Temporal Graph Realization With Bounded Stretch

George B. Mertzios
Department of Computer Science, Durham University, UK

Hendrik Molter
Department of Computer Science, Ben-Gurion University of the Negev, Beer-Sheva, Israel

Nils Morawietz
LaBRI, Université de Bordeaux, France
Institute of Computer Science, Friedrich Schiller University Jena, Germany

Paul G. Spirakis
Department of Computer Science, University of Liverpool, UK

Abstract

A periodic temporal graph, in its simplest form, is a graph in which every edge appears exactly once in the first Δ time steps, and then it reappears recurrently every Δ time steps, where Δ is a given period length. This model offers a natural abstraction of transportation networks where each transportation link connects two destinations periodically. From a network design perspective, a crucial task is to assign the time-labels on the edges in a way that optimizes some criterion. In this paper we introduce a very natural optimality criterion that captures how the temporal distances of all vertex pairs are "stretched", compared to their physical distances, i.e. their distances in the underlying static (non-temporal) graph. Given a static graph G, the task is to assign to each edge one time-label between 1 and Δ such that, in the resulting periodic temporal graph with period Δ, the duration of the fastest temporal path from any vertex u to any other vertex v is at most α times the distance between u and v in G. Here, the value of α measures how much the shortest paths are allowed to be *stretched* once we assign the periodic time-labels.

Our results span three different directions: First, we provide a series of approximation and NP-hardness results. Second, we provide approximation and fixed-parameter algorithms. Among them, we provide a simple polynomial-time algorithm (the *radius-algorithm*) which always guarantees an approximation strictly smaller than Δ, and which also computes the optimum stretch in some cases. Third, we consider a parameterized local search extension of the problem where we are given the temporal labeling of the graph, but we are allowed to change the time-labels of at most k edges; for this problem we prove that it is W[2]-hard but admits an XP algorithm with respect to k.

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, stretch.

Digital Object Identifier 10.4230/LIPIcs.MFCS.2025.20

Related Version *Full Version*: `https://arxiv.org/abs/2504.14258`

Funding *George B. Mertzios*: Supported by the EPSRC grant EP/P020372/1.
Hendrik Molter: Supported by the Israel Science Foundation, grant nr. 1470/24, by the European Union's Horizon Europe research and innovation programme under grant agreement 949707, and by the European Research Council, grant nr. 101039913 (PARAPATH).
Nils Morawietz: Supported by the French ANR, project ANR-22-CE48-0001 (TEMPOGRAL)
Paul G. Spirakis: Supported by the EPSRC grant EP/P02002X/1.

1 Introduction

Graph realization is a classical problem where one is asked to decide whether a graph with a certain property, such as prescribed distances between vertex pairs or prescribed degrees for vertices, exists [7, 20, 28, 29]. These types of problems have a natural analog in the temporal graph setting[1]. Here we are given a static graph, and the task is to assign time-labels to each edge in order to create a temporal graph with some desired property. Temporal graph realization is an emerging topic in temporal graph algorithmics, and several connectivity-related properties have been considered. Previous work considered the property of temporal connectivity between every vertex pair or subsets thereof [3, 6, 32, 39], the size of so-called reachability sets [19], exact reachability graphs [21], fastest temporal paths[2] of a prescribed (exact) duration between every vertex pair [23, 33], and fastest temporal paths with a prescribed upper bound on the duration for every vertex pair [40, 41]. The latter two problems have been studied in the *periodic setting*, where every edge reappears after some given period length Δ.

We focus on the last among the mentioned problem settings, where we assume that the input consists of a graph and an upper bound for every (ordered) vertex pair that shall be respected by a fastest temporal path connecting the vertices. This problem is naturally motivated in the area of network design: we are given a transportation infrastructure and are tasked to develop a (periodic) schedule that guarantees fast delivery. In previous work [40], we showed that this problem is NP-hard even if the input graph is a star and the period length Δ is constant. This suggests that a lot of complexity is "hidden" in the prescribed upper bounds. However, it is arguably unnatural to have arbitrary upper bounds in the input that can be completely unrelated to the topology of the input graph. This leads us to the following problem.

We investigate the setting where we wish that the durations of all fastest temporal paths to be by at most some factor α times the distances of the respective vertex pairs in the (static) input graph. We then say that the realization has a bounded (multiplicative) *stretch*. This is a well-motivated and well-established concept in network design, especially in the context of spanning trees [1, 2, 8, 16, 17] or temporal spanners [5]. It allows the duration of connections between entities of the network to be related to the physical distance of those entities in a natural way. We formally define the problem as follows. The required terminology is formally defined in Section 2.

Stretched Temporal Graph Realization (STGR)	
Input:	A graph G (static, undirected), $\Delta \in \mathbb{N}$, and a rational number $\alpha \geq 1$.
Question:	Let D_G be the distance matrix of G. Can we assign one label in $\{1, \dots, \Delta\}$ to every edge such that in the resulting Δ-periodic temporal graph, for every vertex pair u, v the duration of a fastest path from u to v is at most $\alpha \cdot D_G(u, v)$?

Our Contribution. STGR is a natural and well-motivated special case of the problem that we investigated in previous work [40], where the upper bounds can be arbitrary. In this work, we show that STGR is NP-hard and hard to approximate, where we consider the goal of

[1] A *temporal graph* is a graph where one or more time-labels are assigned to every edge, indicating at which times this edge is active. We give a formal definition in Section 2.

[2] A *temporal path* uses edges with strictly increasing time-labels. A *fastest* temporal path is a temporal path that minimizes the difference between the arrival time and the starting time. We give a formal definition in Section 2.

the optimization version of the problem to minimize the stretch α. Formally, we prove the following in Section 3.

- For every $0 < \varepsilon < 1$ and every $c > 1$, STGR is NP-hard to approximate within a factor of $\Delta^{1-\varepsilon}$ or within a factor of 2^{n^c}.

Note that a stretch of Δ can trivially be achieved by assigning the same label to each edge. To complement these results, we show that we can achieve a strictly better approximation ratio than Δ, especially on graphs with small radius or diameter. Formally, we prove the following in Section 4.

- We can compute in polynomial time a solution for a STGR instance with stretch at most $\Delta - \frac{\Delta-1}{\min(\mathrm{rad}+1,\mathrm{diam})}$, where diam and rad denote the diameter and the radius of G, respectively.

Next, we focus on the decision version of the problem. We show that STGR is NP-hard even if Δ and α are small constants and in cases where the input graphs have a constant diameter. Formally, we prove the following in Section 5.

- STGR is NP-hard if $\Delta = 3$, $\alpha = 1$, and the input graph has diameter 2. This answers an open question by Erlebach et al. [23].
- STGR is NP-hard for each $\Delta \geq 3$, even if the diameter is in $\mathcal{O}(\Delta)$.

Recall that we can assume that α is at most Δ. We also show that STGR is NP-hard for each constant value of $\alpha \geq 1$. To complement these hardness results, we give the following algorithmic results in Section 6. We use standard terminology from parameterized complexity theory [11].

- STGR is fixed-parameter tractable when parameterized by the neighborhood diversity of the input graph and Δ.
- STGR is fixed-parameter tractable when parameterized by the treewidth of the input graph, the diameter of the input graph, and Δ.

These results are obtained by using extensions of monadic second order logic (MSO) [9,10,34] to formulate STGR, and then applying extensions of Courcelle's theorem [9,10,34].

Finally, we develop a local search algorithm for our problem. This algorithm, given a labeling, checks whether the stretch can be improved by changing at most a given number k of labels. We call k the search radius. We show the following in Section 7.

- Given a periodic labeling for a graph G and some constant k, we can compute the best possible stretch that can be obtained by changing at most k labels in polynomial time; that is, we present an algorithm that is in XP with respect to the parameter k.
- We show that the above-described local search problem is W[2]-hard when parameterized by the search radius k.

Proofs of results marked with ($\star$) are deferred to the full version of the paper [38].

Related Work. Since the 1960s, graph realization problems have been studied in many different settings. A complete literature review is beyond the scope of this work. While in the static setting, many different properties for realization have been studied, in particular, the realization of degree sequences, the previous work in the temporal setting considers mostly connectivity-related properties. We review the relevant work in the introduction.

The local search variant of our problem can also be seen as a temporal graph modification problem: we may modify up to k labels to manipulate the graph's connectivity properties. Many problems in this setting have been studies, where the modifications are typically edge removal, edge delay [14,18,37,42], or vertex removal [25,31,46]. Finally, *periodic* temporal graphs also have been investigated in other problem settings, most prominently in the context of cop and robber games [4,12,13,22,44,45].

2 Preliminaries

An undirected graph $G = (V, E)$ consists of a set V of vertices and a set $E \subseteq \binom{V}{2}$ of edges. We denote by $V(G)$ and $E(G)$ the vertex and edge set of G, respectively. Whenever no confusion arises, we denote $V(G)$ and $E(G)$ by just V and E, respectively. We use standard concepts and terminology from graph theory like *diameter*, *radius*, and *eccentricity* [15].

Let $G = (V, E)$ and $\Delta \in \mathbb{N}$, and let $\lambda : E \to \{1, \ldots, \Delta\}$ 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 $L(e) = \{\lambda(e) + i\Delta \mid i \geq 0\}$. In this case, we call λ a Δ-*periodic labeling* of G. When it is clear from the context, we drop Δ and denote the (Δ-periodic) temporal graph by (G, λ). We assume that Δ is encoded in binary in instances of STGR. Hence, the size of an instance is linear in n, m, $\log \Delta$, and the encoding length of α.

A *temporal* (s, z)*-walk* (or *temporal walk*) of length k from vertex $s = v_0$ to vertex $z = v_k$ in a Δ-periodic temporal graph (G, L) is a sequence $P = ((v_{i-1}, v_i, t_i))_{i=1}^{k}$ of triples that we call *transitions*, such that for all $i \in [k]$ we have that $t_i \in L(\{v_{i-1}, v_i\})$ and for all $i \in [k-1]$ we have that $t_i < t_{i+1}$. Moreover, we call P a *temporal* (s, z)*-path* (or *temporal path*) of length k if $v_i \neq v_j$ for all $i, j \in \{0, \ldots, k\}$ with $i \neq j$. Given a temporal path $P = ((v_{i-1}, v_i, t_i))_{i=1}^{k}$, we denote the set of vertices of P by $V(P) = \{v_0, v_1, \ldots, v_k\}$. A temporal (s, z)-path $P = ((v_{i-1}, v_i, t_i))_{i=1}^{k}$ is *fastest* if for all temporal (s, z)-path $P' = \left((v'_{i-1}, v'_i, t'_i)\right)_{i=1}^{k'}$ we have that $t_k - t_0 \leq t'_{k'} - t'_0$. We say that the *duration* of P is $d(P) = t_k - t_0 + 1$.

Let P be a temporal path on the edges $e_1, \ldots, e_k$ in a Δ-periodic labeling λ of G. For every $i = 1, \ldots, k-1$ let $\lambda(e_i) \in \{1, \ldots, \Delta\}$ be the label assigned to edge e_i, and let v_i be the common vertex of the edges e_i and e_{i+1}. The *waiting time* $\mathrm{wait}(v_i)$ of P on vertex v_i is $\lambda(e_{i+1}) - \lambda(e_i)$ (if $\lambda(e_{i+1}) > \lambda(e_i)$), or $\Delta + \lambda(e_{i+1}) - \lambda(e_i)$ (if $\lambda(e_{i+1}) < \lambda(e_i)$), or Δ (if $\lambda(e_{i+1}) = \lambda(e_i)$). Then, the duration of P is $\sum_{i=1}^{k} \mathrm{wait}(v_i) + 1$.

The next observation follows easily from the fact that, the worst-case for the stretch for (u, v) is realized when all edges of the graph have the same time-label.

▶ **Observation 2.1.** *Let λ be a Δ-labeling for a graph G. Then for any two vertices u and v, the stretch for (u, v) under λ is at most $\Delta - \frac{\Delta - 1}{\mathrm{dist}(u,v)}$.*

Finally, we give a brief argument that we can use polynomially many calls to a decision oracle for STGR so find the optimal stretch for a given input graph. Note that naïvely trying out all possible values for α does not yield polynomial time.

▶ **Lemma 2.2** (⋆). *Given a graph G and Δ, one can compute the smallest α, s.t. (G, Δ, α) is a yes-instance of STGR with $\mathcal{O}(\mathrm{diam}(G) \cdot \log(\Delta))$ calls to a decision oracle for STGR.*

3 Approximation Hardness

In this section, we show that STGR is NP-hard to approximate. In particular, we rule out constant factor approximations and even approximation algorithms with approximation factors that are sublinear in Δ or single exponential in n. Formally, we show the following.

▶ **Theorem 3.1.** *Assume that $P \neq NP$. Then, for all constants $0 < \varepsilon < 1$ and $c \geq 1$:*

- *there is no polynomial-time $\Delta^{1-\varepsilon}$-approximation algorithm for STGR;*
- *there is no polynomial-time 2^{n^c}-approximation algorithm for STGR.*

Proof. We present a straightforward gap-introducing reduction from Gossiping [26].

Gossiping	
Input:	A graph $G = (V, E)$.
Question:	Is there a labeling $\lambda\colon E \to \mathbb{N}$, such that for each pair (u, v) of vertices of V, there is a temporal path in the temporal graph (G, λ)?

Given an instance G of Gossiping, we produce a STGR instance as follows. We use the same graph G and specify how to set Δ later to get the two approximation hardness results.

Intuitively, if G is a yes-instance of Gossiping, then there exists a labeling where it is not necessary to use any label from the second Δ-period and hence the stretch is independent from Δ. Whereas if G is a no-instance of the gossiping problem, then there exists no such labeling. Then, informally speaking, there must be a path that crosses the period and has a duration that depends on Δ and hence also the stretch depends on Δ. Now we can set Δ to a very large value to create a gap.

Formally, assume that G is a yes-instance of Gossipingand let λ be a labeling such that the non-periodic temporal graph (G, λ) is temporally connected. We can assume without loss of generality that the largest label in λ is $\binom{n}{2}$. Now we use this labeling for our STGR instance. A very naïve estimation yields that the stretch is at most $\frac{1}{2} \cdot \binom{n}{2}$: Consider a vertex pair u, v of distance 2 in G. Then the temporal path from u to v in (G, λ) has duration at most $\binom{n}{2}$.

Now assume that G is a no-instance of Gossipingand consider the instance (G, Δ) (for some $\Delta > \binom{n}{2}$ that we specify later) of STGR. Let λ be a Δ-periodic labeling for G that minimizes the stretch. Note that we can obtain an equivalent labeling by adding a constant (modulo Δ) to every label. Hence, assume that $\delta = \Delta - \max_{e \in E(G)} \lambda(e)$ is maximized. Then we have that $\delta \geq \frac{\Delta}{\binom{n}{2}}$. Since G is a no-instance of the gossiping problem, there is a vertex pair u, v in G such that the temporal path from u to v in the Δ-periodic temporal graph (G, λ) uses labels from different periods and hence has duration at least δ and hence a stretch of at least $\frac{\delta}{n}$.

Let $\alpha^\star$ denote the optimal stretch of (G, Δ). Summarizing, we have the following.

- If G is a yes-instance of Gossiping, then we have that $\alpha^\star \leq \frac{1}{2} \cdot \binom{n}{2} = \alpha_{\text{yes}}$.
- If G is a no-instance of Gossiping, then we have that $\alpha^\star \geq \frac{\Delta}{n \cdot \binom{n}{2}} = \alpha_{\text{no}}$.

If follows that if we set $\Delta > \frac{n}{2} \cdot \binom{n}{2}^2$, then we create a gap that allows us to distinguish yes-instance of Gossipingfrom no-instances.

In the remainder, we show how to set Δ to obtain the approximation hardness results in the theorem statement. Note that in particular, we can distinguish yes-instance from no-instances whenever $\Delta \geq n^5$, since $n^5 > \frac{n}{2} \cdot \binom{n}{2}^2$. This will make the calculations easier.

For the first statement, let $0 < \varepsilon < 1$ and let $\varepsilon' = \frac{5(1-\varepsilon)}{\varepsilon}$. Note that $1 - \varepsilon = \frac{\varepsilon'}{5+\varepsilon'}$. Now we set $\Delta = n^{5+\varepsilon'}$. Then we have that $\Delta^{1-\varepsilon} = \Delta^{\frac{\varepsilon'}{5+\varepsilon'}} = n^{\varepsilon'}$. It follows that $\frac{\Delta}{\Delta^{1-\varepsilon}} = n^5$ and hence $\Delta^{1-\varepsilon} \cdot \alpha_{\text{yes}} < \alpha_{\text{no}}$. We can conclude that there is no $\Delta^{1-\varepsilon}$-approximation algorithm.

For the second statement, let $c \geq 1$ and set $\Delta = n^5 2^{n^c}$. Note that the encoding of Δ is polynomial in n, as $\log \Delta = n^c + 5 \log n$. We have that $\frac{\Delta}{2^{n^c}} = n^5$ and hence $2^{n^c} \cdot \alpha_{\text{yes}} < \alpha_{\text{no}}$. We can conclude that there is no 2^{n^c}-approximation algorithm. ◀

4 Approximation Algorithm

In this section, we give an approximation algorithm for STGR that runs in polynomial time and achieves an approximation ratio of $\Delta - \frac{\Delta - 1}{\min(\text{rad} + 1, \text{diam})}$, where diam and rad denote the diameter and the radius of the input graph G, respectively. Formally, we show the following.

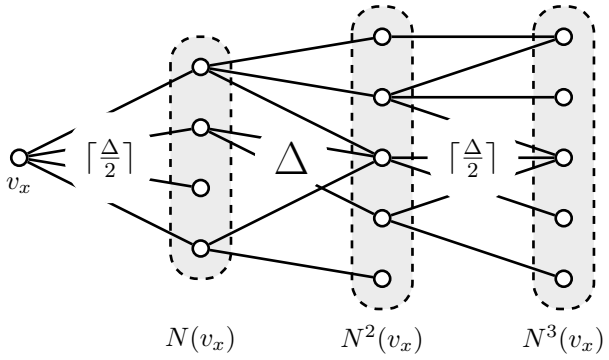

Figure 1 Example of a graph G with radius 3, where $v_x \in V(G)$ is a vertex of eccentricity equal to the radius. The gray areas depict the distance 1-3 neighborhoods of v_x. The labels given by the radius algorithm are illustrated. Edges between vertices in the same neighborhood are not depicted and are given arbitrary labels by the algorithm.

▶ **Theorem 4.1.** *A solution for STGR with stretch at most* $\Delta - \frac{\Delta-1}{\min(\text{rad}+1,\text{diam})}$ *can be computed in polynomial time.*

Note that this is strictly better than the "trivial" stretch obtained by Observation 2.1 if the radius and diameter differ. To show Theorem 4.1, we give the following algorithm, which we will refer to as *the radius algorithm*. Assume we are given an instance (G, Δ) of (the optimization version of) STGR. We perform the following steps.

- Take an arbitrary *root*, that is, a vertex $v_x \in V(G)$ of eccentricity equal to the radius.
- For each $i \in [1, \text{rad}]$, label all edges of $E(N^{i-1}(v_x), N^i(v_x))$ with label $\lceil \frac{\Delta}{2} \rceil$ if i is odd, and with label Δ, otherwise.
- All other edges are labeled arbitrarily.

For an illustration see Figure 1. We show that this algorithm achieves the following stretch.

▶ **Lemma 4.2.** *Let* $(G = (V, E), \Delta)$ *be an instance of STGR. Then, the radius algorithm computes a labeling with stretch at most* $\Delta - \frac{\Delta-1}{\min(\text{rad}+1,\text{diam})}$.

Proof. Let v_x be an arbitrary vertex of eccentricity equal to rad and let λ be the Δ-labeling produced by the radius algorithm for root v_x. We give for each distance $\ell \in [2, \text{diam}]$ an upper bound α_ℓ for the stretch of distance-ℓ vertex pairs.

For $\ell \leq \text{rad}+1$, let (u, v) be a pair of vertices of distance exactly ℓ in G. Due to Observation 2.1, $\alpha_\ell := \Delta - \frac{\Delta-1}{\ell}$ is an upper bound for the stretch of distance-ℓ vertex pairs.

For $\ell \geq \text{rad}+2$, let (u, v) be a pair of vertices of distance exactly ℓ in G. Consider the journey J that starts in u, goes over any shortest path P_u to v_x and then goes over any shortest path P_v to v. The edges of P_u (P_v) are alternatively labeled with Δ and $\lceil \frac{\Delta}{2} \rceil$ by the algorithms. Moreover, both path each have a length of at most rad. Hence, in the worst case, J traverses $2\,\text{rad}$ edges. This implies that the duration of J is at most $\text{rad} \cdot \Delta + 1$, since J only waits a full period at vertex v_x and otherwise alternates between waiting $\lfloor \frac{\Delta}{2} \rfloor$ and $\lceil \frac{\Delta}{2} \rceil$. Note that this also implies that there is a path of at most that duration from u to v. Consequently, $\alpha_\ell := \frac{\text{rad} \cdot \Delta + 1}{\ell} = \Delta + \frac{(\text{rad} - \ell) \cdot \Delta + 1}{\ell} = \Delta - \frac{\Delta-1}{\ell} - \frac{(\ell - \text{rad} - 1) \cdot \Delta}{\ell}$ is an upper bound for the stretch of distance-ℓ vertex pairs.

Note that the stretch achieved by λ is thus at most $\max\{\alpha_\ell \mid 2 \leq \ell \leq \text{diam}\}$, if $\text{diam} \geq 2$. (For $\text{diam} = 1$, the stretch is always 1.) We show that the maximum is achieved for $\ell = \text{rad}+1$ if $\text{rad} < \text{diam}$, and for $\ell = \text{rad}$ if $\text{rad} = \text{diam}$.

▷ Claim 4.3 (⋆). Let $\ell^* = \text{rad}+1$ if rad $<$ diam, and let $\ell^* = \text{rad}$ if rad $=$ diam. Then, $\alpha_{\ell^*} \geq \max\{\alpha_\ell \mid 2 \leq \ell \leq \text{diam}\}$.

This thus proves that the stretch achieved by λ is at most $\Delta - \frac{\Delta-1}{\min(\text{rad}+1,\text{diam})}$. ◀

Theorem 4.1 follows directly from Lemma 4.2 and the straightforward observation that the radius algorithm runs in polynomial time. However, we can show that in several restricted cases, the algorithm is guaranteed to achieve a better stretch.

▶ **Lemma 4.4** (⋆). *Let $(G = (V, E), \Delta)$ be an instance of STGR. If $2 \leq \text{rad} < \text{diam}$ and there is a root v_x such that for each distance-$(\text{rad}+1)$ vertex pair (u, v), $\text{dist}(v_x, u) + \text{dist}(v_x, v) < 2\,\text{rad}$, then the radius algorithm (for root v_x) computes a labeling with stretch at most $\Delta - \frac{\Delta-1}{\text{rad}}$.*

As we will show in Section 5, there are instances, where this stretch is achieved by the radius algorithm and it is NP-hard to decide whether a better stretch is possible. This then shows that this simple algorithm produces the best possible stretch for some instances in polynomial time, unless P = NP.

Finally, we can show that the radius algorithm performs well if the input graph is a tree.

▶ **Lemma 4.5.** *Let $(G = (V, E), \Delta)$ be an instance of STGR where G is a tree. Then the radius algorithm computes a labeling with stretch at most $\frac{\Delta+1}{2}$.*

Proof. Similar to the previous proofs, we give for each distance $\ell \in [2, \text{diam}]$ an upper bound α_ℓ for the stretch of distance ℓ vertex pairs.

Let u, v be a vertex pair of distance-ℓ. We make the following case distinction. Consider the case where u is an ancestor of v if the graph G is rooted at v_x or v is an ancestor of u. In both cases, the edges of the fastest paths from u to v are alternatively labeled with Δ and $\lceil \frac{\Delta}{2} \rceil$ by the algorithm. Hence, we have that the duration of the fastest path from u to v is at most $\frac{\ell-2}{2} \cdot \Delta + \lfloor \frac{\Delta}{2} \rfloor + 1$ if ℓ is even, and at most $\frac{\ell-1}{2} \cdot \Delta + 1$ if ℓ is odd. Hence, the stretch is $\alpha \leq \frac{\Delta}{2} + \frac{1}{\ell}$. Since $\ell \geq 2$, we have that $\alpha \leq \frac{\Delta+1}{2}$.

Now consider the case that neither u is an ancestor of v, nor is v an ancestor of u. Let w be the closest common ancestor of v and u. Then we have that a fastest temporal path from u to v can be decomposed into a fastest temporal path from u to w and a fastest temporal path from w to v. Note that the waiting time at w is Δ, since all edges from w to its children have the same label. All other waiting times are $\lfloor \frac{\Delta}{2} \rfloor$ and $\lceil \frac{\Delta}{2} \rceil$, alternatingly. It follows that the duration of a fastest temporal path from u to v is at most $\frac{\ell-1}{2} \cdot \Delta + \lfloor \frac{\Delta}{2} \rfloor + 1$ if ℓ is even, and at most $\frac{\ell}{2} \cdot \Delta + 1$ if ℓ is odd. Then, again, we have that the stretch is $\alpha = \frac{\Delta}{2} + \frac{1}{\ell}$. Since $\ell \geq 2$, we have that $\alpha \leq \frac{\Delta+1}{2}$. ◀

On trees with a large maximum degree, we can show that our algorithm is optimal.

▶ **Lemma 4.6** (⋆). *The radius algorithm computes the optimum stretch for trees with maximum degree at least $\Delta + 1$ and it is a 2-approximation algorithm on general trees.*

5 General Hardness Results

In this section, we present NP-hardness results for STGR for (i) all constant values of $\Delta \geq 3$ and (ii) all constant values of $\alpha \geq 1$. All of our results are achieves by reductions from 3-Coloring, which is NP-hard [30].

3-Coloring	
Input:	A graph $G = (V, E)$.
Question:	Is there a *proper* 3-*coloring* χ of G, that is, a function $\chi\colon V \to \{1, 2, 3\}$, such that for each edge $\{u, v\} \in E$, $\chi(u) \neq \chi(v)$?

First, we will present two hardness results for $\Delta = 3$: one for $\alpha \in [1, 1.5)$ and one for $\alpha \in [1.5, 2)$. The first reduction answers an open question by Erlebach et al. [23] and the second reduction proves that our presented radius-algorithm is tight on some hard instances. That is, we show that on the build instances, the radius-algorithm is guaranteed to achieve a stretch of 2 and it is NP-hard to decide whether a better stretch is possible on these instances. Afterwards, we present hardness results for all other constant values of $\Delta \geq 4$ and $\alpha \geq 2$. To this end, we will adapt the latter reduction for $\Delta = 3$ by replacing parts of the instance by some gadgets that we define for each $\Delta > 3$.

Hardness for $\Delta = 3$. We now start by presenting our first hardness result.

▶ **Lemma 5.1.** *For each $\alpha \in [1, 1.5)$, STGR is NP-hard even if $\Delta = 3$ and G has diameter 2.*

Proof. We reduce from 3-Coloring. Let $G = (V, E)$ be an instance of 3-Coloring and assume that each vertex of V has at least one non-neighbor. Moreover, let $\alpha \in [1, 1.5)$. We obtain an instance $I' := (G' = (V', E'), \Delta = 3, \alpha)$ of STGR as follows: We initialize the graph $G' := (V', E')$ as the star with center vertex c and leaf set V. Additionally, we add for each non-edge $\{u, v\}$ of G the vertices $x_{u,v}$ and $x_{v,u}$, and the edges $\{u, x_{u,v}\}, \{u, x_{v,u}\}, \{v, x_{u,v}\}, \{v, x_{v,u}\}$, and $\{x_{u,v}, x_{v,u}\}$ to G'. That is, $G'[\{u, v, x_{u,v}, x_{v,u}\}]$ is a diamond. Finally, we add another vertex c^* to G' and making $\{c^*\} \cup (V' \setminus V)$ into a clique in G'. This completes the construction of I'. Note that G' has a diameter of 2, since both vertices c and c^* are adjacent to all vertices. In the following, let D be the distance matrix of G' and let $C := \{c^*\} \cup (V' \setminus V)$. The intuition of this reduction is that for each edge $\{u, v\}$ of G, (u, c, v) and (u, c^*, v) are the only paths of length less than $\Delta > 2 \cdot \alpha = \alpha \cdot D_{u,v}$. Hence, on both paths, the labels of both edges need to be distinct in any labeling λ of stretch at most α. Next, we show that G is 3-colorable if and only if I is a yes-instance of STGR.

($\Rightarrow$) Let $\chi\colon V \to \{1, 2, 3\}$ be a 3-coloring of G. We define a edge labeling λ of G' of stretch 1 as follows: For each vertex $v \in V$, we set $\lambda(\{c, v\}) := \chi(v)$ and $\lambda(\{c^*, v\}) := 4 - \chi(v)$. For each non-edge $\{u, v\}$ of G, we set $\lambda(\{u, x_{u,v}\}) := \lambda(\{v, x_{v,u}\}) := 1$ and $\lambda(\{u, x_{v,u}\}) := \lambda(\{v, x_{u,v}\}) := 3$. For all other edges e of G', we set $\lambda(e) := 2$. Note that these latter edges are the edges of the clique $V' \setminus V$. We now show that λ has a stretch of 1, that is, for each pair of vertices of distance 2 in G', there are temporal paths of duration 2 between them. Let (a, b) be a pair of vertices of distance two in G'. Since $V' \setminus V$ is a clique in G', at least one of a and b is a vertex of V. Without loss of generality assume that $a \in V$. We distinguish two cases.

First, assume that $b \notin V$. By construction of G' this implies that $b = x_{u,v}$ for some non-edge u, v of G with $a \notin \{u, v\}$. Since we assumes that each vertex of V has at least one non-neighbor in G, there is a vertex $w \in V$ such that the vertices $x_{a,w}$ and $x_{w,a}$ exist. Hence, by definition of λ, the path $(a, x_{a,w}, x_{u,v} = b)$ uses the labels 1 and 2, and the path $(b = x_{u,v}, x_{w,a}, a)$ uses the labels 2 and 3. Consequently, there are temporal paths between a and b of durations 2.

Second, assume that $b \in V$. If $\{a, b\}$ is a non-edge of G, then the vertices $x_{a,b}$ and $x_{b,a}$ exist. Hence, by definition of λ, the paths $(a, x_{b,a}, b)$ and $(b, x_{a,b}, a)$ uses the labels 3 and 1.

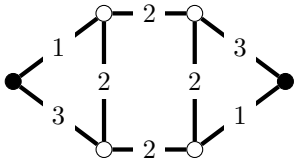

Figure 2 The sunglasses gadgets for $\Delta = 3$ with the sunglasses labeling. The black vertices are the docking points and the white vertices are the central vertices.

Consequently, there are temporal paths between a and b of durations 2. Otherwise, that is, if $\{a, b\}$ is an edge of G, then $\chi(a) \neq \chi(b)$ since χ is a proper coloring of G. Since $\Delta = 3$, this implies that $\chi(b) = \chi(a) + 1$ (modulo Δ) or that $\chi(a) = \chi(b) + 1$ (modulo Δ). Assume without loss of generality that the first is the case. Hence, by definition of λ, the path (a, c, b) uses consecutive time-labels (modulo Δ) and the path (b, c^*, a) uses consecutive lime labels (modulo Δ). Consequently, there are temporal paths between a and b of durations 2. Concluding, λ has a stretch of $1 \leq \alpha$, which implies that I' is a yes-instance of STGR.

($\Leftarrow$) Let $\lambda\colon E' \to \{1, 2, 3\}$ be an edge labeling of G' with stretch at most α. We define a 3-coloring χ of the vertices of V as follows: For each vertex $v \in V$, we set $\chi(v) := \lambda(\{c, v\})$. Next, we show that for each edge $\{u, v\} \in E$, u and v receive distinct colors under χ. Recall that for λ to have a stretch of α for G', at least one path of length at most $\alpha \cdot D_{u,v} = \alpha \cdot D_{v,u} = \alpha \cdot 2 < 3$ from u to v in G' has duration at most $\alpha \cdot 2 < 3$. Since $\{u, v\}$ is an edge of E, the vertices $x_{u,v}$ and $x_{v,u}$ do not exist, which implies that (u, c, v) and (u, c^*, v) are the only paths from u to v in G' of length less than 3 and that (v, c, u) and (v, c^*, u) are the only paths from v to u in G' of length less than 3. Assume towards a contradiction that $\chi(u) = \chi(v)$. This implies that $\lambda(\{c, u\}) = \lambda(\{c, v\})$. Hence, the paths (u, c, v) and (v, c, u) have a duration of exactly $\Delta + 1 = 4 > 2 \cdot \alpha = \alpha \cdot D_{u,v} = \alpha \cdot D_{v,u}$. Consequently, for λ to have a stretch of α for G', both paths (u, c^*, v) and (v, c^*, u) must have a duration of 2. Since $\Delta = 3$, this is impossible. This contradicts the assumption that λ is an edge labeling of G' with stretch at most α. Thus, $\chi(u) = \lambda(\{c, u\}) \neq \lambda(\{c, v\}) = \chi(v)$, which implies that χ is a proper 3-coloring for G' ◀

Note that Lemma 5.1 answers (by setting $\alpha = 1$) an open question by Erlebach et al. [23] about the complexity of finding a periodic Δ-labeling for a diameter-2 graph G, such that the fastest paths between any two vertices of G equals their distance.

Next, we show that for each $\Delta \geq 3$, it is NP-hard to decide whether a stretch in $[\frac{\Delta}{2}, \frac{\Delta+1}{2})$ can be achieved. To this end, we define gadget graphs for each Δ and some associated labelings. First, we define this gadgets for $\Delta = 3$ and present the first reduction. Afterwards, we define these gadgets for odd values of $\Delta > 3$ and even values of $\Delta \geq 4$ and present similar reductions.

▶ **Definition 5.2** (Sunglasses gadgets for $\Delta = 3$)**.** *A* 3-sunglasses gadget with docking points u and v *is the graph shown in Figure 2, where the black vertices indicate the docking points* u *and* v*, and the white vertices are* central vertices.

Figure 2 also shows what we call the *sunglasses labeling*.

▶ **Definition 5.3.** *Let* $\Delta > 1$*, let* $G = (V, E)$ *be a graph, and let* $\lambda\colon E \to [1, \Delta]$*. Moreover, let* u *and* v *be vertices of distance exactly* 2 *in* G*. We call a vertex* z *of* G *a* nice neighbor of u and v*, if (i)* z *is a common neighbor of* u *and* v *and (ii)* $\lambda(\{u, z\}) \neq \lambda(\{v, z\})$*.*

▶ **Observation 5.4.** *Let $\Delta > 1$, let $G = (V, E)$ be a graph, and let $\lambda\colon E \to [1, \Delta]$. Moreover, let u and v be vertices of distance exactly 2 in G. If u and v have a nice common neighbor z, then (u, z, v) and (v, z, u) are paths of duration at most Δ each.*

We now show that there are graphs with diameter 3 and radius 2, for which it is NP-hard to decide whether one can achieve a better stretch than the one achieved by the radius-algorithm. This then shows that this simple and natural algorithm is tight on some hard instances.

▶ **Lemma 5.5.** *For $\Delta = 3$ and each $\alpha \in [1.5, 2)$, STGR is NP-hard even on graphs with diameter 3, radius 2, and where the radius algorithm produces a labeling of stretch 2.*

Proof. Let $\Delta = 3$ and let $\alpha \in [1.5, 2)$. We again reduce from 3-Coloring. Let $G = (V, E)$ be an instance of 3-Coloring. Again, assume that each vertex of V has at least one non-neighbor, as otherwise we would know that this vertex receives a unique color in any 3-coloring and the remaining vertices can only be colored in two colors, which can be checked in polynomial time. We obtain an equivalent instance $I := (G', \Delta, \alpha)$ of STGR as follows: We initialize the graph $G' := (V', E')$ as the star with center vertex c and leaf set V. Additionally, we add for each non-edge $\{u, v\}$ of G a 3-sunglasses gadget $S_{\{u,v\}}$ with docking points u and v to G'. Let X denote the set of the central vertices of all added sunglasses gadgets.

To complete the definition of G', we add the vertices $\widehat{X} := \{\widehat{x} \mid x \in X\}$ to G' and making each vertex of $\widehat{X}$ adjacent to each vertex of $X \cup \widehat{X} \cup \{c\}$ in G'.

This completes the construction of I. In the following, let D be the distance matrix of G'. The intuitive idea is that for each edge $\{u, v\} \in E$, the path (u, c, v) $((v, c, u))$ is the only path of length less than $\Delta + 1 = 4 > \alpha \cdot D_{u,v} = 2 \cdot \alpha$ from u (v) to v (u) in G'. Hence, each labeling λ of G' of stretch at most α has to assign distinct labels to the edges $\{c, u\}$ and $\{c, v\}$. In other words, if labeling λ of G' has stretch at most α, then the edges between c and the vertices of V imply a 3-coloring for G.

Structural Properties. Note that radius of G' is at most 2: The neighborhood of vertex c is $V \cap \widehat{X}$ and each vertex of $X = V' \setminus (V \cup \widehat{X} \cup \{c\})$ is a neighbor of each vertex of $\widehat{X}$. Moreover, G' has a diameter of 3, since (i) each vertex of V has distance at most 2 to each vertex of $V \cup \widehat{X}$ by going over c and thus distance at most 3 to each vertex of X and (ii) each vertex of X has distance at most 2 to each vertex of $X \cup \{c\}$ by going over some vertex of $\widehat{X}$ and thus distance at most 3 to each vertex of V. In particular, all vertex pairs of distance exactly 3 in G' contain one vertex of V and one vertex of X. Since c is a neighbor of all vertices of V, G' fulfills the property of Lemma 4.4, which implies that the radius algorithm produces a labeling of stretch of at most $\Delta - \frac{\Delta - 1}{\mathrm{rad}} = 2$.

Correctness. We now show that G is 3-colorable if and only if I is a yes-instance of STGR.

($\Leftarrow$) Let $\lambda\colon E' \to \{1, 2, 3\}$ be an edge labeling of G' with stretch at most α. We define a 3-coloring $\chi\colon V \to \{1, 2, 3\}$ as follows: For each vertex $v \in V$, we set $\chi(v) := \lambda(\{c, v\})$. Next, we show that for each edge $\{u, v\} \in E$, u and v receive distinct colors under χ. Since $\{u, v\}$ is an edge of E, there is no sunglasses gadget with docking points u and v in G'. This implies that (u, c, v) is the only path from u to v in G' of length less than $\Delta + 1 = 4 > \alpha \cdot D_{u,v} = 2 \cdot \alpha$. Assume towards a contradiction that $\chi(u) = \chi(v)$. This would imply that $\lambda(\{c, u\}) = \lambda(\{c, v\})$. Hence, the unique path (u, c, v) from u to v in G' of length less than $\Delta + 1 = 4$ has a duration of exactly $4 = \Delta + 1 > 2 \cdot \alpha = \alpha \cdot D_{u,v}$. This contradicts the assumption that λ is an edge labeling of G' with stretch at most α. Thus, $\chi(u) = \lambda(\{c, u\}) \neq \lambda(\{c, v\}) = \chi(v)$, which implies that χ is a proper 3-coloring for G'.

($\Rightarrow$) Let $\chi\colon V \to \{1,2,3\}$ be a 3-coloring of G. Assume without loss of generality that each of the three colors is assigned at least once. We define an edge labeling λ of G' of stretch $\frac{\Delta}{2} = 1.5$ as follows: For each vertex $v \in V$, we set $\lambda(\{c,v\}) := \chi(v)$. For each non-edge $\{a,b\}$ of G (that is, for each added sunglasses gadget), we label the edges of the sunglasses gadget $S_{x,y}$ according the sunglasses labeling (see Figure 2). To complete the definition of λ, it remains to define the labels of the edges incident with the vertices of $\widehat{X}$. For each vertex $\widehat{x} \in \widehat{X}$, we set $\lambda(\{x,\widehat{x}\}) := 1$. For each other edge e incident with at least one vertex of $\widehat{X}$, we set $\lambda(e) := 2$.

This completes the definition of λ. Next, we show that λ has a stretch of $1.5 = \frac{\Delta}{2} \leq \alpha$.

First, we consider vertex pairs $\{y,z\}$ of distance 2 in G'. For these vertex pairs, we show that there are paths of duration at most $3 = 1.5 \cdot D_{y,z}$ between y and z.

- If $y = c$, then by the initial argumentation, $z \in X$. Hence, $\widehat{z}$ is a nice neighbor of c and z, which implies that the path $(c,\widehat{z},z)$ has duration at most $\Delta = 3$ is both directions.
- If $y \in \widehat{X}$, then by the initial argumentation, $z \in V$. Since z is the docking point of at least one sunglasses gadget, there are at least two vertices x_1 and x_2 of X adjacent to z. For at least one $i \in \{1,2\}$, $z \neq \widehat{x_i}$. Hence, the edge $\{x_i,z\}$ receives label 2 under λ. This implies that x_i is a nice neighbor, since the edge $\{y,x_i\}$ receives label either 1 or 3 under λ. Consequently, the path $(c,\widehat{z},z)$ has duration at most $\Delta = 3$ is both directions.
- If $y \in X$, then $z \in X \cup V \cup \{c\}$. The case for $z = c$ is covered by the above cases. If $z \in X$, the path $(y,\widehat{y},z)$ has labels $(1,2)$ and thus has duration at most $\Delta = 3$ in both directions. Otherwise, if $z \in V$, then by construction, z is a docking point of the sunglasses gadget that contains y. Hence, by the sunglasses labeling, there is a nice neighbor of y and z in this sunglasses gadget which implies the existence of paths of duration at most 3 between in y and z.
- If $y \in V$, then $z \in X \cup \widehat{X} \cup V$. The case for $z \in X \cup \widehat{X}$ is covered by the above cases. If $z \in V$, we consider two cases. If $\{y,z\}$ is an edge of E, c is a nice neighbor of y and z, since χ is a proper 3-coloring of G. Otherwise, if $\{y,z\}$ is not an edge of G, there is a sunglasses gadget with docking points y and z. By the sunglasses labeling, this gadget contains a path with labels $(1,2,3)$ from y to z and a path with labels $(1,2,3)$ from z to y. In both cases, there are paths of duration at most 3 between y and z.

Next, we consider vertex pairs $\{y,z\}$ of distance 3 in G'. For these vertex pairs, we show that there are paths of duration at most $4 < 4.5 = 1.5 \cdot D_{y,z}$ between y and z. By the initial argumentation about the structural properties of G', we can assume without loss of generality that $y \in V$ and $z \in X$. Moreover, z is not part any sunglasses gadget attached to y, as otherwise, the distance between y and z would be at most 2. Take an arbitrary sunglasses gadget attached to y and let x_1 and x_3 be the neighbors of y in this sunglasses gadget. By the sunglasses labeling, we can assume that the label of $\{y,x_i\}$ is equal to i for each $i \in \{1,3\}$. Since $z \in X$, $\widehat{z}$ is a vertex of $\widehat{X}$, $\lambda(\{z,\widehat{z}\}) = 1$, and $\lambda(\{\widehat{z},x_1\}) = \lambda(\{\widehat{z},x_3\}) = 2$. This implies that the path $(z,\widehat{z},x_3,y)$ has labels $(1,2,3)$ and thus a duration of 3. For the other direction, the path $(y,x_1,\widehat{z},z)$ has labels $(1,2,1)$ and thus a duration of 4. Hence, there are paths of duration at most 4 between y and z.

This implies that the stretch of λ is at most $\frac{\Delta}{2} = 1.5 \leq \alpha$, which completes the proof. ◀

In combination with Lemma 5.1, this implies that for $\Delta = 3$, STGR is NP-hard for each $\alpha \in [1,2)$.

▶ **Corollary 5.6.** *For $\Delta = 3$, STGR is NP-hard for each $\alpha \in [1,2)$.*

Hardness for $\Delta > 3$. We now proceed by showing that also for each $\Delta > 3$, STGR is NP-hard for each $\alpha \in [\frac{\Delta}{2}, \frac{\Delta+1}{2})$. To obtain this result, we define for each $\Delta > 4$, Δ-sunglasses gadgets and replace the 3-sunglasses gadgets in the reduction of the proof of Lemma 5.5 by the larger Δ-sunglasses gadgets. The definition of these larger gadgets and the reduction is deferred to the attached full version.

▶ **Theorem 5.7** (⋆). *For each $\Delta \geq 4$ and each $\alpha \in [\frac{\Delta}{2}, \frac{\Delta+1}{2})$, STGR is NP-hard on graphs of diameter $\mathcal{O}(\Delta)$.*

Recall that Lemma 5.1 shows that STGR is NP-hard for each $\alpha \in [1, 1.5)$. Moreover, since for each constant $\alpha \geq 1.5$, there is a constant $\Delta \geq 3$, such that $\alpha \in [\frac{\Delta}{2}, \frac{\Delta+1}{2})$, Lemmas 5.1 and 5.5 and Theorem 5.7 imply the following.

▶ **Theorem 5.8.** *For each constant $\Delta \geq 3$, STGR is NP-hard. For each constant $\alpha \geq 1$, STGR is NP-hard.*

6 Fixed-parameter Algorithms through Monadic Second-Order Logic

In this section, we show that STGR is fixed-parameter tractable with respect to combinations of Δ, the treewidth $\mathrm{tw}(G)$, the diameter $\mathrm{diam}(G)$, and the neighborhood diversity $\mathrm{nd}(G)$ of the input graph G. Formally, we show the following two results.

▶ **Theorem 6.1.** *STGR is in FPT when parameterized by $\mathrm{nd}(G) + \Delta$.*

▶ **Theorem 6.2** (⋆). *STGR is in FPT when parameterized by $\mathrm{tw}(G) + \mathrm{diam}(G) + \Delta$.*

Note that Theorem 6.2 implies that for all graph classes that have locally bounded treewidth (such as planar graphs), we get that STGR is in FPT when parameterized by the diameter of the input graph and Δ. Furthermore, Theorem 6.2 implies that STGR is in FPT when parameterized by the treedepth of the input graph and Δ, since both the treewidth and the diameter of a graph can be upper bounded by a function of the treedepth. We remark that the neighborhood diversity of a graph is unrelated to the combination of the treewidth and the diameter of a graph. Finally, by Lemma 2.2, both theorem statements also hold for the optimization version of STGR.

To show Theorems 6.1 and 6.2 we show that STGR is expressible in monadic second-order (MSO) logic in certain specific ways that allows us to employ Courcelle's famous theorem (and extensions thereof) [9, 10]. Since we define the MSO formulas directly on the input graph G, we do not give the formal definitions of treewidth and neighborhood diversity, since they are not necessary to obtain the results. For more information on treewidth, we refer to standard textbooks on graph theory, e.g. the one by Diestel [15], and for more information on the neighborhood diversity, we refer to Lampis [35], who introduced this parameter.

Assume we are given a graph $G = (V, E)$, an integer Δ, and a real number $\alpha \geq 1$. Let $n = |V|$. A *monadic second-order (MSO) formula* ϕ over G is a formula that uses

- the incidence relation of vertices and edges,
- the logical operators $\wedge$, $\vee$, $\neg$, $=$, and parentheses,
- a finite set of variables, each of which is either taken as an element or a subset of V or E,
- and the quantifiers $\forall$ and $\exists$.

Additionally we will use some folklore shortcuts such as $\Rightarrow$, $\neq$, $\subseteq$, $\in$, and $\setminus$, which can themselves be replaced by MSO formulas. For an edge set E' we use $V(E')$ to denote the set of vertices that are incident with the edges in E'. Furthermore, we use the formula

partition$_i(X_1, X_2, \ldots, X_i, X)$ to express that sets $X_1, \ldots, X_i$ form a partition of X. It is well-known that this formula has size $\mathcal{O}(i^2)$.

We use two different extensions of MSO. The first one is called CMSO and allows for testing the cardinality of a set. We remark that this does not allow for comparing the cardinalities of sets.

- $\text{card}_{n,p}(X) = \text{true}$ if and only if $|X| \equiv n \mod p$.

This extension of MSO was already considered by Courcelle [9]. We have the following, where $|\phi|$ denotes the length of the formula ϕ.

▶ **Theorem 6.3** ([9,10]). *CMSO model checking is in FPT when parameterized by* $\text{tw}(G)+|\phi|$.

The second extension is stronger and allows for linear cardinality constraints, that is, expressions of the type $x_1 \leq x_2$, where x_1 and x_2 are linear expressions over cardinalities of sets. This extension is called $\text{MSO}^{\text{GL}}_{\text{lin}}$ [34] and the following is known.

▶ **Theorem 6.4** ([34]). *MSO^{GL}_{lin} model checking is in FPT when parameterized by* $\text{nd}(G)+|\phi|$.

We now introduce some basic MSO formulas that we will use to compose the formulas to express STGR. It is well-known that connectivity and various related concepts can be expressed in MSO.

- $\text{conngraph}(X, E')$ tests whether the subgraph $(X, E' \cap X^2)$ of G is connected.
- $\text{conn}(v, w, E')$ tests whether the two vertices $v, w \in V$ are connected by a path that only uses edges from $E' \subseteq E$.
- $\text{path}(v, w, E')$ tests whether the two vertices $v, w \in V$ are connected by a path that *exactly* uses edges from $E' \subseteq E$.

It is well-known that all these formulas have constant size. We provide the formulas in the full version of the paper [38].

We first show Theorem 6.1. To this end, we introduce additional predicates that use linear cardinality constraints and hence are $\text{MSO}^{\text{GL}}_{\text{lin}}$ formulas. Afterwards, we show how to replace these predicates with equivalent (larger) CMSO formulas.

- $\text{spath}(v, w, E')$ tests whether the two vertices $v, w \in V$ are connected by a path that *exactly* uses edges from $E' \subseteq E$ and whether this is a shortest path:

$$\text{spath}(v, w, E') := \text{path}(v, w, E') \wedge \forall E'' : |E''| < |E'| \Rightarrow \neg\, \text{conn}(v, w, E'')$$

Now we are ready to give an $\text{MSO}^{\text{GL}}_{\text{lin}}$ formula $\phi_{G,\Delta,\alpha}$ that expresses STGR. We are looking for a partition of E into $E_1, E_2, \ldots, E_\Delta$. We interpret $e \in E_i$ with edge e receiving label i.

$$\phi_{G,\Delta,\alpha} = \exists E_1, \ldots, E_\Delta : \text{partition}_\Delta(E_1, \ldots, E_\Delta, E) \wedge \tag{1}$$

$$\forall v, w\ \exists E^\star : \Big(\text{path}(v, w, E^\star) \wedge \tag{2}$$

$$\exists X_1, \ldots, X_\Delta : \Big(\text{partition}_\Delta(X_1, \ldots, X_\Delta, V(E^\star) \setminus \{v, w\}) \wedge \tag{3}$$

$$\bigwedge_{1 \leq i \leq \Delta} \Big(\forall x \in X_i\ \exists e_1, e_2 \in E^\star : \big(e_1 \neq e_2 \wedge x \in e_1 \cap e_2 \wedge \text{conn}(v, x, E^\star \setminus \{e_1\}) \wedge \tag{4}$$

$$\bigvee_{1 \leq i' \leq \Delta} (e_1 \in E_{i'} \wedge e_2 \in E_{(i'+i) \bmod \Delta}) \big) \Big) \wedge \tag{5}$$

$$\exists E^{\star\star} : \text{spath}(v, w, E^{\star\star}) \wedge \Big(\sum_{1 \leq i \leq \Delta} i \cdot |X_i| \Big) + 1 \leq \alpha \cdot |E^{\star\star}| \Big) \Big) \tag{6}$$

Observe that the size of the formula is in $\mathcal{O}(\Delta^2)$ and that it can be computed in $\mathcal{O}(\Delta^2)$ time. Now we prove that ϕ expresses STGR.

▶ **Lemma 6.5** ($\star$). *Given an instance $I = (G, \Delta, \alpha)$ of STGR, we have that $\phi_{G,\Delta,\alpha}$ is satisfiable if and only if I is a yes-instance.*

Now we have all the ingredients to prove Theorem 6.1.

Proof of Theorem 6.1. Theorem 6.1 follows directly from Lemma 6.5, Theorem 6.4, and from the fact that $\phi_{G,\Delta,\alpha}$ is an $\mathrm{MSO}_{\mathrm{lin}}^{\mathrm{GL}}$-formula with a size in $\mathcal{O}(\Delta^2)$ that can be computed in $\mathcal{O}(\Delta^2)$ time. ◀

7 A Parameterized Local Search Approach

Based on our classical hardness and inapproximability results, it is natural to ask for good polynomial-time heuristic approaches for our problem. From this standpoint, we now consider a 'parameterized local search' version of our problem, that is, we try to improve the stretch of a given labeling by changing the labels of just a few edges. In general, in *parameterized local search* the goal is to improve a given solution by performing a modification that is upper-bounded by k (according to some specified measurement between solutions). Here, k is an additional parameter often referred to as the *search radius* [43]. Marx [36] first considered parameterized local search for the Traveling Salesperson problem in 2008 and since then, these kind of local search problems were considered for many optimization problems (see [24, 27, 36, 43] for some collections of problems).

The following problem we consider in this section generalizes the classical parameterized local search version of STGR, as it does not only ask for any improvement but rather for an improvement to some desired stretch α_0. It is formally defined as follows.

Local Search STGR (LS STGR)

Input: A graph $G = (V, E)$, $\Delta \in \mathbb{N}$, a labeling $\lambda \colon E \to [1, \Delta]$, $k \in \mathbb{N}$, and a number $\alpha_0 \geq 1$.
Question: Does there exist a labeling λ' that disagrees with λ on at most k edges, such that the stretch of λ' is at most α_0?

Note that this problem can also be seen as a generalization of STGR, as STGR is obtained by setting k to the number of edges of the input graph. Generally, our goal is to analyze the parameterized complexity of LS STGR with respect to the parameter k. We present an XP-algorithm for k, showing that we can efficiently decide whether a stretch of α_0 can be achieved from our current labeling by changing only a constant number of edge-labels. Note that by Lemma 2.2, we can also find the optimal stretch in this search space with an additional polynomial factor in the running time. A natural question is then to ask for an FPT algorithm for k. As we show, such an algorithm presumably does not exist, as the problem is W[2]-hard for k, even when asking for any improvement.

▶ **Theorem 7.1.** *LS STGR admits an XP algorithm when parameterized with k.*

Proof. Our algorithm iterates over all $\mathcal{O}(|E|^k)$ subsets $F \subseteq E$ of size at most k. For each such subset F, we then ask the question, whether we can achieve a stretch of α_0 by changing only the labels of edges of F. That is, we iteratively solve the following intermediate problem.

Fixed Edges Relabel STGR	
Input:	A graph $G = (V, E)$, $\Delta \in \mathbb{N}$, a set $F \subseteq E$, a labeling $\lambda' \colon E \setminus F \to [1, \Delta]$, and a number $\alpha_0 \geq 1$.
Question:	Does there exist a labeling $\lambda_F \colon F \to [1, \Delta]$, such that the stretch of $\lambda' \cup \lambda_F$ is at most α_0?

Based on the $|E|^k \cdot n^{\mathcal{O}(1)}$ time (which is XP time) for the iteration over all candidate sets F, to present our XP algorithm, it suffices to show that Fixed Edges Relabel STGR admits an XP algorithm for k.

Let $e_1, e_2, \ldots, e_k$ be the edges of the given set F, enumerated arbitrarily, and assume that $k \neq |E|$, that is, there exists at least one edge of E that is not in F. For every $1 \leq i \leq k$ denote by u_i and v_i the endpoints of the edge e_i; note that some edges of F may have common endpoints. Denote the set of all endpoints of the edges of F by $V_F = \{u_i, v_i : 1 \leq i \leq k\}$. We now define the set $N_F = \bigcup_{v \in V_F} \{e \in E \setminus F : v \in e\}$ of all edges in G that are not in F but share at least one common endpoint with some edge in F. Furthermore, we define the set $L_0 = \{\lambda(e) : e \in N_F\}$ of all distinct time-labels that appear in at least one edge of N_F. Let $\ell_1 < \ldots < \ell_t$ be the labels of L_0 in increasing order. For simplicity of the presentation, we assume without loss of generality that $\ell_1 = 1$; this can be achieved by subtracting $\ell_1 - 1$ from the time-label of every edge. Note that $t = |L_0| \leq |N_F| \leq \sum_{v \in V_F} |N(v)| - |F| \leq k \cdot \deg_{\max} \in \mathcal{O}(n^2)$. Finally, we define the $2t$ subsets $Z_1, \ldots, Z_{2t}$ of time-labels, called *zones*, as follows. For every $j = 1, 2, \ldots, t-1$, we define $Z_{2j-1} = \{\ell_j\}$ and $Z_{2j} = \{\ell_j + 1, \ldots, \ell_{j+1} - 1\}$. For $j = t$, we define $Z_{2t-1} = \{\ell_t\}$ and $Z_{2t} = \{\ell_t + 1, \ldots, \Delta\}$.

For every edge $e_i \in F$, we guess in which zone Z_j the label $\lambda(e_i)$ lies. These are in total $(2t)^k$ different cases, as we have k edges in F and $2t$ potential zones for the label of each edge. Furthermore, for every allocation of the edges of F to the $2t$ different zones of time-labels, we also guess a permutation of the time-labels of the edges of F which are allocated to the same zone. These are in total $k!$ different permutations. Each of these permutations corresponds to a different relative order of the time-labels of the edges of F. For each of these permutations, we also guess whether two consecutive time-labels within the same zone are equal or not; these are 2^k different choices. Summarizing, for every allocation of the edges of F to the $2t$ different zones of time-labels, we guess the relative order of the time-labels of the edges of F in each zone, by distinguishing which time-labels are equal and which are different. We call each of these relative orders a *time-label profile*; there are at most $2^k k!$ different profiles.

Let P be an arbitrary temporal path, and let e_1, e_2 be any two consecutive edges in P, with v being their common endpoint. Note that, once we have fixed an allocation of the edges of F to the $2t$ different zones and a time-label profile, we know the relative order of the time-labels ℓ_1 and ℓ_2 of the edges e_1 and e_2, respectively. More specifically, if $\ell_2 > \ell_1$, then the waiting time at v is $\ell_2 - \ell_1$; if $\ell_2 < \ell_1$, then the waiting time at v is $\Delta + \ell_2 - \ell_1$; if $\ell_2 = \ell_1$, then the waiting time at v is Δ.

Consider two arbitrary vertices z and w in G and an arbitrary edge $e_i = u_i v_i$ of F. Let P be a fastest temporal path P from z to w, and assume that P passes through e_i. Without loss of generality, let P first visit u_i and then v_i. Note that, given a fixed allocation of the edges of F to the $2t$ different zones and a fixed time-label profile, if e_i is neither the first nor the last edge of P, then the duration of P is *independent* of the exact time-label of the edge e_i. The reason is that, in this case, the relative order of the time-label of e_i, compared to the time-labels of the previous and the next edge on P, is fixed in the given time-label profile, regardless of the exact value of the time-label of e_i.

Now suppose that e_i is the *last* (resp. *first*) edge of P, i.e. $w = v_i$ (resp. $w = u_i$). Then, the duration of P is smallest when the time-label of e_i is the smallest (resp. largest) possible, while still respecting the relative order of the time-labels in the given profile.

Note that, if we fix a *specific* time-label for each edge of F, we can trivially compute the stretch α of this specific time-labeling. This can be done by just computing the fastest temporal path from every vertex z to every other vertex w, dividing its duration by the length of the shortest path between z and w in the underlying graph G, and returning the largest of these ratios as the stretch α of this time-labeling.

Our algorithm proceeds as follows, while examining a fixed time-label profile. For each edge $e_i \in F$, we perform binary search for the time-label of e_i in the zone Z_j in which e_i is allocated (indepentently of any other edge $e_{i'}$), until we find a time-labeling (if it exists) that gives a stretch α that is at most the stretch threshold α_0. During this procedure of performing multiple binary searches on each edge of F independently, we only consider those time-labelings that conform with the current time-label profile.

We iterate over all possible $2^k k!$ different profiles and, for each of them, we perform the above binary searches. If, during this procedure, we detect a time-labeling of the edges of F that gives a stretch $\alpha \leq \alpha_0$, then we return this time-labeling; otherwise, we return that such a labeling does not exist. The running time of this algorithm is

$$(2k \cdot \deg_{\max} \cdot \log \Delta)^k 2^k k! \cdot (n + \log \Delta)^{\mathcal{O}(1)} = (2n^2 \log \Delta)^k 2^k k! \cdot (n + \log \Delta)^{\mathcal{O}(1)},$$

as there are at most $2t = 2k \cdot \deg_{\max} \leq 2n^2$ different allocations of each of the k edges of F to a time-label zone, at most $2^k k!$ different profiles, all independent binary searches on the k edges can be performed in $(\log \Delta)^k$ time, which is XP time, since the input size includes $\log \Delta$. Thus this is an XP algorithm for Fixed Edges Relabel STGR with respect to the parameter k. By implementing the above algorithm iteratively for each of the $\mathcal{O}(|E|^k)$ subsets $F \subset E$ of k edges, we obtain an XP algorithm for LS STGR with respect to k.

Finally note that, if $k = |E|$, that is, every edge of the graph is in the set F, then $N_F = \emptyset$ and $t = 0$. In this case, we create just one time-label zone $Z_1 = \{1, \ldots, \Delta\}$ that contains all possible Δ time-labels, and then we perform exactly the same algorithm as above. ◀

We now show that there is presumably no FPT algorithm for LS STGR for parameter k; recall that k is part of the input.

▶ **Theorem 7.2** ($\star$). *LS STGR is W[2]-hard when parameterized by k.*

8 Conclusion

In this paper, we investigated a natural temporal graph realization problem, where the durations of the fastest connections in the produced (periodic) temporal graph shall be at most a multiplicative factor times the corresponding distances in the static graph. Among other results, showed that the problem is hard to solve exactly and also hard to approximate within a constant factor on general instances. We also designed a polynomial time algorithm for general graphs and that achieves a factor-2 approximation on trees, whereas there are NP-hard instances for which the guaranteed stretch is tight. Our work leaves several natural future work directions: Are there instances where the optimal stretch is strictly larger than $\frac{\Delta+1}{2}$? What is the complexity of STGR for $\Delta = 2$? Can we identify larger graph classes than trees, where we can achieve a constant-factor approximation? What is the parameterized complexity of STGR with respect to structural parameters of the input graph (independent of Δ, e.g. treewidth by itself)? What types of results can we achieve when we allow an additive stretch, or if we want to minimize the *average* stretch?

References

1 Ittai Abraham, Yair Bartal, and Ofer Neiman. Nearly tight low stretch spanning trees. In *Proceedings of the 49th Annual IEEE Symposium on Foundations of Computer Science (FOCS)*, pages 781–790. IEEE Computer Society, 2008.

2 Ittai Abraham and Ofer Neiman. Using petal-decompositions to build a low stretch spanning tree. *SIAM Journal on Computing*, 48(2):227–248, 2019.

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

4 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.

5 Davide Bilò, Gianlorenzo D'Angelo, Luciano Gualà, Stefano Leucci, and Mirko Rossi. Sparse temporal spanners with low stretch. In *Proceedings of the 30th Annual European Symposium on Algorithms (ESA)*, pages 19:1–19:16, 2022.

6 Arnaud Casteigts, Michelle Döring, and Nils Morawietz. Realization of Temporally Connected Graphs Based on Degree Sequences. *CoRR*, abs/2504.17743, 2025.

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

8 Edith Cohen. Fast algorithms for constructing t-spanners and paths with stretch t. *SIAM Journal on Computing*, 28(1):210–236, 1998.

9 Bruno Courcelle. The monadic second-order logic of graphs. I. Recognizable sets of finite graphs. *Information and computation*, 85(1):12–75, 1990.

10 Bruno Courcelle and Joost Engelfriet. *Graph structure and monadic second-order logic: a language-theoretic approach*, volume 138. Cambridge University Press, 2012.

11 Marek Cygan, Fedor V. Fomin, Lukasz Kowalik, Daniel Lokshtanov, Dániel Marx, Marcin Pilipczuk, Michal Pilipczuk, and Saket Saurabh. *Parameterized Algorithms*. Springer, 2015.

12 Jean-Lou De Carufel, Paola Flocchini, Nicola Santoro, and Frédéric Simard. Copnumbers of periodic graphs. *CoRR*, abs/2310.13616, 2023.

13 Jean-Lou De Carufel, Paola Flocchini, Nicola Santoro, and Frédéric Simard. Cops & robber on periodic temporal graphs: Characterization and improved bounds. In Sergio Rajsbaum, Alkida Balliu, Joshua J. Daymude, and Dennis Olivetti, editors, *Structural Information and Communication Complexity - 30th International Colloquium (SIROCCO)*, pages 386–405, 2023.

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

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

16 Michael Elkin, Yuval Emek, Daniel A. Spielman, and Shang-Hua Teng. Lower-stretch spanning trees. *SIAM Journal on Computing*, 38(2):608–628, 2008.

17 Yuval Emek and David Peleg. Approximating minimum max-stretch spanning trees on unweighted graphs. *SIAM Journal on Computing*, 38(5):1761–1781, 2008.

18 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.

19 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.

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

21 Thomas Erlebach, Othon Michail, and Nils Morawietz. Recognizing and Realizing Temporal Reachability Graphs. In *Proceedings of the 33rd Annual European Symposium on Algorithms (ESA)*, 2025. To appear. Full version: http://arxiv.org/abs/2503.15771.

22 Thomas Erlebach, Nils Morawietz, Jakob T. Spooner, and Petra Wolf. A cop and robber game on edge-periodic temporal graphs. *Journal of Computer and System Sciences*, 144:103534, 2024.

23 Thomas Erlebach, Nils Morawietz, and Petra Wolf. Parameterized algorithms for multi-label periodic temporal graph realization. In *Proceedings of the 3rd Symposium on Algorithmic Foundations of Dynamic Networks (SAND)*, pages 12:1–12:16, 2024.

24 Michael R. Fellows, Fedor V. Fomin, Daniel Lokshtanov, Frances A. Rosamond, Saket Saurabh, and Yngve Villanger. Local search: Is brute-force avoidable? *Journal of Computer and System Sciences*, 78(3):707–719, 2012.

25 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.

26 Frits Göbel, J Orestes Cerdeira, and Hendrik Jan Veldman. Label-connected graphs and the gossip problem. *Discrete Mathematics*, 87(1):29–40, 1991.

27 Jiong Guo, Danny Hermelin, and Christian Komusiewicz. Local search for string problems: Brute-force is essentially optimal. *Theoretical Computer Science*, 525:30–41, 2014.

28 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.

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

30 Richard M. Karp. Reducibility among combinatorial problems. In *Complexity of computer computations (Proc. Sympos., IBM Thomas J. Watson Res. Center, Yorktown Heights, N.Y., 1972)*, pages 85–103. Plenum, New York, 1972.

31 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.

32 Nina Klobas, George B. Mertzios, Hendrik Molter, and Paul G. Spirakis. The complexity of computing optimum labelings for temporal connectivity. *Journal of Computer and System Sciences*, 146:103564, 2024.

33 Nina Klobas, George B. Mertzios, Hendrik Molter, and Paul G. Spirakis. Temporal graph realization from fastest paths. In *Proceedings of the 3rd Symposium on Algorithmic Foundations of Dynamic Networks (SAND)*, pages 16:1–16:18, 2024.

34 Dusan Knop, Martin Koutecký, Tomás Masarík, and Tomás Toufar. Simplified algorithmic metatheorems beyond MSO: treewidth and neighborhood diversity. *Log. Methods Comput. Sci.*, 15(4), 2019.

35 Michael Lampis. Algorithmic meta-theorems for restrictions of treewidth. *Algorithmica*, 64:19–37, 2012.

36 Dániel Marx. Searching the k-change neighborhood for TSP is W[1]-hard. *Operations Research Letters*, 36(1):31–36, 2008.

37 Kitty Meeks. Reducing reachability in temporal graphs: Towards a more realistic model of real-world spreading processes. In *Proceedings of the 18th Conference on Computability in Europe (CiE)*, pages 186–195, 2022.

38 G.B. Mertzios, H. Molter, N. Morawietz, and P.G. Spirakis. Temporal graph realization with bounded stretch. *CoRR*, abs/2504.14258, 2025.

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

40 George B. Mertzios, Hendrik Molter, Nils Morawietz, and Paul G. Spirakis. Realizing temporal transportation trees. In *Proceedings of the 51st Workshop on Graph-Theoretic Concepts in Computer Science (WG)*, 2025. To appear. Full version: https://arxiv.org/abs/2403.18513.

41 Julia Meusel, Matthias Müller-Hannemann, and Klaus Reinhardt. Directed temporal tree realization for periodic public transport: Easy and hard cases. *CoRR*, abs/2504.07920, 2025.

42 Hendrik Molter, Malte Renken, and Philipp Zschoche. Temporal reachability minimization: Delaying vs. deleting. *Journal of Computer and System Sciences*, 144:103549, 2024.

43 Nils Morawietz. *On the complexity of local search problems with scalable neighborhoods*. PhD thesis, Friedrich-Schiller-Universität Jena, 2024. Dissertation.

44 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)*, pages 71:1–71:14, 2020.

45 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)*, pages 77:1–77:18, 2021.

46 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.