# Parameterized Shortest Path Reconfiguration

**Nicolas Bousquet**
Univ. Lyon, LIRIS, CNRS, Université Claude Bernard Lyon 1, Villeurbanne, France

**Kshitij Gajjar**
Centre for Security, Theory & Algorithmic Research (CSTAR),
International Institute of Information Technology, Hyderabad (IIIT-H), India

**Abhiruk Lahiri**
Department of Computer Science, Heinrich Heine University, Düsseldorf, Germany

**Amer E. Mouawad**
Department of Computer Science, American University of Beirut, Lebanon

## Abstract

An $st$-shortest path, or $st$-path for short, in a graph $G$ is a shortest (induced) path from $s$ to $t$ in $G$. Two $st$-paths are said to be adjacent if they differ on exactly one vertex. A reconfiguration sequence between two $st$-paths $P$ and $Q$ is a sequence of adjacent $st$-paths starting from $P$ and ending at $Q$. Deciding whether there exists a reconfiguration sequence between two given $st$-paths is known to be PSPACE-complete, even on restricted classes of graphs such as graphs of bounded bandwidth (hence pathwidth). On the positive side, and rather surprisingly, the problem is polynomial-time solvable on planar graphs. In this paper, we study the parameterized complexity of the Shortest Path Reconfiguration (SPR) problem. We show that SPR is W[1]-hard parameterized by $k + \ell$, even when restricted to graphs of bounded (constant) degeneracy; here $k$ denotes the number of edges on an $st$-path, and $\ell$ denotes the length of a reconfiguration sequence from $P$ to $Q$. We complement our hardness result by establishing the fixed-parameter tractability of SPR parameterized by $\ell$ and restricted to nowhere-dense classes of graphs. Additionally, we establish fixed-parameter tractability of SPR when parameterized by the treedepth, by the cluster-deletion number, or by the modular-width of the input graph.

**2012 ACM Subject Classification** Theory of computation → Parameterized complexity and exact algorithms

**Keywords and phrases** combinatorial reconfiguration, shortest path reconfiguration, parameterized complexity, structural parameters, treedepth, cluster deletion number, modular width

**Digital Object Identifier** 10.4230/LIPIcs.IPEC.2024.23

**Related Version** *Full Version*: https://doi.org/10.48550/arXiv.2406.12717

**Funding** *Abhiruk Lahiri*: Partly supported by SFF grant "Theoretical AI" of HHU Düsseldorf and the Deutsche Forschungsgemeinschaft (DFG, German Research Foundation) – project number 456558332.

## 1 Introduction

Many algorithmic questions can be posed as follows: given the description of a system state and the description of a state we would "prefer" the system to be in, is it possible to transform the system from its current state into a more desired one without "breaking" the system in the process? And if yes, how many steps are needed? Such problems naturally arise in the fields of mathematical puzzles, operational research, computational geometry [15], bioinformatics, and quantum computing [10]. These questions received a substantial amount of attention under the so-called *combinatorial reconfiguration framework* in the last decade. We refer the reader to the surveys by van den Heuvel [18], Nishimura [16] and Bousquet *et al.* [6] for more background on combinatorial reconfiguration.


19th International Symposium on Parameterized and Exact Computation (IPEC 2024).
Editors: Édouard Bonnet and Paweł Rzążewski; Article No. 23; pp. 23:1–23:14
Leibniz International Proceedings in Informatics
LIPICS Schloss Dagstuhl – Leibniz-Zentrum für Informatik, Dagstuhl Publishing, Germany

**Shortest path reconfiguration.** In this work, we focus on the reconfiguration of $st$-shortest paths (or $st$-paths for short) in undirected, unweighted, simple graphs. It is well-known that one can easily find an $st$-path in a graph in polynomial time. In order to define the reconfiguration variant of the problem, we first require a notion of adjacency between $st$-paths.

As is common in the combinatorial reconfiguration framework, we focus on two models; the token-jumping model (TJ) and the token-sliding model (TS). We say that two $st$-paths are *TJ-adjacent* if they differ on exactly one vertex, i.e., all the vertices are the same except at a unique position $p$. We say that two $st$-paths $P$ and $Q$ are *TS-adjacent* if they are TJ-adjacent and the $p^{\text{th}}$ vertex of $P$ and the $p^{\text{th}}$ vertex of $Q$ are adjacent. A *reconfiguration sequence from $P$ to $Q$* (if it exists) is a sequence of adjacent shortest paths starting at $P$ and ending at $Q$. In the Shortest Path Reconfiguration (SPR) problem, we are given a graph $G$, two vertices $s$ and $t$, two $st$-paths $P$ and $Q$ of length $k$ each, and the goal is to decide whether a reconfiguration sequence from $P$ to $Q$ exists. In the Shortest Shortest Path Reconfiguration (SSPR) problem, we are additionally given an integer $\ell$ which is an upper bound on the length of the desired reconfiguration sequence. Reconfiguration of shortest paths has many applications, e.g., in network design and operational research (we refer the interested reader to [9] for a detailed discussion around these applications).

Many reconfiguration problems, SPR and SSPR included, naturally lie in the class PSPACE. Since there are no simple polynomial-time checkable certificates (as reconfiguration sequences are possibly of exponential length), they are generally not in NP. A decade ago, Bonsma [3] proved that SPR (under token jumping) is PSPACE-complete. In fact, the problem remains PSPACE-complete even when restricted to bipartite graphs [3], line graphs [9], and graphs of bounded bandwidth/pathwidth/treewidth [19]. Several groups studied the complexity of the problem in other restricted graph classes such as grid graphs [1], claw-free graphs, chordal graphs [3], and circle graphs [9]. The most notable result has been obtained by Bonsma who showed that Shortest Path Reconfiguration can be decided in polynomial time for planar graphs [4]. This result is rather surprising in the reconfiguration setting since most reconfiguration problems are known to be PSPACE-complete on planar graphs, see e.g. [13, 14, 5].

**Our results.** Our focus is on the parameterized complexity of shortest path reconfiguration problems; which, to the best of our knowledge, has not been studied so far. Other reconfiguration problems have been widely studied from a parameterized perspective in the last decade, see, e.g., [6] for a survey. A problem is *fixed-parameter tractable*, FPT for short, on a class $\mathcal{C}$ of graphs with respect to a parameter $\kappa$, if there is an algorithm deciding whether a given input instance with graph $G \in \mathcal{C}$ admits a solution in time $f(\kappa) \cdot |V(G)|^c$, for a computable function $f$ and constant $c$.

A *kernelization algorithm* is a polynomial-time algorithm that reduces an input instance to an equivalent instance of size bounded in the parameter only (independent of the input size), known as a *kernel*; we will say that two instances are *equivalent* if they are both yes-instances or both no-instances. Every fixed-parameter tractable problem admits a kernel, however, possibly of exponential or worse size. For efficient algorithms, it is therefore most desirable to obtain polynomial, or even linear, kernels. The W-hierarchy is a collection of parameterized complexity classes $\mathsf{FPT} \subseteq \mathsf{W[1]} \subseteq \mathsf{W[2]} \subseteq \cdots \subseteq \mathsf{W[t]}$, for $t \in \mathbb{N}$. The conjecture $\mathsf{FPT} \subsetneq \mathsf{W[1]}$ can be seen as the analogue of the conjecture that $\mathsf{P} \subsetneq \mathsf{NP}$. Before stating our results precisely, let us formally define the problems we are interested in (we intentionally omit the type of move, i.e., slide or jump, from the definitions, as it will be clear from context in what follows):

Shortest Path Reconfiguration (SPR)
**Input:** A graph $G$, two vertices $s, t$, two $st$-shortest paths $P, Q$.
**Question:** Is there a reconfiguration sequence from $P$ to $Q$?

Shortest Shortest Path Reconfiguration (SSPR)
**Input:** A graph $G$, two vertices $s, t$, two $st$-shortest paths $P, Q$, an integer $\ell$.
**Question:** Is there a reconfiguration sequence from $P$ to $Q$ of length at most $\ell$?

In parameterized complexity, one is usually interested in two types of parameters: parameters related to the size of the solution or parameters related to the structure of the input graph. For shortest path reconfiguration, there are two parameters related to the size of the solution which are the length $\ell$ of a reconfiguration sequence, and the length $k$ of the shortest $st$-paths (number of edges on the shortest $st$-paths) in $G$. Our first results will focus on these parameters. We will then discuss some parameters related to the graph structure such as treedepth and modular width. Our first result is a hardness result. We prove that the following holds (in both the token jumping and the token sliding models):

▶ **Theorem 1.** *SPR is* W[1]*-hard parameterized by $k$, and SSPR is* W[1]*-hard parameterized by $k + \ell$, in both the token jumping and the token sliding models.*

The idea of the proof of Theorem 1 is a reduction from the Multicolored Clique problem. Let $(V_i)_{i \leq k}$ be the vertices of an instance of the Multicolored Clique problem. Intuitively (the real proof being more technical), we will construct a graph where the length of the $st$-paths will be in $\mathcal{O}(k^2)$, each integer representing a vertex of the set $V_i$. The goal would be to transform a path $P$ into a path $Q$, forcing us to select a vertex in each set. For every pair $i, j$, there exists an integer $r$ such that the $r^{\text{th}}$ vertex corresponds to a vertex in $V_i$ and the $(r+1)^{\text{th}}$ vertex corresponds to a vertex in $V_j$. The key argument of the proof consists in finding a mechanism to ensure that the vertex selected in each copy of $V_i$ is the same, which permits us to conclude that the subset of selected vertices is a multicolored clique of the desired size. One can then naturally wonder if this hardness result can be pushed further. The answer is yes, and in fact, we prove (in the full version of the paper) that the problems are hard even restricted to a very simple class of graphs:

▶ **Theorem 2.** *SPR is* W[1]*-hard parameterized by $k$, and SSPR is* W[1]*-hard parameterized by $k + \ell$, even when the inputs are restricted to graphs of constant degeneracy and in both the token jumping and the token sliding models.*

In order to prove that statement we adapt the proof of Theorem 1 to appropriately reduce the degeneracy of the graph. We then complement these negative results with the following positive ones.

▶ **Theorem 3.** *SSPR is* FPT *parameterized by $\ell$ on nowhere-dense classes of graphs (in both the token jumping and the token sliding models).*

The idea of the proof of Theorem 3 consists in proving that if $k$ is too large compared to $\ell$ then there are many positions along the shortest paths that are already occupied by tokens that never have to move. Using this fact, we then contract parts of the paths in order to get $st$-paths of length $\mathcal{O}(f(\ell))$, for some computable function $f$. Now, since $k$ is bounded by some function of $\ell$, one can prove that the existence of a reconfiguration sequence of length $\ell$ can be verified via model checking a first-order formula $\phi$ whose size depends only on $\ell$. Combining this observation with the black-box result of [11] that ensures that the model checking problem can be decided in time $\mathcal{O}(f(|\phi|) \cdot |V(G)|)$ on nowhere-dense graphs, we get the desired result. We proceed (in the full version of the paper) by considering some of the most commonly studied structural graph parameters. In particular, we prove the following:

▶ **Theorem 4.** *SPR and SSPR (in both the token jumping and the token sliding model) are* FPT *when parameterized by either the treedepth, the cluster deletion number, or the modular width of the input graph.*

To motivate the study of these parameters, we refer the reader to Figure 1. Recall that SPR is PSPACE-complete even when restricted to graphs of bounded bandwidth, pathwidth, treewidth, and cliquewidth [19]. This implies para-PSPACE-hardness on the aforementioned classes. Hence, our Theorem 4 almost completes the picture for structural parameterizations of the problems, leaving open the case of feedback vertex set number.

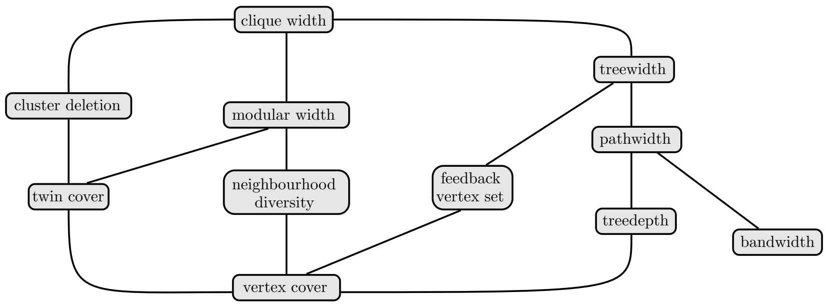


**Figure 1** The graph parameters studied in this paper. A connection between two parameters indicates the existence of a function in the one above that lower-bounds the one below.

**Further discussions and open problems.** As we show in the full version of the paper, it turns out that when solving the SPR problem parameterized by the feedback vertex set number of the graph, one can assume that $k$, the length of $st$-paths, is bounded linearly in the parameter. Hence, the following remains an interesting open question:

▶ **Problem 1.** *Is SPR fixed-parameter tractable when parameterized by feedback vertex set number?*

When the feedback vertex set number is bounded, the graph can be seen as a disjoint union of trees plus a bounded number of additional vertices. One can easily remark that if vertices of the feedback vertex set are far apart in the $st$-paths then the structure is very rigid and very few tokens can move in the graph. However, when vertices of the feedback vertex set are close to one another (along the $st$-paths), there might exist some arbitrarily long paths between two layers in the layered partition of the graph. Here, the layered partition refers to the partitioning of the vertex set based on distance either from $s$ or from $t$. Tokens along these (layer) paths that do not belong to the feedback vertex set are not restricted and can traverse their corresponding layer path in both directions an unbounded number of times. In particular, it implies that, if there exists a reconfiguration sequence, that sequence might be arbitrarily long. So in order to design a reconfiguration sequence (from a kernelization perspective at least, which is known to be equivalent to fixed-parameter tractability), we have to find a way to reduce these long structures into structures of bounded length. We were not able to solve this very special case of the problem.

As far as we know, it also remains an open question whether SPR is in P or is NP-complete on graphs of constant feedback vertex set number. Note that an XP algorithm follows immediately from the fact that (after appropriately discarding parts of the input) the number of $st$-paths is roughly $|V(G)|^f$, where $f$ denotes the feedback vertex set number. Regardless, in case of a positive answer to Problem 1, the next natural question is the following:

▶ **Problem 2.** *Is SPR fixed-parameter tractable when parameterized by $k$ on graphs of bounded pathwidth? What about treewidth? How about parameterization by $k$ plus the treewidth?*

It is an easy exercise to remark that SPR is PSPACE-complete on graphs of bounded bandwidth, pathwidth, and treewidth using a simple reduction from $H$-Word Reconfiguration [19]. When the treewidth is 1, there exists a unique minimum *st*-path and the problem is simple. Trees and forests are graphs which are 1-degenerate and every 1-degenerate graph is a forest, however, the complexity of SPR and SSPR remains open for 2-degenerate graphs.

▶ **Problem 3.** *What is the complexity of SPR and SSPR on* 2*-degenerate graphs?*

**Related work.** Reconfiguration of paths and other subgraphs has been considered before [7, 12, 8]. For some of these past works, i.e., [7, 12], the paths have fixed length and a "move" consists of removing a vertex at one end and adding a vertex at the other end, as in certain "snake-like" games. In contrast, for our work, the two endpoints of the paths are fixed and the paths are required to be unweighted shortest paths between those endpoints.

Demaine *et al.* proved in [7] that the problem of reconfiguring (arbitrary) paths via snake-like moves is PSPACE-complete in general, and polynomial-time solvable for some restricted graph classes. When not restricted to shortest paths, the problem is quite different, since the extremities of the paths are not fixed and the goal is not necessarily to reconfigure shortest paths. In fact, it is proved in [7] that fixed-length path reconfiguration (under the snake-like moves described above) is fixed-parameter tractable parameterized by the path length or by the circuit rank, XP parameterized by the feedback vertex set number, and PSPACE-complete even for graphs of bounded bandwidth [19]. Gupta et al. [12] also show fixed-parameter tractability parameterized by path length for a different type of snake-like moves, i.e., paths are considered directed paths and are required to move forwards only.

Reconfiguration problems on graphs of bounded feedback vertex set number and on graphs of bounded treewidth have already received a considerable amount of attention, and they are usually not easy to place in FPT (unlike their optimization counterparts, where a simple branching strategy or dynamic programming algorithm is usually enough to get an FPT algorithm). For instance, Independent Set Reconfiguration (in the token sliding model) on graphs of bounded feedback vertex set number is FPT; this fact follows easily from the multi-component reduction in [2]. However, the question is still open for the reconfiguration of dominating sets, for instance. The case of bounded treewidth graphs is open for both Independent Set Reconfiguration and Dominating Set Reconfiguration (in the sliding model) [6].

## 2 Hardness results

We start with the case of SPR parameterized by $k$ on general graphs. The same reduction will imply the hardness of SSPR parameterized by $k + \ell$. We describe in the full version of the paper how to modify the construction to obtain a graph of constant degeneracy[1].

Our reduction is from the Regular Multicolored Clique (RMC) problem, which is known to be NP-complete and W[1]-hard when parameterized by solution size $\kappa$ [17]. The problem is defined as follows. We are given a $\kappa$-partite graph $G = (V, E)$ such that $V$ is partitioned into $\kappa$ independent sets $V = V_1 \cup V_2 \cup \cdots \cup V_\kappa$ and each partition has size exactly $n$, i.e., $|V| = \kappa n$. We denote the vertices of $V_i$ by $v^i_1, v^i_2, \ldots, v^i_n$. Moreover, every vertex

[1] Proofs of statements marked with a star are omitted due to space constraints.

$v_j^i \in V_i$ has exactly $r$ neighbors in every set $V_{i'}$, $i \neq i'$. In other words, every vertex in $G$ has degree exactly $r(\kappa - 1)$. Given an instance $(G, \kappa)$ of RMC, the goal is to decide if $G$ contains a clique of size $\kappa$, which we call a multicolored clique since it must contain exactly one vertex from each $V_i$, $i \in [\kappa]$. We reduce $(G, \kappa)$ to an instance $(G', s, t, P, Q)$ of SPR, where $P$ and $Q$ are $st$-paths in $G'$ of length $k = \mathcal{O}(\kappa^2)$.

**Properly colored $st$-paths.** Before discussing $G'$, we start by describing a key gadget of our construction which is a graph called $H$. The graph $H$ consists of $\alpha = 6\kappa^2$ sets of vertices $H_1, H_2, \ldots, H_\alpha$ such that $|H_i| = n$ for each $i \in [\alpha]$. We group every three consecutive sets into $\beta = 2\kappa^2$ groups $R_1 = \{H_1, H_2, H_3\}$, $R_2 = \{H_4, H_5, H_6\}$, $R_3 = \{H_7, H_8, H_9\}$, …, and $R_\beta = \{H_{\alpha-2}, H_{\alpha-1}, H_\alpha\}$. We call $H_i$ the $i$th layer of $H$ and $R_i$ the $i$th group of $H$; it will become clear later that a shortest path will select a vertex from each $H_i$. We also define a mapping $\mu : [\beta] \to [\kappa]$ such that each $R_i$ is mapped to some $V_j$, for $i \in [\beta]$ and $j \in [\kappa]$. In other words, each $R_i = \{H_a, H_b, H_c\}$ will correspond to taking three copies of some $V_j$. We sometimes abuse notation and write $\mu(R_i) = V_j$ to denote the image of a set. We also overload notation and write $\mu(H_p) = V_j$ whenever $H_p \in R_i$ and $\mu(R_i) = V_j$.

Furthermore, we construct $\mu$ in such a way that, for every pair $(j, j')$, $j \neq j'$ and $j, j' \in [\kappa]$, there exists at least one integer $i < \beta$ such that $\mu(i) = j$, $\mu(i+1) = j'$. In other words, for every two sets $V_j$ and $V_{j'}$, there must exist two consecutive groups $R_i$ and $R_{i+1}$ such that $R_i$ is mapped to $V_j$ and $R_{i+1}$ is mapped to $V_{j'}$. One can easily check that it is indeed possible to construct such a function $\mu$ when $\beta = 2\kappa^2$. We define $\mu$ as follows:

$$\text{For each } i \in [\beta],\ R_i \text{ is mapped to } V_{\mu(i)}, \text{ where } \mu(i) = \begin{cases} 1 + \lfloor (i-1)/2\kappa \rfloor & i \text{ is odd;} \\ 1 + ((i-2) \bmod 2\kappa)/2 & i \text{ is even.} \end{cases}$$

▶ **Observation 5.** *For each $(j, j') \in [\kappa] \times [\kappa]$ such that $j \neq j'$, there exists an $i \in [\beta - 1]$ such that $\mu(i) = j$ and $\mu(i+1) = j'$.*

We also define a mapping $\pi_i : R_i \to V_{\mu(i)}$ (and $\pi_i : H_i \to V_{\mu(i)}$) that maps every vertex of $R_i$ ($H_i$) to its corresponding vertex in $V_{\mu(i)}$. We drop the subscript $i$ when clear from context. We note that each vertex of $V_{\mu(i)}$ appears three times in $R_i$ (once in each layer) and all three vertices map to the same vertex of $V_{\mu(i)}$. Let us now describe the edge set of $H$. For every $i \in [\beta]$, we add a matching between vertices of $H_j$ and $H_{j+1}$ and a matching between vertices of $H_{j+1}$ and $H_{j+2}$ whenever there exists a group $R_i$ such that $R_i = \{H_j, H_{j+1}, H_{j+2}\}$. For every two consecutive groups $R_i = \{H_j, H_{j+1}, H_{j+2}\}$ and $R_{i+1} = \{H_{j+3}, H_{j+4}, H_{j+5}\}$, we add in $H$ the edges of $G$ between $H_{j+2}$ and $H_{j+3}$. That is, we add between consecutive sets corresponding to different sets of $G$ the edges corresponding to the edges between those two sets in $G$. More formally, let $a \in H_{j+2}$, $b \in H_{j+3}$, $\pi(a) \in V_{\mu(i)}$, and $\pi(b) \in V_{\mu(i+i)}$. Then, there is an edge between vertices $a$ and $b$ in $H$ if and only if there is an edge between vertices $\pi(a)$ and $\pi(b)$ in $G$.

Assume that we create a new graph $H'$ consisting of $H$ plus two additional vertices $s$ and $t$, where $s$ is connected to all the vertices of $H_1$ and $t$ is connected to all the vertices of $H_\alpha$. Note that any $st$-path in $H'$ must contain exactly one vertex from every layer. We say that an $st$-path $P$ is *properly colored* whenever for any $a \in H_i$ and $b \in H_j$ (on the path) such that $\mu(i) = \mu(j)$, we have $\pi(a) = \pi(b)$. In other words, whenever two layers of $H$ (containing vertices of $P$) map to the same set of $V$ we must select the same vertices in both. We note that any $st$-path $P$ in $H'$ can intersect with a group $R_i$ in one of $n$ ways, i.e., the vertices of $P$ in $R_i$ all map to the same vertex of $V_{\mu(i)}$.

▶ **Observation 6** ($\star$)**.** *$H'$ contains a properly colored $st$-path $P$ (consisting of $6\kappa^2 + 2$ vertices) if and only if $G$ contains a multicolored clique of size $\kappa$.*

**Outline of the reduction.** Assume that we add to the graph $H'$ two new (internally) vertex-disjoint $st$-paths $P$ and $Q$ each containing exactly $\alpha + 2$ vertices ($s$ and $t$ and one vertex per layer of $H$). We add all the edges between the $i$-th vertex of $P$ and the vertices in layers $i$, $i-1$, and $i+1$ of $H$ (with the assumption that $H_0 = \{s\}$ and $H_{\alpha+1} = \{t\}$). Similarly, we add all the edges between the $i$-th vertex of $Q$ and the vertices in layers $i$, $i-1$, and $i+1$ of $H$. We denote the resulting graph by $H' + P + Q$.

Consider the instance $(H' + P + Q, s, t, P, Q)$ of SPR. If there exists a multicolored clique in $G$ then there exists a properly colored $st$-path in $H'$ by Observation 6. By the definition of the edge set, on can easily see that we can transform $P$ into $Q$ by first moving the vertices of $P$ onto a properly colored $st$-path in $H'$ and then moving all the vertices to $Q$ one by one. Unfortunately, the converse is not necessarily true since we might not be consistent in the selection of vertices in $H'$, i.e., we might select vertices $a \in H_i$ and $b \in H_j$ in the path such that $\mu(i) = \mu(j)$ and $\pi(a) \neq \pi(b)$ ($H_i$ and $H_j$ belong to different groups).

By considerably complicating the gadgetry, we will prove that we can handle this issue. To do so, we create a new gadget that will force us to select the same vertex for a fixed value of the image of $\mu$. We replicate our gadget to enforce the consistency of all the images of $\mu$. In addition to enforcing consistent selection of vertices, our construction further guarantees that choices cannot be undone.

Another issue in the simplistic construction of $H'$ described above is that we implicitly assume that we move from $P$ to a path fully contained in $H$ before going to $Q$. But nothing prevents an $st$-path from containing some vertices of $P$, then some vertices from $H$, then some vertices from $Q$, then more vertices from $H$, and so on. To avoid this phenomenon, we shall add what we call buffer space. We formalize all these ideas next.

**Buffers and collapses.** Most of the time, we will consider matchings and edges between sets of size $n$. Given two sets of size $n$ (with an implicit ordering), we define the natural matching as the matching that matches the vertices in increasing index order (in the natural way). We will sometimes consider edges between a set $A$ of size $n$ and a set $B$ of size larger than $n$ with a canonical mapping function to $\{1, \ldots, n\}$. By abuse of notation, we still denote by the natural matching the set of edges (that is not a matching anymore) that links the $i$-th vertex of $A$ and all the vertices that map to $i$ in $B$.

We denote by $I^n$ ($J^n$) the independent set on $n$ vertices[2]. We drop the superscript $n$ when clear from context. We let $\mathcal{I}^q$ ($\mathcal{J}^q$) denote the graph obtained by taking $q$ copies of $I^n$ ($J^n$) where consecutive copies of $I^n$ ($J^n$) are linked with the natural matching. Note that $\mathcal{I}^q$ ($\mathcal{J}^q$) consists of exactly $n$ paths on $q$ vertices. We use $I_i$ (resp. $J_i$) to denote the $i$th copy of $I^n$ (resp. $J^n$) in $\mathcal{I}^q$ (resp. $\mathcal{J}^q$).

Let $R = R_1, R_2, \ldots, R_\gamma$ be a graph where edges are between consecutive sets and there is a canonical mapping from $R_1$ and $R_\gamma$ to $\{1, \ldots, n\}$ (in our proof, $R$ will be $H$ or a graph close to $H$). We write $\Gamma(p, H, q) = \mathcal{I}^p \oplus H \oplus \mathcal{J}^q$ (or $\Gamma$ when $p, q, H$ are clear from context) to denote the graph obtained by taking a copy of $\mathcal{I}^p$, a copy of $\mathcal{J}^q$, a copy of $H$, and then adding the natural matching between the vertices of $I_q$ and $H_1$ as well as a matching between the vertices of $H_\alpha$ and $J_1$. If we denote by $I_i$ the sets of $\mathcal{I}^p$ and $J_i$ the sets of $\mathcal{J}^q$, for $i \in [p + \alpha + q]$, we call $L_i$ the $i$-th layer of $\Gamma(p, H, q)$, where $L_i = I_i$ when $i \leq p$, $L_i = H_{i-q}$ when $p < i \leq p + \alpha$, and $L_i = J_{i-(q+\alpha)}$ when $i > q + \alpha$.

[2] These two notations that denote the same graph will permit to simplify the description of the constructions in the rest of the paper.

Given the graph $H$ (recall that $H$ consists of $\alpha = 6\kappa^2$ sets of vertices $H_1, \ldots, H_\alpha$) and a vertex $h^i_j \in H_i$, we let $H(h^i_j)$ denote the graph obtained from $H$ by deleting all but one vertex from each set $H_{i'}$, where $\mu(i') = \mu(i)$; we delete all vertices except for $h^{i'}_j \in H_{i'}$ (deleting vertices implies the deletion of edges incident on those vertices). That is, we restrict all the layers of $H$ corresponding to $V_{\mu(i)}$ to a single vertex (the same vertex); we have $\pi(h^i_j) = \pi(h^{i'}_j)$ for all $i, i'$ with $\mu(i) = \mu(i')$. We say that $H(h^i_j)$ is a *collapse of $H$ on $h^i_j$*, or, equivalently, collapsing $H$ on $h^i_j$ results in $H(h^i_j)$.

Now, for every $i \leq \kappa, j \leq n$, we define $\Gamma_{i,j}(p, q)$ as $\Gamma(p, H(h^i_j), q)$. Finally, we let $\Gamma^i(p, q)$ denote the union of the $n$ graphs $\Gamma_{i,j}(p, q)$. We write $\Gamma_{i,j} = \Gamma_{i,j}(p, q)$ whenever $p$ and $q$ are clear from context. Note that all the $\Gamma_{i,j}$ being disjoint, if we have a path fully included in one of the $\Gamma_{i,j}(p, q)$ at some point, then all the selected vertices in sets mapping to $i$ by $\mu$ are the same. That is, $\Gamma^i(p, q)$ will allow us to verify that for any $H_j, H'_j$ in the selection gadget such that $\mu(j) = \mu(j') = i$ we always pick vertices $a \in H_j$, $b \in H_{j'}$ such that $\pi(a) = \pi(b)$.

**Construction.** We are now ready to describe the construction of the instance $(G', s, t, P, Q)$ of SPR. We consider the token jumping model (changes required for sliding can be found in the full version of the paper). We start from an empty graph $G'$ and add two new vertices $s$ and $t$. We let $q = 2\kappa^2$ and $\delta = 2q + \alpha = 10\kappa^2$. We add two internally vertex-disjoint $st$-paths $P$ and $Q$ consisting of $\delta$ internal vertices each.

The next step consists of adding $\Gamma_\star = \Gamma(q, H, q)$ to $G'$ and connecting $s$ to every vertex in $I_1$ and $t$ to every vertex in $J_q$. Moreover, we let the $i$th internal vertex of $P$, $i \geq 2$, be adjacent to every vertex in layer $i-1$ of $\Gamma_\star$. We call $\Gamma_\star$ the *selection gadget*. The rest of the gadgets will be verification and boundary gadgets that allow us to guarantee that properties similar to those in Observation 6 will hold.

We then create a graph $\Gamma^1(q-1, q+1)$ denoted by $\Gamma^1$ which will be the *verification gadget for $i = 1$*. We deal with the graphs of $\Gamma^1$ first (and slightly differently than the rest) as they require special attention given that they exist at the "boundary" of our construction. Notice that, in $G'$, all the graphs in $\Gamma^1$ are "shifted one position to the left with respect to $\Gamma_\star$" (in the sense that the number of independent sets at the left has reduced by one), see Figure 2 for an illustration. In particular, the graph $H$ of each $\Gamma_{1,j}$, $j \in [n]$, starts (or appears) one layer before the graph $H$ in $\Gamma_\star$. We now describe the edges between $\Gamma_\star$ and any $\Gamma_{1,j}$ (in $\Gamma^1$). Let $L$ denote some layer of $\Gamma_{1,j}$ (ignoring the last layer) and let $L'$ be the layer after $L$ in $\Gamma_\star$. If $L$ and $L'$ correspond to independent sets (not sets of $H$) they are connected by the natural matching. Otherwise, we have two cases:

- If layer $L$ of $\Gamma_{1,j}$ corresponds to a set $H_p$ with $\mu(p) = 1$ then we deleted all vertices of $L$ except for $h^p_j$ (collapse). We connect $h^p_j$ to its image in $L'$, which must exists since layer $L'$ of $\Gamma_\star$ corresponds to a set $H_{p'}$ with $\mu(p) = \mu(p') = 1$.
- Otherwise, we have the same number of vertices in $L$ and $L'$ and we add a matching between the pairs of vertices having the same image in $G$.

We now add a boundary gadget that will separate all the verification gadgets and allow us to simplify some of the arguments. Picturing the graph being constructed from top to bottom with $P$ and $Q$ encircling all of the graph, we assume that $\Gamma_{i,j}$ is drawn before $\Gamma_{i,j+1}$. Similarly, we only insert $\Gamma_{i+1,j}$ after inserting all graphs of $\Gamma^i$ (see again Figure 2 for an illustration). After $\Gamma_{1,n}$ is inserted, we insert another graph (connecting $s$ and $t$) that we denote by $\Gamma_{1,\star} = \Gamma(q-2, H, q+2)$ which is called the *boundary gadget of* $\Gamma^1$. Note that $\Gamma_{1,\star}$ is again shifted one position to the left compared to all the graphs in $\Gamma^1$. We add edges between layers of $\Gamma_{1,\star}$ and layers of $\Gamma_{1,j}$, for each $j \in [n]$. Like before, we let $L$ denote some layer of $\Gamma_{1,\star}$ (ignoring the last layer) and let $L'$ be the layer after $L$ in $\Gamma_{1,j}$. If $L$ and $L'$ correspond to independent sets (not sets of an $H$) then we connect them via a matching in the natural way. Otherwise, we have again two cases:

- $|L| = n$, $|L'| = 1$, and we connect by an edge the unique vertex of $L'$ to its image in $L$; or
- $|L| = |L'| = n$ (by construction) and we connect the two layers by a matching.

We can now complete the construction as follows. For $i \in [\kappa - 1]$, after $\Gamma_{i,\star}$ is inserted we proceed just like before by assuming that $\Gamma_{i,\star}$ now takes the role of the selection gadget $\Gamma_\star$. Formally, for $i \in [\kappa - 1]$ and $j \in [n]$ (processing in increasing order), we create a graph $\Gamma_{i+1,j}$, where $\Gamma_{i+1,j} = \Gamma(q - (2i+1), H(h_j^{i+1}), q + (2i+1))$. We connect $s$ to all the first-layer vertices and $t$ to all the last-layer vertices in the obvious way. Let $\Gamma^{i+1}$ denote the collection of the $n$ graphs of the form $\Gamma_{i+1,j}$. We add edges between $\Gamma_{i,\star}$ and graphs in $\Gamma^{i+1}$ just like before. Similarly, we then add a new graph $\Gamma_{i+1,\star}$ and proceed as described until we reach $\Gamma_{\kappa,\star}$. We connect all the vertices of a layer of $\Gamma_{\kappa,\star}$ to the vertex of $Q$ on the preceding layer (see Figure 2). This completes the construction of the SPR instance $(G', s, t, P, Q)$[3]. Note that $|V(P)| = |V(Q)| = 10\kappa^2 + 2$.

**Safeness of the reduction.** Before we dive into the technical details of the proof, let us give some high-level intuition. Simply put, the purpose of every set of graphs $\Gamma^i$, $i \in [\kappa]$, is to verify that all the sets/layers of $\Gamma_\star$ mapping to the same $V_i$ use the same vertex of $V_i$. The trickier part of the proof is in showing that tokens are "well-behaved".

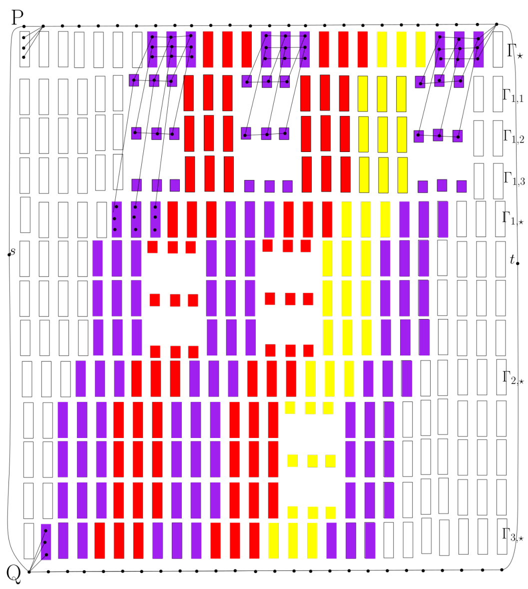


**Figure 2** An example of our reduction in the case of token jumping.

Let us start by proving the easier direction. We assume, without loss of generality, that all of our gadgets $H$ start with a copy of $V_1$ and end with a copy of $V_\kappa$. Moreover no two consecutive groups of any $H$ map to the same $V_i$.

[3] We note that most of the buffer space "to the right" of the construction is not needed but was added to favor a symmetric construction.

▶ **Lemma 7.** *If $(G,\kappa)$ is a yes-instance of (*Regular*) *Multicolored Clique* then there exists a reconfiguration sequence from $P$ to $Q$ whose length is $20(\kappa^3+\kappa^2)$.*

**Proof.** Let $\{v^1_{j_1}, v^2_{j_2}, \ldots, v^i_{j_i}, \ldots, v^\kappa_{j_\kappa}\}$ denote the vertices of a multicolored clique in $G$. Let us exhibit a reconfiguration sequence from $P$ to $Q$. To do so, let us first give a reconfiguration sequence from $P$ to a path that contains vertices in $\Gamma_\star$ as follows: We move one by one the tokens of $P$ to $\Gamma_\star$ by increasing distance to $s$ (in ascending order). For every layer $i \le q$, we jump (in order) the token at layer $i \ge 1$ in $P$ to vertex $v_{j_1}$ in the $i$th layer of $\Gamma_\star = \Gamma(q, H, q)$ as long as $i \le q+1$ (as $H_1$ maps to $V_1$ by assumption). In other words, we map all the vertices at the beginning of the path to the copy of vertex $v^1_{j_1}$. Then, for any layer $q+1 < i \le q+1+\alpha$, we jump the token at layer $i$ of $P$ to vertex $h^{\mu(i)}_{j_{\mu(i)}}$ of $\Gamma_\star$. For every $i > q+1+\alpha$, we jump the $i$th vertex of $P$ to vertex $v_{j_\kappa}$ (since we assume that $H$ ends with a set that maps to $V_\kappa$). The fact that we maintain an $st$-path after every token jump follows from Observation 6 combined with the fact that vertices of $P$ are connected to all vertices of the preceding layer of $\Gamma_\star$.

Once we have reached a properly colored $st$-path $P_1$ fully contained in $\Gamma_\star$ (in exactly $10\kappa^2$ steps), we can use a similar strategy to reach a properly colored $st$-path $P_2$ fully contained in $\Gamma_{1,j_1}$. More formally, we move by increasing order all the tokens of $P_1$ in such a way the $i$-th vertex of $P_2$ is a the copy of the $(i+1)$-th vertex of $P_1$. Note that it is well-defined since, for every $i$ such that $\mu(i) = 1$, the vertex $h_{j_1}$ belongs to $P_2$. Observe that during that transformation the vertices "shift one layer to the left". We then use a similar transformation to transform $P_2$ into a path $P_3$ fully contained in $\Gamma_{1,\star}$. We use $20\kappa^2$ steps from $P_1$ to $P_3$.

We repeat this procedure for every $2 \le i \le \kappa$ to transform the path in $\Gamma_{i-1,\star}$ into a path in $\Gamma_{i,\star}$ in $20\kappa^2$ jumps. Then we need an extra $10\kappa^2$ steps to go from $\Gamma_{\kappa,\star}$ to $Q$ (using the converse of the transformation from $P$ to $\Gamma_\star$). Hence, the length of the reconfiguration sequence is exactly $20(\kappa^3+\kappa^2)$. ◀

In order to prove the other direction, we first establish some useful properties of our construction. We let $\Gamma_\star = \Gamma_{0,0}$ and $\Gamma_{i,\star} = \Gamma_{i,n+1}$. We say $\Gamma_{i,j}$ *comes before or above* $\Gamma_{i',j'}$ whenever $i < i'$ or $i = i'$ and $j < j'$ (we also assume that $P$ appears first and $Q$ appears last, i.e., $P = \Gamma_{-1,-1}$ and $Q = \Gamma_{n+1,n+1}$). We say that two consecutive internal vertices $v_p$ and $v_{p+1}$ of an $st$-path $P$ are *siblings* if they belong to the same graph $\Gamma_{i,j}$ (that is they belong to the same row in the representation of Figure 2). Otherwise, we say $v_p$ is *above* (or *below*) $v_{p+1}$ if the graph of $v_p$ is above (below) that of $v_{p+1}$ (that is $v_p$ is in the row above or below $v_{p+1}$ in the representation of Figure 2).

▶ **Lemma 8** ($\star$)**.** *Let $P$ be a shortest path from $s$ to $t$ in $G'$. Let $v_p$ denote the pth internal vertex of $P$. Then:*

- *For every $p$, $v_p$ is a vertex of the pth layer of $G'$.*
- *For every two consecutive internal vertices of $P$, $v_p$ and $v_{p+1}$, either $v_p$ and $v_{p+1}$ are siblings or $v_p$ is below $v_{p+1}$.*
- *For every $p$, if $v_p$ belongs to $\Gamma_{i,j}$ then no vertex $v_{p'}$ with $p' \ge p$ is below $v_p$.*
- *For every $p$, if $v_p$ belongs to $\Gamma_{i,j}$ then $v_{p-1}$ is either in $\Gamma_{i,j}$ or $\Gamma_{i,n+1}$ and $v_{p+1}$ is either in $\Gamma_{i,j}$ or $\Gamma_{i-1,n+1}$.*

Our next result states that the sequence described in Lemma 7 is best possible.

▶ **Lemma 9** ($\star$)**.** *Any reconfiguration sequence from $P$ to $Q$ requires at least $20(\kappa^3+\kappa^2)$ token moves. Moreover, if there exists a reconfiguration sequence from $P$ to $Q$ then there exists one of length exactly $20(\kappa^3+\kappa^2)$.*

Given Lemma 8 and Lemma 9, it is easy to see that a shortest reconfiguration from $P$ to $Q$ in $G'$ must be *monotone*, i.e., tokens always move towards $Q$ and every path in the reconfiguration sequence consists of a sequence of vertices (ordered from $s$ to $t$) whose distance from $Q$ monotonically increases.

▶ **Lemma 10** ($\star$). *Assume that there exists a reconfiguration sequence $\sigma$ from $P$ to $Q$ in $G'$. For $i \in [\kappa]$, let $\mu^{-1}(i) = \{H_{j_1}, H_{j_2}, \ldots\}$ denote the $H$-layers (layers that belong to $H$) in $\Gamma_\star$ that map to $V_i$. Then:*

- *For every two consecutive sets $H_j$ and $H_{j+1}$ in $\Gamma_\star$ there exists at least one st-path $P'$ in the sequence $\sigma$ such that $P'$ contains one vertex in both $H_j$ and $H_{j+1}$.*
- *If $\sigma$ is a shortest sequence then the intersection of $\bigcup_{P' \in \sigma} V(P')$ with $\bigcup_{H_j \in \mu^{-1}(i)} V(H_j)$ includes only vertices that map to the same vertex of $V_i$. In other words, for any two vertices $w$ and $w'$ in $W = \bigcup_{P' \in \sigma} V(P') \cap \bigcup_{H_j \in \mu^{-1}(i)} V(H_j)$, we have $\pi(w) = \pi(w')$.*

We now have all the ingredients to finish the proof.

▶ **Lemma 11.** *If $(G', s, t, P, Q)$ is a yes-instance of* Shortest Path Reconfiguration *then $(G, \kappa)$ is a yes-instance of (*Regular*)* Multicolored Clique*.*

**Proof.** Let $(G', s, t, P, Q)$ be a yes-instance and let $\sigma$ be a shortest reconfiguration sequence from $P$ to $Q$. For $i \in [\kappa]$ and $P' \in \sigma$, let $W_i = \bigcup_{P' \in \sigma} V(P') \cap \bigcup_{H_j \in \mu^{-1}(i)} V(H_j)$. Moreover, let $\pi(W_i) = \{\pi(w) \mid w \in W_i\}$. By Lemma 10, we have $|\pi(W_i)| = 1$ and we denote the vertex by $v^i_{j_i}$. Consider the $\kappa$ vertices $\{v^1_{j_1}, \ldots, v^i_{j_i}, \ldots, v^\kappa_{j_\kappa}\}$. The fact that those vertices must form a multicolored clique in $G$ again follows from Lemma 10; as every pair must appear consecutively in two $H$-layers of $\Gamma_\star$ and some path of $\sigma$ must intersect with both. ◀

▶ **Corollary 12** ($\star$). *SPR is W[1]-hard parameterized by $k$ and SSPR is W[1]-hard parameterized by $k + \ell$ under both the token jumping and the token sliding model.*

## 3 FPT algorithms

First, we observe that both SPR and SSPR are easily shown to be fixed-parameter tractable when parameterized by $k + \Delta(G)$, where $\Delta(G)$ denotes the maximum degree of $G$; by only retaining vertices that belong to some shortest *st*-path one can easily bound the size of the graph since the $i$-th layer, consisting of all the vertices at distance exactly $i$ from $s$, will contain at most $\Delta(G)^i$ vertices. In the remainder of this section, we investigate the complexity of the problem further (and for different parameters) in order to identify the boundary between tractability and intractability. As a warm-up, let us first prove that the following holds:

▶ **Lemma 13.** *SSPR is* FPT *parameterized by $k + \ell$ on nowhere-dense classes of graphs for both the sliding and the jumping models.*

**Proof.** The proof easily follows from the fact that FO-model checking is FPT on nowhere dense classes of graphs [11]. Such an argument has already been used in various proofs for reconfiguration problems, see e.g., [6].

For every $i \le k$ and $j \le \ell$, let us create a variable $x_{i,j}$ that represents the $i$-th vertex of the path at the $j$-th step of the reconfiguration sequence. Let us prove that we can formulate the existence of a reconfiguration sequence of length $\ell$ between $P$ and $Q$ as a FO-formula on the set of variables $x_{i,j}$. First we set $x_{i,1} = p_i$ where $p_i$ is the $i$-th vertex of the path $P$. Similarly $x_{i,\ell} = q_i$ where $q_i$ is the $i$-th vertex of the path $Q$. We now need to ensure that at every step $j \le \ell$, the set of variables $x_{i,1}, \ldots, x_{i,\ell}$ is a path of $G$, that is, for every $i \le k - 1$ and every $j \le \ell$, $x_{i,j} x_{i+1,j}$ is an edge.

We further want to ensure that if one vertex is modified between the $j$-th path and the $(j+1)$-th path then all the other vertices are the same. That is, for every $i, i' \le k$ and $j \le \ell - 1$, we have $(x_{i,j} \ne x_{i,j+1} \Rightarrow (x_{i',j} = x_{i',j+1})$. If we want a reconfiguration sequence with the token sliding rule, we have to add the following constraint: for every $i \le k, j \le \ell - 1$, $x_{i,j} = x_{i,j+1}$ or $x_{i,j}x_{i,j+1}$ is an edge. Finally, we add the constraints $x_{1,j} = s$ and $x_{k,j} = t$ for every $j \le \ell$.

Let us denote by $\phi$ the resulting formula. Let us prove that there exists a reconfiguration sequence from $P$ to $Q$ of length at most $\ell$ if and only if $\phi$ is satisfiable. If there exists a reconfiguration sequence $P_1 = P, \dots, P_r = Q$ with $r \le \ell$ then we simply have to set $x_{i,j}$ to be the $i$-th vertex of $P_j$ and $x_{i,j'} = q_i$ for every $j' \ge r$ in order to satisfy all the constraints.

Conversely, assume that there exists an assignment of the variables that satisfies all the constraints. Let us denote by $P_j$ the set of ordered vertices $x_{i,j}$ for $1 \le i \le k$. Note that, by hypothesis, $P_j$ is an $st$-path for every $j$. Moreover, by definition $P_j$ and $P_{j+1}$ differ on at most one vertex and $P_1 = P$ and $P_\ell = Q$. By removing consecutive paths that are the same we obtain a reconfiguration sequence from $P$ to $Q$, which completes the proof. ◀

Let us now generalize the previous result and prove that the following holds:

▶ **Theorem 14.** *SSPR is* FPT *parameterized by $\ell$ on nowhere dense classes of graphs for both the sliding and the jumping models.*

**Proof.** The idea of the proof consists of proving that there exists an equivalent instance where the distance between $s$ and $t$ is bounded by a function of $\ell$. The conclusion then directly follows from Lemma 13. To do so, we will prove that we can bound (by a function of $\ell$) the set of indices $i$ on which there is a relevant modification on the $i$-th vertex of the path at some step of the reconfiguration sequence. We will then prove that we can “forget” the vertices which are not in these positions by reducing the length of the shortest paths.

Let $(G, s, t, P, Q, \ell)$ be an instance of SSPR. Let us denote by $S$ the set of positions on which $P$ and $Q$ differ. Note that if $|S| > \ell$ then we can immediately return false since more than $\ell$ steps are needed to transform $P$ into $Q$. So we can assume that $|S| \le \ell$ in the rest of the proof.

▷ Claim 15. If there is a reconfiguration sequence from $P$ to $Q$ of length at most $\ell$ then there is a reconfiguration sequence from $P$ to $Q$ that only modifies vertices whose indices are at distance at most $\ell$ from an index of $S$.

Proof. Let $\mathcal{R}$ be a reconfiguration sequence from $P$ to $Q$ of length at most $\ell$. At each step, there is exactly one position where a vertex is modified. Let us denote by $R$ that set of positions where a vertex is modified. We have $|R| \le \ell$. A *component* $R'$ of $R$ is a maximal subset of $R$ containing consecutive integers. Every component $R'$ has a minimum and a maximum value (that might be equal). We say that a component is *important* if it contains a vertex of $S$ and *useless* otherwise.

We claim that if there is a useless component $R'$, removing from $\mathcal{R}$ all the modifications at position $c$ for every $c \in R'$ leaves a reconfiguration sequence from $P$ to $Q$. Indeed, let us denote by $\mathcal{R}'$ the resulting reconfiguration sequence. First note that since $R'$ is a useless component, the final shortest path is still $Q$ (we cancel modifications on positions where $P$ and $Q$ were identical). Assume now, for a contradiction, that at some step of the reconfiguration sequence in $\mathcal{R}'$, the set of vertices $P_i$ is not a shortest $st$-path. Let us denote by $u, v$ the consecutive vertices of $P_i$ that are not adjacent. Since the path is only modified at positions of indices of $R'$, either the index of $u$ or $v$ is in $R'$. Moreover, both of them are not in $R'$ since by definition of $\mathcal{R}'$ all the vertices of indices in $R'$ remain the same all along the reconfiguration sequence and the initial set of vertices is indeed a path. So we can assume by symmetry that the position of $u$ is the index just before the minimum value of $R'$ and $v$ is the minimum value of $R'$. Since the vertex $v$ belongs to all the sets in the reconfiguration sequence $\mathcal{R}'$, it means that $u$ has been modified. But then $u$ should be added in the component $R'$ of $v$, a contradiction.

Thus, if there is a reconfiguration sequence from $P$ to $Q$ of length $\ell$, there is one with no useless component. But the width of a component is at most $\ell$ since only $\ell$ vertices are modified in a reconfiguration sequence. So if there is a reconfiguration sequence, there is one that only moves tokens on vertices whose indices are at distance at most $\ell$ from an index of $S$, as claimed. ◁

Let $X(i, s)$ be the set of vertices at distance exactly $i$ from $s$ in $G$. Let $I_S$ be the set of indices at distance at most $\ell$ from an index of $S$. Note that $I_S$ has size at most $2\ell \cdot |S|$. An empty interval for $I_S$ is an interval maximal by inclusion in $\{0, \ldots, d(s,t)\} \setminus I_S$. Note that $I_S$ has at most $|S|$ empty intervals. We create the graph $G'$ from $G$ as follows:

- $G'$ contains $s, t$ and, for every $i \in I_S$, $G'$ contains all the vertices of $X(i, s)$.
- For all the integers $i \notin I_S$ but at distance one from an integer of $I_S$, $G'$ contains the vertex at position $i$ in $P$ (and $Q$).
- There is an edge between $x$ and $y$ if $xy$ is an edge of $G$, or if $x, y$ are the unique two vertices of $G$ whose positions are in the same empty interval for $I_S$ [4].

Let us denote by $P'$ and $Q'$ in $G'$ the set $P \cap V(G')$ and $Q \cap V(G')$. One can easily remark that $P'$ and $Q'$ are shortest $st$-paths in $G'$.

▷ Claim 16. There is a reconfiguration sequence from $P$ to $Q$ in $G$ if and only if there is a reconfiguration sequence from $P'$ to $Q'$ in $G'$.

Proof. The proof follows from the fact that we can assume that a transformation from $P$ to $Q$ of length at most $\ell$ in $G$ only modifies vertices whose indices are at distance at most $\ell$ from an index of $S$. All those vertices are in $G'$ and all the vertices of $G'$ that contain non-movable tokens are unique at their corresponding distance from $s$ (hence cannot move in $G'$). ◁

One can remark that the distance between $s$ and $t$ in $G'$ is at most $4\ell^2$. So by Lemma 13, we can decide in FPT-time in $\ell$ if there is a reconfiguration sequence from $P'$ to $Q'$ in $G'$, which completes the proof. ◀

## References


1 John Asplund, Kossi D. Edoh, Ruth Haas, Yulia Hristova, Beth Novick, and Brett Werner. Reconfiguration graphs of shortest paths. *Discret. Math.*, 341(10):2938–2948, 2018. doi:10.1016/J.DISC.2018.07.007.

2 Valentin Bartier, Nicolas Bousquet, and Amer E. Mouawad. Galactic token sliding. *J. Comput. Syst. Sci.*, 136:220–248, 2023. doi:10.1016/J.JCSS.2023.03.008.

3 Paul S. Bonsma. The complexity of rerouting shortest paths. *Theor. Comput. Sci.*, 510:1–12, 2013. doi:10.1016/j.tcs.2013.09.012.

4 Paul S. Bonsma. Rerouting shortest paths in planar graphs. *Discret. Appl. Math.*, 231:95–112, 2017. doi:10.1016/j.dam.2016.05.024.


[4] Informally, we link the vertex of $P$ just after an interval of $I_S$ with the vertex of $P$ just before the beginning of the next interval of $I_S$.

5 Nicolas Bousquet, Felix Hommelsheim, Yusuke Kobayashi, Moritz Mühlenthaler, and Akira Suzuki. Feedback vertex set reconfiguration in planar graphs. *Theor. Comput. Sci.*, 979:114188, 2023. doi:10.1016/j.tcs.2023.114188.

6 Nicolas Bousquet, Amer E. Mouawad, Naomi Nishimura, and Sebastian Siebertz. A survey on the parameterized complexity of the independent set and (connected) dominating set reconfiguration problems. *CoRR*, abs/2204.10526, 2022. doi:10.48550/arXiv.2204.10526.

7 Erik D. Demaine, David Eppstein, Adam Hesterberg, Kshitij Jain, Anna Lubiw, Ryuhei Uehara, and Yushi Uno. Reconfiguring undirected paths. In *Algorithms and Data Structures - 16th International Symposium, WADS 2019, Edmonton, AB, Canada, August 5-7, 2019, Proceedings*, pages 353–365, 2019. doi:10.1007/978-3-030-24766-9_26.

8 Naoki Domon, Akira Suzuki, Yuma Tamura, and Xiao Zhou. The shortest path reconfiguration problem based on relaxation of reconfiguration rules. In Ryuhei Uehara, Katsuhisa Yamanaka, and Hsu-Chun Yen, editors, *WALCOM: Algorithms and Computation - 18th International Conference and Workshops on Algorithms and Computation, WALCOM 2024, Kanazawa, Japan, March 18-20, 2024, Proceedings*, volume 14549 of *Lecture Notes in Computer Science*, pages 227–241. Springer, 2024. doi:10.1007/978-981-97-0566-5_17.

9 Kshitij Gajjar, Agastya Vibhuti Jha, Manish Kumar, and Abhiruk Lahiri. Reconfiguring shortest paths in graphs. In *Thirty-Sixth AAAI Conference on Artificial Intelligence, AAAI 2022, Virtual Event, February 22 - March 1, 2022*, pages 9758–9766, 2022. doi:10.1609/aaai.v36i9.21211.

10 Sevag Gharibian and Jamie Sikora. Ground state connectivity of local hamiltonians. In *Automata, Languages, and Programming - 42nd International Colloquium, ICALP 2015, Kyoto, Japan, July 6-10, 2015, Proceedings, Part I*, volume 9134 of *Lecture Notes in Computer Science*, pages 617–628. Springer, 2015. doi:10.1007/978-3-662-47672-7_50.

11 Martin Grohe, Stephan Kreutzer, and Sebastian Siebertz. Deciding first-order properties of nowhere dense graphs. In David B. Shmoys, editor, *Symposium on Theory of Computing, STOC 2014, New York, NY, USA, May 31 - June 03, 2014*, pages 89–98. ACM, 2014. doi:10.1145/2591796.2591851.

12 Siddharth Gupta, Guy Sa'ar, and Meirav Zehavi. The parameterized complexity of motion planning for snake-like robots. *J. Artif. Intell. Res.*, 69:191–229, 2020. doi:10.1613/JAIR.1.11864.

13 Takehiro Ito, Erik D. Demaine, Nicholas J. A. Harvey, Christos H. Papadimitriou, Martha Sideri, Ryuhei Uehara, and Yushi Uno. On the complexity of reconfiguration problems. *Theor. Comput. Sci.*, 412(12-14):1054–1065, 2011. doi:10.1016/j.tcs.2010.12.005.

14 Takehiro Ito, Marcin Kaminski, and Erik D. Demaine. Reconfiguration of list edge-colorings in a graph. *Discret. Appl. Math.*, 160(15):2199–2207, 2012. doi:10.1016/j.dam.2012.05.014.

15 Anna Lubiw and Vinayak Pathak. Flip distance between two triangulations of a point set is NP-complete. *Comput. Geom.*, 49:17–23, 2015. doi:10.1016/j.comgeo.2014.11.001.

16 Naomi Nishimura. Introduction to reconfiguration. *Algorithms*, 11(4):52, 2018. doi:10.3390/a11040052.

17 Krzysztof Pietrzak. On the parameterized complexity of the fixed alphabet shortest common supersequence and longest common subsequence problems. *Journal of Computer and System Sciences*, 67(4):757–771, 2003. doi:10.1016/S0022-0000(03)00078-3.

18 Jan van den Heuvel. The complexity of change. In Simon R. Blackburn, Stefanie Gerke, and Mark Wildon, editors, *Surveys in Combinatorics 2013*, volume 409 of *London Mathematical Society Lecture Note Series*, pages 127–160. Cambridge University Press, 2013. doi:10.1017/CBO9781139506748.005.

19 Marcin Wrochna. Reconfiguration in bounded bandwidth and tree-depth. *J. Comput. Syst. Sci.*, 93:1–10, 2018. doi:10.1016/J.JCSS.2017.11.003.