Uniwersytet Warszawski
Wydział Matematyki, Informatyki i Mechaniki

Marcin Wrochna
Nr albumu: 290715

Reconfiguration and structural graph theory

Praca magisterska
na kierunku INFORMATYKA

Praca wykonana pod kierunkiem
dra hab. Marcina Kamińskiego
Instytut Informatyki UW

Czerwiec 2014

Oświadczenie kierującego pracą

Potwierdzam, że niniejsza praca została przygotowana pod moim kierunkiem i kwalifikuje się do przedstawienia jej w postępowaniu o nadanie tytułu zawodowego.

Data Podpis kierującego pracą

Oświadczenie autora (autorów) pracy

Świadom odpowiedzialności prawnej oświadczam, że niniejsza praca dyplomowa została napisana przeze mnie samodzielnie i nie zawiera treści uzyskanych w sposób niezgodny z obowiązującymi przepisami.

Oświadczam również, że przedstawiona praca nie była wcześniej przedmiotem procedur związanych z uzyskaniem tytułu zawodowego w wyższej uczelni.

Oświadczam ponadto, że niniejsza wersja pracy jest identyczna z załączoną wersją elektroniczną.

Data Podpis autora (autorów) pracy

Streszczenie

Rekonfiguracja problemu kombinatorycznego polega na przeprowadzaniu małych zmian na jego rozwiązaniach tak, by zachować własność bycia rozwiązaniem. W pracy zbadano jak do szukania ciągów takich zmian między dwoma danymi rozwiązaniami można wykorzystać strukturę grafu opisującego rekonfigurowany problem. Ogólniejsze spojrzenie na rekonfigurację dowolnego problemu opisywanego przez homomorfizm pozwala formalnie opisać pewne zależności i pomaga zrozumieć dotychczas obserwowane wyjątki.

Pokazujemy, że spośród parametrów opisujących rzadkość grafu jedynie ograniczenie głębokości drzewiastej prowadzi do efektywnych algorytmów rekonfiguracji. Dla kontrastu podajemy algorytm, który wykorzystując wyjątkową strukturę tzw. grafów *bez pazurów*, znajduje ciągi rekonfiguracyjne dla problemu zbioru niezależnego. Wreszcie dla rekonfiguracji problemu 3-kolorowania, która nieco zaskakująco okazuje się łatwiejsza niż sam problem znajdowania 3-kolorowań, przedstawiono nowy dowód tego faktu akcentujący rolę jednego warunku, który wystarczy do przeprowadzenia większości dowodu w znacznie szerszym kontekście homomorfizmów.

Słowa kluczowe

rekonfiguracja, algorytm wielomianowy, homomorfizm, PSPACE-zupełność, grafy bez pazurów, zbiór niezależny

Dziedzina pracy (kody wg programu Socrates-Erasmus)

11.3 Informatyka

Klasyfikacja tematyczna

Theory of computation
    Design and analysis of algorithms
        Graph algorithms analysis

Tytuł pracy w języku polskim

Rekonfiguracja a strukturalna teoria grafów

Abstract

Reconfiguration of a combinatorial problem is the application of small transformations to a solution of the problem in a way that preserves the property of being a solution. This thesis studies how the structure of the underlying problem can be used to find sequences of such transformations. A more general view on the reconfiguration of any problem described by a homomorphism allows to formally describe certain patterns and helps to understand exceptions known before.

We show that among parameters that describe the sparsity of a graph, only bounded treedepth leads to effective reconfiguration algorithms. In contrast, we give an algorithm which uses the special structure of the so-called *claw-free graphs* to find reconfiguration sequences between independent sets. Finally, for the reconfiguration of 3-colorings, which somewhat surprisingly has been shown easier than the problem of finding 3-colorings, we give a new proof of this fact which highlights the role of one assumption that suffices to carry out most of the proof in the more general context of homomorphisms.

Keywords

reconfiguration, polynomial algorithm, homomorphism, PSPACE-completeness, claw-free graphs, independent set

Contents

Introduction

Reconfiguration is a recently developed framework that studies problems arising when a step-by-step transformation is sought between solutions of a combinatorial problem. The aim is to explore the solution space of such problems, explain the behavior of local search heuristics, but also to directly model situations which require careful transformation of a system – puzzles with sliding blocks give a typical example.

Various transformations steps and constraints on what constitutes a solution have been analyzed, with sometimes surprisingly different results. Nonetheless, some patterns arise, most important of which is the observation that simple combinatorial problems often give rise to tractable reconfiguration variants. Examples that contradict this pattern include on one hand the reconfiguration of shortest s-t paths: finding such a path is easy, while finding a sequence of transformation steps between two paths is PSPACE-complete, as proven by Bonsma [Bon13]. On the other hand we have the 3-Coloring problem: finding a solution is NP-complete, while Cereceda et al. [CHJ11] gave a polynomial algorithm that finds a reconfiguration sequence between two given solutions if one exists. In this work we try to give an insight into what causes those exceptions and how general results about reconfiguration can be formalized, giving in particular new, short proofs of both results as corollaries to more general observations.

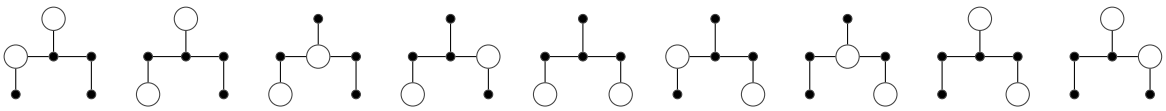


Figure 1: A shortest reconfiguration sequence, in the *token sliding* model – an independent set is reconfigured by sliding its elements along edges.

Reconfiguration of homomorphisms

To give a precise meaning to patterns observed in the area of reconfiguration, we study a large class of problems that arises from a generalization of k-colorings – homomorphisms. Homomorphisms allow to describe many interesting combinatorial properties and form a special case, though already quite general by itself, of constraint satisfaction problems (CSPs). As the study of reconfiguration was initiated by the work on colorings and the work of Gopalan et al. [Gop+09] on generalized satisfaction problems, which are CSPs limited to the boolean domain, we believe this to be a natural direction for exploring reconfiguration problems, many of which can be described in such a framework. Indeed, in Chapters 2 and 4 we argue that this view allows to answer questions posed about reconfiguration without our definitions in mind.

Sparse graphs

In Chapter 2 we ask how different invariants that describe how sparse a graph is can be applied to reconfiguration problems. Very general results are known for such invariants, one of the most celebrated being Courcelle's theorem [Cou90], stating that for every class of graphs of bounded treewidth, every problem definable in Monadic Second Order logic can be solved in time linear in the size of the graph. We will not define the logic here, let us only mention that this and similar results give linear time algorithms for k-Coloring, H-Homomorphism and Maximum Independent Set (for every integer k and any digraph H) on any class of graphs with bounded treewidth. The notion of sparsity is central to several algorithmic meta-theorems and connects many results concerning property testing, homomorphisms and constraint satisfaction problems, as highlighted in the book of Nešetřil and Ossona de Mendez [NdM12].

A few results showing that reconfiguration problems become tractable when limited to specific graph classes are known – this motivated the question of whether such a result can be proved for classes of sparse graphs, in particular graphs of bounded treewidth, as posed by Bonsma [Bon12]. He further motivated the question by showing that the techniques used for such classes – dynamic programming – apply to reconfiguration, by using them to show polynomial algorithms for shortest path reconfiguration in planar graphs and independent set reconfiguration in cographs (i.e., P_4-free graphs) [Bon14].

We answer this question in the negative by demonstrating that several reconfiguration problems remain PSPACE-hard when limited to graphs of bounded bandwidth, a parameter much more restrictive than treewidth or pathwidth. The common part of the reductions is the proof of hardness of the reconfiguration of H-homomorphisms on the class of directed paths, for some fixed digraph H. This problem is restated as a simple word reconfiguration problem, highlighting that the complexity of reconfiguration can have a source entirely different from the complexity of the underlying problem and that the reconfiguration of very simple systems can directly simulate the execution of a Turing Machine.

On the other hand, we show that a large class of reconfiguration problems has polynomial algorithms for graphs of bounded treedepth. This being very restrictive, the algorithm is not very surprising nor practical, but by connecting the fact with the PSPACE-hardness reductions, we argue that this result is tight, in a sense, and nothing more general can be achieved. Specifically, we show that a class of digraphs closed under subdigraphs and reorienting arcs has polynomial algorithms for all H-homomorphism reconfiguration problems if and only if it has bounded treedepth.

Claw-free graphs

Having shown that reconfiguration problems, including that of independent sets, may be very hard in seemingly simple classes of graphs, we focus in Chapter 3 on an example where a special structure tightly connected with the problem itself helps devise solutions effectively. Claw-free graphs are graphs that exclude the *claw* $K_{1,3}$ as an induced subgraph. It is long known that they allow polynomial time algorithms for the Maximum Independent Set problem, as proven independently by Sbihi [Sbi80] and Minty [Min80] (see [NT01] for the weighted variant, other approaches with better running times have been made in [Sch03] and [NS13]). Decomposition theorems describing the structure of claw-free graphs have recently been shown by Chudnovsky and Seymour [CS05], while Faenza et al. [FOS11] presented a weaker variant that allowed them to solve the Maximum Weighted Independent Set problem in cubic time.

We study the reconfiguration of independent sets in claw-free graphs in two models known as *token sliding* and *token jumping*. We give an algorithm for the reachability problem in both models, describe several cases when the solution graph is connected and prove hardness of the problem where we ask for *shortest* reconfiguration sequences. The techniques are mostly self-contained arguments about the local conditions that firmly bound possible solutions. We do not use the powerful decompositions, but observe that a basic building block – circular arc graphs – unambiguously appear as a component in the last, hardest case of our algorithm. This suggests that they may have a more general role in the structure of claw-free graphs than what is implied by known decompositions, in a way that is not well understood, as our approach needs an intricate and technical construction to handle this case.

The class of claw-free graphs includes line graphs, proper interval graphs and proper circular arc graphs. An independent set of the line graph of a graph G is the same as a matching of G, thus our result may be seen as a generalization of the algorithm for matching reconfiguration (in the jumping model) of [Ito+12]. An independent set in a proper interval or proper circular arc graph corresponds to a set of pairwise disjoint intervals or arcs. This models various scheduling and packing problems (with circular arc graphs corresponding periodic schedules), see e.g. [KNC07]. Our results can thus be related to problems arising when schedules need to be changed gradually, for example.

Restricting the constraints

Chapter 4 tries a different approach where instead of limiting the graphs on which combinatorial objects are reconfigured, the structure of combinatorial constraints themselves is limited. Following a few general observations about H-homomorphism reconfiguration problems, we consider the problem of reconfiguring 3-colorings. Deciding 3-colorability is known to be NP-hard even for 4-regular planar graphs [Dai80]. Cereceda et al. [CHJ11] showed that reconfiguration sequences between 3-colorings can be found in polynomial time, and this was improved to shortest reconfiguration sequences by Johnson et al. [Joh+14]. We give a new, simple proof of the latter result by considering the reconfiguration of more general H-homomorphism for graphs H with the *monochromatic neighborhood property* – the property that all neighbors of a vertex whose color is being changed must have the same color. It is natural to ask whether the somewhat surprising algorithmic tractability of 3-coloring reconfiguration is a consequence of this property. We argue that this is indeed true by proving several lemmas which strongly restrict the possibilities of reconfiguration based on this property alone. A short proof specific to the case of 3-colorings then suffices to give an algorithm that describes all possible shortest reconfiguration sequences between two 3-colorings.

Acknowledgements

Most of the results described in Chapter 3 were obtained together with Marcin Kamiński and Paul Bonsma. My contributions include all of Sections 3.4, 3.6 and 3.8, a shortened proof of the external case as presented in 3.5 and several key ideas in all of the remaining proofs. The results have been published in [BKW14] and accepted for the 14th Scandinavian Symposium and Workshops on Algorithm Theory (SWAT 2014).

The results of Chapter 2 are available as a preprint in [Wro14]. A short version describing the hardness results appeared in [Mou+14] (together with results by co-authors about parameterized variants that are not considered here) and will be submitted to the 9th International Symposium on Parameterized and Exact Computation (IPEC 2014).

I would like to thank Marcin, my advisor, for his patience and understanding, and for introducing me to reconfiguration – an area very new and full of possibilities, yet deeply connected with a range of topics I enjoyed exploring. Above all, warm thanks are due to Naomi Nishimura and Amer Mouawad for inviting me to the University of Waterloo. Their expertise, keen ear and comments greatly helped improve the exposition of Chapter 4. There is also another Marcin whom I would like to thank, well, for everything.

Chapter 1

Definitions and notation

1.1. Basic notions

For general graph theory terminology we follow [Die12]. For definitions of complexity classes and reductions, see [Koz06].

Graph, digraph Digraphs are directed graphs. We denote an edge between u and v, or an arc from u to v, as uv. Our definition of graphs does not allow loops (edges vv) or multiple edges, unless we explicitly mention it to be *a graph with loops allowed* and except for Chapter 4, where graphs with loops allowed are always considered. Our definition of digraphs does not allow multiple arcs, but allows loops (arcs vv) and 2-cycles (arcs uv, vu for some vertices u, v). We write $V(G)$ for the vertex set of G and $E(G)$ for the edge or arc set of G.

Neighborhood Given a graph $G = (V, E)$, the *open neighborhood* of a set of vertices S is the set $N_G(S) = \{w \in V \setminus S \mid \exists_{v \in S} vw \in E\}$, the subscript is omitted when it is clear from the context. The neighborhood of a vertex is defined as $N(v) = N(\{v\})$. The *closed neighborhood* is defined as $N[S] = N(S) \cup S$ and $N[v] = N[\{v\}]$. The *step neighborhood* is defined as $N^+(S) = \{w \in V \mid \exists_{v \in S} vw \in E\}$ and $N^+(v) = N^+(\{v\})$. In particular, the step neighborhood of a vertex includes it if and only if G has a loop at v.

Set operators, induced subgraphs Sets $I \setminus \{v\}$ and $I \cup \{v\}$ are denoted by $I - v$ and $I + v$ respectively. The symmetric difference of two sets I and J is denoted by $I \Delta J = (I \setminus J) \cup (J \setminus I)$. We denote the subgraph of G induced by S by $G[S]$. By $G - A$ we denote the graph $G[V \setminus A]$.

Independent set, dominating set An independent set of a graph G (also called a *stable set*) is a set I of vertices such that no two vertices of I are joined by an edge in G. The size of the largest independent set of G is denoted by $\alpha(G)$. An independent set is *maximum* if there is no larger independent set, *maximal* if it is not a proper subset of some larger independent set. A dominating set of G is a set S of vertices such that $N[S] = V(G)$.

Paths, walks and distances, cycles The directed path on n vertices is the graph with vertex set $\{1, \ldots, n\}$ and edges $\{12, 23, \ldots, (n-1)n\}$. The directed cycle on n vertices is obtained from P_n by adding an arc from n to 1. Undirected paths and cycles are their underlying graphs, and are denoted as P_n and C_n respectively (n being the number of vertices).

Cliques and bicliques The clique on n vertices, denoted K_n, is the graph with edges between every two different vertices. The biclique $K_{n,m}$ is the graph with $n + m$ vertices, the first n being all pairwise adjacent to the last m.

1.2. Reconfiguration problems

Reconfiguration problems are concisely stated by defining the *solution graph*, the vertices of which are solutions to an instance of the underlying combinatorial problem, and whose edges are defined by describing transformation steps – the changes (usually small and local) one is allowed to make – two solutions are adjacent iff one can be obtained from the other in one transformation step. We are mostly concerned with the computational complexity of the REACHABILITY problem, where given an instance and two solutions one asks whether there is a path between them in the solution graph.

Coloring The simplest example might come from the k-COLORING problem, defined for every integer $k \geqslant 2$. An instance of the problem is a graph G, and one asks whether there exists a *k-coloring*, i.e., an assignment $c : V(G) \to \{1, \ldots, k\}$ such that every for every edge $uv \in E(G)$ we have $c(u) \neq c(v)$. For a given instance G, we can define the solution graph to have all k-colorings of G as vertices and a transformation step to consist of changing the color of one vertex (so there are edges between every two colorings that differ on only one vertex). k-COLORING REACHABILITY is then the following problem: given a graph G and two of its k-colorings, can one be transformed into the other by changing one color at a time, going only through k-colorings.

Independent Set, TS, TJ, TAR Another example comes from INDEPENDENT SET, the problem where given a graph G and a number k, one is asked whether there is a *independent set* of vertices of size at least k, i.e., a set such that no two vertices are joined by an edge in G (also known as a *stable set*). Different adjacency relations have been studied. In the *Token Jumping* model, the solutions are independent sets of some fixed size k, and the only transformation step allowed is removing a vertex and adding another to the set (the name comes from viewing the set as k tokens placed on the graph's vertices). In the *Token Sliding* model, the transformation step is the same, except the vertex we added must be adjacent to the one we removed (so we view this as sliding tokens along edges). In the *Token Addition Removal* model, the solutions are all independent sets of size $\geqslant k - 1$, and the transformation steps allowed are removing any vertex or adding any vertex. This gives rise to solution graphs denoted as $\mathrm{TJ}_k(G)$, $\mathrm{TS}_k(G)$ and $\mathrm{TAR}_k(G)$ respectively and problems TJ, TS and TAR REACHABILITY.

Note that the solution graph TS_k is a subgraph of TJ_k with the same vertex set; proofs for these two models can be very different though. On the other hand Token Jumping with parameter k can be seen as a special case of Token Addition Removal with the same parameter where the two input independent sets are restricted to have size exactly k (a TAR sequence then never needs to use independent sets of size other than k and $k - 1$, see also [KMM11]). We also state some result for the reconfiguration of maximum independent sets, that is, the Token Jumping and Token Sliding models with $k = \alpha(G)$ – the models turn out to be exactly equivalent in this case, $\mathrm{TS}_k(G)$=$\mathrm{TJ}_k(G)$, because a token jumping from one vertex to another could be placed in both places if they were not adjacent. In particular, a hardness result for MAXIMUM INDEPENDENT SET REACHABILITY implies hardness of TS, TJ and TAR REACHABILITY.

Shortest Path In SHORTEST PATH an instance consists of a graph with two distinguished vertices s, t, and solutions are shortest paths between s and t. We define a transformation step as changing one vertex of the path. SHORTEST PATH REACHABILITY provided the first natural example of a PSPACE-hard reconfiguration problem [Bon12] whose underlying combinatorial problem is easy. SHORTEST PATH REACHABILITY can easily be reduced to MAXIMUM INDEPENDENT SET REACHABILITY (by 'locally complementing' the instance graph G) [KMM11].

List Coloring In k-List-Coloring an instance consists of a graph whose vertices are labeled with lists $l(v) \subseteq \{1, \ldots, k\}$. Solutions are k-list colorings, i.e., k-colorings $c : V(G) \to \{1, \ldots, k\}$ such that $s(v) \in l(v)$ for all $v \in V(G)$. We define a transformation step as a change of the color of one vertex.

Shortest Reachability Shortest Reachability problems are defined as the Reachability problems, but an integer ℓ is additionaly given on input and the question is whether there is, in the solution graph, a path *of length at most* ℓ between two given configuration.

1.3. Graph structure

Treewidth A *tree decomposition* of a graph G is a tree T whose vertices $(X_t)_{t \in T}$ (called *bags*) are subsets of $V(G)$ such that: (1) the union of all X_t equals $V(G)$, (2) for every edge $uv \in E(G)$, there is a $t \in T$ such that $u, v \in X_t$, (3) for every vertex v, the set $\{t \mid v \in X_t\}$ is connected in T. The *width* of a tree decomposition is the maximum size of a bag minus one. The treewidth of G is the minimum width among all possible tree decompositions of G.

Pathwidth is defined analogously to treewidth, with the additional restriction that T must be a path. Clearly, a graph of pathwidth k has treewidth at most k. We only mention pathwidth and treewidth when discussing our results, as they are better known than the stronger ones defined below.

Bandwidth is the minimum over all injective assignments $f : V(G) \to \mathbb{N}$ of the quantity $\max_{uv \in E(G)} |f(u) - f(v)|$. A graph of bandwidth b can easily be seen to have pathwidth and treewidth at most b (bags containing $b + 1$ consecutive vertices from the sequence $f^{-1}(0), f^{-1}(1), \ldots$, form a tree decomposition) and maximum degree at most $2b$. The family of stars $K_{1,n}$ gives an example with bounded pathwidth but unbounded bandwidth. Bandwidth is thus strictly more restrictive than many other parameters and the hardness results of Chapter 2 immediately imply the same results for bounded degree, treewidth, etc.

A *bucket arrangement* of a graph is a partition of the vertex set into a sequence of buckets, such that the endpoints of any edge are either in one bucket or in two consecutive buckets. If a graph has a bucket arrangement where each bucket has at most b vertices, then it has bandwidth at most $2b$ (arrange one bucket after another, with any ordering within one bucket; see also [FT05]).

Treedepth is a graph parameter also known as *vertex ranking number*, *ordered chromatic number*, and *minimum elimination tree height*. Introduced by Nešetřil and Ossona de Mendez [NOdM06], it found various algorithmic applications. It is defined as the minimum height of a rooted forest closure F such that the graph is a subgraph of F. A rooted forest closure of height h is any graph obtained from a disjoint union of rooted trees of height h by adding edges from every vertex to all of its ancestors. There are only finitely many graphs of any given treedepth with no non-trivial automorphisms. Intuitively, graphs of bounded treedepth can be large only by having many copies of the same subgraph, all acting the same way.

Graphs of treedepth k have pathwidth and treewidth at most k, a decomposition can be obtained by creating a bag for each leaf of a tree whose closure contains the graph, containing the leaf and all of its ancestors. Treedepth is incomparable with bandwidth: stars $K_{1,n}$ have treedepth 2 but unbounded bandwidth; paths P_n have bandwidth 1 but unbounded treedepth.

See also [Sch99] for an overview of several graph parameters, relations between them and their application in CSPs.

Claw-free graphs A claw is the graph $K_{1,3}$, with vertex set $\{x, u, v, w\}$ and edges $\{xu, xv, xw\}$. A claw-free graph is a graph that does not contain a claw as an induced subgraph. The class of claw-free graphs includes complements of all triangle-free graphs, the class of line graphs, proper interval graphs and proper circular arc graphs.

Line graphs are graphs obtained as follows: given a graph G, the line graph of G, denoted as $L(G)$, has the edges of G as vertices, with two such edges adjacent whenever they share an edge.

Circular arc graphs are graphs obtained as follows: given a set of arcs on a circle (as subsets in $\mathbb{R}^2$), the corresponding graph has the arcs as vertices, with two arcs adjacent whenever they intersect. A graph is a proper circular arc graph if it can be obtained from a set of arcs on a circle such that no arc contains another.

Interval graphs and proper interval graphs are defined similarly as circular arc graphs, considering intersections of interval on the real line, instead of arcs on a circle. They form a proper subclass of circular arc graphs.

1.4. Homomorphisms

As a generalization of colorings Given two digraphs G, H, a homomorphism from G into H is a function $\mu : V(G) \to V(H)$ such that if $uv \in E(G)$ then $\mu(u)\mu(v) \in E(H)$ (in other words, arcs of G are mapped to arcs of H in the same direction). For two graphs G, H with loops allowed, a homomorphism from G to H is a function from vertices of G to vertices of H such that edges of G are mapped to edges of H; this is the same as a homomorphism between symmetric orientations of G and H (i.e., the digraphs obtained by replacing each edge uv by two arcs uv, vu).

Such an assignment of vertices of H (called *colors*) to vertices of G is also called an *H-coloring*, especially when H is fixed in the context. This generalizes proper k-colorings: a K_k-coloring of a graph (a homomorphism to the k-clique, which has edges everywhere except for loops) is the same as a proper k-coloring.

As constraint satisfaction problems (CSP for short), are triples $(\Delta, X, C\}$ where Δ is a finite, nonempty domain, $X = \{x_1, \ldots, x_n\}$ is a set of variables and $C = \{C_1, \ldots, C_m\}$ is a set of constraints. An *evaluation* assigns to each variable an element of the domain. Each *constraint* is a pair $C_i = (\vec{x}_i, R_i)$, where $\vec{x}_i$ is a k-tuple of variables and $R_i \subseteq \Delta^k$ is a k-ary relation. An evaluation *satisfies* a constraint $C_i = (\vec{x}_i, R_i)$ if the tuple of values assigned to the variables in $\vec{x}_i$ is in the relation R_i. An evaluation is a *solution* if it satisfies all constraints. The *constraint hypergraph* has variables as vertices and each tuple $\vec{x}_i$ as a hyperedge.

For a set of relations Γ, a CSP(Γ) instance is a CSP in which all relations belong to Γ. In the special case where Γ contains only one, binary relation R, so $H = (\Delta, R)$ is a digraph, solutions are exactly H-colorings of the constraint digraph. CSPs can also be defined as homomorphisms of more general relational systems.

A celebrated conjecture of Feder and Vardi [FV98] states that for every Γ, the problem of deciding the existence of a solution to a given CSP(Γ)-instance is either in P or NP-complete. They proved that such a dichotomy for H-Coloring problems (deciding the existence of an H-coloring of a given digraph) for digraphs would imply a dichotomy for general CSPs. The special case where H is a graph with loops allowed (that is, the adjacency relation is symmetric) has been proved by Hell and Nešetřil [HN90]. In this case, H-coloring is in P when H has a loop or is bipartite, and is NP-complete otherwise.

See [Neš07] for a broad survey of these and other results on graph homomorphism, or Hell and Nešetřil's book [HN04] for a more detailed overview.

In reconfiguration The systematic study of reconfiguration was initiated by the work of Gopalan et al. [Gop+09], where they studied the reconfiguration of SAT(Γ) problems, which are CSP(Γ) problems limited to the boolean domain (so variables take the value `true` or `false`, and a reconfiguration step consists of flipping the value of one variable). These generalized SAT problems were introduced by Schaefer [Sch78] to state his well known dichotomy theorem: for a finite set of boolean relations Γ, the problem of deciding satisfiability of SAT(Γ) formulas is either in P or is NP-complete (assuming P$\neq$NP). The sets for which it is in P are now known as *Schaefer* sets. The results of [Gop+09] (with a slight correction by Schwerdtfeger [Sch13], imply a similar dichotomy for SAT(Γ) Reachability problems, showing that for all Schaefer sets the problem is in P, for some non-Schaefer sets it is still in P, and for all others it is PSPACE-complete. In particular reconfiguration variants are in some cases easier than their underlying satisfiability problem, similarly as in the case of reconfiguring k-colorings. Similar results about connected components in the solution graphs of SAT(Γ) instances and their diameter are the main motivation behind this line of work, as they are hoped to give insight into the performance of heuristic methods (especially the *survey propagation algorithm*) in satisfiability algorithms.

In this thesis we consider the natural extension of SAT reconfiguration to arbitrary finite domains, but limit our attention to a single binary relation, or equivalently, graph and digraph homomorphisms. For graphs or digraphs G, H, the solution graph of homomorphisms from G to H has as vertices all H-colorings of G and edges between any two colorings that can be obtained from one another in one reconfiguration step – changing the color of one vertex of G. The H-Coloring Reachability problem asks whether in a given digraph G, two given H-colorings are in the same component of the solution graph; in other words, whether one can be transformed into another by changing one color (i.e., the mapping of one vertex of G) at a time, maintaining an H-coloring throughout.

In combinatorial algebraic topology Reconfiguration of homomorphisms has already been studied in the field of combinatorial algebraic topology, though from a different angle. The notion of $\times$-homotopy of homomorphisms as defined by Dochtermann [Doc09] is identical to reachability in the solution graph. The so-called Hom-complex, studied for its interesting categorical and topological properties, is a construction similar to the clique complex of our solution graph. These definitions were introduced to provide lower bounds on the chromatic number of graphs, a notoriously hard problem. A typical theorem derived from such methods is that for loopless graphs G, H, if the solution graph of $G \to H$ homomorphisms is connected for all G of degree at most d, then the chromatic number of H is at least $d/2$ (and is conjectured to be at least d) [BW04]. Studies have thus been mostly concerned with highly regular graphs for which the solution graph can be proved to be in some sense strongly connected. In particular we are not aware of any prior work on the computational complexity of general homomorphism reconfiguration problems.

Chapter 2

Sparse graphs

This chapter studies the complexity of reconfiguration problems in graphs that can be considered sparse. The general idea in the PSPACE-hardness results is to construct an arbitrarily complicated set of local rules with a fixed instance of the problem – connecting such instances in a path then allows to simulate the tape of a Turing Machine in a graph of bounded bandwidth. To formalize this into clearly delineated parts we give reductions from the word problem in string rewriting systems (also known as semi-Thue systems and essentially equivalent to unrestricted grammars and finitely presented monoids), whose ability to directly simulate Turing Machines is a well-known, classical result. We construct a very limited PSPACE-complete string rewriting system and interpret it as an intermediary reconfiguration problem, from which reductions to other problems are easy.

This intermediary problem is then seen as H-Coloring Reachability on the class of directed paths. We show that while for undirected graphs H-Coloring Reachability is always P in the class of paths (and more generally, trees), it is hard too for very simple graphs, such as the class of cycles. We then recall that graphs without long paths have bounded treedepth and provide an algorithm for homomorphism reconfiguration problems on such graphs. Together, this gives an exact characterization of classes of digraphs closed under subdigraphs and reorientation which admit polynomial algorithms for all homomorphism reconfiguration problems. The final section comments on the limitations of our approach, contrasting it with positive results obtained for similar questions.

2.1. String rewriting systems

A *string rewriting system* (SRS for short) is a pair (Σ, R) where Σ is a finite alphabet and R is a set of rules, where each rule is an ordered pair of words $(\alpha, \beta) \in \Sigma^* \times \Sigma^*$ (this is also known as a semi-Thue system). A rule can be applied to a word by replacing one subword by the other, that is, for two words $s, t \in \Sigma^*$, we write $s \to_R t$ if there is a rule $(\alpha, \beta) \in R$ and words $u, v \in \Sigma^*$ such that $s = u\alpha v$ and $t = u\beta v$. The reflexive transitive closure of this relation defines a reachability relation $\to_R^*$, where a word t can be reached from another s iff it can be obtained from s by repeated application of rules from R. The *word problem* of R is the problem of deciding, given two word $s, t \in \Sigma^*$, whether $s \to_R^* t$.

A string rewriting system is called *symmetric* when $(\alpha, \beta) \in R \iff (\beta, \alpha) \in R$, in other words, rules are unordered pairs and the reachability relation is symmetric (this is also known as a Thue system). A SRS is called *balanced* if for each rule $(\alpha, \beta) \in R$ we have $|\alpha| = |\beta|$, and 2-*balanced* if for each rule $(\alpha, \beta) \in R$, $|\alpha| = |\beta| = 2$. In a balanced system, only words of the same length can be equivalent.

The word problem of certain 2-balanced symmetric SRSs is known to be PSPACE-complete. This fact is a folklore variant of the undecidability of general SRSs, whose proof by Emil Post [Pos47] (and independently by A. A. Markov [Mar47]) was described as "the first unsolvability proof for a problem from classical mathematics". The essential steps are: encoding the configurations of a Turing machine as a string so that a transition corresponds to string rewriting, padding the encoding so that the corresponding system is balanced, and noticing that the non-reversibility of a TM transition (the asymmetry of the corresponding rewriting system) is not essential in deterministic TMs (see [LP80] for more on symmetric computation). An explicit proof of the fact for a balanced symmetric SRS can be found in [BO84] and can easily be adapted to give a 2-balanced symmetric SRS. We include a self-contained proof here for completeness.

Theorem 2.1. *There is a 2-balanced symmetric string rewriting system whose word problem is PSPACE-complete (under $\leqslant^m_{log}$-reducibility).*

Proof. Since only words of the same length can be reached by application of rules in a balanced SRS, it suffices to nondeterministically search all words of the same length to solve the problem in nondeterministic polynomial space. By Savitch's theorem [Sav70], this places the problem in PSPACE.

Let $M = (\Sigma, Q, q_0, q_{acc}, q_{rej}, \delta)$ be a deterministic Turing Machine working in space bounded by a polynomial $p(|x|)$ which accepts any PSPACE-complete language. (By starting from a fixed PSPACE-complete problem we show the word problem to be hard for a certain fixed SRS; starting from any language in PSPACE we would only show that the more general word problem, where the SRS is given as input, is PSPACE-complete). Σ is the tape alphabet of M, Q is the set of states, q_0, q_{acc}, q_{rej} are the initial, accepting, and rejecting state respectively, and $\delta : Q \times \Sigma \to Q \times \Sigma \times \{\cdot, L, R\}$ is the transition function of M. Let $\$, \not\in \Sigma$ denote the left and right end-markers. Assume w.l.o.g. that the machine clears the tape and moves its head to the left end when reaching the accepting state.

For any input $x \in \Sigma^*$ we encode a configuration of the Turing Machine by a word of length exactly $p(|x|)$ over the alphabet $\Gamma = \Sigma \cup (\Sigma \times Q) \cup \{\textvisiblespace\}$. If the tape content is $\$a_1a_2 \dots a_n\not\in$ for some $a_1, \dots, a_n \in \Sigma$, the head's position is $i \in \{0, 1, \dots, n+1\}$ and the machine's state is q, then we define the corresponding word to be the tape content padded with $\textvisiblespace$ symbols and with a_i replaced by (q, a_i), that is $\$a_1 \dots a_{i-1}(q, a_i)a_{i+1} \dots a_n\not\in\textvisiblespace\textvisiblespace \dots \textvisiblespace \in \Gamma^{p(|x|)}$. The initial configuration is then encoded as $s_x = (q_0, \$)x\not\in\textvisiblespace\textvisiblespace \dots \textvisiblespace \in \Gamma^{p(|x|)}$ and the only possible accepting configuration is encoded as $t_x = (q_{acc}, \$)\not\in\textvisiblespace\textvisiblespace \dots \textvisiblespace \in \Gamma^{p(|x|)}$. Since M never uses more than $p(|x|)$ space on input x, our encoding is well defined for all configurations appearing in the execution of M on x. So M accepts input x if and only if from s_x one reaches the configuration t_x by repeatedly applying the transition function. Such an application corresponds exactly to the following (ordered) string rewriting rules, in the encodings:

- $\big((q, a)c \ , \ (p, b)c\big)$ for $q \in Q$, $a, c \in \Sigma$ and $\delta(q, a) = (p, b, \cdot)$,
- $\big((q, a)c \ , \ b(p, c)\big)$ for $q \in Q$, $a, c \in \Sigma$ and $\delta(q, a) = (p, b, R)$,
- $\big(c(q, a) \ , \ (p, c)b\big)$ for $q \in Q$, $a, c \in \Sigma$ and $\delta(q, a) = (p, b, L)$.

The transition relation is not symmetric, but since the machine M is deterministic, the configuration digraph (with machine configurations as vertices and the transition function as the adjacency relation) has out-degree 1. The configuration t_x (which is a configuration in the accepting state) has a loop. Therefore from any configuration, t_x is reachable by a

directed path if and only if it is reachable by any path. This means that M accepts input x if and only if applying the transition rules to s_x leads to t_x if and only if $s_x \leftrightarrow_R^* t_x$, where R is the symmetric closure of the above rules, i.e., the 2-balanced symmetric SRS over Γ with rules:

- $\{(q,a)c, (p,b)c\}$ for $q \in Q$, $a, c \in \Sigma$ and $\delta(q,a) = (p,b,\cdot)$,
- $\{(q,a)c, b(p,c)\}$ for $q \in Q$, $a, c \in \Sigma$ and $\delta(q,a) = (p,b,R)$,
- $\{c(q,a), (p,c)b\}$ for $q \in Q$, $a, c \in \Sigma$ and $\delta(q,a) = (p,b,L)$.

Since the map $x \mapsto (s_x, t_x)$ is computable in logarithmic space, this proves the world problem of (Γ, R) to be PSPACE-hard. □

This result can be slightly strengthened to give a system where only one symbol at a time can be changed. To that aim, it suffices to replace a rule changing two symbols with a sequence of rules using two new intermediary symbols.

Lemma 2.2. *There is a 2-balanced symmetric SRS (Γ, R) whose word problem is PSPACE-complete and such that for every rule $\{a_1a_2, b_1b_2\} \in R$ either $a_1 = b_1$ or $a_2 = b_2$.*

Proof. Let (Σ, R) be the 2-balanced symmetric SRS from Theorem 2.1. Suppose $\{a_1a_2, b_1b_2\}$ is a rule of R in which $a_1 \neq b_1$ and $a_2 \neq b_2$. We construct a 2-balanced symmetric SRS (Γ, S) with one such rule fewer, preserving PSPACE-completeness of the word problem. The claim then follows inductively.

Let $\Gamma = \Sigma \cup \{X, Y\}$ where X and Y are new symbols. Let S be equal to R with rules $\{a_1a_2, Xa_2\}, \{Xa_2, XY\}, \{XY, b_1Y\}, \{b_1Y, b_1b_2\}$ added and rule $\{a_1a_2, b_1b_2\}$ removed. We show that for any $s, t \in \Sigma^*$ it holds that $s \leftrightarrow_R^* t$ if and only if $s \leftrightarrow_S^* t$, which implies that our construction preserves PSPACE-completeness.

Clearly if $s \leftrightarrow_R^* t$ then $s \leftrightarrow_S^* t$, because replacing a_1a_2 with b_1b_2 can be done in S by replacing a_1a_2 with Xa_2, then XY, then b_1Y and finally b_1b_2. Suppose now $s \leftrightarrow_S^* t$ for some $s, t \in \Sigma^*$. Then there is a sequence $s = u_0, u_1, u_2, \ldots, u_l = t$ of words $u_i \in \Gamma^*$ such that $u_i \leftrightarrow_S u_{i+1}$. Let $\phi : \Gamma^* \to \Sigma^*$ be defined by replacing all XY substrings of a word with a_1a_2, then replacing all remaining X symbols with a_1 and all remaining Y symbols with b_2. It is easy to check that $\phi(u_i) \leftrightarrow_R \phi(u_{i+1})$ or $\phi(u_i) = \phi(u_{i+1})$. Since $\phi(u_0) = \phi(s) = s$ and $\phi(u_l) = \phi(t) = t$, this implies that $s \leftrightarrow_R^* t$. □

2.2. A simple intermediary problem

We define an intermediary problem that highlights how simple a reconfiguration problem achieving PSPACE-hardness can be. Given a pair $H = (\Sigma, E)$, where Σ is an alphabet and $E \subseteq \Sigma^2$ a binary relation between symbols, we say that a word over Σ is an *H-word* if every two consecutive symbols are in the relation (put differently, no element of $\Sigma^2 \setminus E$ is a subword). If one looks at H as a digraph (possibly with loops), a word is an H-word iff it is a walk in H. The H-Word Reachability problem asks whether two given H-words of equal length can be transformed into one another by changing one symbol at a time so that all intermediary steps are also H-words.

Theorem 2.3. *There is a digraph H for which H-Word Reachability is PSPACE-complete.*

Proof. Let (Γ, S) be the 2-balanced symmetric string rewriting system from Lemma 2.2 (so if $\{a_1a_2, b_1b_2\} \in S$ then $a_1 = b_1$ or $a_2 = b_2$). Let $S = \{S_1, \ldots, S_m\}$.

Let $\textvisiblespace, \$, \not\in, x_1, \ldots, x_m$ be new symbols, let $\Delta_1 = \{\$, \not\in, x_1, \ldots, x_m\}$, $\Delta_2 = (\Gamma \cup \{\textvisiblespace\}) \times (\Gamma \cup \{\textvisiblespace\})$, and let $\Delta = \Delta_1 \cup \Delta_2$. We will call Δ_1 *special symbols* and Δ_2 *pair symbols*. Let $H = (\Delta, E)$, where we define $E \subseteq \Delta^2$ as the relation containing the following pairs

- $((a, b), (b, c))$ for any $a, b, c \in \Gamma$,
- $(\$, (\textvisiblespace, a))$ for any $a \in \Gamma$,
- $((a, \textvisiblespace), \not\in)$ for any $a, b, c \in \Gamma$,
- $((\cdot, a_1), x_i)$,
- $((\cdot, b_1), x_i)$,
- $(x_i, (a_2, \cdot))$,
- $(x_i, (b_2, \cdot))$ for any $\cdot \in \Gamma$ and $i \in \{1, \ldots, m\}$ such that $S_i = \{a_1a_2, b_1b_2\}$.

Let $(s, t) \in \Gamma^* \times \Gamma^*$ be an instance of the word problem for S, w.l.o.g. $|s| = |t| = n$. Define $\psi : \Gamma^n \to \Delta^{n+3}$ as

$$\psi(a_1a_2 \ldots a_n) = \$(\textvisiblespace, a_1)(a_1, a_2)(a_2, a_3) \ldots (a_{n-1}, a_n)(a_n, \textvisiblespace)\not\in$$

It is easy to see that if $s \leftrightarrow_S^* t$ then $\psi(s)$ can be transformed into $\psi(t)$, e.g., applying the rule $S_i = \{a_1a_2, b_1a_2\}$ corresponds to replacing $(\cdot, a_1)(a_1, a_2)(a_2, \cdot)$ by $(\cdot, a_1)x_i(a_2, \cdot)$, then $(\cdot, b_1)x_i(a_2, \cdot)$, then $(\cdot, b_1)(b_1, a_2)(a_2, \cdot)$. We will show the other direction, that if $\psi(s)$ can be transformed into $\psi(t)$, then $s \leftrightarrow_S^* t$. Since ψ is computable in logarithmic space, this will imply our claim of PSPACE-completeness.

Indeed, suppose that there is a sequence of H-words $\psi(s) = u_0, u_1, \ldots, u_l = \psi(t)$ with $u_j \in \Delta^{n+3}$, such that u_j differs from u_{j+1} only at one position. In any H-word $v = v_1v_2 \ldots v_{n+3} \in \Delta^{n+3}$ there cannot be two consecutive special symbols. We can thus define a word $\phi(v)$ of length n over Γ such that its i-th symbol, for $i \in \{1, \ldots, n\}$, is the second element of v_{i+1} if v_{i+1} is a pair symbol and the first element of v_{i+2} if v_{i+2} is a pair symbol (either case must hold and if both do, the definitions agree by construction of E). In particular $\phi(\psi(v)) = v$ for any $v \in \Gamma^n$. We argue that $\phi(u_{j-1}) \leftrightarrow_S^* \phi(u_j)$ for $j \in \{1, \ldots, l\}$.

Notice that the special symbol \$ must precede a pair symbol $(\textvisiblespace, \cdot)$ for some $\cdot \in \Gamma$ and any such pair symbol must be preceded by \$. Since only one symbol at a time can be changed, it follow inductively that for each $j \in \{0, \ldots, l\}$ the first two symbols of u_j must be $\$(\textvisiblespace, \cdot)$ for some $\cdot \in \Gamma$ and \$ appears nowhere else. Similarly for the last two symbols, $(\cdot, \textvisiblespace)\not\in$ for some $\cdot \in \Gamma$.

Since u_{j-1} and u_j differ at only one position, there are non-empty words $v, w \in \Delta^*$ and symbols $a, b \in \Delta$, $a \neq b$ such that $u_{j-1} = vaw$ and $u_j = vbw$. If a or b is a special symbol then both the last symbol of v and the first symbol of w are pair symbols, so $\phi(u_{j-1}) = \phi(u_j)$. Otherwise, let $a = (a_1, a_2), b = (b_1, b_2)$. Assume without loss of generality that $a_1 \neq b_1$ and $a_2 = b_2$ (the case $a_1 = b_1, a_2 \neq b_2$ is analogous and the case $a_1 \neq b_1, a_2 \neq b_2$ can be split by showing that $\phi(u_{j-1}) \leftrightarrow_S^* \phi(u')$ and $\phi(u') \leftrightarrow_S^* \phi(u_j)$ for $u' = v(b_1, a_2)w$, which can easily be checked to be an H-word). If the last symbol of v is a pair symbol (c, d), then $d = a_1$ and $d = b_1$, contradicting our assumption. If the last symbol of v is \$, then $a_1 = b_1 = \textvisiblespace$.

Finally if the last symbol of v is x_i for some $i \in \{1, \ldots, m\}$, then S_i must be equal $\{ca_1, c'b_1\}$ for some $c, c' \in \Gamma$. Since $a_1 \neq b_1$, we have $c = c'$ and the last but one symbol of v must be a pair $(\cdot, c)$ for some $\cdot \in \Gamma \cup \{\_\}$. Thus $\phi(v(b_1, a_2)w)$ is obtained from $\phi(v(a_1, a_2)w)$ by replacing the symbol a_1 at position $|v|$, which is preceded by a c, by the symbol b_1, that is, $\phi(v(b_1, a_2)w) \leftrightarrow_S \phi(v(a_1, a_2)w)$. □

Notice that in the case of H-Word Reachability, the decision problem asking for the existence of a solution of given length is trivial (even more so since H is fixed), and even counting solutions or extensions of a partial assignment to solutions is easy. This shows the complexity of a reconfiguration variant of a combinatorial problem can be very different from the complexity of the original problem and its static variants.

2.3. Hardness in bounded bandwidth

In this section we give simple reductions that show several reconfiguration problems studied in earlier literature to be PSPACE-complete in graphs of bounded bandwidth.

Proposition 2.4. *There is an integer b such that* Shortest Path Reachability *is PSPACE-complete even when limited to graphs of bandwidth at most b.*

Proof. Let $H = (\Sigma, R)$ be the graph from Theorem 2.3, we show the lemma for $b = 2|\Sigma|$. Let $s, t \in \Sigma^*$ be an instance of H-Word Reachability with $|s| = |t| = n$. We construct an instance (G_n, P_s, P_t) of Shortest Path Reachability as follows. The graph G_n depends only on n (and the fixed graph H). Its vertex set contains v_0, v_{n+1} and vertices v_i^a for all $i \in \{1, \ldots, n\}$ and $a \in \Sigma$. Its edge set contains $v_0 v_1^a$ and $v_n^a v_{n+1}$ for all $a \in \Sigma$, and $v_i^a v_{i+1}^b$ for all $(a, b) \in R$ and $i \in \{1, \ldots, n-1\}$. Let $V_i = \{v_i^a \mid a \in \Sigma\}, V_0 = \{v_0\}, V_{n+1} = \{v_{n+1}\}$. The sets V_i give a bucket arrangement, so the bandwidth of G_n is at most $2|\Sigma|$.

A shortest path from v_0 to v_{n+1} must go through exactly one vertex in each set V_i and thus defines a word. It is easy to see that this defines an bijection between H-words and shortest paths. We let P_s, P_t be the paths corresponding to s, t. Changing one symbol corresponds to changing one vertex of the path, so the instances are clearly equivalent. □

The same reduction can be applied to Maximum Independent Set Reachability by taking complements of edge sets on each bucket pair $V_i \cup V_{i+1}$ from the previous construction. This is equivalent to using the reduction from [KMM11, Theorem 4].

Proposition 2.5. *There is an integer b such that* Maximum Independent Set Reachability*, and thus* TS, TJ *and* TAR Reachability*, are PSPACE-complete even when limited to graphs of bandwidth at most b.*

The *chain of onions of width b and length n* is the graph with vertex set $\{u_1, \ldots, u_n\} \cup \{v_i^j \mid i = 1, \ldots, n-1,\ j = 1, \ldots, b\}$ and edges $u_i v_i^j$ and $v_i^j u_{i+1}$ for $i \in \{1, \ldots, n-1\}, j \in \{1, \ldots, b\}$. Clearly such graphs have bandwidth b (and pathwidth 2).

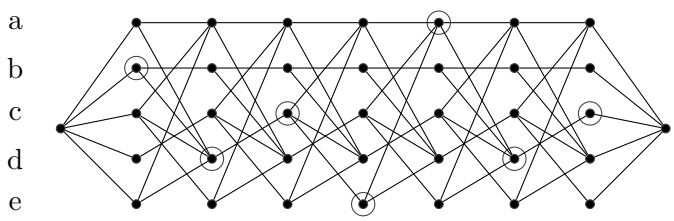


Figure 2.6: Example of a shortest path configuration corresponding to the word *bdceadc*, as constructed in Proposition 2.4 for a digraph H with vertices $\{a, b, c, d, e\}$ and arcs $aa, ad, bb, bd, ca, cd, ce, dc, ea, ed$.

Proposition 2.7. *There are integers k, b such that k-List-Coloring Reachability is PSPACE-complete even when limited to chains of onions of width b.*

Proof. Let $H = (\Sigma, R)$ be the graph from Theorem 2.3, let $b = |\Sigma^2 \setminus R|$, $k = 2|\Sigma|$ and let $s, t \in \Sigma$ be an instance of H-Word Reachability with $|s| = |t| = n$. We construct the following instance of k-List-Coloring Reachability. The graph will be the chain of onions of width b and length n with vertices named as above. Let Σ' be a disjoint copy of Σ, we write a' for the copy of $a \in \Sigma$, the set of available colors will be $\Sigma \cup \Sigma'$. The list of a vertex u_i is Σ for i even and Σ' for i odd. For each $(a, b) \in \Sigma^2 \setminus R$ we choose a different $j \in \{1, \ldots, |\Sigma^2 \setminus R|\}$ and let v_i^j have the list $\{a, b'\}$ for i even and $\{a', b\}$ for i odd.

A coloring of vertices u_i defines a word in Σ^n (by letting the i-th symbol be a if the color of u_i is a or a'). If $(a, b) \in \Sigma^2 \setminus R$, then for any even i, vertices u_i, u_{i+1}, with lists Σ, Σ' respectively, are adjacent to some vertex v_i^j with list $\{a, b'\}$. Thus in any proper coloring it cannot be that u_i has color a and u_{i+1} has color b'. Similarly for odd indices, thus ab is never a subword of a word corresponding to a proper coloring. Since this holds for all $(a, b) \in \Sigma^2 \setminus R$, such a word is an H-word. Conversely, for any H-word, the corresponding coloring of u_i can easily be extended to a proper coloring of all the graph. Changing one color corresponds to changing at most one symbol of the corresponding H-word, and changing one symbol corresponds to changing the colors of one vertex u_i and of those vertices adjacent to it that would have the same color. This can be done while maintaining a proper coloring, so the instances are equivalent. □

Proposition 2.8. *There are integers k, b such that k-Coloring Reachability is PSPACE-complete even when limited to graphs of bandwidth at most b.*

Proof. Let k, b be integers from Proposition 2.7 and let $b' = b + k$. An instance of k-List-Coloring Reachability of bandwidth b can be transformed into an equivalent instance of k-Coloring Reachability of bandwidth at most b' simply by adding a clique of size k to the graph for each original vertex, assigning to their vertices all the colors in some order and replacing the list of each original vertex by edges to vertices of its clique that have colors outside of the list. The colors of any k-clique clearly can never be changed in the new instance, and an extension of the cliques' coloring to a coloring of the new instance is proper if and only if it is a proper list-coloring of the original instance. □

2.4. Homomorphism reconfiguration on paths, trees and cycles

In this section we describe how the intermediary problem relates to homomorphisms and explore its variants for undirected graphs. Notice that an H-coloring of a directed path is the same as an H-word (colors correspond to symbols). Therefore Theorem 2.3 can be reformulated as follows.

Corollary 2.9. *There is a digraph H for which H-Coloring Reachability is PSPACE-complete even on directed paths.*

For an undirected graph H, reconfiguring H-colorings a path turns out to be computationally easy. The idea is to reach a coloring $ababa\ldots$, for some edge ab of H in quadratically many steps, and then only moving between such alternating colorings. This is easily generalized to trees, as in the following proposition.

Proposition 2.10. *Let H be a graph (possibly with loops) and let α, β be two H-colorings of a tree T with root r (chosen arbitrarily) and at least 2 vertices. Then α can be reconfigured to β if and only if there is in H a walk of even length between the colors assigned to r in the two colorings.*

Proof. Let T_i be the vertices at distance exactly i from r in the tree T. We first prove inductively that for $i = 0, \ldots, n-1$, any H-coloring γ of T can be reconfigured into a coloring γ_i such that

- γ and γ_i assign the same colors to each vertex in $T_0, T_1, \ldots, T_{n-i}$.
- for every $v \in T_j$ with $j \geqslant n - i + 1$, γ_i assigns the same color to v and its grandparent.

For $i = 0$ it suffices to take $\gamma_0 = \gamma$. For $i + 1$, it suffices to take the coloring γ_i and recolor one by one each vertex v in $T_{n-i}, T_{n-i+2}, T_{n-i+4}, \ldots$ (in that order) to the color of v's grandparent – since all of v's sons have the same color as v's parent, which is a color adjacent in H to the color of v's grandparent, this remains a valid H-coloring.

Let $S = T_0 \cup T_2 \cup \ldots$ and $S' = T_1 \cup T_3 \cup \ldots$. In α_{n-1} all vertices in S have the same color, say a, and also all vertices in S' have the same color, say a'. Similarly for β_{n-1} with colors b, b'. If there is an even-length walk $a_0, a_1, \ldots, a_{2l}$ in H with $a_0 = a$ and $a_{2l} = b$, then all S' can be recolored from a' to a_1, S from a to a_2, S' to a_3 and so on until S' is recolored a_{2l-1} and S is recolored a_{2l}, after which S' can be recolored to b' to get β_{n-1}.

If, on the other hand, there is no even-length walk from a to b, then they must belong to different components of H or to different sides of a bipartition of a component of H. Since every vertex has some neighbor, the colors of vertices in T must remain in the same component of H and if the component is bipartite, the colors of S must be and remain on one side, while the colors of S' remain on the other side. In either case, the color a cannot be changed to b. □

Nevertheless, H-Coloring Reachability for an undirected graph H can still be hard on very simple graphs. We prove this for cycles, but the following proof can easily be adapted to give other graphs, like paths with a triangle attached to each end.

Proposition 2.11. *There is a graph H for which H-Coloring Reachability is PSPACE-complete even on cycles.*

Proof. Let $H = (\Sigma, E)$ be the digraph from Corollary 2.9. The construction can be easily adapted so that H-Coloring Reachability is PSPACE-complete on directed cycles of length divisible by 3. Let $H' = (\Sigma', E')$ be a graph with $\Sigma' = \Sigma \times \{0, 1, 2\}$ and $E' = \{\{(a, i), (b, i+1 \mod 3)\} \mid (a, b) \in E\}$. For an H-coloring α of a directed cycle $v_1, \ldots, v_{3n}, v_1$, define $\alpha'(v_i) = (\alpha(v_i), i \mod 3)$ to be an H'-coloring of the underlying undirected cycle. The second element of each pair color $\alpha'(v_i)$ cannot ever be changed (the projection to the second elements gives a 'frozen' 3-coloring). The relation constraining the first elements is hence exactly E, the direction being implied by the 3-coloring. Therefore if α, β are two H-colorings of the directed cycle, then one can be recolored into the other if and only if α' can be recolored into β' as H'-colorings of the undirected cycle. □

2.5. Treedepth

We use the following theorem that formally describes the idea that graphs of bounded treedepth can be reduced to graphs of bounded size by merging copies of vertices that behave the same way.

Theorem 2.12 ([NOdM06]). *For all integers N and t, there is an integer $F(N,t)$ such that for any graph G of treedepth at most t and any mapping $g : V(G) \to \{1,\dots,N\}$, there is a subset A of $V(G)$ of cardinality at most $F(N,t)$ such that G has a g-preserving homomorphism to $G[A]$.*

If one defines the treedepth of a digraph to be the treedepth of the underlying graph, the proof of Theorem 2.12 can easily be extended to digraphs (giving homomorphisms that preserve edge directions). Seeing that the proof is constructive, such a homomorphism can be computed in time polynomial in $F(N,t) \cdot n$. In the following, H refers to any digraph, and may be given as part of the input.

Proposition 2.13. *H-Coloring Reachability on loopless digraphs of treedepth at most t can be solved in time $\mathcal{O}(f(|H|,t)n^{\mathcal{O}(1)})$ for some f.*

Proof. Let G be a loopless digraph of treedepth t and let α, β be two H-colorings. Let $N = |V(H)|^2$ and define $g(v) = (\alpha(v), \beta(v))$. By Theorem 2.12 there is an α- and β-preserving homomorphism μ from G to some induced subdigraph $G[A]$ of cardinality at most $F(N,t)$. We claim that α can be reconfigured to β in G if and only if $\alpha|_A$ can be reconfigured to $\beta|_A$ in $G[A]$. The latter can be checked by brute force after finding the homomorphism, giving the claimed algorithm.

Let $\alpha_0, \alpha_1, \alpha_2, \dots, \alpha_l = \beta$ be a reconfiguration sequence from $\alpha_0 = \alpha$ to $\alpha_l = \beta$ in G. Then the sequence $\alpha_i|_A$ is a reconfiguration sequence in $G[A]$.

Let $\alpha_0, \alpha_1, \alpha_2, \dots, \alpha_l = \beta$ be a reconfiguration sequence from $\alpha_0 = \alpha$ to $\alpha_l = \beta$ in $G[A]$. Define mappings $\alpha'_i : V(G) \to H$ as $\alpha'_i(v) := \alpha_i(\mu(v))$. Suppose α'_i is not a proper coloring – then there is an edge $vw \in E(G)$ such that $\alpha'_i(v)\alpha'_i(w) \notin E(H)$, that is, $\alpha_i(\mu(v))\alpha_i(\mu(w)) \notin E(H)$. But μ is a homomorphism, so $\mu(v)\mu(w)$ is an edge of $G[A]$, contradicting that α_i is a proper H-coloring.

Since μ is α and β-preserving, we have $\alpha'_0 = \alpha$ and $\alpha'_l = \beta$. It suffices to show that α'_i can be reconfigured to α'_{i+1} for all i. Let v be the only vertex of A on which α_i and α_{i+1} differ. Then α'_i differs from α'_{i+1} only on $\mu^{-1}(v)$, where one coloring assigns the color $\alpha_i(v)$ and the other assigns $\alpha_{i+1}(v)$ to all vertices. Since all of these vertices map to v through a homomorphism and there is no loop vv in $E(G)$, $\mu^{-1}(v)$ is an independent set of G. Therefore, α'_i can be reconfigured to α'_{i+1} by changing the colors of vertices in $\mu^{-1}(v)$ from $\alpha_i(v)$ to $\alpha_{i+1}(v)$ one by one, in any order. □

Note that a class of graphs has unbounded treedepth if and only if it has graphs with arbitrarily long (undirected) paths as subgraphs [NOdM08]. Therefore, by Corollary 2.9, if a class $\mathcal{C}$ of graphs closed under taking subgraphs has unbounded treedepth, then for some digraph H, H-Coloring Reachability is PSPACE-complete on orientations of $\mathcal{C}$ (the class of digraphs obtained by orienting edges of a graph in $\mathcal{C}$). This suggests that to find any interesting regularities in the complexity of reachability problems, limiting the structure formed by constraints between variables is not enough, the constraints themselves have to be very simple.

Theorem 2.14. *Let H^* be the digraph from Corollary 2.9 and assume P≠PSPACE. For any class $\mathcal{C}$ of graphs without loops closed under taking subgraphs, the following statements are equivalent:*

- *$\mathcal{C}$ has bounded treedepth.*
- *H-Coloring Reachability on orientations of $\mathcal{C}$ is in P for any digraph H.*
- *H^*-Coloring Reachability on orientations of $\mathcal{C}$ is in P.*

It is well known that much weaker conditions are needed for the underlying problems: H-Coloring is solvable in polynomial time (for every fixed H) for any class of bounded treewidth. See [Gro07] for a tight characterization.

2.6. Final remarks

While we argued that no analogue of Courcelle's theorem can hold for reachability problems, Mouawad et al. [Mou+14] have very recently shown that many *shortest reachability* problems *parameterized by the length of the reconfiguration sequence* ℓ are fixed parameter tractable in graphs of bounded treewidth (solvable in time $f(\mathrm{tw}, \ell) \cdot |V(G)|$ for some function f). The proof applies to any vertex-subset problem to which Courcelle's theorem applies (including, e.g., token jumping) and works by constructing a formula in monadic second order logic describing the existence of the ℓ steps of the reconfiguration sequence. More direct algorithms with better running times are also given for some problems.

Not all natural reachability problems are hard in graphs of bounded treewidth. Notice that Clique Reachability (in any model where adjacency of configurations can be tested in polynomial time) becomes trivial, since a clique must be contained in one bag of a tree decomposition. There are at most $2^{t+1}n$ different cliques in a graph of treewidth t and they can be all enumerated in time $\mathcal{O}(2^{t+1}n)$ (see e.g. [Die12]).

Our results should also be contrasted with positive results about the reconfiguration of colorings in graphs of bounded treewidth. In particular Dyer et al. [Dye+06] proved that for any graph of treewidth t (or even any graph of degeneracy t) and any $k \geqslant t+2$, all k-colorings can be reached from one another (see also [Cer07; BB13]). In Proposition 2.8 the bandwidth, though constant, is strictly larger than the number of colors.

The specific value of bandwidth for which our reductions show hardness depend polynomially on the size (alphabet size times the number of states) of the PSPACE-complete Turing machine we start from. There are surprisingly small undecidable Turing machines and (non-balanced) string rewriting systems (e.g. [Tse58]), but we have been unable to find any similar work about small PSPACE-hard balanced systems or linearly bounded Turing machines. Let us only mention that instances with exponentially long solutions for Shortest Path Reachability in graphs of bandwidth 13 are explicitly given in [KMM11], by a simple construction that may be seen as an illustrative case of some of the ideas presented here.

This chapter only analyzed the complexity of the reachability problem between two solutions. While the techniques can be easily adapted to answer some related questions (like the complexity of reaching any solution containing a given element or the diameter of solution graphs) others (like the complexity of deciding the connectivity of the solution graph) seem for now elusive.

Chapter 3

Independent set reconfiguration in claw-free graphs

In this chapter we show a polynomial time algorithm for TS Reachability and TJ Reachability in claw-free graphs. As a first step, we show that when the initial and target independent sets induce no cycles, the answer is always yes, which will imply the equivalence of reachability in token sliding and token jumping models for connected claw-free graphs. We then show that cycles can be analyzed one by one and it suffices to ask whether each can be resolved in a weak sense. The case when the independent sets being reconfigured are not maximum is considered separately to handle disconnected graphs, but essentially the same idea is used: every cycle can easily be resolved, implying that the answer is always yes in this case, in the token jumping model.

We build upon the well-known techniques of augmenting paths that were used to give polynomial algorithms for finding maximum independent sets in claw-free graphs. By describing the neighborhood of a cycle we need to resolve and how resolving sequences relate to it, the problem is split into two cases, internal and external, which we are able to characterize in a way that allows to find resolving sequences in polynomial time. We then summarize how the different cases are handled in our algorithm. The last section describes reductions that show why the algorithm cannot be improved to give shortest sequences, even with very restricted inputs.

3.1. Preliminaries

We say that a graph G contains a *v-claw with leaves x, y, z* if v, x, y, z are four different vertices of G such that v is adjacent to x, y, z, while x, y, z are non-adjacent to each other. A *claw-free* graph is then a graph that contains no claw. Given an independent set I of a graph G and two adjacent vertices u, v, the set obtained by *sliding* $u \to v$ is $I - u + v$. A token sliding sequence is walk in $TS_k(G)$, i.e., a sequence of independent sets $I_0, \ldots, I_l$ of G such that I_{i+1} is obtained from I_i by sliding a token. We write $I_0 \leftrightarrow_{\mathrm{TS}} I_l$ when I_0 can be reconfigured into I_l in the token sliding model, i.e., when there is such a sequence. TS Reachability is thus the problem of deciding, given G, I, J, whether $I \leftrightarrow_{\mathrm{TS}} J$. Definitions for the token jumping model are analogous, only the requirement of u being adjacent to v is dropped.

Let us begin with a simple observation used implicitly throughout this chapter.

Lemma 3.1. *Let I and J be independent sets in a claw-free graph G. Then every connected component of $G[I \Delta J]$ is an isolated vertex, a path, or an even length cycle.*

Proof. By definition of $c$law-free, no vertex can have three neighbors in an independent set. This implies that $I \Delta J$ induces a subgraph of degree bounded by 2. □

In the next lemma we show that for connected graphs, when the initial and target configuration induce only paths and isolated vertices, there always exists a reconfiguration sequence between them. This would be easy in the token jumping model, but we prove it for token sliding, showing as a corollary that the two models are equivalent in connected graphs.

Lemma 3.2. *Let I and J be independent sets in a connected claw-free graph G with $|I| = |J|$ such that $G[I \Delta J]$ contains no cycles. Then $I \leftrightarrow_{\mathrm{TS}} J$.*

Proof. Define the *minimum distance* $\mathrm{md}(I, J)$ to be the minimum of $\mathrm{d}(u, v)$ over all pairs $u \in I \setminus J$ and $v \in J \setminus I$ (assume $\mathrm{md}(I, J) = \infty$ for $I = J$). We will prove that, with at most two token slides that do not introduce cycles into $G[I \Delta J]$, either $|I \setminus J|$ can be decreased (possibly increasing $\mathrm{md}(I, J)$ up to $\mathrm{diam}(G)$) or $\mathrm{md}(I, J)$ can be decreased without increasing $|I \setminus J|$.

First let us consider the case that $\mathrm{md}(I, J) = 1$. This means that $G[I \Delta J]$ contains at least one edge. Since $G[I \Delta J]$ contains no cycles, it contains a path P with at least two vertices. Choose an end vertex v of P. Suppose first that $v \in J$. Let u be the vertex on P that is adjacent to v (so $u \in I$). Then in I, the token from u can be moved to v, which yields an independent set I' such that $I' \setminus J = (I \setminus J) \setminus \{u\}$. This decreases $|I \setminus J|$, that is $|I' \setminus J| = |(I \setminus J) \setminus \{u\}| = |I \setminus J| - 1$ and clearly $G[I' \Delta J]$ contains no cycles. If on the other hand $v \in I$, then from J we obtain J' by sliding the adjacent token to v, and the statement can be proved analogously.

Now suppose that $\mathrm{md}(I, J) \geqslant 2$, let $d = \mathrm{md}(I, J)$. Choose $u \in I \setminus J$ and $v \in J \setminus I$ such that $\mathrm{d}(u, v) = d$, and let $P = v_0, \ldots, v_d$ be a shortest path between $v_0 = u$ and $v_d = v$. Notice that v_1 cannot have two neighbors x, y in J – one of them would be adjacent to v (otherwise there would be a v_1-claw with leaves v, x, y), and thus not in I, contradicting $\mathrm{md}(I, J) \geqslant 1$. We intend to slide the token from u to v along the path P. Define $I_1 = I - u + v_1$. If I_1 is an independent set, then we can slide a token from u to v_1 decreasing $\mathrm{md}(I, J)$, that is, $\mathrm{md}(I_1, J) \leqslant \mathrm{d}(v_1, v_d) = d - 1 < \mathrm{md}(I, J)$. Since v_1 doesn't have two neighbor in J, $G[I_1 \Delta J]$ contains no cycles.

So now suppose I_1 is not an independent set, that is, v_1 has a neighbor $x \in I$. Then $x = v_2$ or x must be a neighbor of v_2, otherwise there would be a v_1-claw with leaves v_0, v_2, x. Therefore $\mathrm{d}(x, v_d) \leqslant d - 1$ and since $\mathrm{md}(I, J) = d$, it must be that $x \in I \cap J$. Since v_1 has no other neighbor in J and v_0 has no neighbor in J, the set J can be reconfigured by sliding a token from x to v_1 and then to v_0, reaching $J' = J - x + v_0$. This does not increase $|I \setminus J|$, but decreases $\mathrm{md}(I, J)$, that is, $|I \setminus J'| = |(I \setminus J) - v_0 + x| = |I \setminus J| - 1 + 1$ and $\mathrm{md}(I, J') \leqslant \mathrm{d}(x, v_d) \leqslant d - 1 < \mathrm{md}(I, J)$. Since $x \in I \cap J$ has no neighbors in neither I nor J, $G[I \Delta J']$ contains no cycles.

In each case, we give token sliding sequences between I and I' and between J and J' of total length at most two, such that $G[I' \Delta J']$ still does not contain cycles and either $|I' \setminus J'| < |I \setminus J|$, or $|I' \setminus J'| = |I \setminus J|$ and $\mathrm{md}(I', J') < |\mathrm{md}(I, J)|$. Since for any $I \neq J$ we have $\mathrm{md}(I, J) \leqslant \mathrm{diam}(G)$, this inductively implies that in at most $2 \cdot |I \setminus J| \cdot \mathrm{diam}(G)$ steps $I' = J'$ will eventually be achieved. □

Corollary 3.3. *Let I and J be independent sets in a connected claw-free graph G. Then $I \leftrightarrow_{\mathrm{TS}} J$ if and only if $I \leftrightarrow_{\mathrm{TJ}} J$.*

Proof. It suffices to notice that any token jump can be replaced by a sequence of token slides. Indeed, if one independent set is obtained from another by removing a vertex and adding another, then their symmetric difference has exactly two vertices and thus no cycles, so Lemma 3.2 applies. □

As another corollary, the solution graph of token sliding reconfiguration for connected in connected claw-free graphs that are also even-hole-free (this answers a question of [KMM12], where the same has been proved for token jumping).

3.2. Nonmaximum case

In this section we consider the case when the independent set being reconfigured is not maximum. This allows to obtain a configuration with a vertex that has no tokens on it or in its neighborhood, which in turn allows to resolve all cycles easily. This way, we show that any two nonmaximum independent sets of equal size can be reconfigured into one another in the token jumping model, which by Corollary 3.3 implies the same in the token sliding model for connected graphs G.

We use the well known techniques of augmenting paths: for an independent set I in a claw-free graph G, an *augmenting path* is an induced path $v_0, v_1, \ldots, v_l$ with $l > 0$ in G such that $v_1, v_3, \ldots, v_{l-1} \in I$ and v_0, v_l have exactly one neighbor in I (v_1, v_{l-1} respectively). Let us recall the results obtained independently by [Min80; Sbi80] that allow to find maximum independent sets in claw-free graphs (we remark that finding augmenting paths is highly non-trivial – simple algorithms that might come to mind will not find induced paths).

Lemma 3.4 ([Sch03])**.** *Let I be a nonmaximum independent set in a claw-free graph G. Then I is not maximal or has an I-augmenting path.*

Theorem 3.5 ([Sch03])**.** *There is a polynomial time algorithm that given a claw-free graph G, an independent set I of G and two vertices, finds an I-augmenting path between the vertices, if one exists.*

Lemma 3.6. *Let I and J be nonmaximum independent sets of equal size in a claw-free graph G. Then $I \leftrightarrow_{\mathrm{TJ}} J$.*

Proof. If I is maximal, then by Lemma 3.4 there exists an I-augmenting path $P = v_0, v_1, \ldots, v_l$. It is easy to see that $I \Delta P$ is an independent set, and that one can then slide tokens $v_1 \to v_0, v_3 \to v_2, v_5 \to v_4, \ldots, v_{l-1} \to v_{l-2}$ to reach the set $I \Delta (P - v_l)$. This set is not maximal, since v_l can be added.

Assume thus that I is not maximal, i.e., $I \cup \{v\}$ is an independent set for some vertex v. Consider the components of $G[I \Delta J]$.

If $G[I \Delta J]$ contains a cycle $C = c_0, c_1, \ldots, c_{2n-1}, c_0$ in $G[I \Delta J]$ with $c_{2i} \in I$, then one can jump tokens $c_0 \to v, c_2 \to c_1, c_4 \to c_3, \ldots, c_{2n-2} \to c_{2n-3}, v \to c_{2n-1}$ to reach $I \Delta C$ which again isn't maximal.

If $G[I \Delta J]$ contains a path with an even number of vertices $P = v_1, \ldots, v_{2n}$ with $v_{2i} \in I$, then one can jump tokens $v_2 \to v_1, v_4 \to v_3, \ldots v_{2n} \to v_{2n-1}$ to reach $I \Delta P$ which again isn't maximal.

If $G[I \Delta J]$ contains an isolated vertex or a path with an odd number of vertices, then since $|I| = |J|$, it contains an isolated vertex $u \in J \setminus I$ or a path $P = v_0, v_1, \ldots, v_{2n}$ with $v_{2i} \in J \setminus I$. In the first case, one can jump a token from any vertex in $v' \in I \setminus J$ to u – since v'

is not adjacent to u, the set obtained again isn't maximal, as v' can be added. In the second case, one can jump a tokens $v_1 \to v_0, v_3 \to v_2, \ldots, v_{2n-1} \to v_{2n-2}$ to reach $I\Delta(P - v_{2n})$, which again isn't maximal because v_{2n} can be added.

In this way we inductively decrease $|I \setminus J|$ in each case until J is reached. □

Corollary 3.7. *Let I and J be nonmaximum independent sets of equal size in a connected claw-free graph G. Then $I \leftrightarrow_{\mathrm{TS}} J$.*

3.3. Resolving cycles

The following sections need only consider maximum independent sets, in which case a token jump is the same as a token slide, even if G is not connected (any jump $u \to v$ must be between adjacent vertices, otherwise the set with both u and v added would be a larger independent set). We therefore focus on token sliding.

By Lemma 3.2 it remains to handle cycles induced by the problem instance. In this section we show that essentially any way of resolving a cycle is enough and that each cycle can be considered independently.

We write that $[A, B]$ *is an induced cycle in* G if $G[A \cup B]$ is an even cycle alternating between vertices of A and B (equivalently, $G[A \cup B]$ is a cycle and A, B are independent sets). For an independent set I of G, we say that an induced cycle $[A, B]$ with $A \subseteq I$ is *resolved* from I_0 by a token sliding sequence $I, I_1, \ldots, I_l$ if $G[I_l \cup B]$ induces no cycle. Such a cycle is called *resolvable*.

Throughout the proofs we describe what happens in the neighborhood of a cycle to be resolved. To this aim, we classify vertices depending on how many neighbors they have in a set B – define $N_i(B) = \{v \in V(G) \setminus B \mid |N(v) \cap B| = i\}$. The following lemma describes relations between those classes and will be very often referred to.

Lemma 3.8. *Let I be an independent set in a claw-free graph G and let $[A, B]$ be an induced cycle with $A \subseteq I$. Then the following properties hold:*

(a) $V(G) = B \cup N_0(B) \cup N_1(B) \cup N_2(B)$.

(b) There are no edges between vertices in $N_2(B)$ and $N_0(B)$.

(c) For any v, $v \in N(B) \setminus A$ if and only if $v \in N(A) \setminus B$.

(d) $N[B] \cap I = N_2(B) \cap I = A$

Proof. (a) Since B is an independent set and G is claw-free, no vertex can have more than three neighbors in B.

(b) Suppose to the contrary that $vw \in E(G)$ with $v \in N_2(B)$ and $w \in N_0(B)$. Let $N(v) \cap B = \{x, y\}$. Then G contains a v-claw with leaves v, x, y, a contradiction.

(c) Suppose $v \in N(B) \setminus A$ and $v \notin N(A) \setminus B$ (the symmetric case is analogous). Since $v \in N(B)$, we have $v \notin B$, so $v \notin N(A)$. Choose any vertex $x \in B \cap N(v)$. It has two neighbors $y, z \in A$, so G contains an x-claw with leaves v, y, z, a contradiction.

(d) Suppose there is a token $v \in I \setminus A$ in the closed neighborhood of B. By the previous claim, $v \in N(A)$, which contradicts that I is an independent set. □

We use this to describe reconfiguration sequences in more detail. The following lemma is essentially an induction step for the next two corollaries. The first says that any move from inside $N_2(B)$ to the outside is enough to resolve a cycle; the second shows that minimal resolving sequences do not change the neighboring relations to B until the very last move.

Lemma 3.9. *Let I be an independent set of a claw-free graph G and let $[A, B]$ be an induced cycle with $A \subseteq I$. Let $I' = I - u + v$ be an independent set obtained by sliding $u \to v$. Then exactly one of the following cases hold:*

- *$N(u) \cap B = N(v) \cap B$ and there is an induced cycle $[A', B]$ for some $A' \subseteq I'$.*
- *$u \in N_2(B)$, $v \in N_1(B)$ and the one-step sequence I, I' resolves $[A, B]$;*

Proof. There can be no tokens in B nor in $N_1(B)$ (Proposition 3.8(d)). Thus either $u \in N_0(B)$ or $u \in N_2(B)$ (Proposition 3.8(a)).

If $u \in N_0(B)$, then $u \notin A$, so $A \subseteq I'$ and $v \notin A$. Since there can be no tokens in the neighborhood of the cycle in I' (Proposition 3.8(d)), $v \in N_0(B)$. Clearly $N(v) \cap B = \emptyset = N(u) \cap B$, which proves that the first case of the claim holds.

If $u \in N_2(B)$, then $u \in N_2(B) \cap I = A$ (Proposition 3.8(d)). Let $c_1, c_2, \ldots, c_{2n}$ be the vertices of the cycle labeled in the cyclic order, so that $c_2, c_4, \cdots \in A$, $c_1, c_3, \cdots \in B$ and $u = c_2$. Then $N(u) \cap B = \{c_1, c_3\}$. Since v has no neighbors in I', $u = c_2$ is its only neighbor in I, and thus in A. Therefore $N(v) \cap B \subseteq N(u) \cap B$ – otherwise $c_{2i+1} \in N(v) \cap B$ for some $i \neq 0, 1$, which would imply a c_{2i+1}-claw with leaves c_{2i}, c_{2i+2}, v. Since there are no edges between $N_0(B)$ and $N_2(B)$ and $v \notin B$ (because v's only neighbor in A is u), v is either in $N_2(B)$ or $N_1(B)$. If $v \in N_2(B)$, then $N(v) \cap B = N(u) \cap B$; this proves the first case of the claim holds, as $G[I' \cup B]$ contains the cycle $[A - u + v, B] = c_1, v, c_3, c_4, \ldots, c_{2n}$. If $v \in N_1(B)$, then in $G[I' \cup B]$ the only connected components are isolated vertices $I \setminus A$ and a path $c_3, c_4, \ldots, c_{2n}, c_1$ with v attached to either end; hence the second case of the claim holds. □

Corollary 3.10. *Let I be an independent set of a claw-free graph G and let $[A, B]$ be an induced cycle with $A \subseteq I$. Let $I_0, \ldots, I_m$ be a token sliding sequence such that exactly one move, the last one, moves a token from $N_2(B)$ to $N_1(B)$. Then the sequence resolves $[A, B]$.*

Proof. The proof is by induction on m. Let $u \to v$ be the first move in the sequence. If $m = 1$, then $u \in N_2(B)$, $v \in N_1(B)$, so $[A, B]$ is resolved by the previous lemma. If $m > 1$, then $u \notin N_2(B)$ or $v \notin N_1(B)$, so by the previous proposition there is an I_1-bad cycle $[A', B]$. Then by inductive assumption the sequence $I_1, \ldots, I_m$ resolves it, that is, there is no cycle in $G[I_m \cup B]$, which proves our claim. □

Corollary 3.11. *Let I be an independent set of a claw-free graph G, and let $[A, B]$ be an induced cycle with $A \subseteq I$. Let $I_0, \ldots, I_m$ be a token sliding sequence that resolves $[A, B]$ such that no prefix $I_0, \ldots, I_i$ resolves $[A, B]$. Then the following properties hold:*

(a) For every $i \leqslant m - 1$, I_i contains only vertices in $N_0(B)$ and $N_2(B)$.

(b) For every $i \leqslant m - 2$, I_{i+1} is obtained from I_i by a move $u \to v$ such that $N(u) \cap B = N(v) \cap B$.

(c) I_m is obtained from I_{m-1} by a move $u \to v$ such that $u \in N_2(B)$ and $v \in N_1(B)$.

Proof. The proof is by induction on m. Let $u \to v$ be the first move in the sequence. If $m = 1$, then the move $u \to v$ resolves $[A, B]$, so by the previous proposition $u \in N_2(B)$ and $v \in N_1(B)$, which proves the claim. If $m > 1$, then $u \to v$ does not resolve C, so by the previous lemma $N(u) \cap B = N(v) \cap B$ and there is an I_1-bad cycle $[A', B]$. I_0 contains only vertices in $N_0(B)$ and $A \subseteq N_2(B)$ (Proposition 3.8(d)). The cycle $[A', B]$ is resolved by the sequence $I_1, \ldots, I_m$ and none of its prefixes, so the claim holds by induction. □

We conclude this section by showing that the very weak notion of resolving a cycle is enough to guarantee that we can 'rotate' a cycle, that is, replace A by B in a configuration without changing anything else. It then follows that an algorithm only needs to check if every cycle induced by an instance is resolvable – a reconfiguration sequence can then be obtained by 'rotating' cycles one by one.

Lemma 3.12. *Let I, J be independent sets in a claw-free graph G and let $[A, B]$ be a cycle in $G[I \Delta J]$. Then $[A, B]$ is resolvable from I if and only if $I \leftrightarrow_{\mathrm{TS}} I \setminus A \cup B$.*

Proof. If $I \leftrightarrow_{\mathrm{TS}} I \setminus A \cup B$ then clearly $[A, B]$ is resolvable from I, since $(I \setminus A \cup B) \cup B = I \setminus A \cup B$ induces no edges (and hence no cycles).

Suppose now that $[A, B]$ is resolvable from I, that is, there is a shortest token sliding sequence $I_0, \ldots, I_l$ from $I_0 = I$ to some I_l such that $I_l \cup B$ induces no cycles. Intuitively, we show that the tokens from the cycle's neighborhood can be moved to B and all moves outside of the cycle can be reversed. Define independent sets $I'_i = (I_i \setminus N(B)) \cup B$ for $i = 0, \ldots, l$. By Corollary 3.11(b) the tokens do not change their adjacency to B, so $|I_i \cap N(B)| = |I \cap N(B)| = |A| = |B|$. Thus I'_i are sets of the same size as I, and since I'_{i+1} differs by at most two vertices from I'_i, they form a token sliding sequence between I'_0 and I'_l.

Since $I_l \cup B$ induces no cycles, neither does $I_l \Delta I'_l = (I_l \cap N(B)) \cup B$. Therefore $I_l \leftrightarrow_{\mathrm{TS}} I'_l$ by Lemma 3.2. Since $I_0 \leftrightarrow_{\mathrm{TS}} I_l \leftrightarrow_{\mathrm{TS}} I'_l \leftrightarrow_{\mathrm{TS}} I'_0$, we demonstrated a sequence between $I_0 = I$ and $I'_0 = (I_0 \setminus N(B)) \cup B = I \setminus A \cup B$. □

Theorem 3.13. *Let I and J be independent sets in a connected claw-free graph G. Then $I \leftrightarrow_{\mathrm{TS}} J$ if and only if every cycle in $G[I \Delta J]$ is resolvable from I.*

Proof. Suppose $I \leftrightarrow_{\mathrm{TS}} J$. Then every cycle in $G[I \Delta J]$ is clearly resolved by the sequence going to J. The other direction is proved by induction on the number k of cycles in $G[I \Delta J]$. If $k = 0$, then by Lemma 3.2, $I \leftrightarrow_{\mathrm{TS}} J$. Otherwise consider a cycle C in $G[I \Delta J]$ and let $I' = I \Delta C$. Since C is resolvable from I, $I \leftrightarrow_{\mathrm{TS}} I'$ by Lemma 3.12. Now $G[I' \Delta J]$ has one cycle less than $G[I \Delta J]$, and every cycle in $G[I' \Delta J]$ is resolvable from I' (with a sequence going through I), so by inductive assumption, $I' \leftrightarrow_{\mathrm{TS}} J$. □

3.4. How to resolve a cycle

For our algorithm it remains to characterize resolvable cycles. Let $[A, B]$ be an induced cycle with $A \subseteq I$. A move $u \to v$ is called *internal* if $\{u, v\} \subseteq N_2(B)$ and *external* if $\{u, v\} \subseteq N_0(B)$. A resolving sequence $I_0, \ldots, I_m$ for C is called *internal* (*external*) if every move except the last is an internal (external) move. The cycle is called *internally* (*externally*) *resolvable* if such a sequence exist.

Lemma 3.14. *Let I be an independent set in a claw-free graph G and let $C = [A, B]$ be an induced cycle with $A \subseteq I$. Then any shortest token sliding sequence that resolves C is an internal or external resolving sequence.*

Proof. Consider a *shortest* token sliding sequence $S = I_0, I_1, \ldots, I_m$ that resolves C. Let $u \to v$ be the last move of this sequence. By Corollary 3.11(c), $u \in N_2(B)$ and $v \in N_1(B)$. By Proposition 3.8(a,c), $1 \leqslant |N(v) \cap A| \leqslant 2$. So one of the following cases applies to the neighborhood of v.

Case 1: $N(v) \cap I = \{x\}$ for some $x \in A$.
Then the move $x \to v$ yields an independent set again, and since $v \in N_1(B)$, it resolves C (Corollary 3.10). Thus shortest sequences resolving C have only one move and are therefore both internally and externally resolving.

Case 2: $N(v) \cap I = \{x, y\}$ for some $x \in A$ and $y \notin A$.
In this case, we omit all internal moves, to obtain an external sequence that resolves C. More precisely, for every i, define $I'_i = (I_i \setminus N(B)) \cup A$, and consider the sequence $I'_0, \ldots, I'_{m-1}$. For every i such that $I'_i \neq I'_{i+1}$, I'_{i+1} is obtained from I'_i by a move $u_i \to v_i$ where both u_i and v_i are in $N_0(B)$ (Corollary 3.11(a),(b)). Since there are no edges between $N_0(B)$ and $A \subseteq N_2(B)$ (Proposition 3.8(b)), every I'_i is an independent set, and thus this is a token sliding sequence. Since $I_{m-1} \cap N(v) = \{x\}$ and $A \cap N(v) = \{x\}$, it also holds that $I'_{m-1} \cap N(v) = \{x\}$, so from I'_{m-1}, the move $x \to v$ can be made, to resolve C (Corollary 3.10) from $I'_0 = I$. If $I'_i \neq I_i$ for some i, then this forms a shorter resolving sequence. Hence $(I_i) = (I'_i)$ is an externally resolving sequence.

Case 3: $N(v) \cap I = \{x, y\}$ for some $x, y \in A$.
In this case, we omit all external moves, to obtain an internal sequence that resolves C. More precisely, for every i, define $I'_i = (I \setminus N(B)) \cup (I_i \cap N(B))$, and consider the sequence $I'_0, \ldots, I'_{m-1}$. For every i such that $I'_i \neq I'_{i+1}$, I'_{i+1} is obtained from I'_i by a move $u_i \to v_i$ where both u_i and v_i are in $N_2(B)$ (Corollary 3.11(a, b)). Since there are no edges between $N_2(B)$ and $(I \setminus N(B)) \subseteq N_0(B)$ (Proposition 3.8(b)), every I'_i is an independent set, and thus this is a token sliding sequence. Since $I_{m-1} \cap N(v) = \{u\}$ and $N(v) \cap I \subseteq N(B)$, it also holds that $I'_{m-1} \cap N(v) = \{u\}$, so from I'_{m-1}, the move $u \to v$ can be made, to resolve C (Corollary 3.10) from $I'_0 = I$. If $I'_i \neq I_i$ for some i, then this forms a shorter resolving sequence. Hence $(I_i) = (I'_i)$ is an internally resolving sequence. □

3.5. External case

Resolving a cycle externally turns out to be equivalent to sliding tokens along a path alternating in I with one end in the cycle (and all the other vertices outside). We can hence check if a cycle is externally resolvable by searching, in the graph with the cycle deleted, for augmenting paths between every pair of vertices that has one element in the cycle's neighborhood. This can be done in polynomial time by Theorem 3.5.

Theorem 3.15. *Let I be a maximum independent set in a claw-free graph G and let $C = [A, B]$ be an induced cycle with $A \subseteq I$. Then C is externally resolvable if and only if there exists an $(I \setminus A)$-augmenting path in $G - A - B$ between some two vertices $x \in N_0(B)$ and $y \in N_1(B)$.*

Proof. Denote $G' = G - A - B$ and $I' = I \setminus A$, so I' is an independent set of G'.

Suppose that G' contains an I'-augmenting path $P = u_0, v_0, u_1, v_1, \ldots, v_{k-1}, u_k$ with $u_0 \in N_0(B)$ and $u_k \in N_1(B)$. Consider the neighborhood of each vertex u_i in I.

- The neighborhood of u_j in I' is exactly $\{v_{j-1}, v_j\}$ for $1 \leqslant j \leqslant k-1$. Since it cannot have more neighbors in I, $N(u_j) \cap I = \{v_{j-1}, v_j\}$.
- The neighborhood of u_0 in I' is exactly v_0. Since $u_0 \notin N(B)$ and $u_0 \notin B$, we observe that $u_0 \notin N(A)$ (Proposition 3.8(c)), so $N(u_0) \cap I = \{v_0\}$.
- The neighborhood of u_k in I' is exactly v_{k-1}. By Proposition 3.8(c), u_k has at least one neighbor in A, name it w. Since it cannot have more neighbors in I, $N(u_k) \cap I = \{v_{k-1}, w\}$.

It follows that from I we can apply the moves $v_0 \to u_0, v_1 \to u_1, \ldots, v_{k-1} \to u_{k-1}, w \to u_k$. Since $w \in A$ and $u_k \in N_1(B)$, this resolves C (Corollary 3.10), and thus C is externally resolvable.

For the other direction, suppose that $C = [A, B]$ is externally resolvable, and consider a shortest external resolving sequence $I_0, \ldots, I_m$ (with $I_0 = I$) that is, all moves but the last one are between vertices in $N_0(B)$, and the last move of the sequence is $u \to v$ for some $u \in N_2(B)$ and $v \in N_1(B)$ (Corollary 3.11). Consider the component of $G[I \Delta I_m]$ containing u and v. It is either a path or a cycle. Since $u \in N_2(B)$ cannot have neighbors in $N_0(B)$ (Proposition 3.8(b)), it is the end of an induced path P that alternates between I and I_m. If P had an odd number of vertices, then it would be an augmenting path for I_m, contradicting that I and I_m are maximum. Thus P has an even number of vertices and after deleting u it is an I'-augmenting path in G' between $v \in N_1(B)$ and some vertex in $N_0(B)$. □

3.6. Internal case

Shortest *internal* resolving sequences cannot be as easy to describe as external ones, since a token can move several times (see Figure 3.16, and 3.20 for a more general example). Nevertheless, these sequences can be shown to have a very specific structure, which can be characterized using paths in the auxiliary digraphs described below. The proof of the characterization is split into two lemmas, one showing the converse of the other.

Consider an induced cycle $C = [A, B]$ in G with $A \subseteq I$. Let $C = c_0, c_1, \ldots, c_{2n-1}, c_0$ with $c_0, c_2, \cdots \in A$ and $c_1, c_3, \cdots \in B$. Define the *layer* L_i as $\{v \in V(G) \mid N(v) \cap B = N(c_{2i}) \cap B\}$; indices of c_i and L_i are always meant modulo $2n$ and n, respectively. Note that by Corollary 3.11(b), when starting with I and using only internal moves, the token that starts on c_{2i} will stay in the layer L_i.

Since for $n = 2$ the layers are not disjoint, the following is used only for $n > 2$. Define $D(G, C)$ to be a digraph with vertex set $V(G)$, with the following arc set:

- (u, v) for all i and all pairs $u \in L_i, v \in L_{i+1}$ such that $uv \notin E(G)$,
- (b, v) for all i, all vertices $b \in N_1(B)$ with $N(b) \cap B = \{c_{2i-1}\}$ and all $v \in L_i$ such that $bv \notin E(G)$.

We denote the reversed cycle by $C^{rev} = c_0, c_{2n-1}, \ldots, c_1, c_0$, yielding a digraph $D(G, C^{rev})$, that compared to $D(G, C)$ has arcs between layers reversed, and arcs from $N_1(B)$ going to different layers.

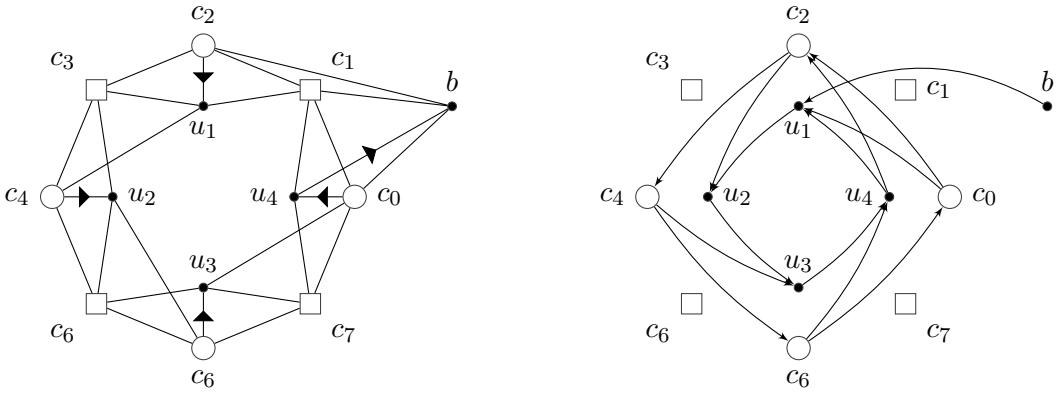


Figure 3.16: An example of a claw-free graph G with an internally resolvable cycle, along with the corresponding auxiliary digraph $D(G, C)$.

Lemma 3.17. *Let I be an independent set in a claw-free graph G, and let $C = [A, B]$ be an induced cycle in with $A \subseteq I$ and $|A| > 2$. If C is internally resolvable from I then in $D(G, C)$ or $D(G, C^{rev})$ there is a directed path from some vertex $b \in N_1(B)$ that has no neighbors in $I \setminus A$ to some vertex in A.*

Proof. Let $C = c_0, c_1, \ldots, c_{2n-1}, c_0$ with $c_i \in A$ for even i. Let $I_0, \ldots, I_m$ be a shortest internal resolving token sliding sequence, $I_0 = I$. By definition of *internally resolvable* and Corollary 3.11, the last move is is from $N_2(B)$ to some $b \in N_1(B)$, and all other moves $u \to v$ satisfy $u, v \in N_2(B)$ and $N(u) \cap B = N(v) \cap B$. We shall prove that all of A is reachable from b by directed paths in $D(G, C)$ or $D(G, C^{rev})$.

Since $n > 2$, I_0 contains exactly one vertex of L_i (namely c_{2i}) for every $i \in \{0, \ldots, n-1\}$ and there are no other vertices in $I_0 \cap N_2(B)$ (Proposition 3.8(d)). Since $N(u) \cap B = N(v) \cap B$ holds for every move $u \to v$ except the last one, we can define v^i_j to be the only vertex in $I_j \cap L_i$ for $j = 0 \ldots m-1$.

W.l.o.g. assume that c_1 is the only neighbor of b in B. Since a token is moved to b in the last move, b has at most one neighbor in I_{m-1}, so it cannot be adjacent to both v^0_{m-1} and v^1_{m-1}. Assume w.l.o.g. it is non-adjacent to v^1_{m-1} (otherwise reverse the order of vertices $c_0, \ldots, c_{2n-1}$, replacing $D(G, C)$ with $D(G, C^{rev})$). Then by definition, $D(G, C)$ contains an arc from b to v^1_{m-1}.

We shall prove inductively that for every $j \leqslant m-1$, all vertices in the set $\{v^0_j, \ldots, v^{n-1}_j\}$ are reachable from b in $D(G, C)$. For every $j = 0, \ldots, m-1$ and every i, $D(G, C)$ contains an arc from v^i_j to v^{i+1}_j, because these vertices are both in I_j and are therefore nonadjacent in G. Therefore if for some $j = 0 \ldots m-1$ one vertex in $\{v^0_j, \ldots, v^{n-1}_j\}$ is reachable from b in $D(G, C)$, then all those vertices are. But $\{v^0_j, \ldots, v^{n-1}_j\}$ and $\{v^0_{j-1}, \ldots, v^{n-1}_{j-1}\}$ share $n-1$ vertices, so by induction every vertex of $\{v^0_0, \ldots, v^{n-1}_0\} = A$ is reachable from b in $D(G, C)$ by a directed path.

It remains to show that b has no neighbors in $I \setminus A$. Suppose that some vertex $x \in I \setminus A$ is adjacent to b. Since $x \notin N_2(B)$ (Proposition 3.8(d)), the token on x is never moved in the sequence and $x \in I_m$. But b is a neighbor of x also contained in the independent set I_m, a contradiction. □

Lemma 3.18. *Let I be an independent set in a claw-free graph G, and $C = [A, B]$ be an induced cycle with $A \subseteq I$ and $|A| > 3$. If $D(G, C)$ or $D(G, C^{rev})$ contains a directed path from a vertex $b \in N_1(B)$ $b \in N_1(B)$ that has no neighbors in $I \setminus A$ to some vertex in A, then C is internally resolvable.*

Proof. Let $C = c_0, c_1, \ldots, c_{2n-1}, c_0$ with $c_i \in A$ for even i. W.l.o.g. assume that $D(G, C)$ contains such a path, and that $N(b) \cap B = \{c_1\}$. Let $P = u_0, \ldots, u_m$ be a shortest path in $D(G, C)$ from $u_0 = b$ to a vertex $u_m \in A$.

By definition of $D(G, C)$, any arc (u_0, x) must have $x \in L_1$. Thus $u_1 \in L_1$ and similarly, $u_j \in L_j$ follows inductively for all $j = 1, \ldots, m$. The vertex u_m is the first vertex of P in A, with $u_m \in L_m$ so $u_m = c_{2m}$. For all j, $D(G, C)$ contains an arc from c_{2j} to $c_{2(j+1)}$, so we can extend P to a directed path $P' = u_0, \ldots, u_{m+n-1}$ by defining $u_{m+j} = c_{2(m+j)}$ for $j = 1 \ldots n-1$. The following properties hold for P', and will be used often in the remainder of the proof:

$$N(u_0) \cap B = \{c_1\},$$
$$N(u_j) \cap B = \{c_{2j-1}, c_{2j+1}\} \text{ for } j = 1, \ldots, m+n-1.$$

The idea is to reconfigure A to subsequent infixes of P' and the remainder of the proof shows that this gives a valid internally resolving sequence.

For $j = 0, \ldots, m$, define

$$I_j = \{u_{m-j}, u_{m-j+1}, \ldots, u_{m-j+n-1}\} \cup (I \setminus A),$$

and consider the sequence $I_0, \ldots, I_m$. This sequence starts with $I_0 = I$, because $A = \{u_m, \ldots, u_{m+n-1}\}$. Since $I_j = I_{j-1} + u_{m-j} - u_{m-j+n}$ for $j = 1, \ldots, m$, the consecutive steps correspond to jumping a token from u_{m-j+n} to u_{m-j}. We will now show that for every j, I_j is an independent set, and that $u_{m-j+n}u_{m-j} \in E(G)$, proving that $I_0, \ldots, I_m$ is a token sliding sequence. Considering the last move, it then follows that this sequence resolves C (Corollary 3.10), proving the lemma. Therefore it suffices to show that:

- $u_j u_{j'} \notin E(G)$, for all $j, j' \in \{0, \ldots, m+n-1\}$ with $1 \leqslant j' - j \leqslant n-1$.
- $u_{m-j}u_{m-j+n} \in E(G)$ for all $j \in \{1, \ldots, m\}$, and

Indeed, the first condition ensures that each I_j is an independent set: $u_0 = b$ has no neighbors in $I \setminus A$ by assumption, and the vertices u_j for $j \geqslant 1$ have no neighbors in $I \setminus A \subseteq N_0(B)$ because there are no edges between $N_0(B)$ and $N_2(B)$ (Proposition 3.8(b)).

We shall prove the above statements with a few claims marked with Greek letters for later reference.

Claim α: $u_j u_{j+1} \notin E(G)$ for all j.
This claim follows directly from the fact that there is an arc $u_j u_{j+1}$ in $D(G, C)$.

Claim β: $u_j u_{j'} \notin E(G)$ for $2 \leqslant j' - j \leqslant n-2$.
If $j > 0$ then u_j and $u_{j'}$ belong to L_j and $L_{j'}$, respectively. So their neighbors in B are exactly c_{2j-1}, c_{2j+1} and $c_{2j'-1}, c_{2j'+1}$, respectively. By choice of j and j', these are four different vertices. Thus if $u_j u_{j'} \in E(G)$, then there would be a $u_{j'}$-claw with leaves $u_j, c_{2j'-1}, c_{2j'+1}$, a contradiction. If $j = 0$, then the proof is analogous, except that u_j has exactly one neighbor in B, namely $c_{2j+1} = c_1$.

Together, claims α and β prove the statement above for all cases except $j' - j = n-1$. It thus remains to show that:

Claim γ: $u_j u_{j+n} \in E(G)$ for all $j \in \{0, \dots, m-1\}$ and
Claim δ: $u_j u_{j+n-1} \notin E(G)$ for all $j \in \{0, \dots, m\}$.
We prove these claims by induction on j.

$\delta(0)$: $u_0 u_{n-1} \notin E(G)$, for otherwise there would be a u_{n-1}-claw with leaves u_0, c_{2n-3}, c_{2n-1} (recall that $N(u_0) \cap B = c_1$, and that $n \geqslant 3$).

$\gamma(0)$: We wish to prove that $u_0 u_n \in E(G)$. We first observe that $N(u_0) \cap A \subseteq \{c_0, c_2\}$. Indeed, if u_0 would be adjacent to another vertex $c_{2i} \in A$, then there is a c_{2i}-claw with leaves u_0, c_{2i-1}, c_{2i+1}. The vertex u_0 is adjacent to at least one of c_0 and c_2; otherwise there would be a c_1-claw with leaves u_0, c_0, c_2. If u_0 is adjacent to exactly one of them, then u_0 has no other neighbors in I and C can trivially be (internally) resolved in one move. So we may now assume that $N(u_0) \cap A = \{c_0, c_2\}$.

If $m \leqslant n$ then $u_n \in A$ so $u_n = c_0$, which shows that $u_0 u_n \in E(G)$.

Now suppose $m = n+1$, so $u_{n+1} = c_2$. Claim α shows that $u_n c_2 = u_n u_{n+1} \notin E(G)$. Since $u_n \in L_0$, it holds that $u_n c_1 \in E(G)$. We conclude that $u_n c_0 \in E(G)$, for otherwise there would be a c_1-claw with leaves c_0, c_2, u_n. Next, we note that $u_{n-1} c_0 \in E(G)$, because otherwise there would be a shorter path $u_0, \dots, u_{n-1}, c_0$ in $D(G, C)$. Furthermore, $u_0 u_{n-1} \notin E(G)$ holds by $\delta(0)$, and $u_{n-1} u_n \notin E(G)$ by α. Combining these facts, we conclude that $u_0 u_n \in E(G)$, because otherwise there would be a c_0-claw with leaves u_0, u_n, u_{n-1} in G.

In the remaining case, $m \geqslant n+2$ holds. Then $u_n c_2 \in E(G)$, for otherwise there would be a shorter path $u_0, \dots, u_n, c_2$ in $D(G, C)$. In this case $u_0 u_n \in E(G)$ follows since otherwise, there would be a c_2-claw with leaves u_0, u_n, c_3. This concludes the proof of Claim γ for the case $j = 0$.

$\delta(1)$: By $\gamma(0)$, it holds that $u_0 u_n \in E(G)$. Recall that $u_0 u_1 \notin E(G)$, $c_{2n-1} u_1 \notin E(G)$ and $c_{2n-1} u_0 \notin E(G)$. So $u_1 u_n \notin E(G)$, because otherwise there would be a u_n-claw with leaves c_{-1}, u_0, u_1.

$\delta(j) \implies \gamma(j)$ for $j = 1, \dots, m-1$:
By $\delta(j)$, $u_j u_{j+n-1} \notin E(G)$ holds, and by α, $u_{j+n} u_{j+n-1} \notin E(G)$ holds. So $u_j u_{j+n} \in E(G)$, for otherwise there would be a c_{2j-1}-claw with leaves u_j, u_{j+n}, u_{j+n-1}.

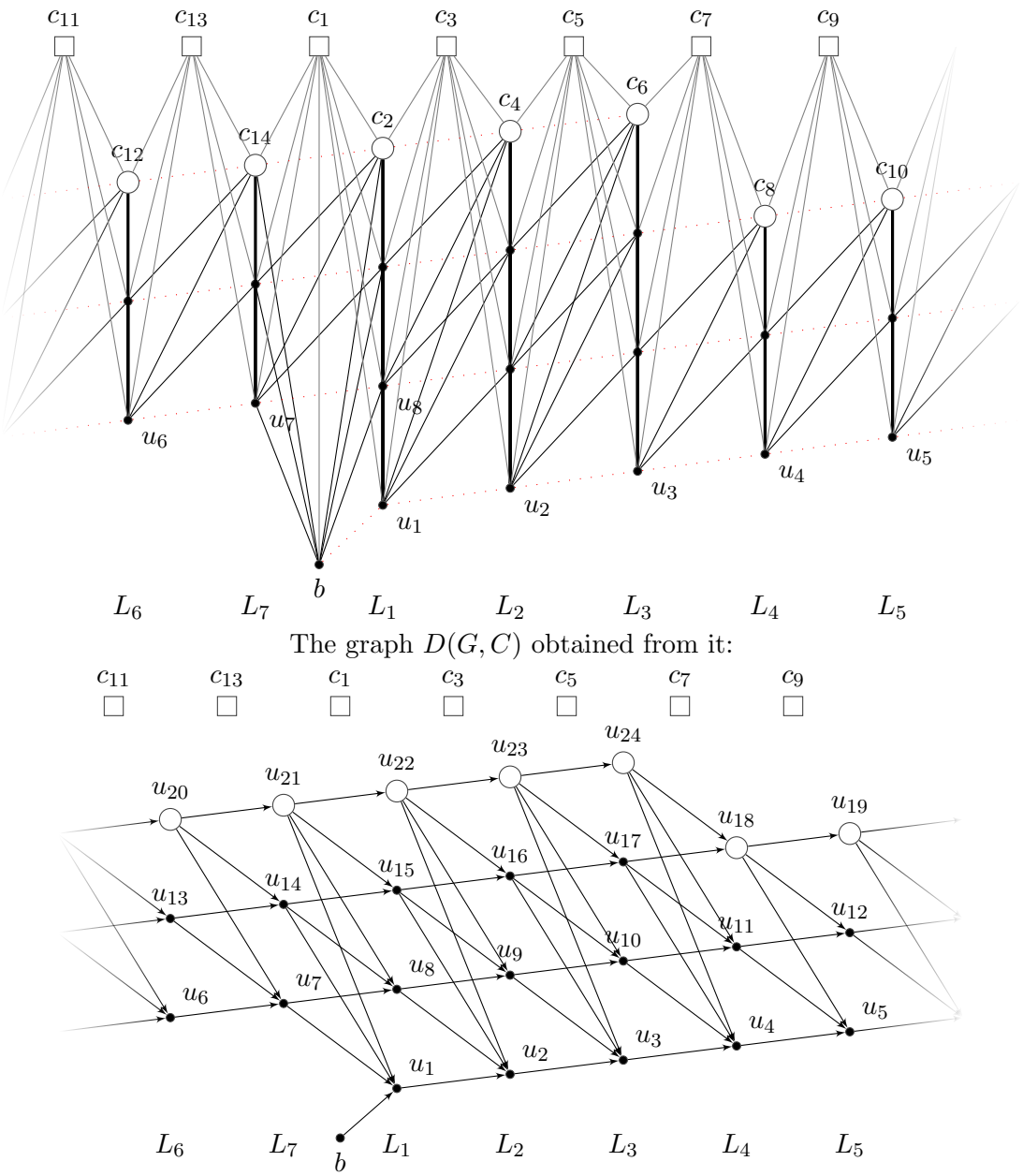
$\gamma(j) \implies \delta(j+1)$ for $j = 1, \dots, m-1$:
We observe that $u_{j+n} u_{j-1} \in E(G)$, for otherwise $u_0, \dots, u_{j-1}, u_{j+n}, \dots, u_m$ would be a shorter path in $D(G, C)$. Next, $u_{j+n} u_j \in E(G)$ by $\gamma(j)$, $u_{j-1} u_j \notin E(G)$ and $u_j u_{j+1} \notin E(G)$ by α, and $u_{j-1} u_{j+1} \notin E(G)$ by β, using that $n \geqslant 4$. We conclude that $u_{j+1} u_{j+n} \notin E(G)$, for otherwise there would be a u_{j+n}-claw with leaves u_{j-1}, u_j, u_{j+1}.

This concludes the induction proof of claims γ and δ. □

Corollary 3.19. *Let I be an independent set in a claw-free graph G and let $[A, B]$ be an induced cycle with $A \subseteq I$. In polynomial time it can be decided whether $[A, B]$ is internally resolvable.*

Proof. If $|A| > 3$, then by Lemmas 3.17, 3.18 it suffices to make a depth-first-search in $D(G, C)$ and $D(G, C^{rev})$ to find an appropriate directed path. If $|A| = 2$ or $|A| = 3$, we can enumerate all pairs or triples of vertices, find all independent sets I' such that $I' \setminus A' = I \setminus A$ for some A' of size $|A|$, in particular all configurations reachable by making internal moves only, and check if C is internally resolvable by brute force. □

Figure 3.20: An example of a graph G with an induced cycle $[A, B]$, $|A| = 7$, which is internally resolvable in $m = 18$ steps. Circles denote I, squares denote B. Faded edges at the boundary of the figure continue on the other side. Vertices in each column $L_1, \ldots, L_7$ form a clique. The directed path from the vertex b to I in $D(G, C)$ is also shown as a red dotted line to clarify the structure of G.



We remark that internal resolving sequences can be succinctly described by the following example. For integers $n, k \geqslant 2$ consider the graph C^{k-1}_{nk+1}, which has as vertex set $\{z_0, z_1, \ldots, z_{nk}\}$ and edges $z_i z_j$ whenever $|i-j| < k \pmod{nk+1}$ (that is, z_i is adjacent to $z_{i-(k-1)}, \ldots, z_{i-1}, z_{i+1}, \ldots, z_{i+k-1}$ where indices are meant mod $nk+1$). C^{k-1}_{nk+1} (named so because it is the $k-1$-th power of a cycle on $kn+1$ vertices) is clearly a unit circular arc graph, and thus claw-free (see [Lin+11] for more on the relation between powers of cycles and proper circular arc graphs). Maximum independent sets are exactly the sets $I_b = \{z_{k\cdot i} \mid i = b+1, \ldots, b+n\}$ for integers b, from which the only allowed token slides are a slide from $z_{k\cdot(b+n)}$ to $z_{k\cdot(b+n)+1} = z_{k\cdot b+(kn+1)} = z_{k\cdot b}$, which corresponds to decreasing b, and a slide from $z_{k\cdot(b+1)}$ to $z_{k\cdot(b+n)-1} = z_{k\cdot(b+n+1)}$ that corresponds to increasing b. A shortest token sliding sequence from I_b to $I_{b'}$ is then equivalent to repeatedly decreasing or increasing b until b' is reached.

It turns out that this is essentially the only example of internally resolving sequences. Define the sequence of vertices u_i as in the proof of Lemma 3.18, let $u_{-i} = c_{-2i+1}$ for $i = 1, \ldots, n$ and define $k = \lceil \frac{m}{n} \rceil + 2$ (or any larger integer). The set $U = \{u_i | i = -n, \ldots, n+m-1\}$ then includes all vertices of the cycle and all vertices used in the resolving sequence. It can be proved that the mapping $u_i \mapsto z_{k\cdot i}$ defines an isomorphism between $G[U]$ and an induced subgraph of C^{k-1}_{nk+1}. In particular, $G[U]$ is a unit circular arc graph and is uniquely determined by two integers n and m. Unfortunately, the only proof of these facts we are aware of is an extension of the proof of Lemma 3.18 in which we tediously prove whether $u_i u_j$ is an edge of G for different pairs i, j. Knowing that proper circular arc graphs form a basic building block of claw-free graphs in the decompositions of [CS05] and [FOS11], we sought proofs that would better explain the uniqueness of $G[U]$, without success.

3.7. Summary of the algorithm

Theorem 3.21. *Let I and J be independent sets in a claw-free graph G. In polynomial time it can be decided whether $I \leftrightarrow_{\mathrm{TS}} J$ and whether $I \leftrightarrow_{\mathrm{TJ}} J$.*

Proof. If $|I| \neq |J|$ then clearly I cannot be reconfigured into J, so assume $|I| = |J|$.

If I and J are not maximum independent set, which can be checked in polynomial time, then $I \leftrightarrow_{\mathrm{TJ}} J$ by Lemma 3.6, so answer YES. If they are maximum, then token jumps are the same as token slides. Thus it suffices to ask whether $I \leftrightarrow_{\mathrm{TS}} J$.

If G is disconnected, then $I \leftrightarrow_{\mathrm{TS}} J$ if and only if this is true inside every connected component of G, since tokens can never slide from one to another. Assume thus that G is connected.

If I and J are not maximum, then $I \leftrightarrow_{\mathrm{TS}} J$ by Corollary 3.7. Otherwise, Theorem 3.13 shows that $I \leftrightarrow_{\mathrm{TS}} J$ if and only if every cycle in $G[I \Delta J]$ is resolvable from I. By Lemma 3.14, it suffices to check for internal and external resolvability of such cycles. Since I is assumed to be a maximum independent set of G, this can be decided in polynomial time by Theorem 3.15 and Corollary 3.19. □

3.8. Shortest Reachability is hard

In this section we show our algorithm cannot be strengthened to give shortest reconfiguration sequences, even in very restricted instances. The general idea is that between an initial token position and a destination, there can be a choice of different paths, each with a different

blocking token that will have to be moved. We construct instances where each initial position in $I \setminus J$ has one intended destination in $J \setminus I$, to which a number of paths go, each blocked by a single token in $I \cap J$. By making such tokens block arbitrary sets of paths, we show that the problem of choosing a minimum family of blocking tokens to be moved (so that each initial position-destination pair has some path unblocked) is as hard as Set Cover where we choose a minimum family of sets so that each element of some universe is covered. Some care has to be taken to make all deviations from the intended behavior nonoptimal.

The reductions show hardness even for line graphs (recall that this is a subclass of claw-free graphs), so they are best described via matchings (independent sets of edges). Matching Sliding, Matching Jumping and the common special case of Maximum Matching reconfiguration are defined analogously as independent set reconfiguration – the solution space consists of matchings of a graph and a transformation step corresponds to sliding or jumping a token in the line graph (sliding an edge of a matching thus means replacing one edge by another with a common vertex).

We show two similar reductions. One yields maximum matchings and thus shows NP-hardness for all the models considered. The other yields pairs of matching that induce no cycles. Note that requiring the matchings (or independent sets in the line graph) to be both maximum and cycle-free makes shortest reconfiguration trivial, because then $I \Delta J$ induces only even alternating paths.

Set Cover is the problem of, given a set of elements $\mathcal{U}$, a family of subsets $\mathcal{F} \subseteq 2^{\mathcal{U}}$ and an integer l, deciding whether there exists a subfamily $\mathcal{F}' \subseteq \mathcal{F}$ of size at most l covering $\mathcal{U}$ (that is, $\mathcal{U} \subseteq \bigcup_{F \in \mathcal{F}'} F$). Given a graph $G = (V, E)$ and an integer l, Vertex Cover is the problem of deciding whether there is a set of at most l vertices such that every edge is incident to one of them. Any instance $(G = (V, E), l)$ of Vertex Cover is equivalent to the instance $(\mathcal{U}, \mathcal{F}, l)$ of Set Cover with $\mathcal{U} = E, \mathcal{F} = \{\{e : e \text{ is incident to } v\} : v \in V\}$. We use this together with the following fact to further restrict the structure of graphs obtained in our reductions. Recall that a cubic graph is a graph whose vertices all have degree 3.

Fact 3.22 ([Moh01])**.** Vertex Cover *is NP-complete even for cubic planar graphs.*

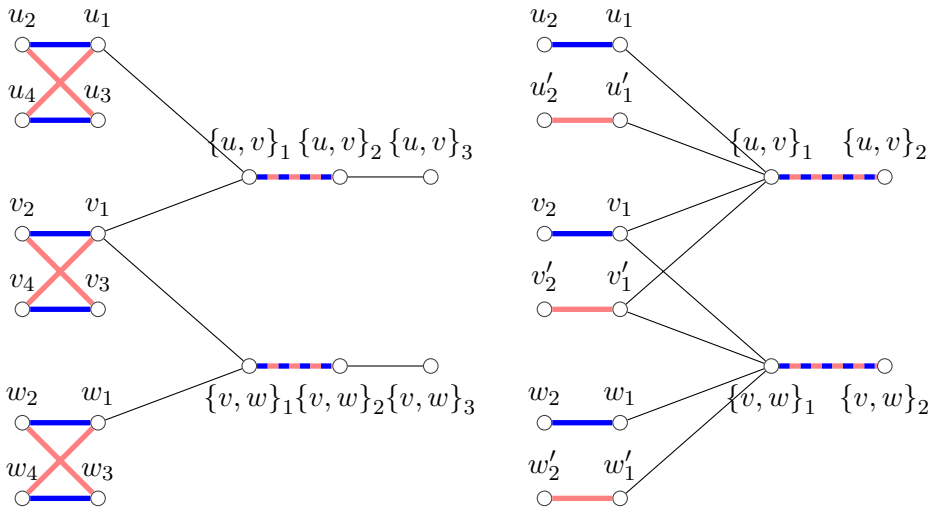


Figure 3.23: Example of graphs and matchings obtained from the reductions of Propositions 3.24, 3.25 from an instance of Set Cover with $\mathcal{U} = \{u, v, w\}$, $\mathcal{F} = \{\{u, v\}, \{v, w\}\}$.

Proposition 3.24. Maximum Matching Shortest Reachability *is NP-hard even on bipartite planar graphs of degree bounded by* 4*, even when the two matchings are maximum.*

Proof. Let $(\mathcal{U}, \mathcal{F}, l)$ be an instance of Set Cover. Define

- $V = \{u_1, u_2, u_3, u_4 : u \in \mathcal{U}\} \cup \{F_1, F_2, F_3 : F \in \mathcal{F}\}$,
- $E = \{u_1u_2, u_2u_3, u_3u_4, u_4u_1 : u \in \mathcal{U}\} \cup \{F_1F_2, F_2F_3 : F \in \mathcal{F}\} \cup \{u_1F_1 : u \in F, u \in \mathcal{U}, F \in \mathcal{F}\}$,
- $I = \{u_1u_2, u_3u_4 : u \in \mathcal{U}\} \cup \{F_1F_2 : F \in \mathcal{F}\}$,
- $J = \{u_2u_3, u_4u_1 : u \in \mathcal{U}\} \cup \{F_1F_2 : F \in \mathcal{F}\}$.

See Figure 3.23 for an example. The matchings I, J are maximum, because every edge in $G = (V, E)$ is incident to one of the vertices u_1, u_3 or F_2 for $u \in \mathcal{U}, F \in \mathcal{F}$ (so every matching has at most $2|\mathcal{U}| + |\mathcal{F}|$ elements) and each of these vertices is matched in I and J. We show that there is an reconfiguration sequence of length $\leqslant 3|\mathcal{U}| + 2l$ from I to J if and only if there is a subfamily $\mathcal{F}' \subseteq \mathcal{F}$ of size $\leqslant l$ covering $\mathcal{U}$.

For one side, suppose $\mathcal{F}'$ is a subfamily of $\mathcal{F}$ covering $\mathcal{U}$ of size at most l. For every $u \in \mathcal{U}$, choose $F(u)$ to be some set in $\mathcal{F}'$ containing u. We can slide edges of the matching I in the following way: first, for each $F \in \mathcal{F}'$, slide F_1F_2 to F_2F_3. Then, for all $u \in \mathcal{U}$, slide u_1u_2 to $u_1F(u)_1$, u_3u_4 to u_2u_3 and then $u_1F(u)_1$ to u_4u_1. Finally, for each $F \in \mathcal{F}'$ slide F_2F_3 back to F_1F_2. This gives a reconfiguration sequence of total length $|\mathcal{F}'| + 3|U| + |\mathcal{F}'| \leqslant 3|U| + 2l$ from I to J.

For the other side, suppose there is a sequence of length $l' \leqslant 3|U| + 2l$ from I to J. Since every step gives a maximum independent set, vertices u_1, u_3, F_2 for $u \in \mathcal{U}, F \in \mathcal{F}$ are always matched. For every $u \in \mathcal{U}$, the edge u_1u_2 must have been moved, and its first slide must have been to $u_1F(u)_1$ for some $F(u) \in \mathcal{F}$. Thus the edge $F(u)_1F(u)_2$ must have been moved, and to reach any destination in J it must have been moved at least twice. For every $u \in \mathcal{U}$ the edge u_3u_4 of I must have moved at least one step and u_1u_2 at least two steps (since $u_1F(u)_1$ is not in J). Let $\mathcal{F}' = \{F(u) : u \in \mathcal{U}\}$. Then the total number of steps l' is at least $2|\mathcal{F}'| + |\mathcal{U}| + 2|\mathcal{U}|$, thus $|\mathcal{F}'| \leqslant \frac{l'-3|\mathcal{U}|}{2} \leqslant l$. The edges $u_1F(u)_1$ only exist if $u \in F(u)$, so $\mathcal{F}'$ is a set cover.

The instances described above are therefore equivalent. To obtain our claim it suffices to apply this reduction to instances $(G' = (V', E'), l)$ of Vertex Cover from Fact 3.22 by setting $\mathcal{U} = E', \mathcal{F} = \{\{e : e \text{ is incident to } v\} : v \in V'\}$. □

Proposition 3.25. Matching Sliding Shortest Reachability *is NP-hard even on bipartite planar graphs of degree bounded by* 7*, even when the two matchings induce no cycle.*

Proof. Let $(\mathcal{U}, \mathcal{F}, l)$ be an instance of Set Cover. Define

- $V = \{u_1, u_2, u_1', u_2' : u \in \mathcal{U}\} \cup \{F_1, F_2 : F \in \mathcal{F}\}$,
- $E = \{u_1u_2, u_1'u_2' : u \in \mathcal{U}\} \cup \{F_1F_2 : F \in \mathcal{F}\} \cup \{u_1F_1, u_1'F_1 : u \in F \in \mathcal{F}\}$,
- $I = \{u_1u_2 : u \in \mathcal{U}\} \cup \{F_1F_2 : F \in \mathcal{F}\}$,
- $J = \{u_1'u_2' : u \in \mathcal{U}\} \cup \{F_1F_2 : F \in \mathcal{F}\}$.

See Figure 3.23 for an example. We show that there is an reconfiguration sequence (in the sliding model) of length $\leqslant 3|\mathcal{U}| + l$ from I to J if and only if there is a subfamily $\mathcal{F}' \subseteq \mathcal{F}$ of size $\leqslant l$ covering $\mathcal{U}$.

For one side, suppose $\mathcal{F}'$ is a subfamily of $\mathcal{F}$ covering $\mathcal{U}$ of size at most l. For every $u \in \mathcal{U}$, choose $F(u)$ to be some set in $\mathcal{F}'$ containing u. For every $F \in \mathcal{F}'$, choose $g(F) \in \mathcal{U}$ to be such that $F(g(F)) = F$ (if there is no such $g(F)$, take $\mathcal{F}' \setminus \{F\}$ as a smaller subfamily covering $\mathcal{U}$). We can slide the edges of I in the following way:

- For each $F \in \mathcal{F}'$, slide F_1F_2 through $g(F)_1F_1$ to $g(F)'_1g(F)'_2$.
- For all $u \in \mathcal{U}$ such that $u \neq g(F(u))$, slide $u_1u_2 \to u_1F(u)_1 \to u'_1F(u)_1 \to u'_1u'_2$.
- For each $F \in \mathcal{F}'$ slide $g(F)_1g(F)_2$ through $g(F)_1F_1$ to F_1F_2.

This gives a reconfiguration sequence of length $2|\mathcal{F}'| + 3(|U| - |\mathcal{F}'|) + 2|\mathcal{F}'| \leqslant 3|U| + l$ from I to J.

For the other side, suppose there is a sequence of length $l' \leqslant 3|U| + l$ from I to J. Let $E_{\mathcal{F}} = E'_{\mathcal{F}} = \{F_1F_2 : F \in \mathcal{F}\}$, $E_{\mathcal{U}} = \{u_1u_2 : u \in \mathcal{U}\}$ and $E'_{\mathcal{U}} = \{u'_1u'_2 : u \in \mathcal{U}\}$. Every edge that ever moved must have moved at least two steps. Every edge that moved from $E_{\mathcal{U}}$ to end at $E'_{\mathcal{U}}$ must have moved at least 3 steps. Let $n_{A,B}$ be the number of edges that moved from E_A to E'_B, for $A, B \in \{\mathcal{F}, \mathcal{U}\}$. The total number of steps l' is then at least $2n_{\mathcal{F},\mathcal{F}} + 2n_{\mathcal{F},\mathcal{U}} + 2n_{\mathcal{U},\mathcal{F}} + 3n_{\mathcal{U},\mathcal{U}}$. Since the number of tokens that left $E_{\mathcal{F}}$ equals the number of tokens that moved onto it, $n_{\mathcal{F},\mathcal{U}} = n_{\mathcal{U},\mathcal{F}}$. Since all tokens from $E_{\mathcal{U}}$ moved, $n_{\mathcal{U},\mathcal{F}} + n_{\mathcal{U},\mathcal{U}} = |\mathcal{U}|$. Thus $3|U|+l \geqslant l' \geqslant 2n_{\mathcal{F},\mathcal{F}}+2n_{\mathcal{F},\mathcal{U}}+2n_{\mathcal{U},\mathcal{F}}+3n_{\mathcal{U},\mathcal{U}} = n_{\mathcal{F},\mathcal{F}}+2n_{\mathcal{F},\mathcal{U}}+3|\mathcal{U}| \geqslant n_{\mathcal{F},\mathcal{F}}+n_{\mathcal{F},\mathcal{U}}+3|\mathcal{U}|$.

Let $\mathcal{F}'$ be the subfamily of those sets $F \in \mathcal{F}$ for which the edge F_1F_2 ever moved. By the above, $|\mathcal{F}'| = n_{\mathcal{F},\mathcal{F}} + n_{\mathcal{F},\mathcal{U}} \leqslant l$. It remains to show that $\mathcal{F}'$ covers $\mathcal{U}$. Suppose otherwise, that some $u \in \mathcal{U}$ is not contained in any $F \in \mathcal{F}'$. Then for all $F \in \mathcal{F}$ containing u the edge F_1F_2 never moves, and therefore no step of the sequence can slide an edge from u_1u_2 onto u_1F_1. Since these are exactly all edges incident with u_1, the edge u_1u_2 can never move, a contradiction.

To obtain our claim it suffices to apply this reduction to instances $(G' = (V', E'), l)$ of Vertex Cover from Fact 3.22 by setting $\mathcal{U} = E', \mathcal{F} = \{\{e : e \text{ is incident to } v\} : v \in V'\}$. □

Corollary 3.26. Maximum Independent Set Shortest Reachability*, and thus* TS, TJ *and* TAR Shortest Reachability*, are NP-hard even on line graphs of bipartite planar graphs of bounded degree.*

Corollary 3.27. Token Sliding Shortest Reachability *is NP-hard even on line graphs of bipartite planar graphs of bounded degree, even when the two independent sets induce no cycle.*

Chapter 4

Homomorphism reconfiguration in general graphs

In this chapter we consider the complexity of H-Coloring Reachability and related problems, considering restrictions to the structure of H in homomorphisms from G to H, as opposed to restricting the structure of G. We begin with several observations about results implied by existing work. The focus is then on the *monochromatic neighborhood case*, after which a new proof of 3-Coloring Shortest Reachability being in P is given, with comments about possible generalizations.

For clarity all graphs in this chapter are graphs with loops allowed, we always consider H-colorings of a graph G, and refer to $V(G)$ as *vertices* and $V(H)$ as *colors*. A reconfiguration sequence in the solution graph of H-colorings is called an H-recoloring sequence for short, and if two H-colorings μ, μ' of G are in the same component of the solution graph (that is, one can be reconfigured into another by changing one color at a time), we say that μ can be H-recolored to μ'.

4.1. Preliminaries

For the problem of deciding the existence of H-colorings, the observation that a composition of homomorphisms is a homomorphism is used to reduce problems via *retracts* (homomorphisms to proper subgraphs), allowing to consider only the case when H is a *core*, i.e., a graph that cannot be further retracted. These notions aren't useful for reconfiguration as several colors would have to be changed in one step to give a reduction. A special case more relevant to reconfiguration is given by the following definitions.

Definition 4.1. *A* fold *of a graph H is the graph $H - a$ obtained by deleting a vertex a such that $N_H^+(a) \subseteq N_H^+(b)$ for some $b \neq a$. A graph is* stiff *if it cannot be folded. A sequence of folds is called a* dismantling.

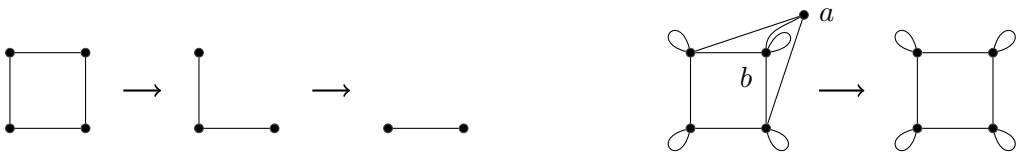


Figure 4.2: Two examples of dismantlings to stiff graphs.

It is well known that every graph can be dismantled to a unique stiff subgraph, see for example [HN04] or [FL12]. Folds preserve much of the properties studied in reconfiguration, in particular giving the following simple fact.

Proposition 4.3. *Let H be a graph and a, b be vertices of H such that $N_H^+(a) \subseteq N_H^+(b)$ and let $H' = H - a$ be a fold. Then H-Coloring Reachability is polynomially equivalent to H'-Coloring Reachability.*

Proof. For any H-coloring μ, define $\mu[a \to b]$ to be the coloring assigning the color b to each vertex to which μ assigns a, and the same color as μ otherwise. If H has a loop at a, that is, $a \in N^+H(a)$, then $a \in N^+(H)(b)$, so ab is an edge of H and $b \in N^+(H)(a)$, implying $b \in N^+(H)(b)$ and in particular that bb is an edge of H'. Therefore $\mu[a \to b]$ is an H'-coloring, because edges colored aa get mapped to bb, edges colored ac for some color $c \in V(H) - a$ get mapped to bc, which by assumption is an edge of H', while all other edges cc' for $c, c' \in V(H) - a$ still map to cc', which is also an edge of H'. This also implies that μ can be H-recolored to $\mu[a \to b]$ by recoloring vertices colored a to b in any order, because an edge colored aa can get recolored to ab and then to bb, while edges colored ac for $c \in V(H) - a$ can get recolored to bc.

Let (G, α, β) be an instance of H-Coloring Reachability. We claim that α can be H-recolored to β if and only if $\alpha[a \to b]$ can be H'-recolored to $\beta[a \to b]$. Indeed, if $\mu_0, \ldots, \mu_l$ is an H-recoloring sequence between $\mu_0 = \alpha$ and $\mu_l = \beta$, then $\mu_0[a \to b], \ldots, \mu_l[a \to b]$ is an H'-recoloring sequence. On the other hand, if $\mu'_0, \ldots, \mu'_l$ is an H'-recoloring sequence between $\mu'_0 = \alpha[a \to b]$ and $\mu'_l = \beta[a \to b]$, then it is also an H-recoloring sequence, and can be extended to a sequence between α and β by H-recoloring α to $\alpha[a \to b]$ and $\beta[a \to b]$ to β.

Let (G, α', β') be an instance of H'-Coloring Reachability. We claim that α' can be H'-recolored to β' if and only if it can be H-recolored. Clearly, an H'-recoloring sequence between α' and β' is also an H-recoloring sequence. On the other hand, if $\mu_0, \ldots, \mu_l$ is an H-recoloring sequence between α' and β', then $\mu_0[a \to b], \ldots, \mu_l[a \to b]$ is an H'-recoloring sequence between $\mu_0[a \to b] = \alpha'[a \to b] = \alpha'$ and $\mu_l[a \to b] = \beta'[a \to b] = \beta'$. □

The previous proposition allows us to limit our attention to stiff graphs H. In this case it is known that the identity mapping is an isolated node in the solution graph of H-colorings of H itself. This is used to prove the following fact. The *loop graph* is the graph with one vertex and one edge. Proofs are given in [HN04], where also another characterization of graphs H dismantlable to the loop graph can be found, by a cop-robber game.

Fact 4.4. *For a graph H, the solution graphs of H-colorings of G is connected for all graphs G if and only if H can be dismantled to the loop graph.*

Another construction useful in studying the complexity of H-Coloring Reachability is the product $G \times H$ of graphs G, H defined as having vertex set $V(G) \times V(H)$ and edges between (g_1, h_1) and (g_2, h_2) whenever $g_1 g_2 \in E(G)$ and $h_1 h_2 \in E(H)$. If μ is an H-coloring and μ' is and H'-coloring of G, then $\mu \times \mu'$ is the $H \times H'$-coloring of G defined as $(\mu \times \mu')(v) = (\mu(v), \mu'(v))$. On the other hand any $H \times H'$-coloring μ gives (via projections to H and H') an H-coloring μ_H and an H'-coloring $\mu_{H'}$ of the same graph such that $\mu = \mu_H \times \mu_{H'}$. It is also clear that an $H \times H'$-recoloring sequence between μ and ν exists if and only if an H-recoloring sequence from μ_H to ν_H and an H'-recoloring sequence from $\mu_{H'}$ to $\nu_{H'}$ exist. This implies the following equivalence.

Proposition 4.5. *Let H and K be graphs and suppose that K-Coloring and K-Coloring Reachability are solvable in polynomial time. Then $H \times K$-Coloring Reachability is polynomially equivalent to the H-Coloring Reachability problem limited to K-colorable input graphs.*

Proof. Let (G, μ_H, ν_H) be an instance of the H-recoloring such that G is K-colorable. Then a K-coloring α of G can be found in polynomial time to give an equivalent instance $(G, \mu_H \times \alpha, \nu_H \times \alpha)$ of $H \times K$-recoloring.

Let (G, μ, ν) be an instance of $H \times K$-recoloring. Then $\mu = \mu_H \times \mu_K$ and $\nu = \nu_H \times \nu_K$. If μ_K cannot be K-recolored to ν_K, which can be decided in polynomial time, then the answer is no. Otherwise the instance is equivalent to the instance (G, μ_H, ν_H) of H-Coloring Reachability where G is K-colorable by the coloring μ_K. □

2-Coloring Reachability is trivially in P. Bonsma and Cereceda [BC09] have shown that the 4-Coloring Reachability is PSPACE-complete even in bipartite graphs. By Proposition 4.5, this is equivalent to the statement that $K_4 \times K_2$-Coloring Reachability is PSPACE-complete. (Note that $K_4 \times K_2$ can also be described as $K_{4,4}$ with a perfect matching removed, the crown graph S_4^0 on 8 vertices, or the cubical graph). This holds even though $K_4 \times K_2$ is bipartite, placing the underlying $K_4 \times K_2$-Coloring problem in P. Indeed, H-Coloring for a graph H is in P whenever H is bipartite or has a loop and NP-complete otherwise, a conjecture ultimately proved by Hell and Nešetřil [HN90]. Recall that, on the other hand K_3-Coloring is NP-complete, while K_3-Coloring Reachability is in P [CHJ11]. This shows that there is no obvious correspondence between the complexity of reconfiguration problems considered here and that of their underlying problems.

4.2. Monochromatic neighborhood

To consider a generalization of 3-Coloring we begin with a few definitions.

Definition 4.6. *A graph H has the* monochromatic neighborhood property *if for every $u, v \in V(H)$, u and v have at most one common neighbor, i.e., $|N_H^+(u) \cap N_H^+(v)| \leqslant 1$ for $u \neq v$. The color in $N_H^+(u) \cap N_H^+(v)$, if there is one, is then called $h(u, v)$.*

The monochromatic neighborhood property is equivalent to the requirement that H contain no C_4, X_3 or X_2 as a subgraph (not necessarily induced), where X_3 is a triangle with one loop added and X_2 is a K_2 with both loops added. This includes all graphs with no cycles of length less than 5, for example.

For two graphs G and H, we call a single reconfiguration step between two H-colorings μ, μ' of G a *flip*, and denote it as $c_1 \xrightarrow{v} c_2$ with $c_1, c_2 \in V(H), v \in V(G)$ if $\mu(v) = c_1$ and $\mu'(v) = c_2$ (by definition no other vertices change color). Every H-recoloring sequence then corresponds to a *flip sequence* and the *flip sequence of* v (for $v \in V(G)$) is the subsequence of flips that change the color of v. *Forgetting* a flip $c_1 \xrightarrow{v} c_2$ gives an H-recoloring sequence of length decreased by one, with a flip sequence where $c_1 \xrightarrow{v} c_2$ is deleted and the next flip of v, if there is one, is $c_1 \xrightarrow{v} c_3$ instead of $c_2 \xrightarrow{v} c_3$.

The next lemma shows that in a shortest recoloring sequence, no vertex changes color from a to b and then (possibly after flips of other vertices) back to a. The statement is slightly stronger, in particular to make the inductive proof work.

Lemma 4.7. *Let G, H be a graphs such that H has the monochromatic neighborhood property. Let S be a shortest H-recoloring sequence of G. Then for any vertex v of G, the flip sequence of v in S never contains consecutive flips $c_1 \xrightarrow{v} c_2$ followed by $c_2 \xrightarrow{v} c_3$ such that $h(c_1, c_2) = h(c_2, c_3)$.*

Proof. Suppose there is counterexample and let S be the shortest one. $S = \mu_0, \ldots, \mu_l$ is then a shortest H-recoloring sequence of G between α and β such that the first flip of S is $c_1 \xrightarrow{v} c_2$, the last flip of S is $c_2 \xrightarrow{v} c_3$, there are no other flips of v in S, and $h(c_1, c_2) = h(c_2, c_3)$.

Consider any neighbor of v, say $w \in N(v)$. Since vw is an edge of G, it must be that $\alpha(w) \in N_H^+(\alpha(v)) = N_H^+(c_1)$, but also of $\alpha(w) \in N_H^+(\mu_1(v)) = N_H^+(c_2)$, because w did not change its color in the first flip. Therefore, by the monochromatic neighborhood property, $\alpha(w) = h(c_1, c_2)$. Similarly $\beta(w) = h(c_2, c_3)$. Suppose that w changes color at some step of S. Then w must change color at least twice (because $\alpha(w) = \beta(w)$), so let $d_1 \xrightarrow{w} d_2, d_2 \xrightarrow{w} d_3$ be the first two flips of w. As v changes color only in the first and last flip of S, it has color c_2 in every H-coloring in the sequence except α and β. Since w is a neighbor of v it must be then that $d_1, d_2, d_3 \in N_H^+(c_2)$. But that means $h(d_1, d_2) = h(d_2, d_3) = c_2$, which contradicts that S is the shortest counterexample.

Thus neighbors of v have the color $h(c_1, c_2) = h(c_2, c_3)$ throughout all the sequence S. But this color satisfies $c_1 \in N_H^+(h(c_1, c_2))$, so the sequence obtained from S by forgetting the first flip $c_1 \xrightarrow{v} c_2$ is a valid reconfiguration sequence, contradicting that S is a shortest reconfiguration sequence between α and β. □

The monochromatic neighborhood property implies a very strong restriction on recoloring sequences: for two adjacent vertices $v, w \in V(G)$, the flip sequence of one is uniquely determined by its initial color, its final color and the flip sequence of its neighbor. In particular it turns out that the flip sequences of adjacent vertices differ in length by at most 1.

Lemma 4.8. *Let G, H be graphs such that H has the monochromatic neighborhood property. Let S be a shortest H-recoloring sequence of G between α, β and let v, w be adjacent vertices of G. If the flip sequence of v in S is*

$$c_0 \xrightarrow{v} c_1, c_1 \xrightarrow{v} c_2, \ldots, c_{k-1} \xrightarrow{v} c_k,$$

then the flip sequence of v and w in S is exactly

$$c_0 \xrightarrow{v} c_1, d_1 \xrightarrow{w} d_2, c_1 \xrightarrow{v} c_2, d_2 \xrightarrow{w} d_3, \ldots, d_{k-1} \xrightarrow{w} d_k, c_{k-1} \xrightarrow{v} c_k,$$

where $d_i = h(c_{i-1}, c_i)$ for $i = 1, \ldots, k$, with a flip $\alpha(w) \xrightarrow{w} d_1$ prepended if $\alpha(w) \neq d_1$ and a flip $d_k \xrightarrow{w} \beta(w)$ appended if $d_k \neq \beta(w)$.

Proof. Let $S = \mu_0, \ldots, \mu_l$ and let $j(1), \ldots, j(k)$ be indices such that $\mu_{j(i)}$ is obtained from $\mu_{j(i)-1}$ by the flip $c_{i-1} \xrightarrow{v} c_i$, that is, $\mu_{j(i)-1}(v) = c_{i-1}$ and $\mu_{j(i)}(v) = c_i$ for $i = 1, \ldots, k$.

Since vw is an edge of G, it must be that $\mu_j(w) \in N_H^+(\mu_j(v))$ for $j = 0, \ldots, l$. As only one color changes in every step, $\mu_{j(i)-1}(w) = \mu_{j(i)}(w)$ for $i = 1, \ldots, k$. Therefore, $\mu_{j(i)-1}(w) = \mu_{j(i)}(w) = h(c_{i-1}, c_i)$.

Let us extend the definitions with $j(0) = 0$ and $j(k) = l + 1$, $d_0 = \alpha(w)$ and $d_{k+1} = \beta(w)$, so that S is divided into subsequences $S_i = \mu_{j(i)}, \mu_{j(i)+1}, \ldots, \mu_{j(i+1)-1}$ for $i = 0, \ldots, k$ satisfying

- v has color c_i throughout the sequence S_i,
- w has color d_i at the beginning and d_{i+1} at the end of S_i.

Consider the flip sequence of w in S_i. If it contained at least two flips, then any consecutive two flips $e_1 \xrightarrow{w} e_2, e_2 \xrightarrow{w} e_3$ would satisfy $e_1, e_2, e_3 \in N_H^+(c_i)$ and thus $h(e_1, e_2) = h(e_2, e_3) = c_i$, contradicting Lemma 4.7. Therefore there is at most one flip of w in this subsequence, which must be $d_i \xrightarrow{w} d_{i+1}$ if $d_i \neq d_{i+1}$, and there can be no flip of w in the opposite case. Notice that for $i = 1, \ldots, k-1$, $d_i \neq d_{i+1}$ by Lemma 4.7. Since S is the concatenation of sequences S_i and the flips happening in between are flips of v, the claim follows. □

Lemma 4.8 implies that in a shortest recoloring sequence, between every two flips of a vertex there is exactly one flip of each of its neighbors. The following lemma shows that by reordering flips we can make this true also for non-neighbors.

Lemma 4.9. *Let G, H be a graphs and let S be a shortest H-recoloring sequence of G between α and β. Then the flip sequence corresponding to S can be reordered (giving a valid shortest H-recoloring sequence between α and β) so that for every $v, w \in V(G)$, the k-th flip of v occurs before the k'-th flip of w whenever both exist and $k < k'$.*

Proof. The proof is by induction on the length of S. The case when S has no flips is trivial. So consider the last flip of S, say f – it is the k-th flip of some vertex v, for some k. Let S' be the sequence obtained from S by forgetting f. By inductive assumption, the flips of the sequence S' can be reordered into a sequence of flips that is a concatenation $F'_1 F'_2 \ldots F'_l$ of sequences such that the k'-th flip of each vertex happens in $F'_{k'}$, giving a valid H-recoloring sequence after which f can be applied. Since v has only $k-1$ flips in S', by Lemma 4.8 all its neighbors have at most k flips in S'. Hence v is non-adjacent to (and different from) the vertices that have flips in $F'_{k'}$ for $k' > k$, implying that ordering of f relative to the flips in $F'_{k'}$ is irrelevant. Therefore $F'_1 F'_2 \ldots F'_k f F'_{k+1} \ldots F'_l$ is a reordering of the flips of S giving a valid H-recoloring sequence, satisfying the claim. □

We conclude this section by proving the intuitively simple idea that Lemma 4.8 can be extended to all of the graph G, uniquely determining all of the recoloring sequence up to some reordering of flips. We say that a family $\mathcal{F}$ of recoloring sequences is listed *up to permutation* if a subfamily $\mathcal{F}'$ is outputted such that every sequence in $\mathcal{F}$ can be obtained from a sequence in $\mathcal{F}'$ by reordering flips.

Lemma 4.10. *Let G, H be graphs such that G is connected and H has the monochromatic neighborhood property. Let α and β be H-colorings of G and let S_v be a flip sequence of a vertex v. Then all shortest H-recoloring sequences S between α and β in which the flip sequence of v is S_v can be listed up to permutation in time polynomial in $|S_v|$ and $|G|$.*

Proof. Let T be a spanning tree of G rooted at v. Any H-recoloring sequence of G is also an H-recoloring of its subgraph T. By Lemma 4.8, the flip sequence S_u of every vertex u in T is uniquely determined (inductively by the flip sequence of its parent), and has length at most $|S_v| + |V(T)|$. All the sequences S_u can thus be computed in time linear in their length. Consider any shortest H-recoloring sequence S of G between α and β in which the flip sequence of v is S_v. Then the flip sequence of u in S is S_u for every vertex u. In particular, the multiset $M(S)$ of flips in S is known and has size at most $(|S_v| + |V(G)|) \cdot |V(G)|$.

Furthermore, the order of flips of each vertex is known. For any edge uu' in G, if the flip sequence of u' in S determined by Lemma 4.8 from S_u is different than $S_{u'}$, then this is a contradiction showing that no such sequence exists; otherwise the position of each flip in S_u relative to the positions of flips of $S_{u'}$ is also uniquely determined by this lemma. Together, this gives a relation on $M(S)$ describing for each pair of flips whether one has

been determined to be earlier than the other. The sequence S obeys this relation, and any total ordering of $M(S)$ that obeys the relation is a valid flip sequence between α and β in which the flip sequence of v is S_v – to verify this, observe that the flip sequence of any two adjacent vertices u and u' is given by Lemma 4.8, which guarantees the edge uu' is always mapped to an edge of H.

Therefore the recoloring sequences sought are exactly sequences obtained by ordering $M(S)$ in a way that obeys the relation. In particular there might be no such ordering, implying that there is no valid reconfiguration sequence with the properties sought. Computing the relation (together with the check for contradictions) and a topological ordering of $M(S)$ with this relation can be done in time $\mathcal{O}(|E(G)| \cdot |M(S)| + |M(S)|^2) = \mathcal{O}(|M(S)|^2)$. □

4.3. 3-Coloring Reachability

We now consider the problem of 3-Coloring Reachability. It is equivalent to K_3-Coloring Reachability and K_3 has the monochromatic neighborhood property, therefore lemmas from the previous section apply for $H = K_3$. The following short observation is the only missing ingredient.

Lemma 4.11. *Let G be a graph and let S be a shortest 3-recoloring sequence of G between α, β. Then there is a vertex whose flip sequence in S has length at most 2.*

Proof. Suppose to the contrary that every vertex of G has at least 3 flips in S. Consider any vertex v of G. Let $a \xrightarrow{v} b$ be the first flip of v in S. By Lemma 4.7, its second flip cannot be $b \xrightarrow{v} a$, it is thus $b \xrightarrow{v} c$ for some colors a, b, c such that $V(K_3) = \{a, b, c\}$. Similarly, the third flip cannot be $c \xrightarrow{v} b$, it is thus $c \xrightarrow{v} a$. In particular, the color of v after its first three flips is $\alpha(v)$ again, and this holds for any vertex v of G.

By Lemma 4.9 the flips can be reordered into a concatenation $F_1F_2 \dots F_l$ of sequences of flips such that the k-th flip of each vertex (if there is one) occurs in F_k and $F_1F_2F_3$ gives a shortest K_3-recoloring sequence. By the previous observations this is a sequence recoloring α to α, which of course cannot be shortest, a contradiction. □

Lemma 4.11 implies that shortest reconfiguration sequences can be found in polynomial time by an algorithm that guesses a vertex v, guesses a flip sequence of v of length at most 2 and executes the algorithm of Lemma 4.10. In disjoint graphs recoloring sequences for different components can of course be described and found independently of one another. This gives in particular a new polynomial algorithm for 3-Coloring Reachability and 3-Coloring Shortest Reachability.

Corollary 4.12. *Let G be a connected graph and let α, β be 3-colorings. Then all shortest 3-recoloring sequences between α and β can be listed up to permutation in polynomial time.*

Lemma 4.11 and the corollary can easily be extended to any connected 2-regular graph H (i.e., cycles and graphs obtained from paths by adding loops to both ends), with a different constant depending on H. Examples can be constructed to show that this cannot be the case for all graphs H with the monochromatic neighborhood property, as shortest sequences may require each vertex to change color a number of times unbounded by functions of H. Nonetheless, we conjecture that this property is enough to guarantee a polynomial time algorithm for H-Coloring Reachability. Note in particular that [CHJ11] first gave an algorithm for 3-Coloring Shortest Reachability that worked only under the assumption

that there is a vertex which cannot ever change its color (it is 'frozen'). Such an assumption would immediately imply a form of Lemma 4.11 and therefore the conjecture.

Conjecture 4.13. *Let H be a graph with the monochromatic neighborhood property. Then H-*Coloring Reachability *is in P.*

It would also be interesting to find more stiff graphs H for which the reachability problems are hard. All known results of this kind follow from the construction given in [HD05] (K_k and $K_k \times K_2$ for $k \geqslant 4$) or Proposition 2.11. The former has already been applied to show PSPACE-hardness of several popular puzzles [HD09] and to show hardness of deciding the equivalence of proofs in a certain logic [HH14]. Resolving such questions could potentially lead to an extension of the dichotomy of [Gop+09] to all constraint satisfaction problems.

Bibliography

[BB13] M. Bonamy and N. Bousquet. "Recoloring bounded treewidth graphs". *Electronic Notes in Discrete Mathematics* 44 (2013), pp. 257–262.

[BC09] P. Bonsma and L. Cereceda. "Finding paths between graph colourings: PSPACE-completeness and superpolynomial distances". *Theoretical Computer Science* 410.50 (2009), pp. 5215–5226.

[BKW14] P. Bonsma, M. Kamiński, and M. Wrochna. "Reconfiguring Independent Sets in Claw-Free Graphs". *arXiv preprint arXiv:1403.0359* (2014).

[BO84] G. Bauer and F. Otto. "Finite complete rewriting systems and the complexity of the word problem". *Acta Informatica* 21.5 (1984). 00067, pp. 521–540.

[Bon12] P. Bonsma. "Rerouting shortest paths in planar graphs". *FSTTCS 2012*. Vol. 18. Leibniz International Proceedings in Informatics (LIPIcs). 2012, pp. 337–349.

[Bon13] P. Bonsma. "The complexity of rerouting shortest paths". *Theoretical Computer Science* 510 (2013), pp. 1–12.

[Bon14] P. Bonsma. "Independent set reconfiguration in cographs". *preprint arXiv:1402.1587* (2014).

[BW04] G. R. Brightwell and P. Winkler. "Graph homomorphisms and long range action". *DIMACS Series in Discrete Mathematics and Theoretical Computer Science* 63 (2004), pp. 29–48.

[Cer07] L. Cereceda. "Mixing graph colourings". PhD thesis. London School of Economics, 2007.

[CHJ11] L. Cereceda, J. van den Heuvel, and M. Johnson. "Finding paths between 3-colorings". *Journal of graph theory* 67.1 (2011), pp. 69–82.

[Cou90] B. Courcelle. "The monadic second-order logic of graphs. I. Recognizable sets of finite graphs". *Information and computation* 85.1 (1990), pp. 12–75.

[CS05] M. Chudnovsky and P. D. Seymour. "The structure of claw-free graphs." *Surveys in combinatorics* 327 (2005), pp. 153–171.

[Dai80] D. P. Dailey. "Uniqueness of colorability and colorability of planar 4-regular graphs are NP-complete". *Discrete Mathematics* 30.3 (1980), pp. 289–293.

[Die12] R. Diestel. *Graph Theory, 4th Edition*. Vol. 173. Graduate texts in mathematics. Springer, 2012.

[Doc09] A. Dochtermann. "Hom complexes and homotopy theory in the category of graphs". *European Journal of Combinatorics* 30.2 (2009), pp. 490–509.

[Dye+06] 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 (2006), pp. 450–465.

[FL12] E. Fieux and J. Lacaze. “Foldings in graphs and relations with simplicial complexes and posets”. *Discrete Mathematics* 312.17 (2012), pp. 2639–2651.

[FOS11] Y. Faenza, G. Oriolo, and G. Stauffer. “An algorithmic decomposition of claw-free graphs leading to an $O(n^3)$-algorithm for the weighted stable set problem”. *SODA*. San Francisco, California: SIAM, 2011, pp. 630–646.

[FT05] U. Feige and K. Talwar. “Approximating the bandwidth of caterpillars”. *Approximation, Randomization and Combinatorial Optimization. Algorithms and Techniques*. Springer, 2005, pp. 62–73.

[FV98] T. Feder and M. Y. Vardi. “The computational structure of monotone monadic SNP and constraint satisfaction: a study through Datalog and group theory”. *SIAM Journal on Computing* 28.1 (1998), pp. 57–104.

[Gop+09] P. Gopalan, P. G. Kolaitis, E. Maneva, and C. H. Papadimitriou. “The connectivity of Boolean satisfiability: computational and structural dichotomies”. *SIAM Journal on Computing* 38.6 (2009), pp. 2330–2355.

[Gro07] M. Grohe. “The complexity of homomorphism and constraint satisfaction problems seen from the other side”. *Journal of the ACM (JACM)* 54.1 (2007), p. 1.

[HD05] R. A. Hearn and E. D. Demaine. “PSPACE-completeness of sliding-block puzzles and other problems through the nondeterministic constraint logic model of computation”. *Theoretical Computer Science* 343.1 (2005), pp. 72–96.

[HD09] R. A. Hearn and E. D. Demaine. *Games, puzzles, and computation*. AK Peters Wellesley, 2009.

[HH14] W. Heijltjes and R. Houston. “No proof nets for MLL with units”. *CSL-LICS*. 2014.

[HN04] P. Hell and J. Nešetřil. *Graphs and homomorphisms*. Vol. 28. Oxford University Press Oxford, 2004.

[HN90] P. Hell and J. Nešetřil. “On the complexity of H-coloring”. *Journal of Combinatorial Theory, Series B* 48.1 (1990), pp. 92–110.

[Ito+12] T. Ito, K. Kawamura, H. Ono, and X. Zhou. “Reconfiguration of list L(2, 1)-labelings in a graph”. *Algorithms and Computation*. Springer, 2012, pp. 34–43.

[Joh+14] M. Johnson, D. Kratsch, S. Kratsch, V. Patel, and D. Paulusma. “Colouring reconfiguration is fixed-parameter tractable”. *preprint arXiv:1403.6347* (2014).

[KMM11] M. Kamiński, P. Medvedev, and M. Milanič. “Shortest paths between shortest paths and independent sets”. *Combinatorial Algorithms*. Springer, 2011, pp. 56–67.

[KMM12] M. Kamiński, P. Medvedev, and Martin Milanič. “Complexity of independent set reconfigurability problems”. *Theoretical Computer Science* 439 (2012), pp. 9–15.

[KNC07] M. Y. Kovalyov, C. Ng, and T. Cheng. “Fixed interval scheduling: Models, applications, computational complexity and algorithms”. *European Journal of Operational Research* 178.2 (2007), pp. 331–342.

[Koz06] D. Kozen. *Theory of computation*. Springer, 2006.

[Lin+11] M. C. Lin, D. Rautenbach, F. J. Soulignac, and J. L. Szwarcfiter. “Powers of cycles, powers of paths, and distance graphs”. *Discrete Applied Mathematics* 159.7 (2011), pp. 621–627.

[LP80] H. R. Lewis and C. H. Papadimitriou. “Symmetric space-bounded computation”. *Automata, Languages and Programming*. Springer, 1980, pp. 374–384.

[Mar47] A. A. Markov. “On the impossibility of certain algorithms in the theory of associative systems”. *Dokl. Akad. Nauk SSSR*. Vol. 55. 1947, pp. 587–590.

[Min80] G. J. Minty. “On maximal independent sets of vertices in claw-free graphs”. *Journal of Combinatorial Theory, Series B* 28.3 (1980), pp. 284–304.

[Moh01] B. Mohar. “Face covers and the genus problem for apex graphs”. *Journal of Combinatorial Theory, Series B* 82.1 (2001), pp. 102–117.

[Mou+14] A. E. Mouawad, N. Nishimura, V. Raman, and M. Wrochna. “Reconfiguration over tree decompositions”. *arXiv preprint arXiv:1405.2447* (2014).

[NdM12] J. Nešetřil and P. O. de Mendez. *Sparsity: Graphs, Structures, and Algorithms*. Vol. 28. Springer, 2012.

[Neš07] J. Nešetril. “Homomorphisms of structures (concepts and highlights)”. *Physics and Theoretical Computer Science: From Numbers and Languages to (Quantum) Cryptography Security* 7 (2007), p. 295.

[NOdM06] J. Nešetřil and P. Ossona de Mendez. “Tree-depth, subgraph coloring and homomorphism bounds”. *European Journal of Combinatorics* 27.6 (2006), pp. 1022–1041.

[NOdM08] J. Nešetřil and P. Ossona de Mendez. “Grad and classes with bounded expansion I. Decompositions”. *European Journal of Combinatorics* 29.3 (2008), pp. 760–776.

[NS13] P. Nobili and A. Sassano. “A reduction algorithm for the weighted stable set problem in claw-free graphs”. *Discrete Applied Mathematics* (2013).

[NT01] D. Nakamura and A. Tamura. “A revision of Minty’s algorithm for finding a maximum weight stable set of a claw-free graph”. *Journal of the Operations Research Society of Japan* 44.2 (2001), pp. 194–204.

[Pos47] E. L. Post. “Recursive unsolvability of a problem of Thue”. *The Journal of Symbolic Logic* 12.1 (1947), pp. 1–11.

[Sav70] W. J. Savitch. “Relationships between nondeterministic and deterministic tape complexities”. *Journal of computer and system sciences* 4.2 (1970), pp. 177–192.

[Sbi80] N. Sbihi. “Algorithme de recherche d’un stable de cardinalité maximum dans un graphe sans étoile”. *Discrete Mathematics* 29.1 (1980), pp. 53–76.

[Sch03] A. Schrijver. *Combinatorial optimization: polyhedra and efficiency*. Vol. 24. Springer, 2003.

[Sch13] K. Schwerdtfeger. “A computational trichotomy for connectivity of boolean satisfiability”. *arXiv preprint arXiv:1312.4524* (2013).

[Sch78] T. J. Schaefer. “The complexity of satisfiability problems”. *STOC ’78*. Vol. 78. 1978, pp. 216–226.

[Sch99] T. Schiex. *A note on CSP graph parameters*. Tech. rep. INRA, 1999.

[Tse58] G. S. Tseitin. “An associative calculus with an insoluble problem of equivalence”. *Trudy Matematicheskogo Instituta im. VA Steklova* 52 (1958), pp. 172–189.

[Wro14] M. Wrochna. “Reconfiguration in bounded bandwidth and treedepth”. *arXiv preprint arXiv:1405.0847* (2014).