Finding Temporal Paths Under Waiting Time Constraints

Arnaud Casteigts
LaBRI, Université de Bordeaux, CNRS, Bordeaux INP, France
arnaud.casteigts@labri.fr

Anne-Sophie Himmel
Technische Universität Berlin, Algorithmics and Computational Complexity, Germany
anne-sophie.himmel@tu-berlin.de

Hendrik Molter
Technische Universität Berlin, Algorithmics and Computational Complexity, Germany
h.molter@tu-berlin.de

Philipp Zschoche
Technische Universität Berlin, Algorithmics and Computational Complexity, Germany
zschoche@tu-berlin.de

Abstract

Computing a (short) path between two vertices is one of the most fundamental primitives in graph algorithmics. In recent years, the study of paths in temporal graphs, that is, graphs where the vertex set is fixed but the edge set changes over time, gained more and more attention. A path is time-respecting, or *temporal*, if it uses edges with non-decreasing time stamps.

We investigate a basic constraint for temporal paths, where the time spent at each vertex must not exceed a given duration Δ, referred to as Δ-*restless temporal paths*. This constraint arises naturally in the modeling of real-world processes like packet routing in communication networks and infection transmission routes of diseases where recovery confers lasting resistance.

While finding temporal paths without waiting time restrictions is known to be doable in polynomial time, we show that the "restless variant" of this problem becomes computationally hard even in very restrictive settings. For example, it is W[1]-hard when parameterized by the feedback vertex number or the pathwidth of the underlying graph. The main question thus is whether the problem becomes tractable in some natural settings. We explore several natural parameterizations, presenting FPT algorithms for three *kinds* of parameters: (1) output-related parameters (here, the maximum length of the path), (2) classical parameters applied to the underlying graph (e.g., feedback edge number), and (3) a new parameter called *timed feedback vertex number*, which captures finer-grained temporal features of the input temporal graph, and which may be of interest beyond this work.

2012 ACM Subject Classification Mathematics of computing → Graph algorithms

Keywords and phrases Temporal graphs, disease spreading, waiting-time policies, restless temporal paths, timed feedback vertex set, NP-hard problems, parameterized algorithms

Digital Object Identifier 10.4230/LIPIcs.ISAAC.2020.30

Related Version Full version on arXiv: `https://arxiv.org/abs/1909.06437`.

Funding *Arnaud Casteigts*: Supported by the ANR, project ESTATE (ANR-16-CE25-0009-03).
Anne-Sophie Himmel: Supported by the DFG, project FPTinP (NI 369/16).
Hendrik Molter: Supported by the DFG, project MATE (NI 369/17).

1 Introduction

A highly successful strategy to control (or eliminate) outbreaks of infectious diseases is contact tracing [18] – whenever an individual is diagnosed positively, every person who is possibly infected by this individual is put into quarantine. However, the viral spread can be too fast to be traced manually, e.g., if the disease is transmittable in a pre-symptomatic (or asymptomatic) stage, then it seems likely that an individual already caused infection chains when diagnosed positively. Hence, large-scale digital systems are recommended which use physical proximity networks based on location and contact data [20] – this allows fast and precise contact tracing while avoiding the harmful effect of mass quarantines to society [20]. Physical proximity networks can be understood as temporal graphs[1] [13, 28, 30, 36, 40], that is, graphs where the vertex set (individuals) remains static but the edge set (physical contacts) may change over time. In this paper, we extend the literature on reachability in temporal graphs [4, 5, 10, 11, 27, 34, 39, 48] by a computational complexity analysis of an important variation of one of the most fundamental combinational problems arising in the above mentioned scenario: given a temporal graph and two individuals s and z, is a chain of infection from s to z possible, that is, is there a *temporal path* from s to z? In particular, we use a reachability concept that captures the standard 3-state *SIR-model* (Susceptible-Infected-Recovered), a canonical spreading model for diseases where recovery confers lasting resistance [6, 35, 42].

In temporal graphs, the basic concepts of paths and reachability are defined in a time-respecting way [34]: a (strict) temporal path, also called "journey", is a path that uses edges with non-decreasing (increasing) time steps. To represent infection chains in the SIR-model, we restrict the time of "waiting" or "pausing" at each intermediate vertex to a prescribed duration[2]. We call these paths *restless* temporal paths. They model infection transmission routes of diseases that grant immunity upon recovery [29]: An infected individual can transmit the disease until it is recovered (reflected by bounded waiting time) and it cannot be infected a second time afterwards since then it is immune (reflected by considering path instead of walk: every vertex can only be visited at most once). Another natural example of restless temporal paths is delay-tolerant networking among mobile entities, where the routing of a packet is performed over time and space by storing the packet for a limited time at intermediate nodes.

In the following we give an example to informally describe our problem setting[3]. In Figure 1 we are given the depicted temporal graph, vertices s and z, and the time bound $\Delta = 2$. We are asked to decide whether there is a restless temporal path from s to z, that is, a path which visits each vertex at most once and pauses at most Δ units of time between consecutive hops. Here, (s, d, b, z) is a feasible solution, but (s, b, z) is not because the waiting time at b exceeds Δ. The walk (s, b, c, d, b, z) is not a valid solution because it visits vertex b twice. Finally, (s, a, c, d, b, z) is also a feasible solution.

Related work. Several types of waiting time constraints have been considered in the temporal graph literature. An empirical study by Pan and Saramäki [44] based on phone calls datasets observed a threshold in the correlation between the duration of pauses between calls and the ratio of the network reached over a spreading process. Casteigts et al. [12]

[1] Also known as time-varying graphs, evolving graphs, or link streams.
[2] Note that the latent period of an infectious disease can be modeled by subdividing time edges.
[3] We refer to Section 2 for a formal definition.

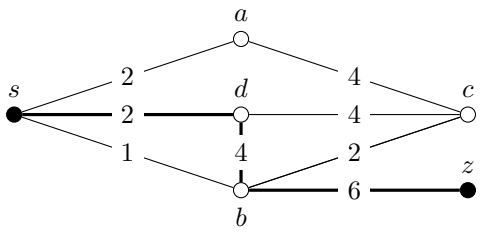

Figure 1 Example of a temporal graph whose edges are labeled with time stamps. Bold edges depict a 2-restless temporal (s, z)-path. (In general, multiple time stamps per edge are possible.)

showed a dramatic impact of waiting time constraints to the expressivity of a temporal graph, when considering such a graph as an automaton and temporal paths as words. In the context of temporal flows, Akrida et al. [2] considered a concept of "vertex buffers", which however pertains to the quantity of information that a vertex can store, rather than a duration. Enright et al. [19] considered deletion problems for reducing temporal connectivity. Closer related to our work, Himmel et al. [27] studied a variant of restless temporal paths where several visits to the same vertex are allowed, i.e., restless temporal *walks*. They showed, among other things, that such walks can be computed in polynomial time.

Many path-related problems have been studied in the temporal setting and the nature of temporal paths significantly increase the computational complexity of many of them (compared to their static counterparts). In the temporal setting, reachability is not an equivalence relation among vertices which makes many problems more complicated. For example, finding a maximum temporally connected component is NP-hard [10]. We further have that in a temporal graph, spanning trees may not exist. In fact, even the existence of *sparse* spanners (i.e., subgraphs with $o(n^2)$-many edges ensuring temporal connectivity) is not guaranteed [5], unless the underlying graph is complete [14], and computing a minimum-cardinality spanner is APX-hard [3, 39]. Yet another example is the problem of deciding whether there are k disjoint temporal paths between two given vertices. In a seminal article, Kempe et al. [34] showed that this problem, whose classical analogue is (again) polynomial-time solvable, becomes NP-hard. They further investigated the related problem of finding temporal separators, which is also NP-hard [21, 34, 49].

Our contributions. We introduce the problem Restless Temporal Path. To get a finer understanding of the computational complexity of this problem, we turn our attention to its parametrized complexity. In stark contrast to both restless temporal walks and non-restless temporal paths, we show that this problem is NP-hard and W[1]-hard when parameterized by the (vertex deletion) *distance to disjoint paths* of the underlying graph[4] (Section 3). This is tight in the sense that the problem can be solved in polynomial time when the underlying graph is a forest. On the positive side, we explore parameters of three different natures. First, we show that the problem is fixed-parameter tractable (FPT) for the *number of hops* of the temporal path (Section 4). We further show that the problem is FPT when parameterized by the *feedback edge number* of the underlying graph (Section 5). Additionally, we show that the previously mentioned parameterizations that allow for FPT results, the problem presumably does not admit a polynomial kernel. Our results provide a fine-grained characterization of

[4] The *underlying graph* of a temporal graph is a static graph that contains every edge that appears at least once in the temporal graph. A formal definition is given in Section 2.

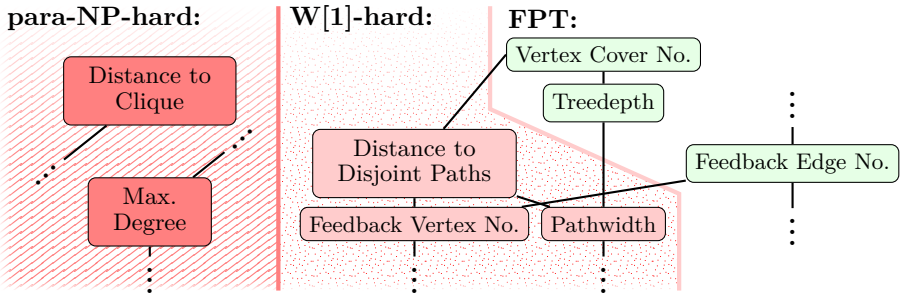

Figure 2 Relevant part of the hierarchy among classic parameters of the underlying graph (cf. Sorge et al. [45]) for our results for Restless Temporal Path.

the tractability boundary of the computation of restless temporal paths for parameters of the underlying graph, as illustrated by the vicinity of the corresponding parameters in Figure 2. Then, going beyond parameters related to the output and to the underlying graph, we define a novel temporal version of the classic *feedback vertex number* called *timed feedback vertex number*. Intuitively, it counts the number of vertex *appearances* that have to be removed from the temporal graph such that its underlying graph becomes cycle-free. We show that finding restless temporal paths is FPT when parameterized by this parameter (Section 6). We believe that the latter is an interesting turn on events compared to our hardness results. Due to space constraints, most proofs (marked with ★) are deferred to the related full version on arXiv.[5]

2 Preliminaries

Here, we formally introduce the most important concepts related to temporal graphs and paths, and give the formal problem definition of (Short) Restless Temporal (s, z)-Path.

An *interval* is an ordered set $[a, b] := \{n \mid n \in \mathbb{N} \wedge a \leq n \leq b\}$, where $a, b \in \mathbb{N}$. Further, let $[a] := [1, a]$. We use standard notation and terminology from (static) graph theory [16] and parameterized complexity theory [15, 17].

Temporal graphs. An (undirected, simple) *temporal graph* is a tuple $\mathcal{G} = (V, E_1, \ldots, E_\ell)$ (or $\mathcal{G} = (V, (E_i)_{i\in[\ell]})$ for short), with $E_i \subseteq \binom{V}{2}$ for all $i \in [\ell]$. We call $\ell(\mathcal{G}) := \ell$ the *lifetime* of $\mathcal{G}$. As with static graphs, we assume all temporal graphs in this paper to be undirected and simple. We call the graph $G_i(\mathcal{G}) = (V, E_i(\mathcal{G}))$ the *layer* i of $\mathcal{G}$ where $E_i(\mathcal{G}) := E_i$. If $E_i = \emptyset$, then G_i is a *trivial* layer. We call layers G_i and G_{i+1} *consecutive*. We call i a *time step*. If an edge e is present at time i, that is, $e \in E_i$, we say that e has *time stamp* i. We further denote $V(\mathcal{G}) := V$. The *underlying graph* $G_\downarrow(\mathcal{G})$ of $\mathcal{G}$ is defined as $G_\downarrow(\mathcal{G}) := (V, \bigcup_{i=1}^{\ell(\mathcal{G})} E_i(\mathcal{G}))$. To improve readability, we remove $(\mathcal{G})$ from the introduced notations whenever it is clear from the context. For every $v \in V$ and every time step $t \in [\ell]$, we denote the *appearance of vertex* v *at time* t by the pair (v, t). For every $t \in [\ell]$ and every $e \in E_t$ we call the pair (e, t) a *time edge*. For a time edge $(\{v, w\}, t)$ we call the vertex appearances (v, t) and (w, t) its *endpoints*. We assume that the *size* (for example when referring to input sizes in running time analyzes) of $\mathcal{G}$ is $|\mathcal{G}| := |V| + \sum_{i=1}^{\ell} |E_i|$, that is, we do not assume that we have compact representations of temporal graphs. Finally, we write n for $|V|$.

[5] https://arxiv.org/abs/1909.06437

A *temporal (s, z)-walk* (or *temporal walk*) of length k from vertex $s = v_0$ to vertex $z = v_k$ in a temporal graph $\mathcal{G} = (V, (E_i)_{i\in[\ell]})$ is a sequence $P = ((v_{i-1}, v_i, t_i))_{i=1}^{k}$ of triples that we call *transitions* such that for all $i \in [k]$ we have that $\{v_{i-1}, v_i\} \in E_{t_i}$ and for all $i \in [k-1]$ we have that $t_i \leq t_{i+1}$. Moreover, we call P a *temporal (s, z)-path* (or *temporal path*) of length k if $v_i \neq v_j$ for all $i, j \in \{0, \dots, k\}$ with $i \neq j$. Given a temporal path $P = ((v_{i-1}, v_i, t_i))_{i=1}^{k}$, we denote the set of vertices visited by P by $V(P) = \{v_0, v_1, \dots, v_k\}$. A *restless* temporal path is not allowed to wait an arbitrary amount of time in a vertex, but has to leave any vertex it visits within the next Δ time steps, for some given value of Δ. Analogously to the non-restless case, a restless temporal walk may visit a vertex multiple times.

▶ **Definition 1.** *A temporal path (walk) $P = ((v_{i-1}, v_i, t_i))_{i=1}^{k}$ is* Δ-restless *if $t_i \leq t_{i+1} \leq t_i + \Delta$, for all $i \in [k-1]$. We say that P* respects *the waiting time Δ.*

Having this definition at hand, we are ready to define the main decision problem of this work.

Restless Temporal Path	
Input:	A temporal graph $\mathcal{G} = (V, (E_i)_{i\in[\ell]})$, two distinct vertices $s, z \in V$, and an integer $\Delta \leq \ell$.
Question:	Is there a Δ-restless temporal (s, z)-path in $\mathcal{G}$?

Note the waiting time at the source vertex s is ignored. This is without loss of generality, since one can add an auxiliary degree one source vertex which is only in the first layer adjacent to s. We also consider a variant, where we want to find Δ-restless paths of a certain maximum length. In the Short Restless Temporal Path problem, we are additionally given a integer $k \in \mathbb{N}$ and the question is whether there is a Δ-restless temporal path of length at most k from s to z in $\mathcal{G}$? Note that Restless Temporal Path is the special case of Short Restless Temporal Path for $k = |V| - 1$ and that both problems are in NP.

Parameterized complexity. We use standard notation and terminology from parameterized complexity theory [15] and give here a brief overview of the most important concepts that are used in this paper. A *parameterized problem* is a language $L \subseteq \Sigma^* \times \mathbb{N}$, where Σ is a finite alphabet. We call the second component the *parameter* of the problem. A parameterized problem is *fixed-parameter tractable* (in the complexity class FPT) if there is an algorithm that solves each instance (I, r) in $f(r) \cdot |I|^{O(1)}$ time, for some computable function f. A decidable parameterized problem L admits a *polynomial kernel* if there is a polynomial-time algorithm that transforms each instance (I, r) into an instance (I', r') such that $(I, r) \in L$ if and only if $(I', r') \in L$ and $|(I', r')| \in r^{O(1)}$. If a parameterized problem is hard for the parameterized complexity class W[1], then it is (presumably) not in FPT. The complexity classes W[1] is closed under parameterized reductions, which may run in FPT-time and additionally set the new parameter to a value that exclusively depends on the old parameter.

Basic observations. If there exists a Δ-restless temporal (s, z)-path $P = ((v_{i-1}, v_i, t_i))_{i=1}^{k}$ in a temporal graph $\mathcal{G}$, then $P' = \big(\{v_0, v_1\}, \dots, \{v_{k-1}, v_k\}\big)$ is an (s, z)-path in the underlying graph $G_\downarrow$. The other direction does not necessarily hold, but for any (s, z)-path in $G_\downarrow$ we can decide in linear time whether this path forms a Δ-restless temporal (s, z)-path in $\mathcal{G}$. As a consequence, we can decide Restless Temporal Path in linear time for any temporal graph where there exists a unique (s, z)-path in the underlying graph, in particular, if the underlying graph is a forest. Some further basic observations are deferred to the full version of this paper.

▶ **Lemma 2.** *Let $\mathcal{G} = (V, (E_i)_{i\in[\ell]})$ be a temporal graph where the underlying graph $G_\downarrow$ is an (s, z)-path with $s, z \in V$. Then there is an algorithm which computes in $O(|\mathcal{G}|)$ time the set $\mathcal{A} = \{t \mid$ there is a Δ-restless temporal (s, z)-path with arrival time $t\}$.*

Proof. Let $V(G_\downarrow) = \{s = v_0, \ldots, v_n = z\}$ be the set of vertices and $E(G_\downarrow) = \{e_1 = \{v_0, v_1\}, \ldots, e_n = \{v_{n-1}, v_n\}\}$ be the set of edges of the underlying path. We further define L_i as the set of layers of $\mathcal{G}$ in which the edge $e_i \in E(G_\downarrow)$ exists, that is, $L_i := \{t \mid e_i \in E_t\}$.

In the following, we construct a dynamic program on the path. We compute for every vertex v_i the table entry $T[v_i]$ which is defined as the set of all layers t such that there exists a Δ-restless temporal (s, v_i)-path with arrival time t. It holds that $T[v_1] = L_1$. Now we can compute the table entries successively:

$$T[v_i] = \{t \in L_i \mid \text{there is a } t' \in T[v_{i-1}] \text{ with } 0 \leq t - t' \leq \Delta\}.$$

For a table entry $T[v_i]$, we check for each layer $t \in L_i$ whether there exists an Δ-restless temporal (s, v_{i-1})-path that arrives in a layer $t' \in T[v_{i-1}]$ such that we can extend the path to the vertex v_i in layer t without exceeding the maximum waiting time Δ, that is, $0 \leq t - t' \leq \Delta$. It is easy to see that $T[v_i]$ contains all layers t such that there exists a Δ-restless temporal (s, v_i)-path with arrival time t. After computing the last entry $T[v_n]$, this entry contains the set $\mathcal{A}$ of all layers t such that there exists a Δ-restless temporal (s, z)-path with arrival time t.

In order to compute a table entries $T[v_i]$ in linear time, we will need sorted lists of layers for L_i and $T[v_{i-1}]$ in ascending order. The sorted lists L_i of layers can be computed in $O(|\mathcal{G}|)$: For every $t \in [\ell]$, we iterate over each $e_i \in E_t$ and add t to L_i. Now assume that L_i and $T[v_{i-1}]$ are lists of layers both in ascending order, then we can compute the table entry $T[v_i]$ in $O(|T[v_{i-1}]| + |L_i|)$ time.

Let $T[v_i]$ be initially empty. Let t be the first element in L_i and let t' be the first element in $T[v_{i-1}]$:

1. If $t' > t$, then replace t with the next layer in L_i and repeat.
2. If $t - t' \leq$, then add t to $T[v_i]$, replace t with the next layer in L_i and repeat.
3. Else, replace t' with the next layer in $T[v_{i-1}]$ and repeat.

This is done until all elements in one of the lists are processed.

The resulting list $T[v_i]$ is again sorted. Due to this and $T[v_1](= L_1)$ being sorted, we can assume that $T[v_{i-1}]$ is given as a sorted list of layers when computing $T[v_i]$. Hence, we can compute each table entry $T[v_i]$ efficiently in $O(|T[v_{i-1}]| + |L_i|)$ time. It further holds that $|T[v_i]| \leq |L_i|$ and $\sum_{i=1}^{n} |L_i| = \sum_{i=1}^{\ell} |E_i|$. Hence, the dynamic program runs in $O(|\mathcal{G}|)$ time. ◀

2.1 Further Basic Observations

Furthermore, it is easy to observe that computational hardness of Restless Temporal Path for some fixed value of Δ implies hardness for all larger finite values of Δ. This allows us to construct hardness reductions for small fixed values of Δ and still obtain general hardness results.

▶ **Observation 3.** *Given an instance $I = (\mathcal{G}, s, z, k, \Delta)$ of Short Restless Temporal Path, we can construct in linear time an instance $I' = (\mathcal{G}', s, z, k, \Delta+1)$ of Short Restless Temporal Path such that I is a* yes*-instance if and only if I' is a* yes*-instance.*

Proof. The result immediately follows from the observation that a temporal graph $\mathcal{G}$ contains a Δ-restless temporal (s, z)-path if and only if the temporal graph $\mathcal{G}'$ contains a $(\Delta + 1)$-restless temporal (s, z)-path, where $\mathcal{G}'$ is obtained from $\mathcal{G}$ by inserting one trivial (edgeless) layer after every Δ consecutive layers. ◀

However, for some special values of Δ we can solve Restless Temporal Path in polynomial time.

▶ **Observation 4.** *Restless Temporal Path on instances $(\mathcal{G}, s, z, \Delta)$ can be solved in polynomial time, if $\Delta = 0$ or $\Delta \geq \ell$.*

Proof. Considering $\Delta = 0$ implies that the entirety of a path between s and z must be realized in a single layer. Thus, the problem is equivalent to testing if at least one of the layers G_i contains a (static) path between s and z.

If $\Delta \geq \ell$, the computation of temporal paths without waiting time constraints was solved for three possible metrics by Bui-Xuan, Ferreira, and Jarry [11]. Any of the three corresponding algorithms apply in this case. ◀

3 Hardness results for restless temporal paths

In this section we present a thorough analysis of the computational hardness of Restless Temporal Path which also transfers to Short Restless Temporal Path.

NP-hardness for few layers. We start by showing that Restless Temporal Path is NP-complete even if the lifetime of the input temporal graph is constant. The reduction is similar in spirit to the classic NP-hardness reduction for 2-Disjoint Paths in directed graphs by Fortune et al. [24].

▶ **Theorem 5 (★).** *Restless Temporal Path is NP-complete for all finite $\Delta \geq 1$ and $\ell \geq \Delta + 2$ even if every edge has only one time stamp.*

The reduction used in the proof of Theorem 5 also yields a running time lower bound assuming the Exponential Time Hypothesis (ETH) [31, 32].

▶ **Corollary 6.** *Restless Temporal Path does not admit a $f(\ell)^{o(|\mathcal{G}|)}$-time algorithm for any computable function f unless the ETH fails.*

Proof. First, note that any 3-SAT formula with m clauses can be transformed into an equisatisfiable Exact $(3, 4)$-SAT formula with $O(m)$ clauses [46]. The reduction presented in the proof of Theorem 5 produces an instance of Restless Temporal Path with a temporal graph of size $|\mathcal{G}| = O(m)$ and $\ell = 3$. Hence an algorithm for Restless Temporal Path with running time $f(\ell)^{o(|\mathcal{G}|)}$ for some computable function f would imply the existence of an $2^{o(m)}$-time algorithm for 3-SAT. This is a contradiction to the ETH [31, 32]. ◀

Furthermore, the reduction behind Theorem 5 can be modified such that it also yields that Restless Temporal Path is NP-hard, even if the underlying graph has constant maximum degree or the underlying graph is a clique where one edge $(\{s, z\})$ is missing.

▶ **Corollary 7.** *Restless Temporal Path is NP-hard, even if the underlying graph has all but one edge or maximum degree six.*

Proof. That Restless Temporal Path is NP-hard, even if the underlying graph has maximum degree six follows directly from the construction used in the proof of Theorem 5. To show that that Restless Temporal Path is NP-hard, even if the underlying graph has all edges except $\{s, z\}$, we reduce from Restless Temporal Path. Let $I = (\mathcal{G} = (V, (E_i)_{i\in[\ell]}), s, z, \Delta)$ be an instance of Restless Temporal Path with $\ell = 3$. We construct an instance $I' := (\mathcal{G}' = (V, E'_1, E'_2, E'_3, E'_4, E'_5), s, z, \Delta)$ of Restless Temporal Path, where $E'_1 = \binom{V\setminus\{s\}}{2}$, $E'_2 := E_1$, $E'_3 := E_2$, $E'_4 := E_3$, and $E'_5 = \binom{V\setminus\{z\}}{2}$. Observe that none of the edges in $E_1 \cup E_5$ can be used in Δ-restless temporal (s, z)-path. Hence, I is a *yes*-instance if and only if I' is a *yes*-instance. Furthermore, $E_1 \cup E_5$ contain all possible edges except $\{s, z\}$. ◀

W[1]-hardness for distance to disjoint paths. In the following, we show that parameterizing Restless Temporal Path with structural graph parameters of the underlying graph of the input temporal graph presumably does not yield fixed-parameter tractability for a large number of popular parameters. In particular, we show that Restless Temporal Path parameterized by the distance to disjoint paths of the underlying graph is W[1]-hard. The *distance to disjoint paths* of a graph G is the minimum number of vertices we have to remove from G such that the reminder of G is a set of disjoint paths. Many well-known graph parameters can be upper-bounded in the distance to disjoint paths, e.g., pathwidth, treewidth, and feedback vertex number [45]. Hence, the following theorem also implies that Restless Temporal Path is W[1]-hard when parameterized by the pathwidth or the feedback vertex number of the underlying graph.

▶ **Theorem 8 (★).** *Restless Temporal Path parameterized by the distance to disjoint path of the underlying graph is W[1]-hard for all $\Delta \geq 1$ even if every edge has only one time stamp.*

4 An FPT-algorithm for short restless temporal path

In this section, we discuss how to find *short* restless temporal paths. Recall that in Short Restless Temporal Path, we are given an additional integer k as input and are asked whether there exists a Δ-restless temporal (s, z)-path that uses at most k time edges. By Theorem 5 this problem is NP-hard. Note that in the contact tracing scenario from the beginning, we can expect to have a small k and a large temporal graph.

▶ **Theorem 9.** *Short Restless Temporal Path is*

- **(i)** *solvable in $2^k \cdot |\mathcal{G}|^{O(1)}$ time with a constant one-side error,*
- **(ii)** *deterministically solvable in $2^{O(k)} \cdot |\mathcal{G}|\Delta$ time,*

Note that we can solve Short Restless Temporal Path such that the running time is independent from the lifetime ℓ of the temporal graph. To show Theorem 9, we first reduce the problem to a specific path problem in directed graphs. Then, we apply known algebraic tools for multilinear monomials detection. Here, Theorem 9 (i) is based on Williams [47]. To get a deterministic algorithm with a running time almost linear in $|\mathcal{G}|$, we show a different approach based on representative sets [22] which results in Theorem 9 (ii).

Reduction to directed graphs. We introduce a so-called Δ-(s, z)-*expansion* for two vertices s and z of a temporal graph with waiting times. That is, a time-expanded version of the temporal graph which reduces reachability questions to directed graphs. While similar approaches have been applied several times [2, 8, 39, 48, 49], to the best of our knowledge,

this is the first time that waiting-times are considered. In a nutshell, the Δ-(s,z)-expansion has for each vertex v at most ℓ many copies $v^1,\dots,v^\ell$ and if an (s,z)-dipath visits v^i, it means that the corresponding Δ-restless temporal (s,z)-walk visits v at time i.

▶ **Definition 10** (Δ-(s,z)-Expansion). *Let $\mathcal{G}=(V,(E_i)_{i\in[\ell]})$ be a temporal graph with two distinct vertices $s,z\in V$ such that $\{s,z\}\notin E_t$, for all $t\in[\ell]$. Let $\Delta\leq\ell$. The* Δ-(s,z)-expansion *of $\mathcal{G}$ is the directed graph $D=(V',E')$ with*

(i) $V':=\{s,z\}\cup\{v^t \mid v\in e, e\in E_t, v\notin\{s,z\}\}$,
(ii) $E_s:=\{(s,v^t)\mid\{s,v\}\in E_t\}$,
(iii) $E_z:=\big\{(v^i,z)\;\big|\;v^i\in V',\{v,z\}\in E_t, 0\leq t-i\leq\Delta\big\}$, *and*
(iv) $E':=E_s\cup E_z\cup\big\{(v^i,w^t)\;\big|\;v^i\in V'\setminus\{s,z\},\{v,w\}\in E_t, 0\leq t-i\leq\Delta\big\}$.

Furthermore, we define $V'(s):=\{s\}$, $V'(z):=\{z\}$, and $V'(v):=\{v^t\in V'\mid t\in[\ell]\}$, for all $v\in V\setminus\{s,z\}$.

Next, we show that a Δ-(s,z)-expansion of a temporal graph can be computed efficiently.

▶ **Lemma 11.** *Given a temporal graph $\mathcal{G}=(V,(E_i)_{i\in[\ell]})$, two distinct vertices $s,z\in V$, and $\Delta\leq\ell$, we can compute its Δ-(s,z)-expansion D with $|V(D)|\in O(|\mathcal{G}|)$ in $O(|\mathcal{G}|\cdot\Delta)$ time.*

Proof. Let $V':=\{s,z\}$ and E' be empty in the beginning. We will fill up V' and E' simultaneously. In order to that efficiently, we will maintain for each vertex $v\in V$ a ordered list L_v such that $t\in L_v$ if and only if $v^t\in V'$. We assume that $|V|\leq\sum_{i=1}^{\ell}|E_i|$, because vertices which are isolated in every layer are irrelevant for the Δ-(s,z)-expansion and can be erased in linear time.

We proceed as follows. For each $t\in\{1,\dots,\ell\}$ (in ascending order), we iterate over E_t. For each $\{v,w\}\in E_t$, we distinguish three cases.

($w=s$): We add v^t to V', (s,v^t) to E', and add t to L_v. This can be done in constant time.

($w=z$): We add v^t to V', and add t to L_v. Now we iterate over all $i\in L_v$ (in descending order) and add (v^i,z) to E' until $t-i>\Delta$. This can be done in $O(\Delta)$ time.

($\{s,z\}\cap\{v,w\}=\emptyset$): We add v^t,w^t to V', and add t to L_v and L_w. Now we iterate over $i\in L_v$ (in descending order) and add (v^i,w^t) to E' until $t-i>\Delta$. Afterwards, we iterate over $i\in L_w$ (in descending order) and add (w^i,v^t) to E' until $t-i>\Delta$. This can be done in $O(\Delta)$ time.

Observe that after this procedure the digraph $D=(V',E')$ is the Δ-(s,z)-expansion of $\mathcal{G}$ and that we added at most 2 vertices for each time-edge in $\mathcal{G}$. Hence, $V'\leq|\mathcal{G}|$. This gives a overall running time of $O(|\mathcal{G}|\cdot\Delta)$. ◀

It is easy to see that there is a Δ-restless temporal (s,z)-walk in the temporal graph if and only if there is an (s,z)-dipath in the Δ-(s,z)-expansion. Next, we identify the necessary side constraint to identify Δ-restless temporal (s,z)-paths in the Δ-(s,z)-expansion.

▶ **Lemma 12.** *Let $\mathcal{G}=(V,(E_i)_{i\in[\ell]})$ be a temporal graph, $s,z\in V$ two distinct vertices, $\Delta\leq\ell$, and $D=(V',E')$ the Δ-(s,z)-expansion of $\mathcal{G}$. There is a Δ-restless temporal (s,z)-path in $\mathcal{G}$ of length k if and only if there is an (s,z)-dipath P' in D of length k such that for all $v\in V$ it holds that $|V'(v)\cap V(P')|\leq 1$.*

Proof. ($\Rightarrow$): Let $P=\big(((s,v_1,t_1),(v_1,v_2,t_2),\dots,(v_{k'-1},z,t_{k'})\big)$ be a Δ-restless temporal (s,z)-path in $\mathcal{G}$ of length k. We can inductively construct an (s,z)-dipath P' in D. Observe that $P'_1:=((s,v_1^{t_1}))$ is an $(s,v_1^{t_1})$-dipath of length 1 in D, because the arc $(s,v_1^{t_1})$ is in E_s of D. Now let $i\in[k'-2]$ and P'_i be an $(s,v_i^{t_i})$-dipath of length i such that

(i) for all $j \in [i]$, we have that $|V'(v_j) \cap V(P'_i)| = 1$, and
(ii) for all $v \in V \setminus \{s, v_1, \ldots, v_i\}$, we have that $|V'(v) \cap V(P'_i)| = 0$.

In order to get an $(s, v_{i+1}^{t_{i+1}})$-dipath P'_{i+1} of length $i+1$, we extend P'_i by the arc $(v_i^{t_i}, v_{i+1}^{t_{i+1}})$. Observe, that $v_{i+1}^{t_{i+1}} \in V'$ because of the time-edge $(\{v_i, v_{i+1}\}, t_{i+1})$ in $\mathcal{G}$ and that the arc $(v_i^{t_i}, v_{i+1}^{t_{i+1}}) \in E'$, because we have $0 \leq t_{i+1} - t_i \leq \Delta$. Observe that

(i) for all $j \in [i+1]$, we have that $|V'(v_j) \cap V(P'_{i+1})| = 1$, and
(ii) for all $v \in V \setminus \{s, v_1, \ldots, v_{i+1}\}$, we have that $|V'(v) \cap V(P'_{i+1})| = 0$.

Hence, we have an $(s, v_{k'-1}^{t_{k'-1}})$-dipath P'_{k-1} of length $k-1$ satisfying (i) and (ii) which can be extended (in a similar way) to an (s, z)-dipath of length k such that for all $v \in V$ it holds that $|V'(v) \cap V(P')| \leq 1$.

($\Leftarrow$): Let P' be a (s, z)-dipath in D of length k such that for all $v \in V$ it holds that $|V'(v) \cap V(P')| \leq 1$. Let $V(P') = \{s, v_1^{t_1}, \ldots, v_{k-1}^{t_{k-1}}, z\}$. Observe that an arc from s to $v_1^{t_1}$ in D implies that there is a time-edge $(\{s, v_1\}, t_1)$ in $\mathcal{G}$. Similarly, an arc from $v_i^{t_i}$ to $v_{i+1}^{t_{i+1}}$ implies that there is a time-edge $(\{v_i, v_{i+1}\}, t_{i+1})$ in $\mathcal{G}$ and that $0 \leq t_{i+1} - t_i \leq \Delta$, for all $i \in [k-2]$. Moreover, an arc from $v_{k-1}^{t_{k-1}}$ to z implies that there is some t_k such that there is a time-edge $(\{v_k, z\}, t_k)$ in $\mathcal{G}$ with $0 \leq t_k - t_{k-1} \leq \Delta$. Hence, $P = \big((s, v_1, t_1), (v_1, v_2, t_2), \ldots, (v_{k'-1}, z, t_{k'})\big)$ is a Δ-restless temporal (s, z)-walk of length k in $\mathcal{G}$. Finally, $|V'(v) \cap V(P')| \leq 1$, for all $v \in V$, implies that $v_i \neq v_j$ for all $i, j \in \{0, \ldots, k\}$ with $i \neq j$. Thus, P is a Δ-restless temporal (s, z)-path of length k. ◀

Obtaining Theorem 9 (i). We now adapt the algorithm of Williams [47] to our specific needs. To this end, we introduce some standard notation from algebraic theory.

An *arithmetic circuit* C over a commutative ring R is a simple labelled directed acyclic graph with its internal nodes labeled by $+$ (sum gates) or $\times$ (product gates) and its nodes of in-degree zero (input gates) labeled with elements from $R \cup X$, where X is a set of variables. There is one node of out-degree zero, called the output gate. The size of C is the number of vertices in the graph. An arithmetic circuit C over R computes a polynomial $P(X)$ over R in the natural way: an input gate represents the polynomial it is labeled by. A sum (product) gate represents the sum (product) of the polynomials represented by its in-degree neighbors. We say C represents $P(X)$ if the polynomial of the output gate of C is equivalent to $P(X)$.

▶ **Lemma 13 (★).** *Let $k \in \mathbb{N}$ and $D = (V, A)$ be a directed graph with partition $V = \biguplus_{i=0}^{n} V_i$, where $V_0 = \{s\}$ and $V_n = \{z\}$. Then, there is an arithmetic circuit C representing a polynomial $Q(X)$ of degree at most $k+1$ such that $Q(X)$ has a multilinear[6] monomial of degree at most $k+1$ if and only if there is an (s, z)-path P of length at most k in D where $|V(P) \cap V_i| \leq 1$ for all $i \in [n]$. Moreover, $|X| = n + 1$, C is of size $O(k(n + |A|))$, has no scalar multiplication, and all product gates in have in-degree two.*

The polynomial in Lemma 13 is recursively defined as $Q(X) = x_\perp Q_z^k$ over variables $X = \{x_\perp, x_0\} \cup \{x_i \mid i \in [n]\}$, where $Q_s^0 := x_0$, $Q_v^0 := x_\perp$ for all for all $v \in V \setminus \{s\}$, and for all $v \in V$ and $j \in [k]$ we define $Q_v^j = \sum_{(u,v) \in A} Q_u^{j-1} x_i$, where $v \in V_i$. The idea of the polynomial is similar to the one of Williams [47], but here instead of having one variable for each vertex we just have one variable for all vertices in one part of the partition of V. Now we can apply the following result of Williams [47].

[6] No variable occurs to a power of two or higher.

▶ **Theorem 14** ([47]). *Let $Q(X)$ be a polynomial of degree at most k, represented by an arithmetic circuit of size n with no scalar multiplications and where all product gates have in-degree two. There is a randomized algorithm that runs in $2^k n^{O(1)}$ time, outputs* yes *with high probability ($\geq 1/5$) if there is a multilinear term in the sum-product expansion of Q, and always outputs* no *if there is no multilinear term.*

Theorem 9 (i) follows from Lemmata 11 to 13 and Theorem 14. This can be derandomized by Theorem 5.2 of Fomin et al. [23] resulting in $O(3.841^k \cdot (|\mathcal{G}|\Delta)^2|V|\log|V|)$ time algorithm. We now show how to improve the polynomial part of a deterministic algorithm.

Obtaining Theorem 9 (ii). To show Theorem 9 (ii), we first note that in the (s,z)-expansion of an (s,z)-path P in the directed graph describes a Δ-restless temporal (s,z)-path exactly when $V(P)$ is an *independent set of some specific matroid*. We then show an algorithm to find such a path P (if there is one). To this end, we introduce a problem, Independent Path, and some standard terminology from matroid theory [43]. A pair $(U,\mathcal{I})$, where U is the *ground set* and $\mathcal{I} \subseteq 2^U$ is a family of *independent sets*, is a *matroid* if the following holds: $\emptyset \in \mathcal{I}$; if $A' \subseteq A$ and $A \in \mathcal{I}$, then $A' \in \mathcal{I}$; and if $A, B \in \mathcal{I}$ and $|A| < |B|$, then there is an $x \in B \setminus A$ such that $A \cup \{x\} \in \mathcal{I}$. An inclusion-wise maximal independent set $A \in \mathcal{I}$ of a matroid $M = (U,I)$ is a *basis*. The cardinality of the bases of M is called the *rank* of M. The *uniform matroid of rank r* on U is the matroid $(U,\mathcal{I})$ with $\mathcal{I} = \{S \subseteq U \mid |S| \leq r\}$. A matroid $(U,\mathcal{I})$ is *linear* or *representable over a field* $\mathbb{F}$ if there is a matrix A with entries in $\mathbb{F}$ and the columns labeled by the elements of U such that $S \in \mathcal{I}$ if and only if the columns of A with labels in S are linearly independent over $\mathbb{F}$. Such a matrix A is called a *representation* of $(U,\mathcal{I})$. Now we are ready to state the Independent Path problem. Given a digraph $D = (V,E)$, two distinct vertices $s, z \in V$, and a representation A_M of a matroid $M = (V,\mathcal{I})$ of rank r over a finite field $\mathbb{F}$, we are asked whether there is an (s,z)-dipath P of length at most k in D such that $V(P) \in \mathcal{I}$.

For the remainder of this section, whenever we speak about *independent sets*, these are independent sets of a matroid and not a set of vertices which induce an edgeless graph.

Agrawal et al. [1] studied, independently from us, a similar problem where the edges of the path shall be an independent set of a matroid. To show Theorem 9, we need a single-exponential algorithm which has only a linear dependency on the input size. To this end, we based on representative families [22] show the following.

▶ **Theorem 15 (★).** *An instance (D, s, z, A_M) of Independent Path can be solved in time of $O(2^{\omega r}m)$ operations over the field $\mathbb{F}$, where $\mathbb{F}$ is the field of A_M, r is rank of M, m is the number of edges in D, and $\omega < 2.373$ is the matrix multiplication exponent.*

Observe that by Lemma 12, there is a Δ-restless temporal (s,z)-path in the temporal graph $\mathcal{G}$ if and only if there is an (s,z)-path P in the Δ-(s,z)-expansion $D = (V',E')$ of $\mathcal{G}$ such that $V(P)$ is an independent set in the *partition matroid*[7] $M = (V', \{X \subseteq V' \mid \forall v \in V\colon |X \cap V'(v)| \leq 1\})$. Note that M is of rank $|V|$ and hence too large to show Theorem 9 with Theorem 15.

A *k-truncation* of a matroid $(U,\mathcal{I})$ is a matroid $(U, \{X \in \mathcal{I} \mid |X| \leq k\})$ such that all independent sets are of size at most k. The k-truncation of a linear matroid is also a linear matroid [38]. In our reduction from Short Restless Temporal Path to Independent Path we use a $(k+1)$-truncation of matroid M. Two general approaches are known to

[7] Partition matroids are linear [38].

compute a representation for a k-truncation of a linear matroid – one is randomized [38] and one is deterministic [37].[8] Both approaches require a large field implying that one operation over that field is rather slow. However, for our specific matroid we employ the Vandermonde matrix to compute a representation over a small finite field. Note that we would not get a running time linear in the input size by applying the algorithm of Lokshtanov et al. [37] or Marx [38] on M.

▶ **Lemma 16 (★).** *Given a universe U of size n, a partition $P_1 \uplus \cdots \uplus P_q = U$, and an integer $k \in \mathbb{N}$, we can compute in $O(kn)$ time a representation A_M for the matroid $M = \left(U, \left\{X \subseteq U \;\middle|\; |X| \leq k \text{ and } \forall i \in [q] \colon |X \cap P_i| \leq 1\right\}\right)$, where A_M is defined over a finite field $\mathbb{F}$ and one operation over $\mathbb{F}$ takes constant time.*

Now Theorem 9 (ii) follows from Lemmata 11, 12 and 16 and Theorem 15. We refer to the full version for more details.

5 Computational complexity landscape for the underlying graph

In this section we investigate the parameterized computational complexity of Restless Temporal Path when parameterized by structural parameters of the underlying graph. We start by showing fixed-parameter tractability results for parameterizations that are directly implied by Theorem 15. We can observe that any path of a graph can contain at most twice as many vertices as the vertex cover number of the graph (plus one), since we cannot visit two vertices outside of the vertex cover directly after another. Essentially the same observation can be made in the temporal setting. If we consider the vertex cover number $\mathrm{vc}_\downarrow$ of the underlying graph, we can deduce that any restless temporal path can have length at most $2\mathrm{vc}_\downarrow + 1$. From a classification standpoint, we can improve this a little further by observing that the length of any restless temporal path is bounded by the length of any path of the underlying graph. The length of a path in the underlying graph can be bounded by $2^{O(\mathrm{td}_\downarrow)}$ [41], where $\mathrm{td}_\downarrow$ is the treedepth of the underlying graph.

▶ **Observation 17.** *Restless Temporal Path parameterized by the treedepth $\mathrm{td}_\downarrow$ of the underlying graph is fixed-parameter tractable.*

One of the few dark spots of the landscape is the *feedback edge number*[9] of the underlying graph which is resolved in the following way.

▶ **Theorem 18 (★).** *Restless Temporal Path can be solved in $2^{O(f)} \cdot |\mathcal{G}|$ time, where f is the feedback edge number of the underlying graph.*

We note that, by Corollary 6, Theorem 18 is asymptotically optimal, unless ETH fails. In a nutshell, our algorithm to prove Theorem 18 has the following five steps:

1. Exhaustively remove all degree-1 vertices from $G_\downarrow$ (except for s and z).
2. Compute a minimum-cardinality feedback edge set F of the graph $G_\downarrow$.
3. Compute a set $\mathcal{P}$ of maximal paths in $G_\downarrow - F$ and note that $|\mathcal{P}| = O(f)$.
4. "Guess" the feedback edges in F and paths in $\mathcal{P}$ of an (s, z)-path in $G_\downarrow$.
5. Verify whether the "guessed" (s, z)-path is a Δ-restless temporal (s, z)-path in $\mathcal{G}$.

[8] For both algorithms, a representation of the original matroid must be given.

[9] For a given graph $G = (V, E)$ a set $F \subseteq E$ is a *feedback edge set* if $G - F$ does not contain a cycle. The *feedback edge number* of a graph G is the size of a minimum feedback edge set for G.

The results from Sections 3 to 5 provide a good picture of the parameterized complexity landscape for Restless Temporal Path, meaning that for most of the widely known (static) graph parameters we know whether the problem is in FPT or W[1]-hard or para-NP-hard, see Figure 2.

Our understanding of the class of temporal graphs where we can solve Restless Temporal Path efficiently narrows down to the following points. We can check efficiently whether there is a Δ-restless temporal (s, z)-path P in temporal graph $\mathcal{G}$ if

1. there is a bounded number of (s, z)-path in $G_{\downarrow}$, or (see Theorem 18 and Lemma 2)
2. there is a bound on the length of P. (see Theorem 9 and Observation 17)

Apart from that we established with Theorems 5 and 8 and Corollary 7 hardness results for temporal graphs having restricted underlying graphs, see Figure 2.

Finally, we show that we presumably cannot expect to obtain polynomial kernels for all parameters considered so far and most structural parameters of the underlying graph.

▶ **Proposition 19 (★).** *Restless Temporal Path parameterized by the number n of vertices does not admit a polynomial kernel for all $\Delta \geq 1$ unless NP $\subseteq$ coNP/poly.*

6 Timed feedback vertex number

In this section we introduce a new temporal version of the well-studied "feedback vertex number"-parameter. Recall that by Theorem 8 we know that Restless Temporal Path is W[1]-hard when parameterized by the feedback vertex number of the underlying graph. This motivates studying larger parameters with the goal to obtain tractability results. We propose a new parameter called *timed feedback vertex number* which, intuitively, quantifies the number of vertex appearances that need to be removed from a temporal graph such that its underlying graph becomes cycle-free. Note that having vertex appearances in the deletion set allows us to "guess" *when* we want to enter and leave the deletion set with a Δ-restless temporal (s, z)-path in addition to guessing in which order the vertex appearances are visited. We remark that there also have been studies of removing edges from temporal graph to destroy *temporal* cycles [26], that is, temporal paths from a vertex back to itself.

Before defining timed feedback vertex number formally, we introduce notation for removing vertex appearances from a temporal graph. Intuitively, when we remove a vertex appearance from a temporal graph, we do not change its vertex set, but remove all time edges that have the removed vertex appearance as an endpoint. Let $\mathcal{G} = (V, (E_i)_{i\in[\ell]})$ be a temporal graph and $X \subseteq V \times [\ell]$ a set of vertex appearances. Then we write $\mathcal{G} - X := (V, (E'_i)_{i\in[\ell]})$, where $E'_i = E_i \setminus \{e \in E_i \mid \exists (v, i) \in X \text{ with } v \in e\}$. Formally, the timed feedback vertex number is defined as follows.

▶ **Definition 20** (Timed Feedback Vertex Number). *Let $\mathcal{G} = (V, (E_i)_{i\in[\ell]})$ be a temporal graph. A* timed feedback vertex set *of $\mathcal{G}$ is a set $X \subseteq V \times [\ell]$ of vertex appearances such that $G_{\downarrow}(\mathcal{G} - X)$ is cycle-free. The* timed feedback vertex number *of a temporal graph $\mathcal{G}$ is the minimum cardinality of a timed feedback vertex set of $\mathcal{G}$.*

We can observe that for any temporal graph the timed feedback vertex number is as least as large as the feedback vertex number of the underlying graph and upper-bounded by the product of the feedback vertex number of the underlying graph and the lifetime. We further remark that the timed feedback vertex number is invariant under reordering the layers. At the end of this section we show how a timed feedback vertex set can be computed efficiently.

Algorithm 1 FPT algorithm for Restless Temporal Path parameterized by timed feedback vertex set.

Input: Temporal graph $\mathcal{G} = (V, (E_i)_{i\in[\ell]})$ with $s, z \in V$, timed feedback vertex set X with $s, z \notin \{v \mid (v,t) \in X\}$, and $\Delta \in \mathbb{N}$.
Output: *yes*, if there is a Δ-restless temporal (s,z)-path, otherwise *no*.

for ***each*** *valid partition* $O \uplus I \uplus U = X$ **do**
  $\mathcal{G}' \leftarrow \mathcal{G} - U$ and $x \leftarrow |I \cup O|$.
  $\mathcal{T} \leftarrow \mathcal{G}' - (\{v \in V \mid (v,t) \in O \cup I\} \cup \{s,z\})$.
  for ***each*** Δ*-ordering* $(v_0,t_0) \leq \cdots \leq (v_{x+1},t_{x+1})$ *of* $I \cup O \cup (\{s,z\} \times \{\perp\})$ **do**
    $\mathcal{P}_i \leftarrow \emptyset$, for all $i \in [x+1]$.
    for $i \leftarrow 1$ **to** $x+1$ **do**
      if $v_{i-1} = v_i$ **then** $\mathcal{P}_i = \{\emptyset\}$.
      for ***each*** $e_1 = (\{v_{i-1}, w\}, t)$, $e_2 = (\{u, v_i\}, t')$ *of* $\mathcal{G}'$ *where* $v_{i-1} \neq v_i$ **do**
        $\mathcal{T}' \leftarrow \mathcal{T} + \{e_1, e_2\}$.
        if $\exists$ (t_{i-1}, t_i)*-valid* Δ*-restless temporal* (v_{i-1}, v_i)*-path* P *in* $\mathcal{T}'$ **then**
          $\mathcal{P}_i \leftarrow \mathcal{P}_i \cup \{V(P) \setminus \{v_{i-1}, v_i\}\}$.
    $G \leftarrow$ intersection graph of the multiset $\{P^{(i)} \in \mathcal{P}_i \mid i \in [x+1]\}$.
    Define $c\colon V(G) \to [x+1], P^{(i)} \mapsto i$.
    if (G,c) *has a multicolored independent set of size* $x+1$ **then return** yes.
return no.

The main result of this section is that Restless Temporal Path is fixed-parameter tractable when parameterized by the timed feedback vertex number of the input temporal graph. To this end, we show the following.

▶ **Theorem 21 (★).** *Given a timed feedback vertex set X of size x for a temporal graph $\mathcal{G} = (V, (E_i)_{i\in[\ell]})$, we can decide in $O(6^x x! \cdot \max\{|\mathcal{G}|^3, |V|^4 x^2\})$ time, whether there is a Δ-restless temporal (s,z)-path in $\mathcal{G}$, where $s, z \in V$, $\Delta \in \mathbb{N}$.*

The algorithm we present to show Theorem 21 solves Chordal Multicolored Independent Set, where given a chordal graph[10] $G = (V, E)$ and a vertex coloring $c\colon V \to [k]$, we are asked to decide whether G contains an independent set of size k that contains exactly one vertex of each color. This problem is known to be NP-complete [9, Lemma 2] and solvable in $O(3^k \cdot |V|^2)$ time [7, Proposition 5.6]. Our algorithm for Restless Temporal Path roughly follows these computation steps:

1. "Guess" which of and in which order the vertex appearances from the timed feedback vertex set appear in the Δ-restless temporal (s,z)-path.
2. Compute the path segments between two timed feedback vertex set vertices by solving a Chordal Multicolored Independent Set instance.

We give a precise description of our algorithm in Algorithm 1. Here, a partition $O \uplus I \uplus U$ of a set of vertex appearances X is *valid* if we have $v \neq v'$, for all distinct $(v,t), (v',t') \in I$ and for all distinct $(v,t), (v',t') \in O$. A vertex appearance $(v,t) \in I$ signals that a Δ-restless temporal (s,z)-path arrives in v at time t and $(v,t) \in O$ signals that it departs from v at time t.

[10] A graph is *chordal* if it does not contain induced cycles of length four or larger.

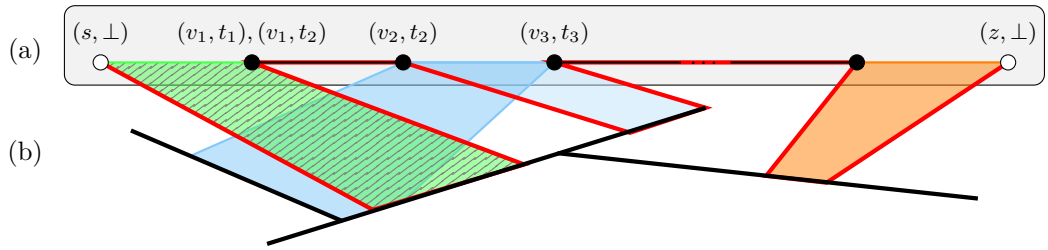

Figure 3 Illustration of Algorithm 1, where (a) depicts the set $(\{s,z\}\times\{\perp\})\cup I\cup O$ and (b) sketches the underlying graph of the temporal graph $\mathcal{T}$ which is a forest. The back solid dots correspond to one or two vertex appearances. The Δ-restless temporal (s,z)-path is the red thick path which uses valid (Definition 22) Δ-restless temporal (s,v_1)- and (v_1,v_2)-paths over $\mathcal{T}$.

Let $M := O\cup I\cup(\{s,z\}\times\{\perp\})$. We call a linear ordering $(v_0,t_0)\leq_M\cdots\leq_M(v_{x+1},t_{x+1})$ of M a *Δ-ordering* if $(v_0,t_0)=(s,\perp)$, $(v_{x+1},t_{x+1})=(t,\perp)$, $t_i\leq t_j$ if and only if $i<j\in[x]$, and for all $v\in V$ with $(v,t_i)\in I$ and $(v,t_j)\in O$ it holds that $i+1=j$ and $t_i\leq t_j\leq t_i+\Delta$. Moreover, observe that for a vertex appearance $(v,t)\in I$, the Δ-restless temporal (s,z)-path has to depart from t not later than $t+\Delta$ and for vertex appearance $(v,t)\in O$, it has to arrive in v not earlier than $t-\Delta$. To this end, we define the notion of a valid path between two consecutive vertex appearances:

▶ **Definition 22.** *Let $O\uplus I\uplus U$ be a valid partition of X, and let $(v_i,t_i),(v_{i+1},t_{i+1})\in I\cup O\cup(\{s,z\}\times\{\perp\})$ with $v_i\neq v_{i+1}$ and $t_i\leq t_{i+1}$, and P_i a Δ-restless temporal (v_i,v_{i+1})-path with departure time t_d and arrival time t_a. Then P_i is* (t_{i-1},t_i,I,O)-valid *if the following holds true*

- **(i)** $(v_{i-1},t_{i-1})\in I\implies t_{i-1}\leq t_d\leq t_{i-1}+\Delta$,
- **(ii)** $(v_{i-1},t_{i-1})\in O\implies t_d=t_{i-1}$,
- **(iii)** $(v_i,t_i)\in I\implies t_a=t_i$, *and*
- **(iv)** $(v_i,t_i)\in O\implies t_a\leq t_i\leq t_a+\Delta$.

If it is clear from context, then we write (t_{i-1},t_i)-valid.

Note that if there exists a (t_i,t_{i+1})-valid Δ-restless temporal (v_i,v_{i+1})-path P_{i+1} and (t_{i+1},t_{i+2})-valid Δ-restless temporal (v_{i+1},v_{i+2})-path P_{i+2}, then we can "glue" them together and get a (t_i,t_{i+2})-valid Δ-restless (v_i,v_{i+2})-walk (not necessarily a path). Thus if there exist a valid Δ-restless temporal path between all consecutive pairs in a Δ-ordering which are pairwise vertex disjoint (except for the endpoints), then there exist aΔ-restless temporal (s,z)-path.

The idea of Algorithm 1 is that a Δ-restless temporal (s,z)-path P induces a valid partition of the timed feedback vertex set X such that $(v,t)\in I$ if P arrives v at time t, $(v,t)\in O$ if P leaves v at time t, or otherwise $(v,t)\in U$. Furthermore, if we order $M:=I\cup O\cup(\{s,z\}\times\{\perp\})$ according to the traversal of P (from s to z), then this is a Δ-ordering such that a subpath P' of P corresponding to consecutive $(v,t),(v',t')\in M$ with $v\neq v'$ is (t,t',I,O)-valid in some temporal graph $\mathcal{T}'$ of Line (9), see Figure 3.

The algorithm, tries all possible partitions of X and all corresponding Δ-orderings. For each of these, we store for all valid Δ-restless temporal (u,w)-path P' of two consecutive $(u,t),(w,t')$ the vertices $V(P)\cap V(\mathcal{T})$ in the family $\mathcal{P}_i$. Here, we assume without loss of generality that no vertex appearance of s,z is in X. Note that, if we have $|\mathcal{P}_i|\geq 0$ for all $i\in\{1,\ldots,x+1\}$, then there is Δ-restless (s,z)-walk in $\mathcal{G}$. Hence, to find a Δ-restless temporal (s,z)-path, we have to find $x+1$ pair-wise disjoint sets $P_1^{(1)},\ldots,P_{x+1}^{(x+1)}$ such that

$P_i \in \mathcal{P}_i$. Here, we observe that the intersection graph of in Line (12) is *chordal* [25] and use an algorithm of Bentert et al. [7] for Chordal Multicolored Independent Set as a subroutine to find such pairwise-disjoint $P_1^{(1)}, \ldots, P_{x+1}^{(x+1)}$.

To conclude from Theorem 21 the fixed-parameter tractability of Restless Temporal Path parameterized the timed feedback vertex number, we need to compute a timed feedback vertex set efficiently. This is clearly NP-hard, since it generalizes the NP-complete Feedback Vertex Set problem [33]. However, we establish the following possibilities to compute a Feedback Vertex Set.

▶ **Theorem 23 (★).** *A minimum timed feedback vertex set of temporal $\mathcal{G}$ can be computed in $4^x \cdot |\mathcal{G}|^{O(1)}$ time, where x is the timed feedback vertex number of $\mathcal{G}$. Furthermore, there is a polynomial-time 8-approximation for timed feedback vertex set.*

7 Conclusion

We have analyzed the (parameterized) computational complexity of Restless Temporal Path, a canonical variant of the problem of finding temporal paths, where the waiting time at every vertex is restricted. Unlike its non-restless counterpart or the "walk-version", this problem turns out to be computationally hard, even in quite restricted cases. On the positive side, we give an efficient algorithm to find short restless temporal paths and we could identify structural parameters of the underlying graph and of the temporal graph itself that allow for fixed-parameter algorithms.

References

1 Akanksha Agrawal, Pallavi Jain, Lawqueen Kanesh, and Saket Saurabh. Parameterized complexity of conflict-free matchings and paths. *Algorithmica*, pages 1–27, 2020. 11

2 Eleni C. Akrida, Jurek Czyzowicz, Leszek Gąsieniec, Łukasz Kuszner, and Paul G. Spirakis. Temporal flows in temporal networks. *Journal of Computer and System Sciences*, 103:46–60, 2019. 3, 8

3 Eleni C Akrida, Leszek Gąsieniec, George B Mertzios, and Paul G Spirakis. The complexity of optimal design of temporally connected graphs. *Theory of Computing Systems*, 61(3):907–944, 2017. 3

4 Eleni C. Akrida, George B. Mertzios, Sotiris Nikoletseas, Christoforos Raptopoulos, Paul G. Spirakis, and Viktor Zamaraev. How fast can we reach a target vertex in stochastic temporal graphs? *Journal of Computer and System Sciences*, 114:65–83, 2020. 2

5 Kyriakos Axiotis and Dimitris Fotakis. On the size and the approximability of minimum temporally connected subgraphs. In *Proceedings of the 43rd International Colloquium on Automata, Languages, and Programming (ICALP '16)*, pages 149:1–149:14, 2016. 2, 3

6 Albert-László Barabási. *Network Science*. Cambridge University Press, 2016. 2

7 Matthias Bentert, René van Bevern, and Rolf Niedermeier. Inductive k-independent graphs and c-colorable subgraphs in scheduling: a review. *Journal of Scheduling*, 22(1):3–20, 2019. 14, 16

8 Kenneth A Berman. Vulnerability of scheduled networks and a generalization of Menger's theorem. *Networks: An International Journal*, 28(3):125–134, 1996. 8

9 René van Bevern, Matthias Mnich, Rolf Niedermeier, and Mathias Weller. Interval scheduling and colorful independent sets. *Journal of Scheduling*, 18(5):449–469, 2015. 14

10 Sandeep Bhadra and Afonso Ferreira. Complexity of connected components in evolving graphs and the computation of multicast trees in dynamic networks. In *International Conference on Ad-Hoc Networks and Wireless*, pages 259–270. Springer, 2003. 2, 3

11 B.-M. Bui-Xuan, Afonso Ferreira, and Aubin Jarry. Computing shortest, fastest, and foremost journeys in dynamic networks. *International Journal of Foundations of Computer Science*, 14(02):267–285, 2003. 2, 7

12 Arnaud Casteigts, Paola Flocchini, Emmanuel Godard, Nicola Santoro, and Masafumi Yamashita. On the expressivity of time-varying graphs. *Theoretical Computer Science*, 590:27–37, 2015. 2

13 Arnaud Casteigts, Paola Flocchini, Walter Quattrociocchi, and Nicola Santoro. Time-varying graphs and dynamic networks. *International Journal of Parallel, Emergent and Distributed Systems*, 27(5):387–408, 2012. 2

14 Arnaud Casteigts, Joseph Peters, and Jason Schoeters. Temporal cliques admit sparse spanners. In *Proceedings of the 46th International Colloquium on Automata, Languages, and Programming (ICALP '19)*, volume 132 of *LIPIcs*, pages 134:1–134:14. Schloss Dagstuhl–Leibniz-Zentrum für Informatik, 2019. 3

15 Marek Cygan, Fedor V. Fomin, Łukasz Kowalik, Daniel Lokshtanov, Dániel Marx, Marcin Pilipczuk, Michał Pilipczuk, and Saket Saurabh. *Parameterized Algorithms*. Springer, 2015. 4, 5

16 Reinhard Diestel. *Graph Theory, 5th Edition*, volume 173 of *Graduate Texts in Mathematics*. Springer, 2016. 4

17 Rodney G Downey and Michael R Fellows. *Fundamentals of Parameterized Complexity*. Springer, 2013. 4

18 Ken TD Eames and Matt J Keeling. Contact tracing and disease control. *Proceedings of the Royal Society of London. Series B: Biological Sciences*, 270(1533):2565–2571, 2003. 2

19 Jessica Enright, Kitty Meeks, George Mertzios, and Viktor Zamaraev. Deleting edges to restrict the size of an epidemic in temporal networks. In *Proceedings of the 44th International Symposium on Mathematical Foundations of Computer Science (MFCS '19)*, volume 138 of *LIPIcs*, pages 57:1–57:15. Schloss Dagstuhl–Leibniz-Zentrum für Informatik, 2019. 3

20 Luca Ferretti, Chris Wymant, Michelle Kendall, Lele Zhao, Anel Nurtay, Lucie Abeler-Dörner, Michael Parker, David Bonsall, and Christophe Fraser. Quantifying SARS-CoV-2 transmission suggests epidemic control with digital contact tracing. *Science*, 2020. 2

21 Till Fluschnik, Hendrik Molter, Rolf Niedermeier, Malte Renken, and Philipp Zschoche. Temporal graph classes: A view through temporal separators. *Theoretical Computer Science*, 806:197–218, 2020. 3

22 Fedor V. Fomin, Daniel Lokshtanov, Fahad Panolan, and Saket Saurabh. Efficient computation of representative families with applications in parameterized and exact algorithms. *Journal of the ACM*, 63(4):29:1–29:60, 2016. 8, 11

23 Fedor V. Fomin, Daniel Lokshtanov, Fahad Panolan, and Saket Saurabh. Representative families of product families. *ACM Transactions on Algorithms*, 13(3):36:1–36:29, 2017. 11

24 Steven Fortune, John E. Hopcroft, and James Wyllie. The directed subgraph homeomorphism problem. *Theoretical Computer Science*, 10:111–121, 1980. 7

25 Fănică Gavril. The intersection graphs of subtrees in trees are exactly the chordal graphs. *Journal of Combinatorial Theory, Series B*, 16(1):47–56, 1974. 16

26 Roman Haag, Hendrik Molter, Rolf Niedermeier, and Malte Renken. Feedback edge sets in temporal graphs. In *Proceedings of the 46th International Workshop on Graph-Theoretic Concepts in Computer Science (WG '20)*, volume 12301 of *Lecture Notes in Computer Science*, pages 200–212. Springer, 2020. 13

27 Anne-Sophie Himmel, Matthias Bentert, André Nichterlein, and Rolf Niedermeier. Efficient computation of optimal temporal walks under waiting-time constraints. In *Proceedings of the 8th International Conference on Complex Networks and their Applications*, volume 882 of *SCI*, pages 494–506. Springer, 2019. 2, 3

28 Petter Holme. Modern temporal network theory: a colloquium. *The European Physical Journal B*, 88(9):234, 2015. 2

29 Petter Holme. Temporal network structures controlling disease spreading. *Physical Review E*, 94.2:022305, 2016. 2

30 Petter Holme and Jari Saramäki (eds.). *Temporal Network Theory*. Springer, 2019. 2

31 Russell Impagliazzo and Ramamohan Paturi. On the complexity of k-SAT. *Journal of Computer and System Sciences*, 62(2):367–375, 2001. 7

32 Russell Impagliazzo, Ramamohan Paturi, and Francis Zane. Which problems have strongly exponential complexity? *Journal of Computer and System Sciences*, 63(4):512–530, 2001. 7

33 Richard M Karp. Reducibility among combinatorial problems. In *Complexity of Computer Computations*, pages 85–103. Springer, 1972. 16

34 David Kempe, Jon Kleinberg, and Amit Kumar. Connectivity and inference problems for temporal networks. *Journal of Computer and System Sciences*, 64(4):820–842, 2002. 2, 3

35 William Ogilvy Kermack and Anderson G McKendrick. A contribution to the mathematical theory of epidemics. *Proceedings of the Royal Society of London, Series A.*, 115(772):700–721, 1927. 2

36 Matthieu Latapy, Tiphaine Viard, and Clémence Magnien. Stream graphs and link streams for the modeling of interactions over time. *Social Network Analysis and Mining*, 8(1):61, 2018. 2

37 Daniel Lokshtanov, Pranabendu Misra, Fahad Panolan, and Saket Saurabh. Deterministic truncation of linear matroids. *ACM Transactions on Algorithms*, 14(2):14:1–14:20, 2018. 12

38 Dániel Marx. A parameterized view on matroid optimization problems. *Theoretical Computer Science*, 410(44):4471–4479, 2009. 11, 12

39 George B Mertzios, Othon Michail, and Paul G Spirakis. Temporal network optimization subject to connectivity constraints. *Algorithmica*, 81(4):1416–1449, 2019. 2, 3, 8

40 Othon Michail. An introduction to temporal graphs: An algorithmic perspective. *Internet Mathematics*, 12(4):239–280, 2016. 2

41 Jaroslav Nešetřil and Patrice Ossona De Mendez. *Sparsity: Graphs, Structures, and Algorithms*. Springer, 2012. 12

42 Mark E J Newman. *Networks*. Oxford University Press, 2018. 2

43 James G. Oxley. *Matroid Theory*. Oxford University Press, 1992. 11

44 Raj Kumar Pan and Jari Saramäki. Path lengths, correlations, and centrality in temporal networks. *Physical Review E*, 84(1):016105, 2011. 2

45 Manuel Sorge and Mathias Weller et al. The graph parameter hierarchy, 2018, 2020. URL: `https://manyu.pro/assets/parameter-hierarchy.pdf`. 4, 8

46 Craig A. Tovey. A simplified NP-complete satisfiability problem. *Discrete Applied Mathematics*, 8(1):85–89, 1984. 7

47 Ryan Williams. Finding paths of length k in $O^*(2^k)$ time. *Information Processing Letters*, 109(6):315–318, 2009. 8, 10, 11

48 H. Wu, J. Cheng, Y. Ke, S. Huang, Y. Huang, and H. Wu. Efficient algorithms for temporal path computation. *IEEE Transactions on Knowledge and Data Engineering*, 28(11):2927–2942, 2016. 2, 8

49 Philipp Zschoche, Till Fluschnik, Hendrik Molter, and Rolf Niedermeier. The complexity of finding separators in temporal graphs. *Journal of Computer and System Sciences*, 107:72–92, 2020. 3, 8