Temporal graph realization from fastest paths

Anonymous author
Anonymous affiliation

Anonymous author
Anonymous affiliation

Anonymous author
Anonymous affiliation

Anonymous author
Anonymous affiliation

Abstract

In this paper we initiate the study of the *temporal graph realization* problem with respect to the fastest path durations among its vertices, while we focus on periodic temporal graphs. Given an $n \times n$ matrix D and a $\Delta \in \mathbb{N}$, the goal is to construct a Δ-periodic temporal graph with n vertices such that the duration of a *fastest path* from v_i to v_j is equal to $D_{i,j}$, or to decide that such a temporal graph does not exist. The variations of the problem on static graphs has been well studied and understood since the 1960's (e.g. [Erdős and Gallai, 1960], [Hakimi and Yau, 1965]).

As it turns out, the periodic temporal graph realization problem has a very different computational complexity behavior than its static (i. e., non-temporal) counterpart. First we show that the problem is NP-hard in general, but polynomial-time solvable if the so-called underlying graph is a tree. Building upon those results, we investigate its parameterized computational complexity with respect to structural parameters of the underlying static graph which measure the “tree-likeness”. We prove a tight classification between such parameters that allow fixed-parameter tractability (FPT) and those which imply W[1]-hardness. We show that our problem is W[1]-hard when parameterized by the *feedback vertex number* (and therefore also any smaller parameter such as *treewidth*, *degeneracy*, and *cliquewidth*) of the underlying graph, while we show that it is in FPT when parameterized by the *feedback edge number* (and therefore also any larger parameter such as *maximum leaf number*) of the underlying graph.

Due to lack of space, the full paper with all proofs is attached in a clearly marked Appendix to be read at the discretion of the Program Committee.

2012 ACM Subject Classification Theory of computation → Graph algorithms analysis; Mathematics of computing → Discrete mathematics

Keywords and phrases Temporal graph, periodic temporal labeling, fastest temporal path, graph realization, temporal connectivity.

Digital Object Identifier 10.4230/LIPIcs.CVIT.2016.23

42nd Conference on Very Important Topics (CVIT 2016).
Editors: John Q. Open and Joan R. Access; Article No. 23; pp. 23:1–23:20
Leibniz International Proceedings in Informatics
Schloss Dagstuhl – Leibniz-Zentrum für Informatik, Dagstuhl Publishing, Germany

1 Introduction

The (static) *graph realization* problem with respect to a graph property $\mathcal{P}$ is to find a graph that satisfies property $\mathcal{P}$, or to decide that no such graph exists. The motivation for graph realization problems stems both from "verification" and from network design applications in engineering. In *verification* applications, given the outcomes of some experimental measurements (resp. some computations) on a network, the aim is to (re)construct an input network which complies with them. If such a reconstruction is not possible, this proves that the measurements are incorrect or implausible (resp. that the algorithm which made the computations is incorrectly implemented). One example of a graph realization (or reconstruction) problem is the recognition of probe interval graphs, in the context of the physical mapping of DNA, see [49, 50] and [35, Chapter 4]. In *network design* applications, the goal is to design network topologies having a desired property [4, 37]. Analyzing the computational complexity of the graph realization problems for various natural and fundamental graph properties $\mathcal{P}$ requires a deep understanding of these properties. Among the most studied such parameters for graph realization are constraints on the distances between vertices [7,8,10,16,17,40], on the vertex degrees [6,22,34,36,39], on the eccentricities [5,9,41,48], and on connectivity [15,28–30,33,36], among others.

In the simplest version of a (static) graph realization problem with respect to vertex distances, we are given a symmetric $n \times n$ matrix D and we are looking for an n-vertex undirected and unweighted graph G such that $D_{i,j}$ equals the distance between vertices v_i and v_j in G. This problem can be trivially solved in polynomial time in two steps [40]: First, we build the graph $G = (V, E)$ such that $v_i v_j \in E$ if and only if $D_{i,j} = 1$. Second, from this graph G we compute the matrix D_G which captures the shortest distances for all pairs of vertices. If $D_G = D$ then G is the desired graph, otherwise there is no graph having D as its distance matrix. Non-trivial variations of this problem have been extensively studied, such as for weighted graphs [40,56], as well as for cases where the realizing graph has to belong to a specific graph family [7,40]. Other variations of the problem include the cases where every entry of the input matrix D may contain a range of consecutive permissible values [7,57,59], or even an arbitrary set of acceptable values [8] for the distance between the corresponding two vertices.

In this paper we make the first attempt to understand the complexity of the graph realization problem with respect to vertex distances in the context of *temporal graphs*, i. e., of graphs whose *topology changes over time*.

▶ **Definition 1** (temporal graph [42]). *A* temporal graph *is a pair (G, λ), where $G = (V, E)$ is an underlying (static) graph and $\lambda : E \to 2^{\mathbb{N}}$ is a* time-labeling *function which assigns to every edge of G a set of discrete time-labels.*

Here, whenever $t \in \lambda(e)$, we say that the edge e is *active* or *available* at time t. In the context of temporal graphs, where the notion of vertex adjacency is time-dependent, the notions of path and distance also need to be redefined. The most natural temporal analogue of a path is that of a *temporal* (or *time-dependent*) path, which is motivated by the fact that, due to causality, entities and information in temporal graphs can "flow" only along sequences of edges whose time-labels are strictly increasing.

▶ **Definition 2** (fastest temporal path). *Let (G, λ) be a temporal graph. A* temporal path *in (G, λ) is a sequence $(e_1, t_1), (e_2, t_2), \ldots, (e_k, t_k)$, where $P = (e_1, \ldots, e_k)$ is a path in the underlying static graph G, $t_i \in \lambda(e_i)$ for every $i = 1, \ldots, k$, and $t_1 < t_2 < \ldots < t_k$. The* duration *of this temporal path is $t_k - t_1 + 1$. A* fastest *temporal path from a vertex u to a* *vertex v in (G, λ) is a temporal path from u to v with the smallest duration. The* duration *of the fastest temporal path from u to v is denoted by $d(u, v)$.*

In this paper we consider *periodic* temporal graphs, i. e., temporal graphs in which the temporal availability of each edge of the underlying graph is periodic. Many natural and technological systems exhibit a periodic temporal behavior. For example, in railway networks an edge is present at a time step t if and only if a train is scheduled to run on the respective rail segment at time t [3]. Similarly, a satellite, which makes pre-determined periodic movements, can establish a communication link (i. e., a temporal edge) with another satellite whenever they are sufficiently close to each other; the existence of these communication links is also periodic. In a railway (resp. satellite) network, a fastest temporal path from u to v represents the fastest railway connection between two stations (resp. the quickest communication delay between two moving satellites). Furthermore, periodicity appears also in (the otherwise quite complex) social networks which describe the dynamics of people meeting [47, 58], as every person individually follows mostly a daily routine [3].

Although periodic temporal graphs have already been studied (see [13, Class 8] and [3, 24, 54, 55]), we make here the first attempt to understand the complexity of a graph realization problem in the context of temporal graphs. Therefore, we focus in this paper on the most fundamental case, where all edges have the same period Δ (while in the more general case, each edge e in the underlying graph has a period Δ_e). As it turns out, the periodic temporal graph realization problem with respect to a given $n \times n$ matrix D of the fastest duration times has a very different computational complexity behavior than the classic graph realization problem with respect to shortest path distances in static graphs.

Formally, let $G = (V, E)$ and $\Delta \in \mathbb{N}$, and let $\lambda : E \to \{1, 2, \ldots, \Delta\}$ be an edge-labeling function that assigns to every edge of G exactly one of the labels from $\{1, \ldots, \Delta\}$. Then we denote by (G, λ, Δ) the Δ-*periodic temporal graph* (G, L), where for every edge $e \in E$ we have $L(e) = \{i\Delta + x : i \geq 0, x \in \lambda(e)\}$. In this case we call λ a Δ-*periodic labeling* of G; see Figure 1 for an illustration. When it is clear from the context, we drop Δ form the notation and we denote the (Δ-periodic) temporal graph by (G, λ). Given a duration matrix D, it is easy to observe that, similarly to the static case, if $D_{i,j} = 1$ then v_i and v_j must be connected by an edge. We call the graph defined by these edges the *underlying graph* of D.

Our contribution. We initiate the study of naturally motivated graph realization problems in the temporal setting. Our target is not to model unreliable communication, but instead to *verify* that particular measurements regarding fastest temporal paths in a periodic temporal graph are plausible (i. e., "realizable"). To this end, we introduce and investigate the following problem, capturing the setting described above:

Simple periodic Temporal Graph Realization (Simple TGR)

Input: An $n \times n$ matrix D, a positive integer Δ.
Question: Does there exist a graph $G = (V, E)$ with vertices $\{v_1, \ldots, v_n\}$ and a Δ-periodic labeling $\lambda : E \to \{1, 2, \ldots, \Delta\}$ such that, for every i, j, the duration of the fastest temporal path from v_i to v_j in the Δ-periodic temporal graph (G, λ, Δ) is $D_{i,j}$?

We focus on exact algorithms. We start by showing NP-hardness of the problem (Theorem 3), even if Δ is a small constant. To establish a baseline for tractability, we show that Simple TGR is polynomial-time solvable if the underlying graph is a tree (Theorem 5).

Building upon these initial results, we explore the possibilities to generalize our polynomial-time algorithm using the *distance-from-triviality* parameterization paradigm [26, 38]. That is,

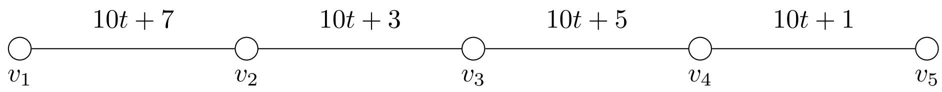


Figure 1 An example of a Δ-periodic temporal graph (G, λ, Δ), where $\Delta = 10$ and the 10-periodic labeling $\lambda : E \to \{1, 2, \ldots, 10\}$ is as follows: $\lambda(v_1v_2) = 7$, $\lambda(v_2v_3) = 3$, $\lambda(v_3v_4) = 5$, and $\lambda(v_4v_5) = 1$. Here, the fastest temporal path from u to v traverses the first edge v_1v_2 at time 7, second edge v_2v_3 a time 13, third edge v_3v_4 at time 15 and the last edge v_4v_5 at time 21. This results in the total duration of 15 for the fastest temporal path from v_1 to v_5.

we investigate the parameterized computational complexity of Simple TGR with respect to structural parameters of the underlying graph that measure its "tree-likeness".

We obtain the following results. We show that Simple TGR is W[1]-hard when parameterized by the *feedback vertex number* of the underlying graph (Theorem 4). To this end, we first give a reduction from Multicolored Clique parameterized by the number of colors [25] to a variant of Simple TGR where the period Δ is infinite, that is, when the labeling is non-periodic. We use a special gadget (the "infinity" gadget) which allows us to transfer the result to a finite period Δ. The latter construction is independent from the particular reduction we use, and can hence be treated as a reduction from the non-periodic to the periodic setting. Note that our parameterized hardness result with respect to the feedback vertex number also implies W[1]-hardness for any smaller parameter, such as *treewidth*, *degeneracy*, *cliquewidth*, *distance to chordal graphs*, and *distance to outerplanar graphs*.

We complement this hardness result by showing that Simple TGR is fixed-parameter tractable (FPT) with respect to the *feedback edge number* k of the underlying graph (Theorem 6). This result also implies an FPT algorithm for any larger parameter, such as the *maximum leaf number*. A similar phenomenon of getting W[1]-hardness with respect to the feedback vertex number, while getting an FPT algorithm with respect to the feedback edge number, has been observed only in a few other temporal graph problems related to the connectivity between two vertices [14, 21, 31].

Our FPT algorithm works as follows on a high level. First we distinguish $O(k^2)$ vertices which we call "important vertices". Then, we guess the fastest temporal paths for each pair of these important vertices; as we prove, the number of choices we have for all these guesses is upper bounded by a function of k. Then we also need to make several further guesses (again using a bounded number of choices), which altogether leads us to specify a small (i. e., bounded by a function of k) number of different configurations for the fastest paths between *all pairs* of vertices. For each of these configurations, we must then make sure that the labels of our solution will not allow any other temporal path from a vertex v_i to a vertex v_j have a *strictly smaller* duration than $D_{i,j}$. This naturally leads us to build one Integer Linear Program (ILP) for each of these configurations. We manage to formulate all these ILPs by having a number of variables that is upper-bounded by a function of k. Finally we use Lenstra's Theorem [46] to solve each of these ILPs in FPT time. At the end, our initial instance is a Yes-instance if and only if at least one of these ILPs is feasible.

The above results provide a fairly complete picture of the parameterized computational complexity of Simple TGR with respect to structural parameters of the underlying graph which measure "tree-likeness". To obtain our results, we prove several properties of fastest temporal paths, which may be of independent interest. Due to space constraints, proofs of results marked with $\star$ are (partially) deferred to the Appendix.

Related work. Graph realization problems on static graphs have been studied since the 1960s. We provide an overview of the literature in the introduction. To the best of our knowledge, we are the first to consider graph realization problems in the temporal setting. However, many other connectivity-related problems have been studied in the temporal setting [2, 12, 18, 19, 23, 27, 32, 43, 52, 53, 61], most of which are much more complex and computationally harder than their non-temporal counterparts, and some of which do not even have a non-temporal counterpart.

There are some problem settings that share similarities with ours, which we discuss now in more detail.

Several problems have been studied where the goal is to assign labels to (sets of) edges of a given static graph in order to achieve certain connectivity-related properties [1, 20, 44, 51]. The main difference to our problem setting is that in the mentioned works, the input is a graph and the sought labeling is not periodic. Furthermore, the investigated properties are temporal connectivity among all vertices [1, 44, 51], temporal connectivity among a subset of vertices [44], or reducing reachability among the vertices [20]. In all these cases, the duration of the temporal paths has not been considered.

Finally, there are many models for dynamic networks in the context of distributed computing [45]. These models have some similarity to temporal graphs, in the sense that in both cases the edges appear and disappear over time. However, there are notable differences. For example, one important assumption in the distributed setting can be that the edge changes are adversarial or random (while obeying some constraints such as connectivity), and therefore they are not necessarily known in advance [45].

Preliminaries and notation. We already introduced the most central notion and concepts. There are some additional definitions we need, to present our proofs and results which we give in the following.

An interval in $\mathbb{N}$ from a to b is denoted by $[a, b] = \{i \in \mathbb{N} : a \leq i \leq b\}$; similarly, $[a] = [1, a]$. An undirected graph $G = (V, E)$ consists of a set V of vertices and a set $E \subseteq V \times V$ of edges. For a graph G, we also denote by $V(G)$ and $E(G)$ the vertex and edge set of G, respectively. We denote an edge $e \in E$ between vertices $u, v \in V$ as a set $e = \{u, v\}$. For the sake of simplicity of the representation, an edge e is sometimes also denoted by uv. A path P in G is a subgraph of G with vertex set $V(P) = \{v_1, \dots, v_k\}$ and edge set $E(P) = \{\{v_i, v_{i+1}\} : 1 \leq i < k\}$ (we often represent path P by the tuple $(v_1, v_2, \dots, v_k)$).

Let $v_1, v_2, \dots, v_n$ be the n vertices of the graph G. For simplicity of the presentation (and with a slight abuse of notation) we refer during the paper to the entry $D_{i,j}$ of the matrix D as $D_{a,b}$, where $a = v_i$ and $b = v_j$. That is, we put as indices of the matrix D the corresponding vertices of G whenever it is clear from the context.

Let $P = (u = v_1, v_2, \dots, v_p = v)$ be a path from u to v in G. Recall that, in our paper, every edge has exactly one time label in every period of Δ consecutive time steps. Therefore, as we are only interested in the fastest duration of temporal paths, many times we refer to (P, λ, Δ) as any of the temporal paths from $u = v_1$ to $v = v_p$ along the edges of P, which starts at the edge v_1v_2 at time $\lambda(v_1v_2) + c\Delta$, for some $c \in \mathbb{N}$, and then sequentially visits the rest of the edges of P as early as possible. We denote by $d(P, \lambda, \Delta)$, or simply by $d(P, \lambda)$ when Δ is clear from the context, the duration of any of the temporal paths (P, λ, Δ); note that they all have the same duration. Many times we also refer to a path $P = (u = v_1, v_2, \dots, v_p = v)$ from u to v in G, as a temporal path in (G, λ, Δ), where we actually mean that (P, λ, Δ) is a temporal path with P as its underlying (static) path.

We remark that a fastest path between two vertices in a temporal graph can be computed in polynomial time [11, 60]. Hence, given a Δ-periodic temporal graph (G, λ, Δ), we can compute in polynomial-time the matrix D which consists of durations of fastest temporal paths among all pairs of vertices in (G, λ, Δ).

2 Hardness results for Simple TGR

In this section we present our main computational hardness results. We first show that Simple TGR is NP-hard even for constant Δ.

▶ **Theorem 3** ($\star$). *Simple TGR is NP-hard for all $\Delta \geq 3$.*

Next, we investigate the parameterized hardness of Simple TGR with respect to structural parameters of the underlying graph. We show that the problem is W[1]-hard when parameterized by the feedback vertex number of the underlying graph. The *feedback vertex number* of a graph G is the cardinality of a minimum vertex set $X \subseteq V(G)$ such that $G - X$ is a forest. The set X is called a *feedback vertex set*. Note that, in contrast to the previous result (Theorem 3), the reduction we use to obtain the following result does not produce instances with a constant Δ.

▶ **Theorem 4** ($\star$). *Simple TGR is W[1]-hard when parameterized by the feedback vertex number of the underlying graph.*

Proof. We present a parameterized reduction from the W[1]-hard problem Multicolored Clique parameterized by the number of colors [25]. Here, given a k-partite graph $H = (W_1 \uplus W_2 \uplus \ldots \uplus W_k, F)$, we are asked whether H contains a clique of size k. If $w \in W_i$, then we say that w has *color* i. W.l.o.g. we assume that $|W_1| = |W_2| = \ldots = |W_k| = n$ and that every vertex has at least one neighbor of every color. Furthermore, for all $i \in [k]$, we assume the vertices in W_i are ordered in some arbitrary but fixed way, that is, $W_i = \{w_1^i, w_2^i, \ldots, w_n^i\}$. Let $F_{i,j}$ with $i < j$ denote the set of all edges between vertices from W_i and W_j. We assume w.l.o.g. that $|F_{i,j}| = m$ for all $i < j$ (if not we can add $k \max_{i,j} |F_{i,j}|$ vertices to each W_i and use those to add up to $\max_{i,j} |F_{i,j}|$ additional isolated edges to each $F_{i,j}$). Furthermore, for all $i < j$ we assume that the edges in $F_{i,j}$ are ordered in some arbitrary but fixed way, that is, $F_{i,j} = \{e_1^{i,j}, e_2^{i,j}, \ldots, e_m^{i,j}\}$.

We give a reduction to a variant of Simple TGR where the period Δ is infinite (that is, the sought temporal graph is not periodic) and we allow D to have infinity entries, meaning that the two respective vertices are not temporally connected. Note that, given the matrix D, we can easily compute the underlying graph G, as follows. Two vertices v, v' are adjacent if G if and only if $D_{v,v'} = 1$, as having an edge between v and v' is the only way that there exists a temporal path from v to v' with duration 1. For simplicity of the presentation of the reduction, we describe the underlying graph G (which directly implies the entries of D where $D(v, v') = 1$) and then we provide the remaining entries of D. At the end of the proof we show how to obtain the result for a finite Δ and a matrix D of durations of fastest paths, that only has finite entries.

In the following, we give an informal description of the main ideas of the reduction. The construction uses several gadgets, where the main ones are an “edge selection gadget” and a “verification gadget”.

Every *edge selection gadget* is associated with a color combination i, j in the Multicolored Clique instance, and its main purpose is to “select” an edge connecting a vertex from color i with a vertex from color j. Roughly speaking, the edge selection gadget consists of m paths, one for every edge in $F_{i,j}$ (see Figure 2 for reference). The distance matrix

D will enforce that the labels on those paths effectively order them temporally, that is, in particular, the labels on one of the paths will be smaller than the labels on all other paths. The edge corresponding to this path is selected.

We have a *verification gadget* for every color i. They interact with the edge selection gadgets as follows. The verification gadget for color i is connected to all edge selection gadgets that involve color i. More specifically, this is connected to every path corresponding to an edge at a position in the path that encodes the endpoint of color i of that edge (again, see Figure 2) for reference. Intuitively, the distances in the verification gadget are only realizable if the selected edges all have the same endpoint of color i. Hence, the distances of all verification gadgets can be realized if and only if the selected edges form a clique.

Furthermore, we use an *alignment gadget* which, intuitively, ensures that the labelings of all gadgets use the same range of time labels. Finally, we use *connector gadgets* which create shortcuts between all vertex pairs that are irrelevant for the functionality of the other gadgets. This allows us to easily fill in the distance matrix with the corresponding values. We ensure that all our gadgets have a constant feedback vertex number, hence the overall feedback vertex number is quadratic in the number of colors of the MULTICOLORED CLIQUE instance and we get the parameterized hardness result.

In the following, for every gadget, we give a formal description of the underlying graph of this gadget (i.e., not the complete distance sub-matrix of the gadget). Due to space constraints, we defer the description of the distance matrix D and the formal proof of correctness for the reduction to the Appendix.

Given an instance H of MULTICOLORED CLIQUE, we construct an instance D of SIMPLE TGR (with infinity entries and no periods) as follows.

Edge selection gadget. We first introduce an *edge selection gadget $G_{i,j}$ for color combination i, j* with $i < j$. We start with describing the vertex set of the gadget.

- A set $X_{i,j}$ of vertices $x_1, x_2, \ldots, x_m$.
- Vertex sets $U_1, U_2, \ldots, U_m$ with $4n+1$ vertices each, that is, $U_\ell = \{u_0^\ell, u_1^\ell, u_2^\ell, \ldots, u_{4n}^\ell\}$ for all $\ell \in [m]$.
- Two special vertices $v_{i,j}^\star, v_{i,j}^{\star\star}$.

The gadget has the following edges.

- For all $\ell \in [m]$ we have edge $\{x_\ell, v_{i,j}^\star\}$, $\{v_{i,j}^\star, u_0^\ell\}$, and $\{u_{4n}^\ell, v_{i,j}^{\star\star}\}$.
- For all $\ell \in [m]$ and $\ell' \in [4n]$, we have edge $\{u_{\ell'-1}^\ell, u_{\ell'}^\ell\}$.

Verification gadget. For each color i, we introduce the following vertices. What we describe in the following will be used as a *verification gadget for color i*.

- We have one vertex y^i and $k+1$ vertices v_ℓ^i for $0 \le \ell \le k$.
- For every $\ell \in [m]$ and every $j \in [k] \setminus \{i\}$ we have $5n$ vertices $a_1^{i,j,\ell}, a_2^{i,j,\ell}, \ldots, a_{5n}^{i,j,\ell}$ and $5n$ vertices $b_1^{i,j,\ell}, b_2^{i,j,\ell}, \ldots, b_{5n}^{i,j,\ell}$.
- We have a set $\hat{U}_i$ of $13n+1$ vertices $\hat{u}_1^i, \hat{u}_2^i, \ldots, \hat{u}_{13n+1}^i$.

We add the following edges. We add edge $\{y^i, v_0^i\}$. For every $\ell \in [m]$, every $j \in [k] \setminus \{i\}$, and every $\ell' \in [5n-1]$ we add edge $\{a_{\ell'}^{i,j,\ell}, a_{\ell'+1}^{i,j,\ell}\}$ and we add edge $\{b_{\ell'}^{i,j,\ell}, b_{\ell'+1}^{i,j,\ell}\}$.

Let $1 \le j < i$ (skip if $i = 1$), let $e_\ell^{j,i} \in F_{j,i}$, and let $w_{\ell'}^i \in W_i$ be incident with $e_\ell^{j,i}$. Then we add edge $\{v_{j-1}^i, a_1^{i,j,\ell}\}$ and we add edge $\{a_{5n}^{i,j,\ell}, u_{\ell'-1}^\ell\}$ between $a_{5n}^{i,j,\ell}$ and the vertex $u_{\ell'-1}^\ell$ of the edge selection gadget of color combination j, i. Furthermore, we add edge $\{v_j^i, b_1^{i,j,\ell}\}$ and edge $\{b_{5n}^{i,j,\ell}, u_{\ell'}^\ell\}$ between $b_{5n}^{i,j,\ell}$ and the vertex $u_{\ell'}^\ell$ of the edge selection gadget of color combination j, i.

We add edge $\{v_{i-1}^i, \hat{u}_1^i\}$ and for all $\ell'' \in [13n]$ we add edge $\{\hat{u}_{\ell''}^i, \hat{u}_{\ell''+1}^i\}$. Furthermore, we add edge $\{\hat{u}_{13n+1}^i, v_i^i\}$.

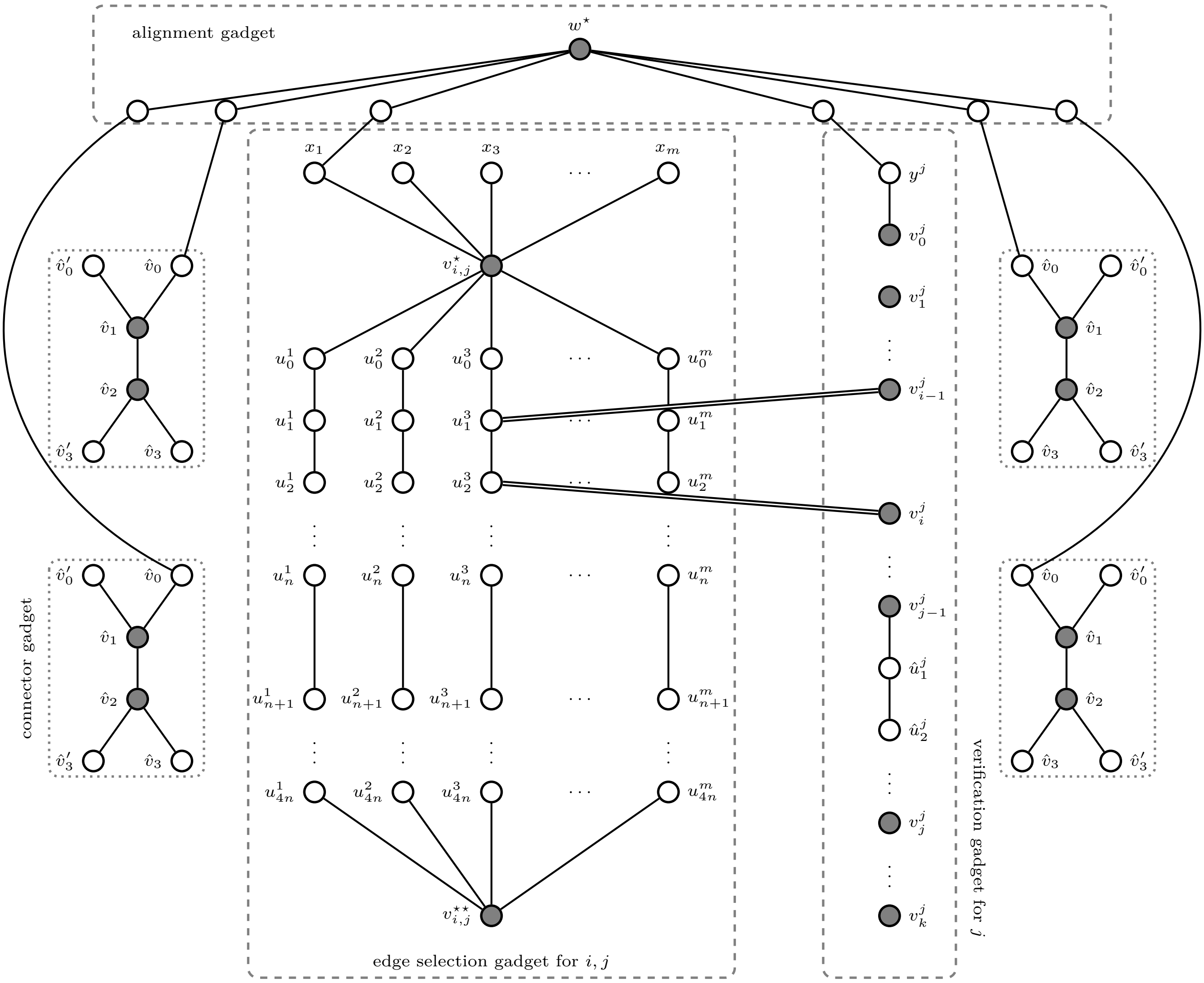


Figure 2 Illustration of part of the underlying graph G. Edges incident with vertices $\hat{v}_1, \hat{v}_2$ of connector gadgets are omitted. Gray vertices form a feedback vertex set. The double line connections, between a vertex v^j_{i-1} in the verification gadget, and u^3_1 in the edge selection gadget, and, between a vertex u^3_2 in the edge selection gadget, and v^j_i in the verification gadget, consist of $5n$ vertices $a^{j,i,3}_1, a^{j,i,3}_2, \ldots, a^{j,i,3}_{5n}$ and $b^{j,i,3}_1, b^{j,i,3}_2, \ldots, b^{j,i,3}_{5n}$, respectively.

Let $i < j \le k$ (skip if $i = k$), let $e^{i,j}_\ell \in F_{i,j}$, and let $w^i_{\ell'} \in W_i$ be incident with $e^{i,j}_\ell$. Then we add edge $\{v^i_{j-1}, a^{i,j,\ell}_1\}$ and edge $\{a^{i,j,\ell}_{5n}, u^\ell_{3n+\ell'-1}\}$ between $a^{i,j,\ell}_{5n}$ and the vertex $u^\ell_{3n+\ell'-1}$ of the edge selection gadget of color combination i, j. Furthermore, we add edge $\{v^i_j, b^{i,j,\ell}_1\}$ and edge $\{b^{i,j,\ell}_{5n}, u^\ell_{3n+\ell'}\}$ between $b^{i,j,\ell}_{5n}$ and the vertex $u^\ell_{3n+\ell'}$ of the edge selection gadget of color combination i, j.

Furthermore, we use *connector gadgets*, two for each edge selection gadget, and two for every verification gadget. They consist of six vertices $\hat{v}_0, \hat{v}'_0, \hat{v}_1, \hat{v}_2, \hat{v}_3, \hat{v}'_3$ and, intuitively, are used to connect many vertex pairs by fast paths, which will make arguing about possible labelings in Yes-instances much easier. Finally, we have an *alignment gadget*, which is a star with a center vertex $w^\star$ and a leaf for every other gadget. Intuitively, this gadget is used to

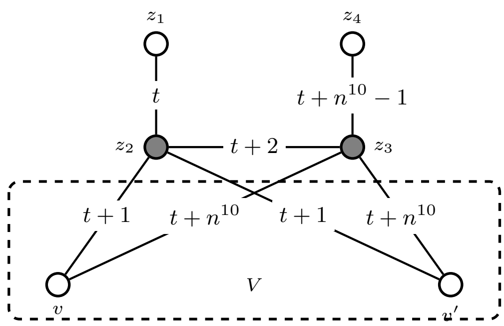


Figure 3 Illustration of the infinity gadget. Gray vertices are added to the feedback vertex set.

relate labels of different gadgets to each other. A formal description of these two gadgets is given in the Appendix.

This finishes the description of the underlying graph G. For an illustration see Figure 2. We can observe that the vertex set containing vertices $v^{\star}_{i,j}$ and $v^{\star\star}_{i,j}$ of each edge selection gadget, vertices v^i_ℓ with $0 \leq \ell \leq k$ of each verification gadget, vertices $\hat{v}_1$ and $\hat{v}_2$ of each connector gadget, and vertex $w^\star$ of the alignment gadget forms a feedback vertex set in G with size $O(k^2)$.

As mentioned before, due to space constraints, we defer the description of the distance matrix D and a formal correctness proof of the reduction to the Appendix.

Infinity gadget. Finally, we show how to get rid of the infinity entries in D and how to allow a finite Δ. To this end, we introduce the *infinity gadget*. We add four vertices z_1, z_2, z_3, z_4 to the graph and we set $\Delta = n^{11}$. Let V denote the set of all remaining vertices. We set the following durations.

- For all $v \in V$ we set $d(z_1, v) = 2$, $d(z_2, v) = d(v, z_2) = 1$, $d(z_3, v) = d(v, z_3) = 1$, and $d(z_4, v) = 2$. Furthermore, we set $d(v, z_1) = n^{11}$ and $d(v, z_4) = n^{10} - 1$.
- We set $d(z_1, z_2) = d(z_2, z_1) = 1$, $d(z_2, z_3) = d(z_3, z_2) = 1$, and $d(z_3, z_4) = d(z_4, z_3) = 1$.
- We set $d(z_1, z_3) = 3$, $d(z_3, z_1) = n^{11} - 1$, $d(z_2, z_4) = n^{10} - 2$, and $d(z_4, z_2) = n^{11} - n^{10} + 4$.
- We set $d(z_1, z_4) = n^{10}$ and $d(z_4, z_1) = 2n^{11} - n^{10} + 2$.
- For every pair of vertices $v, v' \in V$ where previously the duration of a fastest path from v to v' was specified to be infinite, we set $d(v, v') = n^{10}$.

Now we analyse which implications we get for the labels on the newly introduced edges. Assume $\lambda(\{z_1, z_2\}) = t$, then we get the following. For all $v \in V$ we have that $d(z_1, v) = 2$ and hence we get that $\lambda(\{z_2, v\}) = t+1$. Since $d(z_1, z_4) = n^{10}$, we have that $\lambda(z_3, z_4) = t+n^{10}-1$. From this follows that for all $v \in V$, since $d(z_4, v) = 2$, that $\lambda(\{z_3, v\}) = t + n^{10}$. Finally, since $d(z_1, z_3) = 3$, we have that $\lambda(\{z_2, z_3\}) = t+2$. For an illustration see Figure 3. It is easy to check that all duration requirements between vertex pairs in $\{z_1, z_2, z_3, z_4\}$ are met and that all duration requirements between each vertex $v \in V$ and each vertex in $\{z_1, z_2, z_3, z_4\}$ are met. Furthermore, it is easy to check that the gadget increases the feedback vertex set by two (z_2 and z_3 need to be added).

Lastly, consider two vertices $v, v' \in V$. Note that before the addition of the infinity gadget, by construction of G we have that $d(v, v') \leq n^9 + 2$ or $d(v, v') = \infty$. Furthermore, if D is a YES-instance, we have shown in the correctness proof of the reduction that the difference between the smallest label and the largest label is at most $n^9 + 1$. This implies that for a vertex pair $v, v' \in V$ with $d(v, v') = \infty$ we have in the periodic case with $\Delta = n^{11}$, that $d(v, v') \geq n^{11} - n^9 > n^{10}$. Which means, after adding the vertices and edges of the infinity gadget, we indeed have that $d(v, v') = n^{10}$. For all vertex pairs v, v' where in the original construction we have $d(v, v') \neq \infty$, we can also see that adding the infinity gadget and setting $\Delta = n^{11}$ does not change the duration of a fastest path from v to v', since all newly added temporal paths have duration at least n^{10}. We can conclude that the originally constructed instance D is a YES-instance if and only if it remains a YES-instance after adding the infinity gadget and setting $\Delta = n^{11}$. ◀

3 Algorithms for Simple TGR

In this section we provide several algorithms for SIMPLE TGR. By Theorem 3 we have that SIMPLE TGR is NP-hard in general, hence we start by identifying restricted cases where we can solve the problem in polynomial time. We first show in Section 3.1 that if the underlying graph G of an instance (D, Δ) of SIMPLE TGR is a tree, then we can determine desired Δ-periodic labeling λ of G in polynomial time. In Section 3.2 we generalize this result. We show that SIMPLE TGR is fixed-parameter tractable when parameterized by the feedback edge number of the underlying graph. Note that our parameterized hardness result (Theorem 4) implies that we presumably cannot replace the feedback edge number with the smaller parameter feedback vertex number, or any other parameter that is smaller than the feedback vertex number, such as e.g. the treewidth.

3.1 Polynomial-time algorithm for trees

We now provide a polynomial-time algorithm for SIMPLE TGR when the underlying graph is a tree. Let D be the input matrix and let the underlying graph G of D be a tree on n vertices $\{v_1, v_2, \ldots, v_n\}$. Let v_i, v_j be two arbitrary vertices in G, then we know that there exists a unique (static) path $P_{i,j}$ from v_i to v_j. We will heavily exploit this in our algorithm.

▶ **Theorem 5.** *SIMPLE TGR can be solved in polynomial time on trees.*

Proof. Let D be an input matrix for problem SIMPLE TGR of dimension $n \times n$. Let us fix the vertices of the corresponding graph G of D as $v_1, v_2, \ldots, v_n$, where vertex v_i corresponds to the row and column i of matrix D. This can be done in polynomial time as we need to loop through the matrix D once and connect vertices v_i, v_j for which $D_{i,j} = 1$. At the same time we also check if $D_{i,i} = 0$, for all $i \in [n]$. When G is constructed we run DFS algorithm on it and check that it has no cycles. If at any step we encounter a problem, our algorithm stops and returns a negative answer.

Having computed G, our algorithm proceeds as follows. We pick an arbitrary edge f and give it label one, that is, $\lambda(f) = 1$. Now we push all edges incident with f into a (initially empty) queue. Now we repeat the following as long as the queue is not empty:

- Pop edge $e = \{u, v\}$ from the queue. Since e was pushed into the queue, there is an edge e' incident with e that already obtained a label. Let w.l.o.g. $e' = \{v, w\}$. Then we set $\lambda(e) = (\lambda(e') - D_{u,w} + 1) \bmod \Delta$.
- Push all edges incident with e that have not received a label yet into the queue.

When the queue is empty, all edges have received a label. Iterate over all vertex pairs u, v and check whether the fastest path from u to v in (G, λ) has duration $D_{u,v}$. If this check succeeds for all vertex pairs, output the labelling λ, otherwise abort.

It is easy to see that the described algorithm runs in polynomial time. In the remainder, we proof that it is correct.

($\Rightarrow$): Since the algorithm checks at the end whether all durations specified in D are realized by the corresponding fastest paths, we clearly face a yes-instance whenever the algorithm outputs a labeling.

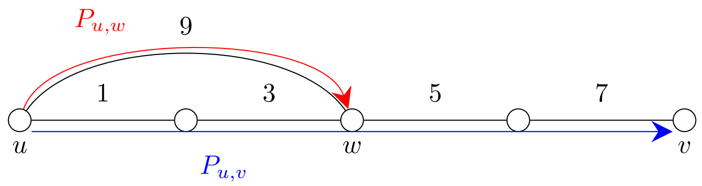


Figure 4 An example of a temporal graph (with $\Delta \geq 9$), where the fastest temporal path $P_{u,v}$ (in blue) from u to v is of duration 7, while the fastest temporal path $P_{u,w}$ (in red) from u to a vertex w, that is on a path $P_{u,v}$, is of duration 1 and is not a subpath of $P_{u,v}$.

($\Leftarrow$): Assume we face a yes-instance, then there exists a labeling $\lambda^\star$ that realizes all durations specified in D. Let $e^\star$ denote the edge initially picked by the algorithm. For all edges e let $\lambda(e) = (\lambda^\star(e) - \lambda^\star(e^\star) + 1) \bmod \Delta$. Clearly, the labeling λ also realizes all durations specified in D since λ is obtained by adding the constant $(1 - \lambda^\star(e^\star))$ modulo Δ to all labels of $\lambda^\star$ which does not change the duration of any temporal path, that is all durations in $(G, \lambda^\star)$ are the same as their counterparts in (G, λ). We claim that our algorithm computes and outputs λ.

We prove that our algorithm computes λ by induction on the distance of the labeled edges to $e^\star$, where the distance of two edges e, e' is defined as the length of a shortest path that uses e as its first edge and e' as its last edge.

Initially, our algorithm labels $e^\star$ with one, which equals $\lambda(e^\star)$. Now let e be an edge popped off the queue by the algorithm in some iteration. Let e' be the edge incident with e that already obtained a label and is considered by the algorithm. Since G is a tree, we have that e' is closer to $e^\star$ than e. By induction we have that the algorithm labeled e' with $\lambda(e')$. Assume that $e = \{u, v\}$ and $e' = \{v, w\}$. Since G is a tree there is only one path from u to w in G and it uses edges e and e'. It follows that $\lambda(e') - \lambda(e) + 1 = D_{u,w}$ if $\lambda(e') > \lambda(e)$, and $\lambda(e') - \lambda(e) + \Delta + 1 = D_{u,w}$ otherwise. Our algorithm labels e with $(\lambda(e') - D_{u,w} + 1) \bmod \Delta$. It is straightforward to verify that the label of e computed by the algorithm equals $\lambda(e)$. It follows that the algorithm computes λ. ◀

3.2 FPT-algorithm for feedback edge number

Recall from Section 3.1 that the main reason, for which Simple TGR is straightforward to solve on trees, is twofold:

- between any pair of vertices v_i and v_j in the tree T, there is a *unique* path P in T from v_i to v_j, and
- in any periodic temporal graph (T, λ, Δ) and any fastest temporal path $P = ((e_1, t_1), \ldots, (e_i, t_i), \ldots, (e_j, t_j), \ldots, (e_\ell, t_\ell))$ from v_1 to v_ℓ we have that the sub-path $P' = ((e_i, t_i), \ldots, (e_j, t_j))$ is also a fastest temporal path from v_i to v_j.

However, these two nice properties do not hold when the underlying graph is not a tree. For example, in Figure 4, the fastest temporal path from u to v is $P_{u,v}$ (depicted in blue) goes through w, however the sub-path of $P_{u,v}$ that stops at w is not the fastest temporal path from u to w. The fastest temporal path from u to w consists only of the single edge uw (with label 9 and duration 1, depicted in red).

Nevertheless, we prove in this section that we can still solve Simple TGR efficiently if the underlying graph is similar to a tree; more specifically we show the following result, which turns out to be non-trivial.

▶ **Theorem 6** ($\star$). *Simple TGR is in FPT when parameterized by the feedback edge number of the underlying graph.*

From Theorem 4 and Theorem 6 we immediately get the following, which is the main result of the paper.

▶ **Corollary 7.** *Simple TGR is:*

- *in FPT when parameterized by the* feedback edge number *or any larger parameter, such as the* maximum leaf number.
- *W[1]-hard when parameterized by the* feedback vertex number *or any smaller parameter, such as:* treewidth*,* degeneracy*,* cliquewidth*,* distance to chordal graphs*, and* distance to outerplanar graphs.

Before presenting the structure of our algorithm for Theorem 6, observe that, in a static graph, the number of paths between two vertices can be upper-bounded by a function $f(k)$ of the feedback edge number k of the graph. Therefore, for any fixed pair of vertices u and v, we can "guess" the edges of the fastest temporal path from u to v. However, for an FPT algorithm with respect to k, we cannot afford to guess the edges of the fastest temporal path for each of the $O(n^2)$ pairs of vertices. To overcome this difficulty, our algorithm follows this high-level strategy:

- We identify a small number $f(k)$ of "important vertices"; these consist of the sets that we call U, U^*, Z^*.
- For each pair u, v of important vertices, we guess the edges of the fastest temporal path from u to v (and from v to u).
- From these guesses we can still not deduce the edges of the fastest temporal paths between many pairs of non-important vertices. However, as we prove, it suffices to guess only a small number of specific auxiliary structures (to be defined later).
- From these guesses we deduce fixed relationships between the labels of most of the edges of the graph.
- For all the edges, for which we do not have deduced a label yet, we introduce a *variable*. Using all these variables, we build an Integer Linear Program (ILP). Among the constraints in this ILP we have that, for each of the $O(n^2)$ pairs of vertices u, v in the graph, the duration of one specific temporal path from u to v (according to our guesses) is *equal* to the desired duration $D_{u,v}$, while the duration of each of the other temporal paths from u to v is *at least* $D_{u,v}$.
- By making any of the above guesses, we restrict the solution space for the problem Simple TGR. This restricted solution space coincides with the set of feasible solutions to the resulting ILP. Furthermore, the set of feasible solutions for all constructed ILPs coincide with the set of all solutions to Simple TGR (i. e., regardless of our guesses). As each ILP can be solved in FPT time with respect to k by Lenstra's Theorem [46] (the number of variables is upper bounded by a function of k), we obtain our FPT algorithm for Simple TGR with respect to k.

We now present the first part of our FPT algorithm, that is, identifying important vertices and guessing information about the fastest temporal paths. A full description of the algorithm is deferred to the Appendix.

Important vertices. Let D be the input matrix of Simple TGR and let G be its underlying graph, on n vertices and m edges. From the underlying graph G of D we first create a graph G' by iteratively removing vertices of degree one from G, and denote with $Z = V(G) \setminus V(G')$,

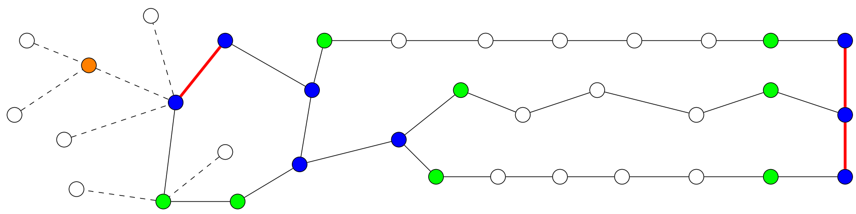


Figure 5 An example of a graph with its important vertices: U (in blue), U^* (in green) and Z^* (in orange). Corresponding feedback edges are marked with a thick red line, while dashed edges represent the edges (and vertices) "removed" from G' at the initial step.

the set of removed vertices. Then we determine the set U (the *"vertices of interest"*), and the set U^* (the neighbors of the vertices of interest), as follows. Let T be a spanning tree of G', with F being the corresponding feedback edge set of G'. Let $V_1 \subseteq V(G')$ be the set of leaves in the spanning tree T, $V_2 \subseteq V(G')$ be the set of vertices of degree two in T which are incident to at least one edge in F, and let $V_3 \subseteq V(G')$ be the set of vertices of degree at least 3 in T. Then $|V_1| + |V_2| \leq 2k$, since every leaf in T and every vertex in V_2 is incident to at least one edge in F, and $|V_3| \leq |V_1|$ by the properties of trees. We denote with

$$U = V_1 \cup V_2 \cup V_3$$

the set of *vertices of interest*. It follows that $|U| \leq 4k$. We set U^* to be the set of vertices in $V(G') \setminus U$ that are neighbors of vertices in U, i. e.,

$$U^* = \{v \in V(G') \setminus U : u \in U, v \in N(u)\}.$$

Again, using the tree structure, we get that for any $u \in U$ its neighborhood is of size $|N(u)| \in O(k)$, since every neighbor of u is the first vertex of a (unique) path to another vertex in U. It follows that $|U^*| \in O(k^2)$. From the construction of Z (by iteratively removing vertices of degree one from G) it follows that Z consists of disjoint trees $T_1, T_2, \ldots$. For a tree T_i we denote with u_i the vertex in G' that is a neighbor of a vertex in T_i, and call it a *clip vertex of the tree* T_i. It follows that there can be many different trees T_i that are incident to the same clip vertex $u_i \in V(G')$, but each tree T_i is incident to exactly one clip vertex $u_i \in V(G')$. Since u_i is the only vertex connecting all of the trees T_i incident to it, from now on we assume that a tree T_{u_i} in Z is a union of trees on vertices from $V(G) \setminus V(G')$, that are clipped at the same vertex $u_i \in V(G')$. For each of the trees T_{u_i} in Z, we select one vertex r_i, that is a neighbor of the clip vertex u_i, and call it *a representative vertex of the tree* T_{u_i}. We now define as Z^* the set of representatives r_i of trees $T_i \in Z$, where the clip vertex v_i of T_i is a vertex of interest, i. e.,

$$Z^* = \{r_i : r_i \in T_i, \text{ where } T_i \in Z, \text{ the clip vertex } u_i \text{ of } T_i \text{ is in } U, \text{ and } r_i u_i \in E(G)\}.$$

Since there are $O(k)$ vertices of interest, we get that $|Z^*| \in O(k)$. Finally, the set of *important vertices* is defined as the set $U \cup U^* \cup Z^*$. For an illustration see Figure 5.

Guesses. For every pair of important vertices $u, v \in U \cup U^* \cup Z^*$, we guess the sequence of edges in the fastest temporal path from u to v. Since $U \cup U^* \cup Z^* \in O(k^2)$ and there are $k^{O(k)}$ possibilities for a sequence of edges between a fixed vertex pair, we have $k^{O(k^5)}$ overall possible guesses. We defer further details to the Appendix (see guesses **G-1** to **G-6**).

With the information provided by the described guesses we are still not able to determine all fastest paths. For example consider the case depicted in Figure 6. Therefore we introduce

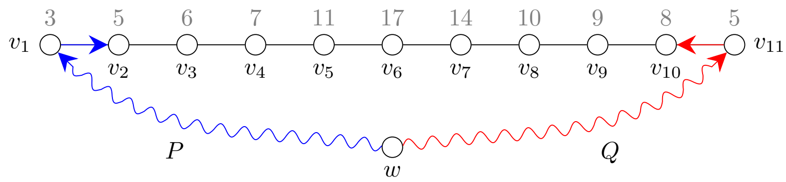


Figure 6 In the above graph vertices v_1, v_{11}, w are in U, while v_2, v_{10} are in U^*. Numbers above all v_i represent the values of the fastest temporal paths from w to each of them (i. e., the entries in the w-th row of matrix D). From the basic guesses we know the fastest temporal path P from w to v_2 (depicted in blue) and the fastest temporal path Q from w to v_{10}. From the values of durations from w to each v_i we cannot determine the fastest paths from w to all v_i. More precisely, we know that w reaches v_2, v_3, v_4, v_5 (resp. v_{10}, v_9, v_9, v_7) by first using the path P (resp. Q) and then proceeding through the vertices, but we do not know how w reaches v_6 the fastest. Therefore we have to introduce some more guesses.

additional guesses that provide us with sufficient information to determine all fastest paths. To do this we have to first define the following.

▶ **Definition 8.** *Let $U \subseteq V(G')$ be a set of vertices of interest and let $u, v \in U$. A path $P = (u = v_1, v_2, \dots, v_p = v)$ in graph G', where all inner vertices are not in U, i. e., $v_i \notin U$ for all $i \in \{2, 3, \dots, p-1\}$, is called a* segment *from u to v. We denote it as $S_{u,v}$.*

Note by Definition 8 that $S_{u,v} \neq S_{v,u}$. Observe that a temporal path in G' between two vertices of interest is either a segment, or it consists of a sequence of some segments. Furthermore, since we have at most $4k$ interesting vertices in G', we can deduce the following important result.

▶ **Corollary 9.** *There are at most $O(k^2)$ segments in G'.*

To describe the next guesses, we introduce the following notation. Let u, v, x be three vertices in G'. We write $u \rightsquigarrow x \to v$ to denote a temporal path from u to v that passes through x, and then goes directly to v (via one edge). We guess the following structures.

G-7. Inner segment guess I. Let $S_{u,v} = (u = v_1, v_2, \dots, v_p = v)$ and $S_{w,z} = (w = z_1, z_2, \dots, z_r = z)$ be two segments. We want to guess the fastest temporal path $v_2 \to u \rightsquigarrow w \to z_2$. We repeat this procedure for all pairs of segments. Since there are $O(k^2)$ segments in G', there are $k^{O(k^5)}$ possible paths of this form.
Recall that $S_{u,v} \neq S_{v,u}$ for every $u, v \in U$. Furthermore note that we did not assume that $\{u, v\} \cap \{w, z\} = \emptyset$. Therefore, by repeatedly making the above guesses, we also guess the following fastest temporal paths: $v_2 \to u \rightsquigarrow z \to z_{r-1}$, $v_2 \to u \rightsquigarrow v \to v_{p-1}$, $v_{p-1} \to v \rightsquigarrow w \to z_2$, $v_{p-1} \to v \rightsquigarrow z \to z_{r-1}$, and $v_{p-1} \to v \rightsquigarrow u \to v_2$. For an example see Figure 7a.

G-8. Inner segment guess II. Let $S_{u,v} = (u = v_1, v_2, \dots, v_p = v)$ be a line segment in G', and let $w \in U \cup Z^*$. We want to guess the following fastest temporal paths $w \rightsquigarrow u \to v_2$, $w \rightsquigarrow v \to v_{p-1} \to \dots \to v_2$, and $v_2 \to u \rightsquigarrow w$, $v_2 \to v_3 \to \cdots v \rightsquigarrow w$.
For fixed $S_{u,v}$ and $w \in U \cup Z^*$ we have $k^{O(k)}$ different possible such paths, therefore we make $k^{O(k^4)}$ guesses for these paths. For an example see Figure 7b.

G-9. Split vertex guess I. Let $S_{u,v} = (u = v_1, v_2, \dots, v_p = v)$ be a line segment in G', and let us fix a vertex $v_i \in S_{u,v} \setminus \{u, v\}$. In the case when $S_{u,v}$ is of length 4, the fixed vertex v_i is the middle vertex, else we fix an arbitrary vertex $v_i \in S_{u,v} \setminus \{u, v\}$. Let $S_{w,z} = (w = z_1, z_2, \dots, z_r = z)$ be another segment in G'. We want to determine the fastest paths from v_i to all inner vertices of $S_{w,z}$. We do this by inspecting the values in matrix D from v_i to inner vertices of $S_{w,z}$. We split the analysis into two cases.

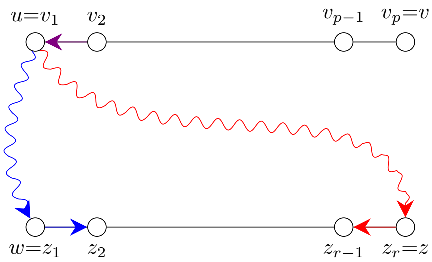
a. There is a single vertex $z_j \in S_{w,z}$ for which the duration from v_i is the biggest. More specifically, $z_j \in S_{w,z} \setminus \{w, z\}$ is the vertex with the biggest value D_{v_i,z_j}. We call this vertex a *split vertex of v_i in the segment S_{wz}*. Then it holds that $D_{v_i,z_2} < D_{v_i,z_3} < \cdots < D_{v_i,z_j}$ and $D_{v_i,z_{r-1}} < D_{v_i,z_{r-2}} < \cdots < D_{v_i,z_j}$. From this it follows that the fastest temporal paths from v_i to $z_2, z_3, \ldots, z_{j-1}$ go through w, and the fastest temporal paths from v_i to $z_{r-1}, z_{r-2}, \ldots, z_{j+1}$ go through z. We now want to guess which vertex w or z is on a fastest temporal path from v_i to z_j. Similarly, all fastest temporal paths starting at v_i have to go either through u or through v, which also gives us two extra guesses for the fastest temporal path from v_i to z_j. Therefore, all together we have 4 possibilities on how the fastest temporal path from v_i to z_j starts and ends. Besides that we want to guess also how the fastest temporal paths from v_i to z_{j-1}, z_{j+1} start and end. Note that one of these is the subpath of the fastest temporal path from v_i to z_j, and the ending part is uniquely determined for both of them, i. e., to reach z_{j-1} the fastest temporal path travels through w, and to reach z_{j+1} the fastest temporal path travels through z. Therefore we have to determine only how the path starts, namely if it travels through u or v. This introduces two extra guesses. For a fixed $S_{u,v}, v_i$ and $S_{w,z}$ we find the vertex z_j in polynomial time, or determine that z_j does not exist. We then make four guesses where we determine how the fastest temporal path from v_i to z_j passes through vertices u, v and w, z and for each of them two extra guesses to determine the fastest temporal path from v_i to z_{j-1} and from v_i to z_{j+1}. We repeat this procedure for all pairs of segments, which results in producing $k^{O(k^5)}$ new guesses. Note, $v_i \in S_{u,v}$ is fixed when calculating the split vertex for all other segments $S_{w,z}$.

b. There are two vertices $z_j, z_{j+1} \in S_{w,z}$ for which the duration from v_i is the biggest. More specifically, $z_j, z_{j+1} \in S_{w,z} \setminus \{w, z\}$ are the vertices with the biggest value $D_{v_i,z_j} = D_{v_i,z_{j+1}}$. Then it holds that $D_{v_i,z_2} < D_{v_i,z_3} < \cdots < D_{v_i,z_j} = D_{v_i,z_{j+1}} > D_{v_i,z_{j+2}} > \cdots > D_{v_i,z_{r-1}}$. From this it follows that the fastest temporal paths from v_i to $z_2, z_3, \ldots, z_j$ go through w, and the fastest temporal paths from v_i to $z_{r-1}, z_{r-2}, \ldots, z_{j+1}$ go through z. In this case we only need to guess the following two fastest temporal paths $u \rightsquigarrow w \to z_2$ and $u \rightsquigarrow z \to z_{r-1}$. Each of this paths we then uniquely extend along the segment $S_{w,z}$ up to the vertex v_j, resp. v_{j+1}, which give us fastest temporal paths from u to v_j and from u to v_{j+1}. In this case we do not introduce any new guesses, as we have aready guessed the fastest paths of the form $u \rightsquigarrow w \to z_2$ and $u \rightsquigarrow z \to z_{r-1}$ (see guess **G-8**).

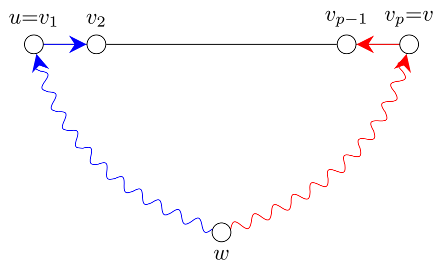
Note that this case results also in knowing the fastest paths from the vertex $v_i \in S_{u,v}$ to $w, z \in S_{w,z}$ for all segments $S_{w,z}$, i. e., we know the fastest paths from a fixed $v_i \in S_{u,v}$ to all vertices of interest in U. For an example see Figure 7c.

G-10. Split vertex guess II. Let $w \in U \cup Z^*$ and let $S_{u,v} = (u = v_1, v_2, \ldots, v_p = v)$. We want to guess a split vertex of w in $S_{u,v}$, and the fastest temporal path that reaches it. We again have two cases, first one where v_i is a unique vertex in $S_{u,v}$ that is furthest away from w, and the second one where v_i, v_{i+1} are two incident vertices in $S_{u,v}$, that are furthest away from w. All together we make two guesses for each pair of vertex $w \in U$ and segment $S_{u,v}$. We repeat this for all vertices of interest, and all segments, which produces $k^{O(k^2)}$ new guesses. For an example see Figure 7d. Detailed analysis follows arguing from above (as in **G-9**) and is deferred to Appendix.

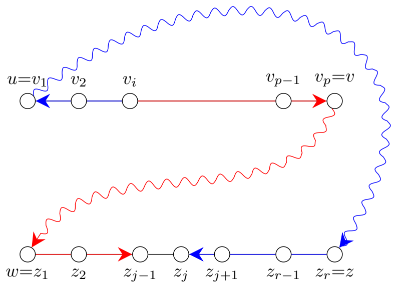
There are two more guesses **G-11** and **G-12** that are deferred to the Appendix. We prove



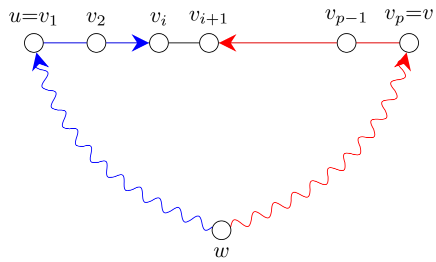
(a) Example of an Inner segment guess I (**G-7**), where we guessed the fastest temporal paths of the form $v_2 \to u \rightsquigarrow w \to z_2$ (in blue) and $v_2 \to u \rightsquigarrow z \to z_{r-1}$ (in red).



(b) Example of an Inner segment guess II (**G-8**), where we guessed the fastest temporal paths of the form $w \rightsquigarrow u \to v_2$ (in blue) and $w \rightsquigarrow v \to v_{p-1}$ (in red).



(c) Example of a Split vertex guess I (**G-9**), where, for a fixed vertex $v_i \in S_{u,v}$, we calculated its corresponding split vertex $z_j \in S_{w,z}$, and guessed the fastest paths of the form $v_i \to v_{i-1} \to \cdots \to u \rightsquigarrow z \to z_{r-1} \cdots \to z_j$ (in blue) and $v_i \to v_{i+1} \to \cdots \to v \rightsquigarrow w \to z_2 \to \cdots \to z_{j-1}$ (in red).



(d) Example of a Split vertex guess II (**G-10**), where, for a vertex of interest w, we calculated its corresponding split vertex $v_i \in S_{u,v}$, and guessed the fastest paths of the form $w \rightsquigarrow u \to v_2 \to \cdots \to v_i$ (in blue) and $w \rightsquigarrow v \to v_{p-1} \to \cdots \to v_{i+1}$ (in red).

Figure 7 Illustration of the guesses **G-7**, **G-8**, **G-9**, and **G-10**.

in the Appendix that, for all guesses **G-1** to **G-12**, there are in total at most $f(k)$ possible choices, and for each one of them we create an ILP with at most $f(k)$ variables and at most $f(k) \cdot |D|^{O(1)}$ constraints. Each of these ILPs can be solved in FPT time by Lenstra's Theorem [46]. For detailed explanation and proofs of this part see Appendix.

4 Conclusion

We believe that our work spawns several interesting future research directions and builds a base upon which further temporal graph realization problems can be investigated.

There are several structural parameters which can be considered to obtain tractability which are either larger or incomparable to the feedback vertex number. We believe that the *vertex cover number* or the *tree depth* are promising candidates. Furthermore, we can consider combining a structural parameter such as the *treewidth* with Δ.

There are many natural variants of our problem that are well-motivated and warrant consideration. We believe that one of the most natural generalizations of our problem is to allow more than one label per edge in every Δ-period. A well-motivated variant (especially from the network design perspective) of our problem would be to consider the entries of the duration matrix D as upper-bounds on the duration of fastest paths rather than exact durations. Our work gives a starting point for many interesting future research directions such as the two mentioned examples.

References

1. 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*, 61:907–944, 2017.
2. Eleni C. Akrida, George B. Mertzios, Paul G. Spirakis, and Christoforos Raptopoulos. The temporal explorer who returns to the base. *Journal of Computer and System Sciences*, 120:179–193, 2021.
3. Emmanuel Arrighi, Niels Grüttemeier, Nils Morawietz, Frank Sommer, and Petra Wolf. Multi-parameter analysis of finding minors and subgraphs in edge-periodic temporal graphs. In *Proceedings of the 48th International Conference on Current Trends in Theory and Practice of Computer Science (SOFSEM)*, pages 283–297, 2023.
4. John Augustine, Keerti Choudhary, Avi Cohen, David Peleg, Sumathi Sivasubramaniam, and Suman Sourav. Distributed graph realizations. *IEEE Transactions on Parallel and Distributed Systems*, 33(6):1321–1337, 2022.
5. Amotz Bar-Noy, Keerti Choudhary, David Peleg, and Dror Rawitz. Efficiently realizing interval sequences. *SIAM Journal on Discrete Mathematics*, 34(4):2318–2337, 2020.
6. Amotz Bar-Noy, Keerti Choudhary, David Peleg, and Dror Rawitz. Graph realizations: Maximum degree in vertex neighborhoods. In *Proceedings of the 17th Scandinavian Symposium and Workshops on Algorithm Theory (SWAT)*, pages 10:1–10:17, 2020.
7. Amotz Bar-Noy, David Peleg, Mor Perry, and Dror Rawitz. Composed degree-distance realizations of graphs. In *Proceedings of the 32nd International Workshop on Combinatorial Algorithms (IWOCA)*, pages 63–77, 2021.
8. Amotz Bar-Noy, David Peleg, Mor Perry, and Dror Rawitz. Graph realization of distance sets. In *Proceedings of the 47th International Symposium on Mathematical Foundations of Computer Science (MFCS)*, pages 13:1–13:14, 2022.
9. Mehdi Behzad and James E Simpson. Eccentric sequences and eccentric sets in graphs. *Discrete Mathematics*, 16(3):187–193, 1976.
10. Robert E Bixby and Donald K Wagner. An almost linear-time algorithm for graph realization. *Mathematics of Operations Research*, 13(1):99–123, 1988.
11. Binh-Minh 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.
12. Arnaud Casteigts, Timothée Corsini, and Writika Sarkar. Invited paper: Simple, strict, proper, happy: A study of reachability in temporal graphs. In *Proceedings of the 24th International Symposium on Stabilization, Safety, and Security of Distributed Systems (SSS)*, pages 3–18, 2022.
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.
14. Arnaud Casteigts, Anne-Sophie Himmel, Hendrik Molter, and Philipp Zschoche. Finding temporal paths under waiting time constraints. *Algorithmica*, 83(9):2754–2802, 2021.
15. Wai-Kai Chen. On the realization of a (p, s)-digraph with prescribed degrees. *Journal of the Franklin Institute*, 281(5):406–422, 1966.
16. Fan Chung, Mark Garrett, Ronald Graham, and David Shallcross. Distance realization problems with applications to internet tomography. *Journal of Computer and System Sciences*, 63(3):432–448, 2001.
17. Joseph C. Culberson and Piotr Rudnicki. A fast algorithm for constructing trees from distance matrices. *Information Processing Letters*, 30(4):215–220, 1989.
18. Argyrios Deligkas and Igor Potapov. Optimizing reachability sets in temporal graphs by delaying. *Information and Computation*, 285:104890, 2022.

19 Jessica Enright, Kitty Meeks, George B. Mertzios, and Viktor Zamaraev. Deleting edges to restrict the size of an epidemic in temporal networks. *Journal of Computer and System Sciences*, 119:60–77, 2021.

20 Jessica Enright, Kitty Meeks, and Fiona Skerman. Assigning times to minimise reachability in temporal graphs. *Journal of Computer and System Sciences*, 115:169–186, 2021.

21 Jessica A. Enright, Kitty Meeks, and Hendrik Molter. Counting temporal paths. *CoRR*, abs/2202.12055, 2022. URL: `https://arxiv.org/abs/2202.12055`.

22 Paul Erdős and Tibor Gallai. Graphs with prescribed degrees of vertices. *Mat. Lapok*, 11:264–274, 1960.

23 Thomas Erlebach, Michael Hoffmann, and Frank Kammer. On temporal graph exploration. *Journal of Computer and System Sciences*, 119:1–18, 2021.

24 Thomas Erlebach and Jakob T. Spooner. A game of cops and robbers on graphs with periodic edge-connectivity. In *Proceedings of the 46th International Conference on Current Trends in Theory and Practice of Informatics (SOFSEM)*, pages 64–75, 2020.

25 Michael R. Fellows, Danny Hermelin, Frances Rosamond, and Stéphane Vialette. On the parameterized complexity of multiple-interval graph problems. *Theoretical Computer Science*, 410(1):53–61, 2009.

26 Michael R. Fellows, Bart M. P. Jansen, and Frances A. Rosamond. Towards fully multivariate algorithmics: Parameter ecology and the deconstruction of computational complexity. *European Journal of Combinatorics*, 34(3):541–566, 2013.

27 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.

28 András Frank. Augmenting graphs to meet edge-connectivity requirements. *SIAM Journal on Discrete Mathematics*, 5(1):25–53, 1992.

29 András Frank. Connectivity augmentation problems in network design. *Mathematical Programming: State of the Art 1994*, 1994.

30 H. Frank and Wushow Chou. Connectivity considerations in the design of survivable networks. *IEEE Transactions on Circuit Theory*, 17(4):486–490, 1970.

31 Eugen Füchsle, Hendrik Molter, Rolf Niedermeier, and Malte Renken. Delay-robust routes in temporal graphs. In *Proceedings of the 39th International Symposium on Theoretical Aspects of Computer Science (STACS)*, pages 30:1–30:15, 2022.

32 Eugen Füchsle, Hendrik Molter, Rolf Niedermeier, and Malte Renken. Temporal connectivity: Coping with foreseen and unforeseen delays. In *Proceedings of the 1st Symposium on Algorithmic Foundations of Dynamic Networks (SAND)*, pages 17:1–17:17, 2022.

33 D.R. Fulkerson. Zero-one matrices with zero trace. *Pacific Journal of Mathematics*, 10(3):831–836, 1960.

34 Petr A. Golovach and George B. Mertzios. Graph editing to a given degree sequence. *Theoretical Computer Science*, 665:1–12, 2017.

35 Martin Charles Golumbic and Ann N. Trenk. *Tolerance Graphs*. Cambridge Studies in Advanced Mathematics. Cambridge University Press, 2004.

36 Ralph E Gomory and Tien Chung Hu. Multi-terminal network flows. *Journal of the Society for Industrial and Applied Mathematics*, 9(4):551–570, 1961.

37 Martin Grötschel, Clyde L Monma, and Mechthild Stoer. Design of survivable networks. *Handbooks in Operations Research and Management Science*, 7:617–672, 1995.

38 Jiong Guo, Falk Hüffner, and Rolf Niedermeier. A structural view on parameterizing problems: Distance from triviality. In *Proceedings of the 1st International Workshop on Parameterized and Exact Computation (IWPEC)*, pages 162–173, 2004.

39 S. Louis Hakimi. On realizability of a set of integers as degrees of the vertices of a linear graph. I. *Journal of the Society for Industrial and Applied Mathematics*, 10(3):496–506, 1962.

40 S. Louis Hakimi and Stephen S. Yau. Distance matrix of a graph and its realizability. *Quarterly of applied mathematics*, 22(4):305–317, 1965.

41 Pavol Hell and David Kirkpatrick. Linear-time certifying algorithms for near-graphical sequences. *Discrete Mathematics*, 309(18):5703–5713, 2009.

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

43 Nina Klobas, George B. Mertzios, Hendrik Molter, Rolf Niedermeier, and Philipp Zschoche. Interference-free walks in time: Temporally disjoint paths. *Autonomous Agents and Multi-Agent Systems*, 37(1):1, 2023.

44 Nina Klobas, George B. Mertzios, Hendrik Molter, and Paul G. Spirakis. The complexity of computing optimum labelings for temporal connectivity. In *Proceedings of the 47th International Symposium on Mathematical Foundations of Computer Science (MFCS)*, pages 62:1–62:15, 2022.

45 Fabian Kuhn and Rotem Oshman. Dynamic networks: Models and algorithms. *SIGACT News*, 42(1):82–96, mar 2011.

46 Hendrik W. Lenstra. Integer programming with a fixed number of variables. *Mathematics of Operations Research*, 8:538–548, 1983.

47 Jure Leskovec and Andrej Krevl. SNAP Datasets: Stanford large network dataset collection. `http://snap.stanford.edu/data`, June 2014.

48 Linda Lesniak. Eccentric sequences in graphs. *Periodica Mathematica Hungarica*, 6:287–293, 1975.

49 Ross M. McConnell and Jeremy P. Spinrad. Construction of probe interval models. In *Proceedings of the 13th Annual ACM-SIAM Symposium on Discrete Algorithms (SODA)*, pages 866–875, 2002.

50 F.R. McMorris, Chi Wang, and Peisen Zhang. On probe interval graphs. *Discrete Applied Mathematics*, 88(1):315–324, 1998. Computational Molecular Biology DAM - CMB Series.

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

52 George B. Mertzios, Hendrik Molter, Malte Renken, Paul G. Spirakis, and Philipp Zschoche. The complexity of transitively orienting temporal graphs. In *Proceedings of the 46th International Symposium on Mathematical Foundations of Computer Science (MFCS)*, pages 75:1–75:18, 2021.

53 Hendrik Molter, Malte Renken, and Philipp Zschoche. Temporal reachability minimization: Delaying vs. deleting. In *Proceedings of the 46th International Symposium on Mathematical Foundations of Computer Science (MFCS)*, pages 76:1–76:15, 2021.

54 Nils Morawietz, Carolin Rehs, and Mathias Weller. A timecop's work is harder than you think. In *Proceedings of the 45th International Symposium on Mathematical Foundations of Computer Science (MFCS)*, volume 170, pages 71–1, 2020.

55 Nils Morawietz and Petra Wolf. A timecop's chase around the table. In *Proceedings of the 46th International Symposium on Mathematical Foundations of Computer Science (MFCS)*, 2021.

56 A.N. Patrinos and S. Louis Hakimi. The distance matrix of a graph and its tree realization. *Quarterly of Applied Mathematics*, 30:255–269, 1972.

57 Elena Rubei. Weighted graphs with distances in given ranges. *Journal of Classification*, 33:282—-297, 2016.

58 Piotr Sapiezynski, Arkadiusz Stopczynski, Radu Gatej, and Sune Lehmann. Tracking human mobility using wifi signals. *PloS one*, 10(7):e0130824, 2015.

59 H. Tamura, M. Sengoku, S. Shinoda, and T. Abe. Realization of a network from the upper and lower bounds of the distances (or capacities) between vertices. In *Proceedings of the 1993 IEEE International Symposium on Circuits and Systems (ISCAS)*, pages 2545—-2548, 1993.

60 Huanhuan Wu, James Cheng, Yiping Ke, Silu Huang, Yuzhen Huang, and Hejun Wu. Efficient algorithms for temporal path computation. *IEEE Transactions on Knowledge and Data Engineering*, 28(11):2927–2942, 2016.

61 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.

APPENDIX

Temporal graph realization from fastest paths

Anonymous author
Anonymous affiliation

Anonymous author
Anonymous affiliation

Anonymous author
Anonymous affiliation

Anonymous author
Anonymous affiliation

Abstract

In this paper we initiate the study of the *temporal graph realization* problem with respect to the fastest path durations among its vertices, while we focus on periodic temporal graphs. Given an $n \times n$ matrix D and a $\Delta \in \mathbb{N}$, the goal is to construct a Δ-periodic temporal graph with n vertices such that the duration of a *fastest path* from v_i to v_j is equal to $D_{i,j}$, or to decide that such a temporal graph does not exist. The variations of the problem on static graphs has been well studied and understood since the 1960's (e.g. [Erdős and Gallai, 1960], [Hakimi and Yau, 1965]).

As it turns out, the periodic temporal graph realization problem has a very different computational complexity behavior than its static (i. e., non-temporal) counterpart. First we show that the problem is NP-hard in general, but polynomial-time solvable if the so-called underlying graph is a tree. Building upon those results, we investigate its parameterized computational complexity with respect to structural parameters of the underlying static graph which measure the "tree-likeness". We prove a tight classification between such parameters that allow fixed-parameter tractability (FPT) and those which imply W[1]-hardness. We show that our problem is W[1]-hard when parameterized by the *feedback vertex number* (and therefore also any smaller parameter such as *treewidth*, *degeneracy*, and *cliquewidth*) of the underlying graph, while we show that it is in FPT when parameterized by the *feedback edge number* (and therefore also any larger parameter such as *maximum leaf number*) of the underlying graph.

2012 ACM Subject Classification Theory of computation → Graph algorithms analysis; Mathematics of computing → Discrete mathematics

Keywords and phrases Temporal graph, periodic temporal labeling, fastest temporal path, graph realization, temporal connectivity.

Digital Object Identifier 10.4230/LIPIcs.CVIT.2016.23

1 Introduction

The (static) *graph realization* problem with respect to a graph property $\mathcal{P}$ is to find a graph that satisfies property $\mathcal{P}$, or to decide that no such graph exists. The motivation for graph realization problems stems both from "verification" and from network design applications in engineering. In *verification* applications, given the outcomes of some experimental measurements (resp. some computations) on a network, the aim is to (re)construct an input network which complies with them. If such a reconstruction is not possible, this proves that the measurements are incorrect or implausible (resp. that the algorithm which made the computations is incorrectly implemented). One example of a graph realization (or reconstruction) problem is the recognition of probe interval graphs, in the context of the physical mapping of DNA, see [49, 50] and [35, Chapter 4]. In *network design* applications, the goal is to design network topologies having a desired property [4, 37]. Analyzing the computational complexity of the graph realization problems for various natural

APPENDIX

and fundamental graph properties $\mathcal{P}$ requires a deep understanding of these properties. Among the most studied such parameters for graph realization are constraints on the distances between vertices [7, 8, 10, 16, 17, 40], on the vertex degrees [6, 22, 34, 36, 39], on the eccentricities [5, 9, 41, 48], and on connectivity [15, 28–30, 33, 36], among others.

In the simplest version of a (static) graph realization problem with respect to vertex distances, we are given a symmetric $n \times n$ matrix D and we are looking for an n-vertex undirected and unweighted graph G such that $D_{i,j}$ equals the distance between vertices v_i and v_j in G. This problem can be trivially solved in polynomial time in two steps [40]: First, we build the graph $G = (V, E)$ such that $v_i v_j \in E$ if and only if $D_{i,j} = 1$. Second, from this graph G we compute the matrix D_G which captures the shortest distances for all pairs of vertices. If $D_G = D$ then G is the desired graph, otherwise there is no graph having D as its distance matrix. Non-trivial variations of this problem have been extensively studied, such as for weighted graphs [40, 56], as well as for cases where the realizing graph has to belong to a specific graph family [7, 40]. Other variations of the problem include the cases where every entry of the input matrix D may contain a range of consecutive permissible values [7, 57, 60], or even an arbitrary set of acceptable values [8] for the distance between the corresponding two vertices.

In this paper we make the first attempt to understand the complexity of the graph realization problem with respect to vertex distances in the context of *temporal graphs*, i. e., of graphs whose *topology changes over time*.

▶ **Definition 1** (temporal graph [42]). *A* temporal graph *is a pair (G, λ), where $G = (V, E)$ is an underlying (static) graph and $\lambda : E \to 2^{\mathbb{N}}$ is a* time-labeling *function which assigns to every edge of G a set of discrete time-labels.*

Here, whenever $t \in \lambda(e)$, we say that the edge e is *active* or *available* at time t. In the context of temporal graphs, where the notion of vertex adjacency is time-dependent, the notions of path and distance also need to be redefined. The most natural temporal analogue of a path is that of a *temporal* (or *time-dependent*) path, which is motivated by the fact that, due to causality, entities and information in temporal graphs can "flow" only along sequences of edges whose time-labels are strictly increasing.

▶ **Definition 2** (fastest temporal path). *Let (G, λ) be a temporal graph. A* temporal path *in (G, λ) is a sequence $(e_1, t_1), (e_2, t_2), \ldots, (e_k, t_k)$, where $P = (e_1, \ldots, e_k)$ is a path in the underlying static graph G, $t_i \in \lambda(e_i)$ for every $i = 1, \ldots, k$, and $t_1 < t_2 < \ldots < t_k$. The* duration *of this temporal path is $t_k - t_1 + 1$. A* fastest *temporal path from a vertex u to a vertex v in (G, λ) is a temporal path from u to v with the smallest duration. The duration of the* fastest *temporal path from u to v is denoted by $d(u, v)$.*

In this paper we consider *periodic* temporal graphs, i. e., temporal graphs in which the temporal availability of each edge of the underlying graph is periodic. Many natural and technological systems exhibit a periodic temporal behavior. For example, in railway networks an edge is present at a time step t if and only if a train is scheduled to run on the respective rail segment at time t [3]. Similarly, a satellite, which makes pre-determined periodic movements, can establish a communication link (i. e., a temporal edge) with another satellite whenever they are sufficiently close to each other; the existence of these communication links is also periodic. In a railway (resp. satellite) network, a fastest temporal path from u to v represents the fastest railway connection between two stations (resp. the quickest communication delay between two moving satellites). Furthermore, periodicity appears also in (the otherwise quite complex) social networks which describe the dynamics of people meeting [47, 58], as every person individually follows mostly a daily routine [3].

APPENDIX

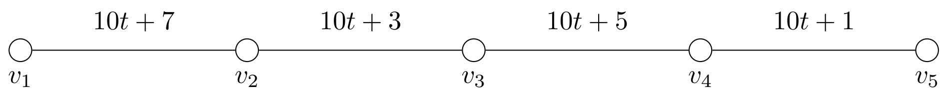


Figure 1 An example of a Δ-periodic temporal graph (G, λ, Δ), where $\Delta = 10$ and the 10-periodic labeling $\lambda : E \to \{1, 2, \dots, 10\}$ is as follows: $\lambda(v_1v_2) = 7$, $\lambda(v_2v_3) = 3$, $\lambda(v_3v_4) = 5$, and $\lambda(v_4v_5) = 1$. Here, the fastest temporal path from u to v traverses the first edge v_1v_2 at time 7, second edge v_2v_3 a time 13, third edge v_3v_4 at time 15 and the last edge v_4v_5 at time 21. This results in the total duration of 15 for the fastest temporal path from v_1 to v_5.

Although periodic temporal graphs have already been studied (see [13, Class 8] and [3, 24, 54, 55]), we make here the first attempt to understand the complexity of a graph realization problem in the context of temporal graphs. Therefore, we focus in this paper on the most fundamental case, where all edges have the same period Δ (while in the more general case, each edge e in the underlying graph has a period Δ_e). As it turns out, the periodic temporal graph realization problem with respect to a given $n \times n$ matrix D of the fastest duration times has a very different computational complexity behavior than the classic graph realization problem with respect to shortest path distances in static graphs.

Formally, let $G = (V, E)$ and $\Delta \in \mathbb{N}$, and let $\lambda : E \to \{1, 2, \dots, \Delta\}$ be an edge-labeling function that assigns to every edge of G exactly one of the labels from $\{1, \dots, \Delta\}$. Then we denote by (G, λ, Δ) the Δ-*periodic temporal graph* (G, L), where for every edge $e \in E$ we have $L(e) = \{i\Delta + x : i \geq 0, x \in \lambda(e)\}$. In this case we call λ a Δ-*periodic labeling* of G; see Figure 1 for an illustration. When it is clear from the context, we drop Δ form the notation and we denote the (Δ-periodic) temporal graph by (G, λ). Given a duration matrix D, it is easy to observe that, similarly to the static case, if $D_{i,j} = 1$ then v_i and v_j must be connected by an edge. We call the graph defined by these edges the *underlying graph* of D.

Our contribution. We initiate the study of naturally motivated graph realization problems in the temporal setting. Our target is not to model unreliable communication, but instead to *verify* that particular measurements regarding fastest temporal paths in a periodic temporal graph are plausible (i. e., "realizable"). To this end, we introduce and investigate the following problem, capturing the setting described above:

Simple periodic Temporal Graph Realization (Simple TGR)

Input: An $n \times n$ matrix D, a positive integer Δ.

Question: Does there exist a graph $G = (V, E)$ with vertices $\{v_1, \dots, v_n\}$ and a Δ-periodic labeling $\lambda : E \to \{1, 2, \dots, \Delta\}$ such that, for every i, j, the duration of the fastest temporal path from v_i to v_j in the Δ-periodic temporal graph (G, λ, Δ) is $D_{i,j}$?

We focus on exact algorithms. We start by showing NP-hardness of the problem (Theorem 3), even if Δ is a small constant. To establish a baseline for tractability, we show that Simple TGR is polynomial-time solvable if the underlying graph is a tree (Theorem 22).

Building upon these initial results, we explore the possibilities to generalize our polynomial-time algorithm using the *distance-from-triviality* parameterization paradigm [26, 38]. That is, we investigate the parameterized computational complexity of Simple TGR with respect to structural parameters of the underlying graph that measure its "tree-likeness".

We obtain the following results. We show that Simple TGR is W[1]-hard when parameterized by the feedback vertex number of the underlying graph (Theorem 4). To this end, we first give a reduction from Multicolored Clique parameterized by the number of colors [25] to a variant of Simple TGR where the period Δ is infinite, that is, when the

APPENDIX

labeling is non-periodic. We use a special gadget (the "infinity" gadget) which allows us to transfer the result to a finite period Δ. The latter construction is independent from the particular reduction we use, and can hence be treated as a reduction from the non-periodic to the periodic setting. Note that our parameterized hardness result rule out fixed-parameter tractability for several popular graph parameters such as *treewidth*, *degeneracy*, *cliquewidth*, *distance to chordal graphs*, and *distance to outerplanar graphs*.

We complement this hardness result by showing that Simple TGR is fixed-parameter tractable (FPT) with respect to the *feedback edge number* k of the underlying graph (Theorem 23). This result also implies an FPT algorithm for any larger parameter, such as the *maximum leaf number*. A similar phenomenon of getting W[1]-hardness with respect to the feedback vertex number, while getting an FPT algorithm with respect to the feedback edge number, has been observed only in a few other temporal graph problems related to the connectivity between two vertices [14, 21, 31].

Our FPT algorithm works as follows on a high level. First we distinguish $O(k^2)$ vertices which we call "important vertices". Then, we guess the fastest temporal paths for each pair of these important vertices; as we prove, the number of choices we have for all these guesses is upper bounded by a function of k. Then we also need to make several further guesses (again using a bounded number of choices), which altogether leads us to specify a small (i. e., bounded by a function of k) number of different configurations for the fastest paths between *all pairs* of vertices. For each of these configurations, we must then make sure that the labels of our solution will not allow any other temporal path from a vertex v_i to a vertex v_j have a *strictly smaller* duration than $D_{i,j}$. This naturally leads us to build one Integer Linear Program (ILP) for each of these configurations. We manage to formulate all these ILPs by having a number of variables that is upper-bounded by a function of k. Finally we use Lenstra's Theorem [46] to solve each of these ILPs in FPT time. At the end, our initial instance is a Yes-instance if and only if at least one of these ILPs is feasible.

The above results provide a fairly complete picture of the parameterized computational complexity of Simple TGR with respect to structural parameters of the underlying graph which measure "tree-likeness". To obtain our results, we prove several properties of fastest temporal paths, which may be of independent interest.

Related work. Graph realization problems on static graphs have been studied since the 1960s. We provide an overview of the literature in the introduction. To the best of our knowledge, we are the first to consider graph realization problems in the temporal setting. However, many other connectivity-related problems have been studied in the temporal setting [2, 12, 18, 19, 23, 27, 32, 43, 52, 53, 62], most of which are much more complex and computationally harder than their non-temporal counterparts, and some of which do not even have a non-temporal counterpart.

There are some problem settings that share similarities with ours, which we discuss now in more detail.

Several problems have been studied where the goal is to assign labels to (sets of) edges of a given static graph in order to achieve certain connectivity-related properties [1, 20, 44, 51]. The main difference to our problem setting is that in the mentioned works, the input is a graph and the sought labeling is not periodic. Furthermore, the investigated properties are temporal connectivity between all vertices [1, 44, 51], temporal connectivity among a subset of vertices [44], or reducing reachability among the vertices [20]. In all these cases, the duration of the temporal paths has not been considered.

Finally, there are many models for dynamic networks in the context of distributed

APPENDIX

computing [45]. These models have some similarity to temporal graphs, in the sense that in both cases the edges appear and disappear over time. However, there are notable differences. For example, one important assumption in the distributed setting can be that the edge changes are adversarial or random (while obeying some constraints such as connectivity), and therefore they are not necessarily known in advance [45].

Preliminaries and notation. We already introduced the most central notion and concepts. There are some additional definitions we need, to present our proofs and results which we give in the following.

An interval in $\mathbb{N}$ from a to b is denoted by $[a, b] = \{i \in \mathbb{N} : a \leq i \leq b\}$; similarly, $[a] = [1, a]$. An undirected graph $G = (V, E)$ consists of a set V of vertices and a set $E \subseteq V \times V$ of edges. For a graph G, we also denote by $V(G)$ and $E(G)$ the vertex and edge set of G, respectively. We denote an edge $e \in E$ between vertices $u, v \in V$ as a set $e = \{u, v\}$. For the sake of simplicity of the representation, an edge e is sometimes also denoted by uv. A path P in G is a subgraph of G with vertex set $V(P) = \{v_1, \ldots, v_k\}$ and edge set $E(P) = \{\{v_i, v_{i+1}\} : 1 \leq i < k\}$ (we often represent path P by the tuple $(v_1, v_2, \ldots, v_k)$).

Let $v_1, v_2, \ldots, v_n$ be the n vertices of the graph G. For simplicity of the presentation (and with a slight abuse of notation) we refer during the paper to the entry $D_{i,j}$ of the matrix D as $D_{a,b}$, where $a = v_i$ and $b = v_j$. That is, we put as indices of the matrix D the corresponding vertices of G whenever it is clear from the context.

Let $P = (u = v_1, v_2, \ldots, v_p = v)$ be a path from u to v in G. Recall that, in our paper, every edge has exactly one time label in every period of Δ consecutive time steps. Therefore, as we are only interested in the fastest duration of temporal paths, many times we refer to (P, λ, Δ) as any of the temporal paths from $u = v_1$ to $v = v_p$ along the edges of P, which starts at the edge v_1v_2 at time $\lambda(v_1v_2) + c\Delta$, for some $c \in \mathbb{N}$, and then sequentially visits the rest of the edges of P as early as possible. We denote by $d(P, \lambda, \Delta)$, or simply by $d(P, \lambda)$ when Δ is clear from the context, the duration of any of the temporal paths (P, λ, Δ); note that they all have the same duration. Many times we also refer to a path $P = (u = v_1, v_2, \ldots, v_p = v)$ from u to v in G, as a temporal path in (G, λ, Δ), where we actually mean that (P, λ, Δ) is a temporal path with P as its underlying (static) path.

We remark that a fastest path between two vertices in a temporal graph can be computed in polynomial time [11, 61]. Hence, given a Δ-periodic temporal graph (G, λ, Δ), we can compute in polynomial-time the matrix D which consists of durations of fastest temporal paths among all pairs of vertices in (G, λ, Δ).

Organization of the paper. In Section 2 we present our hardness results, first the NP-hardness in Section 2.1 and then the parameterized hardness in Section 2.2. In Section 3 we present our algorithmic results. First we give in Section 3.1 a polynomial-time algorithm for the case where the underlying graph is a tree. In Section 3.2 we generalize this and present our FPT result, which is the main result in the paper. Finally, we conclude in Section 4 and discuss some future work directions.

2 Hardness results for Simple TGR

In this section we present our main computational hardness results. In Section 2.1 we show that Simple TGR is NP-hard even for constant Δ. In Section 2.2 we investigate the parameterized computational hardness of Simple TGR with respect to structural parameters

APPENDIX

of the underlying graph. We show that Simple TGR is W[1]-hard when parameterized by the feedback vertex number of the underlying graph.

2.1 NP-hardness of Simple TGR

In this section we prove that in general it is NP-hard to determine a Δ-periodic temporal graph (G, λ) respecting a duration matrix D, even if Δ is a small constant.

▶ **Theorem 3.** *Simple TGR is NP-hard for all* $\Delta \geq 3$.

Proof. We present a polynomial-time reduction from the NP-hard problem NAE 3-SAT [59]. Here we are given a formula ϕ that is a conjunction of so-called NAE (not-all-equal) clauses, where each clause contains exactly 3 literals (with three distinct variables). A NAE clause evaluates to TRUE if and only if not all of its literals are equal, that is, at least one literal evaluates to TRUE and at least one literal evaluates to FALSE. We are asked whether ϕ admits a satisfying assignment.

Given an instance ϕ of NAE 3-SAT, we construct an instance (D, Δ) of Simple TGR as follows.

We start by describing the vertex set of the underlying graph G of D.

- For each variable x_i in ϕ, we create three variable vertices x_i, x_i^T, x_i^F.
- For each clause c in ϕ, we create one clause vertex c.
- We add one additional super vertex v.

Next, we describe the edge set of G.

- For each variable x_i in ϕ we add the following five edges: $\{x_i, x_i^T\}$, $\{x_i, x_i^F\}$, $\{x_i^T, x_i^F\}$, $\{x_i^T, v\}$, and $\{x_i^F, v\}$.
- For each pair of variables x_i, x_j in ϕ with $i \neq j$ we add the following four edges: $\{x_i^T, x_j^T\}$, $\{x_i^T, x_j^F\}$, $\{x_i^F, x_j^T\}$, and $\{x_i^F, x_j^F\}$.
- For each clause c in ϕ we add one edge for each literal. Let x_i appear in c. If x_i appears non-negated in c we add edge $\{c, x_i^T\}$. If x_i appears negated in c we add edge $\{c, x_i^F\}$.

This finishes the construction of G. For an illustration see Figure 2.

We set Δ to some constant larger than two, that is, $\Delta \geq 3$. Next, we specify the durations in the matrix D between all vertex pairs. For the sake of simplicity we write $D_{u,v}$ as $d(u, v)$, where u, v are two vertices of G. We start by setting the value of $d(u, v) = 1$ where u and v are two adjacent vertices in G.

- For each variable x_i in ϕ and the super vertex v we specify the following durations: $d(x_i, v) = 2$ and $d(v, x_i) = \Delta$.
- For each clause c in ϕ and the super vertex v we specify the following durations: $d(c, v) = 2$ and $d(v, c) = \Delta - 1$.
- Let x_i be a variable that appears in clause c, then we specify the following durations: $d(c, x_i) = 2$ and $d(x_i, c) = \Delta$. If x_i appears non-negated in c we specify the following durations: $d(c, x_i^F) = 2$ and $d(x_i^F, c) = \Delta$. If x_i appears negated in c we specify the following duratios: $d(c, x_i^T) = 2$ and $d(x_i^T, c) = \Delta$.
- Let x_i be a variable that does *not* appear in clause c, then we specify the following duratios: $d(x_i, c) = 2\Delta$, $d(c, x_i) = \Delta + 2$ and $d(c, x_i^T) = d(c, x_i^F) = 2$, $d(x_i^T, c) = d(x_i^F, c) = \Delta$.
- For each pair of variables $x_i \neq x_j$ in ϕ we specify the following duratios: $d(x_i, x_j) = 2\Delta + 1$ and $d(x_i, x_j^T) = d(x_i, x_j^F) = \Delta + 1$.
- For each pair of clauses $c_i \neq c_j$ in ϕ we specify the following duratios: $d(c_i, c_j) = \Delta + 1$.

This finishes the construction of the instance (D, Δ) of Simple TGR which can clearly be done in polynomial time. In the remainder we show that (D, Δ) is a Yes-instance of Simple TGR if and only if NAE 3-SAT formula ϕ is satisfiable.

APPENDIX

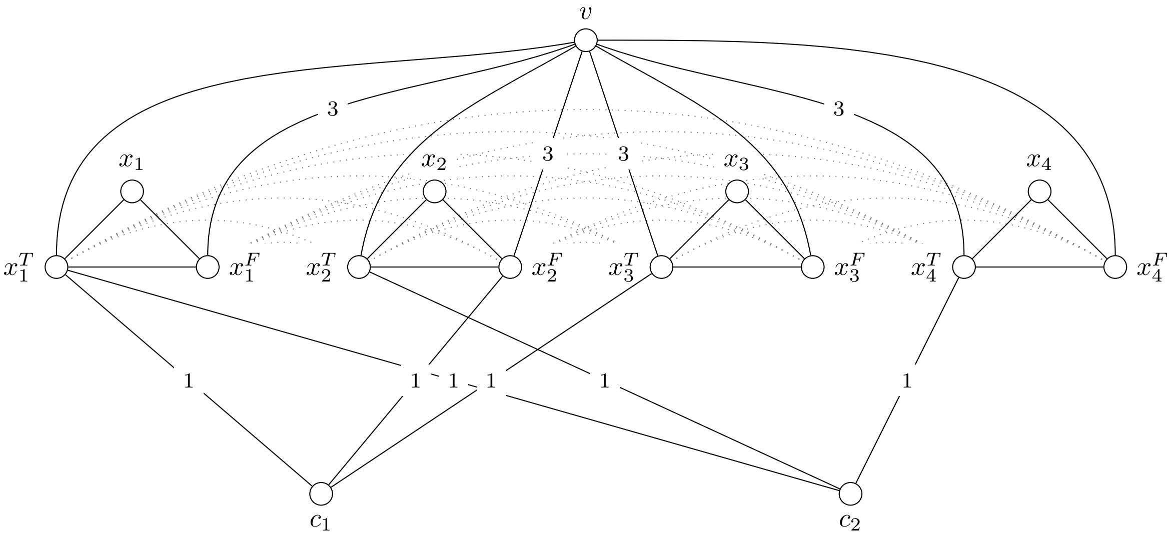


Figure 2 Illustration of the temporal graph (G, λ) from the NP-hardness reduction, where the NAE 3-SAT formula ϕ is of the form $\phi = \text{NAE}(x_1, \overline{x}_2, x_3) \wedge \text{NAE}(x_1, x_2, x_4)$. To improve the readability, we draw edges between vertices x_i^T and x_j^F (where $i \neq j$) with gray dotted lines. Presented is the labeling of G corresponding to the assignment $x_1 = x_2 =$ TRUE and $x_3, x_4 =$ FALSE, where all unlabeled edges get the label 2.

($\Rightarrow$): Assume the constructed instance (D, Δ) of SIMPLE TGR is a YES-instance. Then there exist a label $\lambda(e)$ for each edge $e \in E(G)$ such that for each vertex pair u, w in the temporal graph (G, λ, Δ) we have that a fastest temporal path from u to w is of duration $d(u, w)$.

We construct a satisfying assignment for ϕ as follows. For each variable x_i, if $\lambda(\{x_i, x_i^T\}) = \lambda(\{x_i^T, v\})$, then we set x_i to TRUE, otherwise we set x_i to FALSE.

To show that this yields a satisfying assignment, we need to prove some properties of the labeling λ. First, observe that adding an integer t to all time labels does not change the duration of any temporal paths. Second, observe that if for two vertices u, w we have that $d(u, w)$ equals the distance between u and w in G (i. e., the duration of the fastest temporal path from u to w equals the distance of the shortest path between u and w), then there is a shortest path P from u to w in G such that the labeling λ assigns consecutive time labels to the edges of P.

Let $\lambda(\{x_i, x_i^T\}) = t$ and $\lambda(\{x_i, x_i^F\}) = t'$, for an arbitrary variable x_i. If both $\lambda(\{x_i^T, v\}) \neq t + 1$ and $\lambda(\{x_i^F, v\}) \neq t' + 1$, then $d(x_i, v) > 2$, which is a contradiction. Thus, for every variable x_i, we have that $\lambda(\{x_i^T, v\}) = t + 1$ or $\lambda(\{x_i^F, v\}) = t' + 1$ (or both). In particular, this means that if $\lambda(\{x_i, x_i^F\}) = \lambda(\{x_i^F, v\})$, then we set x_i to FALSE, since in this case $\lambda(\{x_i, x_i^T\}) \neq \lambda(\{x_i^T, v\})$.

Now assume for a contradiction that the described assignment is not satisfying. Then there exists a clause c that is not satisfied. Suppose that x_1, x_2, x_3 are three variables that appear in c. Recall that we require $d(c, v) = 2$ and $d(v, c) = \Delta - 1$. The fact that $d(c, v) = 2$ implies that we must have a temporal path consisting of two edges from c to v, such that the two edges have consecutive labels. By construction of G there are three candidates for such a path, one for each literal of c. Assume w.l.o.g. that x_1 appears in c non-negated (the case of a negated appearance of x_1 is symmetrical) and that the temporal path realizing $d(c, v) = 2$ goes through vertex x_1^T. Let us denote with $t = \lambda(\{x_1^T, v\})$. It follows that

APPENDIX

$\lambda(\{x_1^T, c\}) = \lambda(\{x_1^T, v\}) - 1 = t - 1$. Furthermore, since $d(c, x_1) = 2$ we also have that $\lambda(\{x_1^T, c\}) = \lambda(\{x_1, x_1^T\}) - 1$. Therefore $\lambda(\{x_1, x_1^T\}) = \lambda(\{x_1^T, v\}) = t$. Which implies that x_1 is set to TRUE. Let us observe paths from v to c. We know that $d(v, c) = \Delta - 1$. The underlying path of the fastest temporal path from v to c, that goes through x_1^T is the path $P = (v, x_1^T, c)$. Since $\lambda(\{x_1^T, c\}) > \lambda(\{x_1^T, v\})$ we get that the duration of the temporal path (P, λ) is equal to $d(P, \lambda) = (\Delta + t - 1) - t + 1 = \Delta$. This implies that the fastest temporal path from v to c is not (P, λ) and therefore does not pass through x_1^T. Since there are only two other vertices connected to c, we have only two other edges incident to c, that can be used on a fastest temporal path v to c. Suppose now w.l.o.g. that also x_2 appears in c non-negated (the case of a negated appearance of x_2 is symmetrical) and that the temporal path realizing $d(v, c) = \Delta - 1$ goes through vertex x_2^T. Let us denote with $t' = \lambda(\{x_2^T, v\})$. Since the fastest temporal path from v to c is of the duration $\Delta - 1$, and the edge $x_2^T c$ is the only edge incident to vertex c and edge $\{x_2^T, v\}$, it follows that $\lambda(\{x_2^T, c\}) \geq \lambda(\{x_2^T, v\}) - 2 = t' - 2$. Since $d(x_2, v) = 2$ it follows that $\lambda(\{x_2, x_2^T\}) = \lambda(\{x_2^T, v\}) - 1 = t' - 1$. Knowing this and the fact that $d(x_2, c) = 2$, we get that $\lambda(\{x_2^T, c\})$ must be equal to $t' - 2$. Therefore the fastest temporal path from v to c passes through edges $\{x_2^T, v\}$ and $\{x_2^T, c\}$. In the above we have also determined that $\lambda(\{x_2, x_2^T\}) \neq \lambda(\{x_2^T, v\})$, which implies that x_2 is set to FALSE. But now we have that x_1, x_2 both appear in c non-negated, where one of them is TRUE, while the other is FALSE, which implies that the clause c is satisfied, a contradiction.

($\Leftarrow$): Assume that ϕ is satisfiable. Then there exists a satisfying assignment for the variables in ϕ.

We construct a labeling λ as follows.

- All edges incident with a clause vertex c obtain label one.
- If variable x_i is set to TRUE, we set $\lambda(\{x_i^F, v\}) = 3$.
- If variable x_i is set to FALSE, we set $\lambda(\{x_i^T, v\}) = 3$.
- We set the labels of all other edges to two.

For an example of the constructed temporal graph see Figure 2. We now verify that all duratios are realized.

- For each variable x_i in ϕ we have to check that $d(x_i, v) = 2$ and $d(v, x_i) = \Delta$.
  If x_i is set to TRUE, then there is a temporal path from x_i to v via x_i^F of duration 2, since $\lambda(\{x_i, x_i^F\}) = 2$ and $\lambda(\{x_i^F, v\}) = 3$. For a temporal path from v to x_i we observe the following. The only possible labels to leave the vertex v are 2 and 3, which take us from v to x_j^T or x_j^F of some variable x_j. The only two edges incident to x_i have labels 2, therefore the fastest path from v to x_i cannot finish before the time $\Delta + 2$. The fastest way to leave v and enter to x_i would then be to leave v at edge $\{x_i^F, v\}$ with label 3, and continue to x_i at time $\Delta + 2$, which gives us the desired duration Δ.
  If x_i is set to FALSE, then, by similar arguing, there is a temporal path from x_i to v via x_i^T of duration 2, and a temporal path from v to x_i, through x_i^F of duration Δ.
- For each clause c in ϕ we have to check that $d(c, v) = 2$ and $d(v, c) = \Delta - 1$:
  Suppose x_i, x_j, x_k appear in c. Since we have a satisfying assignment at least one of the literals in c is set to TRUE and at least one to FALSE. Suppose x_i is the variable of the literal that is TRUE in c, and x_j is the variable of the literal that is FALSE in c. Let x_i appear non-negated in c and is therefore set to TRUE (the case when x_i appears negated in c and is set to FALSE is symmetric). Then there is a temporal path from c to v through x_i^T such that $\lambda(\{x_i^T, c\}) = 1$ and $\lambda(\{x_i^T, v\}) = 2$. Let x_j appear non-negated in c and is therefore set to FALSE (the case when x_j appears negated in c and is set to TRUE is symmetric). Then there is a temporal path from v to c through x_j^T such that $\lambda(\{x_j^T, v\}) = 3$ and $\lambda(\{x_j^T, c\}) = 1$, which results in a temporal path from v to c of

APPENDIX

duration $\Delta - 1$.

- Let x_i be a variable that appears in clause c. If x_i appears non-negated in c we have to check that $d(c, x_i) = d(c, x_i^F) = 2$ and $d(x_i, c) = d(x_i^F, c) = \Delta$.
  There is a temporal path from c to x_i via x_i^T and also a temporal path from c to x_i^F via x_i^T such that $\lambda(\{x_i^T, c\}) = 1$ and $\lambda(\{x_i, x_i^T\}) = \lambda(\{x_i^T, x_i^F\}) = 2$, which proves the first equality. There are also the following two temporal paths, first, from x_i to c through x_i^T and second, from x_i^F to c through x_i^T. Both of the temporal paths start on the edge with label 2, as $\lambda(\{x_i, x_i^T\}) = \lambda(\{x_i^T, x_i^F\}) = 2$ and finish on the edge with label 1, as $\lambda(\{x_i^T, c\}) = 1$.
  If x appears negated in c we have to check that $d(c, x_i) = d(c, x_i^T) = 2$ and $d(x_i, c) = d(x_i^T, c) = \Delta$.
  There is a temporal path from c to x via x^F and also a temporal path from c to x^T via x^F such that $\lambda(\{c, x^F\}) = 1$ and $\lambda(\{x, x^F\}) = \lambda(\{x^T, x^F\}) = 2$, which proves the first inequality. There are also the following two temporal paths, first, from x_i to c through x_i^F and second, from x_i^T to c through x_i^F. Both of the temporal paths start on the edge with label 2, as $\lambda(\{x_i, x_i^F\}) = \lambda(\{x_i^T, x_i^F\}) = 2$ and finish on the edge with label 1, as $\lambda(\{x_i^F, c\}) = 1$. Which proves the second equality.
- Let x_i be a variable that does *not* appear in clause c, then we have to check that first, $d(c, x_i^T) = d(c, x_i^F) = 2$, second, $d(x_i^T, c) = d(x_i^F, c) = \Delta$, third, $d(c, x_i) = \Delta + 2$, and fourth $d(x_i, c) = 2\Delta$.
  Let x_j be a variable that appears non-negated in c (the case where x_j appears negated is symmetric). Then there is a temporal path from c to x_i^T via x_j^T and also a temporal path from c to x_i^F via x_j^T such that $\lambda(\{x_j^T, c\}) = 1$ and $\lambda(\{x_j^T, x_i^T\}) = \lambda(\{x_j^T, x_i^F\}) = 2$, which proves the first equality. Using the same temporal path in the opposite direction, i. e., first the edge $x_j^T c$ and then one of the edges $\{x_j^T, x_i^F\}$ or $\{x_j^T, x_i^T\}$ at times 2 and $\Delta + 1$, respectively, yields the second equality. For a temporal path from c to x_i we traverse the following three edges $\{x_j^T, c\}$, $\{x_j^T, x_i^F\}$, and $\{x_i^F, x_i\}$, with labels 1, 2, and 2 respectively (i. e., the path traverses them at time $1, 2$ and $\Delta + 2$, respectively), which proves the third equality. Now for the case of a temporal path from x_i to c, we use the same three edges, but in the opposite direction, namely $\{x_i^F, x_i\}$, $\{x_j^T, x_i^F\}$, and $\{x_j^T, c\}$, again at times 2, $\Delta + 2$, and $2\Delta + 1$, respectively, which proves the last equality. Note that all of the above temporal paths are also the shortest possible, and since the labels of first and last edges (of these paths) are unique, it follows that we cannot find faster temporal paths.
- For each pair of variables $x_i \neq x_j$ in ϕ we have to check that $d(x_i, x_j) = 2\Delta + 1$ and $d(x_i, x_j^T) = d(x_i, x_j^F) = \Delta + 1$.
  There is a path from x_i to x_j that passes first through one of the vertices x_i^T or x_i^F, and then through one of the vertices x_j^T or x_j^F. This temporal path is of length 3, where all of the edges have label 2, which proves the first equality. Now, a temporal path from x_i to x_j^T (resp. x_j^F), passes through one of the vertices x_i^T or x_i^F. This path is of length two, where all of the edges have label 2, which proves the second equality. Note that all of the above temporal paths are also the shortest possible, and since the labels of first and last edges (of these paths) are unique, it follows that we cannot find faster temporal paths.
- For each pair of clauses $c_i \neq c_j$ in ϕ we have to check that $d(c_i, c_j) = \Delta + 1$.
  Let x_k be a variable that appears non-negated in c_i and x_ℓ the variable that appears non-negated in c_j (all other cases are symmetric). There is a path of length three from c_i to c_j that passes first through vertex x_k^T and then through vertex x_ℓ^T. Therefore the temporal path from c_i to c_j uses the edges $\{x_k^T, c_i\}$, $\{x_\ell^T, c_j\}$, and $\{x_k^T, x_\ell^T\}$, with labels 1, 2, and 1 (at times 1, 2, and $\delta + 1$), respectively, which proves the desired equality. Note

APPENDIX

also that this is the shortest path between c_i and c_j, and that the first and the last edge must have the label 1, therefore it follows that this is the fastest temporal path.

Lastly, observe that the above constructed labeling λ uses values $\{1, 2, 3\} \subseteq [\Delta]$, therefore $\Delta \geq 3$. ◀

2.2 Parameterized hardness of Simple TGR

In this section, we investigate the parameterized hardness of Simple TGR with respect to structural parameters of the underlying graph. We show that the problem is W[1]-hard when parameterized by the feedback vertex number of the underlying graph. The *feedback vertex number* of a graph G is the cardinality of a minimum vertex set $X \subseteq V(G)$ such that $G - X$ is a forest. The set X is called a *feedback vertex set*. Note that, in contrast to the result of the previous section (Theorem 3), the reduction we use to obtain the following result does not produce instances with a constant Δ.

▶ **Theorem 4.** *Simple TGR is W[1]-hard when parameterized by the feedback vertex number of the underlying graph.*

Proof. We present a parameterized reduction from the W[1]-hard problem Multicolored Clique parameterized by the number of colors [25]. Here, given a k-partite graph $H = (W_1 \uplus W_2 \uplus \ldots \uplus W_k, F)$, we are asked whether H contains a clique of size k. If $w \in W_i$, then we say that w has *color* i. W.l.o.g. we assume that $|W_1| = |W_2| = \ldots = |W_k| = n$ and that every vertex has at least one neighbor of every color. Furthermore, for all $i \in [k]$, we assume the vertices in W_i are ordered in some arbitrary but fixed way, that is, $W_i = \{w_1^i, w_2^i, \ldots, w_n^i\}$. Let $F_{i,j}$ with $i < j$ denote the set of all edges between vertices from W_i and W_j. We assume w.l.o.g. that $|F_{i,j}| = m$ for all $i < j$ (if not we can add $k \max_{i,j} |F_{i,j}|$ vertices to each W_i and use those to add up to $\max_{i,j} |F_{i,j}|$ additional isolated edges to each $F_{i,j}$). Furthermore, for all $i < j$ we assume that the edges in $F_{i,j}$ are ordered in some arbitrary but fixed way, that is, $F_{i,j} = \{e_1^{i,j}, e_2^{i,j}, \ldots, e_m^{i,j}\}$.

We give a reduction to a variant of Simple TGR where the period Δ is infinite (that is, the sought temporal graph is not periodic) and we allow D to have infinity entries, meaning that the two respective vertices are not temporally connected. Note that, given the matrix D, we can easily compute the underlying graph G, as follows. Two vertices v, v' are adjacent if G if and only if $D_{v,v'} = 1$, as having an edge between v and v' is the only way that there exists a temporal path from v to v' with duration 1. For simplicity of the presentation of the reduction, we describe the underlying graph G (which directly implies the entries of D where $D(v, v') = 1$) and then we provide the remaining entries of D. At the end of the proof we show how to obtain the result for a finite Δ and a matrix D of durations of fastest paths, that only has finite entries.

In the following, we give an informal description of the main ideas of the reduction. The construction uses several gadgets, where the main ones are an "edge selection gadget" and a "verification gadget".

Every *edge selection gadget* is associated with a color combination i, j in the Multicolored Clique instance, and its main purpose is to "select" an edge connecting a vertex from color i with a vertex from color j. Roughly speaking, the edge selection gadget consists of m paths, one for every edge in $F_{i,j}$ (see Figure 3 for reference). The distance matrix D will enforce that the labels on those paths effectively order them temporally, that is, in particular, the labels on one of the paths will be smaller than the labels on all other paths. The edge corresponding to this path is selected.

APPENDIX

We have a *verification gadget* for every color i. They interact with the edge selection gadgets as follows. The verification gadget for color i is connected to all edge selection gadgets that involve color i. More specifically, this is connected to every path corresponding to an edge at a position in the path that encodes the endpoint of color i of that edge (again, see Figure 3) for reference. Intuitively, the distances in the verification gadget are only realizable if the selected edges all have the same endpoint of color i. Hence, the distances of all verification gadgets can be realized if and only if the selected edges form a clique.

Furthermore, we use an *alignment gadget* which, intuitively, ensures that the labelings of all gadgets use the same range of time labels. Finally, we use *connector gadgets* which create shortcuts between all vertex pairs that are irrelevant for the functionality of the other gadgets. This allows us to easily fill in the distance matrix with the corresponding values. We ensure that all our gadgets have a constant feedback vertex number, hence the overall feedback vertex number is quadratic in the number of colors of the MULTICOLORED CLIQUE instance and we get the parameterized hardness result.

In the following, for every gadget, we first give a formal description of the underlying graph of this gadget (i. e., not the complete distance sub-matrix of the gadget). Afterwards, we define the corresponding entries in the distance matrix D.

Given an instance H of MULTICOLORED CLIQUE, we construct an instance D of SIMPLE TGR (with infinity entries and no periods) as follows.

Edge selection gadget. We first introduce an *edge selection gadget $G_{i,j}$ for color combination* i, j with $i < j$. We start with describing the vertex set of the gadget.

- A set $X_{i,j}$ of vertices $x_1, x_2, \ldots, x_m$.
- Vertex sets $U_1, U_2, \ldots, U_m$ with $4n + 1$ vertices each, that is, $U_\ell = \{u_0^\ell, u_1^\ell, u_2^\ell, \ldots, u_{4n}^\ell\}$ for all $\ell \in [m]$.
- Two special vertices $v_{i,j}^\star, v_{i,j}^{\star\star}$.

The gadget has the following edges.

- For all $\ell \in [m]$ we have edge $\{x_\ell, v_{i,j}^\star\}$, $\{v_{i,j}^\star, u_0^\ell\}$, and $\{u_{4n}^\ell, v_{i,j}^{\star\star}\}$.
- For all $\ell \in [m]$ and $\ell' \in [4n]$, we have edge $\{u_{\ell'-1}^\ell, u_{\ell'}^\ell\}$.

Verification gadget. For each color i, we introduce the following vertices. What we describe in the following will be used as a *verification gadget for color i*.

- We have one vertex y^i and $k + 1$ vertices v_ℓ^i for $0 \leq \ell \leq k$.
- For every $\ell \in [m]$ and every $j \in [k] \setminus \{i\}$ we have $5n$ vertices $a_1^{i,j,\ell}, a_2^{i,j,\ell}, \ldots, a_{5n}^{i,j,\ell}$ and $5n$ vertices $b_1^{i,j,\ell}, b_2^{i,j,\ell}, \ldots, b_{5n}^{i,j,\ell}$.
- We have a set $\hat{U}_i$ of $13n + 1$ vertices $\hat{u}_1^i, \hat{u}_2^i, \ldots, \hat{u}_{13n+1}^i$.

We add the following edges. We add edge $\{y^i, v_0^i\}$. For every $\ell \in [m]$, every $j \in [k] \setminus \{i\}$, and every $\ell' \in [5n - 1]$ we add edge $\{a_{\ell'}^{i,j,\ell}, a_{\ell'+1}^{i,j,\ell}\}$ and we add edge $\{b_{\ell'}^{i,j,\ell}, b_{\ell'+1}^{i,j,\ell}\}$.

Let $1 \leq j < i$ (skip if $i = 1$), let $e_\ell^{j,i} \in F_{j,i}$, and let $w_{\ell'}^i \in W_i$ be incident with $e_\ell^{j,i}$. Then we add edge $\{v_{j-1}^i, a_1^{i,j,\ell}\}$ and we add edge $\{a_{5n}^{i,j,\ell}, u_{\ell'-1}^\ell\}$ between $a_{5n}^{i,j,\ell}$ and the vertex $u_{\ell'-1}^\ell$ of the edge selection gadget of color combination j, i. Furthermore, we add edge $\{v_j^i, b_1^{i,j,\ell}\}$ and edge $\{b_{5n}^{i,j,\ell}, u_{\ell'}^\ell\}$ between $b_{5n}^{i,j,\ell}$ and the vertex $u_{\ell'}^\ell$ of the edge selection gadget of color combination j, i.

We add edge $\{v_{i-1}^i, \hat{u}_1^i\}$ and for all $\ell'' \in [13n]$ we add edge $\{\hat{u}_{\ell''}^i, \hat{u}_{\ell''+1}^i\}$. Furthermore, we add edge $\{\hat{u}_{13n+1}^i, v_i^i\}$.

Let $i < j \leq k$ (skip if $i = k$), let $e_\ell^{i,j} \in F_{i,j}$, and let $w_{\ell'}^i \in W_i$ be incident with $e_\ell^{i,j}$. Then we add edge $\{v_{j-1}^i, a_1^{i,j,\ell}\}$ and edge $\{a_{5n}^{i,j,\ell}, u_{3n+\ell'-1}^\ell\}$ between $a_{5n}^{i,j,\ell}$ and the vertex $u_{3n+\ell'-1}^\ell$ of the edge selection gadget of color combination i, j. Furthermore, we add edge $\{v_j^i, b_1^{i,j,\ell}\}$

APPENDIX

and edge $\{b^{i,j,\ell}_{5n}, u^{\ell}_{3n+\ell'}\}$ between $b^{i,j,\ell}_{5n}$ and the vertex $u^{\ell}_{3n+\ell'}$ of the edge selection gadget of color combination i, j.

Connector gadget. Next, we describe *connector gadgets*. Intuitively, these gadgets will be used to connect many vertex pairs by fast paths, which will make arguing about possible labelings in YES-instances much easier. Connector gadgets consist of six vertices $\hat{v}_0, \hat{v}'_0, \hat{v}_1, \hat{v}_2, \hat{v}_3, \hat{v}'_3$. Each connector gadget is associated with two sets A, B with $B \subseteq A$ containing vertices of other gadgets. Let $V^\star$ denote the set of all vertices from all edge selection gadgets and all verification gadgets. The sets A and B will only play a role when defining the matrix D later. Informally speaking, vertices in A should reach all vertices in $V^\star$ quickly through the gadget, except the ones in B. We have the following edges.

- Edges $\{\hat{v}_0, \hat{v}_1\}, \{\hat{v}'_0, \hat{v}_1\}, \{\hat{v}_1, \hat{v}_2\}, \{\hat{v}_2, \hat{v}_3\}, \{\hat{v}_2, \hat{v}'_3\}$.
- An edge between $\hat{v}_1$ and each vertex in $V^\star$.
- An edge between $\hat{v}_2$ and each vertex in $V^\star$.

We add two connector gadgets for each edge selection gadget and two connector gadgets for each verification gadget.

The *first connector gadget for the edge selection gadget of color combination* i, j with $i < j$ has the following sets.

- Sets A and B contain all vertices in $X_{i,j}$ and vertex $v^{\star\star}_{i,j}$.

The *second connector gadget for the edge selection gadget of color combination* i, j with $i < j$ has the following sets.

- Set A contains all vertices from the edge selection gadget $G_{i,j}$ except vertices in $X_{i,j}$.
- Set B is empty.

The *first connector gadget for the verification gadget of color* i has the following sets.

- Sets A and B contain all vertices v^i_ℓ with $0 \leq \ell \leq k$.

The *second connector gadget for the verification gadget of color* i has the following sets.

- Set A contains all vertices of the verification gadget except vertices v^i_ℓ with $0 \leq \ell \leq k$.
- Set B is empty.

Alignment gadget. Lastly, we introduce an *alignment gadget*. It consists of one vertex $w^\star$ and a set of vertices $\hat{W}$ containing one vertex for each edge selection gadget, one vertex for each verification gadget, and one vertex for each connector gadget. Vertex $w^\star$ is connected to each vertex in $\hat{W}$. The vertex x_1 of each edge selection gadget, the vertex y^i of each verification gadget, and the vertex $\hat{v}_1$ of each connector gadget are each connected to one vertex in $\hat{W}$ such that all vertices in $\hat{W}$ have degree two. Intuitively, this gadget is used to relate labels of different gadgets to each other.

Feedback vertex number. This finished the description of the underlying graph G. For an illustration see Figure 3. We can observe that the vertex set containing

- vertices $v^\star_{i,j}$ and $v^{\star\star}_{i,j}$ of each edge selection gadget,
- vertices v^i_ℓ with $0 \leq \ell \leq k$ of each verification gadget,
- vertices $\hat{v}_1$ and $\hat{v}_2$ of each connector gadget, and
- vertex $w^\star$ of the alignment gadget

forms a feedback vertex set in G with size $O(k^2)$.

APPENDIX

Duration matrix D. We proceed with describing the matrix D of durations of fastest paths. For a more convenient presentation, we use the notation $d(v, v') := D_{v,v'}$. For all vertices v, v' that are neighbors in G we have that $d(v, v') = 1$ and $d(v', v) = 1$.

Next, consider a connector gadget consisting of vertices $\hat{v}_0, \hat{v}_0', \hat{v}_1, \hat{v}_2, \hat{v}_3, \hat{v}_3'$ and with sets A and B. Informally, the connector gadget makes sure that all vertices in A can reach all other vertices (of edge selection gadgets and verification gadgets) except the ones in B. We set the following durations. Recall that $V^\star$ denotes the set of all vertices from all edge selection gadgets and all verification gadgets.

- We set $d(\hat{v}_0, \hat{v}_2) = d(\hat{v}_3, \hat{v}_1) = d(\hat{v}_2, \hat{v}_0') = d(\hat{v}_1, \hat{v}_3') = 2$, and $d(\hat{v}_0, \hat{v}_0') = d(\hat{v}_3, \hat{v}_3') = d(\hat{v}_0, \hat{v}_3') = d(\hat{v}_3, \hat{v}_0') = 3$.
- Let $v \in A$, then we set $d(v, \hat{v}_0') = 3$ and $d(v, \hat{v}_3') = 3$.
- Let $v \in V^\star \setminus B$, then we set $d(\hat{v}_0, v) = 3$ and $d(\hat{v}_3, v) = 3$.
- Let $v \in A$ and $v' \in V^\star \setminus B$ such that v and v' are not neighbors, then we set $d(v, v') = 3$.

Now consider two connector gadgets, one with vertices $\hat{v}_0, \hat{v}_0', \hat{v}_1, \hat{v}_2, \hat{v}_3, \hat{v}_3'$ and with sets A and B, and one with vertices $\hat{v}_0', \hat{v}_0'', \hat{v}_1', \hat{v}_2', \hat{v}_3', \hat{v}_3''$ and with sets A' and B'.

- If there is a vertex $v \in A$ with $v \notin A'$, then we set $d(\hat{v}_1, \hat{v}_1') = 3$.
- If there is a vertex $v \in A$ with $v \in A' \setminus B'$, then we set $d(\hat{v}_1, \hat{v}_2') = 3$.
- If there is a vertex $v \in V^\star \setminus (A \setminus B)$ with $v \notin A'$, then we set $d(\hat{v}_2, \hat{v}_1') = 3$.
- If there is a vertex $v \in V^\star \setminus (A \setminus B)$ with $v \in A' \setminus B'$, then we set $d(\hat{v}_2, \hat{v}_2') = 3$.

Next, consider the edge selection gadget for color combination i, j with $i < j$.

- Let $1 \le \ell < \ell' \le m$. We set $d(x_\ell, x_{\ell'}) = 2n \cdot (i + j) \cdot ((\ell')^2 - \ell^2) + 1$.
- For all $\ell \in [m]$ we set $d(x_\ell, v_{i,j}^{\star\star}) = 8n + 5$.

Next, consider the verification gadget for color i. For all $0 \le j < j' < i$ and all $i \le j < j' \le k$ we set the following.

- We set $d(v_j^i, v_{j'}^i) = (20n + 6)(j' - j) - 1$.

For all $0 \le j < i$ and all $i \le j' \le k$ we set the following.

- We set $d(v_j^i, v_{j'}^i) = (20n + 6)(j' - j) + 6n - 1$.

Finally, we consider the alignment gadget. Let x_1 belong to the edge selection gadget of color combination i, j and let $w \in \hat{W}$ denote the neighbor of x_1 in the alignment gadget. Let $\hat{v}_1$ and $\hat{v}_2$ belong to the first connector gadget of the edge selection gadget for color combination i, j. Let $\hat{V}$ contain all vertices $\hat{v}_1$ and $\hat{v}_2$ belonging to the other connector gadgets (different from the first one of the edge selection gadget for color combination i, j).

- We set $d(w^\star, x_1) = (20n + 6)(i + j)$.
- We set $d(w^\star, \hat{v}_1) = n^9$, $d(w, \hat{v}_2) = n^9$, $d(w, \hat{v}_1) = n^9 - (20n + 6)(i + j) + 1$, and $d(w, \hat{v}_2) = n^9 - (20n + 6)(i + j) + 1$.
- For each vertex $v \in (V^\star \cup \hat{V}) \setminus (X_{i,j} \cup \{v_{i,j}^{\star\star}\})$ we set $d(w^\star, v) = n^9 + 2$ and $d(w, v) = n^9 - (20n + 6)(i + j) + 3$.

Let y^i belong to the verification gadget of color i and let $w' \in \hat{W}$ denote the neighbor of y^i in the alignment gadget. Let $\hat{v}_1$ and $\hat{v}_2$ belong to the connector gadget of the verification gadget for color i. Let $\hat{V}$ contain all vertices $\hat{v}_1$ and $\hat{v}_2$ belonging to the other connector gadgets (different from the one of the verification gadget for color i). Let V_i denote the set of all vertices of the verification gadget of color i.

- We set $d(w^\star, y^i) = n^8 - 1$, $d(w', v_0^i) = 2$, and $d(w^\star, v_0^i) = n^8$.
- We set $d(w^\star, \hat{v}_1) = n^9$, $d(w^\star, \hat{v}_2) = n^9$, $d(w', \hat{v}_1) = n^9 - n^8$, and $d(w', \hat{v}_2) = n^9 - n^8$.
- For each vertex $v \in (V^\star \cup \hat{V}) \setminus V_i$ we set $d(w^\star, v) = n^9 + 1$, $d(w, v) = n^9 - n^8 + 2$, and $d(y^i, v) = n^9 - n^8 + 2$.

APPENDIX

Let $\hat{v}_1$ belong to some connector gadget. Then we set $d(w^\star, \hat{v}_1) = n^9$.

All fastest path durations between non-adjacent vertex pairs that are not specified above are set to infinity.

Correctness. This finishes the construction of Simple periodic Temporal Graph Realization instance D, which can clearly be computed in polynomial time. For an illustration see Figure 3. As discussed earlier, we have that the vertex cover number of the underlying graph of the instance is in $O(k^2)$.

In the remainder we prove that D is a Yes-instance of Simple periodic Temporal Graph Realization if and only if the H is a Yes-instance of Multicolored Clique.

($\Rightarrow$): Assume D is a Yes-instance of Simple periodic Temporal Graph Realization and let (G, λ) be a solution. We have that the underlying graph G is uniquely defined by D. We first prove a number of properties of λ that we need to define a set of vertices in H which we claim to be a multicolored clique.

To start, consider the alignment gadget. We can observe that all edges incident with $w^\star$ have the same label.

$\triangleright$ Claim 5. For all $w \in \hat{W}$ we have that $\lambda(\{w^\star, w\}) = t$ for some $t \in \mathbb{N}$.

Proof. Assume for contradiction that there are $w, w' \in \hat{W}$ such that $\lambda(\{w^\star, w\}) = t$ and $\lambda(\{w^\star, w'\}) = t'$ with $t \neq t'$. Let w.l.o.g. $t < t'$. Then w can reach w', however we have that $d(w, w') = \infty$, a contradiction. $\triangleleft$

Claim 5 allows us to assume w.l.o.g. that all edges incident with vertex $w^\star$ of the alignment gadget have label 1. From now we will assume that this is the case.

Next, we analyse the labelings of connector gadgets. We show that all edges incident with vertices of connector gadgets have labels of at least n^9 and at most $n^9 + 2$. More precisely, we show the following.

$\triangleright$ Claim 6. Let $\hat{v}_0, \hat{v}'_0, \hat{v}_1, \hat{v}_2, \hat{v}_3, \hat{v}'_3$ be the vertices of a connector gadget with sets A and B. Then we have that $\lambda(\{\hat{v}_0, \hat{v}_1\}) = n^9$, $\lambda(\{\hat{v}'_0, \hat{v}_1\}) = n^9 + 2$, $\lambda(\{\hat{v}_1, \hat{v}_2\}) = n^9 + 1$, $\lambda(\{\hat{v}_2, \hat{v}_3\}) = n^9$, and $\lambda(\{\hat{v}_2, \hat{v}'_3\}) = n^9 + 2$. Furthermore, for all $v \in V^\star$ we have $n^9 \leq \lambda(\{\hat{v}_1, v\}) \leq n^9 + 2$ and $n^9 \leq \lambda(\{\hat{v}_2, v\}) \leq n^9 + 2$.

Proof. Let $w \in \hat{W}$ denote the vertex of the alignment gadget that is neighbor of $w^\star$ and $\hat{v}_0$. We have $d(w^\star, \hat{v}_0) = n^9$. It follows that $\lambda(\{w, \hat{v}_0\}) = n^9$. Since $d(\hat{v}_1, w) = \infty$ and $d(w, \hat{v}_1) = \infty$, we have that $\lambda(\{\hat{v}_0, \hat{v}_1\}) = n^9$. Note that $\hat{v}_1$ is the only common neighbor of $\hat{v}_0$ and $\hat{v}_2$ and the only common neighbor of $\hat{v}_0$ and $\hat{v}'_0$. Since $d(\hat{v}_0, \hat{v}_2) = 2$ and $d(\hat{v}_0, \hat{v}'_0) = 3$ we have that $\lambda(\{\hat{v}_1, \hat{v}_2\}) = n^9 + 1$ and $\lambda(\{\hat{v}'_0, \hat{v}_1\}) = n^9 + 2$. Similarly, we have that $\hat{v}_2$ is the only common neighbor of $\hat{v}_3$ and $\hat{v}_1$ and the only common neighbor of $\hat{v}_3$ and $\hat{v}'_3$. Since $d(\hat{v}_3, \hat{v}_1) = 2$ and $d(\hat{v}_3, \hat{v}'_3) = 3$ we have that $\lambda(\{\hat{v}_2, \hat{v}_3\}) = n^9$ and $\lambda(\{\hat{v}_2, \hat{v}'_3\}) = n^9 + 2$.

Let $v \in V^\star$. Note that $d(v, \hat{v}_0) = \infty$ and $d(v, \hat{v}_3) = \infty$. It follows that $\lambda(\{\hat{v}_1, v\}) \geq n^9$ and $\lambda(\{\hat{v}_2, v\}) \geq n^9$. Otherwise, there would be a temporal path from v to $\hat{v}_0$ via $\hat{v}_1$ or a temporal path from v to $\hat{v}_3$ via $\hat{v}_2$, a contradiction. Furthermore, note that $d(\hat{v}'_0, v) = \infty$ and $d(\hat{v}'_3, v) = \infty$. It follows that $\lambda(\{\hat{v}_1, v\}) \leq n^9 + 2$ and $\lambda(\{\hat{v}_2, v\}) \leq n^9 + 2$. Otherwise, there would be a temporal path from $\hat{v}'_0$ to v via $\hat{v}_1$ or a temporal path from $\hat{v}_3$ to v via $\hat{v}_2$, a contradiction. $\triangleleft$

Now we take a closer look at the edge selection gadgets. We make a number of observations that will allow us to define a set of vertices in H that we claim to be a multicolored clique.

APPENDIX

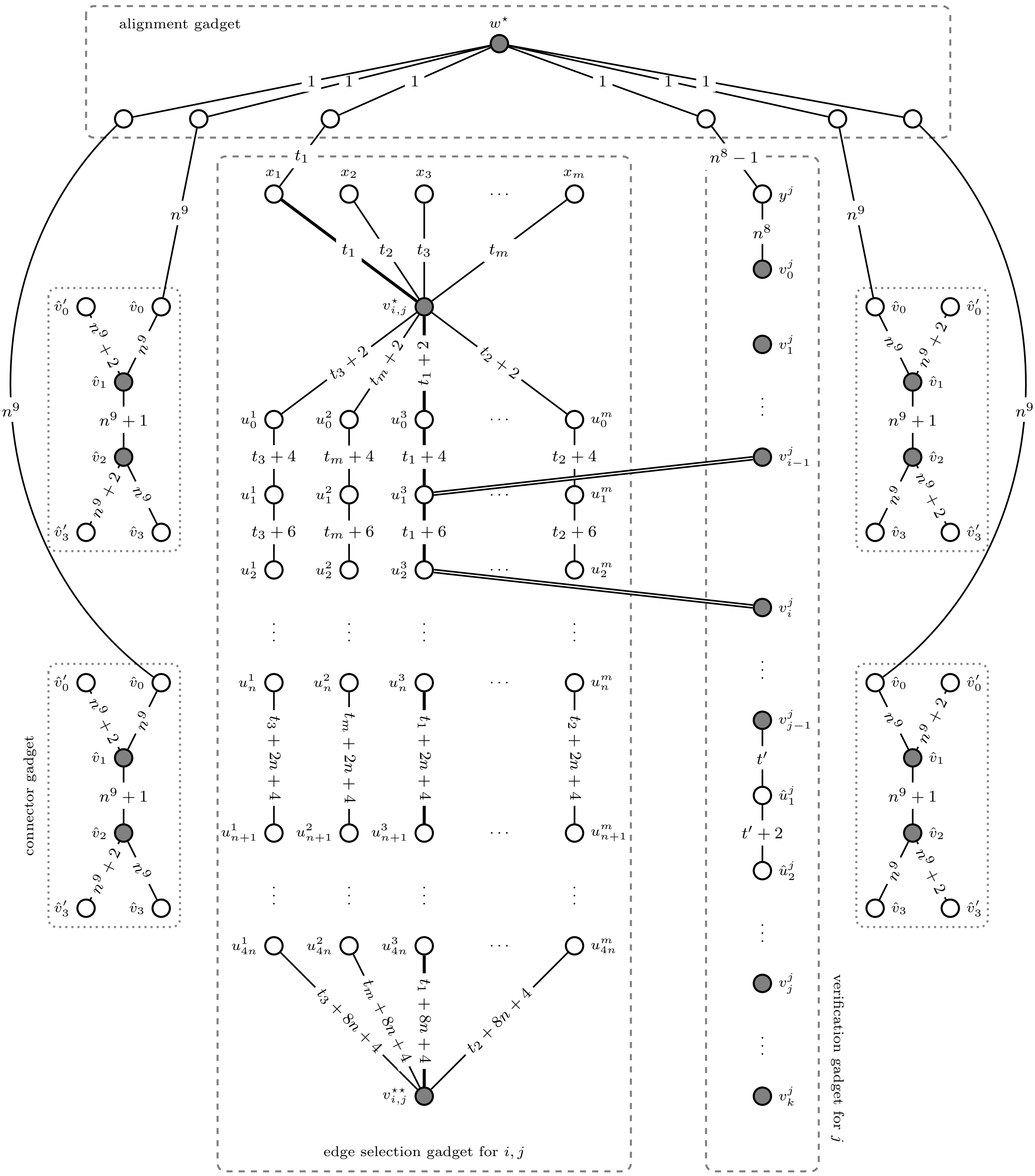


Figure 3 Illustration of part of the underlying graph G and a possible labeling. Edges incident with vertices $\hat{v}_1, \hat{v}_2$ of connector gadgets are omitted. Gray vertices form a feedback vertex set. The double line connections, between a vertex v_{i-1}^j in the verification gadget, and u_1^3 in the edge selection gadget, and, between a vertex u_2^3 in the edge selection gadget, and v_i^j in the verification gadget, consist of $5n$ vertices $a_1^{j,i,3}, a_2^{j,i,3}, \ldots, a_{5n}^{j,i,3}$ and $b_1^{j,i,3}, b_2^{j,i,3}, \ldots, b_{5n}^{j,i,3}$, respectively.

APPENDIX

▷ Claim 7. For all $1 \le i < j \le k$ and $\ell \in [m]$ we have that $\lambda(\{u^\ell_{4n}, v^{\star\star}_{i,j}\}) \le n^9 + 2$, where u^ℓ_{4n} belongs to the edge selection gadget for i, j.

Proof. Consider the first connector gadget of the edge selection gadget for i, j with vertices $\hat{v}_0, \hat{v}'_0, \hat{v}_1, \hat{v}_2, \hat{v}_3, \hat{v}'_3$ and sets A, B. Recall that $v^{\star\star}_{i,j} \in B$ and hence we have that $d(\hat{v}_0, v^{\star\star}_{i,j}) = \infty$. Furthermore, we have that $u^\ell_{4n} \notin B$ and hence $d(\hat{v}_0, u^\ell_{4n}) = 3$. By Claim 6 and the fact that $d(w^\star, \hat{v}_0) = n^9$ we have that both edges incident with $\hat{v}_0$ have label n^9. It follows that a fastest temporal path from $\hat{v}_0$ to u^ℓ_{4n} arrives at u^ℓ_{4n} at time $n^9 + 2$. Now assume for contradiction that $\lambda(\{u^\ell_{4n}, v^{\star\star}_{i,j}\}) > n^9 + 2$. Then there exists a temporal walk from $\hat{v}_0$ to $v^{\star\star}_{i,j}$ via u^ℓ_{4n}, a contradiction to $d(\hat{v}_0, v^{\star\star}_{i,j}) = \infty$. ◁

▷ Claim 8. For all $1 \le i < j \le k$ and $\ell \in [m]$ we have that $\lambda(\{x_\ell, v^\star_{i,j}\}) = (i + j) \cdot (2n\ell^2 + 18n + 6)$, where x_ℓ belongs to the edge selection gadget for i, j.

Proof. We first determine the label of $\{x_1, v^\star_{i,j}\}$, where x_1 belongs to the edge selection gadget for i, j. Note that x_1 is connected to the alignment gadget. Let $w \in \hat{W}$ be the vertex of the alignment gadget that is a neighbor of x_1. Since $d(w^\star, x_1) = (20n + 6)(i + j)$ we have that $\lambda(\{w, x_1\}) = (20n + 6)(i + j)$.

First, assume that $\lambda(\{x_1, v^\star_{i,j}\}) < (20n + 6)(i + j)$. Then there is a temporal path from $v^\star_{i,j}$ to w via x_1. However, we have that $d(x_{i,j^\star}, w) = \infty$, a contradiction. Next, assume that $(20n + 6)(i + j) < \lambda(\{x_1, v^\star_{i,j}\}) < n^9 + 2$. Then there is a temporal path from w to $v_{i,j}$ via x_1 with duration strictly less than $n^9 - (20n + 6)(i + j) + 3$. However, we have that $d(w, v^\star_{i,j}) = n^9 - (20n + 6)(i + j) + 3$, a contradiction. Finally, assume that $\lambda(\{x_1, v^\star_{i,j}\}) \ge n^9 + 2$. Consider a fastest temporal path from x_1 to $v^{\star\star}_{i,j}$. This temporal path cannot visit w as its first vertex, since from there it cannot continue. From this assumption and Claim 6 it follows, that the first edge of the temporal path has a label with value at least n^9. However, by Claims 6 and 7 we have that all edges incident with $v^{\star\star}_{i,j}$ have a label with value at most $n^9 + 2$. It follows that $d(x_1, v^{\star\star}_{i,j}) \le 3$, a contradiction.

We can conclude that $\lambda(\{x_1, v^\star_{i,j}\}) = (20n + 6)(i + j)$. Now let $1 < \ell \le m$. We have that $d(x_1, x_\ell) = 2n \cdot (i + j) \cdot (\ell^2 - 1) + 1$ which implies that $\lambda(\{x_\ell, v^\star_{i,j}\}) \ge (i + j) \cdot (2n\ell^2 + 18n + 6)$. Assume that $(i + j) \cdot (2n\ell^2 + 18n + 6) < \lambda(\{x_\ell, v^\star_{i,j}\}) \le n^9 + 2$. Then the temporal path from x_1 to x_ℓ via $v^\star_{i,j}$ is not a fastest temporal path from x_1 to x_ℓ. Again, we have that a fastest temporal path from x_1 to x_ℓ cannot visit w as its first vertex, since from there it cannot continue. By Claim 6, all other edges incident with x_1 (that is, all different from the one to $v^\star_{i,j}$ and the one to w) have a label of at least n^9 and at most $n^9 + 2$. Similarly, by Claim 6 we have that all other edges incident with x_ℓ (that is, all different from the one to $v^\star_{i,j}$) have a label of at least n^9 and at most $n^9 + 2$. It follows that any temporal path from x_1 to x_ℓ that visits $v^\star_{i,j}$ as its first vertex has a duration strictly larger than $2n \cdot (i + j) \cdot (\ell^2 - 1) + 1$. Any temporal path from x_1 to x_ℓ that visits a vertex different from $v^\star_{i,j}$ as its first vertex has duration of at most 3. In both cases we have a contradiction. Lastly, assume that $\lambda(\{x_\ell, v^\star_{i,j}\}) > n^9 + 2$. Consider a fastest temporal path from x_ℓ to $v^{\star\star}_{i,j}$. Now this temporal path has duration at most 3 since by Claim 6 and the just made assumption all edges incident with x_ℓ have label at least n^9 whereas by Claims 6 and 7 all edges incident with $v^{\star\star}_{i,j}$ have label at most $n^9 + 2$, a contradiction. ◁

▷ Claim 9. For all $1 \le i < j \le k$ there exist a permutation $\sigma_{i,j} : [m] \to [m]$ such that for all $\ell \in [m]$ we have that $\lambda(\{u^\ell_{4n}, v^{\star\star}_{i,j}\}) = (i + j) \cdot (2n \cdot (\sigma_{i,j}(\ell))^2 + 18n + 6) + 8n + 4$, where u^ℓ_{4n} belongs to the edge selection gadget for i, j.

Furthermore, a fastest temporal path from x_ℓ (of the edge selection gadget for i, j) to $v^{\star\star}_{i,j}$ visits $v^\star_{i,j}$ as its second vertex, and $u^{\ell'}_{4n}$ with $\sigma_{i,j}(\ell') = \ell$ (of the edge selection gadget for i, j) as its second last vertex.

APPENDIX

Proof. For every $\ell \in [m]$ we have that $d(x_\ell, v^{\star\star}_{i,j}) = 8n + 5$, where x_ℓ belongs to the edge selection gadget for i, j. From Claims 6 and 8 follows that all edges incident with x_ℓ have a label of at least n^9 except the one to $v^\star_{i,j}$ and, if $\ell = 1$, the edge connecting x_1 to the alignment gadget. In the latter case, no temporal path from x_1 from $v^{\star\star}_{i,j}$ can continue to the neighbor of x_1 in the alignment gadget, since it cannot continue from there.

Now consider $v^{\star\star}_{i,j}$. By Claims 6 and 7 we have that all edges incident with $v^{\star\star}_{i,j}$ have a label of at most $n^9 + 2$. It follows that a fastest temporal path P from x_ℓ to $v^{\star\star}_{i,j}$ has to visit $v^\star_{i,j}$ after x_ℓ, since otherwise we have $d(x_\ell, v^{\star\star}_{i,j}) \leq 2$, a contradiction.

Furthermore, we have by Claim 6 that all edges incident with $v^{\star\star}_{i,j}$ have a label of at least n^9 except the ones incident to $u^{2n}_{\ell'}$ for $\ell' \in [m]$. By Claim 8 we have that $\lambda(\{x_\ell, v^\star_{i,j}\}) \leq 4n^6$. It follows that a fastest temporal path from x_ℓ to $v^{\star\star}_{i,j}$ has to visit $u^{\ell'}_{4n}$ for some $\ell' \in [m]$ as its second last vertex. Otherwise, we have $d(x_\ell, v^{\star\star}_{i,j}) > 8n + 5$ (for sufficiently large n), a contradiction.

We can conclude that a fastest temporal path from x_ℓ to $v^{\star\star}_{i,j}$ has to visit $v^\star_{i,j}$ as its second vertex and $u^{\ell'}_{4n}$ for some $\ell' \in [m]$ as its second last vertex. Recall that in a temporal path, the difference between the labels of the first and last edge determine its duration (minus one). Hence, we have that $\lambda(\{u^{\ell'}_{4n}, v^{\star\star}_{i,j}\}) - \lambda(\{x_\ell, v^\star_{i,j}\}) + 1 = 8n + 5$. By Claim 8 we have that $\lambda(\{x_\ell, v^\star_{i,j}\}) = (i + j) \cdot (2n\ell^2 + 18n + 2)$. It follows that $\lambda(\{u^{\ell'}_{4n}, v^{\star\star}_{i,j}\}) = (i + j) \cdot (2n\ell^2 + 18n + 6) + 8n + 4$. We set $\sigma_{i,j}(\ell') = \ell$.

Finally, we show that $\sigma_{i,j}$ is a permutation on $[m]$. Assume for contradiction that there are $\ell, \ell' \in [m]$ with $\ell \neq \ell'$ such that $\sigma_{i,j}(\ell) = \sigma_{i,j}(\ell')$. Then we have that $\lambda(\{u^\ell_{4n}, v^{\star\star}_{i,j}\}) = \lambda(\{u^{\ell'}_{4n}, v^{\star\star}_{i,j}\})$. However, by Claim 8 we have that all edges from $v^\star_{i,j}$ to a vertex in $X_{i,j}$ have distinct labels. Furthermore, we argued above that every fastest path from a vertex in $X_{i,j}$ to $v^{\star\star}_{i,j}$ visits $v^\star_{i,j}$ as its second vertex and a vertex from the set $\{u^{\ell''}_{4n} : \ell'' \in [m]\}$ as its second last vertex. Since for all $x_{\ell''}$ with $\ell'' \in [m]$ we have that $d(x_{\ell''}, v^{\star\star}_{i,j}) = 8n + 5$, we must have that all edges from vertices in $\{u^{\ell''}_{4n} : \ell'' \in [m]\}$ to $v^{\star\star}_{i,j}$ must have distinct labels. Hence, we have a contradiction and can conclude that $\sigma_{i,j}$ is indeed a permutation. ◁

For all $1 \leq i < j \leq k$, let $\sigma_{i,j}$ be the permutation on $[m]$ as defined in Claim 9. We call $\sigma_{i,j}$ the *permutation of color combination* i, j. Now we have enough information to define a set of vertices of H that form a multicolored clique. To this end, consider the following set X of edges from H.

$$X = \{e^{i,j}_\ell \in F_{i,j} : \sigma_{i,j}(\ell) = 1\}$$

We claim that $\bigcup_{e \in X} e$ forms a multicolored clique in H. From now on, denote $\{e_{i,j}\} = X \cap F_{i,j}$. We show that for all $i \in [k]$ we have that $|(\bigcap_{1 \leq j < i} e_{j,i}) \cap (\bigcap_{i < j \leq k} e_{i,j})| = 1$, that is, for every color i, all edges of a color combination involving i have the same vertex of color i as endpoint. This implies that $\bigcup_{e \in X} e$ is a multicolored clique in H.

Before we proceed, we show some further properties of λ. First, let us focus on the labels on edges of the edge selection gadgets.

▷ Claim 10. For all $1 \leq i < j \leq k$, $\ell \in [m]$, and $\ell' \in [4n]$ we have that $\lambda(\{u^\ell_{\ell'-1}, u^\ell_{\ell'}\}) = (i + j) \cdot (2n \cdot (\sigma_{i,j}(\ell))^2 + 18n + 6) + 2\ell' + 2$, where $u^\ell_{\ell'-1}$ and $u^\ell_{\ell'}$ belong to the edge selection gadget for i, j and $\sigma_{i,j}$ is the permutation of color combination i, j.

Proof. Let $1 \leq i < j \leq k$ and $\ell \in [m]$. By Claim 9 we know that a fastest temporal path from $x_{\sigma_{i,j}(\ell)}$ (of the edge selection gadget for i, j) to $v^{\star\star}_{i,j}$ visits $v^\star_{i,j}$ as its second vertex, and u^ℓ_{4n} (of the edge selection gadget for i, j) as its second last vertex. Furthermore, by Claim 8 we have that $\lambda(\{x_{\sigma_{i,j}(\ell)}, v^\star_{i,j}\}) = (i + j) \cdot (2n \cdot (\sigma_{i,j}(\ell))^2 + 18n + 2)$ and by Claim 9 we have

APPENDIX

that $\lambda(\{u^\ell_{4n}, v^{\star\star}_{i,j}\}) = (i+j)\cdot(2n\cdot(\sigma_{i,j}(\ell))^2+18n+2)+8n+4$. It follows that there exist a temporal path P from $v^\star_{i,j}$ to u^ℓ_{4n} that starts at $v^\star_{i,j}$ later than $(i+j)\cdot(2n\cdot(\sigma_{i,j}(\ell))^2+18n+6)$ and arrives at u^ℓ_{4n} earlier than $(i+j)\cdot(2n\cdot(\sigma_{i,j}(\ell))^2+18n+6)+8n+4$. Hence, the temporal path P has duration at most $8n+3$.

We investigate the temporal path P from its destination u^ℓ_{4n} back to its start vertex $v^\star_{i,j}$. Consider the neighbors of u^ℓ_{4n} that are different from $v^{\star\star}_{i,j}$. By Claim 6 we have that all edges from u^ℓ_{4n} to neighbors of u^ℓ_{4n} that are vertices of connector gadgets have a label of at least n^9. Hence, P does not visit any of those neighbors. Next, consider neighbors of u^ℓ_{4n} in verification gadgets. Assume u^ℓ_{4n} has a neighbor in the verification gadget of color i' for some $i'\in[k]$. Then this neighbor is vertex $b^{i',j,\ell}_{5n}$. Note that if P visits $b^{i',j,\ell}_{5n}$, then it also visits all of $\{b^{i',j,\ell}_{\ell'} : \ell'\in[5n]\}$, since all these vertices have degree two. Now consider the second connector gadget of a verification gadget i' with sets A,B, we have that all vertices $\{b^{i',j,\ell}_{\ell'} : \ell'\in[5n]\}$ are contained in A and are not contained in B. Hence, we have that all non-adjacent pairs of vertices in $\{b^{i',j,\ell}_{\ell'} : \ell'\in[5n]\}$ are on duration 3 apart, according to D, and that $|\lambda(\{b^{i',j,\ell}_{\ell'}, b^{i',j,\ell}_{\ell'+1}\}) - \lambda(\{b^{i',j,\ell}_{\ell'+1}, b^{i',j,\ell}_{\ell'+2}\})| \geq 2$ for all $\ell'\in[5n-2]$. It follows that P would have a duration larger than $8n+3$. We can conclude that P does not visit $b^{i',j,\ell}_{5n}$. It follows that P visits u^ℓ_{4n-1}. Here, we can make an analogous investigation. Additionally, we have to consider the case that P visits a neighbor of u^ℓ_{4n-1} in verification gadget of color i' for some $i'\in[k]$ that is vertex $a^{i',j,\ell}_{5n}$. However, we can exclude this by a similar argument as above.

By repeating the above arguments, we can conclude that P visits (exactly) all vertices in $\{u^\ell_{\ell'} : 0\leq \ell'\leq 4n\}$ and $v^\star_{i,j}$. Consider the second connector gadget of the edge selection gadget of i,j with set A and B. Note that all vertices visited by P are contained in $A\setminus B$. It follows that all pairs of non-adjacent vertices visited by P are on duration 3 apart, according to D. In particular, we have $d(u^\ell_{\ell'-1}, u^\ell_{\ell'+1}) = 3$ for all $\ell'\in[4n-1]$ and $d(v^\star_{i,j}, u^\ell_1) = 3$. If follows that for every $\ell'\in[4n-1]$ we have that $\lambda(\{u^\ell_{\ell'}, u^\ell_{\ell'+1}\}) - \lambda(\{u^\ell_{\ell'-1}, u^\ell_{\ell'}\}) \geq 2$ and $\lambda(\{u^\ell_1, u^\ell_2\}) - \lambda(\{v^\star_{i,j}, u^\ell_1\}) \geq 2$.

By investigating the sets A,B of the first connector gadget of the edge selection gadget of i,j, we get that $d(x_{\sigma_{i,j}(\ell)}, u^\ell_1) = 3$ and hence $\lambda(\{v^\star_{i,j}, u^\ell_1\}) - \lambda(\{x_{\sigma_{i,j}(\ell)}, v^\star_{i,j}\}) \geq 2$. Furthermore, we get that $d(u^\ell_{4n-1}, v^{\star\star}_{i,j}) = 3$ and hence $\lambda(\{v^{\star\star}_{i,j}, u^\ell_{4n}\}) - \lambda(\{u^\ell_{4n-1}, u^\ell_{4n}\}) \geq 2$. Considering that P visits $4n+2$ vertices, we have that all mentioned inequalities of differences of labels have to be equalities, otherwise P has a duration larger than $8n+3$ or we have that $\lambda(\{v^\star_{i,j}, u^\ell_1\}) - \lambda(\{x_{\sigma_{i,j}(\ell)}, v^\star_{i,j}\}) < 2$ or $\lambda(\{v^{\star\star}_{i,j}, u^\ell_{4n}\}) - \lambda(\{u^\ell_{4n-1}, u^\ell_{4n}\}) < 2$. Since by Claims 8 and 9 the labels $\lambda(\{x_{\sigma_{i,j}(\ell)}, v^\star_{i,j}\})$ and $\lambda(\{v^{\star\star}_{i,j}, u^\ell_{4n}\})$ are determined, then also all labels of edges traversed by P are determined and the claim follows. $\triangleleft$

Next, we investigate the labels of the verification gadgets.

$\triangleright$ Claim 11. For all $i\in[k]$ we have that $\lambda(\{y^i, v^i_0\}) = n^8$.

Proof. Let $w\in\hat{W}$ denote the neighbor of y^i in the alignment gadget. Note that we have $d(w^\star, y^i) = n^8-1$. It follows that $\lambda(\{w, y^i\}) = n^8-1$. Furthermore, we have that $d(w, v^i_0) = 2$ and note that y^i has degree 2. It follows that $\lambda(\{y^i, v^i_0\}) = n^8$. $\triangleleft$

$\triangleright$ Claim 12. For all $1 < i \leq k$ and all $\ell\in[m]$ we have that $\lambda(\{v^i_0, a^{i,1,\ell}_1\}) \leq n^8$ or $\lambda(\{v^i_0, a^{i,1,\ell}_1\}) \geq n^9+2$. For $i=1$ we have that $\lambda(\{v^i_0, \hat{u}^i_1\}) \leq n^8$ or $\lambda(\{v^i_0, \hat{u}^i_1\}) \geq n^9+2$.

Proof. Let $1 < i \leq k$ and $\ell\in[m]$. Assume that $n^8 < \lambda(\{v^i_0, a^{i,1,\ell}_1\}) < n^9+2$. Then, since by Claim 11 we have $\lambda(\{y^i, v^i_0\}) = n^8$, there is a temporal path from $w^\star$ to $a^{i,1,\ell}_1$ via v^i_0

that arrives at $a_1^{i,1,\ell}$ strictly earlier than n^9+2. However, we have $d(w^\star, a_1^{i,1,\ell}) = n^9+2$, a contradiction. The argument for case where $i=1$ is analogous. ◁

▷ Claim 13. For all $1 \leq i < k$ and all $\ell \in [m]$ we have that $\lambda(\{v_k^i, b_1^{i,k,\ell}\}) \leq n^9+2$. For $i=k$ we have that $\lambda(\{v_k^i, \hat{u}_{13n+1}^i\}) \leq n^9+2$.

Proof. Let $1 \leq i < k$ and $\ell \in [m]$. Consider the first connector gadget of verification gadget for color i with vertices $\hat{v}_0, \hat{v}'_0, \hat{v}_1, \hat{v}_2, \hat{v}_3, \hat{v}'_3$ and sets A, B. Recall that $v_k^i \in B$ and hence we have that $d(\hat{v}_0, v_k^i) = \infty$. Furthermore, we have that $b_1^{i,k,\ell} \notin B$ and hence $d(\hat{v}_0, b_1^{i,k,\ell}) = 3$. By Claim 6 and the fact that $d(w^\star, \hat{v}_0) = n^9$ we have that both edges incident with $\hat{v}_0$ have label n^9. It follows that a fastest temporal path from $\hat{v}_0$ to $b_1^{i,k,\ell}$ arrives at $b_1^{i,k,\ell}$ at time n^9+2. Now assume for contradiction that $\lambda(\{v_k^i, b_1^{i,k,\ell}\}) > n^9+2$. Then there exists a temporal walk from $\hat{v}_0$ to v_k^i via $b_1^{i,k,\ell}$, a contradiction to $d(\hat{v}_0, v_k^i) = \infty$. The argument for case where $i=k$ is analogous. ◁

Now we are ready to prove for all $i \in [k]$ that $|(\bigcap_{1\leq j<i} e_{j,i}) \cap (\bigcap_{i<j\leq k} e_{i,j})| = 1$. Assume for contradiction that for some color $i \in [k]$ we have that $|(\bigcap_{1\leq j<i} e_{j,i}) \cap (\bigcap_{i<j\leq k} e_{i,j})| \neq 1$. Consider the verification gadget for color i. Recall that $d(v_0^i, v_k^i) = k(20n+6)+6n-1$. Let P be a fastest temporal path from v_0^i to v_k^i. We first argue that P cannot visit any vertex of a connector gadget or the alignment gadget.

▷ Claim 14. Let $i \in [k]$. Let P be a fastest temporal path from v_0^i to v_k^i. Then P does not visit any vertex of a connector gadget.

Proof. Assume for contradiction that P visits a vertex of a connector gadget. Then by Claim 6 we have that the arrival time of P is at least n^9. By Claim 6 and Claim 13 we have that the arrival time of P is at most n^9+2. This means that the second vertex visited by P cannot be a vertex from a connector gadget, because by Claim 6 this would imply $d(v_0^i, v_k^i) \leq 2$. Now we can deduce with Claim 12 that P must have a starting time of at most n^8. It follows that the arrival time of P must be smaller than n^9, a contradiction to the assumption that P visits a vertex of a connector gadget. ◁

▷ Claim 15. Let $i \in [k]$. Let P be a fastest temporal path from v_0^i to v_k^i. Then P does not visit any vertex of the alignment gadget.

Proof. Note that P starts outside the alignment gadget. This means that if P visits a vertex of the alignment gadget, then the first vertex of the alignment gadget visited by P is a neighbor of $w^\star$. However, these vertices have degree two and the edge to $w^\star$ has label one. It follows that P cannot continue from the vertex of the alignment gadget, a contradiction. ◁

It follows that the second vertex visited by P is a vertex $a_1^{i,1,\ell}$ for some $\ell \in [m]$ or vertex $\hat{u}_1^i$ if $i=1$. In the former case, P has to follow the path segment consisting of vertices in $\{a_{\ell'}^{i,1,\ell} : \ell' \in [5n]\}$ until it reaches the edge selection gadget of color combination $1, i$. From there it can reach vertex v_1^i by traversing some path segment consisting of vertices $\{b_{\ell''}^{i,1,\ell'} : \ell'' \in [5n]\}$ for some $\ell' \in [m]$. Alternatively, it can reach vertex v_{i-1}^1 or v_i^1 by traversing some path segment consisting of vertices $\{a_{\ell''}^{1,i,\ell'} : \ell'' \in [5n]\}$ for some $\ell' \in [m]$ or $\{b_{\ell''}^{1,i,\ell'} : \ell'' \in [5n]\}$ for some $\ell' \in [m]$, respectively. In the latter case ($i=1$), the temporal path P has to follow the path segment consisting of vertices in $\{\hat{u}_\ell^i : \ell \in [13n+1]\}$ until it reaches v_1^i. More generally, we can make the following observation.

APPENDIX

▷ Claim 16. Let $i, j \in \{0, 1, \dots, k\}$. Let P be a temporal path starting at v^i_j and visiting at most $13n + 1$ vertices and no vertex of a connector gadget or the alignment gadget. Then P cannot visit vertices in $\{v^{i'}_{j'} : i', j' \in \{0, 1, \dots, k\}\} \setminus \{v^i_{j-1}, v^i_j, v^i_{j+1}, v^j_{i-1}, v^j_i, v^{j+1}_{i-1}, v^{j+1}_i\}$.

Proof. Consider the edge selection gadget of color combination i', j' for some $i', j' \in [k]$ and let $u^\ell_{\ell'}$ be a vertex of that gadget. Disregarding connections via connector gadgets and the alignment gadget, we have that $u^\ell_{\ell'}$ is (potentially) connected to the verification gadget for color i' and the verification gadget of color j'. More specifically, by construction of G, we have that $u^\ell_{\ell'}$ is potentially connected to

- vertex $v^{i'}_{j'-1}$ by a path along vertices $\{a^{i',j',\ell}_{\ell''} : \ell'' \in [5n]\}$,
- vertex $v^{i'}_{j'}$ by a path along vertices $\{b^{i',j',\ell}_{\ell''} : \ell'' \in [5n]\}$,
- vertex $v^{j'}_{i'-1}$ by a path along vertices $\{a^{j',i',\ell}_{\ell''} : \ell'' \in [5n]\}$, and
- vertex $v^{j'}_{i'}$ by a path along vertices $\{b^{j',i',\ell}_{\ell''} : \ell'' \in [5n]\}$.

Furthermore, by construction of G, we have that the duration of a fastest path from $u^\ell_{\ell'}$ to any $v^{i''}_{j''}$ with $i'', j'' \in \{0, 1, \dots, k\}$ not mentioned above is at least $10n$ (disregarding edges incident with connector gadgets or the alignment gadget).

Now consider v^i_j and assume $i < j$ ($i > j$). This vertex is (if $j \neq i - 1$ and $j \neq k$) connected to some vertex $u^\ell_{\ell'}$ in the edge selection gadget for color combination $i, j + 1$ ($j + 1, i$) via a path along vertices $\{a^{i,j,\ell}_{\ell''} : \ell'' \in [5n]\}$. Furthermore, v^i_j is (if $j \neq 0$ and $j \neq i$) connected to some vertex $u^{\ell'}_{\ell'''}$ in the edge selection gadget for color combination i, j (j, i) via a path along vertices $\{b^{i,j,\ell'}_{\ell''} : \ell'' \in [5n]\}$.

We can conclude that v^i_j can reach a vertex $u^\ell_{\ell'}$ of the edge selection gadget for $i, j + 1$ (or $j + 1, i$) and a vertex $u^{\ell'}_{\ell'''}$ of the edge selection gadget for color combination i, j (or j, i), each along paths of length at least $5n$. From $u^\ell_{\ell'}$ and $u^{\ell'}_{\ell'''}$ we have that any other vertex of the edge selection gadget for $i, j + 1$ (or $j + 1, i$) and the edge selection gadget for color combination i, j (or j, i), respectively, can be reached by a path of length at most $3n$. Together with the observation made in the beginning of the proof, we can conclude that v^i_j can potentially reach any vertex in $\{v^i_{j-1}, v^i_j, v^i_{j+1}, v^j_{i-1}, v^j_i, v^{j+1}_{i-1}, v^{j+1}_i\}$ by a path that visits at most $13n + 1$ vertices.

Lastly, consider the case that $j = i - 1$ or $j = i$. Then we have that v^i_{i-1} and v^i_i are connected via a path inside the verification gadget for color i, visiting the $13n + 1$ vertices in $\{\hat{u}^i_\ell : \ell \in [13n + 1]\}$. The claim follows. ◁

Furthermore, we can make the following observation on the duration of the temporal paths characterized in Claim 16.

▷ Claim 17. Let $i, j \in \{0, 1, \dots, k\}$. Let P be a temporal path from v^i_j to a vertex in $\{v^i_{j-1}, v^i_j, v^i_{j+1}, v^j_{i-1}, v^j_i, v^{j+1}_{i-1}, v^{j+1}_i\}$ and visiting no vertex of a connector gadget or the alignment gadget. Then P has duration at least $20n$.

Proof. As argued in the proof of Claim 16, a temporal path P from v^i_j to a vertex in $\{v^i_{j-1}, v^i_j, v^i_{j+1}, v^j_{i-1}, v^j_i, v^{j+1}_{i-1}, v^{j+1}_i\}$ has to either traverse two segments of $5n$ vertices in $\{a^{i',j',\ell}_{\ell'} : \ell' \in [5n]\}$ or $\{b^{i',j',\ell}_{\ell'} : \ell' \in [5n]\}$ for some $\ell \in [m]$ and $i', j' \in \{i - 1, i, j, j + 1\}$ or a segment of the $13n + 1$ vertices in $\{\hat{u}^i_\ell : \ell \in [13n + 1]\}$. We analyse the former case first.

Consider the second connector gadget of a verification gadget i' with sets A, B, we have that all vertices $\{a^{i',j',\ell}_{\ell'} : \ell' \in [5n], j' \in [k] \setminus \{i'\}\} \cup \{b^{i',j',\ell}_{\ell'} : \ell' \in [5n], j' \in [k] \setminus \{i'\}\}$ are contained in A and are not contained in B. It follows that all non-adjacent pairs of vertices in $\{a^{i',j',\ell}_{\ell'} : \ell' \in [5n], j' \in [k] \setminus \{i'\}\} \cup \{b^{i',j',\ell}_{\ell'} : \ell' \in [5n], j' \in [k] \setminus \{i'\}\}$ are on duration 3 apart, according to D. It follows that $|\lambda(\{a^{i',j',\ell}_{\ell'}, a^{i',j',\ell}_{\ell'+1}\}) - \lambda(\{a^{i',j',\ell}_{\ell'+1}, a^{i',j',\ell}_{\ell'+2}\})| \geq 2$ for all $\ell' \in [5n-2]$

APPENDIX

and $j' \in [k] \setminus \{i'\}$. Analogously, we have that $|\lambda(\{b^{i',j',\ell}_{\ell'}, b^{i',j',\ell}_{\ell'+1}\}) - \lambda(\{b^{i',j',\ell}_{\ell'+1}, b^{i',j',\ell}_{\ell'+2}\})| \geq 2$ for all $\ell' \in [5n-2]$ and $j' \in [k] \setminus \{i'\}$. It follows that two segments of $5n$ vertices in $\{a^{i',j',\ell}_{\ell'} : \ell' \in [5n]\}$ or $\{b^{i',j',\ell}_{\ell'} : \ell' \in [5n]\}$ for some $\ell \in [m]$ and $i', j' \in \{i-1, i, j, j+1\}$ traversed by P both have duration $10n$ and hence P has duration at least $20n$.

In the latter case, where P traverses a segment of the $13n+1$ vertices in $\{\hat{u}^i_\ell : \ell \in [13n+1]\}$, we can make an analogous argument, since all vertices in $\{\hat{u}^i_\ell : \ell \in [13n+1]\}$ are contained in the set A of the second connector gadget of the verification gadget of color i but not in the set B of that connector gadget. $\lhd$

Recall that P denotes a fastest temporal path from v^i_0 to v^i_k and that $d(v^i_0, v^i_k) = k(20n+6) + 6n - 1$. By Claims 14–16 he have that P needs to visit at least one vertex in $\{v^{i'}_{j'} : i', j' \in \{0, 1, \ldots, k\}\} \setminus \{v^i_0, v^i_k\}$. Next, we analyse which vertices in this set are visited by P.

$\rhd$ Claim 18. Let $i \in [k]$. Let P be a fastest temporal path from v^i_0 to v^i_k. Then P visits all vertices in $\{v^j_i : 0 \leq j \leq k\}$ and no vertex in $\{v^{i'}_{j'} : i', j' \in \{0, 1, \ldots, k\}\} \setminus \{v^j_i : 0 \leq j \leq k\}$. Furthermore, P visits the vertices in order $v^i_0, v^i_1, v^i_2, \ldots, v^i_{k-1}, v^i_k$.

Proof. Let $X \subseteq \{v^{i'}_{j'} : i', j' \in \{0, 1, \ldots, k\}\}$ denote the set of vertices in $\{v^{i'}_{j'} : i', j' \in \{0, 1, \ldots, k\}\}$ that are visited by P. By Claims 16 and 17 we have that $|X| \leq k+1$, since otherwise the duration of P would be at least $20n(k+1) > k(20n+6)+6n-1$, a contradiction.

To prove the claim, we use the notion of a *potential p^i with respect to i* of a vertex $v^{i'}_j$. We say that the first potential of vertex $v^{i'}_j$ with respect to i is $p^i(v^{i'}_j) = i' + j - i$. The temporal path P starts at vertex v^i_0 with $p^i(v^i_0) = 0$, and ends at vertex v^i_k with $p^i(v^i_k) = k$.

Assume the path P is at some vertex $v^{i'}_j$ with $p^i_1(v^{i'}_j) = i' + j - i$. By Claim 16 we have that the next vertex in $\{v^{i'}_{j'} : i', j' \in \{0, 1, \ldots, k\}\}$ visited by P is some $v^{i''}_{j'} \in \{v^{i'}_{j-1}, v^{i'}_j, v^{i'}_{j+1}, v^j_{i'-1}, v^j_i, v^{j+1}_{i'-1}, v^{j+1}_{i'}\}$. We can observe that $|p^i(v^{i'}_j) - p^i(v^{i''}_{j'})| \leq 1$, that is, the first potential changes at most by one when P goes from one vertex in $\{v^{i'}_{j'} : i', j' \in \{0, 1, \ldots, k\}\}$ to the next one. Since $|X| \leq k+1$ we and $p^i(v^i_k) - p^i(v^i_0) = k$ have that the potential has to increase by exactly one every time P goes from one vertex in $\{v^{i'}_{j'} : i', j' \in \{0, 1, \ldots, k\}\}$ to the next one. We can conclude that $|X| = k+1$. Furthermore, we have that if the path P is at some vertex $v^{i'}_j$, the next vertex in $\{v^{i'}_{j'} : i', j' \in \{0, 1, \ldots, k\}\}$ visited by P is either $v^{i'}_{j+1}$ or $v^{j+1}_{i'}$.

By Claim 17 we have that the temporal path segments from $v^{i'}_j$ to $v^{i'}_{j+1}$ and $v^{j+1}_{i'}$, respectively, have duration at least $20n$. However, for the temporal path from $v^{i'}_j$ to $v^{j+1}_{i'}$ (with $j \neq i'-1$) we can obtain a larger lower bound. As argued in the proof of Claim 15, a temporal path segment from $v^{i'}_j$ to $v^{j+1}_{i'}$ has to either traverse two segments of $5n$ vertices in $\{a^{i',j',\ell}_{\ell'} : \ell' \in [5n]\}$ or $\{b^{i',j',\ell}_{\ell'} : \ell' \in [5n]\}$ for some $\ell \in [m]$ and $i', j' \in \{i-1, i, j, j+1\}$. More precisely, the temporal path segment has to traverse part of the edge selection gadget of color combination $i', j+1$. To this end, it traverses the $5n$ vertices in $\{a^{i',j+1,\ell}_{\ell''} : \ell'' \in [5n]\}$ for some $\ell \in [m]$. Then it traverses some vertices in the edge selection gadget, and then it traverses the $5n$ vertices in $\{b^{j+1,i',\ell'}_{\ell''} : \ell'' \in [5n]\}$ for some $\ell' \in [m]$.

By construction of G, the first vertex of the edge selection gadget visited by the path segment (after traversing vertices in $\{a^{i',j+1,\ell}_{\ell'} : \ell'' \in [5n]\}$) is some vertex $u^\ell_{\ell''}$ with $\ell'' \in \{0, 1, \ldots, 4n\}$. The last vertex of the edge selection gadget visited by the path segment is (before traversing the vertices in $\{b^{j+1,i',\ell'}_{\ell''''} : \ell'''' \in [5n]\}$) some vertex $u^{\ell'}_{\ell'''}$ with $\ell''' \in \{0, 1, \ldots, 4n\}$. By construction of G, the duration of a fastest path between $u^\ell_{\ell''}$ and $u^{\ell'}_{\ell'''}$ (in G) is at least $3n$. Investigating the second connector gadget of the edge selection gadget for $i', j+1$ we can see that a temporal path from $u^\ell_{\ell''}$ and $u^{\ell'}_{\ell'''}$ has duration at least $6n$.

APPENDIX

It follows that the temporal path segment from $v_j^{i'}$ to $v_{i'}^{j+1}$ (with $j \neq i'-1$) has duration at least $26n$. Furthermore, recall that P starts at v_0^i and ends at v_k^i. We have that if P contains a path segment from some $v_j^{i'}$ to $v_{i'}^{j+1}$ some (with $j \neq i'-1$), then P visits a vertex $v_{j'}^{i''}$ with $i'' \neq i$. Hence, it needs to contain at least one additional path segment from some $v_j^{i'}$ to some $v_{i'}^{j+1}$ (with $j \neq i-1$). However, then we have that the duration of P is at least $20kn + 12n > k(20n+6) + 6n - 1$, a contradiction.

We can conclude that P only contains temporal path segments from v_{j-1}^i to v_j^i for $j \in [k]$ and the claim follows. $\lhd$

Now we have by Claims 16 and 18 that we can divide P into k segments, the subpaths from v_{j-1}^i to v_j^i for $j \in [k]$. We show that all subpaths except the one from v_{i-1}^i to v_i^i have duration $20n+5$. The subpath from v_{i-1}^i to v_i^i has duration $26n+5$.

$\rhd$ Claim 19. Let $i \in [k]$ and $j \in [k] \setminus \{i\}$. Let P be a temporal path from v_{j-1}^i to v_j^i that does not visit vertices from connector gadgets and the alignment gadget. If P has duration at most $20n+5$, then it visits exactly two vertices $u_{\ell'-1}^{\ell}, u_{\ell'}^{\ell}$ with $\ell \in [m]$, and $\ell' \in [4n]$ of the edge selection gadget for color combination i, j (or j, i).

Proof. By the construction of G (and as also argued in the proofs of Claims 16 and 17), a temporal path P with duration at most $20n+5$ that does not visit vertices from connector gadgets and the alignment gadget from v_{j-1}^i to v_j^i has to first traverse a segment of $5n$ vertices in $\{a_{\ell'}^{i,j-1,\ell} : \ell' \in [5n]\}$ and then a segment of $5n$ vertices $\{b_{\ell'}^{i,j,\ell} : \ell' \in [5n]\}$ for some $\ell \in [m]$. By construction of G, the two vertices visited in the edge selection gadget for color combination i, j (or j, i) are $u_{\ell'-1}^{\ell}$ and $u_{\ell'}^{\ell}$ for some $\ell' \in [4n]$. By inspecting the connector gadgets in an analogous way as in the proof of Claim 17 we can deduce that all consecutive edges traversed by P must have labels that differ by at least 2. If follows that if all consecutive edges have labels that differ by exactly two, then P has duration $20n+5$. $\lhd$

$\rhd$ Claim 20. Let $i \in [k]$. Let P be a temporal path from v_{i-1}^i to v_i^i that does not visit vertices from connector gadgets and the alignment gadget. Then P has duration at least $26n+5$.

Proof. By construction of G we have that v_{i-1}^i and v_i^i are connected via a path inside the verification gadget for color i, visiting the $13n+1$ vertices in $\{\hat{u}_\ell^i : \ell \in [13n+1]\}$. Assume P follows this path. By inspecting the connector gadgets of the verification gadget of color i, we can see that all consecutive edges traversed by P must have labels that differ by at least two. It follows that P has duration at least $26n+5$. By construction of G we have that if P does not follow the vertices in $\{\hat{u}_\ell^i : \ell \in [13n+1]\}$ it has to visit at least three different edge selection gadgets: The one of color combination $i-1, i$, then one of $i-1, i+1$, and then the one of $i, i+1$. If follows that P needs to visit at least four segments of length $5n$ composed of vertices $\{a_{\ell'}^{i',j',\ell} : \ell' \in [5n]\}$ or $\{b_{\ell'}^{i',j',\ell} : \ell' \in [5n]\}$ for some $\ell \in [m]$ and $i', j' \in [k]$. By inspecting the connector gadgets of the verification gadgets we know that it takes at least $10n$ time steps to traverse such a segment. Hence, the duration of P is at least $40n$. $\lhd$

Furthermore, we need the following observation which is relevant when we try to connect the above mentioned segments to a temporal path.

$\rhd$ Claim 21. Let $i \in [k]$ and $0 \leq j \leq k$. The absolute difference of labels of any two different edges incident with v_j^i is at least two.

APPENDIX

Proof. This follows by inspecting the connector gadgets of the verification gadget of color i. $\lhd$

From Claims 14, 15, and 18–21 we get that a fastest temporal path P from v_0^i to v_k^i has the following properties.

1. The path P can be segmented into temporal path segments P_j from v_{j-1}^i to v_j^i for $j \in [k] \setminus \{i\}$ such that P_j is a temporal path from v_{j-1}^i to v_j^i that does not visit vertices from connector gadgets and the alignment gadget and has duration $20n + 5$.
2. The segment of P from v_{i-1}^i to v_i^i has duration $26n + 5$.
3. The path P dwells at each vertex v_j^i with $j \in [k-1]$ for exactly two time steps, that is, the absolute difference of the labels on the edges incident with v_j^i that are traversed by P is exactly two.

If any of the properties does not hold, then we can observe that $d(v_0^i, v_k^i) > 8n + 5$ would follow.

Now assume $i \in [k]$ and $j \in [k] \setminus \{i\}$ and consider a fastest temporal path P_j from v_{j-1}^i to v_j^i that does not visit vertices from connector gadgets and the alignment gadget and a fastest temporal path P_{j+1} from v_j^i to v_{j+1}^i that does not visit vertices from connector gadgets and the alignment gadget. By Claim 19 we know that P_j visits vertices $u_{\ell'-1}^{\ell}, u_{\ell'}^{\ell}$ with $\ell \in [m]$, and $\ell' \in [4n]$ of the edge selection gadget for color combination i, j. By Claim 10 we have that $\lambda(\{u_{\ell'-1}^{\ell}, u_{\ell'}^{\ell}\}) = (i + j) \cdot (2n \cdot (\sigma_{i,j}(\ell))^2 + 18n + 6) + 2\ell' + 2$, where $\sigma_{i,j}$ is the permutation of color combination i, j (or j, i). Analogously, we have by Claim 19 that P_{j+1} visits vertices $u_{\ell'''-1}^{\ell''}, u_{\ell'''}^{\ell''}$ with $\ell'' \in [m]$, and $\ell''' \in [4n]$ of the edge selection gadget for color combination $i, j + 1$. By Claim 10 we have that $\lambda(\{u_{\ell'''-1}^{\ell''}, u_{\ell'''}^{\ell''}\}) = (i + j + 1) \cdot (2n \cdot (\sigma_{i,j+1}(\ell''))^2 + 18n + 6) + 2\ell''' + 2$, where $\sigma_{i,j+1}$ is the permutation of color combination $i, j + 1$ (or $j + 1, i$). We have that

$$
\begin{aligned}
&\lambda(\{u_{\ell'''-1}^{\ell''}, u_{\ell'''}^{\ell''}\}) - \lambda(\{u_{\ell'-1}^{\ell}, u_{\ell'}^{\ell}\}) = \\
&(i + j + 1) \cdot (2n \cdot (\sigma_{i,j+1}(\ell''))^2 + 18n + 6) + 2\ell''' + 2 \\
&- ((i + j) \cdot (2n \cdot (\sigma_{i,j}(\ell))^2 + 18n + 6) + 2\ell' + 2) = \\
&(i + j + 1) \cdot 2n \cdot (\sigma_{i,j+1}(\ell''))^2 - (i + j) \cdot 2n \cdot (\sigma_{i,j}(\ell))^2 + 2(\ell''' - \ell') + 18n + 6
\end{aligned}
$$

By the arguments made before we also have that if P_j and P_{j+1} are both path segments of P, then

$$\lambda(\{u_{\ell'''-1}^{\ell''}, u_{\ell'''}^{\ell''}\}) - \lambda(\{u_{\ell'-1}^{\ell}, u_{\ell'}^{\ell}\}) = 20n + 6.$$

It follows that

$$(i + j + 1) \cdot 2n \cdot (\sigma_{i,j+1}(\ell''))^2 - (i + j) \cdot 2n \cdot (\sigma_{i,j}(\ell))^2 + 2(\ell''' - \ell') = 2n.$$

Assume that $\sigma_{i,j}(\ell) \neq \sigma_{i,j+1}(\ell'')$, then we have that $(i + j + 1) \cdot 2n \cdot (\sigma_{i,j+1}(\ell''))^2 - (i + j) \cdot 2n \cdot (\sigma_{i,j}(\ell))^2 < 6n$ or $(i + j + 1) \cdot 2n \cdot (\sigma_{i,j+1}(\ell''))^2 - (i + j) \cdot 2n \cdot (\sigma_{i,j}(\ell))^2 > 10n$, since $|(\sigma_{i,j}(\ell''))^2 - (\sigma_{i,j}(\ell))^2| \geq 3$ and $(i + j) \geq 3$. However, we have that $\ell', \ell''' \in [4n]$ and hence $|2(\ell''' - \ell')| < 8n$. We can conclude that $\sigma_{i,j}(\ell) = \sigma_{i,j+1}(\ell'')$. In this case we have that $(i + j + 1) \cdot 2n \cdot (\sigma_{i,j+1}(\ell''))^2 - (i + j) \cdot 2n \cdot (\sigma_{i,j}(\ell))^2 = 2n \cdot (\sigma_{i,j}(\ell'))^2$. It follows that $2n(\sigma_{i,j}(\ell))^2 - 2(\ell''' - \ell') = 2n$. Again, since $|2(\ell''' - \ell')| < 8n$, we have that $\sigma_{i,j}(\ell) = 1$ and in turn this implies that $\ell' = \ell'''$.

Note that if $i = 1$ or $i = k$ we can already conclude that $|(\bigcap_{1 \leq j < i} e_{j,i}) \cap (\bigcap_{i < j \leq k} e_{i,j})| = 1$. By construction of G we have that for all $j \in [k] \setminus \{i\}$ that v_{j-1}^i and v_j^i are connected to $u_{\ell'-1}^{\ell}$ and $u_{\ell'}^{\ell}$ of the edge selection gadget of color combination i, j (or j, i), respectively, via paths

APPENDIX

using vertices $\{a^{i,j,\ell}_{\ell''} : \ell'' \in [5n]\}$ and $\{b^{i,j,\ell}_{\ell''} : \ell'' \in [5n]\}$, respectively, if the vertex $w^i_{\ell'} \in W_i$ (for $i = k$, or vertex $w^i_{\ell'-3n} \in W_i$ for $i = 1$) is incident with edge $e^{i,j}_{\ell} \in F_{i,j}$. Note that since $\sigma_{i,j}(\ell) = 1$ we have that $e^{i,j}_{\ell} \in X$. Since ℓ' is independent from ℓ and j, it follows that $(\bigcap_{1\leq j<i} e_{j,i}) \cap (\bigcap_{i<j\leq k} e_{i,j}) = \{w^i_{\ell'}\}$ for $i = k$ and $(\bigcap_{1\leq j<i} e_{j,i}) \cap (\bigcap_{i<j\leq k} e_{i,j}) = \{w^i_{\ell'-3n}\}$ for $i = 1$.

Assume now that $1 \neq i \neq k$. By Claim 20 we know that the duration of the path segment P_i from v^i_{i-1} to v^i_i is $26n + 5$. Consider the path segment $P^\star$ from v^i_{i-2} to v^i_{i+1}. By the arguments above we know that $P^\star$ visits vertices $u^{\ell}_{\ell'-1}, u^{\ell}_{\ell'}$ with $\sigma_{i-1,i}(\ell) = 1$, and $\ell' \in [4n]$ of the edge selection gadget for color combination $i-1, i$ and afterwards $P^\star$ visits vertices $u^{\ell''}_{\ell'''-1}, u^{\ell''}_{\ell'''}$ with $\sigma_{i,i+1}(\ell'') = 1$, and $\ell''' \in [4n]$ of the edge selection gadget for color combination $i, i+1$. By analogous arguments as above and the fact that the duration of P_i is $26n + 5$ we get that

$$\lambda(\{u^{\ell''}_{\ell'''-1}, u^{\ell''}_{\ell'''}\}) - \lambda(\{u^{\ell}_{\ell'-1}, u^{\ell}_{\ell'}\}) = 46n + 6.$$

It follows that

$$(2i + 1) \cdot (20n + 6) + 2\ell''' + 2 - ((2i - 1) \cdot (20n + 6) + 2\ell' + 2) = 46n + 6,$$

and hence $\ell''' - \ell' = 3n$. By construction of G we have that v^i_{i-2} and v^i_{i-1} are connected to $u^{\ell}_{\ell'-1}$ and $u^{\ell}_{\ell'}$ of the edge selection gadget of color combination $i-1, i$, respectively, via paths using vertices $\{a^{i,i-1,\ell}_{\ell''''} : \ell'''' \in [5n]\}$ and $\{b^{i,i-1,\ell}_{\ell''''} : \ell'''' \in [5n]\}$, respectively, if the vertex $w^i_{\ell'} \in W_i$ is incident with edge $e^{i-1,i}_{\ell} \in F_{i-1,i}$. Furthermore, we have that v^i_i and v^i_{i+1} are connected to $u^{\ell''}_{3n+\ell'-1}$ and $u^{\ell''}_{3n+\ell'}$ of the edge selection gadget of color combination $i, i+1$, respectively, via paths using vertices $\{a^{i,i+1,\ell''}_{\ell''''} : \ell'''' \in [5n]\}$ and $\{b^{i,i+1,\ell''}_{\ell''''} : \ell'''' \in [5n]\}$, respectively, if the vertex $w^i_{\ell'} \in W_i$ is incident with edge $e^{i,i+1}_{\ell''} \in F_{i,i+1}$.

Note that since $\sigma_{i-1,i}(\ell) = \sigma_{i,i+1}(\ell'') = 1$ we have that $e^{i-1,i}_{\ell} \in X$ and $e^{i,i+1}_{\ell''} \in X$. Since, again, ℓ' is independent from ℓ and j, it follows that $e^{i-1,i}_{\ell} \cap e^{i,i+1}_{\ell''} = \{w^i_{\ell'}\}$. By arguments analogous to the ones above we can also deduce that $\bigcap_{1\leq j<i} e_{j,i} = \{w^i_{\ell'}\}$ and $\bigcap_{i<j\leq k} e_{i,j} = \{w^i_{\ell'}\}$. It follows that $(\bigcap_{1\leq j<i} e_{j,i}) \cap (\bigcap_{i<j\leq k} e_{i,j}) = \{w^i_{\ell'}\}$.

We can conclude that indeed $\bigcup_{e\in X} e$ forms a multicolored clique in H.

($\Leftarrow$): Assume H is a Yes-instance of Multicolored Clique and let X be a solution. We construct the following labeling for the underlying graph G, see also Figure 3 for an illustration.

We start with the labels for edges from the alignment gadget.

- For every $w \in \hat{W}$ we set $\lambda(\{w^\star, w\}) = 1$.
- Let $\hat{v}_0$ belong to some connector gadget and let $w \in \hat{W}$ be neighbor of $\hat{v}_0$. Then we set $\lambda(\{w, \hat{v}_0\}) = n^9$.
- Let y^i belong to the verification gadget of color i and let $w \in \hat{W}$ be neighbor of y^i. Then we set $\lambda(\{w, y^i\}) = n^8 - 1$. Furthermore, we set $\lambda(\{y_i, v^i_0\}) = n^8$.
- Let x_1 belong to the edge selection gadget for color combination i, j and let $w \in \hat{W}$ be neighbor of x_1. Then we set $\lambda(\{w, x_1\}) = (i + j)(20n + 6)$.

Next, consider a connector gadget with vertices $\hat{v}_0, \hat{v}'_0, \hat{v}_1, \hat{v}_2, \hat{v}_3, \hat{v}'_3$ and set A, B.

- We set $\lambda(\{\hat{v}_0, \hat{v}_1\}) = \lambda(\{\hat{v}_, \hat{v}_3\}) = n^9$.
- We set $\lambda(\{\hat{v}'_0, \hat{v}_1\}) = \lambda(\{\hat{v}_, \hat{v}'_3\}) = n^9 + 2$.
- We set $\lambda(\{\hat{v}_1, \hat{v}_2\}) = n^9 + 1$.
- For all vertices $v \in A \setminus B$ we set $\lambda(\{\hat{v}_1, v\}) = n^9$ and $\lambda(\{\hat{v}_2, v\}) = n^9 + 2$.
- For all vertices $v \in B$ we set $\lambda(\{\hat{v}_1, v\}) = \lambda(\{\hat{v}_2, v\}) = n^9$.

APPENDIX

- For all vertices $v \in V^\star \setminus A$ we set $\lambda(\{\hat{v}_1, v\}) = \lambda(\{\hat{v}_2, v\}) = n^9 + 2$. (Recall that $V^\star$ denotes the set of all vertices from all edge selection gadgets and all verification gadgets).

Recall that the following duration requirements were specified in the construction of the instance. It is straightforward to verify that durations requirements we recall in the following are all met, assuming no faster connections are introduced.

- We have set $d(\hat{v}_0, \hat{v}_2) = d(\hat{v}_3, \hat{v}_1) = d(\hat{v}_2, \hat{v}_0') = d(\hat{v}_1, \hat{v}_3') = 2$, and $d(\hat{v}_0, \hat{v}_0') = d(\hat{v}_3, \hat{v}_3') = d(\hat{v}_0, \hat{v}_3') = d(\hat{v}_3, \hat{v}_0') = 3$.
- Let $v \in A$, then we have set $d(v, \hat{v}_0') = 3$ and $d(v, \hat{v}_3') = 3$.
- Let $v \in V^\star \setminus B$, then we have set $d(\hat{v}_0, v) = 3$ and $d(\hat{v}_3, v) = 3$.
- Let $v \in A$ and $v' \in V^\star \setminus B$ such that v and v' are not neighbors, then we have set $d(v, v') = 3$.

For two connector gadgets, one with vertices $\hat{v}_0, \hat{v}_0', \hat{v}_1, \hat{v}_2, \hat{v}_3, \hat{v}_3'$ and with sets A and B, and one with vertices $\hat{v}_0', \hat{v}_0'', \hat{v}_1', \hat{v}_2', \hat{v}_3', \hat{v}_3''$ and with sets A' and B', we have set the following durations.

- If there is a vertex $v \in A$ with $v \notin A'$, then we have set $d(\hat{v}_1, \hat{v}_1') = 3$.
- If there is a vertex $v \in A$ with $v \in A' \setminus B'$, then we have set $d(\hat{v}_1, \hat{v}_2') = 3$.
- If there is a vertex $v \in V^\star \setminus (A \setminus B)$ with $v \notin A'$, then we have set $d(\hat{v}_2, \hat{v}_1') = 3$.
- If there is a vertex $v \in V^\star \setminus (A \setminus B)$ with $v \in A' \setminus B'$, then we have set $d(\hat{v}_2, \hat{v}_2') = 3$.

For the alignment gadget the following requirements were specified. Let x_1 belong to the edge selection gadget of color combination i, j and let $w \in \hat{W}$ denote the neighbor of x_1 in the alignment gadget. Let $\hat{v}_1$ and $\hat{v}_2$ belong to the first connector gadget of the edge selection gadget for color combination i, j. Let $\hat{V}$ contain all vertices $\hat{v}_1$ and $\hat{v}_2$ belonging to the other connector gadgets (different from the first one of the edge selection gadget for color combination i, j).

- We have set $d(w^\star, x_1) = (20n + 6)(i + j)$.
- We have set $d(w^\star, \hat{v}_1) = n^9$, $d(w, \hat{v}_2) = n^9$, $d(w, \hat{v}_1) = n^9 - (20n + 6)(i + j) + 1$, and $d(w, \hat{v}_2) = n^9 - (20n + 6)(i + j) + 1$.
- For each vertex $v \in (V^\star \cup \hat{V}) \setminus (X_{i,j} \cup \{v_{i,j}^{\star\star}\})$ we have set $d(w^\star, v) = n^9 + 2$ and $d(w, v) = n^9 - (20n + 6)(i + j) + 3$.

Let y^i belong to the verification gadget of color i and let $w' \in \hat{W}$ denote the neighbor of y^i in the alignment gadget. Let $\hat{v}_1$ and $\hat{v}_2$ belong to the connector gadget of the verification gadget for color i. Let $\hat{V}$ contain all vertices $\hat{v}_1$ and $\hat{v}_2$ belonging to the other connector gadgets (different from the one of the verification gadget for color i). Let V_i denote the set of all vertices of the verification gadget of color i.

- We have set $d(w^\star, y^i) = n^8 - 1$, $d(w', v_0^i) = 2$, and $d(w^\star, v_0^i) = n^8$.
- We have set $d(w^\star, \hat{v}_1) = n^9$, $d(w^\star, \hat{v}_2) = n^9$, $d(w', \hat{v}_1) = n^9 - n^8$, and $d(w', \hat{v}_2) = n^9 - n^8$.
- For each vertex $v \in (V^\star \cup \hat{V}) \setminus V_i$ we have set $d(w^\star, v) = n^9 + 1$, $d(w, v) = n^9 - n^8 + 2$, and $d(y^i, v) = n^9 - n^8 + 2$.

Let $\hat{v}_1$ belong to some connector gadget. We have set $d(w^\star, \hat{v}_1) = n^9$.

We will make sure that no faster connections are introduced by only using even numbers as labels and labels that are strictly smaller than $n^8 - 1$. Furthermore, we can already see that no vertex except the ones in $\hat{W}$ can reach $w^\star$ and no two vertices $w, w' \in \hat{W}$ can reach each other, as required.

Next, consider the edge selection gadget for color combination i, j with $i < j$. To describe the labels, we define a permutation $\sigma_{i,j} : [m] \to [m]$ as follows. Let $\{w_{\ell'}^i\} = X \cap W_i$ and $\{w_{\ell''}^j\} = X \cap W_j$. Then, since X is a clique in H, we have that $\{w_{\ell'}^i, w_{\ell''}^j\} = e_\ell^{i,j} \in F_{i,j}$. We set $\sigma_{i,j}(\ell) = 1$ and $\sigma_{i,j}(1) = \ell$. For all $\ell''' \in [m]$ with $1 \neq \ell''' \neq \ell$ we set $\sigma_{i,j}(\ell''') = \ell'''$.

APPENDIX

Let $x_1, x_2, \ldots, x_m$ belong to the edge selection gadget for color combination i, j.

- For all $\ell''' \in [m]$ we set $\lambda(\{x_{\ell'''}, v^{\star}_{i,j}\}) = (i+j) \cdot (2n(\ell''')^2 + 18n + 6)$.

Note that using these labels, we obey the following duration constraints.

- For all $1 \leq \ell''' < \ell'''' \leq m$ we have set $d(x_{\ell'''}, x_{\ell''''}) = 2n \cdot (i+j) \cdot ((\ell'''')^2 - (\ell''')^2) + 1$.

Furthermore, we set the following labels.

- For all $\ell''' \in [m]$ we set $\lambda(\{u_0^{\ell'''}, v^{\star}_{i,j}\}) = (i+j) \cdot (2n \cdot (\sigma_{i,j}(\ell'''))^2 + 18n + 6) + 2$, where $u_0^{\ell'''}$ belongs to the edge selection gadget for i, j.
- For all $\ell''' \in [m]$ and $\ell'''' \in [4n]$ we set $\lambda(\{u^{\ell'''}_{\ell''''-1}, u^{\ell'''}_{\ell''''}\}) = (i+j) \cdot (2n \cdot (\sigma_{i,j}(\ell'''))^2 + 18n + 6) + 2\ell'''' + 2$, where $u^{\ell'''}_{\ell''''-1}$ and $u^{\ell'''}_{\ell''''}$ belong to the edge selection gadget for i, j.
- For all $\ell''' \in [m]$ we set $\lambda(\{u^{\ell'''}_{4n}, v^{\star\star}_{i,j}\}) = (i+j) \cdot (2n \cdot (\sigma_{i,j}(\ell'''))^2 + 18n + 6) + 8n + 4$, where $u^{\ell'''}_{4n}$ belongs to the edge selection gadget for i, j.

It is straightforward to verify that with these labels we get for all $\ell''' \in [m]$ that $d(x_{\ell'''}, v^{\star\star}_{i,j}) = 8n+5$, as required. Furthermore, we get that for all $\ell''' \in [m]$ that $d(v^{\star\star}_{i,j}, x_{\ell'''}) = \infty$. To see this, consider the following. Vertex $v^{\star\star}_{i,j}$ is not temporally connected to vertices $x_{\ell'''}$ with $\ell''' \in [m]$ via any of the connector gadgets, since for all connector gadgets where $v^{\star\star}_{i,j} \in A$ we have that all vertices $x_{\ell'''}$ with $\ell''' \in [m]$ are either contained in B or they are not contained in A. By the construction of the labels of the connector gadgets, it follows that $v^{\star\star}_{i,j}$ cannot reach any vertex $x_{\ell'''}$ with $\ell''' \in [m]$ via the connector gadgets. We can observe that in all other connections in the underlying graph from $v^{\star\star}_{i,j}$ to a vertex $x_{\ell'''}$ with $\ell''' \in [m]$ are paths which have non-increasing labels, hence they also do not provide a temporal connection.

Furthermore, we get that for all $1 \leq \ell''' \leq \ell'''' \leq m$ we get that $d(x_{\ell'''}, x_{\ell''''}) = 2n \cdot (i+j) \cdot ((\ell'''')^2 - (\ell''')^2) + 1$, through a temporal path via $v^{\star}_{i,j}$. By similar observations as in the previous paragraph, we also have that $d(x_{\ell''''}, x_{\ell'''}) = \infty$.

Finally, consider the verification gadget for color i. Let $1 \leq j < i$. Let $\{w^i_{\ell'}\} = X \cap W_i$ and $\{w^j_{\ell''}\} = X \cap W_j$ and $\{w^i_{\ell'}, w^j_{\ell''}\} = e^{j,i}_{\ell} \in F_{j,i}$. Recall that we set $\sigma_{j,i}(\ell) = 1$ and $\sigma_{j,i}(1) = \ell$. For all $\ell'' \in [m]$ with $1 \neq \ell'' \neq \ell$ we set $\sigma_{j,i}(\ell'') = \ell''$. Recall that we set $\lambda(\{u^{\ell}_{\ell'-1}, u^{\ell}_{\ell'}\}) = (i+j) \cdot (20n+6) + 2\ell' + 2$, where $u^{\ell}_{\ell'-1}$ and $u^{\ell}_{\ell'}$ belong to the edge selection gadget for j, i. Now we set for all $\ell'' \in [5n-1]$ and all $\ell''' \in [m]$ the following.

- $\lambda(\{a^{i,j,\ell'''}_{5n}, u^{\ell'''}_{\ell''''}\}) = (i+j) \cdot (20n+6) + 2\ell'$ for all ℓ'''' such that this edge exists.
- $\lambda(\{a^{i,j,\ell'''}_{1}, v^{i}_{j-1}\}) = (i+j) \cdot (20n+6) + 2\ell' - 10n - 2$.
- $\lambda(\{a^{i,j,\ell'''}_{\ell''}, a^{i,j,\ell'''}_{\ell''+1}\}) = (i+j) \cdot (20n+6) + 2\ell' - 10n + 2\ell''$.
- $\lambda(\{b^{i,j,\ell'''}_{5n}, u^{\ell'''}_{\ell''''}\}) = (i+j) \cdot (20n+6) + 2\ell' + 4$ for all ℓ'''' such that this edge exists.
- $\lambda(\{b^{i,j,\ell'''}_{1}, v^{i}_{j}\}) = (i+j) \cdot (20n+6) + 2\ell' + 10n + 6$.
- $\lambda(\{b^{i,j,\ell'''}_{\ell''}, b^{i,j,\ell'''}_{\ell''+1}\}) = (i+j) \cdot (20n+6) + 2\ell' + 10n - 2\ell'' + 4$.

For all $\ell'' \in [13n]$ we set the following.

- $\lambda(\{\hat{u}^i_{\ell''}, \hat{u}^i_{\ell''+1}\}) = 2i \cdot (20n+6) + 2\ell' - 10n + 2\ell'' - 2$.
- $\lambda(\{v^i_{i-1}, \hat{u}^i_{1}\}) = 2i \cdot (20n+6) + 2\ell' - 10n - 2$.
- $\lambda(\{v^i_{i}, \hat{u}^i_{13n+1}\}) = 2i \cdot (20n+6) + 2\ell' + 16n + 4$.

Let $i < j \leq k$. Let $\{w^i_{\ell'}\} = X \cap W_i$ and $\{w^j_{\ell''}\} = X \cap W_j$ and $\{w^i_{\ell'}, w^j_{\ell''}\} = e^{i,j}_{\ell} \in F_{i,j}$. Recall that we set $\sigma_{i,j}(\ell) = 1$ and $\sigma_{i,j}(1) = \ell$. For all $\ell'' \in [m]$ with $1 \neq \ell'' \neq \ell$ we set $\sigma_{i,j}(\ell'') = \ell''$. Recall that we set $\lambda(\{u^{\ell}_{3n+\ell'-1}, u^{\ell}_{3n+\ell'}\}) = (i+j) \cdot (20n+6) + 2\ell' + 6n + 2$, where $u^{\ell}_{3n+\ell'-1}$ and $u^{\ell}_{3n+\ell'}$ belong to the edge selection gadget for i, j. Now we set for all $\ell'' \in [5n-1]$ and all $\ell''' \in [m]$ the following.

- $\lambda(\{a^{i,j,\ell'''}_{5n}, u^{\ell'''}_{\ell''''}\}) = (i+j) \cdot (20n+6) + 2\ell' + 6n$ for all ℓ'''' such that this edge exists.
- $\lambda(\{a^{i,j,\ell'''}_{1}, v^{i}_{j-1}\}) = (i+j) \cdot (20n+6) + 2\ell' - 4n - 2$.

APPENDIX

- $\lambda(\{a^{i,j,\ell'''}_{\ell''}, a^{i,j,\ell'''}_{\ell''+1}\}) = (i+j)\cdot(20n+6) + 2\ell' - 4n + 2\ell''$.
- $\lambda(\{b^{i,j,\ell'''}_{5n}, u^{\ell'''}_{\ell''''}\}) = (i+j)\cdot(20n+6) + 2\ell' + 6n + 4$ for all ℓ'''' such that this edge exists.
- $\lambda(\{b^{i,j,\ell'''}_{1}, v^{i}_{j}\}) = (i+j)\cdot(20n+6) + 2\ell' + 16n + 6$.
- $\lambda(\{b^{i,j,\ell'''}_{\ell''}, b^{i,j,\ell'''}_{\ell''+1}\}) = (i+j)\cdot(20n+6) + 2\ell' + 16n - 2\ell'' + 4$.

Now we verify that we meet the duration requirements. For all $0 \le j < j' < i$ and all $i \le j < j' \le k$ we have set the following.

- We set $d(v^i_j, v^i_{j'}) = (20n+6)(j'-j) - 1$.

To see that this holds, we analyse the fastest paths from vertices v^i_{j-1} to vertices v^i_j for $j \in [k] \setminus \{i\}$. Let $\{w^i_{\ell'}\} = X \cap W_i$ and $\{w^j_{\ell''}\} = X \cap W_j$ and $\{w^i_{\ell'}, w^j_{\ell''}\} = e^{i,j}_{\ell} \in F_{i,j}$. Then, starting at v^i_{j-1}, we follow the vertices in $\{a^{i,j,\ell}_{\ell''} : \ell'' \in [5n]\}$ to arrive at $u^{\ell}_{\ell'-1}$. From there we move to $u^{\ell}_{\ell'}$ and from there we continue along the vertices in $\{b^{i,j,\ell}_{\ell''} : \ell'' \in [5n]\}$ to arrive at v^i_j. By construction this describes a fastest temporal path from v^i_{j-1} to v_j with duration $20n+5$. To get from v^i_j to $v^i_{j'}$ for $0 \le j < j' < i$ we move from v^i_j to v^i_{j+1} in the above described fashion and from there to v^i_{j+1} and so on until we arrive at $v^i_{j'}$. By construction this yields a fastest temporal path from v^i_j to $v^i_{j'}$ with duration $(20n+6)(j'-j)-1$, as required. The case where $i \le j < j' \le k$ is analogous.

For all $0 \le j < i$ and all $i \le j' \le k$ we have set the following.

- We set $d(v^i_j, v^i_{j'}) = (20n+6)(j'-j) + 6n - 1$.

Here we move from v^i_j to v^i_{i-1} in the above described fashion. Then we move from v^i_{i-1} to v^i_i along vertices $\{\hat{u}^i_{\ell''} : \ell'' \in [13+1]\}$ and then we move from v^i_i to $v^i_{j'}$ again in the above described fashion. By construction this yields a fastest temporal path from v^i_j to $v^i_{j'}$ with duration $(20n+6)(j'-j) + 6n - 1$, as required.

By similar observations as in the analysis for the edge selection gadgets, we also get that for all $1 \le j < j' \le k$ that $d(v^i_{j'}, v^i_j) = \infty$.

This finishes the proof.

Infinity gadget. Finally, we show how to get rid of the infinity entries in D and how to allow a finite Δ. To this end, we introduce the *infinity gadget*. We add four vertices z_1, z_2, z_3, z_4 to the graph and we set $\Delta = n^{11}$. Let V denote the set of all remaining vertices. We set the following durations.

- For all $v \in V$ we set $d(z_1, v) = 2$, $d(z_2, v) = d(v, z_2) = 1$, $d(z_3, v) = d(v, z_3) = 1$, and $d(z_4, v) = 2$. Furthermore, we set $d(v, z_1) = n^{11}$ and $d(v, z_4) = n^{10} - 1$.
- We set $d(z_1, z_2) = d(z_2, z_1) = 1$, $d(z_2, z_3) = d(z_3, z_2) = 1$, and $d(z_3, z_4) = d(z_4, z_3) = 1$.
- We set $d(z_1, z_3) = 3$, $d(z_3, z_1) = n^{11} - 1$, $d(z_2, z_4) = n^{10} - 2$, and $d(z_4, z_2) = n^{11} - n^{10} + 4$.
- We set $d(z_1, z_4) = n^{10}$ and $d(z_4, z_1) = 2n^{11} - n^{10} + 2$.
- For every pair of vertices $v, v' \in V$ where previously the duration of a fastest path from v to v' was specified to be infinite, we set $d(v, v') = n^{10}$.

Now we analyse which implications we get for the labels on the newly introduced edges. Assume $\lambda(\{z_1, z_2\}) = t$, then we get the following. For all $v \in V$ we have that $d(z_1, v) = 2$ and hence we get that $\lambda(\{z_2, v\}) = t+1$. Since $d(z_1, z_4) = n^{10}$, we have that $\lambda(z_3, z_4) = t + n^{10} - 1$. From this follows that for all $v \in V$, since $d(z_4, v) = 2$, that $\lambda(\{z_3, v\}) = t + n^{10}$. Finally, since $d(z_1, z_3) = 3$, we have that $\lambda(\{z_2, z_3\}) = t+2$. For an illustration see Figure 4. It is easy to check that all duration requirements between vertex pairs in $\{z_1, z_2, z_3, z_4\}$ are met and that all duration requirements between each vertex $v \in V$ and each vertex in $\{z_1, z_2, z_3, z_4\}$ are met. Furthermore, it is easy to check that the gadget increases the feedback vertex set by two (z_2 and z_3 need to be added).

APPENDIX

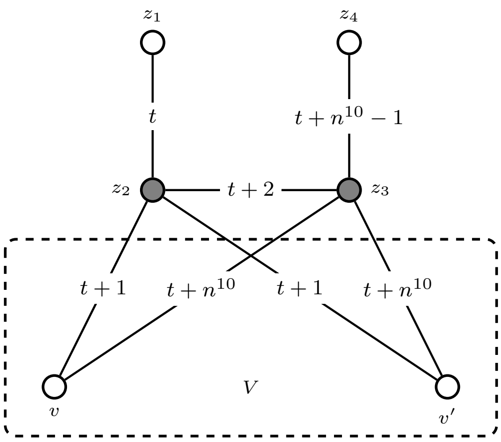


Figure 4 Illustration of the infinity gadget. Gray vertices need to be added to the feedback vertex set.

Lastly, consider two vertices $v, v' \in V$. Note that before the addition of the infinity gadget, by construction of G we have that $d(v, v') \leq n^9 + 2$ or $d(v, v') = \infty$. Furthermore, if D is a Yes-instance, we have shown in the correctness proof of the reduction that the difference between the smallest label and the largest label is at most $n^9 + 1$. This implies that for a vertex pair $v, v' \in V$ with $d(v, v') = \infty$ we have in the periodic case with $\Delta = n^{11}$, that $d(v, v') \geq n^{11} - n^9 > n^{10}$. Which means, after adding the vertices and edges of the infinity gadget, we indeed have that $d(v, v') = n^{10}$. For all vertex pairs v, v' where in the original construction we have $d(v, v') \neq \infty$, we can also see that adding the infinity gadget and setting $\Delta = n^{11}$ does not change the duration of a fastest path from v to v', since all newly added temporal paths have duration at least n^{10}. We can conclude that the originally constructed instance D is a Yes-instance if and only if it remains a Yes-instance after adding the infinity gadget and setting $\Delta = n^{11}$. ◀

3 Algorithms for Simple TGR

In this section we provide several algorithms for Simple TGR. By Theorem 3 we have that Simple TGR is NP-hard in general, hence we start by identifying restricted cases where we can solve the problem in polynomial time. We first show in Section 3.1 that if the underlying graph G of an instance (D, Δ) of Simple TGR is a tree, then we can determine desired Δ-periodic labeling λ of G in polynomial time. In Section 3.2 we generalize this result. We show that Simple TGR is fixed-parameter tractable when parameterized by the feedback edge number of the underlying graph. Note that our parameterized hardness result (Theorem 4) implies that we presumably cannot replace the feedback edge number with the smaller parameter feedback vertex number, or any other parameter that is smaller than the feedback vertex number, such as e.g. the treewidth.

3.1 Polynomial-time algorithm for trees

We now provide a polynomial-time algorithm for Simple TGR when the underlying graph is a tree. Let D be the input matrix and let the underlying graph G of D be a tree on n vertices $\{v_1, v_2, \ldots, v_n\}$. Let v_i, v_j be two arbitrary vertices in G, then we know that there exists a unique (static) path $P_{i,j}$ from v_i to v_j. We will heavily exploit this in our algorithm.

APPENDIX

▶ **Theorem 22.** *SIMPLE TGR can be solved in polynomial time on trees.*

Proof. Let D be an input matrix for problem SIMPLE TGR of dimension $n \times n$. Let us fix the vertices of the corresponding graph G of D as $v_1, v_2, \ldots, v_n$, where vertex v_i corresponds to the row and column i of matrix D. This can be done in polynomial time as we need to loop through the matrix D once and connect vertices v_i, v_j for which $D_{i,j} = 1$. At the same time we also check if $D_{i,i} = 0$, for all $i \in [n]$. When G is constructed we run DFS algorithm on it and check that it has no cycles. If at any step we encounter a problem, our algorithm stops and returns a negative answer.

Having computed G, our algorithm proceeds as follows. We pick an arbitrary edge f and give it label one, that is, $\lambda(f) = 1$. Now we push all edges incident with f into a (initially empty) queue. Now we repeat the following as long as the queue is not empty:

- Pop edge $e = \{u, v\}$ from the queue. Since e was pushed into the queue, there is an edge e' incident with e that already obtained a label. Let w.l.o.g. $e' = \{v, w\}$. Then we set $\lambda(e) = (\lambda(e') - D_{u,w} + 1) \bmod \Delta$.
- Push all edges incident with e that have not received a label yet into the queue.

When the queue is empty, all edges have received a label. Iterate over all vertex pairs u, v and check whether the fastest path from u to v in (G, λ) has duration $D_{u,v}$. If this check succeeds for all vertex pairs, output the labelling λ, otherwise abort.

It is easy to see that the described algorithm runs in polynomial time. In the remainder, we proof that it is correct.

($\Rightarrow$): Since the algorithm checks at the end whether all durations specified in D are realized by the corresponding fastest paths, we clearly face a yes-instance whenever the algorithm outputs a labeling.

($\Leftarrow$): Assume we face a yes-instance, then there exists a labeling $\lambda^\star$ that realizes all durations specified in D. Let $e^\star$ denote the edge initially picked by the algorithm. For all edges e let $\lambda(e) = (\lambda^\star(e) - \lambda^\star(e^\star) + 1) \bmod \Delta$. Clearly, the labeling λ also realizes all durations specified in D since λ is obtained by adding the constant $(1 - \lambda^\star(e^\star))$ modulo Δ to all labels of $\lambda^\star$ which does not change the duration of any temporal path, that is all durations in $(G, \lambda^\star)$ are the same as their counterparts in (G, λ). We claim that our algorithm computes and outputs λ.

We prove that our algorithm computes λ by induction on the distance of the labeled edges to $e^\star$, where the distance of two edges e, e' is defined as the length of a shortest path that uses e as its first edge and e' as its last edge.

Initially, our algorithm labels $e^\star$ with one, which equals $\lambda(e^\star)$. Now let e be an edge popped off the queue by the algorithm in some iteration. Let e' be the edge incident with e that already obtained a label and is considered by the algorithm. Since G is a tree, we have that e' is closer to $e^\star$ than e. By induction we have that the algorithm labeled e' with $\lambda(e')$. Assume that $e = \{u, v\}$ and $e' = \{v, w\}$. Since G is a tree there is only one path from u to w in G and it uses edges e and e'. It follows that $\lambda(e') - \lambda(e) + 1 = D_{u,w}$ if $\lambda(e') > \lambda(e)$, and $\lambda(e') - \lambda(e) + \Delta + 1 = D_{u,w}$ otherwise. Our algorithm labels e with $(\lambda(e') - D_{u,w} + 1) \bmod \Delta$. It is straightforward to verify that the label of e computed by the algorithm equals $\lambda(e)$. It follows that the algorithm computes λ. ◀

3.2 FPT-algorithm for feedback edge number

Recall from Section 3.1 that the main reason, for which SIMPLE TGR is straightforward to solve on trees, is twofold:

APPENDIX

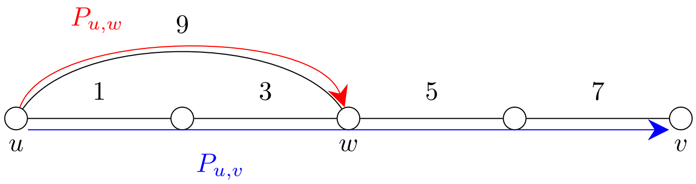


Figure 5 An example of a temporal graph (with $\Delta \geq 9$), where the fastest temporal path $P_{u,v}$ (in blue) from u to v is of duration 7, while the fastest temporal path $P_{u,w}$ (in red) from u to a vertex w, that is on a path $P_{u,v}$, is of duration 1 and is not a subpath of $P_{u,v}$.

- between any pair of vertices v_i and v_j in the tree T, there is a *unique* path P in T from v_i to v_j, and
- in any periodic temporal graph (T, λ, Δ) and any fastest temporal path $P = ((e_1, t_1), \ldots, (e_i, t_i), \ldots, (e_j, t_j), \ldots, (e_\ell, t_\ell))$ from v_1 to v_ℓ we have that the sub-path $P' = ((e_i, t_i), \ldots, (e_j, t_j))$ is also a fastest temporal path from v_i to v_j.

However, these two nice properties do not hold when the underlying graph is not a tree. For example, in Figure 5, the fastest temporal path from u to v is $P_{u,v}$ (depicted in blue) goes through w, however the sub-path of $P_{u,v}$ that stops at w is not the fastest temporal path from u to w. The fastest temporal path from u to w consists only of the single edge uw (with label 9 and duration 1, depicted in red).

Nevertheless, we prove in this section that we can still solve Simple TGR efficiently if the underlying graph is similar to a tree; more specifically we show the following result, which turns out to be non-trivial.

▶ **Theorem 23.** *Simple TGR is in FPT when parameterized by the feedback edge number of the underlying graph.*

From Theorem 4 and Theorem 23 we immediately get the following, which is the main result of the paper.

▶ **Corollary 24.** *Simple TGR is:*
- *in FPT when parameterized by the* feedback edge number *or any larger parameter, such as the* maximum leaf number.
- *W[1]-hard when parameterized by the* feedback vertex number *or any smaller parameter, such as:* treewidth, degeneracy, cliquewidth, distance to chordal graphs, *and* distance to outerplanar graphs.

Before presenting the structure of our algorithm for Theorem 23, observe that, in a static graph, the number of paths between two vertices can be upper-bounded by a function $f(k)$ of the feedback edge number k of the graph. Therefore, for any fixed pair of vertices u and v, we can "guess" the edges of the fastest temporal path from u to v. However, for an FPT algorithm with respect to k, we cannot afford to guess the edges of the fastest temporal path for each of the $O(n^2)$ pairs of vertices. To overcome this difficulty, our algorithm follows this high-level strategy:
- We identify a small number $f(k)$ of "important vertices"; these consist of the sets that we call U, U^*, Z^*.
- For each pair u, v of important vertices, we guess the edges of the fastest temporal path from u to v (and from v to u).
- From these guesses we can still not deduce the edges of the fastest temporal paths between many pairs of non-important vertices. However, as we prove, it suffices to guess only a small number of specific auxiliary structures (to be defined later).

APPENDIX

- From these guesses we deduce fixed relationships between the labels of most of the edges of the graph.
- For all the edges, for which we do not have deduced a label yet, we introduce a *variable*. Using all these variables, we build an Integer Linear Program (ILP). Among the constraints in this ILP we have that, for each of the $O(n^2)$ pairs of vertices u, v in the graph, the duration of one specific temporal path from u to v (according to our guesses) is *equal* to the desired duration $D_{u,v}$, while the duration of each of the other temporal paths from u to v is *at least* $D_{u,v}$.
- By making any of the above guesses, we restrict the solution space for the problem Simple TGR. This restricted solution space coincides with the set of feasible solutions to the resulting ILP. Furthermore, the set of feasible solutions for all constructed ILPs coincide with the set of all solutions to Simple TGR (i. e., regardless of our guesses). As each ILP can be solved in FPT time with respect to k by Lenstra's Theorem [46] (the number of variables is upper bounded by a function of k), we obtain our FPT algorithm for Simple TGR with respect to k.

For the remainder of this section, we fix the following notation. Let D be the input matrix of Simple TGR i. e., the matrix of the fastest temporal paths between all pairs of n vertices, and let G be its underlying graph, on n vertices and m edges. With F we denote a minimum feedback edge set of G, and with k the feedback edge number of G. We are now ready to present our FPT algorithm. For an easier readability we split the description and analysis of the algorithm in five subsections. We start with a preprocessing procedure for graph G, where we define a set of interesting vertices which then allows us to guess the desired structures. Next we introduce some extra properties of our problem, that we then use to create ILP instances and their constraints. At the end we present how to solve all instances and produce the desired labeling λ of G, if possible.

3.2.1 Preprocessing of the input

From the underlying graph G of D we first create a graph G' by iteratively removing vertices of degree one from G, and denote with

$$Z = V(G) \setminus V(G').$$

Then we determine a minimum feedback edge set F of G'. Note that F is also a minimum feedback edge set of G. Lastly, we determine sets U, of *vertices of interest*, and U^* of the neighbors of vertices of interest, in the following way. Let T be a spanning tree of G', with F being the corresponding feedback edge set of G'. Let $V_1 \subseteq V(G')$ be the set of leaves in the spanning tree T, $V_2 \subseteq V(G')$ be the set of vertices of degree two in T, that are incident to at least one edge in F, and let $V_3 \subseteq V(G')$ be the set of vertices of degree at least 3 in T. Then $|V_1| + |V_2| \leq 2k$, since every leaf in T and every vertex in V_2 is incident to at least one edge in F, and $|V_3| \leq |V_1|$ by the properties of trees. We denote with

$$U = V_1 \cup V_2 \cup V_3$$

the set of vertices of interest. It follows that $|U| \leq 4k$. We set U^* to be the set of vertices in $V(G') \setminus U$ that are neighbors of vertices in U, i. e.,

$$U^* = \{v \in V(G') \setminus U : u \in U, v \in N(u)\}.$$

Again, using the tree structure, we get that for any $u \in U$ its neighborhood is of size $|N(u)| \in O(k)$, since every neighbor of u is the first vertex of a (unique) path to another

APPENDIX

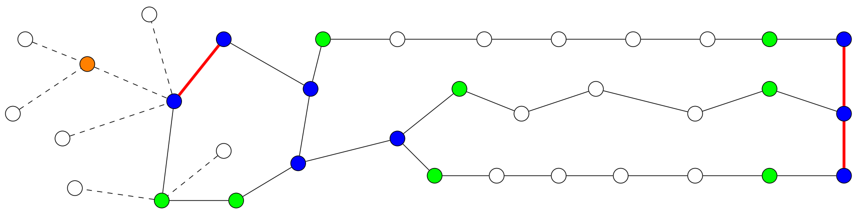


Figure 6 An example of a graph with its important vertices: U (in blue), U^* (in green) and Z^* (in orange). Corresponding feedback edges are marked with a thick red line, while dashed edges represent the edges (and vertices) "removed" from G' at the initial step.

vertex in U. It follows that $|U^*| \in O(k^2)$. From the construction of Z (by iteratively removing vertices of degree one from G) it follows that Z consists of disjoint trees $T_1, T_2, \ldots$. For a tree T_i we denote with u_i the vertex in G' that is a neighbor of a vertex in T_i, and call it a *clip vertex of the tree* T_i. It follows that there can be many different trees T_i that are incident to the same clip vertex $u_i \in V(G')$, but each tree T_i is incident to exactly one clip vertex $u_i \in V(G')$. Since u_i is the only vertex connecting all of the trees T_i incident to it, from now on we assume that a tree T_{u_i} in Z is a union of trees on vertices from $V(G) \setminus V(G')$, that are clipped at the same vertex $u_i \in V(G')$. For each of the trees T_{u_i} in Z, we select one vertex r_i, that is a neighbor of the clip vertex u_i, and call it *a representative vertex of the tree* T_{u_i}. We now define as Z^* the set of representatives r_i of trees $T_i \in Z$, where the clip vertex v_i of T_i is a vertex of interest, i. e.,

$$Z^* = \{r_i : r_i \in T_i, \text{ where } T_i \in Z, \text{ the clip vertex } u_i \text{ of } T_i \text{ is in } U, \text{ and } r_i u_i \in E(G)\}.$$

Since there are $O(k)$ vertices of interest, we get that $|Z^*| \in O(k)$. Finally, the set of *important vertices* is defined as the set $U \cup U^* \cup Z^*$. For an illustration see Figure 6. Note that determining sets U, U^*, and Z^* takes linear time.

Recall that a labeling λ of G satisfies D if the duration of a fastest temporal path from vertex v_i to v_j equals D_{v_i,v_j}. In order to find a labeling that satisfies this property we split our analysis in nine cases. We consider fastest temporal paths where the starting vertex is in one of the sets $U, V(G') \setminus U, Z$, and similarly the destination vertex is in one of the sets $U, V(G') \setminus U, Z$. In each of these cases we guess the underlying path P that at least one fastest temporal path from the vertex v_i to v_j follows, which results in one equality constraint for the labels on the path P. For all other temporal paths from v_i to v_j we know that they cannot be faster, so we introduce inequality constraints for them. This results in producing $f(k) \cdot |D|^{O(1)}$ constraints. Note that we have to do this while keeping the total number of variables upper-bounded by some function in k.

For an easier understanding and the analysis of the algorithm we give the following definition.

▶ **Definition 25.** *Let $U \subseteq V(G')$ be a set of vertices of interest and let $u, v \in U$. A path $P = (u = v_1, v_2, \ldots, v_p = v)$ in graph G', where all inner vertices are not in U, i. e., $v_i \notin U$ for all $i \in \{2, 3, \ldots, p-1\}$, is called a* segment *from u to v. We denote it as $S_{u,v}$.*

Note from Definition 25 we get that $S_{u,v} \neq S_{v,u}$, since we consider paths to be directed. Observe that a temporal path in G' between two vertices of interest is either a segment, or consists of a sequence of some segments. Furthermore, since we have at most $4k$ interesting vertices in G', we can deduce the following important result.

▶ **Corollary 26.** *There are at most $O(k^2)$ segments in G'.*

APPENDIX

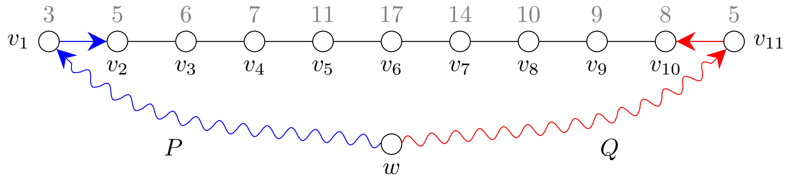


Figure 7 In the above graph vertices v_1, v_{11}, w are in U, while v_2, v_{10} are in U^*. Numbers above all v_i represent the values of the fastest temporal paths from w to each of them (i. e., the entries in the w-th row of matrix D). From the basic guesses we know the fastest temporal path P from w to v_2 (depicted in blue) and the fastest temporal path Q from w to v_{10}. From the values of durations from w to each v_i we cannot determine the fastest paths from w to all v_i. More precisely, we know that w reaches v_2, v_3, v_4, v_5 (resp. v_{10}, v_9, v_9, v_7) by first using the path P (resp. Q) and then proceeding through the vertices, but we do not know how w reaches v_6 the fastest. Therefore we have to introduce some more guesses.

3.2.2 Guessing necessary structures

Once the sets U, U^* and Z^* are determined, we are ready to start guessing the necessary structures. Note that whenever we say that we guess the fastest temporal path between two vertices, we mean that we guess the underlying path of a representative fastest temporal path between those two vertices. To describe the guesses, we introduce the following notation. Let u, v, x be three vertices in G'. We write $u \rightsquigarrow x \rightarrow v$ to denote a temporal path from u to v that passes through x, and then goes directly to v (via one edge). If there is an edge (i. e., a unique fastest path) between two vertices, we denote it by $\rightarrow$, if the fastest path between two vertices is not uniquely determined, we denote it by $\rightsquigarrow$.

For every pair of important vertices $u, v \in U \cup U^* \cup Z^*$, we guess the sequence of edges in the fastest temporal path from u to v. Since $U \cup U^* \cup Z^* \in O(k^2)$ and there are $k^{O(k)}$ possibilities for a sequence of edges between a fixed vertex pair, we have $k^{O(k^5)}$ overall possible guesses.

G-1. The fastest temporal paths between all pairs of vertices of U. For a pair u, v of vertices in U, there are $k^{O(k)}$ possible paths in G' between them. Therefore, we have to try all $k^{O(k)}$ possible paths, where at least one of them will be a fastest temporal path from u to v, respecting the values from D. Repeating this procedure for all pairs of vertices $u, v \in U$ we get $k^{O(k^3)}$ different variations of the fastest temporal paths between all pairs of vertices in U.

G-2. The fastest temporal paths between all pairs of vertices in Z^*, which by similar arguing as for vertices in U, gives us $k^{O(k^3)}$ guesses.

G-3. The fastest temporal paths between all pairs of vertices in U^*. This gives us $k^{O(k^5)}$ guesses.

G-4. The fastest temporal paths from vertices of U to vertices in U^*, and vice versa, the fastest temporal paths from vertices in U^* to vertices in U. This gives us $k^{O(k^4)}$ guesses.

G-5. The fastest temporal paths from vertices of U to vertices in Z^*, and vice versa. This gives us $k^{O(k^3)}$ guesses.

G-6. The fastest temporal paths from vertices of U^* to vertices in Z^*, and vice versa. This gives us $k^{O(k^4)}$ guesses.

With the information provided by the described guesses we are still not able to determine all fastest paths. For example consider the case depicted in Figure 7. Therefore we introduce additional guesses that provide us with sufficient information to determine all fastest paths. We guess the following structures.

APPENDIX

G-7. Inner segment guess I. Let $S_{u,v} = (u = v_1, v_2, \ldots, v_p = v)$ and $S_{w,z} = (w = z_1, z_2, \ldots, z_r = z)$ be two segments. We want to guess the fastest temporal path $v_2 \to u \rightsquigarrow w \to z_2$. We repeat this procedure for all pairs of segments. Since there are $O(k^2)$ segments in G', there are $k^{O(k^5)}$ possible paths of this form.

Recall that $S_{u,v} \neq S_{v,u}$ for every $u, v \in U$. Furthermore note that we did not assume that $\{u, v\} \cap \{w, z\} = \emptyset$. Therefore, by repeatedly making the above guesses, we also guess the following fastest temporal paths: $v_2 \to u \rightsquigarrow z \to z_{r-1}$, $v_2 \to u \rightsquigarrow v \to v_{p-1}$, $v_{p-1} \to v \rightsquigarrow w \to z_2$, $v_{p-1} \to v \rightsquigarrow z \to z_{r-1}$, and $v_{p-1} \to v \rightsquigarrow u \to v_2$. For an example see Figure 8a.

G-8. Inner segment guess II. Let $S_{u,v} = (u = v_1, v_2, \ldots, v_p = v)$ be a line segment in G', and let $w \in U \cup Z^*$. We want to guess the following fastest temporal paths $w \rightsquigarrow u \to v_2$, $w \rightsquigarrow v \to v_{p-1} \to \cdots \to v_2$, and $v_2 \to u \rightsquigarrow w$, $v_2 \to v_3 \to \cdots v \rightsquigarrow w$.

For fixed $S_{u,v}$ and $w \in U \cup Z^*$ we have $k^{O(k)}$ different possible such paths, therefore we make $k^{O(k^4)}$ guesses for these paths. For an example see Figure 8b.

G-9. Split vertex guess I. Let $S_{u,v} = (u = v_1, v_2, \ldots, v_p = v)$ be a line segment in G', and let us fix a vertex $v_i \in S_{u,v} \setminus \{u, v\}$. In the case when $S_{u,v}$ is of length 4, the fixed vertex v_i is the middle vertex, else we fix an arbitrary vertex $v_i \in S_{u,v} \setminus \{u, v\}$. Let $S_{w,z} = (w = z_1, z_2, \ldots, z_r = z)$ be another segment in G'. We want to determine the fastest paths from v_i to all inner vertices of $S_{w,z}$. We do this by inspecting the values in matrix D from v_i to inner vertices of $S_{w,z}$. We split the analysis into two cases.

a. There is a single vertex $z_j \in S_{w,z}$ for which the duration from v_i is the biggest. More specifically, $z_j \in S_{w,z} \setminus \{w, z\}$ is the vertex with the biggest value D_{v_i,z_j}. We call this vertex a *split vertex of v_i in the segment S_{wz}*. Then it holds that $D_{v_i,z_2} < D_{v_i,z_3} < \cdots < D_{v_i,z_j}$ and $D_{v_i,z_{r-1}} < D_{v_i,z_{r-2}} < \cdots < D_{v_i,z_j}$. From this it follows that the fastest temporal paths from v_i to $z_2, z_3, \ldots, z_{j-1}$ go through w, and the fastest temporal paths from v_i to $z_{r-1}, z_{r-2}, \ldots, z_{j+1}$ go through z. We now want to guess which vertex w or z is on a fastest temporal path from v_i to z_j. Similarly, all fastest temporal paths starting at v_i have to go either through u or through v, which also gives us two extra guesses for the fastest temporal path from v_i to z_j. Therefore, all together we have 4 possibilities on how the fastest temporal path from v_i to z_j starts and ends. Besides that we want to guess also how the fastest temporal paths from v_i to z_{j-1}, z_{j+1} start and end. Note that one of these is the subpath of the fastest temporal path from v_i to z_j, and the ending part is uniquely determined for both of them, i. e., to reach z_{j-1} the fastest temporal path travels through w, and to reach z_{j+1} the fastest temporal path travels through z. Therefore we have to determine only how the path starts, namely if it travels through u or v. This introduces two extra guesses. For a fixed $S_{u,v}, v_i$ and $S_{w,z}$ we find the vertex z_j in polynomial time, or determine that z_j does not exist. We then make four guesses where we determine how the fastest temporal path from v_i to z_j passes through vertices u, v and w, z and for each of them two extra guesses to determine the fastest temporal path from v_i to z_{j-1} and from v_i to z_{j+1}. We repeat this procedure for all pairs of segments, which results in producing $k^{O(k^5)}$ new guesses. Note, $v_i \in S_{u,v}$ is fixed when calculating the split vertex for all other segments $S_{w,z}$.

b. There are two vertices $z_j, z_{j+1} \in S_{w,z}$ for which the duration from v_i is the biggest. More specifically, $z_j, z_{j+1} \in S_{w,z} \setminus \{w, z\}$ are the vertices with the biggest value $D_{v_i,z_j} = D_{v_i,z_{j+1}}$. Then it holds that $D_{v_i,z_2} < D_{v_i,z_3} < \cdots < D_{v_i,z_j} = D_{v_i,z_{j+1}} > D_{v_i,z_{j+2}} > \cdots > D_{v_i,z_{r-1}}$. From this it follows that the fastest temporal paths from v_i to $z_2, z_3, \ldots, z_j$ go through w, and the fastest temporal paths from v_i to

$z_{r-1}, z_{r-2}, \dots, z_{j+1}$ go through z. In this case we only need to guess the following two fastest temporal paths $u \rightsquigarrow w \to z_2$ and $u \rightsquigarrow z \to z_{r-1}$. Each of this paths we then uniquely extend along the segment $S_{w,z}$ up to the vertex v_j, resp. v_{j+1}, which give us fastest temporal paths from u to v_j and from u to v_{j+1}. In this case we do not introduce any new guesses, as we have aready guessed the fastest paths of the form $u \rightsquigarrow w \to z_2$ and $u \rightsquigarrow z \to z_{r-1}$ (see guess **G-8**).

Note that this case results also in knowing the fastest paths from the vertex $v_i \in S_{u,v}$ to $w, z \in S_{w,z}$ for all segments $S_{w,z}$, i. e., we know the fastest paths from a fixed $v_i \in S_{u,v}$ to all vertices of interest in U. For an example see Figure 8c.

G-10. Split vertex guess II. Let $w \in U \cup Z^*$ be either a vertex of interest or a representative vertex of a tree, whose clipped vertex is a of interest, and let $S_{u,v} = (u = v_1, v_2, \dots, v_p = v)$ be a line segment in G'. Similarly as above, in guess **G-9**, we want to guess a split vertex of w in $S_{u,v}$, and the fastest temporal path that reaches it. We again have two cases, first one where v_i is a unique vertex in $S_{u,v}$ that is furthest away from w, and the second one where v_i, v_{i+1} are two incident vertices in $S_{u,v}$, that are furthest away from w. In first case we know exactly how the fastest paths from w to all vertices $v_j \in S_{u,v} \setminus \{v_i\}$ travel through the segment $S_{u,v}$ (i. e., either through u or v). Therefore we have to guess how the fastest path from w reaches vertex v_i, we have two options, either it travels through $u \to v_2 \to \cdots \to v_{i-1} \to v_i$ or $v \to v_{p-1} \to \cdots \to v_{i+1} \to v_i$. Which produces two new guesses. In the second case we know exactly how the fastest temporal path reaches v_i and v_{i+1}, and consequently all the inner vertices. Therefore no new guesses are needed. Note that the above guesses, together with the guesses from **G-8**, uniquely determine fastest temporal paths from w to all vertices in $S_{u,v}$ (this also holds for the case when $w \in S_{u,v}$, i. e., $w = u$ or $w = v$).

All together we make two guesses for each pair of vertex $w \in U$ and segment $S_{u,v}$. We repeat this for all vertices of interest, and all segments, which produces $k^{O(k^2)}$ new guesses. For an example see Figure 8d.

There are two more guesses **G-11** and **G-12** that we make during the creation of the ILP instances, we explain these guesses in detail in Section 3.2.4. We will prove that, for all guesses **G-1** to **G-12**, there are in total at most $f(k)$ possible choices, and for each one of them we create an ILP with at most $f(k)$ variables and at most $f(k) \cdot |D|^{O(1)}$ constraints. Each of these ILPs can be solved in FPT time by Lenstra's Theorem [46].

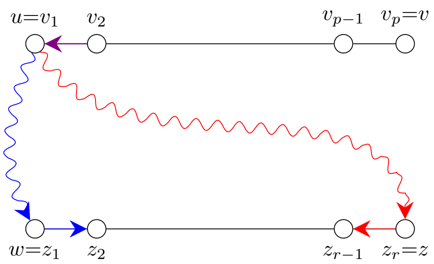
3.2.3 Properties of Simple TGR

In this section we study the properties of our problem, that then help us creating constraints of our ILP instances. Recall that with G we denote our underlying graph of D. We want to determine labeling λ of each edge of G. We start with an empty labeling of edges and try to specify each one of them. Note, that this does not necessarily mean that we assign numbers to the labels, but we might specify labels as variables or functions of other labels. We say that the label of an edge f is *determined with respect to* the label of the edge edge e, if we have determined $\lambda(f)$ as a function of $\lambda(e)$.

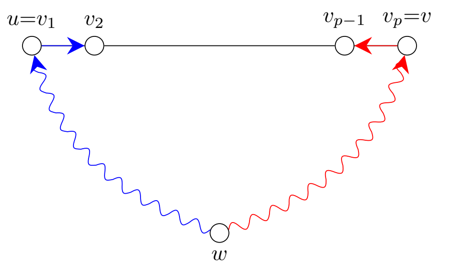
We first start with defining certain notions, that will be of use when solving the problem.

▶ **Definition 27** (Travel delays). *Let (G, λ) be a temporal graph satisfying conditions of* Simple TGR. *Let $e_1 = uv$ and $e_2 = vz$ be two incident edges in G with $e_1 \cap e_2 = v$. We define the* travel delay *from u to z at vertex v, denoted with τ_v^{uz}, as the difference of the labels of e_2 and e_1, where we subtract the value of the label of e_1 from the label of e_2, modulo Δ.*

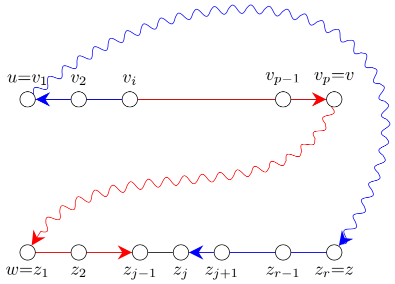
APPENDIX



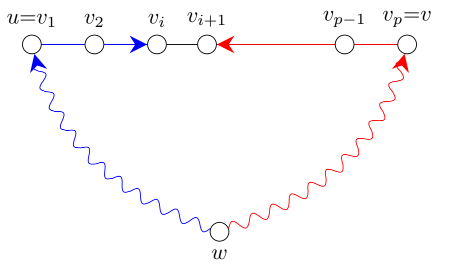
(a) Example of an Inner segment guess I (**G-7**), where we guessed the fastest temporal paths of the form $v_2 \to u \rightsquigarrow w \to z_2$ (in blue) and $v_2 \to u \rightsquigarrow z \to z_{r-1}$ (in red).



(b) Example of an Inner segment guess II (**G-8**), where we guessed the fastest temporal paths of the form $w \rightsquigarrow u \to v_2$ (in blue) and $w \rightsquigarrow v \to v_{p-1}$ (in red).



(c) Example of a Split vertex guess I (**G-9**), where, for a fixed vertex $v_i \in S_{u,v}$, we calculated its corresponding split vertex $z_j \in S_{w,z}$, and guessed the fastest paths of the form $v_i \to v_{i-1} \to \cdots \to u \rightsquigarrow z \to z_{r-1} \cdots \to z_j$ (in blue) and $v_i \to v_{i+1} \to \cdots \to v \rightsquigarrow w \to z_2 \to \cdots \to z_{j-1}$ (in red).



(d) Example of a Split vertex guess II (**G-10**), where, for a vertex of interest w, we calculated its corresponding split vertex $v_i \in S_{u,v}$, and guessed the fastest paths of the form $w \rightsquigarrow u \to v_2 \to \cdots \to v_i$ (in blue) and $w \rightsquigarrow v \to v_{p-1} \to \cdots \to v_{i+1}$ (in red).

Figure 8 Illustration of the guesses **G-7**, **G-8**, **G-9**, and **G-10**.

More specifically:

$$\tau_v^{uz} \equiv \lambda(e_2) - \lambda(e_1) \pmod{\Delta}. \tag{1}$$

Similarly, $\tau_v^{zu} \equiv \lambda(e_1) - \lambda(e_2) \pmod{\Delta}$.

Intuitively, the value of τ_v^{uz} represents how long a temporal path waits at vertex v when first taking edge $e_1 = uv$ and then edge $e_2 = vz$.

From the above definition and the definition of the duration of the temporal path P we get the following two observations.

▶ **Observation 28.** *Let* $P = (v_0, v_1, \dots, v_p)$ *be the underlying path of the temporal path* (P, λ) *from* v_0 *to* v_p. *Then* $d(P, \lambda) = \sum_{i=1}^{p-1} \tau_{v_i}^{v_{i-1}v_i} + 1$.

Proof. For the simplicity of the proof denote $t_i = \lambda(v_{i-1}v_i)$, and suppose that $t_i \leq t_{i+1}$, for all $i \in \{1, 2, 3, \dots, p\}$. Then

$$\begin{aligned}
\sum_{i=1}^{p-1} \tau_{v_i}^{v_{i-1}v_i} + 1 &= \sum_{i=1}^{p-1} (t_{i+1} - t_i) + 1 \\
&= (t_2 - t_1) + (t_3 - t_2) + \cdots + (t_p - t_{p-1}) + 1 \\
&= t_{p-1} - t_1 + 1 \\
&= d(P, \lambda)
\end{aligned}$$

APPENDIX

Now in the case when $t_i > t_{i+1}$ we get that $\tau_{v_i}^{v_{i-1}v_{i+1}} = \Delta + t_{i+1} - t_i$. At the end this still results in the correct duration as the last time we traverse the path P is not exactly t_p but $k\lambda + t_p$, for some k. ◀

We also get the following.

▶ **Observation 29.** *Let (G, λ) be a temporal graph satisfying conditions of the* Simple *TGR problem. For any two incident edges $e_1 = uv$ and $e_2 = vz$ on vertices $u, v, z \in V$, with $e_1 \cap e_2 = v$, we have $\tau_v^{zu} = \Delta - \tau_v^{uz} \pmod{\Delta}$.*

Proof. Let $e_1 = uv$ and $e_2 = vz$ be two edges in G for which $e_1 \cap e_2 = v$. By the definition $\tau_v^{uz} \equiv \lambda(e_2) - \lambda(e_1) \pmod{\Delta}$ and $\tau_v^{zu} \equiv \lambda(e_1) - \lambda(e_2) \pmod{\Delta}$. Summing now both equations we get $\tau_v^{uz} + \tau_v^{zu} \equiv \lambda(e_2) - \lambda(e_1) + \lambda(e_1) - \lambda(e_2) \pmod{\Delta}$, and therefore $\tau_v^{uz} + \tau_v^{zu} \equiv 0 \pmod{\Delta}$, which is equivalent as saying $\tau_v^{uz} \equiv -\tau_v^{zu} \pmod{\Delta}$ or $\tau_v^{zu} = \Delta - \tau_v^{uz}$ $\pmod{\Delta}$. ◀

In our analysis we exploit the following greatly, that is why we state is as an observation.

▶ **Observation 30.** *Let P be the underlying path of a fastest temporal path from u to v, where $e_1, e_p \in P$ are its first and last edge, respectively. Then, knowing the label $\lambda(e_1)$ of the first edge and the duration $d(P, \lambda)$ of the temporal path (P, λ), we can uniquely determine the label $\lambda(e_p)$ of the last edge of P. Symmetrically, knowing $\lambda(e_p)$ and $d(P)$, we can uniquely determine $\lambda(e_1)$.*

The correctness of the above statement follows directly from Definition 2. This is because the duration of (P, λ) is calculated as the difference of labels of last and first edge plus 1, where the label of last edge is considered with some delta periods, i. e., $d(P, \lambda) = p_i\Delta + \lambda(e_p) - \lambda(e_1) + 1$, for some $p_i \geq 0$. Therefore $d(P, \lambda) \pmod{\Delta} \equiv (\lambda(e_p) - \lambda(e_1) + 1) \pmod{\Delta}$. Note that if $\lambda(e_1)$ and $\lambda(e_p)$ are both unknown, then we can determine one with respect to the other.

In the following we prove that knowing the structure (the underlying path) of a fastest temporal path P from a vertex of interest u to a vertex of interest v, results in determining the labeling of each edge in the fastest temporal path from u to v (with the exception of some constant number of edges), with respect to the label of the first edge. More precisely, if path P from u to v is a segment, then we can determine labels of all edges as a function of the label of the first edge. If P consists of ℓ segments, then we can determine the labels of all but $\ell - 1$ edges as a function of the label of the first edge. For the exact formulation and proofs see Lemmas 31 and 32.

▶ **Lemma 31.** *Let $u, v \in U$ be two arbitrary vertices of interest and suppose that $P = (u = v_1, v_2, \ldots, v_p = v)$, where $p \geq 2$, is a path in G', which is also the underlying path of a fastest temporal path from u to v. Moreover suppose also that P is a segment. We can determine the labeling λ of every edge in P with respect to the label $\lambda(uv_2)$ of the first edge.*

Proof. We claim that u reaches all of the vertices in P the fastest, when traveling along P (i. e., by using a subpath of P). To prove this suppose for the contradiction that there is a vertex $v_i \in P \setminus \{u, v\}$, that is reached from v on a path different than $P_i = (u, v_2, v_3, \ldots, v_i)$ faster than through P_i. Since the only vertices of interest of P are u and v, it follows that all other vertices on P are of degree 2. Then the only way to reach v_i from u, that differs from P, would be to go from u to v using a different path P_2, and then go from v to $v_{p-1}, v_{p-2}, \ldots, v_i$. But since P is the fastest temporal path from u to v, we get that $d(P_2) \geq d(P)$ and $d(P_2 \cup (v, v_{p-1}, \ldots, v_i)) > d(P) > d(P_i)$.

APPENDIX

Now to label P we use the fact that the fastest temporal path from u to any $v_i \in P$ is a subpath of P, therefore we can label each edge using Observation 30, where the duration from u to v_i equals to D_{u,v_i} and we set the label of the first edge of P to be a constant $c \in [\Delta]$. This gives us a unique label for each edge of P, that depends on the value $\lambda(uv_2)$. ◀

▶ **Lemma 32.** *Let $u, v \in U$ be two arbitrary vertices of interest and suppose that $P = (u = v_1, v_2, \ldots, v_p = v)$, where $p \geq 2$, is a path in G', which is also the underlying path of a fastest temporal path from u to v. Let $\ell_{u,v} \geq 1$ be the number of vertices of interest in P different to u, v, namely $\ell_{u,v} = |\{P \setminus \{u, v\}\} \cap U|$. We can determine the labeling λ of all but $\ell_{u,v}$ edges of P, with respect to the label $\lambda(uv_2)$ of the first edge, such that the labeling λ respects the values from D.*

For the proof of the above lemma, we first prove a weaker statement, for which we need to introduce some extra definitions and fix some notations. In the following we only consider *wasteless* temporal paths. We call a temporal path $P = ((e_1, t_1), \ldots, (e_k, t_k))$ a *wasteless* temporal path, if for every $i = 1, 2, \ldots, k-1$, we have that t_{i+1} is the first time after t_i that the edge e_{i+1} appears.

Let $u, v \in V$, and let $t \in \mathbb{N}$. Given that a temporal path starts within the period $[t, t+\Delta-1]$, we denote with $A_t(u, v)$ the *arrival* of the fastest path in (G, λ) from u to v, and with $A_t(u, v, P)$, the *arrival* along path P in (G, λ) from u to v. Whenever $t = 1$, we may omit the index t, i. e., we may write $A(u, v, P) = A_1(u, v, P)$ and $A(u, v) = A_1(u, v)$.

Suppose now that we know the underlying path $P_{u,v} = (u = v_1, v_2, \ldots, v_p = v)$ of the fastest temporal path between vertices of interest u and v in G'. Let $v_i \in U$ with $u \neq v_i \neq v$ be a vertex of interest on the path $P_{u,v}$. Suppose that v_i is reached the fastest from u by a path $P = (u = u_1, u_2, \ldots, u_{j-1}, v_i)$. We split the path with $P_{u,v}$ into a path $Q = (u = v_1, v_2, \ldots, v_i)$ and $R = (v_i, v_{i+1}, \ldots, v_p = v)$ (for details see Figure 9).

From the above we get the following assumptions:

1. $d(u, v_i) = d(u, v_i, P) \leq d(u, v_i, Q)$, and
2. $d(u, v_p) = d(u, v_p, Q \cup R) \leq d(u, v_p, P \cup R)$.

In the remainder, we denote with δ_0 the difference $d(u, v_i, Q) - d(u, v_i) \geq 0$. Let $t_{v_2} \in [\Delta]$ be the label of the edge uv_2, and denote by t_{u_2} the appearance of the edge uu_2 within the period $[t_{v_2}, t_{v_2} + \Delta - 1]$. Note that $1 \leq t_{v_2} \leq \Delta$ and that $t_{v_2} \leq t_{u_2} \leq 2\Delta$. From Assumption 1 we get

$$\delta_0 = d(u, v_i, Q) - d(u, v_i) = A_{t_{v_2}}(u, v_i, Q) - A_{t_{v_2}}(u, v_i, P) + (t_{u_2} - t_{v_2})$$

and thus

$$A_{t_{v_2}}(u, v_i, P) - A_{t_{v_2}}(u, v_i, Q) = t_{u_2} - (t_{v_2} + \delta_0). \tag{2}$$

We use all of the above discussion, to prove the following lemma.

▶ **Lemma 33.** *If $t_{u_2} \neq t_{v_2}$, then $\delta_0 \leq \Delta - 2$ and $t_{u_2} \geq t_{v_2} + \delta_0 + 1$.*

Proof. First assume that $\delta_0 \geq \Delta - 1$. Then, it follows by Equation (2) that $A_{t_{v_2}}(u, v_i, P) - A_{t_{v_2}}(u, v_i, Q) \leq t_{u_2} - t_{v_2} - \Delta + 1 \leq 0$, and thus $A_{t_{v_2}}(u, v_i, P) \leq A_{t_{v_2}}(u, v_i, Q)$. Therefore, since we can traverse path P from u to v_i by departing at time $t_{u_2} \geq t_{v_2} + 1$ and by arriving no later than traversing path Q, we have that $d(u, v_p, P \cup Q) < d(u, v_p, Q \cup R)$, which is a contradiction to the second initial assumption. Therefore $\delta_0 \leq \Delta - 2$.

Now assume that $t_{v_2} + 1 \leq t_{u_2} \leq t_{v_2} + \delta_0$. Then, it follows by Equation (2) that $A_{t_{v_2}}(u, v_i, P) \leq A_{t_{v_2}}(u, v_i, Q)$ which is, similarly to the previous case, a contradiction. Therefore $t_{u_2} \geq t_{v_2} + \delta_0 + 1$. ◀

APPENDIX

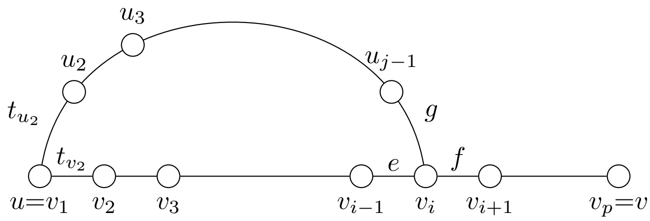


Figure 9 An example of the situation in Lemma 32, where we assume that the fastest temporal path from u to v is $P_{u,v} = (u = v_1, v_2, \dots v_p)$, and the fastest temporal path from u to v_i in $P_{u,v}$ is $P = (u, u_2, u_3, \dots, v_i)$. We denote with $Q = (u = v_1, v_2, \dots, v_i)$ and with $R = (v_i, v_{i+1}, \dots, v_p = v)$.

The next corollary follows immediately from Lemma 33.

▶ **Corollary 34.** *If $t_{u_2} \neq t_{v_2}$, then $1 \leq A_{t_{v_2}}(u, v_i, P) - A_{t_{v_2}}(u, v_i, Q) \leq \Delta - 1 - \delta_0$.*

We are now ready to prove the following result.

▶ **Lemma 35.** *$d(u, v_{i-1}, P \cup \{v_i v_{i-1}\}) > d(u, v_{i-1}, Q \setminus \{v_i v_{i-1}\})$.*

Proof. Let $e \in [\Delta]$ be the label of the edge $v_{i-1}v_i$, and let $f \in [e+1, e+\Delta]$ be the time of the first appearance of the edge $v_i v_{i+1}$ after time e. Let $A_{t_{v_i}}(u, v_i, Q) = x\Delta + e$. Then $A_{t_{v_i}}(u, v_{i+1}, Q \cup \{v_i v_{i+1}\}) = x\Delta + f$. Furthermore let g be such that $A_{t_{v_i}}(u, v_i, P) = x\Delta + g$.

Case 1: $t_{u_2} \neq t_{v_2}$. Then Corollary 34 implies that $e + 1 \leq g \leq e + (\Delta - 1 - \delta_0)$. Assume that $g < f$. Then, we can traverse path P from u to v_i by departing at time $t_{u_2} \geq t_{v_2} + 1$ and by arriving at most at time $x\Delta + f - 1$, and thus $d(u, v_p, P \cup R) < d(u, v_p, Q \cup R)$, which is a contradiction to the second initial assumption. Therefore $g \geq f$. That is,

$$e + 1 \leq f \leq g \leq e + (\Delta - 1 - \delta_0).$$

Consider the path $P^* = P \cup \{v_i v_{i-1}\}$. Assume that we start traversing P^* at time t_{u_2}. Then we arrive at v_i at time $x\Delta + g$, and we continue by traversing edge $v_i v_{i-1}$ at time $(x+1)\Delta + e$. That is, $d(u, v_{i-1}, P^*) = (x+1)\Delta + e - t_{u_2} + 1$.

Now consider the path $Q^* = Q \setminus \{v_i v_{i-1}\}$. Let $h \in [1, \Delta]$ be such that $A_{t_{v_i}}(u, v_{i-1}, Q^*) = x\Delta + e - h$. That is, if we start traversing Q^* at time t_{v_2}, we arrive at v_{i-1} at time $x\Delta + e - h$, i. e., $d(u, v_{i-1}, Q^*) = x\Delta + e - h - t_{v_2} + 1$. Summarizing, we have:

$$\begin{aligned} d(u, v_{i-1}, P^*) - d(u, v_{i-1}, Q^*) &= \Delta + h - (t_{u_2} - t_{v_2}) \\ &\geq (\Delta - \delta_0) + h > 0, \end{aligned}$$

which proves the statement of the lemma.

Case 2: $t_{u_2} = t_{v_2}$. Then, it follows by Equation (2) that $A_{t_{v_2}}(u, v_i, P) = A_{t_{v_2}}(u, v_i, Q) - \delta_0 \leq A_{t_{v_2}}(u, v_i, Q)$. Therefore $g \leq e$. Similarly to Case 1 above, consider the paths $P^* = P \cup \{v_i v_{i-1}\}$ and $Q^* = Q \setminus \{v_i v_{i-1}\}$. Assume that we start traversing P^* at time $t_{u_2} = t_{v_2}$. Then we arrive at v_i at time $x\Delta + g$, and we continue by traversing edge $v_i v_{i-1}$, either at time $(x+1)\Delta + e$ (in the case where $g = e$) or at time $x\Delta + e$ (in the case where $g \neq e$). That is, $d(u, v_{i-1}, P^*) \geq x\Delta + e - t_{v_2} + 1$.

Similarly to Case 1, let $h \in [1, \Delta]$ be such that $A_{t_{v_i}}(u, v_{i-1}, Q^*) = x\Delta + e - h$. That is, if we start traversing Q^* at time t_{v_2}, we arrive at v_{i-1} at time $x\Delta + e - h$, i. e., $d(u, v_{i-1}, Q^*) = x\Delta + e - h - t_{v_1} + 1$. Summarizing, we have:

$$d(u, v_{i-1}, P^*) - d(u, v_{i-1}, Q^*) \geq h \geq 1,$$

APPENDIX

which proves the statement of the lemma. ◀

From the above it follows that if P is a fastest path from u to v, then all vertices of P, with the exception of vertices of interest $v_i \in P \setminus \{u, v\}$, are reached using the same path P. We use this fact in the following proof.

Proof of Lemma 32. For every vertex of interest $v_i \in U \cap (P \setminus \{u, v\})$ we have two options. First, when the fastest temporal path P' from u to v_i is a subpath of P. In this case we determine the labeling of P' using Lemma 31. Second, when the fastest temporal path P' from u to v_i is not a subpath of P. In this case we know exactly how to label all of the edges of P, with the exception of edges of from $v_{i-1}v_i$, that are incident to v_i in P. ◀

▶ **Lemma 36.** *Suppose that $S_{u,v}, S_{w,z}$ are two segments with $v_i \in S_{u,v}$ and $z_j \in S_{w,z}$, where z_j is a split vertex of v_i in the segment $S_{w,z}$. W.l.o.g. suppose that the fastest temporal path from v_i to z_j travels through vertices u and w. Then the fastest temporal path from v_i to any other vertex of $S_{w,z}$, that is closer to w, travels through the same two vertices u and w. Similarly it holds for the cases when the fastest temporal path travels through w, v or z, u or z, v.*

Proof. Let z_ℓ be a vertex of $S_{w,z}$, that is closer to w than z in the segment. Let us denote with P_{v_i,z_j} the underlying path of the fastest temporal path from v_i to z_j. Denote with $P^\ell_{v_i,z_j}$ the subpath of the fastest temporal path from v_i to z_j, that terminates in z_ℓ. We want to show that $P^\ell_{v_i,z_j}$ is an underlying path of a fastest temporal path from v_i to z_j. Let us observe the following possibilities.

First, suppose for the contradiction, that the fastest temporal path from v_i to z_ℓ travels through vertices u and z. Denote this path as $P^1_{v_i,z_\ell}$. Then it follows that $d(P^1_{v_i,z_\ell}, \lambda) \leq d(P^\ell_{v_i,z_j}, \lambda)$, which would imply that the duration of the temporal path from v_i to z_j using the subpath of $P^1_{v_i,z_\ell}$, would be strictly smaller than the duration of (P_{v_i,z_j}, λ), which cannot be possible.

Second, suppose that the fastest temporal path from v_i to z_ℓ travels through vertices v and w. Denote this path as $P^2_{v_i,z_\ell}$. Note that $P^\ell_{v_i,z_j}$ and $P^2_{v_i,z_\ell}$ intersect on a segment $S_{w,z}$ from the vertex w to z_ℓ. Therefore since $d(P^2_{v_i,z_\ell}, \lambda) \leq d(P^\ell_{v_i,z_j}, \lambda)$, and since there is unique way to extend the path $P^2_{v_i,z_\ell}$ from z_ℓ to z_j, denote the extended path as $P^j_{v_i,z_\ell}$, we get that $d(P^j_{v_i,z_\ell}, \lambda) \leq (P_{v_i,z_j}, \lambda)$. Which implies that $d(P^j_{v_i,z_\ell}, \lambda) = d(P_{v_i,z_j}, \lambda)$. Now using the similar argument it follows that $d(P^\ell_{v_i,z_j}, \lambda) = d(P^2_{v_i,z_\ell}, \lambda)$, therefore $P^\ell_{v_i,z_j}$ is also a fastest temporal path from v_i to z_j.

Third, suppose that the fastest temporal path from v_i to z_ℓ travels through vertices v and z. Denote this path as $P^3_{v_i,z_\ell}$. Then the duration of the temporal path from v_i to z_j using the subpath of $P^3_{v_i,z_\ell}$, would be strictly smaller than the duration of (P_{v_i,z_j}, λ), which cannot be possible. ◀

▶ **Lemma 37.** *Let $S_{u,v}$ be a segment in G of length at least 5, i. e., $S_{u,v} = (u = v_1, v_2, \ldots, v_p = v)$, where $p > 5$. It cannot happen that an inner edge $f = v_iv_{i+1}$ from $S_{u,v} \setminus \{u, v\}$, is not a part of any fastest temporal path, of length at least 2, between vertices in $S_{u,v}$, i. e., there has to be a pair $v_j, v_{j'} \in S_{u,v}$ s. t., the fastest temporal path from v_j to $v_{j'}$ passes through edge f. In the case when $p = 5$ all temporal paths of length 2 avoid f if and only if f has the same label as both of the edges incident to it.*

Proof. For an easier understanding and better readability we present the proof for $S_{u,v}$ of fixed length 5. The case where $S_{u,v}$ is longer easily follows from presented results.

APPENDIX

Let $S_{u,v} = (u = v_1, v_2, v_3, v_4, v_5, v_6 = v)$. We distinguish two cases, first that $f = v_2v_3$ (note that the case with $f = v_4v_5$ is symmetrical), and the second that $f = v_3v_4$. Throughout the proof we denote with t_i the label of edge v_iv_{i+1}. Suppose for the contradiction, that none of the fastest temporal paths between vertices of $S_{u,v}$ traverses the edge f.

Case 1: $f = v_2v_3$. Let us observe the case of fastest temporal paths between v_1 and v_3. Denote with $Q = (v_1, v_2, v_3)$ and with $P' = (v_3, v_4, v_5, v_6)$. From our proposition it follows that

- the fastest temporal path P^+ from v_1 to v_3 is of the following form $P^+ = v_1 \rightsquigarrow v_6 \to v_5 \to v_4 \to v_3$, and
- the fastest temporal path P^- from v_3 to v_1 is of the following form $P^- = v_3 \to v_4 \to v_5 \to v_6 \rightsquigarrow v_1$.

It follows that $d(v_1, v_3, P^+) \leq d(v_1, v_3, Q)$, and $d(v_1, v_3, P^-) \leq d(v_1, v_3, Q)$. Let Note that $d(v_1, v_3, P^+) \geq 1 + d(v_6, v_3, P')$, and by the definition $d(v_6, v_3, P') = 1 + (t_4 - t_5)_\Delta + (t_3 - t_4)_\Delta$, where $(t_i - t_j)_\Delta$ denotes the difference of two consecutive labels t_i, t_j modulo Δ. Similarly holds for $d(v_1, v_3, P^+)$. Summing now both of the above equations we get

$$\begin{aligned} d(v_1, v_3, P^+) + d(v_3, v_1, P^-) &\leq d(v_1, v_3, Q) + d(v_3, v_1, Q) \\ 1 + d(v_6, v_3, P') + 1 + d(v_3, v_6, P') &\leq d(v_1, v_3, Q) + d(v_3, v_1, Q) \\ 3 + (t_4 - t_5)_\Delta + (t_3 - t_4)_\Delta + 1 + (t_4 - t_3)_\Delta + (t_5 - t_4)_\Delta &\leq 1 + (t_2 - t_1)_\Delta + 1 + (t_1 - t_2)_\Delta \\ (t_4 - t_5)_\Delta + (t_5 - t_4)_\Delta + (t_4 - t_3)_\Delta + (t_3 - t_4)_\Delta + 2 &\leq (t_2 - t_1)_\Delta + (t_1 - t_2)_\Delta. \end{aligned} \tag{3}$$

Note that if $t_i \neq t_j$ we get that the sum $(t_i - t_j)_\Delta + (t_j - t_i)_\Delta$ equals exactly Δ, and if $t_i = t_j$ the sum equals 2Δ. This follows from the definition of travel delays at vertices (see Observation 29). Therefore we get from Equation (3), that the right part is at most 2Δ, while the left part is at least $2\Delta + 1$, for any relation of labels $t_1, t_2, \ldots, t_5$, which is a contradiction.

Case 2: $f = v_3v_4$. Here we consider the fastest paths between vertices v_2 and v_4. By similar arguments as above we get

$$(t_5 - t_1)_\Delta + (t_4 - t_5)_\Delta + (t_5 - t_4)_\Delta + (t_1 - t_5)_\Delta + 2 \leq (t_3 - t_2)_\Delta + (t_2 - t_3)_\Delta,$$

which is impossible.

In the case when $S_{u,v}$ is longer, we would get even bigger number on the left hand side of Equation (3), so we conclude that in all of the above cases, it cannot happen that all fastest paths of length 2, between vertices in S_u, v, avoid an edge f.

Let us observe now the case when $S_{u,v} = (u = v_1, v_2, v_3, v_4, v_5 = v)$ is of length 4. Let $f = v_2v_3$ (the case with $f = v_3v_4$ is symmetrical). Suppose that the fastest temporal paths between v_1 and v_3 do not use the edge f. We denote with R^+ the fastest path from v_1 to v_3, which is of the form $u \rightsquigarrow v \to v_4 \to v_3$, and similarly with R^- the fastest path from v_3 to v_1, which is of the form $v_3 \to v_4 \to v \rightsquigarrow u$. We denote with $R' = (v_3, v_4, v_5)$ and with $S = (v_1, v_2, v_3)$. Again we get the following.

$$\begin{aligned} d(v_1, v_3, R^+) + d(v_3, v_1, R^-) &\leq d(v_1, v_3, S) + d(v_3, v_1, S) \\ 1 + d(v_5, v_3, R') + 1 + d(v_3, v_5, R') &\leq d(v_1, v_3, S) + d(v_3, v_1, S) \\ (t_3 - t_4)_\Delta + (t_4 - t_3)_\Delta + 2 &\leq (t_2 - t_1)_\Delta + (t_1 - t_2)_\Delta. \end{aligned}$$

The only case when the equation has a valid solution is when $t_1 = t_2$ and $t_3 \neq t_4$, since in this case the left hands side evaluates to $\Delta + 2$, while the right side evaluates to 2Δ.

APPENDIX

Repeating the analysis for the fastest paths between v_2 and v_4, we get that the only valid solution is when $t_2 = t_3$ and $t_1 \neq t_4$. Altogether, we get that f is not in a fastest path of length 2 in $S_{u,v}$ if and only if the label of edge f is the same as the labels on the edges incident to it, while the last remaining edge has a different label. ◀

3.2.4 Adding constraints and variables to the ILP

We start by analyzing the case where we want to determine the labels on fastest temporal paths between vertices of interest. We proceed in the following way. Let $u, v \in U$ be two vertices of interest and let $P_{u,v}$ be the fastest temporal path from u to v. If $P_{u,v}$ is a segment we determine all the labels of edges of $P_{u,v}$, with respect to the label of the first edge (see Lemma 31). In the case when $P_{u,v}$ is a sequence of ℓ segments, we determine all but $\ell - 1$ labels of edges of $P_{u,v}$, with respect to the label of the first edge (see Lemma 32). We call these $\ell - 1$ edges, *partially determined* edges. After repeating this step for all pairs of vertices in U, the edges of fastest temporal paths from u to v, where $u, v \in U$, are determined with respect to the label of the first edge of each path, or are partially determined. If the fastest temporal path between two vertices $u, v \in U$ is just an edge e, then we treat it as being determined, since it gets assigned a label $\lambda(e)$ with respect to itself. All other edges in G' are called the *not yet determined* edges. Note that the not yet determined edges are exactly the ones that are not a part of any fastest temporal path.

Now we want to relate the not yet determined segments with the determined ones. Let $S_{u,v}$ and $S_{w,z}$ be two segments. At the beginning we have guessed the fastest path from v_i to all vertices in $S_{w,z}$ (see guess G-9). We did this by determining which vertices z_j, z_{j+1} in $S_{w,z}$ are furthest away from v_i (remember we can have the case when $z_j = z_{j+1}$), and then we guessed how the path from v_i leaves the segment $S_{u,v}$ (i. e., either through the vertex u or v), and then how it reaches z_j (in the case when $z_j \neq z_{j+1}$ there is a unique way, when $z_j = z_{j+1}$ we determined which of the vertices w or z is on the fastest path). W.l.o.g. assume that we have guessed that the fastest path from v_i to z_j passes through w and z_{j-1}. Then the fastest temporal path from v_i to z_{j+1} passes through z. And all fastest temporal paths from v_i to any $z_{j'} \in S_{w,z}$ use all of the edges in $S_{w,z}$ with the exception of the edge $z_j z_{j+1}$. Using this information and Observation 30, we can determine the labels on all edges, with respect to the first or last label from the segment $S_{u,v}$, with the exception of the edge $z_j z_{j+1}$. Therefore, all edges of $S_{w,z}$ but $z_j z_{j+1}$ become determined. Since we repeat that procedure for all pairs of segments, we get that for a fixed segment $S_{w,z}$ we end up with a not yet determined edge $z_j z_{j+1}$ if and only if this is a not yet determined edge in relation to every other segment $S_{u,v}$ and its fixed vertex v_i. We repeat this procedure for all pairs of segments. Each specific calculation takes linear time, since there are $O(k^2)$ segments, this calculation takes $O(k^4)$ time. At this point the edges of every segment are fully determined, with the exception of at most three edges per segment (the first and last edge and potentially one extra somewhere in the segment). We will now relate also these edges. More precisely, let $S_{u,v} = (u = v_1, v_2, \dots, v_p = v)$ be a segment with three not yet determined edges, and let $e_i = v_i v_{i+1}$ denote an edge of $S_{u,v}$. From the above procedure (when we were determining labels of edges of segments with each other) we conclude that all of the edges e_i of $S_{u,v}$ are in the following relation. There are some edges $e_1, e_2, \dots e_{i-1}$, whose label is determined with respect to the label $\lambda(e_1)$, we have an edge $f = e_i = v_i v_{i+1}$ which is not yet determined, and then there follow the edges $e_{i+1}, e_{i+2}, \dots, e_{p-1}$, whose labels are determined with respect to the $\lambda(e_{p-1})$. We want to now determine all of the edges in such segment $S_{u,v}$ with respect to just one edge (either the first or the last one). For this we use the fact that at least one of the temporal paths between vertices in $S_{u,v}$ has to pass through f, when $S_{u,v}$ has at least 5 edges (see Lemma 37). We proceed as follows.

G-11. Let $S_{u,v} = (u = v_1, v_2, \ldots, v_p = v)$ be a segment with a not yet determined edge $v_i v_{i+1} = f \in S_{u,v}$. For the fastest temporal paths from v_{i-1} to v_{i+1}, from v_{i+1} to v_{i-1}, from v_i and v_{i+2}, and v_{i+2} and v_i, we guess whether it passes through the edge f. Thus we create 4 guesses for every such segment $S_{u,v}$, therefore we create $O(k^2)$ new guesses in total, as there are at most $O(k^2)$ segments.

Once we know which two fastest temporal paths pass through f, we can determine the label of edge f with respect to the first edge of the segment (when considering the fastest temporal path between v_{i-1} and v_{i+1}), and with respect to the last edge of the segment (when considering the fastest temporal path between v_i and v_{i+2}). Both these steps together determine all of the labels of $S_{u,v}$ with respect to just one label. Note, from Lemma 37 it follows that the above procedure holds only for segments with at least 5 edges. In the case when segment has 4 edges, it can happen that the fastest temporal paths from above, do not traverse f. But in this case we get that 3 labels of edges in the segment have to be the same, while one is different, which results in a segment with two not yet determined edges. In the case when the segment $S_{u,v}$ has just three, two or one edge, this procedure does not improve anything, therefore these segments remain with three, two or one not yet determined edges, respectively. From now on we refer to segments of length less than 4.

At this point G is a graph, where each edge e has a value for its label $\lambda(e)$ that depends on (i. e., is a function of) some other label $\lambda(f)$ of edge f, or it depends on no other label. We now describe how we create variables and start building our ILP instances. For every edge e in G' that is incident to a vertex of interest we create a variable x_e that can have values from $\{1, 2, \ldots, \Delta\}$. Besides that we create one variable for each edge that is still not yet determined on a segment. Since each vertex of interest is incident to at most k edges, and each segment has at most one extra not yet determined edge, we create $O(k^2)$ variables. At the end we create our final guess.

G-12. We guess the permutation of all $O(k^2)$ variables, together with the relation of each variable to the labels of edges incident to these not yet determined edges. Namely, for an edge e that is not yet determined, we set its value to x_e and check labels of all of its neighbors, which are determined by some other label, and variables of the not yet determined neighbors, and guess if x_e is smaller, equal or bigger than the labels of the edges of its neighbors. So, for any two variables x_e and x_f, we know if $x_e < x_f$ or $x_e = x_f$, or $x_e > x_f$, and for any neighboring edge g of e we know if $x_e < \lambda(g)$ or $x_e = \lambda(g)$, or $x_e > \lambda(g)$. This results in $O(k^2)! = k^{O(k^2)}$ guesses and consequently each of the ILP instances we created up to now is further split into $k^{O(k^2)}$ new ones.

We have now finished creating all ILP instances. From Section 3.2.2 we know the structure of all guessed paths, to which we have just added also the knowledge of permutation of all variables. We proceed with adding constraints to each of our ILP instances. First we add all constraints for the labels of edges that we have determined up to now. We then continue to iterate through all pairs of vertices and start adding equality (resp. inequality) constraints for the fastest (resp. not necessarily fastest) temporal paths between them.

We now describe how we add constraints to a path. Whenever we say that a duration of a path gives an equality or inequality constraint, we mean the following. Let $P = (u = v_1, v_2, \ldots, v_p = v)$ be the underlying path of a fastest temporal path from u to v, and let $Q = (u = z_1, z_2, \ldots, z_r = v)$ be the underlying path of another temporal path from u to v. Then we know that $d(P, \lambda) = D_{u,v}$ and $d(Q, \lambda) \geq D_{u,v}$. Using Observation 28 we create an *equality constraint* for P of the form

$$D_{u,v} = \sum_{i=2}^{p-1} (\lambda(v_i v_{i+1}) - \lambda(v_{i-1} v_i))_\Delta + 1, \tag{4}$$

and an *inequality constraint* for Q

$$D_{u,v} \leq \sum_{i=2}^{r-1} (\lambda(z_i z_{i+1}) - \lambda(z_{i-1} z_i))_\Delta + 1. \tag{5}$$

In both cases we implicitly assume that if the difference of $(\lambda(z_i z_{i+1}) - \lambda(z_{i-1} z_i))$ is negative, for some i, we add the value Δ to it (i. e., we consider the difference modulo Δ), therefore we have the sign Δ around the brackets. Note that we know if the difference of two consecutive labels is positive or negative. In the case when two consecutive labels are determined with respect to the same label $\lambda(e)$ the difference between them is easy to determine, if one or both consecutive labels are not yet determined then we have guessed in what kind of relation they are (see guess **G-12**). Therefore we know when Δ has to be added, which implies that Equations (4) and (5) are calculated correctly for all paths.

We iterate through all pairs of vertices x, y and make sure that the fastest temporal path from x to y produces the equality constrain Equation (4), and all other temporal paths from x to y produce the inequality constraint Equation (5). For each pair we argue how we determine these paths.

Fastest paths between $u, v \in U$. Let $u, v \in U$, i. e., both u, v are vertices of interest. For the path from u to v (resp. from v to u) in G', which we guessed that it coincides with the fastest in **G-1**, we introduce an equality constraint. We then iterate over all other paths from u to v (resp. from v to u) in G', and for each one we introduce an inequality constraint. There are $O(k^2)$ pairs of vertices $u, v \in U$, and there are $k^{O(k)}$ possible paths from u to v (resp. from v to u) in G', therefore in this step we introduce $k^{O(k^3)}$ constraints in total.

Fastest paths from $u \in U$ to $x \in V(G') \setminus U$. From the guesses **G-8** and **G-10** we know the fastest temporal paths from u to all vertices in a segment $S_{w,v}$. In this case we create an equality constraint for the fastest path and we iterate through all other paths, for which we introduce the inequality constraints. There are $k^{O(k)}$ possible paths of the form $u \rightsquigarrow w$ (resp. $u \rightsquigarrow v$), and a unique way how to extend these paths from w (resp. v) to reach x in $S_{w,v}$. Therefore we add $k^{O(k)}$ inequality constraints for the pair u, x.

Fastest paths from $x \in V(G') \setminus U$ to $u \in U$. Let x be a vertex in the segment $S_{w,z} = (w = z_1, z_2, \dots, z_r = z)$, and let $u \in U$. If $S_{w,z}$ is of length 3 or less, then we already know the fastest temporal path from every vertex in the segment to u (since $S_{w,z}$ has at most 2 inner vertices, we determined the fastest temporal paths from them to u in guess **G-4**). Assume that $S_{w,z}$ is of length at least 4. This implies that there are at most two not yet determined edges in it. We can easily compute the vertices $z_i, z_{i+1} \in S_{w,z} \setminus \{w, z\}$ for which the fastest temporal path from z_i to u has the biggest duration. Denote with P^+ the fastest temporal path of the form $z_2 \to z \rightsquigarrow u$, and with P^- the fastest temporal path of the form $z_{r-1} \to w \rightsquigarrow u$. Note that we know these paths from guess **G-8**. Then we know that all vertices z_j in $S_{w,z} \setminus \{z_i, z_{i+1}\}$ that are closer to w than z_i, z_{i+1} reach u on the following fastest temporal path $(z_j \to z_{j-1} \to \dots \to z_2) \cup P^+$ and all the vertices z_j in $S_{w,z} \setminus \{z_i, z_{i+1}\}$ that are closer to z than z_i, z_{i+1} reach u on the following fastest temporal

APPENDIX

path $(z_j \to z_{j+1} \to \cdots \to z_{r-1}) \cup P^-$. Since the first part of fastest paths is unique, and we know the second part is the fastest, the above paths are the fastest temporal paths. We still have to determine the fastest temporal paths from z_i, z_{i+1} to u. We distinguish the following two options.

(i) $z_i \neq z_{i+1}$. Then the fastest temporal path from z_i to u is $(z_i \to z_{i-1} \to \cdots \to z_2) \cup P^+$, and the fastest temporal path from z_{i+1} to u is $(z_{i+1} \to z_{i+2} \to \cdots \to z_{r-1}) \cup P^-$.

(ii) $z_i = z_{i+1}$, i. e., let z_i be the unique vertex, that is furthest away from u in $S_{w,z}$. In this case we have to determine if the fastest temporal path from z_i to u, goes first through vertex z_{i-1} (and then through w), or it goes first through z_{i+1} (and then through z). Since we know the values $D_{z_{i-1},u}, D_{z_{i+1},u}$, and since there are at most two not yet determined labels in $S_{w,z}$, we can uniquely determine one of the following waiting times: the waiting time $\tau_{v_{i-1}}^{v_i,v_{i-2}}$ at vertex v_{i-1} when traveling from v_i to v_{i-2}, or the waiting time $\tau_{v_{i+1}}^{v_i,v_{i+2}}$ at vertex v_{i+1} when traveling from v_i to v_{i+2}. Suppose we know the former (for the latter case, analysis follows similarly). Then we set $c = D_{z_{i-1},u} + \tau_{v_{i-1}}^{v_i,v_{i-2}}$. We now compare c and the value $D_{z_i,u}$. If $c < D_{z_i,u}$ we conclude that our ILP has no solution and we stop with calculations, if $c = D_{z_i,u}$ then the fastest temporal path from z_i to u is of the form $(z_i \to z_{i-1} \to \cdots \to z_2) \cup P^+$, if $c > D_{z_i,u}$ then the fastest temporal path from z_i to u is of the form $(z_i \to z_{i+1} \to \cdots \to z_{r-1}) \cup P^-$.

Once the fastest temporal path from c to u is determined, we introduce an equality constraint for it. For each of the other $k^{O(k)}$ paths from x to u (which correspond to all paths of the form $w \rightsquigarrow u$ and $z \rightsquigarrow u$, together with the unique subpath on $S_{w,z}$), we introduce an inequality constraint. Therefore we add $k^{O(k)}$ inequality constraints for the pair x, u.

Fastest paths between $x, y \in V(G') \setminus U$. Let $x, y \in V(G') \setminus U$. We have two options.

(i) Vertices x, y are in the same segment $S_{u,v} = (u, v_1, v_2, \ldots, v_p, v)$. If the length of $S_{u,v}$ is less than 4 then we know what is the fastest path between vertices. Suppose now that $S_{u,v}$ is of length at least 5. Then there are at most two not yet determined edges in $S_{u,v}$.

W.l.o.g. suppose that x is closer to u in $S_{u,v}$ than y. Denote with $x = v_i$. Let $v_k \in S_{u,v}$ be a vertex for which the duration from v_i is the biggest (note that in the case when we have two such vertices, v_k and v_{k+1} we know exactly what are the fastest paths from x to every vertex in $S_{u,v}$, by similar arguing as in case (i) from above, when we were determining the fastest path from $x \in V(G')$ to $u \in U$. Then we know that $D_{x,v_{i+1}} < D_{x,v_{i+2}} < \cdots < D_{x,v_k}$ and $D_{x,v_{i-1}} < D_{x,v_{i-2}} < \cdots < D_{x,v_k}$, where indices are taken modulo p. Therefore we know exactly the structure of all the fastest paths from x to every vertex in $S_{u,v}$, with the exception of the fastest path from x to v_k. Since there is at most one undetermined edge in $S_{u,v}$, and since we know the exact durations $D_{x,v_{k-1}}$ and $D_{x,v_{k+1}}$, we can determine either $c = D_{x,v_{k-1}} + \tau_{v_k-1}^{v_k,v_{k-2}}$ or $c' = D_{x,v_{k+1}} + \tau_{v_k+1}^{v_k,v_{k+2}}$. We then compare (one of) these values to D_{x,v_k} which then uniquely determines the fastest temporal path from x to v_k (for details see case (ii) from above, when we were determining the fastest path from $x \in V(G') \setminus U$ to $u \in U$).

(ii) Vertices x and y are in different segments. Let x be a vertex in the segment $S_{u,v} = (u = v_1, v_2, \ldots, v_p = v)$ and let y be a vertex in the segment $S_{w,z} = (w = z_1, z_2, z_3, \ldots, z_r = z)$. By checking the durations of the fastest paths from x to every vertex in $S_{w,z} \setminus \{w, z\}$ we can determine the vertex $z_i \in S_{w,z}$, for which the duration from x is the biggest. Note that if there are two such vertices z_i and z_{i+1}, we know exactly how all fastest temporal paths enter $S_{w,z}$ (we use similar arguing as in case (i) from above, when we were determining the fastest path from $x \in V(G')$ to $u \in U$). This implies that the

APPENDIX

fastest temporal paths from x to all vertices $z_2, z_3, \dots, z_{i-1}$ (resp. $z_{i+1}, z_{i+2}, \dots, z_{r-1}$) pass through w (resp. z). Now we determine the vertex $v_j \in S_{u,v} \setminus \{u, v\}$, for which the value of the durations of the fastest paths from it to the vertex y is the biggest. Again, if there are two such vertices v_j and v_{j+1} we know exactly how the fastest temporal paths, starting in these two vertices, leave the segment $S_{u,v}$. We use similar arguing as in case (i) from above, when we were determining the fastest path from $x \in V(G')$ to $u \in U$. Knowing the vertex v_j implies that the fastest temporal paths from the vertices $v_2, v_2, \dots, v_{j-1}$ (resp. $v_{j+1}, v_{j+2}, \dots, v_{p-1}$) to the vertex y passes through u (resp. v). Since we know the following fastest temporal paths (see guess **G-7**) $z_2 \to w \rightsquigarrow u \to v_2$, $z_2 \to w \rightsquigarrow v \to v_{p-1}$, $z_{r-1} \to z \rightsquigarrow v \to v_{p-1}$ and $z_{r-1} \to z \rightsquigarrow v \to v_{p-1}$, we can uniquely determine all fastest temporal paths from $x \neq v_j$ to any $y \in S_{u,v} \setminus \{z_i\}$.
We have to now consider the case when $x = v_j$ and $y = z_i$. If at least one of the segments $S_{u,v}$ and $S_{w,z}$ is of length more than 5, then this segment has no inner edges with not yet determined labels and we can uniquely determine the fastest path from v_j to z_i, using similar arguing as in case (ii) from above, when we were determining the fastest path from $x \in V(G')$ to $u \in U$. If at least one of them is of length 3 or less, we can again uniquely determine the fastest path from v_j to z_i, using the same approach, and the knowledge of fastest paths to (or from) all vertices of the segment of length 3 (as we guessed them in guess **G-7**). If both segments are of the length 4, then we know how all vertices reach each other, as we guessed the fastest paths in guesses **G-7** and **G-9**.

Once the fastest path is determined we introduce the equality constraint for it and iterate through all other paths, for which we introduce inequality constraints. To enumerate all these non-fastest temporal paths, we just consider all possible paths $u \rightsquigarrow w$, where u and w are the vertices of interest that are the endpoints of segments to which x and y belong; once the correct segment is reached, there is a unique path to the desired vertex x (resp. y). Therefore we introduce $k^{O(k)}$ inequality constraints for each pair of vertices x, y.

Considering the paths for vertices from Z. All of the above is enough to determine the labeling λ of G'. Now we have to make sure that the labeling considers also the vertices in Z that we initially removed from G. Remember that removed vertices form disjoint trees in G. Let us denote Z as the set of disjoint trees, i. e., $Z = T_1 \cup T_2 \cup \dots \cup T_t$, where T_i represents one of the trees. Since there is a unique (static) path between any two vertices z_1, z_2 in a tree T_i, it follows that there is also a unique (therefore also the fastest) temporal path between them. Thus determining the label of an edge in T_i uniquely determines the labels on all other edges of tree T_i. Let us describe now how to determine the labels on edges of an arbitrary $T_i \in Z$. Recall that for every tree T_i there is a representative vertex v_i of T_i, and a clip vertex $u_i \in V(G')$, such that $v_i \in N_G(u_i)$. To determine the correct label of all edges of T_i we use the following property.

▶ **Lemma 38.** *Let T_i be a tree in Z and let $e_i = (u_i, r_i)$ be an edge in G, where $u_i \in V(G')$ is a clip vertex of T_i and $r_i \in T_i$ is a representative of T_i. Let $v \in N_{G'}(u_i)$ be the closest vertex to r_i, regarding the values of D, i. e., $D_{r_i,v} \leq D_{r_i,w}$ for all $w \in N_{G'}(u)$. Then the path $P^* = (r_i, u_i, v)$ has to be the fastest temporal path from r_i to v in G.*

Proof. Suppose that this is not true. Then there exists a faster path P_2^* from r_i to v, that goes through the clip vertex u_i of T_i (as this is the only neighbor of r_i), through some other vertex $w \in N_{G'}(u) \setminus \{v\}$, and through some other path P' in G, before it finishes in v, where P' is at least an edge (from w to v). Therefore $P_2^* = (r_i, u_i, w, P', v)$, where $d(P_2^*) \leq d(P^*)$.

APPENDIX

Now since $D_{r_i,v} \leq D_{r_i,w}$ for all $w \in N_{G'}(u)$ the first part of path P_2^* from r_i to w takes at least $D_{r_i,v}$ time. Since $v \neq w$ we need at least one more time-step (one more edge) to traverse from w to reach v. So $d(P_2^*) \geq D_{r_i,v} + 1$, and so P_2^* cannot be faster than P^*. ◀

Suppose now that we know that (r_i, u_i, v) is the fastest temporal path from the representative r_i of T_i to the vertex v in the neighborhood of the clip vertex v_i of T_i. Then we can determine the label of edge $r_i u_i$ as $\lambda(r_i u_i) \equiv \lambda(u_i v) + 1 - D_{r_i,v} \pmod{\Delta}$. Now, using the algorithm for trees (see Theorem 22), we determine labels on all edges of T_i. We repeat this procedure for all trees in Z. What remains, is to add the equality (resp. inequality) constraints for the fastest (resp. non-fastest) temporal paths from vertices of Z to all other vertices in G' and vice versa. Note that since there is a unique path between vertices of tree T_i, and since all edges of tree T_i are determined with respect to the same label, we present the study for cases when we find fastest temporal paths from and to the representative vertex r_i of tree T_i. Each of these paths are then uniquely extended to all vertices in T_i.

Fastest paths from $z \in Z$ to $u \in U$. Let $u \in U$ and $z \in Z$. Then z belongs to a tree T_i, and let r_i be the representative of T_i. We distinguish the following two cases.

- **(i)** The clip vertex x of the tree T_i is not a vertex of interest. Let $x = z_i$ be a part of a segment $S_{w,z} = (w = z_1, z_2, \ldots, z_r = z)$, and denote with z_{i-1} and z_{i+1} the neighbouring vertices of x, where z_{i-1} is closer to w in $S_{w,z}$ and z_{i+1} is closer to z in $S_{w,z}$. From guess **G-8** we know the following fastest paths $z_2 \to w \rightsquigarrow u$ and $z_{r-1} \to z \rightsquigarrow u$. Denote thew with Q_1 and Q_r respectively. There are two options.
  - **(a)** The segment $S_{w,z}$ is of length at least 5 and has no not yet determined edges, with the exception of the first/last one. Which results in knowing all the waiting times at vertices of $S_{w,z}$ when traversing the segment. Then we also know that the labels of tree T_i edges are determined with respect to that same edge label. This results in knowing the value of waiting time $\tau_x^{r_i,z_{i-1}}$ at vertex x when traversing it from r_i to z_{i-1} and the value of waiting time $\tau_x^{r_i,z_{i+1}}$ at vertex x when traversing it from r_i to z_{i+1}. We also know the value $D_{x,u}$ and the underlying path of the fastest temporal path from x to u (which we determined in previous steps). W.l.o.g. suppose that the fastest path from x to u goes through z_{i-1} and uses the path Q_1. Denote with $P^- = (r_i, x, z_{i-1}, z_2) \cup Q_1$ and with $P^+ = (r_i, x, z_{i+1}, z_{r-1}) \cup Q_r$. Then we calculate the duration $d(P^-)$ as $d(P^-) = D_{x,u} + \tau_x^{r_i,z_i-1}$ and compare it to $D_{r_i,u}$. If $d(P^-) < D_{r_i,u}$ then we stop with the calculation and determine that our input graph has no solution. If $d(P^-) = D_{r_i,u}$ then we know that P^- is the underlying path of the fastest temporal path from r_i to u. If $d(P^-) > D_{r_i,u}$ then the fastest temporal path from r_i to u has to be P^+. For the fastest temporal path we introduce the equality constraint, for all other paths we introduce the inequality constraints. By similar arguing as in cases above, we introduce $k^{O(k)}$ inequality constraints.
  - **(b)** The segment $S_{w,z}$ is of length 4 or less and has an extra not yet determined edge p. If $p \cap \{x\} = \emptyset$, we can proceed with the same approach as above. So suppose now that $p = xz_{i+1}$. Then, from knowing that p is a not yet determined edge we conclude that all fastest temporal paths from x to any vertex of interest u' go through the edge $z_{i-1}x$, not trough p (this is true as if a fastest temporal path from x to some vertex of interest w' went through p, then p would be determined). Now, if the edges of tree are determined with respect to the label of the edge $z_{i-1}x$ (not p), we use the same approach as above to determine the fastest temporal path from r_i to u. Therefore, suppose that the edges of the tree T_i are determined with respect to the label of the

APPENDIX

edge p. Which means that $D_{r_i,z_{i+1}} < D_{r_i,z_{i-1}}$. We want to now determine if the fastest temporal path from r_i to u is of the form $r_i \to x \to z_{i-1} \to \cdots \to w \rightsquigarrow u$ or $r_i \to x \to z_{i+1} \to \cdots \to z \rightsquigarrow u$. We do the following. Denote with c the value $c = D_{r_i z_{i-1}} + D_{xu} + 1$. We now claim the following, note that we do not know what is the fastest temporal path from r_i to x_{i-1}. It can be of the form $P^- = (r_i, x, z_{i-1})$, or of the form $P^+ = (r_i, x, z_{i+1}, z_{i+2}, \ldots, z) \cup Q \cup (w, z_2, \ldots, z_{i-1})$, where Q is some path from z to w. Denote with R^x the underlying path of the fastest temporal path from x to u, and with R^{i-1} the underlying path of a fastest temporal path from z_{i-1} to u. Note $R^{i-1} \subseteq R^x$. Similarly, denote with R^{i+1} the underlying path of the fastest temporal path from z_{i+1} to u, for which we know it goes through the vertex z.

- If $c < D_{r_i,u}$, then we have a contradiction and we stop with the calculation. This is true since we have found a temporal path from r_i to u, with faster duration than the fastest temporal path from r_i to u, which cannot happen.
- If $c = D_{r_i,u}$, then the fastest temporal path is of the form $r_i \to x \to z_{i-1} \to \cdots \to w \rightsquigarrow u$.
  We have two options, first when the fastest temporal path from r_i to z_{i-1} is P^-. In this case we have determined that $P^- \cup R^{i-1}$ is the fastest temporal path from r_i to u. In the second case we suppose that the fastest temporal path from r_i to z_{i-1} is P^+. But then the duration of the path $P^+ \cup R^{i-1}$ from r_i to u equals the duration of the fastest path from r_i to u. But note that $P^+ \cup R^{i-1}$ is actually not a path but a walk, since there is repetition of edges between w and z_{i-1}, therefore it includes a path from r_i to u, which is even faster, a contradiction. Therefore we get that in this case P^- is always the underlying path of the fastest path from r_i to z_{i-1}. And the fastest path from r_i to z_{i-1} is $P^- \cup R^{i-1}$.
- If $c > D_{r_i,u}$, then the fastest temporal path is of the form $r_i \to x \to z_{i+1} \to \cdots \to z \rightsquigarrow u$.
  We again have two options. First when the fastest temporal path from r_i to z_{i-1} is P^-. In this case we easily deduce that $P^- \cup R^{i-1}$ is not the underlying path of the fastest temporal path from r_i to u. And therefore it follows that the the underlying path of the fastest temporal path from r_i to u is $(r_i, x, z_{i+1}) \cup R^{i+1}$. In the second case, suppose that P^+ is the underlying path of the fastest temporal path from x_i to z_{i-1}. We want to now prove that the fastest temporal path from r_i to u travels through vertices $z_{i+1}, z_{i+2}, \ldots z$. Suppose for the contradiction that this is not true. Then $S = (r_i, x, z_{i-1}) \cup R^{i-1}$ is the underlying path of the fastest temporal path from r_i to u. Then we get that the duration $d(S)$ of S equals to $D_{r_i,u}$. Let $D(r_i, z_{i-1}, S)$ be the duration of the temporal path from r_i to z_{i-1} along the path S. By the definition we get that $d(S) = D(r_i, z_{i-1}, S) + D_{x,u} - 1$. From this it follows that $D(r_i, z_{i-1}, S) = D_{r_i,z_{i-1}}$, which is in contradiction with our assumption. Therefore we get that in this case $(r_i, x, z_{i+1}) \cup R^{i+1}$ is always the underlying path of the fastest path from r_i to z_{i-1}.

In all of the cases, we have uniquely determined the underlying path of the fastest temporal path from r_i to u, which gives us an equality constraint. For all other $k^{O(k)}$ paths we add the inequality constraints.

(ii) The clip vertex w of the tree T_i is a vertex of interest. In this case we know exactly the fastest path from a representative vertex r_i to u (we determined it in guess **G-8**). We create an equality constraint for this path, and create inequality constraints for all other paths. Since there are $k^{O(k)}$ possible paths from w to u, and there is a unique path (edge) from r_i to w, we create $k^{O(k)}$ inequality constraints.

Fastest path from $u \in U$ to $z \in Z$. Let $u \in U$ and $z \in Z$. Then z belongs to a tree T_i, and let r_i be the representative of T_i. In this case we split again the analysis in two cases.

- The clip vertex $x \in S_{w,z}$ of T_i is not a vertex of interest. In this case we know the fastest paths from u to x, and to both of its neighbors x_i and x_j, on the segment $S_{w,z}$, which is enough to determine the exact fastest path from u to r_i (we use the same procedure as in the case i when determining the fastest paths from x to u).
- The clip vertex of T_i is a vertex of interest. In this case we know exactly what is the fastest path (see guess **G-8**).

The procedure produces one equality constraint (for the fastest path) and $k^{O(k)}$ inequality constraints.

Fastest path from $z \in Z$ to $y \in V(G') \setminus U$. Let $y \in V(G') \setminus U$ and $z \in Z$. Then z belongs to a tree T_i, and let r_i be the representative of T_i. Since y is not a vertex of interest it holds that $y \in S_{u,v} = (u = v_1, v_2, \ldots, v_p = v)$. We use the similar approach as in the above case, when we were determining paths between vertices in Z and U, to determine the fastest temporal path from r_i to y. The difference in this case is that we know the fastest temporal path from the clip vertex of T_i to y (when the clip vertex is not a vertex of interest), or we know the exact fastest path from the representative r_i of T_i to y (when the clip vertex is a vertex of interest). The latter follows from the guess **G-10**.

Since $y \in S_{u,v}$ there are $k^{O(k)}$ paths from the clip vertex to y, when the clip vertex is not a vertex of interest, and $k^{O(k)}$ paths from the clip vertex to y, when the clip vertex is a vertex of interest. In all of the cases, we can uniquely determine the underlying path of the fastest temporal path from r_i to y, which gives us an equality constraint. For all other paths we add the inequality constraints.

Fastest path from $y \in V(G') \setminus U$ to $z \in Z$. Let $y \in V(G') \setminus U$ and $z \in Z$. Then z belongs to a tree T_i, and let r_i be the representative of T_i. In this case we split again the analysis in two cases.

- The clip vertex $x \in S_{w,z}$ of T_i is not a vertex of interest. In this case we know the fastest paths from y to x, and to both of its neighbors x_i and x_j, on the segment $S_{w,z}$, which is enough to determine the exact fastest path from x to r_i (we use the same procedure as in the case i when determining the fastest paths from x to u).
- The clip vertex u_i of T_i is a vertex of interest. Let $y \in S_{u,v} = (u = v_1, v_2, \ldots, v_p = v)$. In this case we know the fastest paths from v_2 and v_{p-1} to r_i (see guess **G-8**). Since the path from y to v_2 (resp. v_{p-1}) is uniquely extended, we use the same procedure as in the case i, when determining the fastest paths from x to u, to determine which is the fastest path from y to r_i.

The procedure produces one equality constraint (for the fastest path) and $k^{O(k)}$ inequality constraints.

Fastest paths between z, z' where $z \in Z$ and $z' \in Z$. Let $z, z' \in Z$, and let z (resp. z') belong to the tree T_i (resp. T_j). Let r_i, r_j be representative vertices and let w_i, w_j be the clip vertices of the trees T_i and T_j, respectively. We distinguish following cases.

- Both clip vertices w_i, w_j are vertices of interest. In this case $r_i, r_j \in Z^*$ and therefore we know the fastest paths between them (see guess **G-8**).

- One clip vertex is a vertex of interest and the other is not. Let $w_i \in U$ and $w_j \in S_{u,v} \setminus \{u, v\}$. Denote with w_{j-1} and w_{j+1} the two neighbors of $w_j \in S_{u,v}$. We know the fastest paths from the representative r_i of tree T_i, to vertices w_{j-1}, w_j, w_{j+1} (we determined them in the above case when we were determining the fastest paths from $z \in Z$ to $y \in V(G') \setminus U$).

  Now we use the same procedure as in the case i, when determining the fastest paths from x to u, to determine the exact fastest path from r_i to r_j.

- None of the clip vertices w_i, w_j is a vertex of interest. Let $w_i \in S_{w,z}$ and $w_j \in S_{u,v} \setminus \{u, v\}$. Denote with w_{j-1} and w_{j+1} the two neighbors of $w_j \in S_{u,v}$, and similarly with w_{i-1} and w_{i+1} the two neighbors of $w_i \in S_{w,z}$. We know all the fastest paths from r_i to w_{j-1}, w_j, w_{j+1}, and similarly all fastest paths from r_j to w_{i-1}, w_i, w_{i+1}, together with all the fastest paths between each pair of the following vertices $w_{j-1}, w_j, w_{j+1} w_{i-1}, w_i, w_{i+1}$.

  Now we use the same procedure as in the case i, when determining the fastest paths from x to u, to determine the exact fastest path from r_i to r_j.

The procedure produces one equality constraint (for the fastest path) and $k^{O(k)}$ inequality constraints.

3.2.5 Solving ILP instances

All of the above finishes our construction of ILP instances. We have created $f(k)$ instances (where f is a double exponential function), each with $O(k^2)$ variables and $O(n^2)g(k)$ constraints (again, g is a double exponential function). We now solve each ILP instance I, using results from Lenstra [46], in the FPT time, with respect to k. If none of the ILP instances gives a positive solution, then there exists no labeling λ of G that would realize the matrix D (i. e., for any pair of vertices $u, v \in V(G)$ the duration of a fastest temporal path from u to v has to be $D_{u,v}$). If there is at least one I that has a valid solution, we use this solution and produce our labeling λ, for which (G, λ) realizes the matrix D. We have proven in the previous subsections that this is true since each ILP instance corresponds to a specific configuration of fastest temporal paths in the graph (i. e., considering all ILP instances is equivalent to exhaustively searching through all possible temporal paths between vertices). Besides that, in each ILP instance we add also the constraints for durations of all temporal paths between each pair of vertices. This results in setting the duration of a fastest path from a vertex $u \in V(G)$ to a vertex $v \in V(G)$ as $D_{u,v}$, and the duration of all other temporal paths from u to v, to be greater or equal to $D_{u,v}$, for all pairs of vertices u, v. Therefore, if there is an instance with a positive solution, then this instance gives rise to the desired labeling, as it satisfies all of the constraints. For the other direction we can observe that if there is a labeling λ meeting all duration requirements specified by D, then this labeling produces a specific configuration of fastest temporal paths. Since we consider all configurations, one of the produced ILP instances will correspond the configuration implicitly defined by λ, and hence our algorithm finds a solution.

To create the labeling λ from a solution X, of a positive ILP instance, we use the following procedure. First we label each edge e, that corresponds to the variable x_e by assigning the value $\lambda(e) = x_e$. We then continue to set the labels of all other edges. We know that the labels of all of the remaining edges depend on the label of (at least one) of the edges that were determined in previous step. Therefore, we easily calculate the desired labels for all remaining edges.

APPENDIX

4 Conclusion

We have introduced a natural and canonical temporal version of the graph realization problem with respect to distance requirements, called Simple periodic Temporal Graph Realization. We have shown that the problem is NP-hard in general and polynomial-time solvable if the underlying graph is a tree. Building upon those results, we have investigated its parameterized computational complexity with respect to structural parameters of the underlying graph that measure "tree-likeness". For those parameters, we essentially gave a tight classification between parameters that allow for tractability (in the FPT sense) and parameters that presumably do not. We showed that our problem is W[1]-hard when parameterized by the feedback vertex number of the underlying graph, and that it is in FPT when parameterized by the feedback edge number of the underlying graph. Note that most other common parameters that measure tree-likeness (such as the treewidth) are smaller than the vertex cover number.

We believe that our work spawns several interesting future research directions and builds a base upon which further temporal graph realization problems can be investigated.

Further parameterizations. There are several structural parameters which can be considered to obtain tractability which are either larger or incomparable to the feedback vertex number.

- The *vertex cover number* measures the distance to an independent set, on which we trivially only have no-instances of our problem. We believe this is a promising parameter to obtain tractability.
- The *tree-depth* measures "star-likeness" of a graph and is incomparable to both the feedback vertex number and the feedback edge number. We leave the parameterized complexity of our problem with respect to this parameter open.
- Parameters that measure "path-likeness" such as the *pathwidth* or the *vertex deletion distance to disjoint paths* are also natural candidates to investigate.

Furthermore, we can consider combining a structural parameter with Δ. Our NP-hardness reduction (Theorem 3) produces instances with constant Δ, so as a single parameter Δ cannot yield fixed-parameter tractability. However, in our parameterized hardness reduction (Theorem 4) the value for Δ in the produced instance is large. This implies that our result does not rule out e.g. fixed-parameter tractability for the combination of the treewidth and Δ as a parameter. We believe that investigating such parameter combinations is a promising future research direction.

Further problem variants. There are many natural variants of our problem that are well-motivated and warrant consideration. In the following, we give two specific examples. We believe that one of the most natural generalizations of our problem is to allow more than one label per edge in every Δ-period. A well-motivated variant (especially from the network design perspective) of our problem would be to consider the entries of the duration matrix D as upper-bounds on the duration of fastest paths rather than exact durations. Our work gives a starting point for many interesting future research directions such as the two mentioned examples.

References

1 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*, 61:907–944, 2017.

APPENDIX

2 Eleni C. Akrida, George B. Mertzios, Paul G. Spirakis, and Christoforos Raptopoulos. The temporal explorer who returns to the base. *Journal of Computer and System Sciences*, 120:179–193, 2021.

3 Emmanuel Arrighi, Niels Grüttemeier, Nils Morawietz, Frank Sommer, and Petra Wolf. Multi-parameter analysis of finding minors and subgraphs in edge-periodic temporal graphs. In *Proceedings of the 48th International Conference on Current Trends in Theory and Practice of Computer Science (SOFSEM)*, pages 283–297, 2023.

4 John Augustine, Keerti Choudhary, Avi Cohen, David Peleg, Sumathi Sivasubramaniam, and Suman Sourav. Distributed graph realizations. *IEEE Transactions on Parallel and Distributed Systems*, 33(6):1321–1337, 2022.

5 Amotz Bar-Noy, Keerti Choudhary, David Peleg, and Dror Rawitz. Efficiently realizing interval sequences. *SIAM Journal on Discrete Mathematics*, 34(4):2318–2337, 2020.

6 Amotz Bar-Noy, Keerti Choudhary, David Peleg, and Dror Rawitz. Graph realizations: Maximum degree in vertex neighborhoods. In *Proceedings of the 17th Scandinavian Symposium and Workshops on Algorithm Theory (SWAT)*, pages 10:1–10:17, 2020.

7 Amotz Bar-Noy, David Peleg, Mor Perry, and Dror Rawitz. Composed degree-distance realizations of graphs. In *Proceedings of the 32nd International Workshop on Combinatorial Algorithms (IWOCA)*, pages 63–77, 2021.

8 Amotz Bar-Noy, David Peleg, Mor Perry, and Dror Rawitz. Graph realization of distance sets. In *Proceedings of the 47th International Symposium on Mathematical Foundations of Computer Science (MFCS)*, pages 13:1–13:14, 2022.

9 Mehdi Behzad and James E Simpson. Eccentric sequences and eccentric sets in graphs. *Discrete Mathematics*, 16(3):187–193, 1976.

10 Robert E Bixby and Donald K Wagner. An almost linear-time algorithm for graph realization. *Mathematics of Operations Research*, 13(1):99–123, 1988.

11 Binh-Minh 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.

12 Arnaud Casteigts, Timothée Corsini, and Writika Sarkar. Invited paper: Simple, strict, proper, happy: A study of reachability in temporal graphs. In *Proceedings of the 24th International Symposium on Stabilization, Safety, and Security of Distributed Systems (SSS)*, pages 3–18, 2022.

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.

14 Arnaud Casteigts, Anne-Sophie Himmel, Hendrik Molter, and Philipp Zschoche. Finding temporal paths under waiting time constraints. *Algorithmica*, 83(9):2754–2802, 2021.

15 Wai-Kai Chen. On the realization of a (p, s)-digraph with prescribed degrees. *Journal of the Franklin Institute*, 281(5):406–422, 1966.

16 Fan Chung, Mark Garrett, Ronald Graham, and David Shallcross. Distance realization problems with applications to internet tomography. *Journal of Computer and System Sciences*, 63(3):432–448, 2001.

17 Joseph C. Culberson and Piotr Rudnicki. A fast algorithm for constructing trees from distance matrices. *Information Processing Letters*, 30(4):215–220, 1989.

18 Argyrios Deligkas and Igor Potapov. Optimizing reachability sets in temporal graphs by delaying. *Information and Computation*, 285:104890, 2022.

19 Jessica Enright, Kitty Meeks, George B. Mertzios, and Viktor Zamaraev. Deleting edges to restrict the size of an epidemic in temporal networks. *Journal of Computer and System Sciences*, 119:60–77, 2021.

20 Jessica Enright, Kitty Meeks, and Fiona Skerman. Assigning times to minimise reachability in temporal graphs. *Journal of Computer and System Sciences*, 115:169–186, 2021.

APPENDIX

21 Jessica A. Enright, Kitty Meeks, and Hendrik Molter. Counting temporal paths. *CoRR*, abs/2202.12055, 2022. URL: `https://arxiv.org/abs/2202.12055`.

22 Paul Erdős and Tibor Gallai. Graphs with prescribed degrees of vertices. *Mat. Lapok*, 11:264–274, 1960.

23 Thomas Erlebach, Michael Hoffmann, and Frank Kammer. On temporal graph exploration. *Journal of Computer and System Sciences*, 119:1–18, 2021.

24 Thomas Erlebach and Jakob T. Spooner. A game of cops and robbers on graphs with periodic edge-connectivity. In *Proceedings of the 46th International Conference on Current Trends in Theory and Practice of Informatics (SOFSEM)*, pages 64–75, 2020.

25 Michael R. Fellows, Danny Hermelin, Frances Rosamond, and Stéphane Vialette. On the parameterized complexity of multiple-interval graph problems. *Theoretical Computer Science*, 410(1):53–61, 2009.

26 Michael R. Fellows, Bart M. P. Jansen, and Frances A. Rosamond. Towards fully multivariate algorithmics: Parameter ecology and the deconstruction of computational complexity. *European Journal of Combinatorics*, 34(3):541–566, 2013.

27 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.

28 András Frank. Augmenting graphs to meet edge-connectivity requirements. *SIAM Journal on Discrete Mathematics*, 5(1):25–53, 1992.

29 András Frank. Connectivity augmentation problems in network design. *Mathematical Programming: State of the Art 1994*, 1994.

30 H. Frank and Wushow Chou. Connectivity considerations in the design of survivable networks. *IEEE Transactions on Circuit Theory*, 17(4):486–490, 1970.

31 Eugen Füchsle, Hendrik Molter, Rolf Niedermeier, and Malte Renken. Delay-robust routes in temporal graphs. In *Proceedings of the 39th International Symposium on Theoretical Aspects of Computer Science (STACS)*, pages 30:1–30:15, 2022.

32 Eugen Füchsle, Hendrik Molter, Rolf Niedermeier, and Malte Renken. Temporal connectivity: Coping with foreseen and unforeseen delays. In *Proceedings of the 1st Symposium on Algorithmic Foundations of Dynamic Networks (SAND)*, pages 17:1–17:17, 2022.

33 D.R. Fulkerson. Zero-one matrices with zero trace. *Pacific Journal of Mathematics*, 10(3):831–836, 1960.

34 Petr A. Golovach and George B. Mertzios. Graph editing to a given degree sequence. *Theoretical Computer Science*, 665:1–12, 2017.

35 Martin Charles Golumbic and Ann N. Trenk. *Tolerance Graphs.* Cambridge Studies in Advanced Mathematics. Cambridge University Press, 2004.

36 Ralph E Gomory and Tien Chung Hu. Multi-terminal network flows. *Journal of the Society for Industrial and Applied Mathematics*, 9(4):551–570, 1961.

37 Martin Grötschel, Clyde L Monma, and Mechthild Stoer. Design of survivable networks. *Handbooks in Operations Research and Management Science*, 7:617–672, 1995.

38 Jiong Guo, Falk Hüffner, and Rolf Niedermeier. A structural view on parameterizing problems: Distance from triviality. In *Proceedings of the 1st International Workshop on Parameterized and Exact Computation (IWPEC)*, pages 162–173, 2004.

39 S. Louis Hakimi. On realizability of a set of integers as degrees of the vertices of a linear graph. I. *Journal of the Society for Industrial and Applied Mathematics*, 10(3):496–506, 1962.

40 S. Louis Hakimi and Stephen S. Yau. Distance matrix of a graph and its realizability. *Quarterly of applied mathematics*, 22(4):305–317, 1965.

41 Pavol Hell and David Kirkpatrick. Linear-time certifying algorithms for near-graphical sequences. *Discrete Mathematics*, 309(18):5703–5713, 2009.

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

APPENDIX

43 Nina Klobas, George B. Mertzios, Hendrik Molter, Rolf Niedermeier, and Philipp Zschoche. Interference-free walks in time: Temporally disjoint paths. *Autonomous Agents and Multi-Agent Systems*, 37(1):1, 2023.

44 Nina Klobas, George B. Mertzios, Hendrik Molter, and Paul G. Spirakis. The complexity of computing optimum labelings for temporal connectivity. In *Proceedings of the 47th International Symposium on Mathematical Foundations of Computer Science (MFCS)*, pages 62:1–62:15, 2022.

45 Fabian Kuhn and Rotem Oshman. Dynamic networks: Models and algorithms. *SIGACT News*, 42(1):82–96, mar 2011.

46 Hendrik W. Lenstra. Integer programming with a fixed number of variables. *Mathematics of Operations Research*, 8:538–548, 1983.

47 Jure Leskovec and Andrej Krevl. SNAP Datasets: Stanford large network dataset collection. `http://snap.stanford.edu/data`, June 2014.

48 Linda Lesniak. Eccentric sequences in graphs. *Periodica Mathematica Hungarica*, 6:287–293, 1975.

49 Ross M. McConnell and Jeremy P. Spinrad. Construction of probe interval models. In *Proceedings of the 13th Annual ACM-SIAM Symposium on Discrete Algorithms (SODA)*, pages 866–875, 2002.

50 F.R. McMorris, Chi Wang, and Peisen Zhang. On probe interval graphs. *Discrete Applied Mathematics*, 88(1):315–324, 1998. Computational Molecular Biology DAM - CMB Series.

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

52 George B. Mertzios, Hendrik Molter, Malte Renken, Paul G. Spirakis, and Philipp Zschoche. The complexity of transitively orienting temporal graphs. In *Proceedings of the 46th International Symposium on Mathematical Foundations of Computer Science (MFCS)*, pages 75:1–75:18, 2021.

53 Hendrik Molter, Malte Renken, and Philipp Zschoche. Temporal reachability minimization: Delaying vs. deleting. In *Proceedings of the 46th International Symposium on Mathematical Foundations of Computer Science (MFCS)*, pages 76:1–76:15, 2021.

54 Nils Morawietz, Carolin Rehs, and Mathias Weller. A timecop's work is harder than you think. In *Proceedings of the 45th International Symposium on Mathematical Foundations of Computer Science (MFCS)*, volume 170, pages 71–1, 2020.

55 Nils Morawietz and Petra Wolf. A timecop's chase around the table. In *Proceedings of the 46th International Symposium on Mathematical Foundations of Computer Science (MFCS)*, 2021.

56 A.N. Patrinos and S. Louis Hakimi. The distance matrix of a graph and its tree realization. *Quarterly of Applied Mathematics*, 30:255–269, 1972.

57 Elena Rubei. Weighted graphs with distances in given ranges. *Journal of Classification*, 33:282—-297, 2016.

58 Piotr Sapiezynski, Arkadiusz Stopczynski, Radu Gatej, and Sune Lehmann. Tracking human mobility using wifi signals. *PloS one*, 10(7):e0130824, 2015.

59 Thomas J. Schaefer. The complexity of satisfiability problems. In *Proceedings of the 10th Annual ACM Symposium on Theory of Computing (STOC)*, pages 216–226, 1978.

60 H. Tamura, M. Sengoku, S. Shinoda, and T. Abe. Realization of a network from the upper and lower bounds of the distances (or capacities) between vertices. In *Proceedings of the 1993 IEEE International Symposium on Circuits and Systems (ISCAS)*, pages 2545—-2548, 1993.

61 Huanhuan Wu, James Cheng, Yiping Ke, Silu Huang, Yuzhen Huang, and Hejun Wu. Efficient algorithms for temporal path computation. *IEEE Transactions on Knowledge and Data Engineering*, 28(11):2927–2942, 2016.

62 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.