The Complexity of Rerouting Shortest Paths[☆]

Paul Bonsma[1]

Humboldt University Berlin, Computer Science Department, Unter den Linden 6, 10099 Berlin, Germany.

Abstract

The Shortest Path Reconfiguration problem has as input a graph G with unit edge lengths, with vertices s and t, and two shortest st-paths P and Q. The question is whether there exists a sequence of shortest st-paths that starts with P and ends with Q, such that subsequent paths differ in only one vertex. This is called a rerouting sequence.

This problem is shown to be PSPACE-complete. For claw-free graphs and chordal graphs, it is shown that the problem can be solved in polynomial time, and that shortest rerouting sequences have linear length. For these classes, it is also shown that deciding whether a rerouting sequence exists between *all* pairs of shortest st-paths can be done in polynomial time. Finally, a polynomial time algorithm for counting the number of isolated paths is given.

Keywords: shortest path, reconfiguration, reachability, PSPACE-hard, claw-free graph, chordal graph

1. Introduction

In this paper, we study the *Shortest Path Reconfiguration (SPR) Problem*, introduced by Kamiński et al [16, 17]. The input consists of a graph G, with vertices s and t, and two shortest st-paths P and Q. The question is whether P can be modified to Q by changing one vertex at a time, and maintaining a shortest st-path throughout. Edges have unit lengths, so all shortest st-paths have the same number of vertices. We define the following *solution graph* $\mathrm{SP}(G, s, t)$: its vertex set is the set of all shortest st-paths in G. Two paths P and Q are adjacent if they differ in one vertex, so $|V(P)\backslash V(Q)| = 1$ (and thus $|V(Q)\backslash V(P)| = 1$). SPR can now be reformulated as: does there exist a walk from P to Q in $\mathrm{SP}(G, s, t)$? Such a walk is also called a *rerouting sequence*.

[☆]An extended abstract of this paper appeared in the proceedings of MFCS 2012. This is the author's version of the accepted journal paper. The published version is available at `http://dx.doi.org/10.1016/j.tcs.2013.09.012`.

[1]*Email adress:* `bonsma@informatik.hu-berlin.de`. Current address: University of Twente, Faculty of EEMCS, PO Box 217, 7500 AE Enschede, the Netherlands.

Shortest paths form a central concept in graph theory, optimization, algorithms and networking. Questions related to rerouting (shortest) paths are often studied in networking applications. Although we are not aware of an application where this reachability question is studied, it is a very natural question, and its study may provide insight to practical rerouting problems. Nevertheless, the main motivation for this research is of a more theoretical nature. Similar reconfiguration problems can be defined based on many different combinatorial problems: Consider all solutions to a problem (or all solutions of at least/at most given weight, in the case of optimization problems), and define a (symmetric) adjacency relation on them. Such problems have been studied often in recent literature. Examples include reconfiguration problems based on satisfiability problems [11], independent sets [12, 14, 18], vertex colorings [1, 4, 5, 6, 7], matchings [14], list edge-colorings [15], matroid bases [14], subsets of a (multi)set of numbers [10]. Of course, to obtain a reconfiguration problem, one needs to define an adjacency relation between solutions. Usually, the most natural adjacency relation is considered, e.g. two independent sets I and J are considered adjacent in [14] if J can be obtained from I by removing one vertex and adding another; boolean assignments are considered adjacent in [11] if exactly one variable differs, etc. We remark that in the context of local search, similar problems have been studied earlier, with the important distinction that the neighborhood is not symmetric, and the objective is to reach a local optimum, instead of a given target solution, see e.g. [19].

An initial motivation of these questions was to explore the solution space of NP-hard problems, to study e.g. the performance of heuristics [11] and random sampling methods [5]. This has revealed interesting, often recurring patterns in the complexity behavior of these problems. This is perhaps best exemplified by the known results on the reconfiguration of vertex colorings using k colors: in the problem k-Color Path, two k-colorings of a graph are given, and the question is whether one can be modified to the other by changing one vertex color at a time, and maintaining a k-coloring throughout. This problem is polynomial time solvable for $k \leq 3$ [7], and PSPACE-complete for $k \geq 4$ [4]. Note that the corresponding decision problem of deciding whether a graph admits a k-coloring is polynomial time solvable for $k \leq 2$, and NP-complete for $k \geq 3$. This gives an example of the following common pattern: for instance classes for which deciding whether a solution exists is in P, the reconfiguration problem is often in P as well. See [11, 13, 14] for more extensive examples. This motivated Ito et al [13] to ask for examples of reconfiguration problems that break this pattern. Secondly, it has been observed that there is a strong correlation between the complexity of reconfiguration problems and the diameter of the components of the solution graph: for all known 'natural' reconfiguration problems in P, the diameter is polynomially bounded (see e.g. [1, 7, 11, 14, 18]), and for all PSPACE-complete reconfiguration problems, the diameter may be superpolynomial or exponential (see e.g. [4, 11]). The latter is unsurprising, since polynomial diameter would imply NP=PSPACE (assuming that the property of being a solution and adjacency of solutions can be tested in polynomial time, which holds for all aforementioned problems). One can easily construct artificial instance classes of reconfiguration problems such that the problem is in P, but has exponential diameter [4], but to our knowledge no natural examples are known. (That is, not constructed specifically to prove something about the reconfiguration problem at hand.)

With the goal of breaking one of these patterns, Kamiński et al [16, 17] introduced the SPR problem. Finding a shortest path can be done in polynomial time. Nevertheless, in [16, 17] examples were constructed where the solution graph has exponential diameter. This shows that regardless of whether SPR is in P or PSPACE-complete, one of the patterns is broken. The main open question from [17] was therefore that of determining the complexity of SPR.

In this paper, we answer that question by showing that SPR is PSPACE-complete. Therefore, this also answers the question posed in [13], by giving a rare example of a PSPACE-complete reconfiguration problem based on a decision problem in P. We remark that it is not the first example: in [4] it is shown that 4-Color Path is also PSPACE-hard for bipartite graphs. Since every bipartite graph is 2-colorable, the corresponding decision problem is trivial. Our PSPACE-completeness result is presented in Section 3. We remark that our PSPACE-completeness result, after it appeared in a preprint [2], has already proved its usefulness for showing PSPACE-completeness of other problems: in [18], the result has been applied to show that Independent Set Reconfiguration remains PSPACE-hard even when restricted to perfect graphs.

Furthermore, we give the following positive results on SPR in this paper: we show that when G is chordal or claw-free, SPR can be decided in polynomial time. A graph is *chordal* if it contains no induced cycle of length more than 3. This is a well-studied class of perfect graphs, which includes for instance k-trees and interval graphs [9]. A graph is *claw-free* if it contains no induced $K_{1,3}$ subgraph. This is again a well-studied graph class, see e.g. [8]. We also show that for these graph classes, the diameter of components of $\mathrm{SP}(G,s,t)$ is always linearly bounded. For claw-free graphs G, we show that if there exists a rerouting sequence from P to Q, then in polynomial time we can find one of length at most $2n+2d-6$, where $n=|V(G)|$ and d is the length of P and Q. For chordal graphs, we show that in polynomial time, we can find a rerouting sequence of length $|V(P)\backslash V(Q)|$. Hence we can actually find a *shortest* rerouting sequence efficiently in this case. In contrast, in [16], it was shown that for general graphs, finding a shortest rerouting sequence is NP-hard, even for graph classes where there always exists one of polynomial length. Recently, a positive result for SPR similar to the results in this paper has been found: in [3], it is shown that for planar graphs, SPR can be decided in polynomial time.

In the context of reconfiguration problems, other types of questions are commonly studied as well. Above, we considered the *reachability question*: can one given solution be reached from another given solution? The related *connectivity question* has also been well-studied [5, 6, 10, 11]: can every solution reach every other solution? In other words, is the solution graph connected? For chordal graphs G, we answer affirmatively: we show that $\mathrm{SP}(G,s,t)$ is always connected. Furthermore, we show that if G is claw-free, it can be decided in polynomial time whether $\mathrm{SP}(G,s,t)$ is connected. Our results on chordal graphs are presented in Section 4, and the results on claw-free graphs in Section 5.

Another type of question that has been studied in this context is related to the existence of isolated states [10]. In the case of SPR, an *isolated st-path* is a shortest st-path in G that has no neighbor in $\mathrm{SP}(G, s, t)$. The reader may observe that deciding whether a given path is an isolated st-path is a trivial problem, that can be decided in linear time. Similarly, deciding whether all shortest st-paths are isolated can trivially be done in polynomial time as well. The problem of deciding whether there exists an isolated st-path is less trivial. In Section 6 we give an algorithm for this problem. In fact, we give a polynomial time algorithm for the more general problem of *counting* the number of isolated paths. In Section 7, we end with a discussion.

2. Preliminaries

For graph theoretical notions not defined here, we refer to [9]. We will consider undirected and simple graphs throughout. A *walk* of length k from v_0 to v_k in a graph G is a vertex sequence $v_0, \ldots, v_k$, such that for all $i \in \{0, \ldots, k-1\}$, $v_i v_{i+1} \in E(G)$. It is a *path* if all vertices are distinct. It is a *cycle* if $k \geq 3$, $v_0 = v_k$, and $v_0, \ldots, v_{k-1}$ is a path. With a path or cycle $W = v_0, \ldots, v_k$ we associate a subgraph of G as well, with vertex set $V(W) = \{v_0, \ldots, v_k\}$ and edge set $E(W) = \{v_i v_{i+1} \mid i \in \{0, \ldots, k-1\}\}$. A path from s to t is also called an *st-path*. The *distance* from s to t is the length of a *shortest st-path*. The *diameter* of a graph is the maximum distance from s to t over all vertex pairs s, t.

A hypergraph $H = (V, E)$ consists of a vertex set V, and a set E of hyperedges, which are subsets of V. A walk in H of length k is a sequence of vertices $v_0, \ldots, v_k$ such that for every i, there exists a hyperedge $e \in E$ with $\{v_i, v_{i+1}\} \subseteq e$. Using this notion of walks, the notions of connectivity and components of hypergraphs are defined the same as for graphs.

Throughout this paper, we will consider a graph G with vertices $s, t \in V(G)$. We will only be interested in shortest st-paths in G, and use d to denote their length. For $i \in \{0, \ldots, d\}$, we define $L_i \subseteq V(G)$ to be the set of vertices that lie on a shortest st-path, at distance i from s. So $L_0 = \{s\}$, and $L_d = \{t\}$ (even if there may be more vertices at distance d of s). A set L_i is also called a *layer*. With respect to a given layer L_i, the *previous layer* is L_{i-1}, and the *next layer* is L_{i+1}. Clearly, if there is an edge $xy \in E(G)$ with $x \in L_i$ and $y \in L_j$, then $|j - i| \leq 1$. Note that a shortest st-path P contains exactly one vertex from every layer. For $i \in \{0, \ldots, d\}$, this vertex will be called the *L_i-vertex* of P.

The graph G will be undirected, so we use the notation $N(v)$ to denote the set of neighbors of a vertex $v \in V(G)$. However, if $v \in L_i$, then we will use $N^-(v)$ to denote $N(v) \cap L_{i-1}$, and call these neighbors the *in-neighbors of* v. Similarly, $N^+(v)$ denotes $N(v) \cap L_{i+1}$, and these are called the *out-neighbors of* v.

Recall that for two shortest st-paths P and Q, a rerouting sequence from P to Q is a walk in $\mathrm{SP}(G, s, t)$ from P to Q. So this is a sequence $Q_0, \ldots, Q_k$ of shortest st-paths with $Q_0 = P$, $Q_k = Q$, such that for every $j \in \{0, \dots, k-1\}$, Q_j and Q_{j+1} differ in exactly one vertex. Since these are all shortest st-paths, this implies that there is a unique layer L_i such that the L_i-vertices of Q_j and Q_{j+1} differ. If these L_i-vertices are u and v respectively, then we also say that Q_{j+1} is obtained from Q_j with a rerouting step $u \to v$ (in layer L_i). Observe that the entire rerouting sequence can be deduced from the knowledge of the starting path Q_0, and the rerouting steps from Q_j to Q_{j+1} for every j. So for short, we will also describe rerouting sequences from a given starting path by just giving the sequence of rerouting steps. We will often use the following basic observation: Let Q be a shortest st-path with L_i-vertex u, and let v be another L_i-vertex. Then a rerouting step $u \to v$ is possible if and only if v is adjacent to both the L_{i-1}-vertex of Q and the L_{i+1}-vertex of Q.

3. PSPACE-completeness

In this section we prove that the SPR problem is PSPACE-complete. We first define the problem that we reduce from. A *k-color assignment* α for a graph G is a function $\alpha : V(G) \to \{1, \dots, k\}$. A *$k$-coloring* α for a graph G is a color assignment such that for all $uv \in E(G)$, $\alpha(u) \neq \alpha(v)$. For a given graph G, the *k-color graph* $\mathcal{C}_k(G)$ has as vertex set all k-colorings of G, where two colorings are adjacent if they differ only in one vertex. A walk in $\mathcal{C}_k(G)$ from α to β will also be called a *recoloring sequence* from α to β. The problem *k-Color Path* is defined as follows.

k-Color Path:
INSTANCE: Graph G, two k-colorings α and β of G.
QUESTION: Is there a walk between α and β in $\mathcal{C}_k(G)$?

This problem has been shown to be PSPACE-complete for $k \geq 4$, in [4]. In Section 3.1 we give the transformation from an instance G, α, β of 4-Color Path to an instance G', P_α, P_β of SPR. In Section 3.2 we prove that these instances are equivalent; there is a recoloring sequence between α and β if and only if there is a rerouting sequence between P_α and P_β. This shows that SPR is PSPACE-hard.

3.1. Construction

Let G be a graph with two 4-colorings α and β, and $V(G) = \{v_1, \dots, v_n\}$. This is an instance of 4-Color Path. In this section we will use G to construct an equivalent SPR instance G' with two shortest st-paths P_α and P_β. Every shortest st-path in G' will correspond to a 4-color assignment for G (though not necessarily a 4-coloring!). To indicate this correspondence, some vertices of G' will be colored with the four colors $\{1, 2, 3, 4\}$. The other vertices will be colored with a fifth color, namely *black*. Note that this 5-color assignment for G' will not be a coloring of G'.

G' will consist of one *main strand*, which contains the paths P_α and P_β, and $6n$ *recoloring strands*: one for every combination of a vertex $v_i \in V(G)$ and two

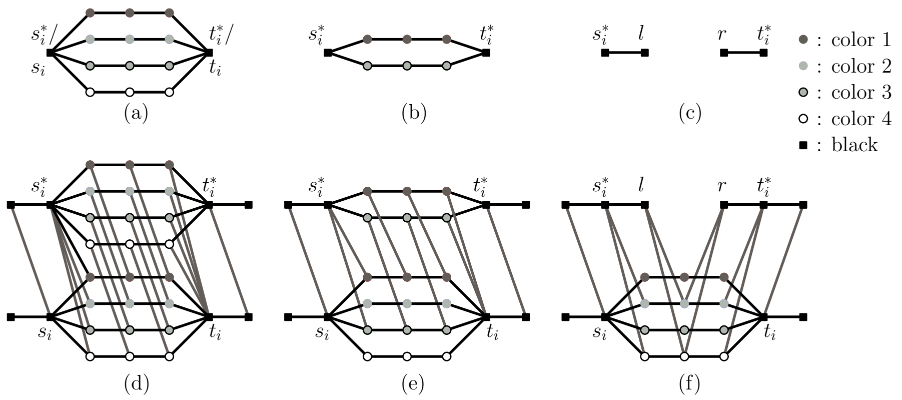

Figure 1: Gadgets H_i and H_i^* used in the construction, and edges between them.

colors $\{c_1, c_2\} \subset \{1, 2, 3, 4\}$. The recoloring strand for vertex v_i and colors c_1, c_2 will be used for rerouting paths in a way that will correspond to recoloring v_i from color c_1 to c_2, or from c_2 to c_1.

The construction of G' starts by introducing the vertices s and t. The *main strand* is constructed as follows. For each $v_i \in V(G)$, introduce a vertex gadget H_i as shown in Figure 1 (a). The leftmost vertex of H_i is labeled s_i, and the rightmost vertex t_i. These vertices are colored black. H_i consists of four disjoint s_it_i-paths of length 4, one for each color. All internal vertices of the paths are colored in the color assigned to the path. The four vertices of H_i that are neither adjacent to s_i nor to t_i are called *middle vertices* of H_i. These gadgets H_i are connected as follows: add edges ss_1 and t_nt, and for every $i \in \{1, \dots, n-1\}$, add an edge t_is_{i+1}.

At this point the graph is connected, and every vertex lies on a shortest st-path. Observe that the distance from s to s_i (resp. t_i) is $5i - 4$ (resp. $5i$), and the distance from s to t is $5n + 1$. So for every vertex v, this determines uniquely the layer L_i such that $v \in L_i$ (with $i \in \{0, \dots, 5n+1\}$).

Now we show how the *recoloring strands* are constructed. For each $v_i \in V(G)$ and each color pair $\{c_1, c_2\} \subset \{1, 2, 3, 4\}$, we introduce a recoloring strand called the $v_i, \{c_1, c_2\}$*-strand*, defined as follows. Let $\{1, 2, 3, 4\} \backslash \{c_1, c_2\} = \{c_3, c_4\}$. First we introduce gadgets H_j^* for every $j \in \{1, \dots, n\}$. (For every recoloring strand, we will introduce gadgets H_j^*, for $j = 1, \dots, n$. Whenever we mention H_j^* gadgets below, this should be interpreted as the H_j^**-gadgets for the* $v_i, \{c_1, c_2\}$*-strand*. The same holds for the vertices s_j^*, t_j^*, l and r that we will introduce below for every strand.)

- If $j \neq i$ and $v_iv_j \notin E(G)$, then define H_j^* to be isomorphic to H_j (see Figure 1 (a)), with the same 5-color assignment. The leftmost and rightmost (black) vertices are now labeled s_j^* and t_j^* respectively.

- If $j \neq i$ and $v_i v_j \in E(G)$, then define H_j^* to be as shown in Figure 1 (b). The leftmost and rightmost (black) vertices are labeled s_j^* and t_j^* again. Now there are only two disjoint paths from s_j^* to t_j^*, which are colored with the colors c_3 and c_4.

- H_i^* is the gadget shown in Figure 1 (c). Here s_i^* has one neighbor labeled l, and t_i^* has one neighbor labeled r.

Complete the strand by adding edges ss_1^*, t_n^*t and $t_i^*s_{i+1}^*$ for every $i \in \{1, \ldots, n-1\}$. Note that if we add edges from l and r to the same vertex in layer L_{5i-2}, which we will do below, then all vertices of the new strand lie on st-paths of length $5n+1$ as well, and no shorter st-paths have been created. This defines for every vertex in the new strand which distance layer it is part of. We will refer to these layers in the next step, where we show how to connect the vertices of this recoloring strand to the main strand, see Figure 1 (d)–(f). For all $j < i$:

- Add edges between s_j^* and every main-strand vertex in the next layer that has a color that is also used in H_j^*.

- For every non-black vertex v of H_j^*, add an edge between v and the main-strand vertex in the next layer that has the same color as v, or is black.

- Add an edge $t_j^*s_{j+1}$.

Similarly, for all $j > i$:

- Add edges between t_j^* and every main-strand vertex in the previous layer that has a color that is also used in H_j^*.

- For every non-black vertex v of H_j^*, add an edge between v and the main-strand vertex in the previous layer that has the same color as v, or is black.

- Add an edge $s_j^*t_{j-1}$.

For H_i^* we add edges as follows.

- Connect s_i^* to the main strand vertices in the next layer with colors c_1 and c_2.

- Connect t_i^* to the main strand vertices in the previous layer with colors c_1 and c_2.

- Connect both remaining vertices l and r of H_i^* to both middle vertices of H_i that have colors c_1 and c_2.

Introducing such a $v_i, \{c_1, c_2\}$*-strand* for every $v_i \in V(G)$ and $\{c_1, c_2\} \subset \{1, 2, 3, 4\}$ completes the construction of G'.

Finally, we show how to construct a path P_γ for any given 4-coloring γ of G, see Figure 2. The path P_γ contains only main strand vertices. Since it should

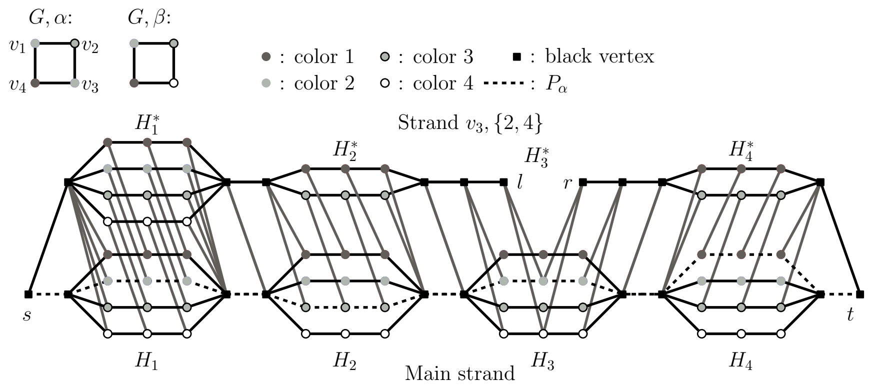

Figure 2: A k-Color Path instance G, α, β, and two strands of the resulting graph G'.

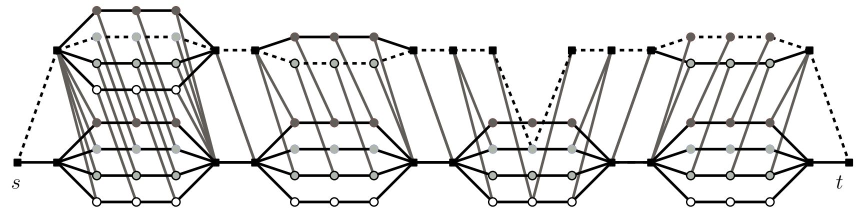

Figure 3: An intermediate path in a rerouting sequence from P_α to P_β, using the $v_3, \{2, 4\}$-strand.

be a shortest st-path, it contains exactly one vertex of every layer. Every layer contains vertices of a unique gadget H_i of the main strand. In the case that the layer contains a single black vertex from H_i, this is the vertex that is included in P_γ. In the case that the layer contains vertices of colors $1, \dots, 4$ of H_i, use the vertex of color $\gamma(v_i)$ for P_γ. This way, we define the paths P_α and P_β, using the given colorings α and β, respectively.

For an example of the construction see Figure 2. Here G is a cycle on four vertices. Two colorings α and β are shown, which differ only in vertex v_3, $\alpha(v_3) = 2$ and $\beta(v_3) = 4$. A part of the resulting graph G' is illustrated: only the main strand and the $v_3, \{2, 4\}$ strand are shown. The marked path in G' is P_α.

3.2. Equivalence of the instances

We first show that if γ and δ are adjacent colorings in $\mathcal{C}_k(G)$, which differ in vertex v_i, then the $v_i, \{\gamma(v_i), \delta(v_i)\}$-strand can be used to reroute the path P_γ to P_δ.

Lemma 1. *If there is a recoloring sequence for G from α to β, then there is a rerouting sequence from P_α to P_β for G'.*

Proof: It suffices to show that for any two adjacent colorings γ and δ in $\mathcal{C}_k(G)$, there is a rerouting sequence from P_γ to P_δ, where P_γ and P_δ are the shortest st-paths in G' that are constructed using γ and δ as explained at the end of Section 3.1. If this can be done for every consecutive pair in the recoloring sequence from α to β, then a rerouting sequence from P_α to P_β exists. So, let γ and δ be adjacent colorings in $\mathcal{C}_k(G)$. Let v_i be the unique vertex in which they differ, and let $c_1 = \gamma(v_i)$ and $c_2 = \delta(v_i)$.

P_γ can be transformed into P_δ as follows: Let H_j^* denote the gadgets of the $v_i, \{c_1, c_2\}$-strand of G'. First, for the layers $d = 1, \ldots, 5i-3$, replace the vertex v of P_γ in layer L_d by the unique vertex of H_j^* (in the same layer) that has the same color as v. This is possible by making the changes in increasing layer order. Similarly, for the layers $d = 5n, 5n-1, \ldots, 5i-1$, replace the vertex v in layer L_d by the vertex of H_j^* with the same color as v. This is possible by making the changes in decreasing layer order. (When starting with the path shown in Figure 2, this gives the path shown in Figure 3.) Note that these changes are possible if and only if P_γ does not use vertices of color c_1 or c_2 from gadgets H_j with $v_iv_j \in E(G)$. The latter property is ensured by the construction of P_γ, since γ is a coloring of G, so all neighbors v_j of v_i have $\gamma(v_j) \notin \{c_1, c_2\}$.

Now we can change the middle vertex of H_i that is used in the path: replace the middle vertex of color c_1 with the one of color c_2. Next, we can move the entire path from the $v_i, \{c_1, c_2\}$-strand back to the main strand, similar to before (but in reverse order). This yields a rerouting sequence from P_γ to P_δ. Since we can do this for every recoloring step in the recoloring sequence (there is a strand for every v_i and every $\{c_1, c_2\}$), this concludes the proof. $\square$

Loosely speaking, we now show that any rerouting sequence from P_α to P_β must consist of a sequence of rerouting sequences that are of the type given in the previous proof. This establishes the converse of the above lemma.

Lemma 2. *If there is a rerouting sequence from P_α to P_β for G', then there is a recoloring sequence in G from α to β.*

Proof: First we define how *any* shortest st-path P in G' is mapped to a color assignment of G: v_i receives the same color as the vertex of P in layer L_{5i-2}; this is the layer that contains the middle vertices of H_i. Note that this defines a color assignment for G, but that this is not necessarily a (proper) coloring.

In a rerouting sequence from P_α to P_β, consider a step where the sequence moves from a path P that corresponds to a color assignment γ, to a path P' that corresponds to a different color assignment δ. We will prove that if γ is a coloring of G, then δ is a coloring of G as well. (So then γ and δ are adjacent vertices in $\mathcal{C}_k(G)$, since a rerouting step changes at most one color.) Say γ and δ differ in v_i, where $\gamma(v_i) = c_1$ and $\delta(v_i) = c_2$.

First we observe that the path P contains the l and r vertices of the $v_i, \{c_1, c_2\}$-strand: l is the only vertex in layer L_{5i-3} that is adjacent to both a vertex of color c_1 in L_{5i-2} and a vertex of color c_2 in L_{5i-2}. Similarly, r is the only such vertex in layer L_{5i-1}.

Therefore, all vertices of P except the one in layer L_{5i-2} are part of the $v_i, \{c_1, c_2\}$-strand. Indeed, the only neighbor of l in layer L_{5i-4} is part of this strand (this is s_i^*), and the only neighbor of s_i^* in layer L_{5i-5} is part of this strand (this is t_{i-1}^*), all neighbors of t_{i-1}^* in layer L_{5i-6} are part of this strand, etc. Similarly, starting from r we can argue that the vertices of P in layers $5i$, $5i + 1$, etc. are part of this strand.

Since we now have that all internal vertices of P lie in the $v_i, \{c_1, c_2\}$-strand, we conclude that for all for neighbors v_j of v_i, $\gamma(v_j) \in \{1, 2, 3, 4\}\backslash\{c_1, c_2\}$: This follows from the construction of the $v_i, \{c_1, c_2\}$-strand (recall that for neighbors v_j of v_i, H_j^* contains no vertices of color c_1 or c_2).

So if γ is modified by changing the color of v_i from $\gamma(v_i) = c_1$ to $\delta(v_i) = c_2$, then again a coloring of G is obtained, which is δ. We conclude that all paths in the rerouting sequence correspond to colorings, since we started with one that corresponded to a coloring, namely P_α. □

Theorem 3. *SPR is PSPACE-complete.*

Proof: 4-Color Path is PSPACE-complete [4]. Our transformation from G to G' is polynomial; G' has $6n + 1$ strands and $O(n^2)$ vertices and edges. By Lemmas 1 and 2, G, α, β is a YES-instance for 4-Color Path if and only if G', P_α, P_β is a YES-instance for SPR. This proves PSPACE-hardness. Membership in PSPACE follows from Savitch's Theorem [20] which states that PSPACE = NPSPACE; the problem is easily seen to be in NPSPACE. □

4. Chordal Graphs

Recall that with a cycle C, we associate a vertex set $V(C)$ and edge set $E(C)$. A *chord* of C is an edge uv with $u, v \in V(C)$ but $uv \notin E(C)$. A graph G is *chordal* if every cycle of length at least 4 has a chord. We will show in this section that for chordal graphs G, the SPR problem can be decided in polynomial time. In fact, we prove a much stronger statement: if G is chordal, then $\mathrm{SP}(G, s, t)$ is connected and has diameter at most $d - 1$, where d is the distance from s to t. This requires the following property of edge pairs that lie on shortest st-paths in a chordal graph G.

Proposition 4. *Let v_iv_{i+1} and $v'_iv'_{i+1}$ be two edges of a chordal graph G, that both lie on a shortest st-path, with $\{v_i, v'_i\} \subseteq L_i$ and $\{v_{i+1}, v'_{i+1}\} \subseteq L_{i+1}$ for some i. Then $v_iv'_{i+1} \in E(G)$ or $v'_iv_{i+1} \in E(G)$.*

Proof: If $v_i = v'_i$ or $v_{i+1} = v'_{i+1}$ then the statement follows immediately, so now assume that $v_i \neq v'_i$ and $v_{i+1} \neq v'_{i+1}$, and thus $1 \leq i \leq d - 2$, where d is the distance from s to t.

Let P_L be a shortest path from v_i to v'_i in $G[L_0 \cup \ldots \cup L_{i-1} \cup \{v_i, v'_i\}]$. (Combining the initial parts of the shortest st-paths on which v_i and v'_i lie gives a walk from v_i to v'_i in this subgraph, so such a shortest path P_L indeed exists.) Analogously, we may define P_R to be a shortest path from v_{i+1} to v'_{i+1} in $G[L_{i+2} \cup \ldots \cup L_d \cup \{v_{i+1}, v'_{i+1}\}]$. The paths P_L and P_R are vertex disjoint, so combining these paths with the edges $v_i v_{i+1}$ and $v'_i v'_{i+1}$ gives a cycle C in G. This cycle has length at least four, and therefore contains a chord xy. Since P_L and P_R are both shortest paths, w.l.o.g. we may assume that $x \in V(P_L)$ and $y \in V(P_R)$. Since all edges of G are between vertices in the same layer or in consecutive layers, it follows that $x \in \{v_i, v'_i\}$ and $y \in \{v_{i+1}, v'_{i+1}\}$. The statement now follows because $xy \notin E(C)$. □

Theorem 5. *Let G be a chordal graph, and let P and Q be two shortest st-paths in G, of length d. Then a rerouting sequence from P to Q exists, of length at most $|V(P)\backslash V(Q)| \le d - 1$.*

Proof: We prove the statement by induction over $c = |V(P)\backslash V(Q)|$. If $c = 0$ then $P = Q$, so the statement is trivial. So now assume that $c \ge 1$; P and Q differ in at least one vertex. Let $P = u_0, u_1, \ldots, u_d$, and $Q = v_0, v_1, \ldots, v_d$. Let i be the lowest index such that $u_i \ne v_i$ (such an i exists since $c \ge 1$). Consider the edges $u_i u_{i+1}$ and $v_i v_{i+1}$. By Proposition 4, $u_i v_{i+1}$ or $v_i u_{i+1}$ is an edge of G as well. If $v_i u_{i+1} \in E(G)$, then we can apply the rerouting step $u_i \to v_i$ to P, since then v_i is also adjacent to both $u_{i-1} = v_{i-1}$ and u_{i+1}. This gives a new shortest st-path P' that has one more vertex in common with Q. So by induction, the distance from P' to Q in $\mathrm{SP}(G, s, t)$ is at most $c - 1$. Hence the distance from P to Q is at most c. Similarly, if $u_i v_{i+1} \in E(G)$, then applying the rerouting step $v_i \to u_i$ to Q gives a shortest st-path Q' that has one more vertex in common with P, and the claim follows analogously. □

The above proof gives a polynomial time algorithm for constructing the rerouting sequence. Obviously a rerouting sequence from P to Q requires at least $|V(P)\backslash V(Q)|$ rerouting steps, so we may conclude:

Corollary 6. *Let G be a chordal graph with shortest st-paths P and Q. In polynomial time, a shortest rerouting sequence from P to Q can be constructed.*

5. Claw-free Graphs

In this section we show that deciding SPR, and deciding whether $\mathrm{SP}(G, s, t)$ is connected can both be done in polynomial time in the case where G is claw-free. A *claw* is a $K_{1,3}$ graph. A graph G is *claw-free* if it contains no claw as induced subgraph. In other words, G is not claw-free if and only if it contains a subgraph H that consists of one vertex c of degree 3, and three leaves l_1, l_2, l_3, such that the leaves are pairwise nonadjacent in G. Such an *induced* subgraph will be called a *c-claw with leaves l_1, l_2, l_3* for short.

Consider a graph G, and layers L_i defined with respect to $s, t \in V(G)$ as before. Let $u \in L_i$. We say that *u has maximal in-neighborhood* if there is no $v \in L_i$ with $N^-(u) \subset N^-(v)$. (Note that we distinguish between subset $\subseteq$ and strict subset $\subset$.) In that case, the vertex set $N^-(u)$ is called a *maximal in-neighborhood in* L_{i-1}. These notions are defined analogously for out-neighborhoods.

With a layer L_i, we associate the following hypergraph $\mathcal{H}_i$: $\mathcal{H}_i$ has vertex set L_i, and the hyperedges correspond to the maximal in-neighborhoods of L_i. So for every $e \in E(\mathcal{H}_i)$, there exists a vertex $a \in L_{i+1}$ with $N^-(a) = e$.

The main result of this section is proved as follows. We first give some simple reduction rules. These are based on the fact that it is safe to delete a vertex v, if we know that it is not part of any shortest st-path that can be reached from the given shortest st-path P. We give two ways to identify such vertices. For reduced, claw-free SPR instances G', P, Q that do not have such vertices, we actually show that $\mathrm{SP}(G', s, t)$ is connected.

This is done by first rerouting P to a shortest st-path P' in which every vertex has maximal in-neighborhood, and rerouting Q to a shortest st-path Q' in which every vertex has maximal out-neighborhood. We show that this is possible in reduced claw-free graphs. Clearly it then suffices to decide whether Q' is reachable from P'. It is easy to see that these maximal neighborhood properties are useful if we wish to reroute P' to Q' layer by layer, in increasing order of layers (that is, intermediate paths will start with a subpath of Q', and end with a subpath of P'). Indeed, this is our main strategy. However rerouting in one layer is already nontrivial, and may require multiple rerouting steps. To find a short rerouting sequence for one layer, we use a shortest path in the hypergraph $\mathcal{H}_i$.

We first prove the properties underlying the reduction rules. Note that the next proposition does not require G to be claw-free.

Proposition 7. *Let P be a shortest st-path in a graph G. For every shortest st-path Q that is reachable from P in $SP(G, s, t)$ and every i, the L_i-vertex of Q is part of the same component of $\mathcal{H}_i$ as the L_i-vertex of P.*

Proof: Whenever a rerouting step $x \to y$ in layer L_i is made, there is a vertex $z \in L_{i+1}$ with $x, y \in N^-(z)$, so x and y are in the same component of $\mathcal{H}_i$. □

Proposition 8. *Let P be a shortest st-path of length d in a claw-free graph G. For every shortest st-path Q that is reachable from P in $SP(G, s, t)$ and every $i \in \{2, \ldots, d-2\}$, the L_i-vertex of Q is adjacent to the L_i-vertex of P.*

Proof: Consider a rerouting sequence $Q_0, \ldots, Q_k$ from $Q_0 = P$ to $Q_k = Q$, and let x_j be the L_i-vertex of Q_j, for every $j \in \{0, \ldots, k\}$. Assume that the claim is not true, so then we may choose ℓ to be the lowest index such that $x_0 x_\ell \notin E(G)$.

If x_0 and x_ℓ have a common neighbor z in either L_{i-1} or L_{i+1}, then a z-claw with leaves x_0, x_ℓ and y exists, for some vertex $y \in L_{i-2}$ or $y \in L_{i+2}$, respectively. (Here we use the fact that by definition of L_{i-1} or L_{i+1}, z lies on a shortest st-path, so it has at least one neighbor in L_{i-2} or L_{i+2}, respectively. Clearly this neighbor y is not adjacent to the vertices x_0 and x_ℓ in layer L_i. By definition, $x_0 x_\ell \notin E(G)$.) So since G is claw-free, we may conclude that $N^-(x_0) \cap N^-(x_\ell) = \emptyset$, and $N^+(x_0) \cap N^+(x_\ell) = \emptyset$.

If $x_{\ell-1}$ has a neighbor $y \in L_{i-1} \backslash N^-(x_0)$ and a neighbor $z \in L_{i+1} \backslash N^+(x_0)$, then an $x_{\ell-1}$-claw with leaves x_0, y, z exists. So w.l.o.g. we may assume that $N^-(x_{\ell-1}) \subseteq N^-(x_0)$. But then $N^-(x_{\ell-1}) \cap N^-(x_\ell) = \emptyset$, which contradicts that a rerouting step $x_{\ell-1} \to x_\ell$ is possible. □

Now we can define the notion of a reduced claw-free instance, and prove that such an instance can always be found in polynomial time.

Definition 9. *Let G be a claw-free graph with vertices s and t at distance d from each other. Then G is called st-*reduced *if*

- *all vertices lie on a shortest st-path,*
- *for every $i \in \{1, \dots, d-1\}$, $\mathcal{H}_i$ is connected, and*
- *for every $i \in \{2, \dots, d-2\}$, L_i is a clique.*

Lemma 10. *Let G be a claw-free graph, with shortest st-path P. In polynomial time, we can construct an induced (claw-free) subgraph G' of G such that*

- *G' is st-reduced, and*
- *a shortest st-path Q of G is reachable from P in SP(G, s, t) if and only if $V(Q) \subseteq V(G')$ and Q is reachable from P in SP(G', s, t).*

Proof: Let d denote the length of P. If we know that a given vertex v is not part of any shortest st-path that can be reached from P in SP(G, s, t), then it is easily seen that deleting v is *safe*, that is, the resulting graph $G' = G - v$ satisfies the second property from the lemma statement.

To obtain G' from G we apply the following three reduction rules, which all delete vertices. First, we delete every vertex that does not lie on a shortest st-path, which clearly is safe. Secondly, for every $i \in \{1, \dots, d-1\}$, we delete the vertices that do not lie in the same component of $\mathcal{H}_i$ as the L_i-vertex of P. By Proposition 7, this is safe. Finally, for every $i \in \{2, \dots, d-2\}$, we delete every vertex in L_i that is not adjacent to the L_i-vertex of P. By Proposition 8, this is safe. We apply these three reduction rules iteratively, until no rule can be applied anymore. Every reduction rule application deletes at least one vertex, so this process terminates in polynomial time. Call the resulting graph G'. Clearly, G' is an st-reduced graph. Since we only deleted vertices, G' is an induced subgraph of G, and therefore again claw-free. □

The last property from Definition 9 shows that every pair of vertices in one layer is adjacent; this makes it much easier in our proofs to obtain a contradiction by exhibiting an induced claw. We use this to prove the following three statements. First, we show that indeed maximal in-neighborhoods (or out-neighborhoods) can be guaranteed for every vertex.

Lemma 11. *Let P be a shortest st-path of length d in a claw-free st-reduced graph G. In polynomial time, a rerouting sequence of length at most $d-1$ can be* *constructed, from P to a shortest st-path P' in which every vertex has maximal out-neighborhood. Similarly, a rerouting sequence of length at most $d-1$ can be constructed, from P to a shortest st-path P'' in which every vertex has maximal in-neighborhood.*

Proof: Let $P = u_0, u_1, \dots, u_{d-1}, u_d$. Define $v_0 := u_0 (= s)$. For $i = 1, \dots, d-1$, in increasing order, we change the L_i-vertex u_i of P as follows. If the out-neighborhood of u_i is not maximal, then choose $v_i \in L_i$ with $N^+(u_i) \subset N^+(v_i)$, and $N^+(v_i)$ maximal. If possible, choose v_i such that $v_i \in N^+(v_{i-1})$. Then, apply the rerouting step $u_i \to v_i$. If u_i already has maximal out-neighborhood then simply define $v_i = u_i$.

It remains to show that $u_i \to v_i$ is in fact a rerouting step. By definition, $u_{i+1} \in N^+(v_i)$, so the L_{i+1}-vertex of the current path $v_0, \dots, v_{i-1}, u_i, u_{i+1}, \dots, u_d$ poses no problem. It might however be that v_i is not adjacent to v_{i-1}, the L_{i-1}-vertex of the current path. In that case, $i \geq 2$. Choose a vertex $x \in N^-(v_i)$. Since $v_i \in N^+(x) \backslash N^+(v_{i-1})$, but $N^+(v_{i-1})$ is maximal, there exists at least one $y \in N^+(v_{i-1}) \backslash N^+(x)$. By choice of v_i, there exists at least one $z \in N^+(v_i) \backslash N^+(y)$, otherwise y has maximal out-neighborhood as well, and we would have chosen $v_i = y$ (since we gave preference to out-neighbors of v_{i-1}). This however gives a v_i-claw with leaves x, y, z, a contradiction. (Since G is st-reduced, $v_i y \in E(G)$.)

If we wish to obtain a path with maximal in-neighborhoods for every vertex, we can follow an analog method, starting with layer L_{d-1} instead. □

The next two propositions are required to prove Lemma 14.

Proposition 12. *Let G be a claw-free, st-reduced graph, with distance d from s to t. For $i \in \{1, \dots, d-1\}$, let $x_0, \dots, x_\ell$ be a shortest path in $\mathcal{H}_i$. Then for all $j \in \{1, \dots, \ell-1\}$ and $k \in \{0, \dots, \ell\}$, it holds that $N^-(x_j) \subseteq N^-(x_k)$.*

Proof: Suppose to the contrary that there exists a vertex $y \in N^-(x_j) \backslash N^-(x_k)$, for some $j \in \{1, \dots, \ell-1\}$ and $k \in \{0, \dots, \ell\}$. W.l.o.g. we may assume that $k > j$. Let $a_0, \dots, a_{k-1}$ be vertices in L_{i+1} such that for all $p \in \{0, \dots, k-1\}$, $\{x_p, x_{p+1}\} \subseteq N^-(a_p)$. By definition of $\mathcal{H}_i$, such vertices exist.

We now claim that there exists an x_j-claw, with leaves y, a_{j-1}, x_k. These three vertices are all adjacent to x_j (x_k is adjacent since G is st-reduced so L_i is a clique). Clearly, $y \in L_{i-1}$ and $a_{j-1} \in L_{i+1}$ are not adjacent. By choice of y, it is not adjacent to x_k. If $x_k \in N^-(a_{j-1})$, then a shorter path from x_0 to x_k in $\mathcal{H}_i$ would exist, namely $x_0, \dots, x_{j-1}, x_k, \dots, x_\ell$, a contradiction. Hence this is indeed an induced claw, and thus from this contradiction we may conclude that for all j, k as stated, $N^-(x_j) \subseteq N^-(x_k)$. □

Proposition 13. *Let G be a claw-free and st-reduced graph. If $u, v \in L_i$ have distinct maximal in-neighborhoods, then $N^+(u) = N^+(v)$. If $u, v \in L_i$ have distinct maximal out-neighborhoods, then $N^-(u) = N^-(v)$.*

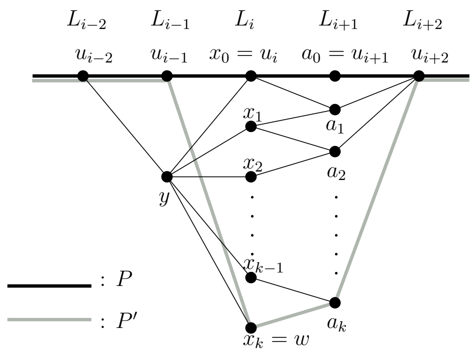

Figure 4: An illustration of the proof of Lemma 14. Vertical edges are omitted.

Proof: Suppose $u, v \in L_i$ have distinct maximal in-neighborhoods, and that there exists $x \in N^+(u) \backslash N^+(v)$. Then we may choose $y \in N^-(u) \backslash N^-(v)$. This gives a u-claw with leaves x, y, v. Note that $uv \in E(G)$ since G is st-reduced. The other cases are analog. □

Now we have all the tools for proving the following lemma, which is concerned with rerouting a single layer L_i, and requires maximal neighborhoods. Note that the L_{i+1}-vertex of the resulting path is not determined by the 'input' P and w, but this will not be a problem later.

Lemma 14. *Let G be a claw-free, st-reduced graph, and let $P = u_0, \ldots, u_d$ be a shortest st-path. Let $i \in \{1, \ldots, d-1\}$ such that u_{i-1} has maximal out-neighborhood and u_{i+1} has maximal in-neighborhood. Then for every $w \in N^+(u_{i-1})$, using at most $2|L_i|$ rerouting steps, P can be modified to a shortest st-path $P' = v_0, \ldots, v_d$ with $v_i = w$, and $v_j = u_j$ for all $j \in \{0, \ldots, d\} \backslash \{i, i+1\}$. Such a rerouting sequence can be found in polynomial time.*

Proof: If $i = d-1$, the proof is trivial: $u_{i+1} = t$, which is adjacent to every vertex in L_{d-1}. Therefore, we may simply apply the single rerouting step $u_i \to v$, to obtain the desired shortest st-path P'. So now assume that $1 \le i \le d-2$. The following notations and proof are illustrated in Figure 4.

Consider a shortest path $x_0, \ldots, x_k$ in $\mathcal{H}_i$ from u_i to w, so $x_0 = u_i$ and $x_k = w$. For $j \in \{1, \ldots, k\}$, let $a_j \in L_{i+1}$ be a vertex with maximal in-neighborhood such that $\{x_{j-1}, x_j\} \in N^-(a_j)$. (Such a vertex exists by the definition of the hypergraph $\mathcal{H}_i$.) Choose $a_1 = u_{i+1}$ if u_{i+1} satisfies this condition. In addition, let $a_0 = u_{i+1}$.

The plan is to use this path to reroute P to P'. In particular, the rerouting sequence that we construct below uses the rerouting steps $x_0 \to x_1 \to \ldots \to x_k$ in layer L_i. But it may be necessary to make changes in layers L_{i-1} and L_{i+1} as well.

Firstly, a rerouting step in L_{i-1} is required if there exists a j with $u_{i-1}x_j \notin E(G)$. In this case, let $y \in L_{i-1}$ be an in-neighbor of x_j with maximal out-neighborhood. (Note that x_j has at least one in-neighbor y that has maximal out-neighborhood.) We claim that $u_{i-2}y \in E(G)$: this follows since u_{i-1} has maximal out-neighborhood as well, so $N^-(u_{i-1}) = N^-(y)$ (Proposition 13). Secondly, Proposition 12 shows that for every ℓ, $yx_\ell \in E(G)$. On the other hand, if u_{i-1} is actually adjacent to every x_j, in particular if $i = 1$, then for the remainder of the proof we simply choose $y = u_{i-1}$. The rerouting sequence from P to P' is given by the following series of modifications (below we prove that these are all actual rerouting steps):

(1)	in L_{i-1}:	$u_{i-1} \to y$	(Skip if $y = u_{i-1}$.)
(2)	in L_{i+1}:	$a_0 \to a_1$	(Recall that $a_0 = u_{i+1}$. Skip if $a_0 = a_1$.)
(3)	in L_i:	$x_0 \to x_1$	(Recall that $x_0 = u_i$.)
(4)	in L_{i+1}:	$a_1 \to a_2$	
(5)	in L_i:	$x_1 \to x_2$	
	$\vdots$		
$(2k)$	in L_{i+1}:	$a_{k-1} \to a_k$	
$(2k+1)$	in L_i:	$x_{k-1} \to x_k$	(Recall that $x_k = w$.)
$(2k+2)$	in L_{i-1}:	$y \to u_{i-1}$	(Skip if $y = u_{i-1}$.)

Let $Q_0, \ldots, Q_m$ be the vertex sequences that result from these changes, starting with with $Q_0 = P$. We first verify that for every $\ell \in \{0, \ldots, m\}$, Q_ℓ is a shortest st-path. In other words, we show that for every ℓ, the ℓth change $p \to q$ above is a rerouting step; we verify that the L_{i-1}-vertex and L_{i+1}-vertex of $Q_{\ell-1}$ are both also adjacent to q.

As observed above, if $i \geq 2$, then y is adjacent to u_{i-2}, so in every Q_ℓ the L_{i-2}-vertex and L_{i-1}-vertex are adjacent. Furthermore, y is adjacent to every x_j, so it is adjacent to the L_i-vertex of every Q_ℓ. This shows that for every Q_ℓ, the L_{i-1}-vertex and the L_i-vertex are adjacent.

Now we show that L_i-vertex and L_{i+1}-vertex are adjacent in every Q_ℓ. If a rerouting step $x_j \to x_{j+1}$ is made in layer L_i, then at that point, the vertex in layer L_{i+1} is a_{j+1}, which by definition is adjacent to both x_j and x_{j+1}. Similarly, if a rerouting step $a_j \to a_{j+1}$ is made in layer L_{i+1}, then at that point the L_i-vertex is x_j, which is adjacent to both.

Finally, we show that the L_{i+1}-vertex and L_{i+2}-vertex are adjacent in every Q_ℓ. We first argue that whenever a rerouting step $a_j \to a_{j+1}$ is applied, a_j and a_{j+1} have distinct maximal in-neighborhoods. For $j \geq 1$, this follows from the fact that $x_0, \ldots, x_k$ is a *shortest* path in $\mathcal{H}_i$, so there is no in-neighborhood that contains both x_{j-1} and x_{j+1}. For a_0 and a_1 it follows from the choice of a_1: recall that $a_0 = u_{i+1}$, which we assumed to have a maximal in-neighborhood. So if $x_1 \notin N^-(a_0)$, then a_0 and a_1 again have distinct maximal in-neighborhoods. If $x_1 \in N^-(a_0)$, then we have chosen $a_1 = a_0$, and in fact no rerouting step is made. Hence we may now conclude that Proposition 13 can be applied for every rerouting step $a_j \to a_{j+1}$, which shows that $N^+(a_j) = N^+(a_{j+1})$ for every j. Therefore, u_{i+2} is an out-neighbor of every a_j.

This concludes the proof that every Q_ℓ is a shortest st-path, so $Q_0, \ldots, Q_m$ is a rerouting sequence, which results in the path $Q_m = u_0, \ldots, u_{i-1}, w, a_k, u_{i+2}, \ldots, u_d$, which is of the form we required for P'. Observe that the above rerouting sequence can be found in polynomial time (by finding a shortest path in $\mathcal{H}_i$, etc). Finally, note that this rerouting sequence used at most $2k+2$ rerouting steps. Since $k \leq |L_i|-1$, this proves the statement. □

Combining Lemmas 11 and 14 gives the main combinatorial result, for *st*-reduced claw-free graphs.

Theorem 15. *Let G be an st-reduced claw-free graph on n vertices, with distance d from s to t. Between any two shortest st-paths P and Q in G, a rerouting sequence of length at most $2n+2d-6$ exists, which can be constructed in polynomial time.*

Proof: First apply at most $d-1$ rerouting steps to P to obtain a shortest *st*-path P' in which every vertex has a maximal in-neighborhood (Lemma 11). Similarly, apply at most $d-1$ rerouting steps to Q to obtain a shortest *st*-path Q' in which every vertex has a maximal out-neighborhood (Lemma 11).

Now P' can be modified to Q' in $d-1$ stages i, with $i \in \{1,\dots,d-1\}$. Denote $P_0 = P' = u_0,\dots,u_d$, and $Q' = v_0,\dots,v_d$. At the start of the ith stage, we have a shortest *st*-path $P_{i-1} = v_0,\dots,v_{i-1},a,u_{i+1},\dots,u_d$ for some $a \in L_i$ (note that for $i=1$, P_0 is of this form). Using at most $2|L_i|$ rerouting steps, P_{i-1} can be modified into a shortest *st*-path $P_i = v_0,\dots,v_i,a',u_{i+2},\dots,u_d$ for some $a' \in L_{i+1}$. This follows from Lemma 14; note that in particular the conditions that the L_{i-1}-vertex of P_{i-1} has maximal out-neighborhood and that the L_{i+1}-vertex of P_{i+1} has maximal in-neighborhood are satisfied, since these are vertices from Q' and P', respectively.

After $d-1$ stages, this procedure terminates with a path $v_0,\dots,v_{d-1},u_d$, which equals Q'. The total number of rerouting steps for these stages is at most $\sum_{i\in\{1,\dots,d-1\}} 2|L_i| = 2(n-2)$. In total, this shows that P and Q can both be rerouted to a common shortest *st*-path Q', in at most $2(n-2)+(d-1)$ and $d-1$ steps, respectively. Combining these rerouting sequences gives a rerouting sequence from P to Q of length at most $2n+2d-6$. Since the rerouting sequences from Lemmas 11 and 14 can both be found in polynomial time, the entire rerouting sequence can be found in polynomial time. □

Now we can easily deduce our main two algorithmic results.

Theorem 16. *Let G be a claw-free graph on n vertices, and let P and Q be two shortest st-paths in G, of length d. In polynomial time it can be decided whether Q is reachable from P in $SP(G,s,t)$, and if so, a rerouting sequence of length at most $2n+2d-6$ exists.*

Proof: By Lemma 10, in polynomial time we can construct an *st*-reduced induced subgraph G' of G such that any shortest *st*-path Q' is reachable from P in G if and only if it is reachable from P in G'. So if Q is not a shortest *st*-path of G' (at least one of its vertices was deleted), we may conclude it is not reachable. Otherwise, Theorem 15 shows that Q is reachable from P, with a rerouting sequence of length at most $2|V(G')|+2d-6 \leq 2n+2d-6$. □

Theorem 17. *Let G be a claw-free graph on n vertices. In polynomial time it can be decided whether $SP(G, s, t)$ is connected.*

Proof: In polynomial time we can first delete all vertices of G that do not lie on a shortest st-path, to obtain G'. Clearly, $\mathrm{SP}(G, s, t) = \mathrm{SP}(G', s, t)$, and G' is again claw-free. Choose an arbitrary shortest st-path P. Using G' and P, Lemma 10 can be applied to obtain an st-reduced subgraph G'' of G' in polynomial time. If $G'' = G'$ then Theorem 15 shows that $\mathrm{SP}(G'', s, t) = \mathrm{SP}(G', s, t)$ is connected. Otherwise, there exists at least one vertex $v \in V(G') \backslash V(G'')$, and we may conclude that $\mathrm{SP}(G', s, t)$ is not connected: G' has a shortest st-path Q' with $v \in V(Q')$, which is not part of G'', but all shortest st-paths that are reachable from P are part of G'' (Lemma 10). □

6. Isolated Paths

In this section we give a polynomial time algorithm for counting the number of isolated paths. Recall that an *isolated st-path* is a shortest st-path in G that has no neighbor in $\mathrm{SP}(G, s, t)$. For this, we need to consider isolated sy-paths for vertices $y \neq t$. For three vertices s, x, y with $s \neq y$, we use $\mathrm{iso}_{sy}(x)$ to denote the number of isolated sy-paths that contain the vertex x. We will use this notation for the case where x is the second-to-last vertex on a shortest sy-path, so it is adjacent to y.

Proposition 18. *Let y and z be vertices at distance i and $i{+}1$ of s, respectively, with $i \geq 1$. Then $iso_{sz}(y) = \sum_x iso_{sy}(x)$, where the summation is over all vertices x at distance $i-1$ from s such that $N(x) \cap N(z) = \{y\}$.*

Proof: Let x be such a vertex. There are $\mathrm{iso}_{sy}(x)$ isolated sy-paths that end with x and y. Since y is the only common neighbor of x and z, extending every one of these paths with the vertex z gives a set of isolated sz-paths that contain y, which are all distinct. All of these paths have x as L_{i-1}-vertex, so when choosing a different L_{i-1}-vertex in the role of x, a different set of sz-paths is obtained. This shows that there are at least $\sum_x \mathrm{iso}_{sz}(y)$ distinct isolated sz-paths that contain y, where the summation is over all vertices x at distance $i-1$ from s such that $N(x) \cap N(z) = \{y\}$.

We conclude the proof by observing that every isolated sz-path that contains y also contains some vertex x at distance $i-1$ from s with $N(x) \cap N(z) = \{y\}$, such that removing z yields an isolated sy-path. Indeed, obviously $y \in N(x) \cap N(z)$, and if $|N(x) \cap N(z)| \geq 2$, then the path is not isolated. Therefore, all isolated sz-paths that contain y are counted this way, and thus we have equality. □

Theorem 19. *Let G be a graph with $s, t \in V(G)$. In polynomial time, the number of isolated st-paths can be computed.*

Proof: Let d be the distance from s to t. As usual, the layers L_i for $i \in \{0, \dots, d\}$ are defined with respect to s and t. The algorithm works by computing the values $\text{iso}_{sz}(y)$ for various choices of y and z, in increasing distance from s to z. First, for every $z \in L_1$, initialize $\text{iso}_{sz}(s) = 1$ (which is trivially correct). Then, for $i = 2, \dots, d$, in increasing order of i, do the following. For every $z \in L_i$ and every $y \in L_{i-1}$, compute $\text{iso}_{sz}(y)$ using Proposition 18. Note that the required values $\text{iso}_{sy}(x)$ have all been computed earlier. In the end, return $\sum_{y \in L_{d-1}} \text{iso}_{st}(y)$, which is the total number of isolated st-paths.

Now we analyze the complexity. Let $n = |V(G)|$. Note that the number of combinations of y and z for which $\text{iso}_{sz}(y)$ is computed is less than n^2. For every such combination, evaluating the expression from Proposition 18 can be done in polynomial time. □

The following example shows that this counting result is nontrivial: if we choose G'' to be just the main strand of the instance G' constructed in Section 3.1 (based on a graph G on n vertices), then G'' contains 4^n isolated paths, which is exponential in the number of vertices of G'' (which is $14n + 2$).

7. Discussion

In this paper we showed that SPR is PSPACE-complete, which is somewhat surprising since the problem of finding shortest paths is easy. Nevertheless, the results in this paper otherwise confirm the typical behavior of reconfiguration problems: for instances where we can decide SPR in polynomial time (chordal and claw-free graphs), the diameter is polynomially bounded – in this case, even linearly bounded. In addition, for these graph classes it can be decided efficiently whether the solution graph $\text{SP}(G, s, t)$ is connected. The main question that is left open here is: *What is the complexity of deciding whether SP(G, s, t) is connected, for general graphs G?* Note that for the SPR instances G', P_α, P_β constructed in Section 3, $\text{SP}(G', s, t)$ is always disconnected (unless the given 4-Color Path instance consists of an edgeless graph G): there exist shortest st-paths that correspond to colorings, and shortest st-paths that do not. The proof of Lemma 2 shows that these lie in different components of $\text{SP}(G', s, t)$.

We showed that for chordal graphs G, we could even find *shortest* rerouting sequences in polynomial time. Is this possible for claw-free graphs as well? To be precise, for two shortest st-paths P and Q in a claw-free graph G and $k \in \mathbb{N}$, can it be decided in polynomial time whether a rerouting sequence from P to Q of length at most k exists, or is this problem NP-complete? Recall that for general graphs, the NP-hardness of finding a shortest rerouting sequence was proved in [16]. By our linear diameter result, this (decision) problem lies in NP for claw-free graphs. Finally, it is interesting to search for other graph classes for which SPR can be solved in polynomial time. Recently this was shown for planar graphs [3]. Graphs of bounded treewidth form another prime candidate.

Acknowledgements

The author would like to thank Paul Medvedev and Martin Milanič for the introduction to this problem, and for comments on an earlier version of this paper. This research was supported by DFG grant BO 3391/1-1.

[1] M. Bonamy, M. Johnson, I. Lignos, V. Patel, and D. Paulusma. Reconfiguration graphs for vertex colourings of chordal and chordal bipartite graphs. *Journal of Combinatorial Optimization*, pages 1–12, 2012. http://dx.doi.org/10.1007/s10878-012-9490-y.

[2] P. Bonsma. Shortest path reconfiguration is PSPACE-hard. ArXiv preprint arXiv:1009.3217v1, 2010.

[3] P. Bonsma. Rerouting shortest paths in planar graphs. In *FSTTCS 2012*, volume 18 of *LIPIcs*, pages 337–349. Schloss Dagstuhl - Leibniz-Zentrum fuer Informatik, 2012.

[4] P. Bonsma and L. Cereceda. Finding paths between graph colourings: PSPACE-completeness and superpolynomial distances. *Theoret. Comput. Sci.*, 410(50):5215–5226, 2009.

[5] L. Cereceda, J. van den Heuvel, and M. Johnson. Connectedness of the graph of vertex-colourings. *Discrete Appl. Math.*, 308(5–6):913–919, 2008.

[6] L. Cereceda, J. van den Heuvel, and M. Johnson. Mixing 3-colourings in bipartite graphs. *European J. Combin.*, 30(7):1593–1606, 2009.

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

[8] M. Chudnovsky and P. Seymour. The structure of claw-free graphs. In *Surveys in combinatorics 2005*, pages 153–171. Cambridge Univ. Press, 2005.

[9] R. Diestel. *Graph theory*, volume 173 of *Graduate Texts in Mathematics*. Springer, Berlin, fourth edition, 2010.

[10] C.E.J. Eggermont and G.J. Woeginger. Motion planning with pulley, rope, and baskets. In *STACS 2012*, volume 14 of *LIPIcs*, pages 374–383, 2012.

[11] P. Gopalan, P.G. Kolaitis, E. Maneva, and C.H. Papadimitriou. The connectivity of boolean satisfiability: Computational and structural dichotomies. *SIAM J. Comput.*, 38(6), 2009.

[12] R.A. Hearn and E.D. Demaine. PSPACE-completeness of sliding-block puzzles and other problems through the nondeterministic constraint logic model of computation. *Theoret. Comput. Sci.*, 343(1–2):72–96, 2005.

[13] 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. In *ISAAC 2008*, volume 5369 of *LNCS*, pages 28–39. Springer, 2008.

[14] 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. *Theoret. Comput. Sci.*, 412(12–14):1054–1065, 2011.

[15] T. Ito, M. Kamiński, and E.D. Demaine. Reconfiguration of list edge-colorings in a graph. In *WADS 2009*, volume 5664 of *LNCS*, pages 375–386. Springer, 2009.

[16] M. Kamiński, P. Medvedev, and M. Milanič. Shortest paths between shortest paths. *Theoret. Comput. Sci.*, 412(39):5205–5210, 2011.

[17] M. Kamiński, P. Medvedev, and M. Milanič. Shortest paths between shortest paths and independent sets. In *IWOCA 2010*, volume 6460 of *LNCS*, pages 56–67. Springer, 2011.

[18] M. Kamiński, P. Medvedev, and M. Milanič. Complexity of independent set reconfigurability problems. *Theoret. Comput. Sci.*, 439:9–15, 2012.

[19] C.H. Papadimitriou, A.A. Schäffer, and M. Yannakakis. On the complexity of local search. In *STOC 1990*, pages 438–445, New York, 1990. ACM.

[20] W.J. Savitch. Relationships between nondeterministic and deterministic tape complexities. *J. Comput. System. Sci.*, 4:177–192, 1970.