $f1, 2g$ such that each vertex is either incident to three weight-2 edges (\OR vertex") or one weight-2 edge and two weight-1 edges (\AND vertex"), which we call an AND/OR *graph*. A (valid) *NCL con guration* is an orientation of the edges of the graph such that the total weights of incoming arcs at each vertex is at least two[2]. Two NCL con gurations are *adjacent* if they di er in a single edge direction. In NCL RECONFIGURATION, we are given an AND/OR graph and two NCL con gurations, and the objective is to determine whether there exists a sequence of adjacent NCL con gurations that transforms one into the other. It is shown in [90] that NCL RECONFIGURATION is PSPACE-complete.

Theorem 64 (Hearn, Demaine [90])**.** NCL RECONFIGURATION *is PSPACE-complete, even restricted to planar graphs.*

Many recon guration problems are proven to be PSPACE-complete using reduction from NCL RECONFIGURATION. (In some sense NCL RECONFIGURATION can be considered as the equivalent of 3-SAT for recon guration). We will not discuss this reduction nor the uncrossing gadget (for NCL and then further for ISR). Using Theorem 64, they obtain the following corollary, which is the starting point of an important line of research in combinatorial recon guration:

[2] To explain a bit the notions, note that for an OR vertex, we can simply select any incoming edge (we simply need an OR on the three incoming edges) while for an AND vertex, if the weight 2-arc is not incoming we need to select the second AND third arc as incoming arcs.

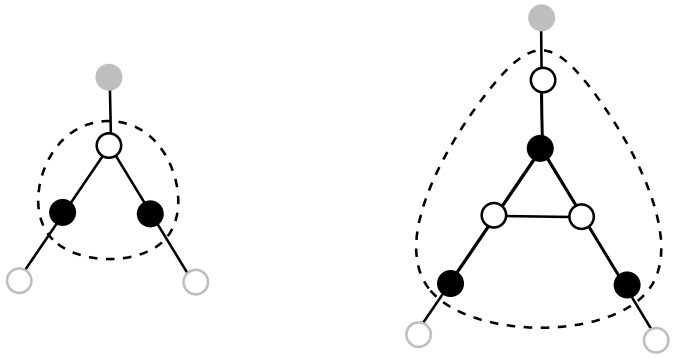

Figure 4.2: The gadgets of the hardness reduction from Hearn and Demaine to prove that TS-ISR is PSPACE-complete.

Theorem 65. *TS-ISR and TJ-ISR are PSPACE-complete, even restricted to planar graphs of maximum degree* 3 *and restricted to maximum independent sets.*

Sketch of the proof. The proof is a simple reduction from NCL recon guration. The gadgets are represented in Figure 4.2. The set of edges leaving the gadget are called port edges and are linked to their corresponding neighbors in the AND/OR graph. For every edge uv, if uv is oriented from u to v in an NCL con guration, we place a token on the port edge of uv in the gadget of v. One can easily notice that every independent set with exactly one token on each port edge corresponds to a valid con guration of the AND/OR graph (and conversely) and that a valid NCL recon guration sequence gives a valid TS-ISR sequence. □

Word reconfiguration and hardness on bounded bandwidth graphs. Let us rst de ne bandwidth. A graph G has *bandwidth* at most k if there exists a path P and a bag function B that assigns to each vertex of the graph G a vertex of P in such a way:

- For every vertex p *2* P, $B^{-1}(p)$ has size at most k. In other words, at most k vertices are assigned to the same vertex of P.
- For every edge uv of G, $B(u)$ and $B(v)$ are either the same or adjacent in P.

One can easily remark that a graph of bandwidth at most k has pathwidth at most $2k$. Indeed assume that P has r nodes, we can then de ne a path P' on $r - 1$ nodes such that the bag of the i-th vertex of P' consists of the bags of the i-th and $(i + 1)$-th nodes of P. The path P' is indeed a path decomposition of P with bags of size at most $2k$.

The proof of the following theorem is based on Theorem 57 which ensures that H-WORD REACHABILITY is PSPACE-complete:

Theorem 66 (Wrochna [134])**.** *TS-ISR and TJ-ISR are PSPACE-complete, even on bounded bandwidth graphs and restricted to maximum independent sets.*

Sketch of the proof. Let A, B be two H-words of the same length n with alphabet . The idea consists in creating n cliques of size . An independent set of size n has to contain exactly one vertex per clique which corresponds to a letter. Now, between every pair of consecutive cliques we put an edge between a and b if a cannot be followed by b in the graph H. In other words, between consecutive pairs, we always have the same bipartite graph. The graph indeed has bounded bandwidth because every vertex is only adjacent to its own clique, the clique before and the clique after it. □

Note that the constant in the proof of Theorem 66 is not explicit (but anyway large). Determining an explicit upper bound is open:

Question 67. *Give an explicit upper bound r on which TS-ISR and TJ-ISR are PSPACE-complete on graphs of bandwidth at most r? Pathwidth at most r? Treewidth at most r?*

Since there is no known upper bound on which the problem is known to be hard, it is natural to study it for small values of r for understanding its behavior. Several groups tried to determine if the problem is polynomial for graphs of treewidth at most 2. As far as I know, none of them succeeded. Even worse, the problem remains open even on very restricted subclasses of graphs of treewidth at most 2, like outerplanar graphs.

Conjecture 68. *TS-ISR and TJ-ISR can be decided in polynomial time on outerplanar graphs and more generally on graphs of treewidth 2.*

Usually, on these classes of graphs, it is easy to design dynamic programming algorithms. Unfortunately, as we will see later, designing dynamic programming algorithms for reconfiguration problems is much harder than in the classical setting. Solving this conjecture, which can be seen as a toy problem, might need a new strategy which might be helpful for other problems too.

Up to this point, all the hardness results we mentioned on one model also hold for the other. One can wonder if the two models are equivalent. The answer is negative! There are a lot of classes where the behavior in the two models are different. We will discuss in details a bit later the particular case of chordal graphs which is trivial for token jumping and hard for token sliding. But let us first discuss the class of bipartite graphs:

Theorem 69 (Mouawad, Lokshtanov [111])**.** *TS-ISR is PSPACE-complete and TJ-ISR is NP-complete.*

Sketch of the proof. We will only discuss the TJ-ISR part of the theorem, the other follows from a classical hardness reduction. Note that since reconfiguration problems do not clearly lie in NP, we have to prove both inclusions. The authors prove that the problem is NP-hard from the fact that computing treewidth in co-bipartite graphs is NP-hard and they can prove that both problems are equivalent. So, in the rest of the sketch we will focus on the fact that the problem belongs to NP.

Instead of Independent Set Reconfiguration, we will consider the Vertex Cover Reconfiguration problem. On bipartite graphs, the two problems are the same since we only have to see that if we replace edges by non-edges between the two sides of the bipartite graph, an independent set becomes a vertex cover and conversely. Note moreover that instead of TJ we will consider the TAR version where we are allowed to add and remove vertices from the vertex cover (without going above a fixed threshold). The proof follows from two different and both very nice lemmas.

The first idea of the proof consists in proving that we can perturb the graph in order to ensure that the minimum vertex cover reachable from a vertex cover X is unique. To do so, we consider the connected component of X in the configuration graph and let us denote by Y a vertex cover of minimum size. Now, we modify the graph by duplicating $2n$ 1 times the vertices of Y and $2n$ times the others. Let us denote by G_Y the new graph. The new threshold on the size of the vertex cover is $\nu = 2nk$ $2n$ 1. Let Z be a vertex cover of size at most ν in G_Y. We say that Z saturates a 2 $V(G)$ if all the copies of a in G_Y contain a token. One can easily prove that if Z is a vertex cover then it should saturate a vertex cover of G. Moreover, if a transformation initially exists, then we can adapt it in G_Y. So we can from now on assume that, in the connected component of our vertex cover, we have a unique minimum size vertex cover[3].

The second idea of the proof consists in proving that, if we start from a vertex cover X in a graph G, then a smaller vertex cover in the connected component of X (if it exists) can be reached in at most $O(n)$ steps. To prove that, the authors show that there should be an illegal crown. A *crown* in a bipartite graph (A, B) is a pair (C, H) such that $jCj > jHj$ and $N(C) \setminus A = H$. The idea is that if a vertex cover contains at least one vertex of H then it is never interesting to move a vertex from H. Indeed otherwise, we should cover all the edges between that vertex of H by covering all the vertices of H whose size is larger than C. This (actually technical !) lemma being proved, we can simply remove that vertex (and one token from the graph). Using this lemma, they succeed to prove that, since a shortest path to a vertex cover of smaller size cannot contain any vertex of a crown, then such a path should have linear length at most. In order to prove it they

[3]Note that this reduction actually works for any possible graph class but as far as I know has never been used for other classes.

succeed to prove that in order to reach a vertex cover of smaller size, one can do it via a monotone path meaning that a token will be at most once added and removed from the vertex cover during the sequence. Note that proving such a monotonicity result in recon guration is really rare (and also note that this proof is nothing but simple). This indeed ensures the existence of a linear transformation (if it exists) that decrease smthe size of the vertex cover.

Let us now conclude by recombining all the arguments. Let X, Y be two vertex covers. Let us guess the smallest vertex cover in the component of X and apply the reduction from it. The new graph G' has quadratic size and the new threshold is ν. Now we now that we can in $O(|V'|) = O(n^2)$ step reduces by one the size of the vertex cover and we can reduce the size of the vertex cover at most $O(n^3)$ times. So the oracle can give us, if it exists a transformation from X, Y to the smallest vertex cover in at most $O(n^4)$ steps if it exists which concludes the fact that the problem belongs to NP. □

4.3 Chordal graphs

Chordal graphs are very interesting since their behavior really di ers depending if we consider the Token Sliding or Token Jumping model. Let us rst remark that the following holds:

Theorem 70. *For every chordal graph G, the k-TJ con guration graph is connected and has diameter at most $2k$.*

Sketch of the proof. The proof is by induction on k. The proof follows from the fact that chordal graphs contain simplicial vertices. A vertex is *simplicial* if its neighborhood is a clique. One can easily prove that, if v is a simplicial vertex, then there exists a maximum independent set that contains v. Now, consider two independent sets I, J of size k. Since $N(v)$ is a clique, $N(v) \setminus I$ and $N(v) \setminus J$ have size at most one. So, in at most 2 steps, we can add v in I and in J. (Indeed if v is not in I then we simply remove the unique vertex of $N(v) \setminus I$ if it exists or any other vertex from the independent set and add v). Now, $G \setminus N(v)$ is still a chordal graph and the conclusion follows by induction. □

One can wonder if the same holds for Token Sliding. The answer is negative. We obtained the rst step in that direction with Marthe Bonamy in [18] where we showed that deciding if the TS-con guration graph is connected is co-NP-hard even restricted to split graphs[4], a simple subclasses of chordal graphs. Belmonte et al. later gave in [13] a reduction from NCL RECONFIGURATION to prove that the following holds:

[4] A graph is *split* if its vertex set can be partitioned into a clique and an independent set (with arbitrary edges between these two sets).

Theorem 71 (Belmonte et al. [13]). *TS-ISR is PSPACE-complete, even restricted to split graphs.*

One can then naturally wonder on which subclass of chordal graphs TS-ISR can be decided in polynomial time. There are two natural ways to simplify chordal graphs: either we reduce the size of the cliques or we reduce the number of branchings in the clique tree. In the extremal cases, we obtain trees in the first case (clique number two) and interval graphs in the second (the clique tree is a path). Both problems have been proven to be decidable in polynomial time.

Theorem 72 (Demaine et al. [63]). *TS-ISR can be decided in linear time on trees. And every connected component of $G_{TS}(T, k)$ has diameter at most $O(n^2)$ for every tree T and every integer k.*

Sketch of the proof. Let I be an independent set of size k of a tree T. The main step of the proof consists in proving that we can, in linear time, determine all the vertices of I that are *rigid*, that is vertices that belong to all the independent sets of the connected component of I in $G_{TS}(T, k)$.

Let I, J be two independent sets of T. If there exists a transformation from I to J then I and J must have the same rigid vertices. If the sets of rigid vertices are different, we return false. Otherwise, we can simply remove all these vertices and their neighborhoods from the graph safely.

The second step of the proof consists in proving that the answer is positive as long as there is the same number of tokens in each connected component in each connected component of the remaining forest. In other words, there always exists a transformation between two non rigid independent sets in every tree. □

As we already observed, Theorem 72 corresponds to chordal graphs with clique number equals 2. One can naturally wonder if the following holds:

Question 73. *Can TS-ISR be decided in polynomial time on chordal graphs of bounded clique number?*

Note that in the case of split graphs (on which TS-ISR is PSPACE-complete), the clique number is indeed arbitrarily large. Moreover with Valentin Bartier and Amer Mouawad, we proved that the problem is FPT in that case [8].

With Marthe Bonamy, we also proved that TS-ISR can be decided in polynomial time on interval graphs:

Theorem 74 (Bonamy, Bousquet [18]). *TS-ISR can be decided in polynomial time on interval graphs.*

We prove this result using dynamic programming. While it is usually easy to design dynamic programming algorithms for chordal or interval graphs for optimization problems, it is much harder for reconfiguration problems. Consider for instance the very simple example of the computing of a maximum independent set on a bounded treewidth graph G. If one wants to construct a maximum independent set, there is a very simple dynamic programming algorithm on the tree decomposition rooted on an arbitrary node r. For every node s and every subset R of nodes in the bag S of s, we define $v(s, R)$ as the size of the maximum independent set of the subgraph rooted on s containing R and not containing $S \setminus R$. Since S separates the subgraph rooted on s with the rest of the graph, one can now easily compute this value with a dynamic programming algorithm bottom-up from the leaves.

This does not work for reconfiguration of independent sets since TS-ISR is PSPACE-complete. One can wonder why. The problem with reconfiguration is that we cannot really consider separately the vertices that belong to the subtree rooted on a node s and the other part of the graph. Indeed, imagine that we have an instance where we want to transform I into J where $I \setminus S = J \setminus S$. Let us denote by H the subgraph of G rooted on s and $G' = (G \setminus H) \cup S$. The fact that I and J agree on S does not imply that these vertices do not move during a transformation (that is, we cannot simply remove vertices in the intersection of the solutions and restrict to the other vertices). Let us denote by V_1 the subset of vertices in $V(H) \cap S$ and $V_2 = V \setminus (V_1 \cup S)$. In order to find a transformation between I and J, we might have to move tokens of the independent set from V_1 to V_2 via S to free some space and allow moves on V_1 (and conversely). Even worse, there is no reason why the number of back and forth moves between V_1 and V_2 would be bounded (and since the problem is PSPACE-complete, it is indeed not the case). For interval graphs, we succeeded in dealing with these back and forth moves.

Sketch of the proof of Theorem 74. In an interval graph, there are two natural orders on the vertices: we can sort them by increasing right endpoint (*end ordering*) or by increasing left endpoint (*beginning ordering*).

The first claim of the proof is the following: to decide if we can transform an independent set I into an independent set J in polynomial time, we can prove that it is enough to decide in polynomial time if the leftmost possible vertex (for the end ordering) that can be reached from I and from J are the same (the conclusion then follows by induction). In other words, even if in general it is not true that if a vertex belongs to the intersection of the independent sets we can delete it (as well as its neighborhood), it is valid for that special vertex.

So we simply have to find an independent set in the component of I in $G_{TS}(G, k)$ with the leftmost possible right endpoint. To do so, we prove that

we can reach such an independent set by repeating the following procedure until it reaches a fixed point:

- Try to push left the leftmost token of the independent set while the other tokens of the independent set are fixed,
- Try to push right the second token of the independent set while the first token is fixed (but all the other tokens are allowed to move).

The first item can trivially be performed in polynomial time since we simply try to move a token to another in a graph (which is simply a connectivity question). We prove that the second can be solved by induction using a non-trivial dynamic programming (the goal being to only have to remember in total a polynomial number of configurations to start with which can actually be done). □

The drawback of this proof is that it does not guarantee that every connected component of $G_{TS}(G, k)$ has polynomial diameter. Brianski et al. improved our result in [50] by proving that the diameter of each connected component of $G_{TS}(G, k)$ is quadratic when G is an interval graph.

As we already observed, interval graphs can be seen as an extremal case of chordal graphs since clique trees are paths in interval graphs. One can wonder what can happen if we put slightly less restrictions on the clique tree. If we decide to bound the number of nodes of degree at least 3 in the clique tree, then the problem becomes PSPACE-complete since the clique tree of split graphs is a star. One can then naturally ask what happens if the number of leaves in the clique tree is bounded, a question left open in [18]:

Conjecture 75 (Bonamy, Bousquet [18])**.** *TS-ISR can be decided in polynomial time on chordal graphs whose clique tree has a bounded number of leaves.*

Note that TS-ISR has been proven to be decidable in polynomial time in other restricted classes such as block graphs [93] (chordal graphs where all the 2-connected components are cliques). The problem can also be decided in polynomial time on cactus graphs [95] (graphs where all the 2-connected components are cycles).

4.4 Other polynomial time algorithms

H-free graphs. For Token Sliding, Kaminski et al. gave in [102] a linear-time algorithm to decide TS-ISR for cographs (which are P_4-free graphs). Bonsma et al. proved in [32] that TS-ISR can be solved in polynomial time for claw-free graphs. This result generalizes a result of Ito et al. [96] who

proved that TS-MATCHING RECONFIGURATION can be decided in polynomial time (a matching is an independent set of a line graph, which is a claw-free graph).

In the Token Jumping model, Bonsma proved that TJ-ISR can be decided in polynomial time for cographs in [33] as well as claw-free graphs [32]. With Marthe Bonamy, we proved that deciding if $G_{TJ}(G,k)$ is connected is polynomial for cographs [17] answering a question of [33].

One can generally wonder what is the complexity of TS-ISR and TJ-ISR for more general graphs H. We proved with Valentin Bartier and Moritz Mühlenthaler [10] that the following holds using an Alekseev trick:

Theorem 76 (Bartier, Bousquet, Mühlenthaler [10])**.** *TS-ISR and TJ-IRS are PSPACE-complete on H-free graphs for every connected graph H that is not the subgraph of a subdivision of a claw (even restricted to maximum independent sets)*[5]*.*

Sketch of the proof. The proof follows the same line as Alekseev's proof [2] to prove the hardness of maximum independent set in H-free graphs when H is not the subdivision of a claw. Let I and J be two maximum independent sets of a graph G and let G' be a (well-chosen) subdivision of G. Alekseev observed that every maximum independent set of G can be associated to a maximum independent set of G'. We denote the independent sets of G' corresponding to I and J by I' and J' respectively. Using Alekseev reduction, we only have to show that there exists a reconfiguration sequence that transforms I into J if and only if there exists one that transforms I' into J' to complete the proof. □

Since the TS-ISR and TJ-ISR can only be polynomial on H-free graphs where H is the subdivision of the claw (for connected graphs H), and since we know that the problem can be decided in polynomial time on claw-free graphs [32], one can naturally wonder what happens on fork[6]-free graphs. We proved the following:

Theorem 77 (Bartier, Bousquet, Mühlenthaler [10])**.** *The following holds:*

- *TS-ISR and TJ-IRS can be decided in polynomial time on fork-free graphs for maximum independent sets.*
- *TS-ISR can be decided in polynomial time on fork-free graphs.*

The idea of this proof consists in reducing the problem to the claw-free case that has been done by [32]. While it is not complicated for maximum independent sets using a short but nice argument, the argument is much

[5] A graph G_1 is a *subdivision* of G_2 if it can be obtained from G_2 by subdividing edges.

[6] A *fork* is a star with 3 leaves where one of the branches is subdivided once. In other words, it is a claw where one of the edges has been subdivided once.

more complicated for non necessarily maximum independent sets for Token Sliding. We did not succeed to prove it for Token Jumping and left the problem as open. While the structure of the symmetric di erence between two independent sets is well understood (paths, cycles, complexes), we did not succeed to treat complexes (which can easily be treated for Token Sliding). We nevertheless conjecture that the following holds:

Conjecture 78. *TJ-ISR can be decided in polynomial time on fork-free graphs.*

In general, it is not true that the recon guration of maximum independent sets and non maximum independent sets have the same complexity status. For instance, on split graphs, TS-ISR is PSPACE-complete [13] while TS-ISR on maximum independent sets is indeed polynomial.

For the optimization variant of independent sets, the complexity status of computing a MIS in a P_5-free graph remained open during decades until Lokshtanov et al. proved in [113] using potential maximum cliques that it is possible to compute a MIS in polynomial time in P_5-free graphs. This result has been extended to P_6-free graphs in [83]. And the complexity status of computing a MIS in P_k-free graphs is still open for every k 7.

For recon guration, we tried to adapt the potential maximum clique method for P_5-free graphs but did not succeed. A new method may be needed to tackle the problem (and may provide some new insight for its optimization counterpart):

Question 79. *Can TS-ISR and TJ-ISR be decided in polynomial time on P_5-free graphs? On P_k-free graphs?*

Other graph classes. For TJ-ISR, one can notice that the problem has been proven to be polynomial on even-hole free graphs in [101]. As far as I know, the following is still open:

Question 80. *Can TS-ISR be decided in polynomial time on even-hole free graphs?*

TS-ISR and TJ-ISR has also been proven to be decidable in polynomial time on bipartite permutation and bipartite distance-hereditary graphs [81].

4.5 Parameterized algorithms

This part is an extension of the section on parameterized aspects of recon guration of independent sets from the survey [48] we wrote with Naomi Nishimura, Amer Mouawad and Sebastian Siebertz on parameterized aspects of recon guration problems (restricted to independent sets and dominating sets).

4.5.1 Hardness results

On general graphs, the following (unsurprisingly) holds:

Theorem 81 (Mouawad et al. [117]). TJ-ISR *is W[1]-hard when parameterized by the number of tokens* k *(and even when parameterized by* $k + \ell$ *where* ℓ *is the lengh of the sequence).*

Sketch of the proof. The reduction is from the INDEPENDENT SET problem parameterized by solution size k. Given an instance (G, k) of INDEPENDENT SET, we construct an instance $(G + K_{k+1;k+1}, k + 1, I_s, I_t)$ of TJ-ISR, where $G + K_{k+1;k+1}$ is the graph consisting of the disjoint union of G with a biclique having $k+1$ vertices in each part. We let L and R denote the two parts of the biclique and we set $I_s = L$ and $I_t = R$. It is not hard to see that (G, k) is a yes-instance of INDEPENDENT SET if and only if $(G + K_{k+1;k+1}, k + 1, I_s, I_t)$ is a yes-instance of TJ-ISR; before any token can jump to R we must have at least k tokens in G forming an independent set. □

Actually, we proved that TJ-ISR remains W[1]-hard even restricted to $fC_4, \ldots, C_pg$-free graphs [4], for any p 4 adapting a proof that computing a maximum independent set in such a graph is W[1] -hard [29].

TJ-ISR and TS-ISR are then very unlikely to admit FPT algorithms parameterized by k. But one can wonder if they really belong to the class mentioned above. The answer is negative. Bodlaender, Groenland and Swennenhuis characterized in [16] exactly to which class these problems belong to: XL. The class XL consists of the parameterized problems that can be solved by a symmetric Turing machine that uses $f(k)$ $\log n$ space, where k is the parameter, n the input size and f any computable function[7]. (We will explain why the problems belong to that class in the following proof).

Theorem 82 (Bodlaender, Groenland and Swennenhuis [16]). *TS-ISR and TJ-ISR are XL-complete.*

Sketch of the proof. The reduction is from ACCEPTING LOG-SPACE SYMMETRIC TURING MACHINE which is an XL-complete problem. A *Symmetric Turing Machine* is a Nondeterministic Turing Machine, where the transitions are symmetric, that is for any transition we can also take its reverse. Informally speaking[8] it means that if we perform a move in an execution of the Turing machine, we can immediately cancel that move and come back to the previous situation. Note that in recon guration we are indeed symmetric since when we just performed a move, we can cancel it and come back to the previous independent set.

[7] The class XL can be seen as the counterpart of the class L (logarithmic space) for classical complexity setting.

[8] We will not give the formal de nition since it is not that simple.

The typical XL-complete problem is the following:

ACCEPTING LOG-SPACE SYMMETRIC TURING MACHINE
Input: A symmetric Turing Machine $M = (S, \ , T, s_{start}, A)$ [9] with = [1, n] and a work tape with k cells.
Parameter: k.
Question: Does M accept?

Note that there is no word to read but simply the initial tape. Also note that a symmetric Turing machine is not determinist but one can actually easily prove that XL=XSL, that is, what can be done with a deterministic Turing machine is actually equivalent to what can be done with a symmetric non-deterministic Turing machine (see [16] for more details). (This statement can be seen as the parameterized counterpart of the fact that reachability in an undirected graph is in L (while in directed graphs it is in NL). In particular, one can easily see that a recon guration sequence is indeed (i) a symmetric transformation (what has been done can be undone) and (ii) can be represented using $f(k) \log n$ bits since there are at most n^k independent sets. So, from an independent set I where each element of the work tape corresponds to a vertex of I, we simply guess the transformation to J and check that, each time we perform a token move, we are keeping an independent set. So TJ and TS-IRT belong to XSL, which is equal to XL.

In order to prove the hardness result, Bodlaender et al. proved that both TS-ISR and TJ-ISR are equivalent under XL-reductions; so proving the hardness of one of them is enough to get the result. They actually show that it su ces to prove that PARTITIONED TS-INDEPENDENT SET RECONFIGURATION is XL-complete to complete the proof. In this variant, the vertex set is partitioned into $k+1$ sets $V_1, \ldots, V_k, V_{k+1}$ containing exactly one token in the source and target independent sets. And every token is forced to stay on its own subset V_i of vertices all along the transformation. For every $i \quad k$, the token on V_i represents the current letter of on the i-th position of the work tape. The last set V_{k+1} will permit to represent all the pairs (q, i) where q is a state and i the position of the machine. Now if we have a transition, they prove that we can represent it using three additional vertices (for each transition). Very informally speaking, the rst vertex will permit to check that the tape indeed contains the right letters at the correct positions (and that we are allowed to perform this transition), the middle vertex is a waiting vertex and the last vertex permits to check that we have modi ed the tape as claimed by the transition at the end. We refer the reader interested to [16] for more details. □

[9] S is the set of states, is the alphabet, T is the set of transitions, s_{start} is the initial state and A is the set of accepting states.

4.5.2 Independent set reconfiguration under token jumping

Having established hardness, we now focus on the parameterized complexity of TJ-ISR on sparse classes of graphs.

Graphs of bounded degree. We first explain why TJ-ISR is fixed-parameter tractable for the simple case of graphs G of maximum degree [4]. Let I_s, I_t be the source and target independent sets. Since $I_s \cup I_t$ has at most $2k$ neighbors, there are two options, depending on the number of vertices in the graph. If G has at most $3k(\ +1)$ vertices, then we can construct the configuration graph via brute force in FPT-time. On the other hand, if G has more than $3k(\ +1)$ vertices, then $V(G) \setminus N[I_s \cup I_t]$ contains an independent set J of size at least k. It is then possible to form a reconfiguration sequence by moving the tokens in I_s into J one token at a time and then moving tokens from J into I_t in the same manner. Thus, TJ-ISR parameterized by k admits a kernel of linear size on classes of graphs of bounded degree.

In more general sparse classes of graphs, a reconfiguration sequence might not exist even for arbitrarily large graphs since there might exist (almost) universal vertices, i.e., vertices connected to (almost) all. However, due to the sparsity constraints, very few such vertices can exist. As we shall see, in many cases, e.g., planar, nowhere dense, as well as $K_{h,h}$-free graphs, one can use such vertices to find one or more irrelevant vertices.

Planar graphs and $K_{h,h}$-free graphs. Another typical sparse graph class is the class of planar graphs. Ito et al. [98] proved the following:

Theorem 83 (Ito, Kaminski, Ono [98])**.** TJ-ISR *parameterized by k is fixed-parameter tractable on the class of planar graphs.*

Sketch of the proof. For I_s and I_t the source and target independent sets, respectively, we consider projection classes of neighborhoods in $X = I_s \cup I_t$, called *X-projection classes*[10], and prove either the existence of a yes-instance or that each X-projection class can be reduced to having at most $f(k)$ vertices, for some computable function f. Since $|I_s \cup I_t| \leq 2k$ there are at most $2^{2k} = 4^k$ projection classes[11], and we therefore obtain the desired kernel.

Recall that for $Y \subseteq X$, C_Y denotes the set of vertices y of $V(G) \setminus X$ such that $N(y) \cap X = Y$. The vertices of C_Y form the *Y-class*; a Y-class is a *(Y, r)-class* when $|Y| = r$. Since planar graphs are $K_{3,3}$-free, no three vertices can share more than two neighbors, and hence the number of vertices in a (Y, r)-class with $r \geq 3$ is at most 2.

[10] More formally, a vertex v is in the class Y if $N(v) \cap X = Y$.

[11] Actually one can prove that planar graphs have a linear number of classes.

As every planar graph is 4-colorable, every subgraph of size at least $4k$ contains an independent set of size at least k. Thus, if any (Y, r)-class where r 1 is large enough, we can simply transform I_s into an independent set J contained in C_Y and then transform J into I_t; adding as possible rst and last steps the jumps of the token in $I_s \setminus Y$ and the token in $I_t \setminus Y$, if those tokens exist.

To complete the proof, it su ces to consider $(Y, 2)$-classes. We can prove that if C_Y is large enough, then it can be replaced by an independent set of size k. Indeed suppose that at some point of the recon guration sequence a token is moved to a vertex in C_Y to form an independent set I. Clearly, I cannot contain any vertex in Y. In addition, the fact that G does not contain $K_{3,3}$ ensures that every vertex of $V(G) \setminus Y$ has at most two neighbors in C_Y (recall that I does not intersect with Y and $|C_Y|$ 3). In particular, no vertex in I can have more than two neighbors in C_Y, so that the set I has at most $2k$ neighbors in C_Y. Thus, for large enough C_Y, we can always nd a vertex in C_Y that is not in the neighborhood of the current independent set I. Hence, when C_Y is large enough we can instead retain an independent set of size k. □

Since we can bound the size of all the neighborhood classes of X, one can easily prove that we get a kernel of polynomial size. One can then ask the following question:

Question 84. *Does* TJ-ISR *admit a linear kernel on planar graphs?*

The intuition behind the proof of Theorem 83 for $K_{3,3}$-free graphs is that (Y, r)-classes for r 3 are of bounded size and (Y, r)-classes for r 2 are either of bounded size, or can be reduced to bounded size in forming a kernel. However, when we consider $K_{4,4}$-free graphs (or more generally $K_{,}$-free graphs), there can exist a $(Y, 2)$-class C_Y that does not immediately imply a yes-instance (in the sense that we cannot guarantee that we can immediately move source and target independent sets to an independent set of C_Y) nor is easily reducible (in the sense that a vertex of $V(G) \setminus Y$ can be adjacent to arbitrarily many vertices of C_Y). One can then wonder if the problem remains FPT on that class of graphs. We proved with Arnaud Mary and Aline Parreau that the answer is positive:

Theorem 85. *For every* ℓ, TJ-ISR *parameterized by* k *is xed-parameter tractable on* $K_{,}$*-free graphs.*

Sketch of the proof. Note that if no vertex is adjacent to many vertices in C_Y, then we can replace the Y-class by an independent set of size k, just as in the case of $K_{3,3}$-free graphs. Otherwise, a result of Kovari, Sos, Turan [104] ensures that in $K_{,}$-free graphs, for every $\epsilon > 0$, the number of vertices Z_Y incident to an ϵ-fraction of the vertices of the set C_Y is bounded (in terms of ϵ and h).

The key idea of the proof to adapt the proof of Theorem 83 for $K_{i,j}$-free graphs entails updating the set $X = I_s \cup I_t$ by adding to X the vertices of Z_Y for every $Y \subseteq X$, thereby forming the set X'. We can now refine the categorization of the projection classes depending on their neighborhoods in X'. Let $Y' \subseteq X'$ such that the Y'-class is large. Either $Y' \not\subseteq X$ and then the Y'-class is a refinement of a Y-class where $Y \subseteq X$ and $|Y| < |Y'|$. Since (Y', ℓ)-classes have size at most $\ell - 1$, we intuitively "gained" something since we cannot increase too often the neighborhood of a class in X. But if $Y' \subseteq X$ we did not gain anything.

The algorithm proposed in [47] consists of repeating this refinement[12] by iteratively defining sets $X_1 = X, X_2, X_3, X_4, \ldots, X_r$ (where r only depends on k and ℓ). The authors finally prove that if, after all these steps, the size of a class is large enough, then replacing this class by an independent set of size k does not modify the existence of a reconfiguration sequence from I_s to I_t. In other words, even if the neighborhood in X_r of a class has not increased compared to X, then we can nevertheless reduce it. □

Although this proof provides a fixed-parameter tractable algorithm, in contrast to the case of planar graphs, it does not give a kernel whose size is polynomial (in k and ℓ). This difference leads to the following question:

Question 86. *Does* TJ-ISR *admit a polynomial kernel (polynomial in k and ℓ) on $K_{i,j}$-free graphs?*

Degenerate and nowhere dense graphs. Even if degenerate and nowhere dense graphs are included in biclique-free graphs, we think that it is worth mentioning the proof techniques of [112], which are of independent interest.

For both graph classes, the authors in [112] show that one can find irrelevant vertices by making use of the classical result of Erdős and Rado [70], also known in the literature as the sunflower lemma. A *sunflower* with r *petals* and a *core* Y is a collection of sets $S_1, \ldots, S_r$ such that $S_i \cap S_j = Y$ for all $i \neq j$; the sets $S_i \setminus Y$ are petals and we require all of them to be non-empty. Note that a family of pairwise disjoint sets is a sunflower (with an empty core). Now assume that one can find a large (in terms of the number of tokens k) sunflower with core Y in the collection of sets defined by the closed neighborhoods of a set of vertices $v_1, \ldots, v_q$ in the graph. That is, we have $N[v_1], \ldots, N[v_p]$, $p \leq q$, such that such that $N[v_i] \setminus N[v_j] = Y$ for all $i \neq j$. Note that if any independent set I of size k intersects with the core then I cannot contain any vertex from $v_1, \ldots, v_p$. Otherwise, I can intersect with at most k petals. Moreover, any vertex of the graph which is not included in the core or in some petal cannot be adjacent to a vertex from $v_1, \ldots, v_p$. Hence, if p is large enough (in terms of k), then we can always

[12] The exact refinement is actually slightly more involved but the following paragraph, though not entirely accurate, supplies the intuition behind the proof.

delete one of the vertices from $v_1, \ldots, v_p$ without a ecting the existence of a recon guration sequence from I_s to I_t. This follows from the fact that we can always nd another vertex in $v_1, \ldots, v_p$ that will be used to \mimic" the behavior of the deleted vertex.

To nd large sun owers in degenerate graphs one can immediately apply the sun ower lemma; in a degenerate graph at least half of the vertices have degree at most twice the degeneracy (since the average degree of a d-degenerate graph is at most $2d$). Hence, as long as the graph is large enough, one can always nd a sun ower of the appropriate size and delete one irrelevant vertex. One can wonder if the following holds:

Question 87. *Does* TJ-ISR *admit a polynomial kernel (polynomial in* k *and* d*) on* d*-degenerate graphs?*

The algorithm for nowhere dense graphs closely mimics the previous algorithm in the following sense. Instead of using the sun ower lemma to nd a large sun ower, the authors use the so-called notion of uniform quasi-wideness [62] to nd a \large enough almost sun ower" with an initially \unknown" core and then use structural properties of the graph to nd this core and complete the sun ower. At a high level, uniform quasi-wideness states that if G comes from a nowhere dense class and $A \quad V(G)$ is large enough, then we can nd a small set $X \quad V(G)$ whose deletion leaves a large set $B \quad A$ that is 2-independent in $G \quad X$ (where a set B is 2-independent whenever its vertices are pairwise non-adjacent and pairwise do not share any common neighbors). The trick consists of rst looking at $(I_s \ [\ I_t)$-projection classes and then nding a large class in which we can nd the sets B and X (using uniform quasi-wideness). Then, we further classify the vertices of B into X-projection subclasses. The petals of the sun ower, which consist of vertices of B and their neighbors (recall that B is 2-independent) can then be found in a large X-projection subclass and the core will be a subset of $X \ [\ I_s \ [\ I_t$. The existence of a large X-projection subclass can be guaranteed by appropriately choosing the sizes of B and X.

The curious case of bipartite graphs. Although bipartite graphs are not sparse nor appear in Figure 2.1, they merit attention, since we believe that there remain several interesting questions that have yet to be answered. It is still unknown if TJ-ISR is W[1]-hard on bipartite graphs. However, Agrawal et al. [1] showed that the problem is unlikely to be xed-parameter tractable. The proof is based on the fact that BALANCED BICLIQUE[13] does not admit an FPT-time 2-approximation algorithm assuming Gap-ETH[14] [56]. Equivalently, we cannot distinguish in FPT-time (param-

[13] Given a graph G, the goal is to nd the largest k such that G admits a $K_{k;k}$ as an induced subgraph.

[14] Informally speaking, Gap-ETH states that we cannot, for some > 0, distinguish in subexponential time 3-SAT formulas that are satis able from those which are -far from

eterized by k) a graph that admits a balanced biclique of size k from a graph that does not admit a balanced biclique of size $k/2$.

The proof of [1] simply consists in constructing a \weak" reduction from BALANCED BICLIQUE to TJ-ISR on bipartite graphs with the following properties:

- if we have a yes-instance of BALANCED BICLIQUE, then we have a yes-instance of TJ-ISR and,
- if we have a yes-instance of TJ-ISR, then the original graph admits a balanced biclique of size $k/2$.

A xed-parameter tractable algorithm for TJ-ISR on bipartite graphs would imply that we could distinguish in FPT-time graphs having balanced bicliques of size k from graphs having balanced bicliques of size less than $k/2$, a contradiction under Gap-ETH.

More open problems. Probably the most exciting research direction is related to dense graph classes on which almost nothing is known. In particular one might ask the following:

Question 88. *Is* TJ-ISR *parameterized by* k *xed-parameter tractable on graphs of bounded semi-ladder index?*

Question 89. *Is* TJ-ISR *parameterized by* k *xed-parameter tractable on graphs of bounded cliquewidth (and, more generally, bounded twinwidth)?*

We conclude this section with one more interesting open question. There is a correlation between VC-dimension and complete bipartite subgraphs. Namely, a $K_{h;h}$-free graph has VC-dimension at most $O(h)$. Since the TJ-ISR problem is W[1]-hard on general graphs and xed-parameter tractable on $K_{h;h}$-free graphs, one can naturally ask if this result can be extended to graphs of bounded VC-dimension. We proved with Arnaud Mary and Aline Parreau in [47] that TJ-ISR is polynomial-time solvable on graphs of VC-dimension 1, NP-hard on graphs of VC-dimension 2, and W[1]-hard on graphs of VC-dimension 3. The following question remains open:

Question 90. *Is* TJ-ISR *parameterized by* k *xed-parameter tractable on graphs of VC-dimension 2?*

4.5.3 Independent set reconfiguration under token sliding

Just like the token jumping variant, TS-ISR is W[1]-hard on general graphs. It remains hard even when restricted to bipartite graphs or $fC_4, \ldots, C_pg$-free graphs [4], for any p 4. Let us sketch for completion the simple

being satis able.

proof for general graphs, which mimics the reduction for the token jumping model. The reduction is from the MULTICOLORED INDEPENDENT SET (MIS) problem, known to be W[1]-hard. In the MULTICOLORED INDEPENDENT SET problem, we are given a graph consisting of k cliques of arbitrary size and extra edges joining vertices from di erent cliques. The goal is to nd an independent set of size k which must intersect with each clique in exactly one vertex (called a multicolored independent set). Given an instance $(G = (V_1 [\dots [V_k, E), k)$ of MULTICOLORED INDEPENDENT SET, we construct an instance (H, k, I_s, I_t) of TS-ISR as follows. We rst let $H = G + K_{k+1;k+1}$. We let $L = fl_1, \dots, l_{k+1}g$ and $R = fr_1, \dots, r_{l+1}g$ denote the two parts of the biclique. Finally, we add edges between fl_i, r_ig and every vertex in V_i, for i 2 $[k]$. It is not hard to see that (G, k) is a yes-instance of MIS if and only if $(H, k + 1, L, R)$ is a yes-instance of TS-ISR; before any token can slide to R we must have k tokens in G forming a multicolored independent set.

Galactic graphs. While the parameterized complexity of TJ-ISR on sparse classes of graphs is well-understood, the situation is quite di erent for TS-ISR. Indeed, TS-ISR is more complicated than the token jumping variant even in sparse classes of graphs because of what is called the *bottleneck e ect* [8]. Under the token jumping model, the existence of a large independent set in the non-neighborhood of both the source and target independent sets is enough to ensure the existence of a recon guration sequence from I_s to I_t. To the contrary, under the token sliding model, a small cut might prevent us from nding such a transformation. Let us illustrate that behavior on the simple example of a star to which we attach a long path. If there are at least two tokens on the leaves of the star, none of the tokens on leaves can slide. Consequently, we will not be able to move any token from leaves of the star to the path even if the diameter of the graph and the independence number of the graph are arbitrarily large.

In an ideal world, we would like to determine if there exist frozen tokens (in the sense that they will never be able to slide) and remove them from the graph. While it can be done e ciently for trees [63], this problem is hard in general. So one needs to nd another strategy. Let us return to the example of the star to which a path is attached. One can easily notice that, if the path is long enough, we can form an equivalent instance by reducing its length. Note that by doing so we found (1) a large subset of vertices which can be replaced by a smaller one to form an equivalent instance and, (2) the resulting graph is in the same class (in this case, trees). In order to prove the existence of a xed-parameter tractable algorithm, one usually wants to perform such reductions but, even if it is often easy to nd a subset of vertices that behave \nicely" (in the sense that we understand quite well how the tokens behave in that subset), it is not always easy to reduce the size

of the graph and obtain a resulting graph within the same class. Moreover, the proofs are usually technical since, quite often, neither of the directions of the equivalence are trivial to prove. To overcome this issue, a more general version of TS-ISR was introduced on galactic graphs [9].

A *galactic graph* is a graph where $V(G)$ is partitioned into two sets: the *planets* and the *black holes*. In a galactic graph, the rules of the TS-ISR game are slightly modified. When a token reaches a black hole, the token is *absorbed* by the black hole. Since black holes are considered to make tokens "disappear", it is allowed for tokens to be assigned to adjacent vertices as long as at least one of the vertices is a black hole. Furthermore, each black hole is allowed to absorb up to k tokens. In addition, a black hole can also *project* any of the tokens it previously absorbed onto any vertex in its neighborhood, be it a planet or a black hole.

Why galactic graphs? At first glance, they might seem very artificial. However, in practice, one often finds a large connected structure on which one can prove that one can "hide" as many tokens as one wants. However, it is usually complicated to prove the existence of a smaller structure with the same property while staying in the same class (and even when we find it, we may need to repeat similar proofs several times since the "contracted structure" may have to be adapted to each graph class[15]). Shrinking all this structure into a single vertex drastically simplifies this step as well as the technicalities of the proofs. An important result one can prove using black holes is the following:

Lemma 91 (Bartier, Bousquet, Mouawad [8])**.** *Let G be a galactic graph and I_s, I_t be two independent sets of size k. If G contains a shortest path P of length (k) such that $N_G(V(P)) \cap V(P)$ does not contain any token of $I_s \cup I_t$, then P can be contracted into a black hole.*

Sketch of the proof. The main ingredient of the proof is the following: We prove that if there is a reconfiguration sequence from I_s to I_t then we can find one where, at each step, the number of tokens in $N(P)$ is at most one. To do so, we simply project on P every token that appears in $N(P)$ during the sequence. To guarantee that such a projection exists, we have to prove that when a token has to enter (or leave) on a vertex v in $N(P)$, then this token is not adjacent to a vertex already in P. But since P is a shortest path, every vertex in $N(P)$ is adjacent to at most 3 vertices of P. And then we can slide the vertices of the independent set already in P along the edges of P to be sure that they are not in the neighborhood of v. (A similar argument holds when we want to allow a token to leave the black hole). □

The key argument of the proof of Lemma 91 is that long shortest paths (and more generally blackholes) can swallow tokens in their neighborhood.

[15] e.g. an edge contraction keeps a planar graph while it is not stable for graphs of maximum degree .

Even better, we can easily prove that if there is a vertex v in the neighborhood of a blackhole b such that v has no token on its neighborhood, then b can swallow v, i.e. v and b can be contracted. Using these rules, together with other simple rules, one can prove the following:

Theorem 92 (Bartier, Bousquet, Mouawad [8]). TS-ISR *is xed-parameter tractable parameterized by* $k +$ *, where is the maximum degree of* G.

Sketch of the proof. The proof consists in showing that, after applying black holes reduction rules, the graph has bounded size (in and k) and we can conclude using a brute force algorithm. If the graph has bounded diameter then the conclusion indeed follows (since the maximum degree is bounded). So we can assume that there exists a long shortest path. Since every token is adjacent to at most 3 vertices on that path, there exists a long path with no token on its neighborhood, and this part can be reduced using a black hole.

The key argument to conclude is that even if the graph does not have bounded degree (black holes might have large degree), the classical vertices (called planets) still have bounded degree. And using simple reduction rules one can prove that the number of planets (and of blackholes) is bounded by a function of . □

We insist on the fact that the trick of the proof then consists in considering a more general problem that allows us to easily reduce the graph while (almost) staying in the class. It is (as far as I know) the rst time this type of idea was used in recon guration and really allowed us to reduce the complexity of proof. Theorem 92 implies in particular that TS-ISR is xed-parameter tractable on classes of graphs of bounded degree (and on classes of graphs of bounded bandwidth). To extend results to more general classes of graphs, we need more tools.

Types. A surprisingly hard question is the following: Let G be a graph, I_s, I_t be two independent sets of size k and X be a subset of vertices such that $G \quad X$ contains many connected components. Is it possible to remove one of the components while preserving the existence of a transformation from I_s to I_t?

Since a token might perform an arbitrarily long walk in a component, it is not simple to prove that one of these components can be deleted to form an equivalent instance. With Valentin Bartier and Amer Mouawad, we introduced in [9] the notion of *types* of walks in a component H of $G \quad X$. The intuition behind the proof is that in a walk W of a token t in H, we can nd a subset of important vertices $x_1, \ldots, x_r$, called *con ict vertices*, such that we can express W as $x_1P_1x_2P_2 \ldots, P_{r\ 1}x_r$, where P_i is the path of W linking x_i to x_{i+1}. Then, in the walk W performed by t, the only important information is $N(x_i) \setminus X$ and $\bigcup_{u2P_i} N(u) \setminus X$. Thus, if we can prove that

the number of con ict vertices in every walk W is bounded with respect to k and jXj, the (potentially unbounded length of) information contained in a walk W can be summarized in $f(k, jXj)$ neighborhoods in X. We proved that the following holds:

Lemma 93 (Bartier, Bousquet, Mouawad [9]). *Let G be a graph, X be a subset of vertices, and I_s, I_t be two independent sets of G of size k. Let S be a transformation from I_s to I_t that minimizes the number of token slides involving a vertex of X. Almost all the components H of $G \quad X$ are well-behaved, i.e., the number of con ict vertices of a walk of a token t in H is bounded by a function of k and jXj.*

The proof requires showing that if a walk W of a token t in H has too many con ict vertices, then we should have, at some point, projected a token $t' \in t$ on a component of $G \quad X$ where we can mimic the behavior of the walk W of t in order to decrease the number of token slides involving a vertex of X. Since, we can determine the well-behaved components of $G \quad X$ in FPT-time and since the number of types of walks is bounded, if $G \quad X$ contains too many components, we can safely remove one of them.

To remove a component H of $G \setminus X$, one needs to prove that the token slides we can perform in H can also be performed in another component H' of $G \setminus X$. Let v be a vertex of H and assume that there is a token t that is *projected* on v at some point of the recon guration sequence, meaning that the token t is moved from a vertex of X to v. This token may stay a few steps on v, move to some other vertex w of H, and so on until it eventually goes back to X. Let this sequence of vertices (allowing duplicate consecutive vertices) be denoted by $v_1 = v, v_2, \ldots, v_r$. We call this sequence the *journey* of v.

If we can perform the same journey[16] on a component H', then we keep the existence of a transformation even if H is removed. However, the length of that journey might be arbitrarily long and then we are not sure that H' exists even if $V \setminus X$ contains arbitrarily many connected components.

However, one can wonder what is really important in the sequence $v = v_1, v_2, \ldots, v_r$? Why do we go from v_1 to v_r? Why so many steps in the journey if r is large? To answer these questions, we distinguish two cases.

First, suppose that in the recon guration sequence, the token t was projected from X to v, performed the journey without having to \wait" at any step (so no duplicate consecutive vertices in the journey), and then was moved to a vertex $x' \in X$. In that case, the journey can be summarized as (1) a vertex whose neighborhood in X is $N(v) \setminus X$, (2) a path that avoids the vertices Y of X with a token and, (3) a vertex whose neighborhood in X is the neighborhood in X of the last vertex v_r of the journey of v. If a

[16] Let us forget about tokens on $V(G) \setminus X$ on this outline and only consider edges between the token t and tokens in X.

component H^0 contains a path with these properties (of any possible length, shorter or longer) then we can replace the journey of v in H by a journey in H^0.

Though, we might not be able to go \directly" from $v_1 = v$ to v_r. Indeed, at some point in the sequence, there might be a vertex v_{i_1} whose neighborhood in X currently contains a token. This token will eventually move (since the initial journey with t in H is valid), which will then allow the token t to go further on the journey. But then again, either we can reach the final vertex v_r or the token t will have to wait on another vertex v_{i_2} for some token on X to move, and so on (until the end of the journey). We say that there are *conflicts* during the journey[17].

So we can now compress the journey as a path from v_1 to v_{i_1}, then from v_{i_1} to v_{i_2} (together with the neighborhood in X of these paths), as we explained above. So we can compress a journey into a sequence whose length is essentially the number of conflicts. Bartier et al. proved in [8] that, if there exists a transformation, there exists a transformation where journeys have few conflicts. So the number of types of relevant journeys we can perform in each component is bounded. And then, if there are too many components, one can prove that one of them can be removed since the types of journeys we can perform in them can actually be performed in others.

Applications and open questions. Using these ingredients, i.e., galactic graphs and types, Bartier et al. [8] proved that $\mathrm{TS\text{-}ISR}$ is fixed-parameter tractable on graphs of bounded degree, planar graphs, and chordal graphs of bounded clique number. The main ingredient of these proofs entails proving that, using the multi-component reduction described as well as additional reduction rules, we can reduce the maximum degree to $f(k)$. Then Theorem 92 ensures that $\mathrm{TS\text{-}ISR}$ is fixed-parameter tractable for all these classes. Note that the idea of reducing the degree has also been used in [4] to obtain fixed-parameter tractable algorithms for $\mathrm{TS\text{-}ISR}$ on graphs with girth constraints. The authors in [6] also used these ingredients to show that $\mathrm{TS\text{-}ISR}$ is fixed-parameter tractable on graphs of girth five or more (the problem is W[1]-hard on graphs of girth four or less [4]).

Natural next questions to consider are the following:

Question 94. *Is* $\mathrm{TS\text{-}ISR}$ *parameterized by* k *fixed-parameter tractable on graphs of bounded treewidth? On minor-free graphs? On* d*-degenerate graphs?*

[17] Actually, there might exist another type of conflict we do not explain in this outline for simplicity.

4.6 Diameter of the configuration graph and shortest transformations

Shortest transformations. In general, it has been proven that (unsurprisingly) it is not possible to approximate within a sublogarithmic-factor, unless P = NP, the length of a shortest transformation between two maximum independent sets even restricted to line graphs [40]. (Recall that deciding the existence of a transformation can be decided in polynomial time for line graphs).

There is a line of research trying to determine in which graph classes it is possible to find a shortest transformation in polynomial time. In the Token Sliding variant, Hoang et al. proved in [94] that it is the case in spiders[18]. Similar results have been proven for some subclasses of chordal graphs such as proper interval graphs, trivially perfect graphs, caterpillars in [136]. (Recall that since all these classes are chordal, we know that there is a simple optimal solution in the token jumping variant).

Diameter of the configuration graph. While most of the works for the reconfiguration of independent sets are algorithmic, there are few results providing upper bounds on the diameter of the configuration graph of the k-independent sets. Demaine et al. proved in [63] that, when there exists a transformation, a transformation of quadratic length exists for TS-ISR in trees. Brianski et al. proved in [50] that, when it exists, there exists a TS-transformation of length at most $O(k \quad n^2)$ in an interval graph. They moreover proved that there exist interval graphs and pairs of independent sets for which a shortest transformation between these independent sets has length at least $(k^2 \quad n)$ (which gives an asymptotically tight bound when $k = \quad (n)$). As far as I know, the following is open:

Question 95. *For any xed k, can a connected component of $G_{TS}(G,k)$ have diameter superlinear when G is an interval graph? A chordal graph?*

Even in general, it is not clear how to find non linear transformations. Brianski et al. [50] mentioned that "*[they] are not aware of any example giving a superlinear lower bound on the length of a reconfiguration sequence when the number of tokens is constant - even on the class of all graphs. Specifically, the case k = 2 remains open*".

Let $D(n,k)$ be the maximum diameter of a connected component of $G_{TJ}(G,k)$ over all the graphs G on n vertices. With Bastien Durain, Theo Pierron and Stephan Thomasse we answered that question and proved that the following holds. The lower bound construction also holds for Token Sliding since independent sets are of maximum size in the construction.

[18] Spiders are also called subdivided stars and form a subclass of trees.

Theorem 96 (Bousquet, Durain, Pierron, Thomasse [37]). *The following holds:*

1. *A connected component of $G_{TJ}(G,k)$ has diameter at most $\binom{n}{k-1}$.*
2. *$D(n,2) = n - 2$.*
3. *For every $k \geq 3$, $D(n,k) = o(n^{k-1})$.*
4. *$D(n,3) = \Omega(n^2/e^{O(\sqrt{\log n})})$.*
5. *For every k, $D(n,k) = \Omega(n^{2bk=3c}/e^{O_k(\log n)})$.*

Sketch of the proof. **(1)** The proof of this first point holds easily when we remark that we can label the edges of the configuration graph with independent sets of size $k-1$ corresponding to the intersection between the two independent sets associated to the endpoints of the edge. The conclusion follows from the fact that, in a shortest transformation, there is no repetition of labels (otherwise we can find a shortcut).

(2) The proof is inspired from the upper bound proof of the $(6,3)$-problem and is based on an application of the hypergraph removal lemma. A hypergraph H is (s,t)-*free* if no set of s vertices of H contains at least t hyperedges. The $(6,3)$-problem (or Ruzsa–Szemeredi problem) asks for the maximum number of hyperedges in a $(6,3)$-free n-vertex 3-uniform hypergraph. The so-called $(6,3)$-theorem of Ruzsa-Szemeredi [120] ensures this value is $o(n^2)$.

Let us give an idea of the proof for $k = 3$. We assume by contradiction that there exists a graph G and two independent sets I, J such that the length of a shortest transformation P between I and J has length $\Omega(n^2)$. We consider the 3-hypergraph restricted to the odd independent sets in P. The $(6,3)$-theorem ensures that there exists a subset of 6 vertices containing 3 independent sets in the sequence. We prove that in such a case, it would be possible to find a shortcut in P.

(3) Assume that n is a prime integer and let us denote by $v_0, \ldots, v_{n-1}$ the vertices of the graph. The idea of the proof consists in starting from a clique and little by little removing edges. The goal is to remove edges to create almost linearly many paths in the configuration graph of linear length while keeping these paths complete to each other in the configuration graph (i.e. we do not want to be able to pass through an independent set of a path to another by changing a vertex).

In order to create a path of linear length, one can do the following: select an integer p and, for every vertex v_i, remove the edges v_iv_{i+p} and v_iv_{i+2p} (where all the indices have to be understood modulo n). One can easily remark that this gives a cycle of independent sets of size 3, namely: $(0,p,2p)$ $(p,2p,3p)$ $(2p,3p,4p)$... So we can create $n-1$ such cycles for every integer p.

The involved part of the proof consists in showing that there exists an almost linear subcollection of these cycles that remain independent of each other (i.e. there is no edge between them in the con guration graph). To do so, we restrict the collection of paths to a subset of integers p that form a set S of integers with no arithmetic progression of size 3 and such that all the integers of S have value 1 mod 4. A well known result of number theory [11] ensures that there exists such a set of density at least $1/e^{O(\log n)}$. Using these conditions, we can easily prove that the independent sets of size 3 are exactly the ones described above and two independent sets of size 3 are adjacent only if the condition above is satis ed.

We nally use a last trick consisting of adding a linear number of vertices which permits to glu these cycles[19] together in order to obtain the claimed diameter.

(4) To illustrate the technique, let us give a sketch of a (false) proof providing an intuition of the proof. Let G be a graph such that $G_{TS}(G,k)$ has diameter R. Then, we can construct a graph G' on $|V(G)|+7$ vertices such that $G_{TS}(G',k+2)$ has diameter larger than $2R$. The graph G' consists of a copy of G plus 7 vertices X which induce a antipath. Let A,B be two independent sets of G at distance R. We connect the rst and last vertices of X to $V(G)\setminus A$ and we connect the 4-th vertex of X to $V(G)\setminus B$. The idea is that in order to move the leftmost vertex of X to the rightmost vertex, we should perform the transformation from A to B in G (in order to reach the fourth vertex) and move it back to A in order to reach the last vertex. In some sense in order to pass through the fourth vertex, we have to pay a toll (a transformation from A to B).

The proof of (4) consists in proving that the structure of the construction of (3) is strong enough to replace the antipath in the construction above by the graph of (3). □

Note that the value $n/e^{O(\sqrt{\log n})}$ corresponds to the largest known asymptotic size for a subset of $[1,n]$ without arithmetic progressions of length 3 [11]. Any improvement of this bound would also imply an improvement of the bound of Theorem 96(3). Note that the best bound for the $(6,3)$-problem also has this order of magnitude [120].

Let us nish this part with the following very exciting question we did not succeed to solve:

Conjecture 97.

$$D(n,4) \quad n^{3\ o(1)}.$$

The only best lower bound is given by Theorem 96 which is subquadratic. More generally one can wonder what is the diameter of $D(n,k)$ for any k 5.

[19] That we can easily cut in order to get paths instead.

Chapter 5

Other reconfiguration problems

Many other recon guration problems have been studied in the last 15 years. The goal is not to give an extensive survey of all of them. I prefered focusing only on a few of them who would probably deserve their own chapter in an ideal world where the amount of time devoted to writing a habilitation is in nite.

5.1 Reconfiguration and matroids

There is a natural relation between recon guration and matroids. A matroid can be seen as an object generalizing both spanning trees of graphs and linearly independent families of a vector space (among others). Formally, a *matroid* M is a pair (X, I) where X is a set called the *ground set* and I is a set of subsets of X (called *independent sets*) satisfying the following property:

1. The empty set is independent, i.e. $; 2 I$.
2. (Hereditary property) Every subset of an independent set is independent, i.e. for every $I 2 I$ and $J \quad I$, we have $J 2 I$.
3. (Exchange property) For every $I, J 2 I$ such that $jIj > jJj$, there exists $i 2 I$ such that $J [fig 2 I$.

The exchange property ensures that all the inclusion-wise maximal independent sets have the same size (and are then maximum). An independent set of maximum size of a matroid M is called a *base* of M. Note that, for linearly-independent families of vectors, the exchange property is known as incomplete base theorem. And for spanning trees it simply says that, for every non-connected forest F and every spanning tree T of a graph G, one can indeed add an edge of T in F without creating any cycle.

We say that two bases B, B' of a matroid are *adjacent* if they di er on exactly two elements, that is $jB \setminus B'j = jB' \setminus Bj = 1$. One can now wonder if there exists a (TJ)-recon guration sequence from A to B. Given two sets A, B, we de ne the *symmetric di erence* (A, B) as the set $(A \setminus B) \cup (B \setminus A)$. We denote by $d(A, B) := j (A, B)j/2$. It is well known there always exists a transformation between two bases A, B of a matroid using $d(A, B)$ steps via an iterative application of the exchange property:

Lemma 98. *Let M be a matroid and I, J be two bases of M. One can transform I into J via a sequence of at most $d(I, J)$ exchanges.*

Proof. Let A, B be two bases. Let $a \in A \setminus B$ and $A' = A \setminus a$. The exchange property ensures that there exists an element $b \in B$ such that $A' \cup b$ is independent. Thus, in one step, we can reduce the symmetric di erence between the two bases by 2 and the conclusion follows by induction. □

One can then naturally wonder what happens if we consider generalization of matroids.

Matroid intersection. Let M_1 and M_2 be two matroids on the same ground set. We can de ne the *intersection of M_1 and M_2 denoted by $M_1 \cap M_2$* as the subsets X such that X is both independent in M_1 and M_2. We can similarly de ne the intersection of three or more matroids.

One can naturally wonder if, using the exchange rule, we can always transform any independent set of $M_1 \cap M_2$ into any other using the exchange rule. The answer is negative. Intersection of two matroids generalizes many natural problems such as maximum matchings, maximum weight matchings in bipartite graphs and arborescences in directed graphs. Edmonds proved that the largest independent set in the intersection of two matroids can be computed in polynomial time.

One can then naturally ask the following question:

Question 99. *Can* Matroid Intersection Reconfiguration *be decided in polynomial time when we consider the intersection of two matroids? More?*

As far as I know, this question is still open for two matroids. When we consider the intersection of at least 3 matroids, Mühlenthaler proved[1] that the following holds:

Theorem 100 (Mühlenthaler)**.** Matroid Intersection Reconfiguration *is PSPACE-complete when we consider the intersection of three matroids.*

[1] private communication.

One can now wonder when a transformation exists. White conjectured more that 40 years ago that the following holds:

Conjecture 101 (White [133])**.** *For every matroid M, there exists a reconfiguration sequence between any pair of bases in the intersection of the matroid M and its dual*[2]*.*

The conjecture is known to be true since 1985 for graphic matroids [73] (spanning trees) and it has been very recently extended to regular matroids [14] or union of arborescences [105]. For an up to date complete bibliography on the conjecture, the reader is for instance referred to the introduction of these two papers.

Parity matroid reconfiguration. Let $M = (X, I)$ be a matroid. Suppose that the ground set X of M is partitioned into pairs, called *lines*. In other words, every $v \in S$ has a unique *mate* v such that $\{v, v\}$ is a line. Let us denote by L the set of lines. A set $T \subseteq L$ of lines is called an *(independent) parity set* if $\bigcup_{\ell \in T} \ell \in I$, where we denote $I(T) := \bigcup_{\ell \in T} \ell$. In other words, it is a set of lines such that the union of the elements belonging to these lines is an independent set of the matroid M.

The problem of finding an independent parity set of maximum cardinality is called the *Matroid Parity problem* [108]. Matroid parity generalizes in particular the matroid intersection problem. This problem can be solved in polynomial time for linear matroids [114], while it requires an exponential number of independence oracle calls in the general case.

We consider the problem of the reconfiguration of independent parity sets, which we call Matroid Parity Reconfiguration. We say that two independent parity sets I and I' are *adjacent* if there exist a line ℓ in I and a line ℓ' in I' such that $I' = (I \setminus \ell) \cup \ell'$. Contrary to Matroid Reconfiguration, it is not always possible to find a transformation between any pair of parity sets. Actually, Matroid Parity Reconfiguration is hard in general. Indeed, since a reduction in [126] gives an exact correspondence between the independent sets of size k of a given graph and the independent parity sets of a Matroid Parity instance, Matroid Parity Reconfiguration is PSPACE-complete since Independent Set Reconfiguration also is [90]. One can then ask the following:

Question 102. *When can* Matroid Parity Reconfiguration *be decided in polynomial time?*

A simple case we were able to solve is when parity sets are not maximum.

Theorem 103 (Bousquet, Hommelsheim, Kobayashi, Mühlenthaler and Suzuki [42])**.** Matroid Parity Reconfiguration *is always positive when we transform*

[2] The dual of M has the ground set of M and I is an independent set of the dual if and only if M has a base such that $M \setminus I = \emptyset$.

a non maximum independent parity set into another. In other words, it is always possible to transform a non-maximum parity set any other.

So the problem remains open only for maximum parity sets. Note that this behavior is quite di erent with Independent Set Recon guration which is usually simpler in the maximum case, at least from restricted instances (as we have already discussed in Chapter 4).

In the rest of this section, we focus on a particular case of parity matroids: feedback vertex set on subcubic graphs. A *feedback vertex set* of a graph is a subset of vertices whose deletion leaves in an acyclic graph. Deciding whether a graph has a feedback vertex set of a given size is an NP-complete problem even for planar graphs of maximum degree four [127]. Interestingly, for graphs of maximum degree at most three, also called *subcubic* graphs, a minimum Feedback Vertex Set can be computed in polynomial time by a reduction to a tractable special case of the Matroid Parity problem [114, 130]. Despite a lot of work, we were not able to solve the following question which is a particular case of Question 102:

Conjecture 104. TJ-FEEDBACK VERTEX SET RECONFIGURATION *(FVSR) on subcubic graphs can be decided in polynomial time.*

Together with Yusuke Kobayashi, Felix Hommelsheim, Moritz Mulenthaler and Akira Suzuki we were able to prove that the con guration graph of the feedback vertex sets is connected as long as G is a cubic $K_{3,3}$-minor free graph. And we also proved that the problem TJ-FVSR becomes PSPACE-complete as long as the maximum degree is at least 4.

Theorem 105 (Bousquet, Hommelsheim, Kobayashi, Mulenthaler, Suzuki [42])**.** TJ-FVSR *is always positive for subcubic $K_{3,3}$-minor free graphs.*

Sketch of the proof. The proof follows from a simple but nice argument. Let G be a graph and X be a feedback vertex set of G. Let x be a vertex of X. Then one can easily prove (depending on the number of connected components of $G' := G[V \setminus (X \setminus x)]$) the following property ($\star$): we can remove the vertex x from the feedback vertex set, the set of vertices that can replace x is exactly:

- either a single vertex y_x (if G' has a single connected component),
- or all the vertices of a cycle C (if G' has two components),
- or all the vertices of the graph (minus X) otherwise.

We moreover show that since G is $K_{3,3}$-minor free then y_x and $y_{x'}$ should be di erent for any pair of vertices $x, x' \in X$.

The goal now simply consists in proving that we can either nd a greedy move (we can reduce the symmetric di erence between the source and target

FVS X and X') or we can find a *detour-greedy move* meaning that we can move a vertex of the source and target FVS to the same vertex.

One can remark that if we are in the first two cases of ($\star$) then we indeed have a greedy move. So we can assume that, for every $x \in X$ and $x' \in X$, we are in the third case. If $y_x = y_{x'}$ for some $x \in X$ and $x' \in X$ then we indeed have a detour greedy move. And we can prove that it has to hold since a cubic graph has at less than $4|X|$ vertices. □

One can wonder if this statement can be generalized further. The answer is negative since it does not hold for $K_{3,3}$: the configuration graph is not connected (it is not possible to move the FVS consisting of two vertices on the left side to the FVS consisting of two vertices on the right side).

Other generalizations of matroids There are many other ways to relax matroid constraints, and in particular spanning trees (that are graphic matroids). One of them is to ask for more combinatorial structure on their shape. We explored this direction in two articles with Takehiro Ito, Yusuke Kobayashi, Haruka Mizuta, Paul Ouvrard, Akira Suzuki and Kunihiro Wasa in [43, 44]. In these two articles we studied the complexity of the transformation between two spanning trees via flips when we moreover add conditions on the intermediate trees, for instance on their minimum or maximum degrees, on their number of leaves or on their diameter.

I will not mention these results in detail (many intersecting research directions have not been explored at all). I will only mention one open problem we were not able to solve that is surprisingly simple:

Conjecture 106. HAMILTONIAN PATH RECONFIGURATION *is PSPACE-complete.*

In the HAMILTONIAN PATH RECONFIGURATION problem, we are given a graph G and two hamiltonian paths of G. The goal is to determine if we can transform the first into the second while keeping hamiltonian paths all along the transformation. At each step, we can remove an edge of the current path P and add a new edge that belongs to G.

It is known that HAMILTONIAN CYCLE RECONFIGURATION is PSPACE-complete. At first glance, one may think that it should (almost) directly imply the hardness for the path version. However, the path version is, when we think about it, much harder than its cycle counterpart. Indeed, in the cycle version, we are authorized to replace a pair of edges by another one. In other words, a flip for the hamiltonian cycle problem consists in removing two edges and adding two others that form a C_4 altogether. This operation is natural when one wants to keep the degree of the underlying structure (a similar flip operation was introduced in [21] for perfect matchings).

On the contrary, for HAMILTONIAN PATH RECONFIGURATION, we are (informally speaking) allowed to change one edge at a time. We claim that

it makes the problem much harder to manipulate. Indeed, imagine now that we want to design a hardness reduction from NCL to HAMOLTONIAN PATH RECONFIGURATION. We have to create AND and OR gadgets and, usually in the reductions, a single step modi cation can be interpreted as a modi cation within a gadget that \ ips" an edge (or does nothing).

In the reduction from NCL to HAMILTONIAN CYCLE RECONFIGURATION, the graph is built in order to ensure that every possible edge ip will be within a gadget. Rephrased di erently, there are many pairs of edges in the hamiltonian cycle but most of them do not induce a C_4 and the ones that belong to a C_4 lie in the same gadgets (which ensures that we can interpret in a coherent way this ip in the NCL instance).

For HAMILTONIAN PATH RECONFIGURATION, the reduction should be quite di erent. Indeed we cannot ip *any* pair of edges but only choose one of them. Indeed, in order to keep a path, the created edge has to be attached to one of the endpoints of the current hamiltonian path. This operation is highly non local (the endpoint has no reason to belong to the gadget where we want to perform a ip in the NCL transformation for instance); thus the construction of a hardness proof seems much more complicated.

Note that it is not even known if HAMILTONIAN PATH RECONFIGURATION is NP-hard.

5.2 Dominating set reconfiguration

In this section, we will simply brie y overview some results on dominating set recon guration. In particular, I will not explain most of the proof techniques used to obtain the results in that chapter as I did for recoloring or independent set recon guration.

A *dominating set* of a graph G is a subset of vertices X such that all the vertices of V are either in X or adjacent to a vertex of X. As for independent set recon guration, we can see the vertices of the dominating set as tokens. We can naturally de ne three recon guration rules leading to three models. In the *Token Sliding model*, the recon guration step consists in sliding a token along an edge of the graph. In the *Token Jumping model*, the recon guration step consists in moving a token somewhere else in the graph. In the *TAR-model*, we are allowed to remove or add a vertex to the dominating set at each step. As for Independent Set Recon guration, the TAR and TJ models have been proven to be equivalent. However, in order to prove structural results it is often easier to manipulate the TAR model than its TJ counterpart.

For dominating set recon guration both structural and algorithmic aspects of recon guration have been studied. Let us slightly discuss both of them and mention a few open problems in this area.

5.2.1 Structural aspects of DSR

Recall that two dominating sets X, Y are *TAR-adjacent* if their symmetric difference has size one, that is $|X \triangle Y| = 1$ or $|Y \setminus X| = 1$. In other words, Y can be obtained from X by removing or adding exactly one vertex. One can indeed always transform a dominating set D_s into another one D_t if we do not bound the size of the intermediate solutions: we first add one by one all the vertices in $D_t \setminus D_s$ to D_s, and then remove vertices of $D_s \setminus D_t$ one by one.

Let $G = (V, E)$ be a graph, and k be an integer. The *k-TAR-DS-configuration graph*, denoted by $G_{TAR}(G, k)$, is the graph whose vertices are the dominating sets of G of size at most k, with the TAR-adjacency defined above. There exists a reconfiguration sequence between two dominating sets D_s and D_t both of size at most k with threshold k on the size of the dominating sets all along the transformation if and only if there is a path in $G_{TAR}(G, k)$ between D_s and D_t.

Haas and Seyffarth proved in [84] that being reconfigurable in the TAR-model is not a monotone property, which means that if $G_{TAR}(G, k)$ is connected then $G_{TAR}(G, k+1)$ is not necessarily connected. Indeed, let us denote by $K_{1,n}$ the star graph on $n+1$ vertices. They observed that, for every $n \geq 3$, $G_{TAR}(K_{1,n}, k)$ is connected if $1 \leq k \leq n-1$ but that $R_n(K_{1,n})$ is not since the dominating set of size n which contains all the degree-one vertices is frozen, i.e. it is an isolated vertex in $G_{TAR}(K_{1,n}, n)$. They then asked what is the smallest integer d_0 such that $G_{TAR}(G, k)$ is connected, for any $k \geq d_0$.

Haas and Seyffarth identified a parameter of interest for that question which is $\Gamma(G)$ the maximum size of a dominating set which is minimal by inclusion. In particular Haas and Seyffarth proved that, for every $k \geq \Gamma(G) + 1$, if $G_{TAR}(G, k)$ then $G_{TAR}(G, k+1)$ also is. Another parameter of interest is $\gamma(G)$ which is the minimum size of a dominating set. Haas and Seyffarth proved in [84] that the following holds:

Lemma 107 ([84]). *Let G be a graph with at least two independent edges. If $k \geq \min\{n-1, \Gamma(G) + \gamma(G)\}$ then $G_{TAR}(G, k)$ is connected.*

Sketch of the proof. The idea of the proof consists in proving that, since every dominating set of size larger than $\Gamma(G)$ is not minimal by inclusion, we can reduce its size to get at least γ free tokens[3]. Now one can easily prove that, using these additional tokens we can reach any dominating set of size γ which completes the proof. □

Haas and Seyffarth also showed in [84] that this value can be lowered to $\Gamma(G) + 1$ if G is bipartite or chordal. This result is tight since $\Gamma(K_{1,n}) = n$.

[3] We say that we have a *free token* if the size of the current dominating set is smaller than k.

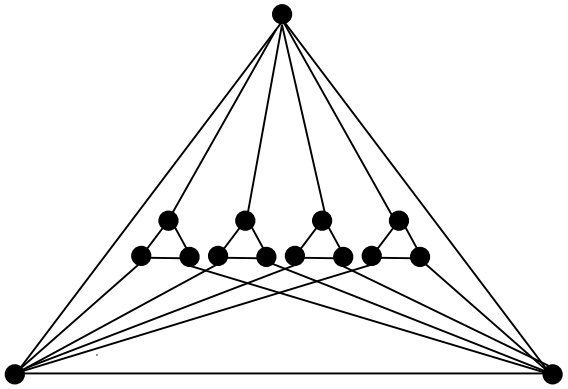

Figure 5.1: An example of graph where $G_{TAR}(G,\ (G)+1)$ is not connected.

This result is tight since $K_{1;n}$ is bipartite and chordal and $G_{TAR}(K_{1;n}, k)$ is not connected. One can then naturally wonder if $G_{TAR}(G,\ (G)+1)$ is connected for every graph (a natural way to prove it would be to mimic the proof of Lemma 107 to reduce the symmetric di erence). Suzuki et al. [128] answered negatively this question by constructing an in nite family of graphs for which $G_{TAR}(G,\ (G)+1)$ is not connected even for planar graphs (see Figure 5.1 for an example of a (non-planar) counter-example). On the positive side, we proved with Alice Jo ard and Paul Ouvrard that $G_{TAR}(G,\ (G)+3)$ is connected for planar graphs. The only case that remains open is $(G)+2$ for which the following has been conjectured:

Conjecture 108 (Bousquet, Jo ard, Ouvrard [46])**.** *For every planar graph* G*,* $G_{TAR}(G,\ (G)+2)$ *is connected.*

Mynhardt et al. [118] strengthened the construction of Suzuki et al. by constructing an in nite collection of graphs with arbitrary 3, arbitrary domination number in the range $1\ \ \gamma\ \ 1$ and for which $G_{TAR}(G,\ (G)+\gamma(G)\ \ 1)$ is not connected for every G in this collection. So Lemma 107 is optimal and cannot be improved in general.

We will describe a slightly weaker construction (also in [118]) which will also allow us to derive other bounds later (see Figure 5.2 for an illustration). They also constructed an in nite family of graphs $G_{`;r}$ (with $`\ \ 3$ and $1\ \ r\ \ `\ \ 1$) for which $2\ (G_{`;r})\ \ 1$ tokens are necessary to guarantee the connectivity of the TAR-DS-con guration graph. Let us describe their construction when $r = `\ \ 1$. The graph $G_{`;`\ 1}$ contains $`\ \ 1$ cliques $C_1, C_2, \ldots, C_{`\ 1}$ called *inner cliques*, each of size $`$. We denote by c_i^j the j-th vertex of the clique C_i. We then add a new clique C_0 of size $`$, called the *outer clique* and we add a new vertex $u_0$ adjacent to all the vertices of $C_0$ (hence, $C_0$ can be seen as a clique of size $`+1$). For every $1\ \ i\ \ `\ \ 1$ and for every $1\ \ j\ \ `$, we add an edge between c_i^j and c_0^j. This completes the construction of $G_{`;`\ 1}$. Mynhardt et al. [118] showed that $(G_{`;`\ 1}) = `$.

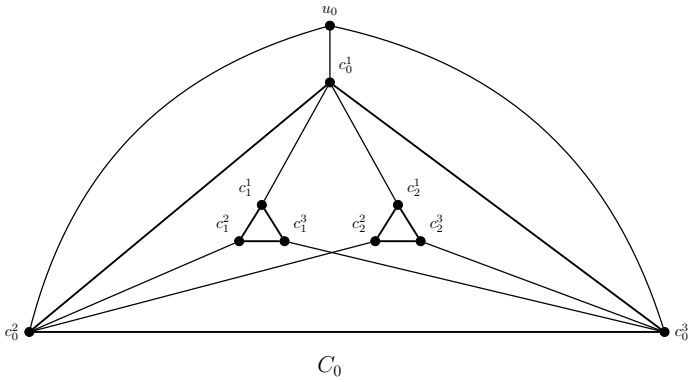

Figure 5.2: The graph $G_{3,2}$

On the positive side, Haas and Sey arth [85] proved that if $k =$ (G) + $\alpha(G)$ 1 (where $\alpha(G)$ is the size of a maximum independent set of G), then $G_{TAR}(G,k)$ is connected. To obtain this result, they proved that all the independent dominating sets of G are in the same connected component of $G_{TAR}(G,$ $(G)+1)$. With Alice Jo ard and Paul Ouvrard, we showed in [46] that if $k =$ (G) + $\alpha(G)$ 1, then $G_{TAR}(G,k)$ has actually linear diameter[4]. It contrasts in particular with a result of Suzuki et al. [128] who provided an in nite family of graphs G for which $G_{TAR}(G,\gamma$ + 1) has exponential diameter.

We showed in [46] that $G_{TAR}(G,k)$ is connected and has linear diameter for some "minor sparse classes"[5]. In particular, we obtained as a corollary that $G_{TAR}(G,k)$ is connected and has linear diameter for $K_{\cdot}$-minor free graphs as long as k (G) + $O(\ell^{p}\overline{\log \ell})$. The gap between the lower and upper bound is not completely closed since the only lower bound we know is $(G)+\ell$ 4 We also studied the dependency on the treewidth and proved that the following holds:

Theorem 109 (Bousquet, Jo ard, Ouvrard [46]). *$G_{TAR}(G,$ $(G)+tw(G)+$ 1) has linear diameter.*

This bound is tight up to an additive constant factor using the construction of Mynhardt et al. [118] described above. Indeed, one can easily show that $G_{TAR}(G_{;;\ 1},2\ell$ 2) is not connected and that $G_{;;\ 1}$ has treewidth ℓ. So $G_{TAR}(G,$ (G) + $tw(G)$ 2) is not necessarily connected.

The pathwidth of $G_{;;\ 1}$ is at most 2ℓ 1. However, it is not clear if and how we can obtain a better upper bound for bounded pathwidth graphs. To sum up $G_{TAR}(G,k)$ is not necessarily connected if $k <$ $(G)+pw(G)/2+O(1)$ and is connected if $k >$ (G) + $pw(G)$ + 1. The following question is still open:

[4] The induction based proof of [85] does not provide a linear diameter.

[5] For a formal de nition, we refer the reader to [46].

Question 110 (Bousquet, Jo ard, Ouvrard [46]). *Is the k-TAR-DS-con guration graph connected when k (G) + $pw(G)/2$ + $O(1)$?*

5.2.2 Algorithmic aspects of DSR

Haddadan et al. [86] studied the algorithmic complexity of TJ-DSR problem. They proved that, given a graph G, two dominating sets D_s and D_t of G and an integer k maxfjD_sj, jD_tjg, it is PSPACE-complete to decide whether there exists a path in $G_{TJ}(G, k)$ between D_s and D_t. Actually, this problem remains PSPACE-complete even restricted to bipartite graphs or split graphs. On the other hand, they proved that this problem can be decided in linear time if the input graph is a tree, an interval graph or a cograph.

On general graphs, $\mathrm{TJ\text{-}DSR}$ is W[2]-hard parameterized by solution size k plus the length of a recon guration sequence ℓ [117]. When parameterized by ℓ alone the problem is xed-parameter tractable on any class where rst-order model-checking is xed-parameter tractable, the most general known cases being nowhere dense classes [82] and classes of bounded twinwidth (assuming a contraction sequence is also given as part of the input) [30]. To demonstrate why, we formulate the $\mathrm{TJ\text{-}DSR}$ problem as follows. We encode the instance (G, k, D_s, D_t) as a colored graph, where the vertices of D_s and D_t are marked by two unary predicates, so that they become accessible to rst-order logic. First-order logic now existentially quanti es the at most ℓ vertices that are changed in a recon guration sequence of length ℓ. Now, it remains to verify that the vertices marked with the rst predicate modi ed by the quanti ed changes form a dominating set. We note that xed-parameter tractability using rst-order model-checking for parameter ℓ also holds for $\mathrm{ISR\text{-}TJ}$, $\mathrm{ISR\text{-}TS}$, and $\mathrm{DSR\text{-}TS}$.

Unfortunately, the approach via rst-order model-checking does not immediately help us to deal with the parameter k. For the parameter ℓ, we can nd a xed sentence expressing the existence of a recon guration sequence of length ℓ. However, when restricted to the parameter k alone, because recon guration sequences can be arbitrarily long, it is not possible to state the existence of such a sequence in rst-order logic.

Instead, the key tool to tackle $\mathrm{TJ\text{-}DSR}$ is based on the notion of domination cores. A *k-domination core* in a graph G is a subset Y $V(G)$ such that every set of size at most k dominating Y also dominates the whole graph (see Figure 5.3).

For a graph G, we x source and target dominating sets D_s and D_t of size k and a k-domination core Y. We let R denote a set containing one vertex from each projection class of C_Y. We call a vertex in R a *representative* of its class.

Let H be the graph induced by the vertices of Y [D_s [D_t [R. Observe that $N_G(v) \setminus Y = N_H(v) \setminus Y$ for all v 2 $V(H)$. For v 2 $V(G)$, let $v_H = v$ if v 2 D_s [D_t [Y and otherwise let v_H be the vertex of R representing the

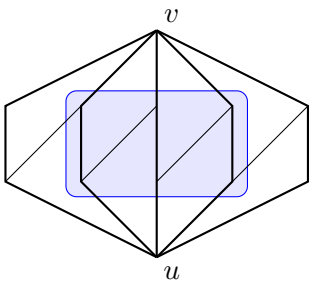

Figure 5.3: A graph with a dominating set of size 2 and a 2-domination core marked by a light-blue box. The only way to dominate the core by 2 vertices is to pick vertices u and v.

projection class of v. For $w \in V(H) \setminus (Y \cup D_s \cup D_t)$ x an arbitrary vertex $w_G \in V(G)$ with $N_H(w) \setminus Y = N_G(w_G) \setminus Y$.

Lemma 111. *If $D_G \subseteq V(G)$ is a dominating set of G, then $D_H = \{v_H : v \in D_G\}$ is a dominating set of H. Conversely, if $D_H \subseteq V(H)$ is a dominating set of size at most k of H, then $D_G = \{w_G : w \in D\}$ is a dominating set of G.*

The first statement of the lemma follows from H being an induced subgraph of G. For the second statement, since D_H dominates H, it in particular dominates Y in H. Because H is an induced subgraph of G, the set D_G dominates Y also in G. By definition of a k-domination core, D_G dominates G.

The use of k-domination cores for $\mathrm{TJ\text{-}DSR}$ is now immediate.

Lemma 112. *There exists a reconfiguration sequence from D_s to D_t in G if and only if there exists a reconfiguration sequence from D_s to D_t in H.*

Since there are at most $2^{|C|}$ different projection classes, it follows that the graph H is small in relation to $|C|$. Even better bounds can be achieved when the VC-dimension of the graph class under consideration is bounded [123, 124].

Lemma 113. *The graph H has at most $2k+|C|+2^{|C|}$ vertices. Furthermore, when the VC-dimension of G is bounded by d, then H has at most $O(k+|C|^d)$ vertices.*

Now it is easy to derive fixed-parameter tractable algorithms for all classes of graphs that admit small domination cores.

Theorem 114. *Assume C is a class of graphs such that for every $G \in C$ and for every $k \in \mathbb{N}$ there exists a polynomial-time computable k-domination*

core of size $f(k)$, for some computable function f. Then TJ-DSR *is fixed-parameter tractable on C with parameter k.*
Furthermore, when f is a polynomial function and C has bounded VC-dimension, then TJ-DSR *admits a polynomial kernel on C.*

Domination cores were first introduced by Dawar and Kreutzer [62] in their study of the DISTANCE-r DOMINATING SET problem on nowhere dense classes. When k is the size of a minimum dominating set, then linear k-domination cores (i.e. cores of size $O(k)$) are known to exist on classes with bounded expansion [64] and polynomial (and in fact almost linear) cores exist on nowhere dense classes [106, 69]. When k is not minimum, in all of these cases k-domination cores are known to be of polynomial size. The most general classes of graphs known to admit domination cores are semi-ladder-free classes [72]. These classes include biclique-free classes, on which the cores are of polynomial size (with the exponent depending on the size of the excluded biclique) and which have bounded VC-dimension. Their use for reconfiguration was first observed in [112].

Corollary 115. TJ-DSR *is fixed-parameter tractable on every semi-ladder-free class of graphs.* TJ-DSR *admits a polynomial kernel on every biclique-free class of graphs.*

Question 116. *Does* TJ-DSR *admit a linear kernel on planar graphs?*

The methods based on domination cores are in fact limited to classes with bounded co-matching index (which in particular have bounded semi-ladder index). It is easily seen that the graph consisting of two cliques A, B of size n such that the edges between A and B induce a co-matching has a dominating set of size 2 but does not contain a non-trivial 2-domination core. For example, while it is known that the DOMINATING SET problem parameterized by k is fixed-parameter tractable on every class of bounded clique-width it is not known whether this is true for TJ-DSR.

Question 117. *Is* TJ-DSR *parameterized by k fixed-parameter tractable on graphs of bounded clique-width?*

The results of this section naturally generalize to the distance-r version of DOMINATING SET. Observe that for $r \geq 2$ the rth powers of classes with unbounded degree contain arbitrarily large cliques. However, for example powers of nowhere dense graphs still have bounded semi-ladder index [72], and hence, TJ-DSR is fixed-parameter tractable on powers of nowhere dense classes. One can obtain almost linear kernels when the underlying nowhere dense graph is given, or equivalently, when considering the reconfiguration variant of the DISTANCE-r DOMINATING SET problem on nowhere dense classes [125].

Token Sliding variant. As for Independent Set Recon guration, the sliding model is much harder to deal with than its jumping counterpart. As a consequence, TS-DSR has not been widely studied. Let us however mention a few results on the topic. When we consider TS-DSR we have to decide if several tokens can be on the same vertex at the same time (which cannot happen for ISR and can be avoided for TJ-DSR). All the results on the topic, as far as we know, authorize this. In other words, dominating sets are seen as multisets of vertices.

Bonamy et al. [25] proved that TS-DSR is PSPACE-complete, even restricted to split, bipartite or bounded treewidth graphs. They also provide polynomial time algorithms for cographs and dually chordal graphs (which contain interval graphs). We extended these result to circular interval graph with Alice Jo ard in [45]. We also proved in [45] that TS-DSR is PSPACE-complete on circle graphs which gives the rst class where the maximum dominating set problem is NP-complete but its TS-recon guration counterpart is polynomial. The parameterized compelxity of TS-DSR remains widely unexplored due to its complexity compared to its jumoing counterpart.

Chapter 6

Conclusion

In this manuscript, we overviewed existing results on Graph Recoloring via single vertex vertices (Chapter 3) and on Independent Set Reconfiguration under the token sliding and token jumping models (Chapter 4). We also briefly discussed some results on (Generalizations of) Matroid Reconfiguration and Dominating Set Reconfiguration (Chapter 5).

Let us conclude this manuscript with some open problems. I will first recall some class of questions already mentioned before that I find particularly interesting and challenging. Then I will discuss questions that were not yet discussed (or at least not too much) in this manuscript and which, I think, are both challenging and exciting for the future of reconfiguration[1].

Already mentioned open problems. All along the manuscript, we mention a lot of open problems which are more or less difficult. Repeating all of them does not really make sense. I would like to mention two types of questions that are particularly interesting in my opinion. First, there remain many exciting structural questions related to the diameter of configuration graphs. And, in particular, finding upper bounds on the number of colors that ensure configuration graphs to be of linear diameter. I think that these questions are interesting since answering them probably needs to introduce new proof techniques or to understand more precisely recent proof techniques. Another question of interest, which is probably the most studied question in graph recoloring, is the Cereceda's conjecture (Conjecture 9).

The second type of questions I would like to mention are algorithmic. As we saw in Chapter 4, while the parameterized complexity of the ISR is quite well understood in the TJ model, it is very poorly understood in the TS model. In particular, it would be interesting to understand it better for sparse graph classes where the problem is known to be FPT on planar

[1] The interested reader can indeed find a lot of other exciting questions all along the manuscript.

graphs [8] but where very few is known for larger classes. It really contrasts with the TJ-model where the problem is FPT on biclique-free graphs [47]. And we can make the same remark for dominating set recon guration! There also remains a lot of interesting questions for Token Jumping, especially for dense classes where almost no recon guration results have been obtained so far.

Lower bounds. All along this manuscript, we surveyed a lot of di erent proof techniques that had permitted us to provide e cient algorithms or upper bounds on the diameter of the con guration graph. Unfortunately, very few techniques permit to obtain non trivial lower bounds on the diameter of con guration graphs (with the exception of Theorems 10 and 96). Note that we know that con guration graphs must then have exponential diameter since the reachability problems are usually PSPACE-complete to decide[2]. However, very few constructions where the con guration graph is superpolynomial have been exhibited.

For graph coloring, Bonsma and Cereceda gave a construction in [31] of a graph where the shortest transformation between two colorings is exponential. (We did the same in Theorem 96 for independent set recon guration.) However, in that example of [31], the con guration graph is not connected. As far as I know, there does not exist any construction that ensures that the following holds:

Question 118. *Give a (or as many as possible) graph G and an integer k such that k-coloring con guration graph $G(G,k)$ is connected but has a super-polynomial diameter.*

If we are interested in a bit more ne-grained combinatorial results on the diameter, it would be interesting to nd a graph such that the following holds:

Question 119. *Give a (or as many as possible) graph G of degree at most $f(d)$ such that $G(G,d+2)$ has super-linear diameter (where d denotes the degeneracy of the graph).*

Positive answers to these questions might help us to understand the shape of bad examples which (i) can be useful to provide upper bounds by showing that they are indeed the worst examples and (ii) can be useful to design gadgets for hardness results for instance. More generally, any construction giving lower bounds in recon guration is interesting since we de nitively lack them!

[2] The fact that they are not in NP ensures that the shortest transformation is unlikely to be polynomial.

The curse of connectivity in reconfiguration. As we have already seen, reconfiguration problems are much harder in the Token Sliding model than in the Token Jumping model due to the bottleneck effect that prevents us for making simple moves even when the graph is very large. Using galactic graphs, we were able to give the first FPT algorithms for reconfiguration problems for TS-ISR but, in many graph classes, deciding if TS-ISR or TS-DSR is still open while FPT algorithms clearly exist for their jumping counterparts. Deciding if TS-ISR is really harder on these graph classes or if designing an FPT algorithm simply is more complicated is, by itself, an interesting question.

But we would like to discuss a related question here. When one wants to design algorithms for connected dominating set reconfiguration even on simple graph classes we end up with problems similar to the ones that we have for token sliding (in the sense that the structures and cases that block reductions are very similar). Actually, when we try to reduce the size of the instances, we face a challenging problem. Assume that the graph contains a long path of vertices of low degree only connected to high degree vertices. One would like to remove some of these vertices of the graph. Indeed, many of them have the same neighborhood in the set of really important vertices (that is vertices of large degree) and then, in terms of domination, some of them are redundant. However, we cannot simply remove these vertices since we might use this path in order to connect some other vertices (maybe the leftmost and rightmost vertices of the path really are important and the others are only there to ensure that we can slide the token from the left to the right vertex). So one can think that it may be a good idea to contract this path but unfortunately we cannot because the vertices of the path might not have the same neighborhood outside and we might create solutions while they do not exist.

For connected dominating sets exactly the same type of situations occur and we cannot neither delete vertices (we might break the existence of a connected dominating set using the contracted vertex) nor contract edges (it would allow us to use one less vertex).

It raises the following questions:

Question 120.
- *Is TS-DSR simpler, harder or incomparable with* TJ-Connected Dominating Set Reconfiguration*?*
- *Is* TS-Connected Dominating Set Reconfiguration *hard (parameterized by k) on a sparse graph class such as bounded treewidth? minor-free graphs?*

Since in the second problem we have a double connectivity condition, it might be simpler to design reduction rules to prove hardness results (which in turn might give us some further ideas for other problems).

Dense graph classes. All the results of parameterized complexity obtained and presented in this manuscript for both ISR and DSR have been obtained on sparse graph classes such as planar, minor closed or even bounded expansion. No result, positive or negative is known on their dense counterparts. For instance:

Question 121.
- *Is TJ-ISR FPT when parameterized by k on graphs of bounded clique-width?*
- *Is TJ-DSR FPT when parameterized by k on graphs of bounded twin-width?*

Proof techniques used for sparse classes usually consist in proving that if we have a large enough part of the graph that is far from tokens (or behave the same with respect to these tokens), then we can reduce that part. One of the reasons why we can reduce it (for ISR) is that it contains an insanely large independent set and then we have many similar choices to move a token in that section which allows us to remove redundancy. Such a technique indeed does not work for dense graph classes since a large fraction of the vertices does not necessarily contain a large indpeendent set. However if the large degree vertices behave \well-enough", it is likely that some part of the graph can be deleted again.

Bibliography

[1] A. Agrawal, R. K. Allumalla, and V. T. Dhanekula. Refuting FPT algorithms for some parameterized problems under Gap-ETH. In P. A. Golovach and M. Zehavi, editors, *16th International Symposium on Parameterized and Exact Computation, IPEC 2021*, volume 214 of *LIPIcs*, pages 2:1{2:12. Schloss Dagstuhl - Leibniz-Zentrum fur Informatik, 2021.

[2] V. E. Alekseev. The e ect of local constraints on the complexity of determination of the graph independence number. *Combinatorial-algebraic methods in applied mathematics*, pages 3{13, 1982.

[3] V. Bartier. *Combinatorial and Algorithmic aspects of Recon guration. (Aspects combinatoires et algorithmiques de la Recon guration)*. PhD thesis, Grenoble Alpes University, France, 2021.

[4] V. Bartier, N. Bousquet, C. Dallard, K. Lomer, and A. E. Mouawad. On girth and the parameterized complexity of token sliding and token jumping. *Algorithmica*, 83(9):2914{2951, 2021.

[5] V. Bartier, N. Bousquet, C. Feghali, M. Heinrich, B. Moore, and T. Pierron. Recoloring planar graphs of girth at least ve. *SIAM J. Discret. Math.*, 37(1):332{350, 2023.

[6] V. Bartier, N. Bousquet, J. Hanna, A. E. Mouawad, and S. Siebertz. Token sliding on graphs of girth ve. *Algorithmica*, 86(2):638{655, 2024.

[7] V. Bartier, N. Bousquet, and M. Heinrich. Recoloring graphs of treewidth 2. *Discrete Mathematics*, 344(12):112553, 2021.

[8] V. Bartier, N. Bousquet, and A. E. Mouawad. Galactic token sliding. In *30th Annual European Symposium on Algorithms, ESA 2022, September 5-9, 2022, Berlin/Potsdam, Germany*, pages 15:1{15:14, 2022.

[9] V. Bartier, N. Bousquet, and A. E. Mouawad. Galactic Token Sliding. *CoRR*, abs/2204.05549, 2022.

[10] V. Bartier, N. Bousquet, and M. Mühlenthaler. Independent set reconfiguration in H-free graphs. *CoRR*, abs/2402.03063, 2024.

[11] F. A. Behrend. On sets of integers which contain no three terms in arithmetical progression. *Proceedings of the National Academy of Sciences*, 32(12):331–332, 1946.

[12] M. Belavadi, K. Cameron, and O. Merkel. Reconfiguration of vertex colouring and forbidden induced subgraphs. *arxiv*, 2022.

[13] R. Belmonte, E. J. Kim, M. Lampis, V. Mitsou, Y. Otachi, and F. Sikora. Token sliding on split graphs. *Theory Comput. Syst.*, 65(4):662–686, 2021.

[14] K. Berczi, B. Matravølgyi, and T. Schwarcz. Reconfiguration of basis pairs in regular matroids. *CoRR*, abs/2311.07130, 2023.

[15] T. Biedl, A. Lubiw, and O. D. Merkel. Building a larger class of graphs for efficient reconfiguration of vertex colouring. *CoRR*, abs/2003.01818, 2020.

[16] H. L. Bodlaender, C. Groenland, and C. M. F. Swennenhuis. Parameterized complexities of dominating and independent set reconfiguration. In *16th International Symposium on Parameterized and Exact Computation, IPEC 2021, September 8-10, 2021, Lisbon, Portugal*, pages 9:1–9:16, 2021.

[17] M. Bonamy and N. Bousquet. Reconfiguring independent sets in cographs. *CoRR*, 1406.1433, 2014.

[18] M. Bonamy and N. Bousquet. Token sliding on chordal graphs. In *Graph-Theoretic Concepts in Computer Science - 43rd International Workshop, WG 2017*, pages 127–139, 2017.

[19] M. Bonamy and N. Bousquet. Recoloring graphs via tree decompositions. *Eur. J. Comb.*, 69:200–213, 2018.

[20] M. Bonamy, N. Bousquet, C. Feghali, and M. Johnson. On a conjecture of mohar concerning kempe equivalence of regular graphs. *J. Comb. Theory, Ser. B*, 135:179–199, 2019.

[21] M. Bonamy, N. Bousquet, M. Heinrich, T. Ito, Y. Kobayashi, A. Mary, M. Mühlenthaler, and K. Wasa. The perfect matching reconfiguration problem. In *44th International Symposium on Mathematical Foundations of Computer Science, MFCS 2019, August 26-30, 2019, Aachen, Germany*, pages 80:1–80:14, 2019.

[22] M. Bonamy, N. Bousquet, and G. Perarnau. Frozen (+ 1)-colourings of bounded degree graphs. *Combinatorics, Probability and Computing*, 30(3):330{343, 2021.

[23] M. Bonamy, P. Charbit, and S. Thomasse. Graphs with large chromatic number induce 3k-cycles. *CoRR*, abs/1408.2172, 2014.

[24] M. Bonamy, V. Delecroix, and C. Legrand-Duchesne. Kempe changes in degenerate graphs. *CoRR*, abs/2112.02313, 2021.

[25] M. Bonamy, P. Dorbec, and P. Ouvrard. Dominating sets recon guration under token sliding. *Discret. Appl. Math.*, 301:6{18, 2021.

[26] M. Bonamy, M. Heinrich, C. Legrand-Duchesne, and J. Narboni. On a recolouring version of hadwiger's conjecture. *J. Comb. Theory, Ser. B*, 164:364{370, 2024.

[27] M. Bonamy, M. Johnson, I. Lignos, V. Patel, and D. Paulusma. Reconguration graphs for vertex colourings of chordal and chordal bipartite graphs. *Journal of Combinatorial Optimization*, pages 1{12, 2012.

[28] M. Bonamy, P. Ouvrard, M. Rabie, J. Suomela, and J. Uitto. Distributed recoloring. In *32nd International Symposium on Distributed Computing, DISC 2018*, pages 12:1{12:17, 2018.

[29] E. Bonnet, N. Bousquet, P. Charbit, S. Thomasse, and R. Watrigant. Parameterized complexity of independent set in h-free graphs. *Algorithmica*, 82(8):2360{2394, 2020.

[30] E. Bonnet, E. J. Kim, S. Thomasse, and R. Watrigant. Twin-width I: tractable FO model checking. *J. ACM*, 69(1):3:1{3:46, 2022.

[31] P. Bonsma and L. Cereceda. Finding Paths Between Graph Colourings: PSPACE-Completeness and Superpolynomial Distances. In *MFCS*, volume 4708 of *Lecture Notes in Computer Science*, pages 738{749, 2007.

[32] P. Bonsma, M. Kaminski, and M. Wrochna. Recon guring Independent Sets in Claw-Free Graphs. In *Algorithm Theory - SWAT 2014 - 14th Scandinavian Symposium and Workshops*, pages 86{97, 2014.

[33] P. S. Bonsma. Independent set recon guration in cographs. In *Graph-Theoretic Concepts in Computer Science - 40th International Workshop, WG 2014, Nouan-le-Fuzelier, France, June 25-27, 2014. Revised Selected Papers*, pages 105{116, 2014.

[34] P. S. Bonsma and D. Paulusma. Using contracted solution graphs for solving recon guration problems. In *41st International Symposium on*

Mathematical Foundations of Computer Science, MFCS 2016, August 22-26, 2016 - Krakow, Poland, pages 20:1{20:15, 2016.

[35] N. Bousquet. A note on the complexity of graph recoloring. *CoRR*, abs/2401.03011, 2024.

[36] N. Bousquet and V. Bartier. Linear transformations between colorings in chordal graphs. In *27th Annual European Symposium on Algorithms, ESA 2019*, pages 24:1{24:15, 2019.

[37] N. Bousquet, B. Durain, T. Pierron, and S. Thomasse. Extremal independent set recon guration. *Electron. J. Comb.*, 30(3), 2023.

[38] N. Bousquet, L. Feuilloley, M. Heinrich, and M. Rabie. Distributed recoloring of interval and chordal graphs. In *25th International Conference on Principles of Distributed Systems, OPODIS 2021, December 13-15, 2021, Strasbourg, France*, pages 19:1{19:17, 2021.

[39] N. Bousquet, L. Feuilloley, M. Heinrich, and M. Rabie. Short and local transformations between (+1)-colorings. *CoRR*, abs/2203.08885, 2022.

[40] N. Bousquet, T. Hatanaka, T. Ito, and M. Mühlenthaler. Shortest recon guration of matchings. In *Graph-Theoretic Concepts in Computer Science - 45th International Workshop, WG 2019, Vall de Nuria, Spain, June 19-21, 2019, Revised Papers*, pages 162{174, 2019.

[41] N. Bousquet and M. Heinrich. A polynomial version of cereceda's conjecture. *J. Comb. Theory, Ser. B*, 155:1{16, 2022.

[42] N. Bousquet, F. Hommelsheim, Y. Kobayashi, M. Mühlenthaler, and A. Suzuki. Feedback vertex set recon guration in planar graphs. *Theor. Comput. Sci.*, 979:114188, 2023.

[43] N. Bousquet, T. Ito, Y. Kobayashi, H. Mizuta, P. Ouvrard, A. Suzuki, and K. Wasa. Recon guration of spanning trees with many or few leaves. In *28th Annual European Symposium on Algorithms, ESA 2020, September 7-9, 2020, Pisa, Italy (Virtual Conference)*, pages 24:1{24:15, 2020.

[44] N. Bousquet, T. Ito, Y. Kobayashi, H. Mizuta, P. Ouvrard, A. Suzuki, and K. Wasa. Recon guration of spanning trees with degree constraints or diameter constraints. *Algorithmica*, 85(9):2779{2816, 2023.

[45] N. Bousquet and A. Jo ard. TS-recon guration of dominating sets in circle and circular-arc graphs. In *Fundamentals of Computation Theory - 23rd International Symposium, FCT 2021, Athens, Greece, September 12-15, 2021, Proceedings*, pages 114{134, 2021.

[46] N. Bousquet, A. Jo ard, and P. Ouvrard. Linear transformations between dominating sets in the TAR-model. In *31st International Symposium on Algorithms and Computation, ISAAC 2020, December 14-18, 2020, Hong Kong, China (Virtual Conference)*, pages 37:1{ 37:14, 2020.

[47] N. Bousquet, A. Mary, and A. Parreau. Token jumping in minor-closed classes. In R. Klasing and M. Zeitoun, editors, *Fundamentals of Computation Theory - 21st International Symposium, FCT 2017, Bordeaux, France, September 11-13, 2017, Proceedings*, volume 10472 of *Lecture Notes in Computer Science*, pages 136{149. Springer, 2017.

[48] N. Bousquet, A. E. Mouawad, N. Nishimura, and S. Siebertz. A survey on the parameterized complexity of the independent set and (connected) dominating set recon guration problems. *CoRR*, abs/2204.10526, 2022.

[49] N. Bousquet and G. Perarnau. Fast recoloring of sparse graphs. *European Journal of Combinatorics*, 52:1{11, 2016.

[50] M. Brianski, S. Felsner, J. Hodor, and P. Micek. Recon guring independent sets on interval graphs. In *46th International Symposium on Mathematical Foundations of Computer Science, MFCS 2021, August 23-27, 2021, Tallinn, Estonia*, pages 23:1{23:14, 2021.

[51] R. Bubley and M. E. Dyer. Path coupling: A technique for proving rapid mixing in markov chains. In *38th Annual Symposium on Foundations of Computer Science, FOCS '97, Miami Beach, Florida, USA, October 19-22, 1997*, pages 223{231, 1997.

[52] S. Cambie, W. C. van Batenburg, and D. W. Cranston. Optimally recon guring list and correspondence colourings. *CoRR*, abs/2204.07928, 2022.

[53] L. Cereceda. *Mixing Graph Colourings*. PhD thesis, London School of Economics and Political Science, 2007.

[54] L. Cereceda, J. van den Heuvel, and M. Johnson. Mixing 3-colourings in bipartite graphs. *Eur. J. Comb.*, 30(7):1593{1606, 2009.

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

[56] P. Chalermsook, M. Cygan, G. Kortsarz, B. Laekhanukit, P. Manurangsi, D. Nanongkai, and L. Trevisan. From Gap-Exponential Time Hypothesis to Fixed Parameter Tractable Inapproximability: Clique, Dominating Set, and More. *SIAM J. Comput.*, 49(4):772{810, 2020.

[57] L. S. Chandran, U. K. Gupta, and D. Pradhan. Towards a conjecture on the recoloring diameter of planar graphs. *CoRR*, abs/2209.05992, 2022.

[58] S. Chen, M. Delcourt, A. Moitra, G. Perarnau, and L. Postle. Improved bounds for randomly sampling colorings via linear programming. In *Proceedings of the Thirtieth Annual ACM-SIAM Symposium on Discrete Algorithms, SODA 2019*, pages 2216{2234, 2019.

[59] D. W. Cranston. List-recoloring of sparse graphs. *Eur. J. Comb.*, 105:103562, 2022.

[60] D. W. Cranston and R. Mahmoud. 5-coloring recon guration of planar graphs with no short odd cycles. *J. Graph Theory*, 105(4):670{679, 2024.

[61] M. Cygan, F. V. Fomin, L. Kowalik, D. Lokshtanov, D. Marx, M. Pilipczuk, M. Pilipczuk, and S. Saurabh. *Parameterized Algorithms*. Springer Publishing Company, Incorporated, 1st edition, 2015.

[62] A. Dawar and S. Kreutzer. Domination problems in nowhere-dense classes. In R. Kannan and K. N. Kumar, editors, *IARCS Annual Conference on Foundations of Software Technology and Theoretical Computer Science, FSTTCS 2009, December 15-17, 2009, IIT Kanpur, India*, volume 4 of *LIPIcs*, pages 157{168. Schloss Dagstuhl - Leibniz-Zentrum fur Informatik, 2009.

[63] E. D. Demaine, M. L. Demaine, E. Fox-Epstein, D. A. Hoang, T. Ito, H. Ono, Y. Otachi, R. Uehara, and T. Yamada. Linear-time algorithm for sliding tokens on trees. *Theoretical Computer Science*, 600:132{142, Oct. 2015.

[64] P. G. Drange, M. S. Dregi, F. V. Fomin, S. Kreutzer, D. Lokshtanov, M. Pilipczuk, M. Pilipczuk, F. Reidl, F. S. Villaamil, S. Saurabh, S. Siebertz, and S. Sikdar. Kernelization and sparseness: the case of dominating set. In N. Ollinger and H. Vollmer, editors, *33rd Symposium on Theoretical Aspects of Computer Science, STACS 2016, February 17-20, 2016, Orleans, France*, volume 47 of *LIPIcs*, pages 31:1{31:14. Schloss Dagstuhl - Leibniz-Zentrum fur Informatik, 2016.

[65] Z. Dvorak and C. Feghali. An update on recon guring 10-colorings of planar graphs. *Electron. J. Comb.*, 27(4):4, 2020.

[66] Z. Dvorak and C. Feghali. An update on recon guring 10-colorings of planar graphs, 2020.

[67] Z. Dvorak and C. Feghali. A thomassen-type method for planar graph recoloring. *Eur. J. Comb.*, 95:103319, 2021.

[68] M. Dyer, A. D. Flaxman, A. M. Frieze, and E. Vigoda. Randomly coloring sparse random graphs with fewer colors than the maximum degree. *Random Structures & Algorithms*, 29(4):450{465, 2006.

[69] K. Eickmeyer, A. C. Giannopoulou, S. Kreutzer, O. Kwon, M. Pilipczuk, R. Rabinovich, and S. Siebertz. Neighborhood complexity and kernelization for nowhere dense classes of graphs. In I. Chatzigiannakis, P. Indyk, F. Kuhn, and A. Muscholl, editors, *44th International Colloquium on Automata, Languages, and Programming, ICALP 2017, July 10-14, 2017, Warsaw, Poland*, volume 80 of *LIPIcs*, pages 63:1{63:14. Schloss Dagstuhl - Leibniz-Zentrum fur Informatik, 2017.

[70] P. Erdos and R. Rado. Intersection theorems for systems of sets. *Journal of the London Mathematical Society*, 35:85{90, 1960.

[71] L. Esperet and P. Ochem. Islands in graphs on surfaces. *SIAM Journal on Discrete Mathematics*, 30(1):206{219, 2016.

[72] G. Fabianski, M. Pilipczuk, S. Siebertz, and S. Torunczyk. Progressive algorithms for domination and independence. In R. Niedermeier and C. Paul, editors, *36th International Symposium on Theoretical Aspects of Computer Science, STACS 2019, March 13-16, 2019, Berlin, Germany*, volume 126 of *LIPIcs*, pages 27:1{27:16. Schloss Dagstuhl - Leibniz-Zentrum fur Informatik, 2019.

[73] M. Farber, B. Richter, and H. Shank. Edge-disjoint spanning trees: A connectedness theorem. *J. Graph Theory*, 9(3):319{324, 1985.

[74] C. Feghali. Paths between colourings of graphs with bounded treewidth. *Inf. Process. Lett.*, 144:37{38, 2019.

[75] C. Feghali. Paths between colourings of sparse graphs. *Eur. J. Comb.*, 75:169{171, 2019.

[76] C. Feghali. Recon guring colorings of graphs with bounded maximum average degree. *Journal of Combinatorial Theory, Series B*, 147:133 { 138, 2021.

[77] C. Feghali and J. Fiala. Recon guration graph for vertex colourings of weakly chordal graphs. *Discret. Math.*, 343(3):111733, 2020.

[78] C. Feghali, M. Johnson, and D. Paulusma. A recon gurations analogue of brooks' theorem and its consequences. *Journal of Graph Theory*, 83(4):340{358, 2016.

[79] C. Feghali, M. Johnson, and D. Paulusma. Kempe equivalence of colourings of cubic graphs. *European Journal of Combinatorics*, 59:1{ 10, 2017.

[80] C. Feghali and O. D. Merkel. Mixing colourings in $2K_2$-free graphs. *Discret. Math.*, 345(11):113108, 2022.

[81] E. Fox-Epstein, D. A. Hoang, Y. Otachi, and R. Uehara. Sliding token on bipartite permutation graphs. In K. M. Elbassioni and K. Makino, editors, *Algorithms and Computation - 26th International Symposium, ISAAC 2015, Nagoya, Japan, December 9-11, 2015, Proceedings*, volume 9472 of *Lecture Notes in Computer Science*, pages 237{247. Springer, 2015.

[82] M. Grohe, S. Kreutzer, and S. Siebertz. Deciding rst-order properties of nowhere dense graphs. *J. ACM*, 64(3):17:1{17:32, 2017.

[83] A. Grzesik, T. Klimosova, M. Pilipczuk, and M. Pilipczuk. Polynomial-time algorithm for maximum weight independent set on p6-free graphs. In *Proceedings of the Thirtieth Annual ACM-SIAM Symposium on Discrete Algorithms, SODA 2019, San Diego, California, USA, January 6-9, 2019*, pages 1257{1271, 2019.

[84] R. Haas and K. Sey arth. The k-dominating graph. *Graphs and Combinatorics*, 30(3):609{617, Mar. 2013.

[85] R. Haas and K. Sey arth. Recon guring dominating sets in some well-covered and other classes of graphs. *Discrete Mathematics*, 340(8):1802 { 1817, 2017.

[86] A. Haddadan, T. Ito, A. E. Mouawad, N. Nishimura, H. Ono, A. Suzuki, and Y. Tebbal. The complexity of dominating set reconguration. *Theoretical Computer Science*, 651:37{49, Oct. 2016.

[87] T. Hatanaka, T. Ito, and X. Zhou. The coloring recon guration problem on speci c graph classes. In *Combinatorial Optimization and Applications - 11th International Conference, COCOA 2017, Shanghai, China, December 16-18, 2017, Proceedings, Part I*, pages 152{162, 2017.

[88] T. Hatanaka, T. Ito, and X. Zhou. Complexity of recon guration problems for constraint satisfaction. *CoRR*, abs/1812.10629, 2018.

[89] T. Hatanaka, T. Ito, and X. Zhou. The coloring recon guration problem on speci c graph classes. *IEICE Trans. Inf. Syst.*, 102-D(3):423{ 429, 2019.

[90] R. A. Hearn and E. 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, 2005.

[91] M. Heinrich. *Recon guration and Combinatorial Games*. PhD thesis, Universite Claude Bernard Lyon 1, 2019.

[92] M. Heinrich. Glauber dynamics for colourings of chordal graphs and graphs of bounded treewidth. *CoRR*, abs/2010.16158, 2020.

[93] D. A. Hoang, E. Fox-Epstein, and R. Uehara. Sliding tokens on block graphs. In *WALCOM: Algorithms and Computation, 11th International Conference and Workshops, WALCOM 2017, Hsinchu, Taiwan, March 29-31, 2017, Proceedings*, pages 460{471, 2017.

[94] D. A. Hoang, A. Khorramian, and R. Uehara. Shortest recon guration sequence for sliding tokens on spiders. In *Algorithms and Complexity - 11th International Conference, CIAC 2019, Rome, Italy, May 27-29, 2019, Proceedings*, pages 262{273, 2019.

[95] D. A. Hoang and R. Uehara. Sliding tokens on a cactus. In *27th International Symposium on Algorithms and Computation, ISAAC 2016, December 12-14, 2016, Sydney, Australia*, pages 37:1{37:26, 2016.

[96] T. Ito, E. Demaine, N. Harvey, C. Papadimitriou, M. Sideri, R. Uehara, and Y. Uno. On the complexity of recon guration problems. *Theor. Comput. Sci.*, 412(12-14):1054{1065, 2011.

[97] T. Ito, M. Kaminski, and E. D. Demaine. Recon guration of list edge-colorings in a graph. *Discret. Appl. Math.*, 160(15):2199{2207, 2012.

[98] T. Ito, M. Kaminski, and H. Ono. Fixed-parameter tractability of token jumping on planar graphs. In H. Ahn and C. Shin, editors, *Algorithms and Computation - 25th International Symposium, ISAAC 2014, Jeonju, Korea, December 15-17, 2014, Proceedings*, volume 8889 of *Lecture Notes in Computer Science*, pages 208{219. Springer, 2014.

[99] M. Jerrum. A very simple algorithm for estimating the number of k-colorings of a low-degree graph. *Random Structures & Algorithms*, 7(2):157{165, 1995.

[100] A. Jo ard. *Graph domination and recon guration problems. (Domination de graphes et problemes de recon guration)*. PhD thesis, University of Lyon, France, 2020.

[101] M. Kaminski, P. Medvedev, and M. Milanic. Complexity of independent set recon gurability problems. *Theor. Comput. Sci.*, 439:9{15, 2012.

[102] M. Kaminski, P. Medvedev, and M. Milanic. Complexity of independent set recon gurability problems. *Theoretical Computer Science*, 439(0):9{15, 2012.

[103] M. Kanzaki, Y. Otachi, and R. Uehara. Computational complexity of jumping block puzzles. In *Computing and Combinatorics - 27th International Conference, COCOON 2021, Tainan, Taiwan, October 24-26, 2021, Proceedings*, pages 655{667, 2021.

[104] T. Kővari, V. Sos, and P. Turan. On a problem of K. Zarankiewicz. *Colloq. Math.*, 3:50{57, 1954.

[105] Y. Kobayashi, R. Mahara, and T. Schwarcz. Recon guration of the union of arborescences. In *34th International Symposium on Algorithms and Computation, ISAAC 2023, December 3-6, 2023, Kyoto, Japan*, pages 48:1{48:14, 2023.

[106] S. Kreutzer, R. Rabinovich, and S. Siebertz. Polynomial kernels and wideness properties of nowhere dense graph classes. *ACM Trans. Algorithms*, 15(2):24:1{24:19, 2019.

[107] M. Las Vergnas and H. Meyniel. Kempe classes and the hadwiger conjecture. *Journal of Combinatorial Theory, Series B*, 31(1):95{104, 1981.

[108] E. Lawler. *Combinatorial Optimization - Networks and Matroids*. Holt, Rinehart and Winston, New York, 1976.

[109] J. Lee, J. A. Noel, and M. H. Siggers. Recon guring graph homomorphisms on the sphere. *Eur. J. Comb.*, 86:103086, 2020.

[110] N. Linial. Locality in distributed graph algorithms. *SIAM J. Comput.*, 21(1):193{201, 1992.

[111] D. Lokshtanov and A. E. Mouawad. The complexity of independent set recon guration on bipartite graphs. *ACM Trans. Algorithms*, 15(1):7:1{7:19, 2019.

[112] D. Lokshtanov, A. E. Mouawad, F. Panolan, M. S. Ramanujan, and S. Saurabh. Recon guration on sparse graphs. *J. Comput. Syst. Sci.*, 95:122{131, 2018.

[113] D. Lokshtanov, M. Vatshelle, and Y. Villanger. Independent set in P_5-free graphs in polynomial time. In *Proceedings of the Twenty-Fifth Annual ACM-SIAM Symposium on Discrete Algorithms, SODA 2014, Portland, Oregon, USA, January 5-7, 2014*, pages 570{581, 2014.

[114] L. Lovasz. Matroid matching and some applications. *Journal of Combinatorial Theory, Series B*, 28(2):208{236, 1980.

[115] O. D. Merkel. Recolouring weakly chordal graphs and the complement of triangle-free graphs. *Discret. Math.*, 345(3):112708, 2022.

[116] B. Mohar. *Kempe Equivalence of Colorings*, pages 287{297. Birkhauser Basel, Basel, 2007.

[117] A. E. Mouawad, N. Nishimura, V. Raman, N. Simjour, and A. Suzuki. On the parameterized complexity of recon guration problems. *Algorithmica*, 78(1):274{297, 2017.

[118] C. Mynhardt, L. Teshima, and A. Roux. Connected k-dominating graphs. *Discrete Mathematics*, 342(1):145{151, Jan. 2019.

[119] N. Nishimura. Introduction to recon guration. *Algorithms*, 11(4):52, 2018.

[120] I. Z. Ruzsa and E. Szemeredi. Triple systems with no six points carrying three triangles. *Combinatorics (Keszthely, 1976), Coll. Math. Soc. J. Bolyai*, 18:939{945, 1978.

[121] J. Salas and A. D. Sokal. Absence of phase transition for antiferromagnetic potts models via the dobrushin uniqueness theorem. *Journal of Statistical Physics*, 86:551{579, oct 1997.

[122] J. Salas and A. D. Sokal. Ergodicity of the wang-swendsen-kotecky algorithm on several classes of lattices on the torus. *Journal of Physics A: Mathematical and Theoretical*, 55(41):415004, oct 2022.

[123] N. Sauer. On the density of families of sets. *J. Comb. Theory, Ser. A*, 13(1):145{147, 1972.

[124] S. Shelah. A combinatorial problem; stability and order for models and theories in in nitary languages. *Paci c Journal of Mathematics*, 41(1):247 { 261, 1972.

[125] S. Siebertz. Recon guration on nowhere dense graph classes. *Electron. J. Comb.*, 25(3):P3.24, 2018.

[126] J. A. Soto. A simple PTAS for weighted matroid matching on strongly base orderable matroids. *Discret. Appl. Math.*, 164:406{412, 2014.

[127] E. Speckenmeyer. On feedback vertex sets and nonseparating independent sets in cubic graphs. *Journal of Graph Theory*, 12(3):405{412, 1988.

[128] A. Suzuki, A. E. Mouawad, and N. Nishimura. Recon guration of dominating sets. *Journal of Combinatorial Optimization*, 32(4):1182{1195, Aug. 2015.

[129] C. Thomassen. Every planar graph is 5-choosable. *J. Comb. Theory, Ser. B*, 62(1):180{181, 1994.

[130] S. Ueno, Y. Kajitani, and S. Gotoh. On the nonseparating independent set problem and feedback set problem for graphs with no vertex degree exceeding three. *Discrete Mathematics*, 72(1-3):355{360, 1988.

[131] J. van den Heuvel. *The Complexity of change*, page 409. Part of London Mathematical Society Lecture Note Series. 2013.

[132] E. Vigoda. Improved bounds for sampling colorings. In *40th Annual Symposium on Foundations of Computer Science, FOCS '99, 17-18 October, 1999, New York, NY, USA*, pages 51{59, 1999.

[133] N. L. White. A unique exchange property for bases. *Linear Algebra and its Applications*, 21:81{91, 1980.

[134] M. Wrochna. Recon guration in bounded bandwidth and tree-depth. *J. Comput. Syst. Sci.*, 93:1{10, 2018.

[135] M. Wrochna. Homomorphism recon guration via homotopy. *SIAM J. Discret. Math.*, 34(1):328{350, 2020.

[136] T. Yamada and R. Uehara. Shortest recon guration of sliding tokens on subclasses of interval graphs. *Theor. Comput. Sci.*, 863:53{68, 2021.