Locally Rainbow Paths

Till Fluschnik[1], **Leon Kellerhals**[2], **Malte Renken**[2]

[1]Institut für Informatik, TU Clausthal, Germany
[2]Technische Universität Berlin, Algorithmics and Computational Complexity, Germany
till.fluschnik@tu-clausthal.de, leon.kellerhals@tu-berlin.de, m.renken@tu-berlin.de

Abstract

We introduce the algorithmic problem of finding a *locally rainbow* path of length ℓ connecting two distinguished vertices s and t in a vertex-colored directed graph. Herein, a path is locally rainbow if between any two visits of equally colored vertices, the path traverses consecutively at leaset r differently colored vertices. This problem generalizes the well-known problem of finding a rainbow path. It finds natural applications whenever there are different types of resources that must be protected from overuse, such as crop sequence optimization or production process scheduling. We show that the problem is computationally intractable even if $r = 2$ or if one looks for a locally rainbow among the shortest paths. On the positive side, if one looks for a path that takes only a short detour (i.e., it is slightly longer than the shortest path) and if r is small, the problem can be solved efficiently. Indeed, the running time of the respective algorithm is near-optimal unless the ETH fails.

Introduction

Many graph connectivity problems are studied with additional constraints to make them applicable to real-world problems. Typical constraints include forbidden pairs of vertices or edges in the solution or — if the graph is colored — requiring that the solutions are rainbow (no two elements in the solution have the same color) or properly colored (no two adjacent elements have the same color). Examples for such constraints on problems can be found for spanning trees (Broersma and Li 1997; Darmann et al. 2011), Steiner trees (de Uña et al. 2016; Ferone, Festa, and Guerriero 2022; Halldórsson et al. 2018), but most notably for paths (Alon, Yuster, and Zwick 1995; Agrawal et al. 2020; Bhattacharya 2010; Bentert, Kellerhals, and Niedermeier 2023).

For paths, the properly edge-colored variant forbids two equally colored edges to appear subsequently in the path. What, to the best of our knowledge, has not been considered yet, is any model that forbids a visited color for the next, say r, subsequent vertices of the path. For example, this allows the modeling of protecting certain types of resources from overuse. This for example is relevant for crop sequence optimization: here, different colors model different types of crops which, depending on the season, have different impacts on soil health (Dury et al. 2012; Turchetta et al. 2022; Benini et al. 2023). Other applications include holiday trip planning (different colors modeling different types of leisure activities), production process scheduling (different colors modeling different workers or machines).

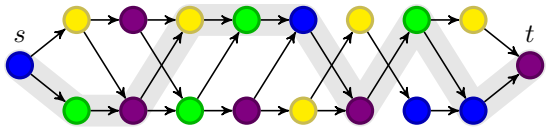

Figure 1: A digraph whose vertices are colored with four colors, with a shortest 2-rainbow (but not 3-rainbow) s-t path.

More concretely, given a vertex-colored graph, we propose the concept of *locally rainbow* paths, in which every subpath of bounded length is required to carry pairwise distinct colors. Formally, a path or walk $W = (v_0, v_1, \dots, v_q)$ in G is *r-rainbow* if for every $i \in [0, q - r]$, the vertices $v_i, v_{i+1}, \dots, v_{i+r}$ have pairwise different color (see Fig. 1). We arrive at the following problem description:

LOCALLY RAINBOW PATH
Input: A digraph G, a vertex-coloring $c\colon V(G) \to C$, two vertices $s, t \in V(G)$, two integers $r, \ell \in \mathbb{N}_0$.
Question: Is there an r-rainbow s-t path of length at most ℓ in G?

We also consider LOCALLY RAINBOW WALK, where we look for s-t walks with the same constraints. LOCALLY RAINBOW PATH becomes the aforementioned problem of finding a rainbow path when $r = \ell$. If $r = 1$, then the problem coincides with finding a properly colored s-t path.

Our contributions. We study the parameterized complexity of LOCALLY RAINBOW WALK and LOCALLY RAINBOW PATH, with a focus on the *locality* parameter r. We show that the path variant is NP-hard for any fixed value of $r \geq 2$ (Theorem 16). In contrast, we are able to design an algorithm with running time $2^{\mathcal{O}(r \log r)} \cdot n^{\mathcal{O}(1)}$ for the walk variant (Theorem 1), with n being the number of vertices. This result is achieved by developing an ordered version of the *representative families* technique. We prove this result to be optimal in the sense that no $2^{o(r \log r)} \cdot n^{\mathcal{O}(1)}$-time algorithm is possible if the ETH holds (Theorem 12).

Note that an r-rainbow s-t walk of length ℓ must always be a path when $\ell \leq \operatorname{dist}(s,t) + r$. Thus, our algorithm for LOCALLY RAINBOW WALK also applies to the path variant when the *detour length* $k := \ell - \operatorname{dist}(s,t)$ is small. Motivated by this observation and the result of Bezáková et al. (2019) that finding s-t paths of detour length k is fixed-parameter tractable for the parameter k, we also investigate this parameter. While both of our problem variants remain NP-hard even when $k = 0$ (Theorem 12), we are able to give a fixed-parameter tractable algorithm for the combined parameter $k + r$ (Theorem 19).

We mention in passing that our results also hold when coloring the edges instead of vertices (and adapting local rainbowness accordingly). Furthermore, our (nontrivial) algorithmic results also hold when looking for paths of length *exactly* ℓ. Proofs of results marked with ★ are deferred to the paper's full version.

Related work. Finding a rainbow path is known to be NP-hard (Chen, Li, and Shi 2011) and fixed-parameter tractable with respect to the number of colors (Kowalik and Lauri 2016; Uchizawa et al. 2013). While finding a properly colored path is trivially linear-time solvable, it is less obvious that this is also solvable in that time in an (undirected) *edge*-colored graph. This was shown by Szeider (2003).

The field of finding paths of detour length exactly or at least k is rather active, with the former being easier to tackle than the latter. Bezáková et al. (2019) prove both variants to be fixed-parameter tractable, however, for the latter variant only on undirected graphs. While there has been some progress on directed graphs, most recently by Jacob, Włodarczyk, and Zehavi (2023), it is open whether finding a path with detour length at least one is polynomial-time solvable.

Another closely related and more applied area is that of finding resource-constrained paths. Here, the graph carries arc (or vertex) weights and the desired s-t path must not accumulate more than a given threshold of that weight. The problem is known to be NP-hard (Handler and Zang 1980) and studied in many variations (Ford et al. 2022; Irnich and Desaulniers 2005; Pugliese and Guerriero 2013). A variation close to our setting introduces so-called *replenishment arcs*, at which one may "drop off" the weight accumulated so far (Smith, Boland, and Waterer 2012). This setting is relevant in airline/train crew scheduling (weight represents duty hours, replenishment arcs correspond to crew overnight rests) and aircraft/train routing (weight represents machine hours, replenishment arcs correspond to maintenance events) and also has ties with electric vehicle routing problems (weight represents battery discharge, replenishment arcs correspond to charging events) (Adler et al. 2016; Zündorf 2014). Our rainbowness constraint is similar in that it "replenishes" any colors that were visited more than r steps ago.

Preliminaries

We denote by $\mathbb{Z}$, $\mathbb{N}_0$, and $\mathbb{N}$ the set of all, the non-negative, and the positive integers, respectively. For $n, m \in \mathbb{Z}$ we denote by $[n,m] := \{i \in \mathbb{Z} \mid n \leq i \leq m\}$ the set of integers between n and m and define $[n] := [1,n]$. We denote by $e \approx 2.718$ Euler's number and by $\omega < 2.373$ the matrix multiplication constant (Alman and Williams 2021).

Let $\sigma := (a_1, \ldots, a_n)$ be a sequence. We denote by $|\sigma| := n$ its *length*, i.e., the number of elements in σ, and also call σ an *n-sequence*. We write $x \in \sigma$ if $x = a_i$ for some $i \in [n]$. If every element in σ is contained in a set U, then we say that σ is a *sequence on* (or *over*) U. A sequence σ' is a *substring* or *consecutive subsequence* of σ if there are $i < j \in [n]$ with $\sigma' = (a_i, a_{i+1}, \ldots, a_j)$. If $i = 1$ or $j = n$, then we also say that σ *begins with* or *ends on* σ', respectively. If $\rho = (b_1, \ldots, b_m)$ is a sequence, then we denote by $\sigma \circ \rho := (a_1, \ldots, a_n, b_1, \ldots, b_m)$ the *concatenation* of σ and ρ. For sequences $\sigma_1, \ldots, \sigma_n$, we denote by $\bigcirc_{i=1}^{n} \sigma_i = \sigma_1 \circ \cdots \circ \sigma_n$ their consecutive concatenation.

Graph theory. For basic notations on (directed) graph theory see, e.g., (Diestel 2016; Bang-Jensen and Gutin 2009). A digraph G is a tuple (V, A) with $A \subseteq V \times V$. In this work, all digraphs contain no self-loops, i.e., no arcs from the set $\{(v,v) \mid v \in V\}$. For a digraph $G = (V, A)$ we also denote by $A(G)$ the arc set A and by $V(G)$ the vertex set V. We call a digraph G symmetric if $(v,w) \in A(G) \iff (w,v) \in A(G)$. The symmetrization of the digraph G is the graph $(V, A(G) \cup \{(v,w) \mid (w,v) \in A(G)\})$. For two vertices $v, w \in V(G)$, a v-w walk $W = (u_0 = v, u_1, \ldots, u_q = w)$ (of length q) is a sequence of vertices from V such that $(u_{i-1}, u_i) \in A(G)$ for every $i \in [q]$. A v-w walk is a path if all vertices are pairwise different. A digraph G is weakly connected if in its symmetrization G^* it holds true that for any $(v,w) \in V \times V$ there is an v-w path. Throughout, unless stated otherwise, we denote by $n := |V(G)|$ and $m := |A(G)|$ and assume the input digraph G to be weakly connected (and hence $n \leq m - 1$). For a vertex v, we denote by $N^-(v) := \{w \in V(G) \mid (w,v) \in A(G)\}$.

Color sequences and compatibility. Let G be a digraph and let $c\colon V(G) \to C$ be a vertex coloring. Recall that we call a path or walk $W = (v_0, v_1, \ldots, v_q)$ in G *r-rainbow* if for every $i \in [0, q-r]$, the vertices $v_i, v_{i+1}, \ldots, v_{i+r}$ have pairwise different color. The *color sequence* of W is $\sigma := (c(v_0), \ldots, c(v_q))$. We sometimes also call σ r-rainbow if W is r-rainbow. For two r-rainbow sequences $\sigma = (a_1, \ldots, a_n)$ and $\rho = (b_1, \ldots b_m)$, we say that σ is *r-compatible* to ρ if a path or walk with color sequence $\sigma \circ \rho$ is r-rainbow. Formally, σ is r-compatible to ρ if $\{a_{\max(1,n-j+1)}, \ldots, a_n\} \cap \{b_1, \ldots, b_{\min(r-j+1,m)}\} = \emptyset$ for all $j \in [r]$.

Parameterized complexity. Let Σ be a finite alphabet and $\Sigma^* = \{x \in \Sigma^n \mid n \in \mathbb{N}_0\}$. A parameterized problem P is a subset $\{(x,k) \mid x \in \Sigma^*, k \in \mathbb{N}_0\} \subseteq \Sigma^* \times \mathbb{N}_0$, where k is referred to as the parameter. A parameterized problem P is fixed-parameter tractable (in FPT) if every instance (x,k) is solvable in $f(k) \cdot |x|^{O(1)}$ time, where f is some computable function only depending on k. The Exponential Time Hypothesis (ETH) (Impagliazzo and Paturi 2001; Impagliazzo, Paturi, and Zane 2001) states that there exists some fixed $\varepsilon > 0$ such that 3-SAT cannot be decided in $2^{\varepsilon \cdot n} \cdot (n+m)^{O(1)}$ time on any input with n variables and m clauses. For more details, see Cygan et al. (2015).

Walks

In this section we study the parameterized complexity of LOCALLY RAINBOW WALK with respect to the parameter r. Note that all results obtained here also hold for finding shortest r-rainbow paths, i.e., paths with length $\ell = \operatorname{dist}(s,t)$. We will see that the problem is fixed-parameter tractable, by providing an $r^{\mathcal{O}(r)} \cdot n^{\mathcal{O}(1)}$-time algorithm. Indeed, although the length of a walk is not bounded in the input size, we can show that the above running time holds even if we ask whether there exists an r-rainbow s-t walk of any length. Finally, we prove a asymptotically tight running time lower bound based on the Exponential Time Hypothesis (ETH).

Fixed-Parameter Tractability

In this section we show the following.

Theorem 1. LOCALLY RAINBOW WALK *can be solved in $\mathcal{O}((r \cdot e)^{\omega r} \cdot \ell m)$ time, where m is the number of arcs in the input graph and ω is the matrix multiplication constant.*

Note that this does not yet prove fixed-parameter tractability for LOCALLY RAINBOW WALK parameterized by r as the walk may become very long, i.e., ℓ may not be bounded polynomially in the input size or by any function in r. Later in this section we will show that we can always find a solution whose length is bounded by a function in r; thus proving fixed-parameter tractability with r.

Our algorithm will build a family $\mathcal{W}_v^p$ of r-rainbow length-p s-v walks for every length p and each vertex v using dynamic programming in a Dijkstra fashion — that is, it will extend the walks along the arcs of the graph. To ensure that the r-rainbowness is maintained in this process we only need to remember the sequence $\sigma = (c_1, \ldots, c_r)$ at the end of the color sequence of any walk W. So we want to compute for each $v \in V(G)$ and $p \in [\ell]$ the family

$$\mathcal{W}_v^p \coloneqq \left\{ \sigma \;\middle|\; \begin{array}{l} |\sigma| = \min\{p+1, r\} \text{ and } G \text{ contains} \\ \text{an } r\text{-rainbow length-}p\ s\text{-}v \text{ walk} \\ \text{whose color sequence ends on } \sigma \end{array} \right\}. \quad (1)$$

Note that trivial dynamic programming on these families would blow up their size to $\mathcal{O}(|C|^r)$, which is too large for our purposes. Yet, σ restricts the choice in colors for the next r vertices on the path: The path may only continue with a sequence ρ of colors to which σ is r-compatible. If however, for some sequence ρ there are multiple sequences in $\mathcal{W}_v^p$ that are r-compatible to ρ, then it suffices to remember only one of them. We call the remaining family an *ordered representative* for $\mathcal{W}_v^p$ and define it formally as follows.

Definition 2 (Ordered representative)**.** Let $p, r \in \mathbb{N}$ with $p \leq r$ and let $\mathcal{W}$ be a family of sequences of length at most p. A subfamily $\widehat{\mathcal{W}}$ of $\mathcal{W}$ is an ordered r-representative for $\mathcal{W}$ (written $\widehat{\mathcal{W}} \subseteq^r_{\text{orep}} \mathcal{W}$) if the following holds for every sequence ρ of length at most r: If there exists a $\sigma \in \mathcal{W}$ that is r-compatible to ρ, then there exists a $\widehat{\sigma} \in \widehat{\mathcal{W}}$ that is r-compatible to ρ.

To compute an ordered r-representative for $\mathcal{W}_v^p$ we make use of an algorithm by Fomin et al. (2016) to compute representatives of (unordered) set families. Let us first define (unordered) representatives for families of sets.

Definition 3 (Unordered representative)**.** Let $\mathcal{F}$ be a family of p-element sets and $q \in \mathbb{N}$. A subfamily $\widehat{\mathcal{F}}$ of $\mathcal{F}$ is a *q-representative* for $\mathcal{F}$ (written $\widehat{\mathcal{F}} \subseteq^q_{\text{rep}} \mathcal{F}$) if the following holds for every set Y of size at most q: If $\mathcal{F}$ contains a set X disjoint from Y, then $\widehat{\mathcal{F}}$ contains a set $\widehat{X}$ disjoint from Y.

While Fomin et al. (2016) state their results for families of independent sets of a matroid, for our purposes, the simpler definition for set families (a special case) suffices.

Proposition 4 ((Fomin et al. 2016))**.** *There is an algorithm that, given a family $\mathcal{F}$ of p-sets over a universe U and an integer $q \in \mathbb{N}_0$, computes in time $\mathcal{O}\big(|\mathcal{F}| \cdot \binom{p+q}{p} p^\omega + |\mathcal{F}| \cdot \binom{p+q}{q}^{\omega-1}\big)$ a q-representative $\widehat{\mathcal{F}}$ for $\mathcal{F}$ of size at most $\binom{p+q}{p}$.*

We will first show how to translate a sequence σ into a corresponding (unordered) set so that we can make use of the concept of representatives for unordered set families. After that, we are ready to devise an algorithm for Theorem 1.

Consider two r-rainbow walks W_σ and W_ρ with color sequences $\sigma = (a_1, \ldots, a_p)$ and $\rho = (b_1, \ldots, b_q)$. We wish to define two functions π and π' that map color sequences to subsets of $C \times [r]$ such that σ is r-compatible to ρ if and only if $\pi(\sigma) \cap \pi'(\rho) = \emptyset$. By definition, σ is r-compatible to ρ if and only if b_i does not equal any of the last $r - i + 1$ entries of σ. Define

$$\begin{aligned} \pi'(\rho) &\coloneqq \{(b_i, i) \mid i \in [\min\{r, q\}]\} \quad \text{and} \\ \pi(\sigma) &\coloneqq \{(a_j, i) \mid i \in [r],\ j \in [p - (r - i), p] \cap \mathbb{N}\}. \end{aligned} \quad (2)$$

Then, $(b_i, i) \notin \pi(\sigma)$ if and only if b_i does not appear among the last $r - i + 1$ entries of σ. In other words, we have the following.

Observation 5. *A sequence σ is r-compatible to a sequence ρ if and only if $\pi(\sigma) \cap \pi'(\rho) = \emptyset$.*

We now have the promised connection between ordered and unordered representatives.

Lemma 6. *Let $\mathcal{W}$ be a family of p-sequences and let $\mathcal{F} \coloneqq \{\pi(\sigma) \mid \sigma \in \mathcal{W}\}$. If $\widehat{\mathcal{F}}$ is an r-representative of $\mathcal{F}$, then $\widehat{\mathcal{W}} \coloneqq \{\sigma \mid \pi(\sigma) \in \widehat{\mathcal{F}}\}$ is an ordered r-representative of $\mathcal{W}$.*

Proof. Consider a sequence $\rho = (b_1, \ldots, b_r)$. Suppose that $\sigma = (a_1, \ldots, a_p) \in \mathcal{W}$ is r-compatible to ρ. Then by Observation 5, $\pi(\sigma)$ is disjoint from $\pi'(\rho)$. Therefore, there is a set $\pi(\widehat{\sigma}) \in \widehat{\mathcal{F}}$ which is disjoint from $\pi'(\rho)$; Thus by Observation 5, $\widehat{\sigma}$ is r-compatible to ρ. □

Consequently, we can use Proposition 4 to compute ordered r-representatives.

Corollary 7 (★). *There is an algorithm that, given a family $\mathcal{W}$ of p-sequences over a universe U and an integer $r \in \mathbb{N}_0$, computes in time $\mathcal{O}\big(|\mathcal{W}| \cdot (r \cdot e)^r r^\omega + |\mathcal{W}| \cdot (r \cdot e)^{(\omega-1)r}\big)$ an ordered r-representative $\widehat{\mathcal{W}}$ of $\mathcal{W}$ of size at most $(r \cdot e)^r$.*

Ordered representatives are transitive, just like their unordered counterparts (Fomin et al. 2016).

Observation 8 (★). *If $\widehat{\mathcal{W}} \subseteq^r_{\text{orep}} \widetilde{\mathcal{W}}$ and $\widetilde{\mathcal{W}} \subseteq^r_{\text{orep}} \mathcal{W}$, then $\widehat{\mathcal{W}} \subseteq^r_{\text{orep}} \mathcal{W}$.*

With a way to efficiently compute ordered r-representatives at hand, we can compute an r-rainbow s-t walk of length ℓ with the following routine. Recall that we are given a graph G with two terminals s and t, a coloring $c\colon V(G) \to C$, and two integers r and ℓ as input.

Algorithm 1. Set $\widehat{\mathcal{W}}_s^0 := \{(c(s))\}$ and for all $v \in V(G) \setminus \{s\}$, set $\widehat{\mathcal{W}}_v^0 := \emptyset$. Now, for each $p = 1, 2, \ldots, \ell$ compute for all $v \in V(G)$ the set $\mathcal{N}_v^p$. If $p < r$, then $\mathcal{N}_v^p :=$

$$\bigcup_{u \in N^-(v)} \left\{ (a_1, \ldots, a_{p+1}) \;\middle|\; \begin{array}{l} (a_1, \ldots, a_p) \in \widehat{\mathcal{W}}_u^{p-1} \text{ and} \\ c(v) = a_{p+1} \notin \{a_1, \ldots, a_p\} \end{array} \right\}$$

If $p \geq r$, then $\mathcal{N}_v^p :=$

$$\bigcup_{u \in N^-(v)} \left\{ (a_2, \ldots, a_{r+1}) \;\middle|\; \begin{array}{l} (a_1, \ldots, a_r) \in \widehat{\mathcal{W}}_u^{p-1} \text{ and} \\ c(v) = a_{r+1} \notin \{a_1, \ldots, a_r\} \end{array} \right\}.$$

Compute an ordered r-representative $\widehat{\mathcal{W}}_v^p \subseteq^r_{\text{orep}} \mathcal{N}_v^p$. Return `yes` if and only if $\widehat{\mathcal{W}}_t^q \neq \emptyset$ for some $q \in [\ell]$.

Let us show that Algorithm 1 indeed computes representatives of the family $\mathcal{W}_v^p$ as defined in equation (1), and hence, is correct.

Lemma 9. *For each $v \in V(G)$ and $p \in [0, \ell]$, the family $\widehat{\mathcal{W}}_v^p$ computed by Algorithm 1 contains at most $(r \cdot e)^r$ sets and is an ordered r-representative of $\mathcal{W}_v^p$ as defined in* (1)*.*

Proof. Our proof is by induction. By the initial assignments of $\widehat{\mathcal{W}}_v^0$, the statement holds for $p = 0$. Now, fix some $p \in [\ell]$ and assume that $\widehat{\mathcal{W}}_u^{p-1} \subseteq^r_{\text{orep}} \mathcal{W}_u^{p-1}$ for all $u \in V(G)$.

Let $\rho = (b_1, \ldots, b_q)$ be a color sequence with $q \leq r$ and let $v \in V(G)$. Suppose that there exists a sequence $\sigma \in \mathcal{W}_v^p$ that is r-compatible to ρ. We claim that there exists a $\widehat{\sigma} \in \widehat{\mathcal{W}}_v^p$ that is r-compatible to ρ; thus proving that $\widehat{\mathcal{W}}_v^p \subseteq^r_{\text{orep}} \mathcal{W}_v^p$. As the bound on $|\widehat{\mathcal{W}}_v^p|$ then follows from Corollary 7, we are done once the claim is proven. We will prove the claim first for $p \geq r$ and afterwards for $p < r$.

If $p \geq r$, then σ is an r-sequence $(a_2, \ldots, a_{r+1})$ and there exists an r-rainbow length-p s-v walk W whose color sequence ends on σ. Let a_1 be the color that W visits just before visiting the colors in σ, that is, the color sequence of W ends on $(a_1, \ldots, a_{r+1})$. Further, let u be the penultimate vertex visited by W and let W' be the length-$(p-1)$ subwalk of W ending on u. Then the color sequence of W' ends on $\sigma' := (a_1, \ldots a_r)$. Let $\rho' := (a_{r+1}) \circ \rho$. Observe that σ' is r-compatible to ρ', due to σ being r-compatible to ρ and W' being r-rainbow. Thus, by our induction hypothesis and the definition of ordered r-representatives, there exists a sequence $\widehat{\sigma}' \in \widehat{\mathcal{W}}_u^{p-1}$ that is r-compatible to ρ'. Let $\widehat{W}'$ be the s-u walk corresponding to $\widehat{\sigma}'$ and let $\widehat{\sigma}' := (\widehat{a}'_1, \ldots, \widehat{a}'_r)$. Define $\widehat{\sigma} := (\widehat{a}'_2, \ldots, \widehat{a}'_r) \circ (a_{r+1})$. As $u \in N^-(v)$ and $c(v) = a_{r+1} \notin \{\widehat{a}'_1, \ldots, \widehat{a}'_r\}$, we have that $\widehat{\sigma} \in \mathcal{N}_v^p$. Finally, observe that $\widehat{\sigma}$ is r-compatible with ρ. Thus, $\mathcal{N}_v^p \subseteq^r_{\text{orep}} \mathcal{W}_v^p$. Since $\widehat{\mathcal{W}}_v^p \subseteq^r_{\text{orep}} \mathcal{N}_v^p$, the claim follows for $p \geq r$ due to the transitivity of ordered r-representatives (Observation 8).

If $p < r$, then σ is a $(p+1)$-sequence $(a_1, \ldots, a_{p+1})$ and there exists an r-rainbow length-p s-v walk W whose color sequence ends on σ. Indeed, σ is the entire color sequence of W. This case is similar to the above, but there is no color that is visited before σ in W. Hence, in this case, $\sigma' := (a_1, \ldots, a_p)$ and $\widehat{\sigma} := \widehat{\sigma} \circ (a_{p+1})$. The remainder of the proof is the same. □

Next, we show that the algorithm runs in the claimed running time. Theorem 1 then follows from Lemmas 9 and 10.

Lemma 10 (★). *Algorithm 1 runs in $\mathcal{O}((r \cdot e)^{\omega r} \cdot \ell m)$ time on m-arc digraphs.*

Bounding the length of the walk. Note that the length of a walk may be significantly longer than the running time of the above algorithm for LOCALLY RAINBOW WALK. Hence, Theorem 1 does not imply fixed-parameter tractability for the problem of finding an r-rainbow s-t walk of *any* length. We can however show that we can always find an r-rainbow s-t walk in which the number of visits to each vertex is bounded by a function in r. The idea is as follows. Consider a vertex v that is visited multiple times by an r-rainbow walk W. Relevant for us are the consecutive subsequences of length $r-1$ of the color sequence τ of W that appear immediately before and after each visit of v. Consider the i-th visit and let σ_i and ρ_i be the consecutive length-$(r-1)$ subsequences of τ before and after the i-th visit of v, that is, the sequence $\sigma_i \circ c(v) \circ \rho_i$ is a consecutive subsequence of τ. Now, if for a later visit, say the j-th visit of v, we have that $\sigma_i \circ c(v)$ is r-compatible to ρ_j, then we can *skip* all vertices between i and j. We will show that the number of visits to v is bounded by a function in r, or else we can skip visits. For this, we will make use of a skewed variant (Frankl 1982) of the seminal Bollobás' Two Families Theorem (Bollobás 1965). Details are deferred to the appendix.

Combining the above with Theorem 1, we can prove that deciding whether a graph contains an r-rainbow s-t walk is fixed-parameter tractable when parameterized by r.

Corollary 11 (★). *Given a vertex-colored m-arc digraph and two vertices s and t, one can decide in $\mathcal{O}((r \cdot e)^{r(\omega+1)} \cdot m)$ time whether the graph contains an r-rainbow s-t walk.*

A Matching Lower Bound

A close look at the above algorithm shows that using the algorithm by Fomin et al. (2016) to compute ordered r-representatives is actually not optimal as the underlying unordered representative family also stores representatives for any set Y which does not correspond to $\pi'(\rho)$ for any sequence ρ. This raises hope for a more efficient algorithm. We can however show that finding an algorithm with a running time with an exponent that is asymptotically smaller than ours (Theorem 1) would break the Exponential Time Hypothesis (ETH) (Impagliazzo and Paturi 2001). The provided reduction also proves the problem to be NP-hard. It also holds for shortest walks, and thus also for shortest paths. Altogether, we will prove the following in this section.

Theorem 12 (★). *Even if $\ell = \operatorname{dist}(s,t)$ and on acyclic digraphs, both* LOCALLY RAINBOW WALK *and* LOCALLY RAINBOW PATH *are* NP*-hard and, unless the ETH fails, cannot be solved in $2^{o(r \log r)} \cdot n^{\mathcal{O}(1)}$ time on n-vertex digraphs.*

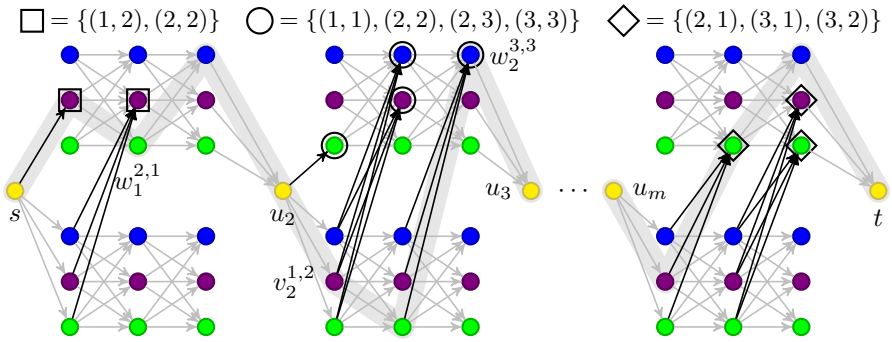

Figure 2: Example for the construction of Theorem 12 on a 3×3-grid with $\mathcal{F} = \{\square, \bigcirc, \ldots, \diamond\}$. Gray (thin) arcs exist independently of the current subset $F_i \in \mathcal{F}$. Black arcs point to elements in F_i and are the only way to reach the top copy. The highlighted path selects the hitting set $\{(1,2),(2,1),(3,3)\}$, thereby visiting, i.a., $w_1^{2,1}$, $v_2^{1,2}$, and $w_2^{3,3}$. The path visits one black arc for each $F_i \in \mathcal{F}$.

We will provide a polynomial-time reduction from the $k \times k$ PERMUTATION HITTING SET problem, where one is given a family $\mathcal{F}$ of subsets of a universe $[k] \times [k]$ (which we will treat like a grid with k rows and columns), and one is asked whether there is a *hitting permutation*, that is, a bijection $\varphi\colon [k] \to [k]$ such that each $F \in \mathcal{F}$ contains an element $(i, \varphi(i))$ with $i \in [k]$. Unless the ETH fails, $k \times k$ PERMUTATION HITTING SET cannot be solved in $2^{o(k \log k)} \cdot (k + |\mathcal{F}|)^{\mathcal{O}(1)}$ time (Lokshtanov, Marx, and Saurabh 2018).

The rough idea for the construction is as follows (see Fig. 2 for an illustrative example): For each set in $\mathcal{F}$, we create a pair of copies (lower and upper) of our universe, which we will be able to traverse column by column. Each row receives a color, and the subpath length $r := k$ is set with the intention that we always have to visit the colors in the same order for each set in $\mathcal{F}$. Hence, for each $F \in \mathcal{F}$, we must pick the same permutation. Now, we always start "left" of the two copies for F, and we can always go to the lower copy, but in order to get to the next set in $\mathcal{F}$, we need to get to the upper copy; This is only possible if one element from our permutation is in F. Hence, there is an r-rainbow s-t walk (indeed, by construction, it will always be a path) if and only if there is a hitting permutation for $\mathcal{F}$.

As $r = k$ in our reduction, an algorithm running in time $2^{o(r \log r)} \cdot n^{\mathcal{O}(1)}$ would refute the ETH. Details about the reduction can be found in the appendix.

Remarkably, the ETH lower bound holds even if one asks whether there exists an r-rainbow s-t walk of arbitrary length. This complements the fixed-parameter tractability of Corollary 11.

Corollary 13. *Unless the ETH breaks, there is no $2^{o(r \log r)} \cdot n^{\mathcal{O}(1)}$-time algorithm for the problem of deciding whether there is an r-rainbow s-t walk of arbitrary length in a given vertex-colored acyclic digraph with n vertices.*

Finally, as every r-rainbow s-t walk in the constructed instance is a shortest s-t path, we can add to every arc (u, v) its antiparallel arc (v, u). The resulting graph thus is symmetric.

Corollary 14. *Even if $\ell = \operatorname{dist}(s,t)$ and on symmetric digraphs, both* LOCALLY RAINBOW WALK *and* LOCALLY RAINBOW PATH *are* NP*-hard and, unless the ETH breaks, cannot be solved in $2^{o(r \log r)} \cdot n^{\mathcal{O}(1)}$-time on n-vertex digraphs.*

Paths

In this section, we study the parameterized complexity of LOCALLY RAINBOW PATH with respect to the locality parameter r and the detour length $k := \ell - \operatorname{dist}(s,t)$.

NP-Hardness for Constant Locality Values

We now provide a dichotomy for LOCALLY RAINBOW PATH parameterized by the locality parameter r. Obviously, if $r = 0$ then any s-t path is a solution. We will now show that the problem remains efficiently solvable when $r \leq 2$, but prove NP-hardness for all values $r \geq 3$.

Clearly, if $r > 0$, we can assume that there is no arc (u, v) with $c(u) = c(v)$ in our digraph. Thus, the task of finding a 1-rainbow s-t path (or s-t walk) reduces to finding any s-t path.

Observation 15. *Finding a shortest 1-rainbow s-t walk or s-t path is linear-time solvable.*

As soon as $r \geq 2$, the problem becomes much harder.

Theorem 16 (★). LOCALLY RAINBOW PATH *is* NP*-hard for any fixed value of $r \geq 2$.*

We provide a polynomial-time reduction from 3-SAT, where given a Boolean formula ϕ in conjunctive normal form such that each clause contains exactly three literals (3-CNF), the question is whether there exists a truth assignment to the variables for which ϕ evaluates to true. The problem is known to be NP-hard, even if each variable appears exactly twice positive and twice negative in the given formula (Berman, Karpinski, and Scott 2003, Theorem 1). We provide our construction for $r = 2$ and describe afterwards how it can be adapted to the case when $r > 2$.

In a nutshell, our reduction works as follows (compare with Fig. 3). Our path first needs to go through the variable gadgets, in which there are two branches (for true and

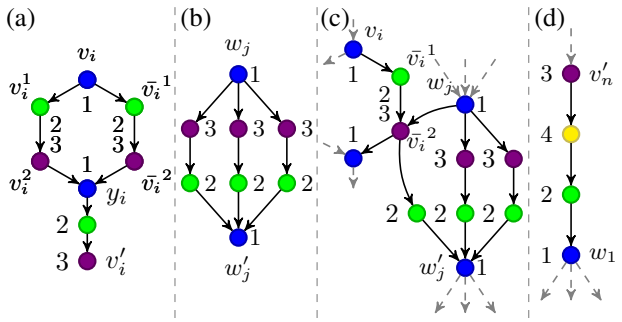

Figure 3: (a) The variable gadget and (b) the clause gadget in the construction of Theorem 16. (c) An example showing how a literal path corresponding to literal $\bar{x}_i$ in clause c_j is attached to the variable gadget at $\bar{v}_i^1$. (d) The connection between the last variable gadget and the first clause gadget.

false) for each variable. Afterwards, it needs to go through the clause gadgets, in which there is a branch for each literal. Each branch visits a vertex of the corresponding variable gadget. As we are looking for a path, this vertex must be on the branch that was not yet visited by our path. Finally, the colors in the graph are chosen such that taking any forbidden turn (e.g., from a variable gadget directly into a clause gadget) would breach the local rainbowness constraint. For details on the proof of Theorem 16 we refer to the appendix.

We close this section by remarking that, on symmetric digraphs, finding a shortest 2-rainbow path becomes efficiently solvable. (The case $r \geq 3$ remains NP-hard by a reduction similar to the one above.) The idea here is to transform the vertex coloring into an edge coloring (i.e., every symmetric arc is assigned one color). We say that a walk is *properly colored* (with respect to some edge coloring) if no two consecutive symmetric arcs share the same color.

Lemma 17 (★). *Let G be a symmetric digraph, c a vertex coloring, and W an s-t walk. Assume that no two adjacent vertices have the same color. Then, W is 2-rainbow if and only if it is properly colored with respect to the edge coloring $c'((u,v)) := \{c(u), c(v)\}$.*

As a properly edge-colored s-t path can be found in linear time in symmetric digraphs (Szeider 2003, Cor. 10), we obtain the following.

Observation 18. *Finding a shortest 2-rainbow s-t walk in a symmetric digraph is solvable in linear time.*

Fixed-Parameter Tractability with Detour Length

We now prove our problem to be fixed-parameter tractable with respect to $r + k$ where k denotes the length of a *detour* the path may take (i.e., the desired length ℓ is $\mathrm{dist}(s,t)+k$).

Let us first exclude some degenerate cases. If $k < 0$, then $\ell < \mathrm{dist}(s,t)$, and we have a trivial no-instance at hand. If $k = 0$, then any solution must be a shortest path. As any shortest walk is also a shortest path, we can use our algorithm for LOCALLY RAINBOW WALK, see Theorem 1. Finally, we may assume that each of the n vertices in G reaches t. In all, we have that $\mathrm{dist}(s,t) < \ell < n$ and thus $0 < k < n - \mathrm{dist}(s,t)$.

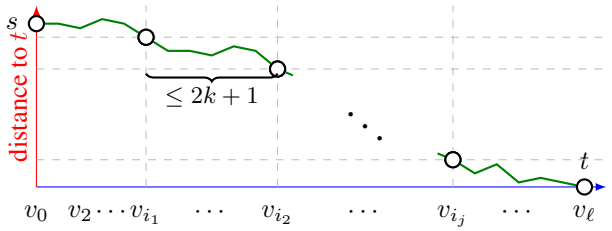

Figure 4: An exemplary s-t path P, circles marking distance separators. The x-axis shows the vertices of P in the order of their appearance. The y-axis shows the distance of the current vertex to t. Our algorithm exploits the property that the subpaths between any two distance separators are short (i.e., of length at most $2k+1$) and internally vertex-disjoint.

Theorem 19 (★). LOCALLY RAINBOW PATH *can be solved in $r^{\mathcal{O}(r+k)} \cdot \ell n^2 m$ time, where n and m are the number of vertices and arcs of the input digraph, and ℓ is the length and k is the detour length of the desired path.*

Our approach for Theorem 19 is to merge our above techniques to keep track of the last r vertices with a central observation for paths with detour length k. To this end, we will show that any hypothetical solution P^* visits in regular intervals so-called *distance separators* — see Fig. 4 for an illustration. At these points, we can partition the search space as we know that the subpath of P^* between two consecutive distance separators lies disjoint from any subpath between two other consecutive distance separators. We then use a subroutine to compute a representative of all r-rainbow u-v paths of some fixed length whose running time is fixed-parameter tractable with respect to its length. This fits into the promised running time as any two distance separators u and v are at most $2k+1$ vertices apart (Lemma 22). Indeed, such distance separators can be found in *any* path with bounded detour length. As mentioned earlier, this approach is inspired by works on parameterizations with respect to the detour length by Bezáková et al. (2019) and Zschoche (2023). The challenge in our setting is that we need to keep track of the ordered representatives.

We start off with a basic observation. We denote for every $v \in V(G)$ by $d(v) := \mathrm{dist}(v,t)$ the distance to t.

Observation 20 (★). *For any s-t path $P = (s = v_0, \ldots, v_\ell = t)$ with $\ell \leq d(s) + k$ we have $i \leq d(s) - d(v_i) + k$ for each $i \in [0, \ell]$.*

Definition 21 (Distance separator)**.** Let $P = (s = v_0, v_1, \ldots, v_\ell = t)$ be a path with detour length $k := \ell - \mathrm{dist}(s,t)$. Then v_i is a *distance separator* if $d(v_i) < d(v_j)$ for all $j < i$ and $d(v_i) > d(v_j)$ for all $j > i$.

By definition, if we have two distance separators v_i and v_j, $j > i$, then we know that between v_i and v_j, P only visits vertices w with $d(v_i) > d(w) > d(v_j)$. Zschoche (2023) showed that a path with detour length k regularly visits distance separators. We need a faintly different statement.

Lemma 22 (★). *Let $P = (s = v_0, v_1, \dots, v_p = v)$ be a path of length at most $d(s) - d(v) + k$ and let v be a distance separator. Then, for all $i \in [0, p - 2k]$, there is a $j \in [0, 2k]$ such that v_{i+j} is a distance separator.*

Our algorithm now guesses the positions of the distance separators. As the subpaths between the distance separators are (internally) vertex-disjoint, we then only need to find an r-rainbow path matching the color sequence of the subpath to the last distance separator and only uses vertices after this distance separator. For any two distance separators u and v in our graph G, we define $G_{u,v} \coloneqq G[B_{u,v} \cup \{u, v\}]$

$$B_{u,v} \coloneqq \begin{cases} \{w \in V(G) \mid d(u) > d(w) > d(v)\} & \text{if } u \neq s, \\ \{w \in V(G) \mid d(w) > d(v)\} & \text{if } u = s. \end{cases}$$

On these graphs, we will compute an ordered r-representative for the family of r-rainbow u-v paths of some length in $G_{u,v}$. Indeed, as we will append these paths to some r-rainbow s-u path that ends on some color sequence τ, we need the family to be r-compatible with τ. We say that a path $P = (v_0, \dots, v_q)$ *fits* τ if τ is r-compatible to $(c(v_0), \dots c(v_{\min\{r-1,q\}}))$. We will need to compute an ordered r-representative for the following family for any two distance separators u and v in G, any integer $q \in \mathbb{N}_0$, and any color sequence τ of length at most r:

$$\mathcal{P}^q_\tau(G_{u,v}) \coloneqq \left\{ \sigma \;\middle|\; \begin{array}{l} |\sigma| = \min\{q+1, r\} \text{ and there is} \\ \text{an } r\text{-rainbow length-}q\ u\text{-}v \text{ path} \\ \text{in } G_{u,v} \text{ that fits } \tau \text{ and whose} \\ \text{color sequence ends on } \sigma \end{array} \right\}$$

Computing such families can be done with an adaptation of Algorithm 1 for walks, the difference being that we additionally need to remember the set of vertices visited so far by our path (refer to the full version). Remembering these vertices comes at the cost of an additional running time factor of $r^{\mathcal{O}(q)}$, where q is the length of the path. As the path length is an upper bound for r, this proves LOCALLY RAINBOW PATH to be fixed-parameter tractable with respect to the path length.

Lemma 23 (★). *Given a digraph G with m arcs, an integer r, two distance-separators $u, v \in V(G)$, an integer $q \in \mathbb{N}_0$, and a color sequence τ of length at most r, one can compute in $r^{\mathcal{O}(r+q)} \cdot m$ time an ordered r-representative for $\mathcal{P}^q_\tau(G_{u,v})$ of size at most $r^{\mathcal{O}(r+q)}$.*

Now that we know how to compute the families $\widehat{\mathcal{P}}^q_\tau(G_{u,v})$, we can state the main algorithm. Herein, for every $p \in [0, \ell]$ and $v \in V(G)$, we are interested in the family

$$\mathcal{R}^p_v \coloneqq \left\{ \sigma \;\middle|\; \begin{array}{l} |\sigma| = \min\{p+1, r\} \text{ and there is an} \\ r\text{-rainbow length-}p\ s\text{-}v \text{ path in } G_{s,v} \\ \text{whose color sequence ends on } \sigma \end{array} \right\}. \quad (3)$$

Hence, there is a length-ℓ r-rainbow s-t path P^* if and only if $\mathcal{R}^\ell_t$ is nonempty. As, by Observation 20, P^* will have v as its p-th vertex only if $p \leq d(s) - d(v) + k$, we only need to consider those families $\mathcal{R}^p_v$ for which this inequality holds.

Algorithm 2. Set $\widehat{\mathcal{R}}^0_s \coloneqq \{(c(s))\}$ and $\widehat{\mathcal{R}}^0_v \coloneqq \emptyset$ for all $v \in V(G) \setminus \{s\}$. Now, for each $p = 1, 2, \dots, \ell$, for each $v \in V(G)$ with $p \leq d(s) - d(v) + k$, compute $\mathcal{S}^p_v$, which is

$$\bigcup_{\substack{u \in V(G), \\ q \in [\min\{2k+1, p\}]}} \left\{ \sigma' \circ \sigma \;\middle|\; \begin{array}{l} \exists \sigma'' \colon \sigma'' \circ \sigma' \in \widehat{\mathcal{R}}^{p-q}_u, \\ |\sigma'| = \max\{0, \min\{p-q+1, r-q\}\} \\ |\sigma| = \min\{q, r\}, \\ \text{and } \sigma \in \widehat{\mathcal{P}}^q_{(\sigma'' \circ \sigma')}(G_{u,v}) \end{array} \right\}$$

and compute $\widehat{\mathcal{R}}^p_v \subseteq^r_{\text{orep}} \mathcal{S}^p_v$ using Corollary 7. Return `yes` if and only if $\widehat{\mathcal{R}}^{\ell'}_t \neq \emptyset$ for some $\ell' \in [\ell]$.

To prove Theorem 19, we still need to analyze the running time and show that that $\widehat{\mathcal{R}}^p_v$ is an ordered r-representative for $\mathcal{R}^p_v$.

Lemma 24 (★). *For each $v \in V(G)$ and $p \in [0, \ell]$ with $p \leq d(s) - d(v) + k$, the family $\widehat{\mathcal{R}}^p_v$ computed in Algorithm 2 is of size at most $(r \cdot e)^r$ and is an ordered r-representative for $\mathcal{R}^p_v$ as defined in* (3)*. Moreover, Algorithm 2 runs in $r^{\mathcal{O}(r+k)} \cdot \ell n^2 m$ time.*

Conclusion

We introduced a *local rainbow* constraint to the classic problem of finding s-t paths and walks, modeling scenarios in which resources (i.e., colors) are replenished over time. For walks, we are able to prove fixed-parameter tractability for the locality parameter r thanks to a new adaptation of the representative sets technique. In contrast, LOCALLY RAINBOW PATH remains NP-hard even for constant r due to the added non-local constraint of forbidding self-intersections. However, when the allowed length of the path is not too large in comparison to the distance between its endpoints, then the no-intersection constraints effectively become local again. This is exploited to prove LOCALLY RAINBOW PATH to be fixed-parameter tractable with the combined parameter $r + k$ where k is the detour length.

Towards future work, we believe that local rainbowness is only the tip of the iceberg when it comes to interesting local constraints. A straightforward generalization would be to allow for multi-colored vertices, so as to model a setting in which multiple types of resources can be used at once. Another natural variant would be to relax the local rainbowness constraint of the subpaths, allowing some bounded number of vertices to share the same color.

One could also extend our local rainbowness constraint to other connectivity problems. Canonical candidates would be the traveling salesperson problem or the problem of finding multiple disjoint s-t paths. Similarly, one could be interested in finding Steiner trees in which all subpaths are locally rainbow (a generalization of rainbow Steiner trees (Ferone, Festa, and Guerriero 2022)), or vertex sets whose deletion destroys all s-t paths that are not locally rainbow.

We already observed that in practice, the locality constraint is usually motivated by some regeneration over time. Therefore, it may be sensible to study the local rainbowness constraints also on temporal graphs, such as finding temporal walks (Bentert et al. 2020) or paths (Casteigts et al. 2021; Zschoche 2023).

Acknowledgments

Till Fluschnik was supported by the DFG, project AFFA (BR 5207/1).

References

Adler, J. D.; Mirchandani, P. B.; Xue, G.; and Xia, M. 2016. The electric vehicle shortest-walk problem with battery exchanges. *Networks and Spatial Economics*, 16(1): 155–173.

Agrawal, A.; Jain, P.; Kanesh, L.; and Saurabh, S. 2020. Parameterized Complexity of Conflict-Free Matchings and Paths. *Algorithmica*, 82(7): 1939–1965.

Alman, J.; and Williams, V. V. 2021. A Refined Laser Method and Faster Matrix Multiplication. In *Proceedings of the 2021 ACM-SIAM Symposium on Discrete Algorithms (SODA '21)*, 522–539.

Alon, N.; Yuster, R.; and Zwick, U. 1995. Color-Coding. *Journal of the ACM*, 42(4): 844–856.

Bang-Jensen, J.; and Gutin, G. Z. 2009. *Digraphs - Theory, Algorithms and Applications, Second Edition*. Springer Monographs in Mathematics. Springer. ISBN 978-1-84800-997-4.

Benini, M.; Blasi, E.; Detti, P.; and Fosci, L. 2023. Solving crop planning and rotation problems in a sustainable agriculture perspective. *Computers and Operations Research*, 159: 106316.

Bentert, M.; Himmel, A.; Nichterlein, A.; and Niedermeier, R. 2020. Efficient computation of optimal temporal walks under waiting-time constraints. *Applied Network Science*, 5(1): 73.

Bentert, M.; Kellerhals, L.; and Niedermeier, R. 2023. Fair Short Paths in Vertex-Colored Graphs. In *Proceedings of the 37th Conference on Artificial Intelligence (AAAI '23)*, 12346–12354. AAAI Press.

Berman, P.; Karpinski, M.; and Scott, A. D. 2003. Approximation Hardness of Short Symmetric Instances of MAX-3SAT. *Electronic Colloquium on Computational Complexity*, TR03-049.

Bezáková, I.; Curticapean, R.; Dell, H.; and Fomin, F. V. 2019. Finding Detours is Fixed-Parameter Tractable. *SIAM Journal on Discrete Mathematics*, 33(4): 2326–2345.

Bhattacharya, S. 2010. Search-Based Path Planning with Homotopy Class Constraints. In Fox, M.; and Poole, D., eds., *Proceedings of the 24th Conference on Artificial Intelligence (AAAI '10)*. AAAI Press.

Bollobás, B. 1965. On generalized graphs. *Acta Mathematica Academiae Scientiarum Hungarica*, 16: 447–452.

Broersma, H.; and Li, X. 1997. Spanning trees with many or few colors in edge-colored graphs. *Discussiones Mathematicae Graph Theory*, 17(2): 259–269.

Casteigts, A.; Himmel, A.; Molter, H.; and Zschoche, P. 2021. Finding Temporal Paths Under Waiting Time Constraints. *Algorithmica*, 83(9): 2754–2802.

Chen, L.; Li, X.; and Shi, Y. 2011. The complexity of determining the rainbow vertex-connection of a graph. *Theoretical Computer Science*, 412(35): 4531–4535.

Cygan, M.; Fomin, F. V.; Kowalik, L.; Lokshtanov, D.; Marx, D.; Pilipczuk, M.; Pilipczuk, M.; and Saurabh, S. 2015. *Parameterized Algorithms*. Springer.

Darmann, A.; Pferschy, U.; Schauer, J.; and Woeginger, G. J. 2011. Paths, trees and matchings under disjunctive constraints. *Discrete Applied Mathematics*, 159(16): 1726–1735.

de Uña, D.; Gange, G.; Schachte, P.; and Stuckey, P. J. 2016. Steiner Tree Problems with Side Constraints Using Constraint Programming. In *Proceedings of the 30th Conference on Artificial Intelligence (AAAI '16)*, 3383–3389. AAAI Press.

Diestel, R. 2016. *Graph Theory*, volume 173 of *Graduate Texts in Mathematics*. Springer, 5th edition.

Dury, J.; Schaller, N.; Garcia, F.; Reynaud, A.; and Bergez, J. E. 2012. Models to suport cropping plan and crop rotation decisions. A review. *Agronomy for Sustainable Development*, 32: 567–580.

Ferone, D.; Festa, P.; and Guerriero, F. 2022. The Rainbow Steiner Tree Problem. *Computers and Operations Research*, 139: 105621.

Fomin, F. V.; Lokshtanov, D.; Panolan, F.; and Saurabh, S. 2016. Efficient Computation of Representative Families with Applications in Parameterized and Exact Algorithms. *Journal of the ACM*, 63(4): 29:1–29:60.

Ford, B. T.; Aggarwal, R.; Kumar, M.; Manyam, S. G.; Casbeer, D.; and Grymin, D. 2022. Backtracking Hybrid A* for Resource Constrained Path Planning. In *Proceedings of the AIAA Scitech 2022 Forum (AIAA '22)*, 1592.

Frankl, P. 1982. An Extremal Problem for Two Families of Sets. *European Journal of Combinatorics*, 3: 125–127.

Halldórsson, M. M.; Kortsarz, G.; Mitra, P.; and Tonoyan, T. 2018. Spanning Trees With Edge Conflicts and Wireless Connectivity. In *Proceedings of the 45th International Colloquium on Automata, Languages, and Programming (ICALP '18)*, volume 107, 158:1–158:15. Schloss Dagstuhl - Leibniz-Zentrum für Informatik.

Handler, G. Y.; and Zang, I. 1980. A dual algorithm for the constrained shortest path problem. *Networks*, 10(4): 293–309.

Impagliazzo, R.; and Paturi, R. 2001. On the Complexity of k-SAT. *Journal of Computer and System Sciences*, 62(2): 367–375.

Impagliazzo, R.; Paturi, R.; and Zane, F. 2001. Which Problems Have Strongly Exponential Complexity? *Journal of Computer and System Sciences*, 63(4): 512–530.

Irnich, S.; and Desaulniers, G. 2005. *Shortest Path Problems with Resource Constraints*, 33–65. Springer US.

Jacob, A.; Włodarczyk, M.; and Zehavi, M. 2023. Long Directed Detours: Reduction to 2-Disjoint Paths. arXiv:2301.06105.

Kowalik, L.; and Lauri, J. 2016. On finding rainbow and colorful paths. *Theoretical Computer Science*, 628: 110–114.

Lokshtanov, D.; Marx, D.; and Saurabh, S. 2018. Slightly Superexponential Parameterized Problems. *SIAM Journal on Computing*, 47(3): 675–702.

Pugliese, L. D. P.; and Guerriero, F. 2013. A survey of resource constrained shortest path problems: Exact solution approaches. *Networks*, 62(3): 183–200.

Smith, O. J.; Boland, N.; and Waterer, H. 2012. Solving shortest path problems with a weight constraint and replenishment arcs. *Computers and Operations Research*, 39(5): 964–984.

Szeider, S. 2003. Finding paths in graphs avoiding forbidden transitions. *Discrete Applied Mathematics*, 126(2-3): 261–273.

Turchetta, M.; Corinzia, L.; Sussex, S.; Burton, A.; Herrera, J.; Athanasiadis, I.; Buhmann, J. M.; and Krause, A. 2022. Learning Long-Term Crop Management Strategies with CyclesGym. In *Proceedings of the 35rd Annual Coference on Advances in Neural Information Processing Systems (NeurIPS '22)*.

Uchizawa, K.; Aoki, T.; Ito, T.; Suzuki, A.; and Zhou, X. 2013. On the Rainbow Connectivity of Graphs: Complexity and FPT Algorithms. *Algorithmica*, 67(2): 161–179.

Zschoche, P. 2023. Restless Temporal Path Parameterized Above Lower Bounds. In *Proceedings of the 40th International Symposium on Theoretical Aspects of Computer Science (STACS '23)*, 55:1–55:16. Schloss Dagstuhl - Leibniz-Zentrum für Informatik.

Zündorf, T. 2014. *Electric Vehicle Routing with Realistig Recharging Models*. Master thesis, Karlsruhe Institute of Technology.