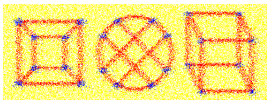
Journal of Graph Algorithms and Applications
http://jgaa.info/ vol. 27, no. 6, pp. 409–431 (2023)
DOI: 10.7155/jgaa.00628

Order Reconfiguration under Width Constraints

Emmanuel Arrighi[1,2] *Henning Fernau*[2]
Mateus de Oliveira Oliveira[1,3] *Petra Wolf*[1,2]

[1]University of Bergen, Norway
[2]University of Trier, Germany
[3]Stockholm University, Sweden

Submitted: March 2022 Reviewed: December 2022 Revised: December 2022

Accepted: April 2023 Final: April 2023 Published: July 2023

Article type: Regular Paper

Communicated by: G. Da Lozzo and P. Kindermann

Abstract. In this work, we consider the following order reconfiguration problem: Given a graph G together with linear orders ω and ω' of the vertices of G, can one transform ω into ω' by a sequence of swaps of adjacent elements in such a way that, at each time step, the resulting linear order has cutwidth (pathwidth) at most k? We show that this problem always has an affirmative answer when the input linear orders ω and ω' have cutwidth (pathwidth) of at most $k/2$. This result also holds in a weighted setting. Using this result, we establish a connection between two apparently unrelated problems: the reachability problem for two-letter string rewriting systems and the graph isomorphism problem for graphs of bounded cutwidth. This opens an avenue for the study of the famous graph isomorphism problem using techniques from term rewriting theory.

1 Introduction

In the field of reconfiguration, one is interested in studying relationships among solutions of a problem instance [30, 41, 44]. Here, by reconfiguration of one solution into another one, we mean a sequence of steps where each step transforms a feasible solution into another one. In this context, three fundamental questions are the following ones:

Special Issue on Parameterized and Approximation Algorithms in Graph Drawing

Emmanuel Arrighi acknowledges support from the Research Council of Norway (Grant no. 274526) and from IS-DAAD (Grant no. 309319). Henning Fernau acknowledges support from DAAD PPP (Grant no. 57525246). Mateus de Oliveira Oliveira acknowledges support from the Research Council of Norway (Grant no. 288761), IS-DAAD (Grant no. 309319) and Sigma2 Network (NN9535K). Petra Wolf acknowledges support from DFG project FE 560/9-1 and DAAD PPP (Grant no. 57525246).

E-mail addresses: emmanuel@arrighi.eu (Emmanuel Arrighi) fernau@uni-trier.de (Henning Fernau) oliveira@dsv.su.se (Mateus de Oliveira Oliveira) mail@wolfp.net (Petra Wolf)

1. Is it the case that any two solutions can be reconfigured into each other?
2. Can any two solutions be reconfigured into each other in a polynomial number of steps?
3. Given two feasible solutions X and Y, can one find in polynomial time a reconfiguration sequence from X to Y?

In this paper, we study reconfiguration problems in the context of linear arrangements of the vertices of a given graph G. The space of feasible solutions is the set of all linear orders of width[1] at most k for some given $k \in \mathbb{N}$. We say that a linear order ω can be *reconfigured* into a linear order ω' *in width* k if there is a sequence $\omega_1, \ldots, \omega_m$ of linear orders of width at most k such that $\omega_1 = \omega$, $\omega_m = \omega'$ and for each $i \in \{2, \ldots, m\}$, ω_i is obtained from ω_{i-1} by swapping two adjacent vertices. Our main result (Theorem 1) states that if ω and ω' are linear orders of cutwidth at most k, then ω can be reconfigured into ω' in width at most $2k$. Additionally, reconfiguration in width at most $2k$ can be done using at most $\mathcal{O}(n^2)$ swaps. Finally, we show that a reconfiguration sequence can be found in polynomial time. We also show that these results transfer to pathwidth instead of cutwidth and to weighted version thereof. Notice that it is known that the mentioned (unweighted) parameters are quite related, see [8] also concerning fixed-parameter tractability results.

Our results on reconfiguration of linear arrangements can be used to establish an interesting connection between two apparently unrelated computational problems: reachability for two-letter string rewriting and graph isomorphism.

A *two-letter rewriting rule* over a given alphabet Σ is a rewriting rule of the form $ab \to cd$ for letters $a, b, c, d \in \Sigma$. A *two-letter string rewriting system* is a collection R of two-letter string rewriting rules. The *reachability problem* for such a rewriting system R is the problem of determining whether a given string $x \in \Sigma^n$ can be transformed into a given string $y \in \Sigma^n$ by the application of a sequence of two-letter rewriting rules of R. On the other hand, in the *graph isomorphism problem*, we are given two graphs, G and G', and the goal is to determine whether there exists a bijection φ from the vertex set of G to the vertex set of G' in such a way that an edge $\{u, v\}$ belongs to G if and only if the edge $\{\varphi(u), \varphi(v)\}$ belongs to G'.

In order to describe more precisely the connections between two-letter term rewriting and graph isomorphism, we briefly discuss the notions of slices and unit decompositions. These notions were originally defined in the study of the partial order theory of concurrency [11] and, later, used to prove algorithmic results in the realm of directed width measures [12, 13]. In this work, we provide simplified definitions of slices and unit decompositions that are more adequate to our context.

A *slice* is a graph $\mathbf{S}$ where the vertices are partitioned into a center C and special in-frontier I and out-frontier O that can be used for composition. A slice $\mathbf{S}_1$ can be glued to a slice $\mathbf{S}_2$ if the out-frontier of $\mathbf{S}_1$ can be coherently identified with the in-frontier of $\mathbf{S}_2$. In this case, the gluing gives rise to a bigger slice $\mathbf{S}_1 \circ \mathbf{S}_2$ which is obtained by identifying the out-frontier of $\mathbf{S}_1$ with the in-frontier of $\mathbf{S}_2$. A *unit slice* is a slice with a unique vertex in the center. Any slice $\mathbf{S}$ can be decomposed into a sequence of unit slices. More specifically, a *unit decomposition* is a sequence $\mathbf{U} = \mathbf{S}_1\mathbf{S}_2 \ldots \mathbf{S}_n$ of unit slices with the property that for each $i \in [n-1]$, $\mathbf{S}_i$ can be glued to the slice $\mathbf{S}_{i+1}$. The result of gluing the unit slices in $\mathbf{U}$ is a slice $\mathring{\mathbf{U}}$ with n center vertices. Conversely, any slice $\mathbf{S}$ with n center vertices can be written as a unit decomposition $\mathbf{U} = \mathbf{S}_1\mathbf{S}_2 \ldots \mathbf{S}_n$ with the property that $\mathring{\mathbf{U}}$ is isomorphic to $\mathbf{S}$.

An important remark connecting unit decompositions and the notion of cutwidth is that if a slice $\mathbf{S}$ has cutwidth k, then $\mathbf{S}$ can be decomposed into a unit decomposition $\mathbf{U} = \mathbf{S}_1\mathbf{S}_2 \ldots \mathbf{S}_n$

[1] This means, in this paper, always cutwidth or vertex separation number, also known as pathwidth, as formally defined in Definitions 1 and 2.

where for each $i \in [n]$, $\mathbf{S}_i$ has at most k vertices in each frontier except for the in-frontier of $\mathbf{S}_1$ and the out-frontier of $\mathbf{S}_n$. Therefore, if we let $\mathbf{\Sigma}(k)$ denote the set of all unit slices with frontiers of size at most k, then any graph G with n vertices of cutwidth at most k can be written as a word (unit decomposition) of length n over the alphabet $\mathbf{\Sigma}(k)$. In this work, for each $k \in \mathbb{N}$, we introduce a suitable two-letter string rewriting system $R(k)$ over the alphabet $\mathbf{\Sigma}(k)$ with the following property: if $\mathbf{U}$ and $\mathbf{U}'$ are two unit decompositions over $\mathbf{\Sigma}(k)$ and if $\mathbf{U}$ can be transformed into $\mathbf{U}'$ using the rewriting rules in $R(k)$, then the graphs $\mathring{\mathbf{U}}$ and $\mathring{\mathbf{U}}'$ are isomorphic. Our second main result is a partial converse for this property. More precisely, we show that given two unit decompositions $\mathbf{U}$ and $\mathbf{U}'$ over $\mathbf{\Sigma}(k)$, if the graphs $\mathring{\mathbf{U}}$ and $\mathring{\mathbf{U}}'$ are isomorphic, then each of these unit decompositions can be transformed into one another by the application of rewriting rules from the string rewriting system $R(2k)$ (Theorem 2).

The proof of Theorem 2 is heavily based on Theorem 1. An important feature of this proof is that, given an isomorphism from $\mathring{\mathbf{U}}$ to $\mathring{\mathbf{U}}'$, one can construct a sequence of rewriting steps transforming $\mathbf{U}$ into $\mathbf{U}'$. Conversely, given any such a sequence, we are able to construct an isomorphism from $\mathring{\mathbf{U}}$ to $\mathring{\mathbf{U}}'$. This result, together with the fact that unit decompositions of minimum cutwidth can be approximated in FPT time, implies that the graph isomorphism problem for graphs of cutwidth at most k is FPT-equivalent to the reachability problem for $R(2k)$ (Theorem 4).

Related Work. The reachability problem for a given string rewriting system R consists in determining if a given string x can be transformed into a given string y by applying rewriting rules from R. Reachability is a central problem in the field of string rewriting [9] and can also be studied under the light of term rewriting theory [36, 7, 3, 9]. The complexity of the reachability problem is highly dependent on the rewriting system R. For general rewriting systems, the problem becomes undecidable [9]. In the case of two-letter rewriting, reachability is in PSPACE, since in this case, strings never grow in size. It is also not difficult to design two-letter rewriting systems for which the reachability problem is PSPACE-complete. Nevertheless, our results imply that for each $k \in \mathbb{N}$, the $R(2k)$-reachability problem for unit decompositions of length n and width at most k is reducible to the graph isomorphism problem. Therefore, it can be solved in time $n^{\mathsf{polylog}(n)}$, independently of k, using Babai's quasi-polynomial time algorithm for graph isomorphism [4]. For a nice historical account and a gentle introduction into the ideas behind this non-trivial algorithm, we refer to [28]. Further progress in the area of graph isomorphism algorithms is nicely reported in [24]. An interesting question we leave open is the complexity of $R(\alpha \cdot k)$-reachability for unit decompositions of width at most k if α is a rational number with $1 \leq \alpha < 2$; we do not know if there is such an α for which the reachability problem becomes PSPACE-hard.

In the field of parameterized complexity theory [15, 9], a computational problem is said to be *fixed-parameter tractable (or* FPT *for short)* with respect to a parameter k if it can be solved in time $f(k) \cdot n^{\mathcal{O}(1)}$ on inputs of size n. Here $f : \mathbb{N} \to \mathbb{N}$ is a computable function depending only on the parameter k, but not on the size n of the input. The Graph Isomorphism problem (GI for short) has been shown to be solvable in time $f(k) \cdot n^{\mathcal{O}(1)}$ (that is, FPT time) whenever the parameter k stands for eigenvalue multiplicity [5], treewidth [38], feedback vertex-set number [37], or the size of the largest color class [18, 2] (for colored input graphs). On the other hand, GI can be solved in time $f_1(k) \cdot n^{f_2(k)}$ (that is, in XP time), whenever the parameter k stands for genus [40], rankwidth [27], maximum degree [39], size of an excluded topological subgraph [22], size of an excluded minor [21], or Weisfeiler-Leman dimension [10], which is in fact related to the mentioned rankwidth parameterization [23]. We note that, in particular, Babai's algorithm and techniques have been recently used to improve the fastest FPT algorithm for graphs of treewidth at most k from $2^{\mathcal{O}(k^5 \cdot \log k)} \cdot n^{\mathcal{O}(1)}$ [38] to $2^{\mathcal{O}(k \cdot \mathsf{polylog}(k))} \cdot n^{\mathcal{O}(1)}$ [26], and for graphs of maximum degree d, the

fastest XP-algorithm has been improved from $n^{\mathcal{O}(d/\log d)}$ [6] to $n^{\mathsf{polylog}(d)}$ [25]. In particular, it is worth noting that graphs of cutwidth k have maximum degree at most $2k$ and treewidth $O(k)$. Therefore, isomorphism of graphs of cutwidth k can be solved in time $2^{O(k\cdot\mathsf{polylog}(k))}\cdot n^{\mathcal{O}(1)}$ [26]. This implies that $R(2k)$-reachability can be solved in $2^{O(k\cdot\mathsf{polylog}(k))}\cdot n^{\mathcal{O}(1)}$ time when restricted to unit decompositions of width at most k. Showing that isomorphism for graphs of cutwidth k can be solved in time $2^{\mathcal{O}(k)}\cdot n^{\mathcal{O}(1)}$ is still an interesting open problem.

Another width parameter for linear orders that has been studied in the context of graph theory is the vertex separation number of a graph [16]. This parameter may be seen as an order-theoretic interpretation of the notion of pathwidth. The techniques used to prove Theorem 1 can be generalized to prove that reconfiguration of linear orders of vertex separation number k can always be achieved in width at most $2k$ (Theorem 5). While we do not provide a string-rewriting interpretation of this result, we do state it formally in Section 5, since this result may be of independent interest in the field of reconfiguration.

A conference version of this work was presented at MFCS in 2021; see [1].

2 Preliminaries

Basics. We let $\mathbb{N}$ denote the set of natural numbers, including 0, and $\mathbb{N}_+$ denote the set of positive natural numbers. For each $n\in\mathbb{N}_+$, we let $[n]=\{1,\ldots,n\}$ denote the discrete interval of the first n positive integers. As a degenerate case, we let $[0]=\emptyset$.

Given a finite set S, we let $\mathcal{P}(S)$ denote the set of all subsets of S. For each $k\in\mathbb{N}$, we let $\mathcal{P}(S,k)$ and $\mathcal{P}(S,\leq k)$ denote the sets of subsets of S of size exactly k and at most k, respectively.

Graphs. In this work, graphs are simple and undirected. Given a graph G, we let $V(G)$ denote the vertex set of G and $E(G)$ denote the edge set of G. Given a subset $S\subseteq V(G)$, we let $G[S]$ be the subgraph of G induced by S. More precisely, $V(G[S])=S$ and $E(G[S])=E(G)\cap\mathcal{P}(S,2)$. An *isomorphism* from a graph G to a graph G' is a bijection $\varphi:V(G)\to V(G')$ such that for each $v,u\in V(G)$, $\{v,u\}\in E(G)$ if and only if $\{\varphi(v),\varphi(u)\}\in E(G')$. If such an isomorphism exists, we call G *isomorphic* to G'.

Order. Let V be a set with $|V|=n$. A linear order on V is a bijection $\omega:[n]\to V$. Intuitively, for each $j\in[n]$ and $v\in V$, $\omega(j)=v$ indicates that v is the j-th element of ω. If $S\subseteq[n]$, then we let $\omega(S)=\{\omega(j)\ :\ j\in S\}$ be the image of S under ω. Given linear orders $\omega,\omega':[n]\to V$ of V and a number $i\in[n-1]$, we write $\omega\xrightarrow{i}\omega'$ to indicate that ω' is obtained from ω by swapping the order of the vertices at positions i and $i+1$. More precisely, $\omega'(j)=\omega(j)$ for every $j\in[n]\setminus\{i,i+1\}$, $\omega'(i)=\omega(i+1)$, and $\omega'(i+1)=\omega(i)$.

Let $\omega:[n]\to V$ be a linear order on a set V. Let $S\subseteq V$. We let $\omega^S:[|S|]\to S$ be the linear order induced by ω on S. More precisely, if we write the elements of S in increasing order according to ω, then for each $i\in[|S|]$, $\omega^S(i)$ is the i-th element in this sequence.

Order Reconfiguration. We say that ω can be reconfigured into ω' in one swap, and denote this fact by $\omega\to\omega'$, if there exists some $i\in[n]$ such that $\omega\xrightarrow{i}\omega'$. We say that ω can be reconfigured into ω' in at most r swaps, and denote this fact by $\omega\to_r\omega'$, if there are numbers $r'\in[r]$, $i_1,\ldots,i_{r'}\in[n]$, and linear orders $\omega_0,\ldots,\omega_{r'}$ such that

$$\omega=\omega_0\xrightarrow{i_1}\omega_1\xrightarrow{i_2}\ldots\xrightarrow{i_{r'}}\omega_{r'}=\omega'.$$

We call this sequence a *reconfiguration sequence* from ω to ω'. The mere existence of a (possibly empty) reconfiguration sequence from ω to ω' is also written as $\omega \to^* \omega'$.

Composition of Linear Orders. Let $i \in \{0, \ldots, n\}$, and $\omega, \omega' : [n] \to V$. We let $\omega \oplus_i \omega' : [n] \to V$ be the linear order that orders the vertices in the subset $\omega([i]) \subseteq V$ according to ω, followed by the vertices in the subset $V \setminus \omega([i])$, ordered according to ω'. More precisely, $\omega \oplus_i \omega'$ is defined as follows for each $j \in [n]$.

$$\omega \oplus_i \omega'(j) = \begin{cases} \omega(j) & \text{if } j \leq i, \\ {\omega'}^{V \setminus \omega([i])}(j-i) & \text{if } j > i. \end{cases} \tag{1}$$

We note that in particular, $\omega \oplus_0 \omega' = \omega'$ and $\omega \oplus_n \omega' = \omega$.

String Rewriting. A *two-letter string rewriting system* is a pair (Σ, R) where Σ is a finite, non-empty set of symbols (an alphabet), and $R \subseteq \Sigma^2 \times \Sigma^2$ is a set of rewriting rules of the form $ab \to cd$. Let x and y be strings in Σ^n and $i \in [n-1]$. We say that x can be *transformed* into y by applying a rewriting rule $ab \to cd$ at position i if $x_i x_{i+1} = ab$, $y_i y_{i+1} = cd$ and $x_j = y_j$ for $j \notin \{i, i+1\}$. We write $x \xrightarrow{i} y$ to denote that x can be transformed into y by the application of some rewriting rule at position i. We write $x \to y$ to denote that x can be transformed into y by the application of some rewriting rule at some position $i \in [n-1]$. We say that y is *reachable* from x if there is a sequence of strings $x = x_0, x_1, \ldots, x_m = y$ such that $x_{i-1} \to x_i$ for each $i \in [m]$. We write $x \to^* y$ to denote that y is reachable from x. We say that x and y are *R-equivalent* if $x \to^* y$ and $y \to^* x$.

3 Linear Order Reconfiguration

Let G be an n-vertex graph with vertex set $V(G)$ and edge set $E(G)$. Given two disjoint sets $S, S' \subseteq V(G)$, we let $E(G, S, S') = \{\{u, v\} \in E(G) \ : \ u \in S, \ v \in S'\}$ be the set of edges with one endpoint in S and the other endpoint in S'. As a special case, we define $E(G, S) = E(G, S, V(G) \setminus S)$. We will often make use of the following two properties without explicit mentioning.

- **Monotonicity property:** If $T \subseteq S$ and $T' \subseteq S'$, then $E(G, T, T') \subseteq E(G, S, S')$.
- **Linearity property:** If $\{S_1, S_2\}$ is a partition of S, then $\{E(G, S_1, S'), E(G, S_2, S')\}$ is a partition of $E(G, S, S')$.

Let G be an n-vertex undirected graph with vertex set $V(G)$ and edge set $E(G)$. Let $\omega : [n] \to V(G)$ be a linear order on the vertices of G. For each $p \in [n]$, we let $\mathrm{cw}(G, \omega, p) = |E(G, \omega([p-1]))|$ be the number of edges that have one endpoint in the first $p-1$ vertices of the linear order ω and the other endpoint in the remaining vertices. $E(G, \omega([p-1]))$ is called the *cut* at position p and $\mathrm{cw}(G, \omega, p)$ is the *size of the cut* at position p.

Definition 1 (Cutwidth) *The* cutwidth *of the linear order ω on $V(G)$ is defined as*

$$\mathrm{cw}(G, \omega) = \max_{p \in [n]} \mathrm{cw}(G, \omega, p)\,.$$

The cutwidth *of the graph G is defined as $\mathrm{cw}(G) = \min_\omega \mathrm{cw}(G, \omega)$, where ω ranges over all linear orders on the vertex set $V(G)$.*

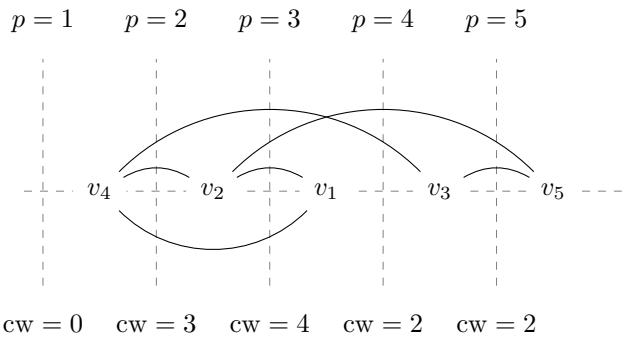

Figure 1: Visual representation of the cutwidth of a graph. ω orders the vertices in the following way v_4, v_2, v_1, v_3, v_5. The value at the top give the position of the cut and at the bottom, the value of this cut. In this example, the cutwidth is 4.

Given a graph G, the cutwidth of a given linear order ω can be intuitively understood by drawing G in a specific way on the plane. First, the vertices of G are placed on a horizontal line following the order given by ω. Then, edges are drawn as curves between the point representing their endpoints. Now, if we draw a vertical line between the $(p-1)$-th and p-th vertices in ω, then the number of edges that intersect this vertical line corresponds to $\mathrm{cw}(G, \omega, p)$. Figure 1 gives an example of such a visual representation of the cutwidth.

If $G = (V, E)$ is a graph that comes with *edge weights* $c : E \to \mathbb{N}_+$,[2] we let $\mathrm{wcw}(G, c, \omega, p) = c(E(G, \omega([p-1])))$, where $c(F) = \sum_{f \in F} c(f)$ for an edge subset F. Now, we can define the *weighted cutwidth* of the linear order ω on $V(G)$ as

$$\mathrm{wcw}(G, c, \omega) = \max_{p \in [n]} \mathrm{wcw}(G, c, \omega, p)\,.$$

The *weighted cutwidth* of the graph G is defined as $\mathrm{wcw}(G, c) = \min_\omega \mathrm{wcw}(G, c, \omega)$, where ω ranges over all linear orders on the vertex set $V(G)$.

For each $k \in \mathbb{N}$, and each n-vertex graph G, we let

$$\mathrm{CW}(G, k) = \{\omega : [n] \to V(G) \ : \ \mathrm{cw}(G, \omega) \leq k\}$$

be the set of linear orders of $V(G)$ of cutwidth at most k. We say that ω can be *reconfigured* into ω' in cutwidth at most k if there is a *reconfiguration sequence*

$$\omega = \omega_0 \xrightarrow{i_1} \omega_1 \xrightarrow{i_2} \cdots \xrightarrow{i_r} \omega_r = \omega'$$

such that for each $j \in \{0, \ldots, r\}$, $\omega_j \in \mathrm{CW}(G, k)$. We can again generalize these notions to the weighted setting by defining

$$\mathrm{WCW}(G, c, k) = \{\omega : [n] \to V(G) \ : \ \mathrm{wcw}(G, c, \omega) \leq k\}$$

for any edge weight function $c : E(G) \to \mathbb{N}_+$, so that we can also speak about reconfiguring ω into ω' in weighted cutwidth at most k (with respect to an arbitrary but fixed edge weight function c).

[2] Other sets of numbers are also possible, but we like to avoid discussions on how to compute with these numbers.

Given two linear orders ω and ω' of the vertices of an n-vertex graph, ω can always be reconfigured into ω'. To see this, notice that the bubble-sort algorithm performs only swaps. Therefore, running the bubble-sort algorithm to sort ω according to ω' gives a valid reconfiguration sequence from ω to ω'. This connection allows us to be more precise.

Lemma 1 *A linear order ω can always be reconfigured into another ordering ω' using a reconfiguration sequence of length at most n^2.*

An interesting question is to ask if ω can be reconfigured into ω' in such a way that every linear order appearing in the reconfiguration sequence has bounded (weighted) cutwidth.

Problem 1 (Bounded (Weighted) Cutwidth Order Reconfiguration) *Let G be an n-vertex graph, let $c : E(G) \to \mathbb{N}_+$ be a weight function, let $\omega, \omega' : [n] \to V(G)$ be linear orders on the vertex set of G, and let $k \in \mathbb{N}$. Is it true that ω can be reconfigured into ω' in weighted cutwidth at most k (with respect to c)?*

Given an instance $(G, c, \omega, \omega', k)$ of the Bounded Cutwidth Order Reconfiguration problem, it should be clear that if k is smaller than the cutwidth of the graph G, then the answer for Problem 1 is trivially no, since in this case neither ω nor ω' are in $\mathrm{WCW}(G, c, k)$. On the other hand, we will show in Theorem 1 below that the answer is always yes if k is at least twice the weighted cutwidth of the thickest input linear order.

Theorem 1 *Let G be an n-vertex graph, let $c : E(G) \to \mathbb{N}_+$ be a weight function and let $\omega, \omega' : [n] \to V(G)$ be linear orders of $V(G)$ of weighted cutwidth at most k. Then, ω can be reconfigured into ω' in weighted cutwidth (with respect to c) which is at most $\mathrm{wcw}(G, c, \omega) + \mathrm{wcw}(G, c, \omega') \leq 2k$.*

To prove this theorem, we need the following three lemmas. First, we will show that taking induced linear order does not increase the cutwidth (Lemma 2). Then, we will show that some specific intermediate linear orders have bounded cutwidth (Lemma 3). Finally, we will show that there is a reconfiguration sequence of bounded cutwidth between each consecutive pair of intermediate linear orders (Lemma 4). Extending on the conference version of this paper, we will see that all these results also hold in the more general setting of edge-weighted graphs.

We start by the monotonicity of the weighted cutwidth by taking induced linear orders.

Lemma 2 *Let G be an n-vertex graph, $S \subseteq V(G)$ and $\omega : [n] \to V(G)$ be a linear order on $V(G)$. Moreover, let $c : E(G) \to \mathbb{N}_+$ be a weight function. Then, ω^S is a linear order on $V(G[S])$. Additionally, $\mathrm{wcw}(G[S], c, \omega^S) \leq \mathrm{wcw}(G, c, \omega)$.*

Proof: As $S = V(G[S])$, ω^S is a linear order on $V(G[S])$. Let $p \in [|S|]$ and let $p' \in [n]$ be the unique number such that $\omega^S(p) = \omega(p')$. Then,

$$\begin{aligned}
\mathrm{wcw}(G[S], c, \omega^S, p) &= c(E(G[S], \omega^S([p-1]))) \\
&= c(E(G[S], \omega^S([p-1]), \{\omega^S(r) \ : \ r \geq p\})) \\
&= c(E(G, \omega^S([p-1]), \{\omega^S(r) \ : \ r \geq p\})) \\
&\leq c(E(G, \omega([p'-1]))) \\
&= \mathrm{wcw}(G, c, \omega, p') \\
&\leq \mathrm{wcw}(G, c, \omega)\,,
\end{aligned}$$

as $\omega^S([p-1]) \subseteq \omega([p'-1])$ and $\{\omega^S(r) \ : \ r \geq p\} \subseteq \{\omega(r') \ : \ r' \geq p'\} = V(G) \setminus \omega([p'-1])$. □

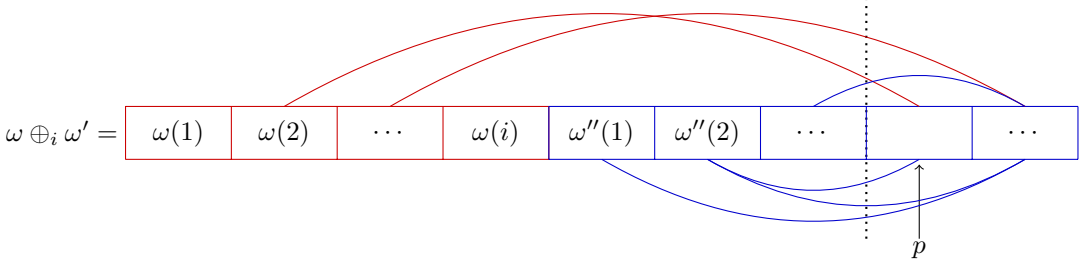

Figure 2: Illustration of Equality (a) in Lemma 3, using the unweighted setting for simplicity. In this figure, $\omega'' = \omega'^{V \setminus \omega([i])}$. The red part of the linear order follows the linear order ω for the first i elements, and the blue part of the linear order follows ω' for the remaining elements. Then, the set of edges crossing the cut at position p can be split in two, the set of edges that start from the red part and the set of edge that start from the blue part. The number of red edges is bounded by the cutwidth of ω and the number of blue edges is bounded by the cutwidth of ω'.

To prove Theorem 1, we will give a reconfiguration sequence that uses the composition of linear orders. At a high level, we will go from one linear order ω to the other ω' by using the composition of the two linear orders at each position $\omega' \oplus_i \omega$. Therefore, we show that the composition of two linear orders has bounded weighted cutwidth.

Lemma 3 *Let G be an n-vertex graph with edge weights $c : E(G) \to \mathbb{N}_+$ and let $\omega, \omega' : [n] \to V(G)$ be linear orders of $V(G)$ with weighted cutwidth of at most k. Then, for each $i \in [n]$, $\omega \oplus_i \omega'$ has weighted cutwidth (with respect to c) that is at most* $\mathrm{wcw}(G, c, \omega) + \mathrm{wcw}(G, c, \omega') \leq 2k$.

Proof: Let $i, p \in [n]$. By definition of the cutwidth, we have that

$$\begin{aligned}\mathrm{wcw}(G, c, \omega \oplus_i \omega', p) &= c(E(G, \omega \oplus_i \omega'([p-1]))) \\ &= c(E(G, \omega \oplus_i \omega'([p-1]), V(G) \setminus \omega \oplus_i \omega'([p-1]))) \,.\end{aligned}$$

There are two cases to be analyzed. First, if $p \leq i$, then, by definition of $\omega \oplus_i \omega'$ we have $\omega \oplus_i \omega'([p-1]) = \omega([p-1])$. Therefore,

$$\mathrm{wcw}(G, c, \omega \oplus_i \omega', p) = c(E(G, \omega([p-1]), V(G) \setminus \omega([p-1]))) = \mathrm{wcw}(G, c, \omega, p) \leq \mathrm{wcw}(G, c, \omega) \,.$$

Secondly, if $p > i$, then we have

$$\begin{aligned}\mathrm{wcw}(G, c, \omega \oplus_i \omega', p) &= c(E(G, \omega \oplus_i \omega'([p-1]), V(G) \setminus \omega \oplus_i \omega'([p-1]))) \\ &\overset{(a)}{=} c(E(G, \omega([i]), V(G) \setminus \omega \oplus_i \omega'([p-1]))) \\ &\quad + c(E(G, (\omega \oplus_i \omega'([p-1])) \setminus \omega([i]), V(G) \setminus \omega \oplus_i \omega'([p-1]))) \\ &\overset{(b)}{\leq} \mathrm{wcw}(G, c, \omega, i+1) + \mathrm{wcw}(G[V(G) \setminus \omega([i])], c, \omega'^{V(G) \setminus \omega([i])}, p-i) \\ &\leq \mathrm{wcw}(G, c, \omega) + \mathrm{wcw}(G, c, \omega').\end{aligned}$$

For Equality (a), observe that given two disjoint subsets S and S' of $V(G)$, and a partition S_1, S_2 of S, we have $c(E(S, S')) = c(E(S_1, S')) + c(E(S_2, S'))$. Moreover, $\omega([i]) \subseteq \omega \oplus_i \omega'([p-1])$,

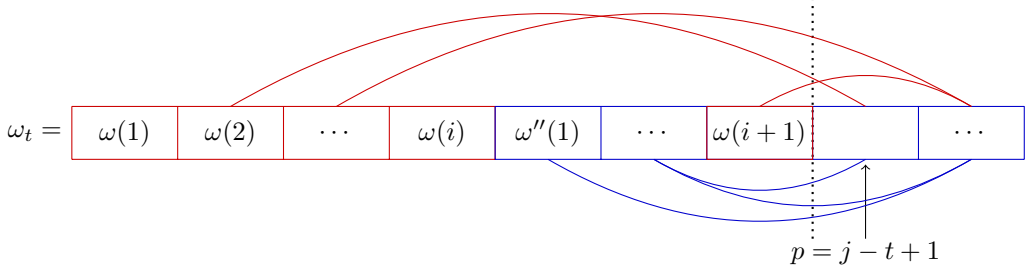

Figure 3: Illustration of the key part in Lemma 4, using the unweighted setting for simplicity. In this figure, $\omega'' = \omega'^{V \setminus \omega([i+1])}$. The red part of the linear order follows the linear order ω for the first $i+1$ elements, and the blue part of the linear order follows ω' for the remaining elements. Then, the set of edges crossing the cut at position $p = j - t + 1$ can be split in two, the set of edges that start from the red part and the set of edge that start from the blue part. The number of red edges is bounded by the cutwidth of ω and the number of blue edges is bounded by the cutwidth of ω'.

therefore, $(\omega \oplus_i \omega'([p-1])) \cap \omega([i]) = \omega([i])$ and $\{\omega([i]), (\omega \oplus_i \omega'([p-1])) \setminus \omega([i])\}$ is a partition of $\omega \oplus_i \omega'([p-1])$. Figure 2 gives a visual representation of the equality. This proves Equality (a). To understand Inequality (b), we need two arguments. As $\omega([i]) \subseteq \omega \oplus_i \omega'([p-1])$,

$$E(G, \omega([i]), V(G) \setminus (\omega \oplus_i \omega'([p-1]))) \subseteq E(G, \omega([i]), V(G) \setminus \omega([i]))\,,$$

which shows that the weight of the first set is upper-bounded by $\mathrm{wcw}(G, \omega, i+1)$. As the edges in $E(G, (\omega \oplus_i \omega'([p-1])) \setminus \omega([i]), V(G) \setminus (\omega \oplus_i \omega'([p-1])))$ only connect vertices with positions beyond i within $\omega \oplus_i \omega'$, after an index shift, we see that only the linear order ω' really matters, which explains the inequality

$$\begin{aligned} &c(E(G, (\omega \oplus_i \omega'([p-1])) \setminus \omega([i]), V(G) \setminus (\omega \oplus_i \omega'([p-1])))) \\ \leq\ &\mathrm{wcw}(G[V(G) \setminus \omega([i])], c, \omega'^{V(G) \setminus \omega([i])}, p-i)\,. \end{aligned}$$

For the last inequality, apply Lemma 2 to derive

$$\mathrm{wcw}(G[V(G) \setminus \omega([i])], c, \omega'^{V(G) \setminus \omega([i])}) \leq \mathrm{wcw}(G, c, \omega')\,.$$

As p is arbitrary, we have that $\mathrm{wcw}(G, c, \omega \oplus_i \omega') \leq \mathrm{wcw}(G, c, \omega) + \mathrm{wcw}(G, c, \omega') \leq 2k$ follows for each $i \in [n]$. □

Finally, we show that we can reconfigure the composition of two linear orders at position i, $\omega' \oplus_i \omega$, to the composition at position $i+1$, $\omega' \oplus_{i+1} \omega$, in bounded weighted cutwidth.

Lemma 4 *Let G be an n-vertex graph, let $c : E(G) \to \mathbb{N}_+$ be an edge weight function and let $\omega, \omega' : [n] \to V(G)$ be linear orders on $V(G)$ of weighted cutwidth at most k (with respect to c), and let $i \in \{0, \ldots, n-1\}$ be an integer. Then, $\omega \oplus_i \omega'$ can be reconfigured into $\omega \oplus_{i+1} \omega'$ in weighted cutwidth at most $\mathrm{wcw}(G, c, \omega) + \mathrm{wcw}(G, c, \omega') \leq 2k$.*

Proof: By Lemma 3, $\omega \oplus_i \omega'$ and $\omega \oplus_{i+1} \omega'$ have weighted cutwidth at most

$$\mathrm{wcw}(G, c, \omega) + \mathrm{wcw}(G, c, \omega') \leq 2k\,.$$

Let $j \in [n]$ such that $\omega \oplus_i \omega'(j) = \omega(i+1)$, i.e., j is the position of $\omega(i+1)$ in $\omega \oplus_i \omega'$. As for each $p \in [i]$, $\omega \oplus_i \omega'(p) = \omega \oplus_{i+1} \omega'(p) = \omega(p)$, we have $j > i$. Let us consider the following sequence of swaps:

$$\omega \oplus_i \omega' = \omega_0 \xrightarrow{j-1} \omega_1 \xrightarrow{j-2} \cdots \xrightarrow{i+1} \omega_{j-i-1} = \omega \oplus_{i+1} \omega'.$$

If $j = i+1$, this sequence is empty and $\omega \oplus_i \omega' = \omega \oplus_{i+1} \omega'$. At each step of this sequence, we swap $\omega(i+1)$ with its left neighbor. This brings $\omega(i+1)$ from position j to position $i+1$. By doing this, we transform $\omega \oplus_i \omega'$ into $\omega \oplus_{i+1} \omega'$. Observe that, at each step, $\omega(i+1)$ is moved one position to the left, therefore, we have $\omega_t(j-t) = \omega(i+1)$ for all $t \in [j-i-1]$.

Consider Figure 3 for an illustration of the key part of the following proof by induction. We are going to inductively show that each element ω_t in the sequence has weighted cutwidth upper-bounded by $\mathrm{wcw}(G, c, \omega) + \mathrm{wcw}(G, c, \omega') \leq 2k$. By Lemma 3,

$$\mathrm{wcw}(G, c, \omega_0) = \mathrm{wcw}(G, c, \omega \oplus_i \omega') \leq \mathrm{wcw}(G, c, \omega) + \mathrm{wcw}(G, c, \omega') \leq 2k\,,$$

which proves the induction basis. Let $t \in [j-i-1]$ and $p \in [n]$ be two integers. As induction hypothesis, we have

$$\mathrm{wcw}(G, c, \omega_{t-1}) \leq \mathrm{wcw}(G, c, \omega) + \mathrm{wcw}(G, c, \omega') \leq 2k\,.$$

If $p \leq j-t$ or $p > j-t+1$, then we have $\omega_{t-1}([p-1]) = \omega_t([p-1])$, so

$$\mathrm{wcw}(G, c, \omega_t, p) = \mathrm{wcw}(G, c, \omega_{t-1}, p) \leq \mathrm{wcw}(G, c, \omega) + \mathrm{wcw}(G, c, \omega') \leq 2k$$

by induction hypothesis. Otherwise, $p = j-t+1 \in \{i, \ldots, j\}$ (Figure 3) and we have

$$\begin{aligned}
\mathrm{wcw}(G, c, \omega_t, p) &= c(E(G, \omega_t([p-1]))) \\
&= c(E(G, \omega_t([p-1]), \{\omega_t(r) \,:\, r \geq p\})) \\
&= c(E(G, \omega_t([i] \cup \{p-1\}), \{\omega_t(r) \,:\, r \geq p\})) \\
&\quad + c(E(G, \{\omega_t(l) \,:\, i < l < p-1\}, \{\omega_t(r) \,:\, r \geq p\}))\,.
\end{aligned}$$

By definition of p, we have $\omega_t(p-1) = \omega_t(j-t)$ and by construction of ω_t, we have $\omega_t(j-t) = \omega(i+1)$. Therefore, we find

$$c(E(G, \omega_t([i] \cup \{p-1\}), \{\omega_t(r) \,:\, r \geq p\})) = c(E(G, \omega([i+1]), \{\omega_t(r) \,:\, r \geq j-t+1\}))\,.$$

As we are swapping $\omega(i+1)$ leftwards,

$$\{\omega_t(r) \,:\, r \geq j-t+1\} \subseteq \{\omega(r) \,:\, r \geq i+2\} = V(G) \setminus \omega([i+1])\,.$$

Again by definition of ω_t and p, the elements in $\{\omega_t(l) \,:\, i < l < p-1\}$ are ordered according to ω', which is also true for $\{\omega_t(r) \,:\, r \geq p\}$. More formally,

$$\{\omega_t(l) \,:\, i < l < p-1\} = \{\omega \oplus_i \omega'(l) \,:\, i+1 \leq l \leq p-2\} = \{\omega'^{V(G)\setminus\omega([i+1])}(l') \,:\, l' \leq p-2-i\}$$

and

$$\{\omega_t(r) \,:\, r \geq p\} = \{\omega \oplus_i \omega'(r) \,:\, r \geq p\} = \{\omega'^{V(G)\setminus\omega([i+1])}(r') \,:\, r' \geq p-i-1\}\,.$$

Therefore,

$$\begin{aligned}
\mathrm{wcw}(G, c, \omega_t, p) &\leq \mathrm{wcw}(G, c, \omega, i+2) + \mathrm{wcw}(G[V(G) \setminus \omega([i+1])], c, \omega'^{V(G)\setminus\omega([i+1])}, p-i-1) \\
&\leq \mathrm{wcw}(G, c, \omega) + \mathrm{wcw}(G[V(G) \setminus \omega([i+1])], c, \omega'^{V(G)\setminus\omega([i+1])}) \\
&\leq \mathrm{wcw}(G, c, \omega) + \mathrm{wcw}(G, c, \omega') \\
&\leq 2k.
\end{aligned}$$

To achieve the penultimate inequality, we again apply Lemma 2. □

Now, we are in the position to prove our first main theorem.

Proof: [Proof of Theorem 1] Consider the following sequence: $\omega = \omega' \oplus_0 \omega \to^* \omega' \oplus_1 \omega \to^* \cdots \to^* \omega' \oplus_n \omega = \omega'$. By Lemma 4, one can realize each step in weighted cutwidth at most $\mathrm{wcw}(G, c, \omega) + \mathrm{wcw}(G, c, \omega') \leq 2k$, which then also upper-bounds the whole reconfiguration sequence. □

We have proved, in Theorem 1, that one can always reconfigure a linear order ω, of weighted cutwidth at most k, into a linear order ω', of weighted cutwidth at most k, in weighted cutwidth at most $2k$. Now, we show that for every $k \in \mathbb{N}_+$, there exists a graph G and two linear orders ω and ω' of cutwidth k such that any reconfiguration sequence between ω and ω' needs linear orders of cutwidth at least $2k$. In other words, the bound given in Theorem 1 is sharp. Here, clearly considering the unweighted variant (i.e., all weights equal one) suffices.

Proposition 1 *For each $k \in \mathbb{N}$, there exists a sequence of graphs $(G_n)_{n \in \mathbb{N}_{\geq 3}}$ where for $n \in \mathbb{N}_{\geq 3}$, G_n has $n \cdot k$ vertices, and linear orders $\omega, \omega' : [n] \to V(G_n)$ such that $\mathrm{cw}(G_n, \omega) = \mathrm{cw}(G_n, \omega') = k$, but any reconfiguration sequence that transforms ω into ω' will have cutwidth of at least $2k$.*

Proof: For simplicity, we will start by giving a sequence of multigraphs with this property and we will see how to build simple graphs from them. Let $k, n \in \mathbb{N}$ with $n \geq 3$. Let $V(G_n) = [n]$ be the vertex set of a multigraph G_n with k edges between i and $i+1$ for each $i \in [n-1]$. Furthermore, let $\omega : [n] \to [n]$ be the identity and $\omega' : [n] \to [n]$ satisfy $\omega'(i) = n + 1 - i$ for $i \in [n]$, in other words, ω' is the reverse of ω. Then, $\mathrm{cw}(G_n, \omega) = \mathrm{cw}(G_n, \omega') = k$, but applying a swap to ω at any position will result in a linear order of cutwidth $2k$.

Now, to prove this result in the case of simple graphs, we will turn each graph G_n into a simple graph by replacing each edge in G with a path of length 2. In other words, we split each edge and add a new vertex in the middle. This is a simple edge subdivision. We call the vertices from G the main vertices and the added vertices the dummy vertices. To complete the construction, we have to extend ω and ω'. For this purpose, we put the dummy vertices on the path from i to $i+1$ for each $i \in [n]$ after (and before, respectively) i and before (and after, respectively) $i+1$ in ω (and ω', respectively). The order between the dummy vertices does not matter. Now it is easy to see that the cutwidth of ω and ω' is k. Any reconfiguration sequence that transforms ω into ω' needs to swap two main vertices at some point. Let i and $i+1$ be the two first main vertices to be swapped. If $i > 1$, then there exist k paths from i to $i+1$ and k paths from i to $i-1$. Those $2k$ paths are disjoint, therefore, the cut between i and $i+1$ has a size of at least $2k$. If $i = 1$ then $i + 1 < n$ and the same reasoning works for $i+1$ instead of i. □

Our results on weighted cutwidth reconfiguration are in themselves interesting to the community working on reconfiguration problems, but in the following, we prove a connection of unweighted cutwidth reconfiguration to the famous Graph Isomorphism problem, which shows a completely different facet of this particular reconfiguration problem.

4 String Rewriting System and Graph Isomorphism

Now, we use our result on the Bounded Cutwidth Order Reconfiguration problem to make a connection between two apparently unrelated computational problems, namely, the Reachability problem for a two-letter rewriting system and the Graph Isomorphism problem. To do so, we introduce a notion of graph decomposition based on cutwidth named *unit decomposition*.

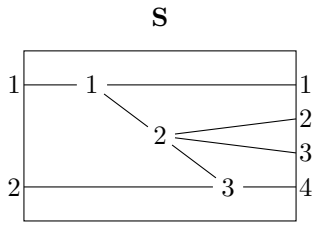

Figure 4: Slices are drawn as tiles. This figure depicts the slice $\mathbf{S} = (I, C, O, E)$ where $I = \{(0,1),(0,2)\}$, $C = \{1,2,3\}$, $O = \{(1,1),(1,2),(1,3),(1,4)\}$ and $E = \{\{(0,1),1\},\{(0,2),3\},\{1,2\},\{2,3\},\{1,(1,1)\},\{2,(1,2)\},\{2,(1,3)\},\{3,(1,4)\}\}$. We omit the first element of the pair for frontier vertices and use the following convention. The in-frontier vertices are on the left of the tile and the out-frontier vertices are on the right of the tile. If the frontier vertices are not explicitly mentioned in the drawing, we assume that frontier vertices are ordered from top to bottom as in this drawing. This convention simplifies future drawings.

In a unit decomposition, the graph is split into small pieces called *slices*. If a graph has bounded cutwidth, it can be decomposed into a unit decomposition of bounded width. Using the slices as an alphabet, we can see a graph as a word. Then, we introduce a rewriting system over this alphabet that preserve isomorphism. Our main result (Theorem 2) of this section states that two graphs are isomorphic if and only if, their unit decompositions are reachable in this rewriting system.

4.1 Slice Rewriting System

We start by describing the notion of slices and unit decompositions. Intuitively, a slice is a piece of graphs. Given two compatible slices, we can combine them using an operation called the gluing operation to build a new bigger slice. After gluing enough slices, we end up with a graph. A unit decomposition is the representation of a graph as a sequence of compatible slices. Then, using slices as letters, we define an alphabet and a rewriting system over unit decompositions.

Slices. Let $\mathcal{I} = \{[a] \ : \ a \in \mathbb{N}\}$ denote the set of intervals of the form $[a] = \{1, \ldots, a\}$ for $a \in \mathbb{N}$ (recall that $[0] = \emptyset$). We let $\mathcal{I}_0 = \{\{0\} \times [a] : [a] \in \mathcal{I}\}$, and $\mathcal{I}_1 = \{\{1\} \times [a] : [a] \in \mathcal{I}\}$. A *slice* $\mathbf{S} = (I, C, O, E)$ is a (multi-)graph where the vertex set $V = I \dot\cup C \dot\cup O$ is partitioned into an *in-frontier* $I \in \mathcal{I}_0$, a *center* $C \in \mathcal{I}$, and an *out-frontier* $O \in \mathcal{I}_1$, and E is a multiset of unordered pairs from $I \cup C \cup O$ in such a way that vertices of $I \cup O$ have degree exactly 1, there is no edge between any two vertices in I, and no edge between any two vertices in O. We depict slices as in Figure 4. We define slices using multigraphs, as the gluing operation, defined below, can take slices which are simple graphs, and create a slice that is a multigraph (see Figure 5). Given a slice $\mathbf{S}$, we define $I(\mathbf{S})$ as the in-frontier of $\mathbf{S}$, $O(\mathbf{S})$ as the out-frontier of $\mathbf{S}$, and $C(\mathbf{S})$ as the center vertices of $\mathbf{S}$. The *width* of a slice $\mathbf{S}$ is defined as $\mathbf{w}(S) = \max(|I(\mathbf{S})|, |O(\mathbf{S})|)$.

Gluing Slices. A slice $\mathbf{S}_1 = (I_1, C_1, O_1, E_1)$ can be glued to $\mathbf{S}_2 = (I_2, C_2, O_2, E_2)$ if for some interval $[a] \in \mathcal{I}$, $O_1 = \{1\} \times [a]$ and $I_2 = \{0\} \times [a]$. In this case, the gluing gives rise to the slice $\mathbf{S}_1 \circ \mathbf{S}_2 = (I_1, C_1 \cup (|C_1| \oplus C_2), O_2, E)$ where $|C_1| \oplus C_2$ is a shift of the elements in C_2 by $|C_1|$, more

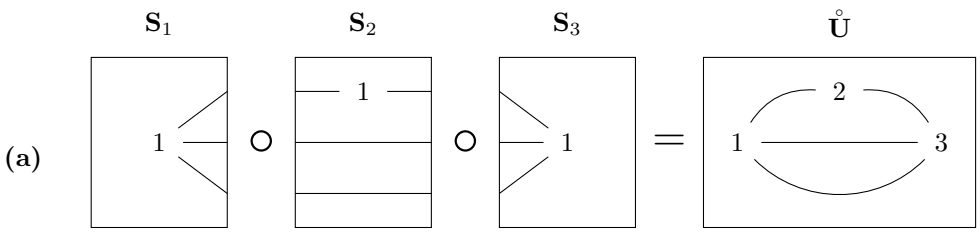

Figure 5: Slice associated with the unit decomposition $\mathbf{U} = \mathbf{S}_1\mathbf{S}_2\mathbf{S}_3$. The resulting slice does not have any vertex in its frontier. It can therefore be seen as a multigraph on 3 vertices.

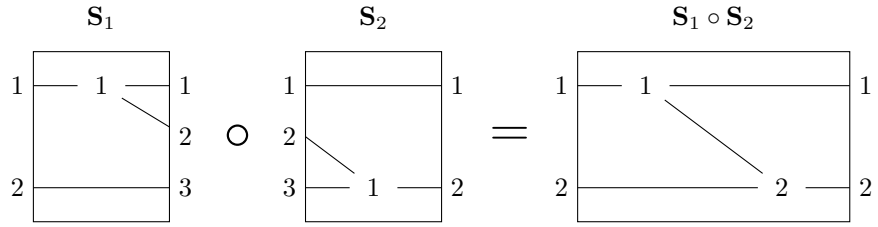

Figure 6: Gluing of two slices $\mathbf{S}_1$ and $\mathbf{S}_2$. The gluing operation is a way to merge two slices into one. In this example, the edge from the center vertex 1 from $\mathbf{S}_1$ to the out-frontier vertex $(1, 2)$ is stitched to the edge from the in-frontier vertex $(0, 2)$ to the center vertex 1 from $\mathbf{S}_2$ to form the edge between the center vertices 1 and 2 in $\mathbf{S}_1 \circ \mathbf{S}_2$. The stitching of edges is done following the order of the frontier vertices.

formally $|C_1| \oplus C_2 = [|C_1| + |C_2|] \setminus [|C_1|] = \{|C_1| + 1, |C_1| + 2, \ldots, |C_1| + |C_2|\}$,

$$\begin{aligned} E = &\{\{x, y\} \in E_1 \ : \ x, y \in I_1 \cup C_1\} \\ &\cup \{\{x, y + |C_1|\} \ : \ \{x, y\} \in E_2 \wedge x \in O_2 \wedge y \in C_2\} \\ &\cup \{\{x + |C_1|, y + |C_1|\} \ : \ \{x, y\} \in E_2 \wedge x, y \in C_2\} \\ &\cup \{\{x, y\} \ : \ \exists i, \{x, (1, i)\} \in E_1 \wedge y \in O_2 \wedge \{(0, i), y\} \in E_2\} \\ &\cup \{\{x, y\} \ : \ \exists i, \{x, (1, i)\} \in E_1 \wedge y \in |C_1| \oplus C_2 \wedge \{(0, i), y - |C_1|\} \in E_2\}. \end{aligned}$$

Note that the gluing operation is associative. Therefore, we will not write parentheses for the gluing of more than two slices. Figure 6 illustrates the gluing of two slices.

Unit Slices and Unit Decompositions. We say that a slice is a *unit slice* if it has a unique vertex in its center. A *unit decomposition* is a sequence $\mathbf{U} = \mathbf{S}_1\mathbf{S}_2 \ldots \mathbf{S}_n$, where $\mathbf{S}_i$ are unit slices and $\mathbf{S}_i \circ \mathbf{S}_{i+1}$ is well defined for each $i \in [n-1]$. The *slice associated with a unit decomposition* $\mathbf{U}$ is defined as $\mathring{\mathbf{U}} = \mathbf{S}_1 \circ \mathbf{S}_2 \circ \ldots \circ \mathbf{S}_n$ (see Figure 7 for an example). Note that if the in-frontier of $\mathbf{S}_1$ and the out-frontier of $\mathbf{S}_n$ are empty, then $\mathring{\mathbf{U}}$ is just a multigraph with vertex set $[n]$ (see Figure 5). For each $k \in \mathbb{N}$, we define the alphabet $\mathbf{\Sigma}(k)$ as the set of all unit slices of width at most k. We now prove that $|\mathbf{\Sigma}(k)|$ is upper-bounded by a function in k.

Proposition 2 *$\mathbf{\Sigma}(k)$ is finite, more precisely, $|\mathbf{\Sigma}(k)| = \mathcal{O}((k+1)^2 \cdot \sum_{i=0}^{k} \binom{k}{i} \frac{k!}{i!})$.*

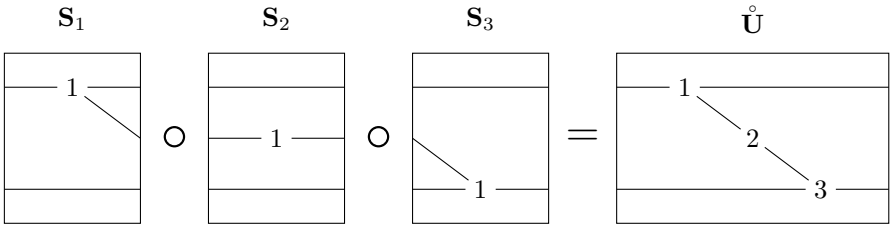

Figure 7: Slice associated with the unit decomposition $\mathbf{U} = \mathbf{S}_1\mathbf{S}_2\mathbf{S}_3$. The gluing operation is associative, therefore parentheses are not needed.

Proof: In a unit slice of width at most k, the in-frontier and out-frontier can have between 0 and k vertices and there is only one center vertex. Therefore, there are $(k+1)^2$ configurations for the vertices of a slice. By definition, vertices of the in-frontier and the out-frontier have degrees exactly 1, vertices in the in-frontier can be connected to the center vertex or an out-frontier vertex and there is no self-loop. Therefore, fixing the connectivity of the in-frontier vertices defines the full unit slice, because, if an out-frontier vertex is not connected to an in-frontier vertex, then it must be connected to the center vertex. Once the number of vertices in the frontiers is chosen, there can be between 0 and k vertices from the in-frontier connected to the center vertex. Let $0 \leq i \leq k$ be the number of such vertices. If i vertices are connected to the center vertex, then the rest of the vertices must be connected to the out-frontier; there are at most $\frac{k!}{i!}$ ways of doing this. There are at most $\binom{k}{i}$ subsets of size i in the in-frontier. Hence, there are at most $\sum_{i=0}^{k} \binom{k}{i} \frac{k!}{i!}$ ways to connect the vertices in the in-frontier. □

We let $\mathbf{\Sigma}(k)^{\circledast}$ denote the set of all unit decompositions over $\mathbf{\Sigma}(k)$.

The order of the unit slices in a unit decomposition $\mathbf{U} = \mathbf{S}_1\mathbf{S}_2\dots\mathbf{S}_n$ induces a linear order $\omega_{\mathbf{U}}$ on the center vertices of the slice $\mathring{\mathbf{U}}$. We extend this linear order to all the vertices of $\mathring{\mathbf{U}}$ by putting the vertices in the in-frontier first, then the center vertices and, finally, the vertices in the out-frontier. More formally, the linear order defined by $\mathbf{U}$ sets $\omega_{\mathbf{U}}(i) = (0,i)$ for each $(0,i) \in I(\mathbf{S}_1)$, $\omega_{\mathbf{U}}(|I(\mathbf{S}_1)| + i) = i$ for each $i \in \{1,\dots,n\}$ and $\omega_{\mathbf{U}}(|I(\mathbf{S}_1)| + n + i) = (1,i)$ for each $(1,i) \in O(\mathbf{S}_n)$.

Given a unit decomposition $\mathbf{U} = \mathbf{S}_1\mathbf{S}_2\dots\mathbf{S}_n$ in $\mathbf{\Sigma}(k)^{\circledast}$, we let $\mathbf{w}(\mathbf{U}) = \max_{i\in[n]} \mathbf{w}(\mathbf{S}_i)$ be the *width* of $\mathbf{U}$. Recall that $\mathbf{w}(\mathbf{S}_i) = \max(|I(\mathbf{S}_i)|, |O(\mathbf{S}_i)|)$.

Equivalence of Slices. Let $\mathbf{S}_1 = (I_1, C_1, O_1, E_1)$ and $\mathbf{S}_2 = (I_2, C_2, O_2, E_2)$ be two slices. We say that $\mathbf{S}_1$ is *equivalent* to $\mathbf{S}_2$, denoted by $\mathbf{S}_1 \sim \mathbf{S}_2$, if and only if $I_1 = I_2$, $O_1 = O_2$, $C_1 = C_2$, and there is an isomorphism ϕ from $\mathbf{S}_1$ to $\mathbf{S}_2$ such that the restriction of ϕ to I_1 and O_1 is the identity function. In other words, $\mathbf{S}_1$ and $\mathbf{S}_2$ are equivalent if they are equal up to the renaming of the center vertices.

We let $\mathcal{R}(k) \subseteq \mathbf{\Sigma}(k)^2 \times \mathbf{\Sigma}(k)^2$ be the set of all rewriting rules of the form $\mathbf{S}_1\mathbf{S}_2 \to \mathbf{S}'_1\mathbf{S}'_2$ such that $\mathbf{S}_1 \circ \mathbf{S}_2 \sim \mathbf{S}'_1 \circ \mathbf{S}'_2$. By Proposition 2, $\mathcal{R}(k)$ is finite. We call two unit decompositions $\mathbf{U}, \mathbf{U}' \in \mathbf{\Sigma}(k)^{\circledast}$ *locally* $\mathcal{R}(k)$*-equivalent*, and denote this fact by $\mathbf{U} \overset{k}{\sim} \mathbf{U}'$, if there exist $\mathbf{W}, \mathbf{W}' \in \mathbf{\Sigma}(k)^{\circledast}$ and $\mathbf{S}_1, \mathbf{S}'_1, \mathbf{S}_2, \mathbf{S}'_2 \in \mathbf{\Sigma}(k)$ with $\mathbf{S}_1 \circ \mathbf{S}_2 \sim \mathbf{S}'_1 \circ \mathbf{S}'_2$ such that $\mathbf{U} = \mathbf{W}\mathbf{S}_1\mathbf{S}_2\mathbf{W}'$ and $\mathbf{U}' = \mathbf{W}\mathbf{S}'_1\mathbf{S}'_2\mathbf{W}'$ (see Figure 8). In other words, $\mathbf{U}$ is locally $\mathcal{R}(k)$-equivalent to $\mathbf{U}'$ if $\mathbf{U}$ can be rewritten in one step into $\mathbf{U}'$ using a rule from $\mathcal{R}(k)$.

We let $\overset{k}{\equiv} \subseteq \mathbf{\Sigma}(k)^{\circledast} \times \mathbf{\Sigma}(k)^{\circledast}$ be the equivalence relation defined on unit decompositions by taking the reflexive, symmetric and transitive closure of $\overset{k}{\sim}$. If $\mathbf{U} \overset{k}{\equiv} \mathbf{U}'$, then we say that $\mathbf{U}'$

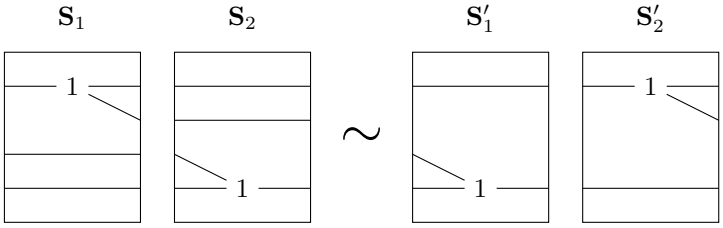

Figure 8: Local equivalence. $\mathbf{S}_1\mathbf{S}_2$ is (locally) $\mathcal{R}(4)$-equivalent to $\mathbf{S}'_1\mathbf{S}'_2$.

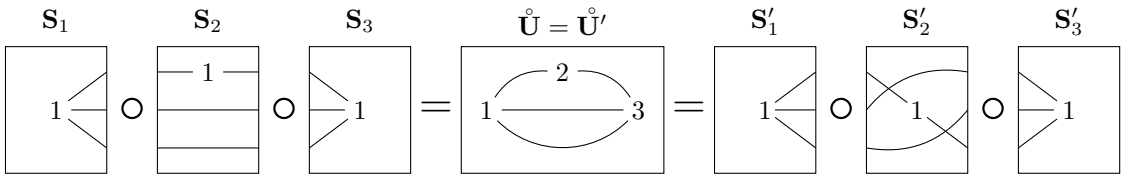

Figure 9: The unit decomposition $\mathbf{U} = \mathbf{S}_1\mathbf{S}_2\mathbf{S}_3$ is a twisting of the unit decomposition $\mathbf{U}' = \mathbf{S}'_1\mathbf{S}'_2\mathbf{S}'_3$. Note that $\mathbf{S}_2 \circ \mathbf{S}_3 = \mathbf{S}'_2 \circ \mathbf{S}'_3$. Note that if we let π be the permutation that sets $\pi(1) = 2$, $\pi(2) = 3$ and $\pi(3) = 1$, then $\mathbf{S}'_2$ is obtained by permuting the out-frontier of $\mathbf{S}_2$ according to π and $\mathbf{S}'_3$ is obtained by permuting the in-frontier of $\mathbf{S}_3$ according to π.

is $\mathcal{R}(k)$-*equivalent* to $\mathbf{U}$. We note that if $\mathbf{U}$ is a unit decomposition in $\mathbf{\Sigma}(k)^{\circledast}$ then any unit decomposition $\mathbf{U}'$ that is $\mathcal{R}(k)$-equivalent to $\mathbf{U}$ is also a unit decomposition in $\mathbf{\Sigma}(k)^{\circledast}$. We also note that there may exist unit decompositions in $\mathbf{\Sigma}(k)^{\circledast}$ that are not $\mathcal{R}(k)$-equivalent but that are $\mathcal{R}(k')$-equivalent for some $k' > k$.

Twisting. Let $\mathbf{U} = \mathbf{S}_1\mathbf{S}_2 \cdots \mathbf{S}_n$ and $\mathbf{U}' = \mathbf{S}'_1\mathbf{S}'_2 \cdots \mathbf{S}'_n$ be two unit decompositions. We say that $\mathbf{U}$ is a *twisting* of $\mathbf{U}'$ if $\mathring{\mathbf{U}} = \mathring{\mathbf{U}}'$. Note that we are not equating slices up to isomorphism. In other words, we are really requiring that the slices $\mathring{\mathbf{U}}$ and $\mathring{\mathbf{U}}'$ are syntactically identical.

Let $\mathbf{S}_1$ and $\mathbf{S}_2$ be unit slices in $\mathbf{\Sigma}(k)$ such that the out-frontier of $\mathbf{S}_1$ and the in-frontier of $\mathbf{S}_2$ have k' vertices for some $k' \leq k$. Given a permutation $\pi : [k'] \to [k']$, let $\mathbf{S}_1^{\pi}$ be the slice obtained by renaming each vertex $(1, i)$ in the out-frontier of $\mathbf{S}_1$ to $(1, \pi(i))$, and let ${}^{\pi}\mathbf{S}_2$ be the slice obtained by renaming each vertex $(0, i)$ in the in-fronter of $\mathbf{S}_2$ to $(0, \pi(i))$. Then, it should be clear that $\mathbf{S}_1 \circ \mathbf{S}_2 = \mathbf{S}_1^{\pi} \circ {}^{\pi}\mathbf{S}_2$. In other words, $\mathbf{S}_1^{\pi}{}^{\pi}\mathbf{S}_2$ is a twisting of $\mathbf{S}_1\mathbf{S}_2$. Additionally, for each two unit slices $\mathbf{S}'_1$ and $\mathbf{S}'_2$ such that $\mathbf{S}'_1\mathbf{S}'_2$ is a twisting of $\mathbf{S}_1\mathbf{S}_2$ ($\mathbf{S}_1 \circ \mathbf{S}_2 = \mathbf{S}'_1 \circ \mathbf{S}'_2$), it should be clear that there is some permutation π such that $\mathbf{S}'_1 = \mathbf{S}_1^{\pi}$ and $\mathbf{S}'_2 = {}^{\pi}\mathbf{S}_2$. Note also that for every two such slices $\mathbf{S}'_1$ and $\mathbf{S}'_2$, the rewriting rule $\mathbf{S}_1\mathbf{S}_2 \to \mathbf{S}'_1\mathbf{S}'_2$ belongs to $\mathcal{R}(k)$. This implies that if a unit decomposition $\mathbf{U} = \mathbf{S}_1\mathbf{S}_2 \ldots \mathbf{S}_n$ is a twisting of a unit decomposition $\mathbf{U}' = \mathbf{S}'_1\mathbf{S}'_2 \ldots \mathbf{S}'_n$, then $\mathbf{U}$ and $\mathbf{U}'$ are $\mathcal{R}(k)$-equivalent and can be transformed into each other by applying a sequence of rewriting rules that "twists" for each $i \in [n-1]$ the out-frontier of $\mathbf{S}_i$ and the in-frontier of $\mathbf{S}_{i+1}$ according to some permutation π_i. This process is illustrated in Figure 9.

Proposition 3 (Twisting) *Let* $\mathbf{U} = \mathbf{S}_1\mathbf{S}_2 \cdots \mathbf{S}_n$ *and* $\mathbf{U}' = \mathbf{S}'_1\mathbf{S}'_2 \cdots \mathbf{S}'_n$ *be two unit decompositions in* $\mathbf{\Sigma}(k)^{\circledast}$ *such that* $\mathbf{U}$ *is a twisting of* $\mathbf{U}'$. *Then,* $\mathbf{U}$ *can be transformed into* $\mathbf{U}'$ *by the application of* $n-1$ *rewriting rules from* $\mathcal{R}(k)$.

4.2 Graph Isomorphism as a Rewriting System

Now, we are ready to see the connection between the Graph Isomorphism problem and the Reachability problem in $\mathcal{R}(k)$. First, we show that a graph G has cutwidth at most k if and only if it has a unit decompositions of width at most k (Proposition 4 and Proposition 5). Then, we show that the rewriting system $\mathcal{R}(k)$ preserves isomorphism (Lemma 5). Building on those results, we show the connection between the Reachability problem in $\mathcal{R}(2k)$ and the Graph Isomorphism for graphs of cutwidth at most k.

Intuitively, a unit decomposition $\mathbf{U}$ is a decomposition of the graph $\mathring{\mathbf{U}}$. This decomposition induces an ordering of the vertices of $\mathring{\mathbf{U}}$. The size of the common frontier between two neighbouring slices in $\mathbf{U}$ corresponds to the size of the cut at the same position in $\mathring{\mathbf{U}}$ with respect to $\omega_{\mathbf{U}}$. Therefore, $\mathbf{U}$ induces an ordering of $\mathring{\mathbf{U}}$ of cutwidth $\mathbf{w}(\mathbf{U})$. This idea is formalized by the following proposition.

Proposition 4 *Let $k \in \mathbb{N}$, and $\mathbf{U} = \mathbf{S}_1\mathbf{S}_2\dots\mathbf{S}_n$ be a unit decomposition in $\mathbf{\Sigma}(k)^{\circledast}$, and $\omega_{\mathbf{U}}$ be the linear order induced by $\mathbf{U}$ on $\mathring{\mathbf{U}}$. Then, $\mathrm{cw}(\mathring{\mathbf{U}}, \omega_{\mathbf{U}}) = \mathbf{w}(\mathbf{U})$.*

Proof: This follows by noticing that each vertex in $I(\mathbf{S}_1)$ has degree 1 and there is no edge between vertices in $I(\mathbf{S}_1)$, therefore for each position p in $\{1, \dots, |I(\mathbf{S}_1)| + 1\}$, $\mathrm{cw}(\mathring{\mathbf{U}}, \omega_{\mathbf{U}}, p) \leq |I(\mathbf{S}_1)|$ and $\mathrm{cw}(\mathring{\mathbf{U}}, \omega_{\mathbf{U}}, |I(\mathbf{S}_1)|+1) = |I(\mathbf{S}_1)|$, in the same way, we have for each p in $\{|I(\mathbf{S}_1)|+n+1, \dots, |I(\mathbf{S}_1)|+n+|O(\mathbf{S}_n)|\}$, $\mathrm{cw}(\mathring{\mathbf{U}}, \omega_{\mathbf{U}}, p) \leq |O(\mathbf{S}_n)|$ and $\mathrm{cw}(\mathring{\mathbf{U}}, \omega_{\mathbf{U}}, |I(\mathbf{S}_1)| + n + 1) = |O(\mathbf{S}_n)|$, and for each $p \in \{|I(\mathbf{S}_1)| + 2, \dots, |I(\mathbf{S}_1)| + n\}$, $\mathrm{cw}(\mathring{\mathbf{U}}, \omega_{\mathbf{U}}, p) = |O(\mathbf{S}_{p-|I(\mathbf{S}_1)|-1})| = |I(\mathbf{S}_{p-|I(\mathbf{S}_1)|})|$. □

The relation between cutwidth and unit decomposition is valid in both directions. The following proposition states the reverse direction compared to Proposition 4.

Proposition 5 *Let G be an n-vertex graph and ω be a linear order on the vertices of G of cutwidth k. Then, we can construct in time $\mathcal{O}(kn)$ a unit decomposition $\mathbf{U}$ such that $\omega = \omega_{\mathbf{U}}$.*

Proof: We will do this construction by first drawing the graph G in the plane. G does not need to be planar for this construction to work. First, we will place the vertices of G on a straight line L isomorphic to $\mathbb{R}$. The i-th vertex of G with respect to the linear order ω is placed at the coordinate i on the line. Then, edges are drawn as curves between their endpoints. Now, we will draw $n + 1$ lines perpendicular to L at coordinates $\{-0.5, 0.5, 1.5, \dots, n - 0.5, n + 0.5\}$. We call these lines *cut-lines*. The cutwidth of ω is k, therefore each cut-line intersects at most k edges in the drawing of G. We put a vertex at the intersection of a cut-line and an edge. The graph between two consecutive cut-lines defines a unit slice of width at most k. Taking all those slices in the order induced by ω on the line L gives a unit decomposition $\mathbf{U}$ of width k such that $\omega = \omega_{\mathbf{U}}$. Figure 10 illustrates this construction. □

Now that we have defined the rewriting system, we are ready to show the connection between the rewriting system $\mathcal{R}(k)$ and the graph isomorphism problem. This connection is formalized in Theorem 2. The next lemma shows one direction in this connection.

Lemma 5 *Let $k \in \mathbb{N}$ and $\mathbf{U}$ and $\mathbf{U}'$ be unit decompositions in $\mathbf{\Sigma}(k)^{\circledast}$. If $\mathbf{U}$ is $\mathcal{R}(k)$-equivalent to $\mathbf{U}'$, then $\mathring{\mathbf{U}}$ is isomorphic to $\mathring{\mathbf{U}}'$.*

Proof: It is enough to show that if $\mathbf{U}$ can be transformed into $\mathbf{U}'$ in one $\mathcal{R}(k)$-rewriting step then $\mathring{\mathbf{U}}$ is isomorphic to $\mathring{\mathbf{U}}'$. Therefore, assume that $\mathbf{U} \to \mathbf{U}'$. Then, there exist unit decompositions

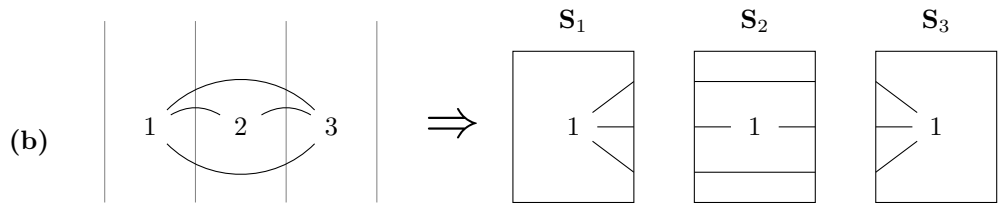

Figure 10: Slicing of the graph G on the left into a unit decomposition $\mathbf{U}$ on the right.

$\mathbf{W}, \mathbf{W}' \in \mathbf{\Sigma}(k)^{\circledast}$ and a rewriting rule $\mathbf{S}_1\mathbf{S}_2 \to \mathbf{S}'_1\mathbf{S}'_2$ in $\mathcal{R}(k)$ such that $\mathbf{U} = \mathbf{W}\mathbf{S}_1\mathbf{S}_2\mathbf{W}'$ and $\mathbf{U}' = \mathbf{W}\mathbf{S}'_1\mathbf{S}'_2\mathbf{W}'$. Since $\mathbf{S}_1 \circ \mathbf{S}_2 \sim \mathbf{S}'_1 \circ \mathbf{S}'_2$, we have an isomorphism φ from $\mathbf{S}_1 \circ \mathbf{S}_2$ to $\mathbf{S}'_1 \circ \mathbf{S}'_2$ that acts as the identity map on frontier vertices. This implies that $\mathring{\mathbf{U}} = \mathring{\mathbf{W}} \circ \mathbf{S}_1 \circ \mathbf{S}_2 \circ \mathring{\mathbf{W}}'$ is isomorphic to $\mathring{\mathbf{U}}' = \mathring{\mathbf{W}} \circ \mathbf{S}'_1 \circ \mathbf{S}'_2 \circ \mathring{\mathbf{W}}'$. □

An interesting question is whether, for each $k \in \mathbb{N}$, there is some $k' \in \mathbb{N}$ such that any two unit decompositions $\mathbf{U}$ and $\mathbf{U}'$ in $\mathbf{\Sigma}(k)$ are $\mathcal{R}(k')$-equivalent if and only if $\mathring{\mathbf{U}}$ is isomorphic to $\mathring{\mathbf{U}}'$. The answer turns out to be yes, as shown in Theorem 2 below.

Theorem 2 *Let $\mathbf{U}$ and $\mathbf{U}'$ be unit decompositions in $\mathbf{\Sigma}(k)^{\circledast}$. Then, $\mathring{\mathbf{U}}$ is isomorphic to $\mathring{\mathbf{U}}'$ if and only if $\mathbf{U}$ and $\mathbf{U}'$ are $\mathcal{R}(2k)$-equivalent.*

Proof: Let $\mathbf{U} = \mathbf{S}_1\mathbf{S}_2 \cdots \mathbf{S}_n$ and $\mathbf{U}' = \mathbf{S}'_1\mathbf{S}'_2 \cdots \mathbf{S}'_n$. Suppose that $\mathbf{U}$ and $\mathbf{U}'$ are $\mathcal{R}(2k)$-equivalent. Then, by Lemma 5, $\mathring{\mathbf{U}}$ is isomorphic to $\mathring{\mathbf{U}}'$.

For the converse direction, suppose that $\mathring{\mathbf{U}}$ is isomorphic to $\mathring{\mathbf{U}}'$ and let φ be an isomorphism from $\mathring{\mathbf{U}}$ to $\mathring{\mathbf{U}}'$. We show that $\mathbf{U}$ and $\mathbf{U}'$ are $\mathcal{R}(2k)$-equivalent.

Given a position $i \in [n-1]$ in the unit decomposition $\mathbf{U}$, a *swap* between $\mathbf{S}_i$ and $\mathbf{S}_{i+1}$ is a rewriting rule in $\mathcal{R}(k')$ for some k' that rewrites $\mathbf{U}$ into the unit decomposition

$$\mathbf{U}_i = \mathbf{S}_1\mathbf{S}_2 \cdots \mathbf{S}_{i-1}\mathbf{S}''_i\mathbf{S}''_{i+1}\mathbf{S}_{i+2} \cdots \mathbf{S}_n$$

such that, the function $\psi : [n] \to [n]$ that sets $\psi(p) = p$ for all $p \notin \{i, i+1\}$, $\psi(i) = i+1$ and $\psi(i+1) = i$ is an isomorphism from $\mathring{\mathbf{U}}$ to $\mathring{\mathbf{U}}_i$.

Intuitively, we swap the center vertex of $\mathbf{S}_i$ with the center vertex of $\mathbf{S}_{i+1}$. Note that, because of the twisting of the frontier, there may be several rewriting rules corresponding to such a swap. Now, a swap in the unit decomposition $\mathbf{U}$ corresponds to a swap in $\omega_{\mathbf{U}}$ as defined for linear orders in Section 2. The isomorphism φ defines a transformation of $\omega_{\mathbf{U}}$ into $\omega_{\mathbf{U}'}$.

By Proposition 4, $\mathrm{cw}(\mathring{\mathbf{U}}, \omega_{\mathbf{U}}) \leq k$ and $\mathrm{cw}(\mathring{\mathbf{U}}', \omega_{\mathbf{U}'}) \leq k$. Now, our result in Section 3 can be used for the slice rewriting system $\mathcal{R}(2k)$. More precisely, it follows from Theorem 1 that we can transform $\omega_{\mathbf{U}}$ into $\omega_{\mathbf{U}'}$ by a sequence of $\mathcal{O}(n^2)$ swaps and at each step, the cutwidth is at most $2k$. By using the rewriting rules from $\mathcal{R}(2k)$, we can replicate these swaps into the unit decomposition $\mathbf{U}$, obtaining in this way a unit decomposition $\mathbf{U}''$ such that $\omega_{\mathbf{U}''} = \omega_{\mathbf{U}'}$. Since $\mathring{\mathbf{U}}'' = \mathring{\mathbf{U}}'$, we have that $\mathbf{U}''$ is a twisting of $\mathbf{U}'$. Therefore, it follows from Proposition 3 that $\mathbf{U}''$ can be further transformed into $\mathbf{U}'$ by applying a sequence of $n-1$ rewriting rules from $\mathcal{R}(k) \subseteq \mathcal{R}(2k)$.

Hence, $\mathbf{U}$ can be rewritten into $\mathbf{U}''$ by applying $\mathcal{O}(n^2)$ rewriting rules from $\mathcal{R}(2k)$. □

Theorem 2 allows us to establish connections between the graph isomorphism problem for graphs of cutwidth at most k and the reachability problem in $\mathcal{R}(2k)$.

Theorem 3 ([20]) *Let G be an n-vertex graph of cutwidth k. We can compute a linear order ω of the vertices of G of cutwidth k in time $2^{\mathcal{O}(k^2 \log k)} \cdot n$.*

Theorem 4 *Graph isomorphism for n-vertex graphs of cutwidth at most k can be reduced in time $2^{\mathcal{O}(k^2 \log k)} \cdot n$ to $\mathcal{R}(2k)$-reachability.*

Proof: Given n-vertex graphs G and G' of cutwidth at most k, we first compute in time $2^{\mathcal{O}(k^2 \log k)} \cdot n$ linear orders ω and ω' of the vertex sets of G and G', respectively, of cutwidth at most k. Then, from Proposition 5, we construct unit decompositions $\mathbf{U}$ and $\mathbf{U}'$ such that $\omega_{\mathbf{U}} = \omega$, $\omega_{\mathbf{U}'} = \omega'$, G is isomorphic to $\mathring{\mathbf{U}}$ and G' is isomorphic to $\mathring{\mathbf{U}}'$. By Proposition 5, those decompositions belong to $\mathbf{\Sigma}(k)^{\circledast}$. By Theorem 2, we have that $\mathring{\mathbf{U}}$ and $\mathring{\mathbf{U}}'$ are isomorphic if and only if $\mathbf{U}$ and $\mathbf{U}'$ are $\mathcal{R}(2k)$-equivalent. □

5 Order Reconfiguration Parameterized by Vertex Separation Number

In this section, we show that the techniques employed in Section 3 for total orders of bounded cutwidth can be generalized to the context of orders of bounded vertex-separation number (Theorem 5). We consider that this generalization may be of independent interest in the theory of reconfiguration since vertex separation number is a width measure for graphs that is strictly more expressive than cutwidth.

Let G be an n-vertex graph with vertex set $V(G)$ and edge set $E(G)$. Given sets $S, S' \subseteq V(G)$, we let $V(G, S, S') = \{u \in S \ : \ \exists v \in S' \colon \{u, v\} \in E(G)\}$ be the set of vertices in S that are adjacent to some vertex in S'. As a special case, we define $V(G, S) = V(G, S, V(G) \setminus S)$. We will often make use of the following two properties without explicit mention.

- **Monotonicity property:** If $T \subseteq S$ and $T' \subseteq S'$, then $V(G, T, T') \subseteq V(G, S, S')$.
- **Linearity property:** If $\{S_1, S_2\}$ is a partition of S, then $\{V(G, S_1, S'), V(G, S_2, S')\}$ is a partition of $V(G, S, S')$.

Definition 2 (Vertex Separation Number) *Let G be an n-vertex undirected graph with vertex set $V(G)$ and edge set $E(G)$. Let $\omega : [n] \to V(G)$ be a linear order on the vertices of G. For each $p \in [n]$, we let*

$$\mathrm{vsn}(G, \omega, p) = |V(G, \omega([p-1]))| = |\{l \in [p-1] \ : \ \exists r \geq p \text{ such that } \{\omega(l), \omega(r)\} \in E(G)\}|.$$

The vertex separation number *of the linear order ω is defined as $\mathrm{vsn}(G, \omega) = \max_{p \in [n]} \mathrm{vsn}(G, \omega, p)$. Finally, the* vertex separation number *of G is defined as $\mathrm{vsn}(G) = \min_{\omega} \mathrm{vsn}(G, \omega)$, where ω ranges over all linear orders on the vertex set V.*

By introducing vertex weights $c : V(G) \to \mathbb{N}_+$, we can again define a weighted variant, called *weighted vertex separation number*, or $\mathrm{wvsn}(G, c, \omega)$. We refrain from giving formal details here, but the following discussions are also valid in this more general case.

For each $k \in \mathbb{N}$ and each n-vertex graph G, let $\mathrm{VSN}(G, k) = \{\omega : [n] \to V(G) \ : \ \mathrm{vsn}(G, \omega) \leq k\}$ be the set of linear orders of $V(G)$ of vertex separation number at most k. We say that ω can be *reconfigured* into ω' in vertex separation number at most k if there is a *reconfiguration sequence* $\omega = \omega_0 \xrightarrow{i_1} \omega_1 \xrightarrow{i_2} \cdots \xrightarrow{i_r} \omega_r = \omega'$ such that for each $j \in [r]$, $\omega_j \in \mathrm{VSN}(G, k)$.

Problem 2 (Bounded Vertex Separation Number Reconfiguration) *Let G be an n-vertex graph, $\omega, \omega' : [n] \to V(G)$ be linear orders on the vertex set of G, and $k \in \mathbb{N}$. Is it true that ω can be reconfigured into ω' in vertex separation number at most k?*

The proof of Theorem 5 below is analogous to the proof of Theorem 1. More precisely, this proof follows by restating Lemma 2, Lemma 3 and Lemma 4 in terms of the vertex separation number of a graph instead of cutwidth, and then by using this last restated lemma to conclude the proof, as done in Theorem 1.

Theorem 5 *Let G be an n-vertex graph and $\omega, \omega' : [n] \to V(G)$ be linear orders of $V(G)$ of vertex separation number at most k. Then, ω can be reconfigured into ω' in vertex separation number at most* $\mathrm{vsn}(G, \omega) + \mathrm{vsn}(G, \omega') \leq 2k$.

As it has been shown by Kinnersley [32], the vertex separation number equals the possibly nowadays better-known graph parameter *pathwidth*. Therefore, the previous theorem also speaks about a reconfiguration problem for pathwidth. From an application point of view, more precisely, concerning possible applications in the area of VLSI design, the relation to the so-called Gate Matrix Layout Problem is possibly even more important. We abstain from giving a precise definition here, but rather refer to articles as [14, 17, 32, 33, 34]. As the optimum value of the gate matrix layout parameter of a graph G equals the pathwidth of G plus one [32, Corollary 3.2], we also obtain reconfiguration results for this graph parameter motivated by VLSI. Due to results of [35], another graph parameter tightly linked to pathwidth is the *node search number* of a graph; putting the mentioned results together, it also equals the gate matrix layout parameter. In [35], the node search number was shown to be equal to the *minimum progressive black pebble demand* of a graph. This is interesting, as the black pebble demand of a directed acyclic graph is tightly related to a model of register allocation for the computation of arithmetic expressions. The black pebble demand corresponds to the number of registers needed to evaluate an arithmetic expression. Also more generally, such graphs can be used to model the storage requirements of a program. Turning to such problems under the umbrella of reconfiguration could be also an interesting avenue. Again related to register allocation problems are job scheduling problems. We are bringing this connection up in this context, because traditionally job scheduling problems are linked to graph coloring, and recoloring problems can be seen as one of the origins of the field of reconfiguration itself. We only like to draw the reader's attention to [44], where recoloring problems were studied for bounded width parameters, this way returning to one of the main topics of this paper.

6 Conclusion

In this work, we have studied the order reconfiguration problem under the framework of the theory of fixed-parameter tractability. In particular, in our main technical result, we have shown that the order reconfiguration problem for linear orders of cutwidth at most k can always be achieved in cutwidth at most $2k$ (Theorem 1), also in the weighted case. Using this result, we have established new connections between the graph isomorphism problem and the reachability problem for a special two-letter string rewriting system operating on unit slices. In particular, we have proven that unit decompositions $\mathbf{U}$ and $\mathbf{U}'$ of width k are $R(2k)$-equivalent if and only if the graphs $\mathring{\mathbf{U}}$ and $\mathring{\mathbf{U}}'$ are isomorphic (Theorem 2).

Theorem 2 opens up the possibility of studying the graph isomorphism problem under the perspective of term rewriting theory. The most immediate question in this direction is the complexity

of deciding $R(2k)$-reachability for unit decompositions of width k. By a reduction to isomorphism of graphs of cutwidth k, this problem can be solved in time $2^{\mathcal{O}(k \cdot \mathsf{polylog} k)} n^{\mathcal{O}(1)}$ using the results from [26]. Can techniques that are intrinsic to string/term rewriting theory be used to improve this running time? Can such techniques be used to obtain algorithms running in time $f(k) \cdot n^{\mathcal{O}(1)}$ for some computable function $f : \mathbb{N} \to \mathbb{N}$? Note that a positive answer to this question would be conceptually relevant even if the function $f(k)$ is substantially worse than the current $2^{\mathcal{O}(k \cdot \mathsf{polylog}(k))}$, since techniques based on rewriting may carry over to contexts where group-theoretic techniques do not. One interesting line of attack for this question would be to study connections between $R(2k)$ and techniques related to Knuth-Bendix completion and their generalizations [42, 43, 31, 29]. It is also interesting to recall that also Wrochna used some relations to string rewriting questions and graph problems of bounded width in [44].

A natural question that arises in the context of reconfiguration of linear orders is the following: given two linear orders ω and ω', what is the minimum cutwidth of a linear order ω'' occurring in a reconfiguration sequence from ω to ω'? Is this problem NP-hard, or hard to approximate? Is it solvable in FPT-time for certain parameters? We thank one of the reviewers of the conference version [1] for bringing these interesting questions to our attention.

We do hope that the avenue connecting graph isomorphism and term rewriting opens up new lines of research in both areas. For instance, could this approach help when considering graph similarity instead of graph isomorphism, as discussed, e.g., in [19]?

References

[1] E. Arrighi, H. Fernau, M. de Oliveira Oliveira, and P. Wolf. Order reconfiguration under width constraints. In F. Bonchi and S. J. Puglisi, editors, *46th International Symposium on Mathematical Foundations of Computer Science, MFCS*, volume 202 of *LIPIcs*, pages 8:1–8:15. Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 2021. doi:10.4230/LIPIcs.MFCS.2021.8.

[2] V. Arvind, B. Das, J. Köbler, and S. Toda. Colored hypergraph isomorphism is fixed parameter tractable. *Algorithmica*, 71(1):120–138, 2015. doi:10.1007/s00453-013-9787-y.

[3] F. Baader and T. Nipkow. *Term Rewriting and All That*. Cambridge University Press, 1999.

[4] L. Babai. Graph isomorphism in quasipolynomial time [extended abstract]. In D. Wichs and Y. Mansour, editors, *Proceedings of the 48th Annual ACM SIGACT Symposium on Theory of Computing, STOC*, pages 684–697. ACM, 2016. doi:10.1145/2897518.2897542.

[5] L. Babai, D. Y. Grigoryev, and D. M. Mount. Isomorphism of graphs with bounded eigenvalue multiplicity. In H. R. Lewis, B. B. Simons, W. A. Burkhard, and L. H. Landweber, editors, *Proceedings of the 14th Annual ACM Symposium on Theory of Computing, STOC*, pages 310–324. ACM, 1982. doi:10.1145/800070.802206.

[6] L. Babai, W. M. Kantor, and E. M. Luks. Computational complexity and the classification of finite simple groups. In *24th Annual Symposium on Foundations of Computer Science, FOCS*, pages 162–171. IEEE Computer Society, 1983. doi:10.1109/SFCS.1983.10.

[7] E. Barendsen. *Term Rewriting Systems*. Cambridge University Press, 2003.

[8] H. L. Bodlaender, M. R. Fellows, and D. M. Thilikos. Derivation of algorithms for cutwidth and related graph layout parameters. *Journal of Computer and System Sciences*, 75(4):231–244, 2009. doi:10.1016/j.jcss.2008.10.003.

[9] R. V. Book and F. Otto. String-rewriting systems. In *String-Rewriting Systems*, Texts and Monographs in Computer Science, pages 35–64. Springer, 1993. doi:10.1007/978-1-4613-9771-7.

[10] J. Cai, M. Fürer, and N. Immerman. An optimal lower bound on the number of variables for graph identification. *Combincatorica*, 12(4):389–410, 1992. doi:10.1007/BF01305232.

[11] M. de Oliveira Oliveira. Hasse diagram generators and Petri nets. *Fundamenta Informaticae*, 105(3):263–289, 2010. doi:10.3233/FI-2010-367.

[12] M. de Oliveira Oliveira. Subgraphs satisfying MSO properties on z-topologically orderable digraphs. In G. Z. Gutin and S. Szeider, editors, *Parameterized and Exact Computation - 8th International Symposium, IPEC*, volume 8246 of *Lecture Notes in Computer Science*, pages 123–136. Springer, 2013. doi:10.1007/978-3-319-03898-8_12.

[13] M. de Oliveira Oliveira. A slice theoretic approach for embedding problems on digraphs. In E. W. Mayr, editor, *Graph-Theoretic Concepts in Computer Science - 41st International Workshop, WG*, volume 9224 of *Lecture Notes in Computer Science*, pages 360–372. Springer, 2015. doi:10.1007/978-3-662-53174-7_26.

[14] N. Deo, M. S. Krishnamoorthy, and M. A. Langston. Exact and approximate solutions for the gate matrix layout problem. *IEEE Trans. Comput. Aided Des. Integr. Circuits Syst.*, 6(1):79–84, 1987. doi:10.1109/TCAD.1987.1270248.

[15] R. G. Downey and M. R. Fellows. *Fundamentals of Parameterized Complexity*. Texts in Computer Science. Springer, 2013. doi:10.1007/978-1-4471-5559-1.

[16] J. A. Ellis, I. H. Sudborough, and J. S. Turner. The vertex separation and search number of a graph. *Information and Computation*, 113(1):50–79, 1994. doi:10.1006/inco.1994.1064.

[17] M. R. Fellows and M. A. Langston. On well-partial-order theory and its application to combinatorial problems of VLSI design. *SIAM J. Discret. Math.*, 5(1):117–126, 1992. doi:10.1137/0405010.

[18] M. L. Furst, J. E. Hopcroft, and E. M. Luks. Polynomial-time algorithms for permutation groups. In *21st Annual Symposium on Foundations of Computer Science, FOCS*, pages 36–41. IEEE Computer Society, 1980. doi:10.1109/SFCS.1980.34.

[19] T. Gervens and M. Grohe. Graph similarity based on matrix norms. In S. Szeider, R. Ganian, and A. Silva, editors, *47th International Symposium on Mathematical Foundations of Computer Science, MFCS*, volume 241 of *LIPIcs*, pages 52:1–52:15. Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 2022. doi:10.4230/LIPIcs.MFCS.2022.52.

[20] A. C. Giannopoulou, M. Pilipczuk, J.-F. Raymond, D. M. Thilikos, and M. Wrochna. Cutwidth: obstructions and algorithmic aspects. *Algorithmica*, 81(2):557–588, 2019. doi:10.1007/s00453-018-0424-7.

[21] M. Grohe. Fixed-point definability and polynomial time on graphs with excluded minors. *Journal of the ACM*, 59(5):27, 2012. doi:10.1145/2371656.2371662.

[22] M. Grohe and D. Marx. Structure theorem and isomorphism test for graphs with excluded topological subgraphs. *SIAM Journal on Computing*, 44(1):114–159, 2015. doi:10.1137/120892234.

[23] M. Grohe and D. Neuen. Isomorphism, canonization, and definability for graphs of bounded rank width. *Commununications of the ACM*, 64(5):98–105, 2021. doi:10.1145/3453943.

[24] M. Grohe and D. Neuen. Recent advances on the graph isomorphism problem. In K. K. Dabrowski, M. Gadouleau, N. Georgiou, M. Johnson, G. B. Mertzios, and D. Paulusma, editors, *Surveys in Combinatorics, 2021: Invited lectures from the 28th British Combinatorial Conference, Durham, UK, July 5-9, 2021*, pages 187–234. Cambridge University Press, 2021. doi:10.1017/9781009036214.006.

[25] M. Grohe, D. Neuen, and P. Schweitzer. A faster isomorphism test for graphs of small degree. In M. Thorup, editor, *59th IEEE Annual Symposium on Foundations of Computer Science, FOCS*, pages 89–100. IEEE Computer Society, 2018. doi:10.1109/FOCS.2018.00018.

[26] M. Grohe, D. Neuen, P. Schweitzer, and D. Wiebking. An improved isomorphism test for bounded-tree-width graphs. *ACM Transactions on Algorithms*, 16(3):34:1–34:31, 2020. doi:10.1145/3382082.

[27] M. Grohe and P. Schweitzer. Isomorphism testing for graphs of bounded rank width. In V. Guruswami, editor, *IEEE 56th Annual Symposium on Foundations of Computer Science, FOCS*, pages 1010–1029. IEEE Computer Society, 2015. doi:10.1109/FOCS.2015.66.

[28] M. Grohe and P. Schweitzer. The graph isomorphism problem. *Commun. ACM*, 63(11):128–134, 2020. doi:10.1145/3372123.

[29] N. Hirokawa, A. Middeldorp, C. Sternagel, and S. Winkler. Abstract completion, formalized. *Log. Methods Comput. Sci.*, 15(3), 2019. doi:10.23638/LMCS-15(3:19)2019.

[30] T. Ito, E. D. Demaine, N. J. A. Harvey, C. H. Papadimitriou, M. Sideri, R. Uehara, and Y. Uno. On the complexity of reconfiguration problems. *Theoretical Computer Science*, 412(12-14):1054–1065, 2011. doi:10.1016/j.tcs.2010.12.005.

[31] D. Kapur and P. Narendran. The Knuth-Bendix completion procedure and Thue systems. *SIAM Journal on Computing*, 14(4):1052–1072, 1985. doi:10.1137/0214073.

[32] N. G. Kinnersley. The vertex separation number of a graph equals its path-width. *Inf. Process. Lett.*, 42(6):345–350, 1992. doi:10.1016/0020-0190(92)90234-M.

[33] N. G. Kinnersley and W. M. Kinnersley. An efficient polynomial-time algorithm for three-track gate matrix layout. *Comput. J.*, 37(5):449–462, 1994. doi:10.1093/comjnl/37.5.449.

[34] N. G. Kinnersley and M. A. Langston. obstruction set isolation for the gate matrix layout problem. *Discret. Appl. Math.*, 54(2-3):169–213, 1994. doi:10.1016/0166-218X(94)90021-3.

[35] L. M. Kirousis and C. H. Papadimitriou. Searching and pebbling. *Theor. Comput. Sci.*, 47(3):205–218, 1986. doi:10.1016/0304-3975(86)90146-5.

[36] J. W. Klop. *Term Rewriting Systems*. Centrum voor Wiskunde en Informatica, 1990.

[37] S. Kratsch and P. Schweitzer. Isomorphism for graphs of bounded feedback vertex set number. In H. Kaplan, editor, *Algorithm Theory - SWAT, 12th Scandinavian Symposium and Workshops on Algorithm Theory*, volume 6139 of *Lecture Notes in Computer Science*, pages 81–92. Springer, 2010. doi:10.1007/978-3-642-13731-0_9.

[38] D. Lokshtanov, M. Pilipczuk, M. Pilipczuk, and S. Saurabh. Fixed-parameter tractable canonization and isomorphism test for graphs of bounded treewidth. In *55th IEEE Annual Symposium on Foundations of Computer Science, FOCS*, pages 186–195. IEEE Computer Society, 2014. doi:10.1109/FOCS.2014.28.

[39] E. M. Luks. Isomorphism of graphs of bounded valence can be tested in polynomial time. *Journal of Computer and System Sciences*, 25(1):42–65, 1982. doi:10.1016/0022-0000(82)90009-5.

[40] G. Miller. Isomorphism testing for graphs of bounded genus. In R. E. Miller, S. Ginsburg, W. A. Burkhard, and R. J. Lipton, editors, *Proceedings of the 12th Annual ACM Symposium on Theory of Computing, STOC*, pages 225–235. ACM, 1980. doi:10.1145/800141.804670.

[41] N. Nishimura. Introduction to reconfiguration. *Algorithms*, 11(4):52, 2018. doi:10.3390/a11040052.

[42] C. Sternagel and R. Thiemann. Formalizing knuth-bendix orders and knuth-bendix completion. In F. van Raamsdonk, editor, *24th International Conference on Rewriting Techniques and Applications, RTA 2013, June 24-26, 2013, Eindhoven, The Netherlands*, volume 21 of *LIPIcs*, pages 287–302. Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 2013. doi:10.4230/LIPIcs.RTA.2013.287.

[43] I. Wehrman, A. Stump, and E. Westbrook. Slothrop: Knuth-Bendix completion with a modern termination checker. In F. Pfenning, editor, *Term Rewriting and Applications, 17th International Conference, RTA*, volume 4098 of *Lecture Notes in Computer Science*, pages 287–296. Springer, 2006. doi:10.1007/11805618_22.

[44] M. Wrochna. Reconfiguration in bounded bandwidth and tree-depth. *Journal of Computer and System Sciences*, 93:1–10, 2018. doi:10.1016/j.jcss.2017.11.003.