Computing Maximum Matchings in Temporal Graphs

George B. Mertzios
Department of Computer Science, Durham University, UK
george.mertzios@durham.ac.uk

Hendrik Molter
Institut für Softwaretechnik und Theoretische Informatik, TU Berlin, Germany
h.molter@tu-berlin.de

Rolf Niedermeier
Institut für Softwaretechnik und Theoretische Informatik, TU Berlin, Germany
rolf.niedermeier@tu-berlin.de

Viktor Zamaraev
Department of Computer Science, University of Liverpool, UK
viktor.zamaraev@liverpool.ac.uk

Philipp Zschoche
Institut für Softwaretechnik und Theoretische Informatik, TU Berlin, Germany
zschoche@tu-berlin.de

Abstract

Temporal graphs are graphs whose topology (i.e. whose edge set) is subject to discrete changes over time. Given a static underlying graph G, a temporal graph is represented by assigning a set of integer time-labels to every edge e of G, indicating the discrete time steps at which e is active in the temporal graph. We introduce and study the complexity of a natural temporal extension of the classical graph problem Maximum Matching, which takes into account the dynamic nature of temporal graphs. In our problem, Maximum Temporal Matching, we are looking for the largest possible number of time-labeled edges (simply *time-edges*) (e, t) such that no vertex is matched more than once within any time window of Δ consecutive time slots, where $\Delta \in \mathbb{N}$ is given. The requirement that a vertex cannot be matched twice in any Δ-window models some necessary "cooling off" (or "recovery") period that needs to pass for an entity (vertex) after being paired up for some activity with another entity. For example, in a mobile sensor networks' context, two devices might need to recharge their batteries for Δ time units after participating in a common activity with each other. Here it is reasonable to focus on inputs with a constant Δ, independent of the input size, as this "recovery" period usually depends on the nature of the interactions and the participating entities (vertices), rather than on the total number of entities. We prove strong computational hardness results for Maximum Temporal Matching, even for basic cases; therefore, we mainly turn our attention to polynomial-time approximation and to fixed-parameter algorithms. We provide a simple $\frac{2}{3}$-approximation algorithm for the base case $\Delta = 2$, which we then generalize to an approximation algorithm with ratio $\frac{\Delta}{2\Delta-1}$ for an arbitrary Δ. Thus, for every constant Δ we break the barrier of $\frac{1}{2}$ in the approximation ratio. With respect to parameterized complexity, we first prove that the problem is fixed-parameter tractable with respect to the parameter "size of the desired solution". Furthermore, motivated by complementing hardness results, we show fixed-parameter tractability with respect to the combined parameter "Δ and size of a maximum matching of the underlying graph"; the latter may be significantly smaller than the cardinality of a maximum temporal matching.

2012 ACM Subject Classification Theory of computation → Graph algorithms analysis, Fixed parameter tractability, Approximation algorithms analysis

Keywords and phrases Temporal Graph, Link Stream, Temporal Line Graph, NP-hardness, APX-hardness, Approximation Algorithm, Fixed-parameter Tractability, Independent Set.

Digital Object Identifier 10.4230/LIPIcs.CVIT.2016.23

Funding George B. Mertzios and Viktor Zamaraev were supported by the EPSRC grant EP/P020372/1. Hendrik Molter was supported by the DFG project MATE (NI 369/17).
Viktor Zamaraev: The main part of this paper was prepared while Viktor Zamaraev was affiliated at the Department of Computer Science, Durham University, UK.

1 Introduction

Computing a maximum matching in an undirected graph (a maximum-cardinality set of "independent edges", i.e., edges which do not share any endpoint) is one of the most fundamental graph-algorithmic primitives. In this work, we lift the study of the algorithmic complexity of computing maximum matchings from static graphs to the—recently strongly growing—field of *temporal graphs* [1–3, 7, 10, 25, 50, 51]. In a nutshell, a temporal graph is a graph whose topology is subject to discrete changes over time. We adopt a simple and natural model for temporal graphs which originates in the foundational work of Kempe et al. [44]. According to this model, every edge of a static graph is given along with a set of time labels, while the vertex set remains unchanged.

▶ **Definition 1** (Temporal Graph). *A* temporal graph $\mathcal{G} = (G, \lambda)$ *is a pair* (G, λ)*, where* $G = (V, E)$ *is an underlying (static) graph and* $\lambda : E \to 2^{\mathbb{N}} \setminus \{\emptyset\}$ *is a* time-labeling *function that specifies which edge is* active *at what time.*

An alternative way to view a temporal graph is as an ordered set (according to the discrete time slots) of graph instances (called *snapshots*) on a fixed vertex set. Due to its vast applicability in many areas, the notion of temporal graphs has been studied from different perspectives under various names such as *time-varying* [58], *evolving* [20, 26], *dynamic* [16, 34], and *graphs over time* [48]; see also the survey papers [14–16] and the references therein.

In this paper we introduce and study the complexity of a natural temporal extension of the classical problem Maximum Matching, which takes into account the dynamic nature of temporal graphs. To this end, we extend the notion of "edge independence" by adding the temporal dimension to it: two time-labeled edges (simply *time-edges*) (e, t) and (e', t') are Δ*-independent* whenever (i) the edges e, e' do not share an endpoint or (ii) their time labels t, t' are at least Δ time units apart from each other.[1] Then, for any given Δ, the problem Maximum Temporal Matching asks for the largest possible set of mutually Δ-independent edges in a temporal graph. That is, in a feasible solution, no vertex can be matched more than once within any time window of length Δ. In particular, it is important to understand the complexity of the problem in the case where Δ is a constant, since this models short "recovery" periods.

Our main motivation for studying Maximum Temporal Matching is of theoretical nature, namely to lift one of the most classical optimization problems, Maximum Matching, to the temporal setting. As it turns out, Maximum Temporal Matching is computationally hard to approximate: we prove that the problem is APX-hard, even when $\Delta = 2$ and the lifetime T of the temporal graph (i.e., the maximum edge label) is 3 (see Section 3.1). That is, unless P=NP, there is no Polynomial-Time Approximation Scheme (PTAS) for any $\Delta \geq 2$ and $T \geq 3$. In addition, we show that the problem remains NP-hard even if the underlying graph G is just a path (see Section 3.2). Consequently, we mainly turn our attention to approximation and to fixed-parameter algorithms (see Section 4). In order to prove our

[1] Throughout the paper, Δ always refers to that number, and never to the maximum degree of a static graph (which is another common use of Δ).

hardness results, we introduce the notion of a *temporal line graph* which is a class of (static) graphs of independent interest and may prove useful in other contexts, too. This notion enables us to reduce Maximum Temporal Matching to the problem of computing a large independent set in a static graph (i.e., in the temporal line graph that is defined from the input temporal graph). Moreover, as an intermediate result, we show (see Theorem 11) that the classic problem Independent Set (on static graphs) remains NP-hard on induced subgraphs of *diagonal grid* graphs, thus strengthening an old result of Clark et al. [19] for unit disk graphs.

During the last few decades it has been repeatedly observed that for many variations of Maximum Matching it is straightforward to obtain online (resp. greedy offline approximation) algorithms which achieve a competitive (resp. an approximation) ratio of $\frac{1}{2}$, while great research efforts have been made to increase the ratio to $\frac{1}{2}+\varepsilon$, for *any* constant $\varepsilon > 0$. Originating in the foundational work of Karp et al. [43] on the randomized online algorithm Ranking for the online bipartite matching problem, there has been a long line of recent research on providing a sequence of $(\frac{1}{2}+\varepsilon)$-competitive algorithms for many different variations of online matching, see e.g. [13, 30, 39, 40]. This difficulty of breaking the barrier of the $\frac{1}{2}$ ratio also appears in offline variations of the matching problem. It is well known that an arbitrary greedy algorithm for matching gives approximation ratio at least $\frac{1}{2}$ [38, 46], while it remains a long-standing open problem to determine how well a randomized greedy algorithm can perform. Aronson et al. [5] provided the so-called Modified Randomized Greedy (MRG) algorithm which approximates the maximum matching within a factor of at least $\frac{1}{2}+\frac{1}{400{,}000}$. Recently, Poloczek and Szegedy [56] proved that MRG actually provides an approximation ratio of $\frac{1}{2}+\frac{1}{256}$. Similarly to the above problems, it is straightforward[2] to approximate Maximum Temporal Matching in polynomial time within a factor of $\frac{1}{2}$. However, we manage to provide a simple approximation algorithm which, for any constant Δ, achieves an approximation ratio $\frac{1}{2}+\varepsilon$ for a constant ε. For $\Delta = 2$ this ratio is $\frac{2}{3}$, while for an arbitrary constant Δ it becomes $\frac{\Delta}{2\Delta-1} = \frac{1}{2}+\frac{1}{2(2\Delta-1)}$ (see Section 4.1).

Apart from approximation algorithms, the classical (static) matching problem (which is polynomially solvable) has recently also attracted many research efforts in the area of parameterized algorithms for polynomial problems. Parameters which have been studied include the solution size [35], the modular-width [47], the clique-width [21], the treewidth [28], the feedback vertex number [52], and the feedback edge number [45, 52]. Given that Maximum Temporal Matching is NP-hard, we show fixed-parameter tractability with respect to the desired solution size parameter. Finally, we show fixed-parameter tractability with respect to the combined parameter of Δ and size of a maximum matching of the underlying graph (which may be significantly smaller than the cardinality of a maximum temporal matching of the temporal graph). Our algorithmic techniques are essentially based on kernelization and matroid theory (see Section 4).

It is worth mentioning that another temporal variation of Maximum Matching, which is related to ours, was recently proposed by Baste et al. [9]. The main difference is that their model requires edges to exist in at least Δ *consecutive* snapshots in order for them to be eligible for a matching. Thus, their matchings need to consist of time-consecutive edge blocks, which requires some data cleaning on real-word instances in order to perform meaningful experiments [9].

[2] To achieve the straightforward $\frac{1}{2}$-approximation it suffices to just greedily compute at every time slot a maximal matching among the edges that are Δ-independent with the edges that were matched in the previous time slots.

It turns out that the model of Baste et al. is a special case of our model, as there is an easy reduction from their model to ours, and thus their results are also implied by ours. Baste et al. [9] showed that solving (using their definition) Maximum Temporal Matching is NP-hard for $\Delta \geq 2$. In terms of parameterized complexity, they provided a polynomial-sized kernel for the combined parameter (k, Δ), where k is the size of the desired solution.

We see the concept of multistage (perfect) matchings, which was introduced by Gupta et al. [37], as the main alternative model for temporal matchings in temporal graphs. This model, which is inspired by reconfiguration or reoptimization problems, is not directly related to ours: roughly speaking, their goal is to find perfect matchings for every snapshot of a temporal graph such that the matchings only slowly change over time. In this setting one mostly encounters computational intractability, which leads to several results on approximation hardness and algorithms [8, 37].

2 Preliminaries

We use standard mathematical and graph-theoretic notation. For an overview of the most important classical notation and terminology we use see Appendix A.1.

Temporal graphs. Throughout the paper we consider temporal graphs $\mathcal{G}$ with *finite lifetime* $T(\mathcal{G}) = \max\{t \in \lambda(e) \mid e \in E\}$, that is, there is a maximum label assigned by λ to an edge of G. When it is clear from the context, we denote the lifetime of $\mathcal{G}$ simply by T. The *snapshot* (or *instance*) of $\mathcal{G}$ *at time* t is the static graph $G_t = (V, E_t)$, where $E_t = \{e \in E \mid t \in \lambda(e)\}$. We refer to each integer $t \in [T]$ as a *time slot* of $\mathcal{G}$. For every $e \in E$ and every time slot $t \in \lambda(e)$, we denote the *appearance of edge* e *at time* t by the pair (e, t), which we also call a *time-edge*. We denote the set of edge appearances of a temporal graph $\mathcal{G} = (G = (V, E), \lambda)$ by $\mathcal{E}(\mathcal{G}) := \{(e, t) \mid e \in E \text{ and } t \in \lambda(e)\}$. For every $v \in V$ and every time slot t, we denote the *appearance of vertex* v *at time* t by the pair (v, t). That is, every vertex v has T different appearances (one for each time slot) during the lifetime of $\mathcal{G}$. For every time slot $t \in [T]$, we denote by $V_t = \{(v, t) : v \in V\}$ the set of all vertex appearances of $\mathcal{G}$ at time slot t. Note that the set of all vertex appearances in $\mathcal{G}$ is $V \times [T] = \bigcup_{1 \leq t \leq T} V_t$. Two vertex appearances (v, t) and (w, t) are *adjacent* if the temporal graph has the time-edge $(\{v, w\}, t)$. For a temporal graph $\mathcal{G} = (G, \lambda)$ and a set of time-edges M, we denote by $\mathcal{G} \setminus M := (G', \lambda')$ the temporal graph $\mathcal{G}$ without the time-edges in M, where $G' := (V, E')$ with $E' := \{e \in E \mid \lambda(e) \setminus \{t \mid (e, t) \in M\} \neq \emptyset\}$ and for all $e \in E'$, $\lambda'(e) := \lambda(e) \setminus \{t \mid (e, t) \in M\}$. For a subset $S \subseteq [T]$ of time slots and a time-edge set M, we denote by $M|_S := \{(e, t) \in M \mid t \in S\}$ the set of time-edges in M with a label in S. For a temporal graph $\mathcal{G}$, we denote by $\mathcal{G}|_S := \mathcal{G} \setminus (\mathcal{E}(\mathcal{G})|_{[T] \setminus S})$ the temporal graph where only time-edges with label in S are present.

In the remainder of the paper we denote by n and m the number of vertices and edges of the underlying graph G, respectively, unless otherwise stated. We assume that there is no compact representation of the labeling λ, that is, $\mathcal{G}$ is given with an explicit list of labels for every edge, and hence the *size* of a temporal graph $\mathcal{G}$ is $|\mathcal{G}| := |V| + \sum_{t=1}^{T} |E_t| \in O(n + mT)$. Furthermore, in accordance with the literature [61, 62] we assume that the lists of labels are given in ascending order.

Temporal matchings. A *matching* in a (static) graph $G = (V, E)$ is a set $M \subseteq E$ of edges such that for all $e, e' \in M$ we have that $e \cap e' = \emptyset$. In the following, we transfer this concept to temporal graphs.

For a natural number Δ, two time-edges (e, t), (e', t') are Δ-*independent* if $e \cap e' = \emptyset$ or $|t - t'| \geq \Delta$. If two time-edges are not Δ-independent, then we say that they are *in conflict*.

A time-edge (e,t) Δ-*blocks* a vertex appearance (v,t') (or (v,t') is Δ-*blocked* by (e,t)) if $v \in e$ and $|t-t'| \leq \Delta - 1$. A Δ-*temporal matching* M of a temporal graph $\mathcal{G}$ is a set of time-edges of $\mathcal{G}$ which are pairwise Δ-independent. Formally, it is defined as follows.

▶ **Definition 2** (Δ-Temporal Matching). *A* Δ-temporal matching *of a temporal graph* $\mathcal{G}$ *is a set* M *of time-edges of* $\mathcal{G}$ *such that for every pair of distinct time-edges* $(e,t),(e',t')$ *in* M *we have that* $e \cap e' = \emptyset$ *or* $|t-t'| \geq \Delta$.

We remark that this definition is similar to the definition of γ-*matchings* by Baste et al. [9].

A Δ-temporal matching is called *maximal* if it is not properly contained in any other Δ-temporal matching. A Δ-temporal matching is called *maximum* if there is no Δ-temporal matching of larger cardinality. We denote by $\mu_\Delta(\mathcal{G})$ the size of a maximum Δ-temporal matching in $\mathcal{G}$.

Having defined temporal matchings, we naturally arrive at the following central problem.

Maximum Temporal Matching	
Input:	A temporal graph $\mathcal{G} = (G, \lambda)$ and an integer $\Delta \in \mathbb{N}$.
Output:	A Δ-temporal matching in $\mathcal{G}$ of maximum cardinality.

We refer to the problem of deciding whether a given temporal graph admits an Δ-temporal matching of a given size k by Temporal Matching.

We discuss some basic observations about our problem settings in Appendix A.2 and discuss the relation between our model and the model of Baste et al. [9] in Appendix A.3.

Temporal line graphs. In the following, we transfer the concept of line graphs to temporal graphs and temporal matchings. In particular, we make use of temporal line graphs in the NP-hardness result of Section 3.2.

The Δ-*temporal line graph* of a temporal graph $\mathcal{G}$ is a static graph that has a vertex for every time-edge of $\mathcal{G}$ and two vertices are connected by an edge if the corresponding time-edges are in conflict, i.e. they cannot be both part of a Δ-temporal matching of $\mathcal{G}$. We say that a graph H is a *temporal line graph* if there exists Δ and a temporal graph $\mathcal{G}$ such that H is isomorphic to the Δ-temporal line graph of $\mathcal{G}$. Formally, temporal line graphs and Δ-temporal line graphs are defined as follows.

▶ **Definition 3** (Temporal Line Graph). *Given a temporal graph* $\mathcal{G} = (G = (V,E), \lambda)$ *and a natural number* Δ*, the* Δ-temporal line graph $L_\Delta(\mathcal{G})$ *of* $\mathcal{G}$ *has vertex set* $V(L_\Delta(\mathcal{G})) = \{e_t \mid e \in E \wedge t \in \lambda(e)\}$ *and edge set* $E(L_\Delta(\mathcal{G})) = \{\{e_t, e'_{t'}\} \mid e \cap e' \neq \emptyset \wedge |t-t'| < \Delta\}$. *We say that a graph* H *is a* temporal line graph *if there is a temporal graph* $\mathcal{G}$ *and an integer* Δ *such that* $H = L_\Delta(\mathcal{G})$.

By definition, Δ-temporal line graphs have the following property.

▶ **Observation 4.** *Let* $\mathcal{G}$ *be a temporal graph and let* $L_\Delta(\mathcal{G})$ *be its* Δ*-temporal line graph. The cardinality of a maximum independent set in* $L_\Delta(\mathcal{G})$ *equals the size of a maximum* Δ*-temporal matching of* $\mathcal{G}$.

It follows that solving Temporal Matching on a temporal graph $\mathcal{G}$ is equivalent to solving Independent Set on $L_\Delta(\mathcal{G})$.

3 Hardness Results

3.1 APX-completeness of Maximum Temporal Matching

In this subsection, we look at Maximum Temporal Matching where we want to maximize the cardinality of the temporal matching. We prove that Maximum Temporal Matching is APX-complete even if $\Delta = 2$ and $T = 3$. For this we provide a so-called *L-reduction* [6] from the APX-complete Maximum Independent Set problem on cubic graphs [4] to Maximum Temporal Matching. Together with the constant-factor approximation algorithm that we present in Section 4.1 this implies APX-completeness for Maximum Temporal Matching. The reduction also implies NP-completeness of Temporal Matching. Formally, we show the following result.

▶ **Theorem 5.** Temporal Matching *is NP-complete and* Maximum Temporal Matching *is APX-complete even if $\Delta = 2$, $T = 3$, and every edge of the underlying graph appears only once. Furthermore, for any $\delta \geq \frac{664}{665}$, there is no polynomial-time δ-approximation algorithm for* Maximum Temporal Matching*, unless P = NP, and* Temporal Matching *does not admit a $2^{o(k)} \cdot |\mathcal{G}|^{f(T)}$-time algorithm for any function f, unless the Exponential Time Hypothesis fails.*

We start by describing the construction behind the reduction. It is easy to check that the construction uses only three time steps and every edge appears in exactly one time step.

▶ **Construction 1.** Let $G = (V, E)$ be an n-vertex cubic graph. We construct in polynomial time a corresponding temporal graph (H, λ) of lifetime three as follows. First, we find a proper 4-edge coloring $c : E \to \{1, 2, 3, 4\}$ of G. Such a coloring exists by Vizing's theorem and can be found in $O(|E|)$ time [57]. Now the underlying graph $H = (U, F)$ contains two vertices v_0 and v_1 for every vertex v of G, and one vertex w_e for every edge e of G. The set F of the edges of H contains $\{v_0, v_1\}$ for every $v \in V$, and for every edge $e = \{u, v\} \in E$ it contains $\{w_e, u_\alpha\}, \{w_e, v_\alpha\}$, where $c(e) \equiv \alpha \pmod 2$. In temporal graph (H, λ) every edge of the underlying graph appears in exactly one of the three time slots:

1. $\lambda(\{w_e, u_\alpha\}) = \lambda(\{w_e, v_\alpha\}) = 1$, where $c(e) \equiv \alpha \pmod 2$, for every edge $e = \{u, v\} \in E$ such that $c(e) \in \{1, 2\}$;
2. $\lambda(\{v_0, v_1\}) = 2$ for every $v \in V$;
3. $\lambda(\{w_e, u_\alpha\}) = \lambda(\{w_e, v_\alpha\}) = 3$, where $c(e) \equiv \alpha \pmod 2$, for every edge $e = \{u, v\} \in E$ such that $c(e) \in \{3, 4\}$.

Construction 1 is illustrated in Figure 4 in Appendix B. We defer the proof that the Construction 1 is indeed an L-reduction to Appendix B.1. It is easy to check that the reduction also implies NP-completeness of Temporal Matching. We show the lower bound on the approximation ratio in Appendix B.2. We show the running time lower bound based on the Exponential Time Hypothesis (ETH) in Appendix B.3. This concludes the proof of Theorem 5.

▶ **Observation 6** ($\star$). Temporal Matching *is NP-complete, even if $\Delta = 2$, $T = 5$, and the underlying graph of the input temporal graph is complete.*

The importance of this observation is due to the following parameterized complexity implication. Parameterizing Temporal Matching by structural graph parameters of the underlying graph that are constant on complete graphs cannot yield fixed-parameter tractability unless P = NP, even if combined with the lifetime T. Note that many structural parameters fall into this category, such as domination number, distance to cluster graph, clique cover number, etc. We discuss how our reduction can be adapted to the model of Baste et al. [9] in Appendix B.5.

3.2 NP-completeness of Temporal Matching with underlying Paths

In this subsection we show NP-completeness of Temporal Matching even for a very restricted class of temporal graphs.

▶ **Theorem 7.** Temporal Matching *is NP-complete even if* $\Delta = 2$ *and the underlying graph of the input temporal graph is a path.*

We show this result by a reduction from Independent Set on connected cubic planar graphs, which is known to be NP-complete [31, 32]. More specifically, we show that Independent Set is NP-complete on the temporal line graphs of temporal graphs that have a path as underlying graph. Recall that by Observation 4, solving Independent Set on a temporal line graph is equivalent to solving Temporal Matching on the corresponding temporal graph. We proceed as follows.

1. We show that 2-temporal line graphs of temporal graphs that have a path as underlying graph have a grid-like structure. More specifically, we show that they are induced subgraphs of so-called *diagonal grid graphs* or *king's graphs* [17, 36].
2. We show that Independent Set is NP-complete on induced subgraphs of diagonal grid graphs which together with Observation 4 yields Theorem 7.
   - We exploit that cubic planar graphs are induced topological minors of grid graphs and extend this result by showing that they are also induced topological minors of diagonal grid graphs.
   - We show how to modify the subdivision of a cubic planar graph that is an induced subgraph of a diagonal grid graph such that NP-hardness of finding independent sets of certain size is preserved.

▶ **Definition 8** (Diagonal Grid Graph [17, 36]). *A* diagonal grid graph $\widehat{Z}_{n,m}$ *has a vertex* $v_{i,j}$ *for all* $i \in [n]$ *and* $j \in [m]$ *and there is an edge* $\{v_{i,j}, v_{i',j'}\}$ *if and only if* $|i-i'|^2 + |j-j'|^2 \leq 2$.

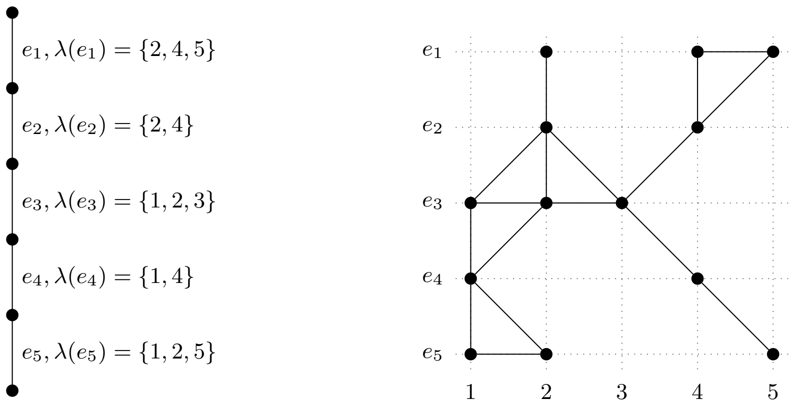
It is easy to check that for a temporal graph with a path as underlying graph and where each edge is active at every time step, the 2-temporal line graph is a diagonal grid graph. For a visualization see Figure 5 in Appendix B.

▶ **Observation 9.** *Let* $\mathcal{G} = (P_n, \lambda)$ *with* $\lambda(e) = [T]$ *for all* $e \in E(P_n)$*, then* $L_2(\mathcal{G}) = \widehat{Z}_{n-1,T}$.

Further, it is easy to see that deactivating an edge at a certain point in time results in removing the corresponding vertex from the diagonal grid graph. See Figure 1 for an example. Hence, we have that every induced subgraph of a diagonal grid graph is a 2-temporal line graph.

▶ **Corollary 10.** *Let* Z' *be a connected* induced *subgraph of* $\widehat{Z}_{n-1,T}$*. Then there is a* λ *and an* $n' \leq n$ *such that* $Z' = L_2((P_{n'}, \lambda))$.

Having these results at hand, it suffices to show that Independent Set is NP-complete on induced subgraphs of diagonal grid graphs. By Observation 4, this directly implies that Temporal Matching is NP-complete on temporal graphs that have a path as underlying graph. Hence, in the remainder of this section, we show the following result.



(a) Temporal graph $\mathcal{G} = (P_6, \lambda)$ with λ as visualized.

(b) 2-Temporal line graph $L_2(\mathcal{G})$.

Figure 1 A temporal line graph with a path as underlying graph where edges are *not* always active and its 2-temporal line graph.

▶ **Theorem 11** ($\star$). Independent Set *on induced subgraphs of diagonal grid graphs is NP-complete.*

This result may be of independent interest and strengthens a result by Clark et al. [19], who showed that Independent Set is NP-complete on unit disk graphs. It is easy to see from Definition 8 that diagonal grid graphs and their induced subgraphs are a (proper) subclass of unit disk graphs.

In the following, we give the main ideas of how we prove Theorem 11. A formal proof is deferred to Appendix B.6. The first building block for the reduction is the fact that we can embed cubic planar graphs into a grid [59]. More specifically, a cubic planar graph admits a planar embedding in such a way that the vertices are mapped to points of a grid and the edges are drawn along the grid lines. Moreover, such an embedding can be computed in polynomial time and the size of the grid is polynomially bounded in the size of the planar graph.

Note that if we replace the edges of the original planar graph by paths of appropriate length, then the embedding in the grid is actually a subgraph of the grid. Furthermore, if we scale the embedding by a factor of two, i.e. subdivide every edge once, then the embedding is also guaranteed to be an *induced* subgraph of the grid. In other words, we argue that every cubic planar graph is an induced topological minor of a polynomially large grid graph. We then show how to modify the embedding in a way that insures that the resulting graph is also an induced topological minor of an polynomially larg *diagonal* grid graph. The last step is to further modify the embedding such that it can be optained from the original graph by subdividing each edge an even number of times, this ensures that NP-hardness of Independent Set is preserved [55].

It is easy to check that Theorem 11, Observation 4, and Corollary 10 together imply Theorem 7. Theorem 7 also has some interesting implications from the point of view of parameterized complexity: Parameterizing Temporal Matching by structural graph parameters of the underlying graph that are constant on a path cannot yield fixed-parameter tractability unless P = NP, even if combined with Δ. Note that a large number of popular structural parameters fall into this category, such as maximum degree, treewidth, pathwidth, feedback vertex number, etc.

Algorithm 4.1: $\frac{\Delta}{2\Delta-1}$-Approximation Algorithm (Theorem 12).

```
1 M ← ∅.
2 foreach Δ-template S do
3     Compute a Δ-temporal matching M^S with respect to S.
4     if |M^S| > |M| then M ← M^S.
5 return M.
```

4 Algorithms

4.1 Approximation of Maximum Temporal Matching

In this section, we present a $\frac{\Delta}{2\Delta-1}$-approximation algorithm for Maximum Temporal Matching, Note that, for $\Delta = 2$ this is a $\frac{2}{3}$-approximation, while for arbitrary constant Δ this is a $(\frac{1}{2} + \varepsilon)$-approximation, where $\varepsilon = \frac{1}{2(2\Delta-1)}$ is a constant too. Specifically, we show the following.

▶ **Theorem 12.** Maximum Temporal Matching *admits an* $O(Tm(\sqrt{n} + \Delta))$*-time* $\frac{\Delta}{2\Delta-1}$*-approximation algorithm.*

The main idea of our approximation algorithm is to compute maximum matchings for slices of size Δ of the input temporal graph that are sufficiently far apart from each other such that they do not interfere with each other, and hence are computable in polynomial time. Then we greedily fill up the gaps. We try out certain combinations of non-interfering slices of size Δ in a systematic way and then take the largest Δ-matching that was found in this way. With some counting arguments we can show that this achieves the desired approximation ratio. In the following we describe and prove this claim formally.

We first introduce some additional notation and terminology. Recall that $\mu_\Delta(\mathcal{G})$ denotes the size of a maximum Δ-temporal matching in $\mathcal{G}$. Let Δ and T be fixed natural numbers such that $\Delta \leq T$. For every time slot $t \in [T - \Delta + 1]$, we define the Δ*-window* W_t as the interval $[t, t + \Delta - 1]$ of length Δ. We use this to formalize slices of size Δ of a temporal graph. An interval of length at most $\Delta - 1$ that either starts at slot 1, or ends at slot T is called a *partial* Δ*-window (with respect to lifetime* T*)*. For the sake of brevity, we write *partial* Δ*-window*, when the lifetime T is clear from the context. The *distance* between two disjoint intervals $[a_1, b_1]$ and $[a_2, b_2]$ with $b_1 < a_2$ is $a_2 - b_1 - 1$.

A Δ*-template (with respect to lifetime* T*)* is a maximal family $\mathcal{S}$ of Δ-windows or partial Δ-windows in the interval $[T]$ such that any two consecutive elements in $\mathcal{S}$ are at distance exactly $\Delta - 1$ from each other. Let $\mathcal{S}$ be a Δ-template. A Δ-temporal matching $M^{\mathcal{S}}$ in $\mathcal{G} = (G, \lambda)$ is called a Δ-temporal matching *with respect to* Δ*-template* $\mathcal{S}$ if $M^{\mathcal{S}}$ has the maximum possible number of edges in every interval $W \in \mathcal{S}$, i.e. $\left|M^{\mathcal{S}}|_W\right| = \mu_\Delta(\mathcal{G}|_W)$ for every $W \in \mathcal{S}$.

Now we are ready to present and analyze our $\frac{\Delta}{2\Delta-1}$-approximation algorithm, see Algorithm 4.1. The idea of the algorithm is simple: for every Δ-template $\mathcal{S}$ compute a Δ-temporal matching $M^{\mathcal{S}}$ with respect to $\mathcal{S}$ and among all of the computed Δ-temporal matchings return a matching of the maximum cardinality. The proof of correctness of Algorithm 4.1 is deferred to Appendix C.1.

We remark that our analysis ignores the fact that the algorithm may add time-edges from the gaps between the Δ-windows defined by the template to the matching if they are not in conflict with any other edge in the matching. Hence, there is potential room for improvement.

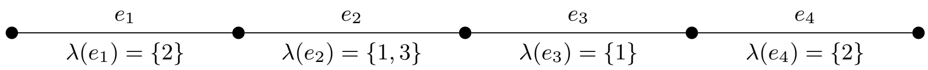


Figure 2 A temporal graph witnessing that the analysis of Algorithm 4.1 is tight for $\Delta = 2$.

On the other hand, our analysis of the approximation factor of Algorithm 4.1 is tight for $\Delta = 2$. Namely, there exists a temporal graph $\mathcal{G}$ (see Figure 2) such that on the instance $(\mathcal{G}, 2)$ our algorithm (in the worst case) finds a 2-temporal matching of size two, while the size of a maximum 2-temporal matching in $\mathcal{G}$ is three. In this example any improvement of the algorithm that utilizes the gaps between the Δ-windows would not lead to a better performance (see Appendix C.2).

4.2 Fixed-parameter tractability for the parameter solution size

In this section we provide a fixed-parameter algorithm for Temporal Matching parameterized by the solution size k. More specifically, we provide a linear-time algorithm for a fixed solution size k. Formally, the main result of this subsection is to show the following.

▶ **Theorem 13** ($\star$). *There is a linear-time* FPT-*algorithm for* Temporal Matching *parameterized by the solution size k.*

We prove Theorem 13 in the remainder of this section. Recall that due to Baste et al. [9] it is already known that Temporal Matching is fixed-parameter tractable when parameterized by the solution size k *and* Δ. In comparison to the algorithm of Baste et al. [9] the running time of our algorithm is independent of Δ, hence improving their result from a parameterized classification standpoint.

The rough idea of our algorithm is the following. We develop a preprocessing procedure that reduces the number of time-edges of the first Δ-window. After applying this procedure, the number of time-edges in the first Δ-window is bounded in a function of the solution size parameter k. This allows us to enumerate all possibilities to select time-edges from the first Δ-window for the temporal matching. Then, for each possibility, we can remove the first Δ-window from the temporal graph and solve the remaining part recursively.

Next, we describe the preprocessing procedure more precisely. Referring to kernelization algorithms, we call this procedure *kernel for the first* Δ*-window*. If we count naively the number of Δ-temporal matchings in the first Δ-window of a temporal graph, then this number clearly depends on Δ. This is too large for Theorem 13. A key observation to overcome this obstacle is that if we look at an edge appearance of a Δ-temporal matching which comes from the first Δ-window, then we can exchange it with the first appearance of the edge.

▶ **Lemma 14** ($\star$). *Let (G, λ) be a temporal graph and let M be a Δ-temporal matching in (G, λ). Let also $e \in E_{t_1} \cap E_{t_2}$, where $t_1 < t_2 \leq \Delta$. If $(e, t_1) \notin M$ and $(e, t_2) \in M$, then $M' = (M \setminus \{(e, t_2)\}) \cup \{(e, t_1)\}$ is a Δ-temporal matching in (G, λ).*

We use Lemma 14 to construct a small set K of time-edges from the first Δ-window such that there exists a maximum Δ-temporal matching M in (G, λ) with the property that the restriction of M to the first Δ-window is contained in K.

▶ **Definition 15** (Kernel for the First Δ-Window). *Let Δ be a natural number and let $\mathcal{G}$ be a temporal graph. We call a set K of time-edges of $\mathcal{G}|_{[1,\Delta]}$ a* kernel for the first Δ-window of $\mathcal{G}$ *if there exists a maximum Δ-temporal matching M in $\mathcal{G}$ with $M|_{[1,\Delta]} \subseteq K$.*

Algorithm 4.2: Kernel for the First Δ-Window (Lemma 16).

1 Let G' be the underlying graph of $\mathcal{G}|_{[1,\Delta]}$. and $K = \emptyset$
2 $A \leftarrow$ a maximum matching of G'.
3 $V_A \leftarrow$ the set of vertices matched by A.
4 **foreach** $v \in V_A$ **do**
5 $\quad R_v \leftarrow \big\{(\{v,w\},t) \mid w \in N_{G'}(v) \text{ and } t = \min\{i \in [\Delta] \mid \{v,w\} \in E_i\}\big\}$.
6 $\quad$ **if** $|R_v| \leq 4\nu$ **then** $K \leftarrow K \cup R_v$.
7 $\quad$ **else**
8 $\quad\quad$ Form a subset $R' \subseteq R_v$ such that $|R'| = 4\nu + 1$ and for every $(e,t) \in R'$ and $(e',t') \in R_v \setminus R'$ we have $t \leq t'$.
9 $\quad\quad K \leftarrow K \cup R'$.
10 **return** K.

Informally, the idea for computing the kernel for the first Δ-window is to first select vertices that are suitable to be matched. Then, for each of these vertices, we select the earliest appearance of a sufficiently large number of incident time-edges, where each of these time-edges corresponds to a different edge of the underlying graph. We show that we can do this in a way that the number of selected time-edges can be bounded in the size ν of a maximum matching of the underlying graph G. Formally, we aim at proving the following lemma.

▶ **Lemma 16** ($\star$). *Given a natural number Δ and a temporal graph $\mathcal{G} = (G, \lambda)$ we can compute in $O(\nu^2 \cdot |\mathcal{G}|)$ time a kernel K for the first Δ-window of $\mathcal{G}$ such that $|K| \in O(\nu^2)$.*

Algorithm 4.2 presents the pseudocode for the algorithm behind Lemma 16. We show correctness of Algorithm 4.2 in Lemma 17 and examine its running time in Lemma 18. Hence, Lemma 16 follows from Lemmas 17 and 18.

▶ **Lemma 17.** *Algorithm 4.2 is correct, that is, the algorithm outputs a size-$O(\nu^2)$ kernel K for the first Δ-window of $\mathcal{G}$.*

Proof. Let M be a maximum Δ-temporal matching of $\mathcal{G}$ such that $\big|M|_{[1,\Delta]} \setminus K\big|$ is minimized. Without loss of generality we can assume that every time-edge in $M|_{[1,\Delta]}$ is the first appearance of an edge. Indeed, by construction, K contains only the first appearances of edges, and therefore if $(e,t) \in M|_{[1,\Delta]}$ is not the first appearance of e, by Lemma 14 it can be replaced by the first appearance, and this would not increase $\big|M|_{[1,\Delta]} \setminus K\big|$. Now, assume towards a contradiction that $M|_{[1,\Delta]} \setminus K$ is not empty and let (e,t) be a time-edge in $M|_{[1,\Delta]} \setminus K$. Since A is a maximum matching in the underlying graph G' of $\mathcal{G}|_{[1,\Delta]}$, at least one of the end vertices of e is matched by A, i.e. belongs to V_A. Then for a vertex $v \in V_A \cap e$ we have that $(e,t) \in R_v$. Moreover, observe that $|R_v| > 4\nu$, because otherwise (e,t) would be in K. For the same reason $(e,t) \notin R'$, where $R' \subseteq R_v$ is the set of time-edges computed in Line 8 of the algorithm. Let $W = \{(w,t) \mid (\{v,w\},t) \in R'\}$ be the set of vertex appearances which are adjacent to vertex appearance (v,t) by a time-edge in R'. Since R_v contains only the first appearances of edges, we know that W contains exactly $4\nu + 1$ vertex appearances of pairwise different vertices.

We now claim that W contains a vertex appearance which is not Δ-blocked by any time-edge in M. To see this, we recall that ν is the maximum matching size of the underlying graph of $\mathcal{G}$. Hence it is also an upper bound on the number of time-edges in $M|_{[1,\Delta]}$ and $M|_{[\Delta+1,2\Delta]}$, which implies that in the first Δ-window vertex appearances of at most 4ν distinct vertices are Δ-blocked by time-edges in M. Since W contains $4\nu + 1$ vertex appearances of pairwise different vertices, we conclude that there exists a vertex appearance $(w', t') \in W$ which is not Δ-blocked by M.

Observe that $t' \leq t$ because $(\{v, w'\}, t') \in R'$ and $(e, t) \in R_v \setminus R'$. Hence, (v, t') is not Δ-blocked by $M \setminus \{(e, t)\}$. Thus, $M^* := (M \setminus \{(e, t)\}) \cup \{(\{v, w'\}, t')\}$ is a Δ-temporal matching of size $|M|$ with $\left|M^*|_{[1,\Delta]} \setminus K\right| < \left|M|_{[1,\Delta]} \setminus K\right|$. This contradiction implies that $M|_{[1,\Delta]} \setminus K$ is empty and thus $M|_{[1,\Delta]} \subseteq K$.

It remains to show that $|K| \in O(\nu^2)$. Since each maximum matching in G' has at most ν edges, we have that $|V_A| \leq 2\nu$. For each vertex in V_A the algorithm adds at most $4\nu + 1$ time-edges to K. Thus, $|K| \leq 2\nu \cdot (4\nu + 1) \in O(\nu^2)$. ◀

▶ **Lemma 18** ($\star$). *Algorithm 4.2 runs in $O(\nu^2(n + m\Delta))$ time. In particular, the time complexity of Algorithm 4.2 is dominated by $O(\nu^2|\mathcal{G}|)$.*

Having Algorithm 4.2 at hand, we can formulate a recursive search tree algorithm which (1) picks a Δ-temporal matchings M in the kernel of the first Δ-window, (2) removes the first Δ-window from the temporal graph, (3) removes all time-edges which are not Δ-independent with M, and (4) calls itself until the temporal graph in empty. The pseudocode of this algorithm and the proof of correctness is deferred to Appendix C.5.

4.3 Fixed-parameter tractability for the combined parameter Δ and maximum matching size ν of the underlying graph

In this subsection, we show that Temporal Matching is fixed-parameter tractable when parameterized by Δ and the maximum matching size ν of the underlying graph.

▶ **Theorem 19** ($\star$). Temporal Matching *can be solved in $2^{O(\nu\Delta)} \cdot |\mathcal{G}| \cdot \frac{T}{\Delta}$ time.*

The proof of Theorem 19 is deferred to the end of this section. Note that Theorem 19 implies that Temporal Matching is fixed-parameter tractable when parameterized by Δ and the maximum matching size ν of the underlying graph, because there is a simple preprocessing step such that we can assume afterwards that the lifetime T is polynomially bounded in the input size. This preprocessing step modifies the temporal graph such that it does not contain Δ consecutive edgeless snapshots. This can be done by iterating once over the temporal graph. Observe, that this procedure does not change the maximum size of a Δ-temporal matching and afterwards each Δ-window contains at least one time-edge. Hence, $\frac{T}{\Delta} \leq |\mathcal{G}|$.

Note that this result is incomparable to the result from the previous subsection (Theorem 13). In some sense, we trade off replacing the solution size parameter k with the structurally smaller parameter ν but we do not know how to do this without combining it with Δ. In comparison to the exact algorithm by Baste et al. [9] (who showed fixed-parameter tractability with k and Δ) we replace k by the structurally smaller ν, hence improving their result from a parameterized classification standpoint. Furthermore, we note that Theorem 19 is asymptotically optimal for any fixed Δ since there is no $2^{o(\nu)} \cdot |\mathcal{G}|^{f(\Delta,T)}$ algorithm for Temporal Matching, unless ETH fails (see Appendix B.3).

In the reminder of this section, we sketch the main ideas of the algorithm behind Theorem 19. The algorithm works in three major steps:

1. The temporal graph is divided into disjoint Δ-windows,
2. for each of these Δ-windows a small family of Δ-temporal matchings is computed, and then
3. the maximum size of a Δ-temporal matching for the whole temporal graph is computed with a dynamic program.

We first discuss how the algorithm performs Step 2. Afterwards we formulate the dynamic program (Step 3) and prove Theorem 19. In a nutshell, Step 2 consists of an iterative computation of a small (upper-bounded in $\Delta + \nu$) family of Δ-temporal matchings for an arbitrary Δ-window such that at least one of them is "extendable" to a maximum Δ-temporal matching for the whole temporal graph.

Families of ℓ-complete Δ-temporal matchings. Throughout this section let $\mathcal{G} = (G = (V, E), \lambda)$ be a temporal graph of lifetime T and let ν be the maximum matching size in G. Let also Δ and ℓ be natural numbers such that $\ell\Delta \leq T$.

A family $\mathcal{M}$ of Δ-temporal matchings of $\mathcal{G}|_{[\Delta(\ell-1)+1,\Delta\ell]}$ is called *ℓ-complete* if for any Δ-temporal matching M of $\mathcal{G}$ there is $M' \in \mathcal{M}$ such that $\left(M \setminus M|_{[\Delta(\ell-1)+1,\Delta\ell]}\right) \cup M'$ is a Δ-temporal matching of $\mathcal{G}$ of size at least $|M|$. A central part of our algorithm is an efficient procedure for computing an ℓ-complete family. Formally, we aim for the following lemma.

▶ **Lemma 20** ($\star$)**.** *There exists a $2^{O(\nu\Delta)} \cdot |\mathcal{G}|$-time algorithm that computes an ℓ-complete family of size $2^{O(\nu\Delta)}$ of Δ-temporal matchings of $\mathcal{G}|_{[\Delta(\ell-1)+1,\Delta\ell]}$.*

In the proof of Lemma 20 we employ representative families and other tools from matroid theory.

Dynamic program. Now we are ready to combine Step 2 of our algorithm with the remaining Steps 1 and 3. More precisely, we employ ℓ-complete families of Δ-temporal matchings of Δ-windows in a dynamic program (Step 3) to compute the Δ-temporal matching of maximum size for the whole temporal graph. The pseudocode of this dynamic program and its proof of correctness is stated in Appendix C.8. This is the algorithm behind Theorem 19. It computes a table $\mathcal{T}$ where each entry $\mathcal{T}[i, M']$ stores the maximum size of a Δ-temporal matching M in the temporal graph $\mathcal{G}|_{[1,\Delta i]}$ such that all the time-edges in $M|_{[\Delta(i-1)+1,\Delta i]} = M'$. Observe that a trivial dynamic program which computes all entries of $\mathcal{T}$ cannot provide fixed-parameter tractability of Temporal Matching when parameterized by Δ and ν, because the corresponding table is simply too large. The crucial point of the dynamic program is that it is sufficient to fix for each $i \in \frac{T}{\Delta}$ an i-complete family $\mathcal{M}_i$ of Δ-temporal matchings for $\mathcal{G}|_{[\Delta(i-1)+1,\Delta i]}$ and then compute only the entries $\mathcal{T}[i, M']$, where $M' \in \mathcal{M}_i$.

Kernelization lower bound. Lastly, we can show that we cannot hope to obtain a polynomial kernel for the parameter combination number n of vertices and Δ. In particular, this implies that we also presumably cannot get a polynomial kernel for the parameter combination ν and Δ, since $\nu \leq \frac{n}{2}$.

▶ **Proposition 21** ($\star$)**.** Temporal Matching *parameterized by the number n of vertices does not admit a polynomial kernel for all $\Delta \geq 2$, unless NP $\subseteq$ coNP/poly.*

5 Conclusion

The following issues remain research challenges. First, on the side of polynomial-time approximability, improving the constant approximation factors is desirable and seems feasible. Beyond, lifting polynomial time to FPT time, even approximation schemes in principle seem possible, thus circumventing our APX-hardness result. Taking the view of parameterized complexity analysis in order to cope with NP-hardness, a number of directions are naturally coming up. For instance, based on our fixed-parameter tractability result for the parameter solution size, the question for the existence of a polynomial-size kernel naturally arises. For instance, based on our fixed-parameter tractability result for the parameter solution size, the following questions naturally arises:

1. Is there a polynomial-size kernel?
2. Is there a faster algorithm or a matching lower-bound for the running time of Theorem 13?

To enlarge the range of promising and relevant parameterizations, one may extend the parameterized studies to structural graph parameters combined with Δ or the lifetime of the temporal graph. In particular, treedepth combined with Δ is left open, since it is a "stronger" parameterization than in Theorem 19 but unbounded in all known NP-hardness reductions.

References

1. Eleni C. Akrida, Leszek Gasieniec, George B. Mertzios, and Paul G. Spirakis. Ephemeral networks with random availability of links: The case of fast networks. *Journal of Parallel and Distributed Computing*, 87:109–120, 2016.
2. Eleni C. Akrida, George B. Mertzios, Sotiris E. Nikoletseas, Christoforos Raptopoulos, Paul G. Spirakis, and Viktor Zamaraev. How fast can we reach a target vertex in stochastic temporal graphs? In *Proceedings of the 46th International Colloquium on Automata, Languages, and Programming (ICALP '19)*, pages 131:1–131:14, 2019.
3. Eleni C. Akrida, George B. Mertzios, Paul G. Spirakis, and Viktor Zamaraev. Temporal vertex cover with a sliding time window. In *Proceedings of the 45th International Colloquium on Automata, Languages, and Programming (ICALP '18)*, pages 148:1–148:14, 2018.
4. Paola Alimonti and Viggo Kann. Some APX-completeness results for cubic graphs. *Theoretical Computer Science*, 237(1-2):123–134, 2000.
5. Jonathan Aronson, Martin Dyer, Alan Frieze, and Stephen Suen. Randomized greedy matching ii. *Random Structures & Algorithms*, 6(1):55–73, 1995.
6. Giorgio Ausiello, Pierluigi Crescenzi, Giorgio Gambosi, Viggo Kann, Alberto Marchetti-Spaccamela, and Marco Protasi. *Complexity and Approximation: Combinatorial Optimization Problems and Their Approximability Properties.* Springer Science & Business Media, 2012.
7. Kyriakos Axiotis and Dimitris Fotakis. On the size and the approximability of minimum temporally connected subgraphs. In *Proceedings of the 43rd International Colloquium on Automata, Languages, and Programming (ICALP '16)*, pages 149:1–149:14, 2016.
8. Evripidis Bampis, Bruno Escoffier, Michael Lampis, and Vangelis Th. Paschos. Multistage matchings. In *Proceedings of the 16th Scandinavian Symposium and Workshops on Algorithm Theory (SWAT '18)*, volume 101 of *LIPIcs*, pages 7:1–7:13. Schloss Dagstuhl - Leibniz-Zentrum fuer Informatik, 2018.
9. Julien Baste, Binh-Minh Bui-Xuan, and Antoine Roux. Temporal matching. *Theoretical Computer Science*, 2019.
10. Matthias Bentert, Anne-Sophie Himmel, Hendrik Molter, Marco Morik, Rolf Niedermeier, and René Saitenmacher. Listing all maximal k-plexes in temporal graphs. In *Proceedings of the 2018 IEEE/ACM International Conference on Advances in Social Networks Analysis and Mining (ASONAM '18)*, pages 41–46, 2018.
11. Hans L Bodlaender, Bart MP Jansen, and Stefan Kratsch. Kernelization lower bounds by cross-composition. *SIAM Journal on Discrete Mathematics*, 28(1):277–305, 2014.
12. Preston Briggs and Linda Torczon. An efficient representation for sparse sets. *ACM Letters on Programming Languages and Systems*, 2:59–69, 1993.
13. Niv Buchbinder, Danny Segev, and Yevgeny Tkach. Online algorithms for maximum cardinality matching with edge arrivals. *Algorithmica*, 81(5):1781–1799, 2019.
14. Arnaud Casteigts and Paola Flocchini. Deterministic Algorithms in Dynamic Networks: Formal Models and Metrics. Technical report, Defence R&D Canada, April 2013. URL: `https://hal.archives-ouvertes.fr/hal-00865762`.
15. Arnaud Casteigts and Paola Flocchini. Deterministic Algorithms in Dynamic Networks: Problems, Analysis, and Algorithmic Tools. Technical report, Defence R&D Canada, April 2013. URL: `https://hal.archives-ouvertes.fr/hal-00865764`.

16 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.

17 Gerard Jennhwa Chang. Algorithmic aspects of domination in graphs. *Handbook of Combinatorial Optimization*, pages 221–282, 2013.

18 Miroslav Chlebík and Janka Chlebíková. Complexity of approximating bounded variants of optimization problems. *Theoretical Computer Science*, 354(3):320–338, 2006.

19 Brent N. Clark, Charles J. Colbourn, and David S. Johnson. Unit disk graphs. *Discrete Mathematics*, 86(1-3):165–177, 1990.

20 Andrea E. F. Clementi, Claudio Macci, Angelo Monti, Francesco Pasquale, and Riccardo Silvestri. Flooding time of edge-markovian evolving graphs. *SIAM Journal on Discrete Mathematics*, 24(4):1694–1712, 2010.

21 David Coudert, Guillaume Ducoffe, and Alexandru Popa. Fully polynomial FPT algorithms for some classes of bounded clique-width graphs. In *Proceedings of the 29th Annual ACM-SIAM Symposium on Discrete Algorithms (SODA '18)*, pages 2765–2784, 2018.

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

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

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

25 Thomas Erlebach, Michael Hoffmann, and Frank Kammer. On temporal graph exploration. In *Proceedings of the 42nd International Colloquium on Automata, Languages, and Programming (ICALP '15)*, pages 444–455, 2015.

26 Afonso Ferreira. Building a reference combinatorial model for MANETs. *IEEE Network*, 18(5):24–29, 2004.

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

28 Fedor V. Fomin, Daniel Lokshtanov, Saket Saurabh, MichaŁPilipczuk, and Marcin Wrochna. Fully polynomial-time parameterized computations for graphs and matrices of low treewidth. *ACM Transactions on Algorithms*, 14(3):34:1–34:45, 2018.

29 Harold N. Gabow and Robert Endre Tarjan. A linear-time algorithm for a special case of disjoint set union. *Journal of Computer and System Sciences*, 30(2):209–221, 1985.

30 Buddhima Gamlath, Michael Kapralov, Andreas Maggiori, Ola Svensson, and David Wajc. *Online matching with general arrivals*, 2019. Technical Report available at `https://arxiv.org/abs/1904.08255`.

31 Michael R. Garey and David S. Johnson. The rectilinear Steiner tree problem is NP-complete. *SIAM Journal on Applied Mathematics*, 32(4):826–834, 1977.

32 Michael R. Garey and David S. Johnson. *Computers and Intractability: A Guide to the Theory of NP-Completeness*. W. H. Freeman, 1979.

33 Michael R Garey, David S Johnson, and Larry Stockmeyer. Some simplified NP-complete problems. In *Proceedings of the sixth annual ACM Symposium on Theory of Computing (STOC '74)*, pages 47–63. ACM, 1974.

34 George Giakkoupis, Thomas Sauerwald, and Alexandre Stauffer. Randomized rumor spreading in dynamic graphs. In *Proceedings of the 41st International Colloquium on Automata, Languages and Programming (ICALP '14)*, pages 495–507, 2014.

35 Archontia C. Giannopoulou, George B. Mertzios, and Rolf Niedermeier. Polynomial fixed-parameter algorithms: A case study for longest path on interval graphs. In *Proceedings of the 10th International Symposium on Parameterized and Exact Computation (IPEC '15)*, pages 102–113, 2015.

36 Daifeng Guo, Hongbo Zhang, and Martin D.F. Wong. On coloring rectangular and diagonal grid graphs for multiple patterning lithography. In *Proceedings of the 23rd Asia and South Pacific Design Automation Conference (ASP-DAC '18)*, pages 387–392. IEEE Press, 2018.

37 Anupam Gupta, Kunal Talwar, and Udi Wieder. Changing bases: Multistage optimization for matroids and matchings. In *Proceedings of the 41st International Colloquium on Automata, Languages, and Programming (ICALP '14)*, pages 563–575. Springer, 2014.

38 Dirk Hausmann and Bernhard Korte. k-greedy algorithms for independence systems. *Zeitschrift für Operations Research*, 22(1):219–228, 1978.

39 Zhiyi Huang, Ning Kang, Zhihao Gavin Tang, Xiaowei Wu, Yuhao Zhang, and Xue Zhu. How to match when all vertices arrive online. In *Proceedings of the 50th Annual ACM Symposium on Theory of Computing (STOC '18)*, pages 17–29, 2018.

40 Zhiyi Huang, Binghui Peng, Zhihao Gavin Tang, Runzhou Tao, Xiaowei Wu, and Yuhao Zhang. Tight competitive ratios of classic matching algorithms in the fully online model. In *Proceedings of the 30th Annual ACM-SIAM Symposium on Discrete Algorithms (SODA)*, pages 2875–2886, 2019.

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

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

43 Richard M. Karp, Umesh V. Vazirani, and Vijay V. Vazirani. An optimal algorithm for on-line bipartite matching. In *Proceedings of the 22nd Annual ACM Symposium on Theory of Computing (STOC '90)*, pages 352–358, 1990.

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

45 Viatcheslav Korenwein, André Nichterlein, Rolf Niedermeier, and Philipp Zschoche. Data reduction for maximum matching on real-world graphs: Theory and experiments. In *Proceedings of the 26th Annual European Symposium on Algorithms (ESA '18)*, pages 53:1–53:13, 2018.

46 Bernhard Korte and Dirk Hausmann. An analysis of the greedy heuristic for independence systems. *Annals of Discrete Mathematics*, 2:65–74, 1978.

47 Stefan Kratsch and Florian Nelles. Efficient and adaptive parameterized algorithms on modular decompositions. In *Proceedings of the 26th Annual European Symposium on Algorithms (ESA '18)*, volume 112, pages 55:1–55:15, 2018.

48 Jure Leskovec, Jon M. Kleinberg, and Christos Faloutsos. Graph evolution: Densification and shrinking diameters. *ACM Transactions on Knowledge Discovery from Data*, 1(1), 2007.

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

50 George B. Mertzios, Othon Michail, Ioannis Chatzigiannakis, and Paul G. Spirakis. Temporal network optimization subject to connectivity constraints. *Algorithmica*, pages 1416–1449, 2019.

51 George B. Mertzios, Hendrik Molter, and Viktor Zamaraev. Sliding window temporal graph coloring. In *Proceedings of the 33rd AAAI Conference on Artificial Intelligence (AAAI '19)*, pages 7667–7674, 2019.

52 George B. Mertzios, André Nichterlein, and Rolf Niedermeier. The power of linear-time data reduction for maximum matching. In *Proceedings of the 42nd International Symposium on Mathematical Foundations of Computer Science (MFCS '17)*, pages 46:1–46:14, 2017.

53 Silvio Micali and Vijay V Vazirani. An $O(\sqrt{|V|} \cdot |E|)$ algorithm for finding maximum matching in general graphs. In *Proceedings of the 21st Annual Symposium on Foundations of Computer Science (FOCS '80)*, pages 17–27. IEEE, 1980.

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

55 Svatopluk Poljak. A note on stable sets and colorings of graphs. *Commentationes Mathematicae Universitatis Carolinae*, 15(2):307–309, 1974.

56 Matthias Poloczek and Mario Szegedy. Randomized greedy algorithms for the maximum matching problem with new analysis. In *Proceedings of the 53rd Annual Symposium on Foundations of Computer Science (FOCS '12)*, pages 708–717. IEEE, 2012.

57 Alexander Schrijver. Bipartite edge coloring in $O(\Delta m)$ time. *SIAM Journal on Computing*, 28(3):841–846, 1998.

58 John Kit Tang, Mirco Musolesi, Cecilia Mascolo, and Vito Latora. Characterising temporal distance and reachability in mobile and online social networks. *ACM Computer Communication Review*, 40(1):118–124, 2010.

59 Leslie G. Valiant. Universality considerations in VLSI circuits. *IEEE Transactions on Computers*, 100(2):135–140, 1981.

60 René van Bevern, Oxana Yu. Tsidulko, and Philipp Zschoche. Fixed-parameter algorithms for maximum-profit facility location under matroid constraints. In *Proceedings of the 11th International Conference on Algorithms and Complexity (CIAC '19)*, pages 62–74, 2019.

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. In *Proceedings of the 43rd International Symposium on Mathematical Foundations of Computer Science (MFCS '18)*, volume 117, pages 45:1–45:17, 2018.

A Additional Material for Section 2

A.1 Extended Preliminaries

Basic Notation Let $\mathbb{N}$ denote the natural numbers without zero. We refer to a set of consecutive natural numbers $[i, j] = \{i, i+1, \dots, j\}$ for some $i, j \in \mathbb{N}$ with $i \le j$ as an *interval*, and to the number $j - i + 1$ as the *length* of the interval. If $i = 1$, then we denote $[i, j]$ by $[j]$. By $\mathbb{F}_p$ we denote the finite field on p elements. For the sake of brevity, the notation $A \uplus B$ denotes the union of two sets A and B and implicitly indicates that the sets are disjoint. We call a family of sets $Z_1, \dots, Z_\ell$ a *partition* of a set A if $Z_1 \uplus \dots \uplus Z_\ell = A$ and $Z_i \neq \emptyset$ for each $i \in \{1, \dots, \ell\}$. A *p-family* is a family of sets where each set is of size exactly p.

Static graphs. We use standard notation and terminology from graph theory [23]. Given an undirected (static) graph $G = (V, E)$ with $E \subseteq \binom{V}{2}$, we denote by $V(G) = V$ and $E(G) = E$ the sets of its vertices and edges, respectively. We call two vertices $u, v \in V$ *adjacent* if $\{u, v\} \in E$. We call two edges $e_1, e_2 \in E$ *adjacent* if $e_1 \cap e_2 \neq \emptyset$. By P_n we denote a graph that is a path with n vertices. By $\nu(G)$ we denote the size of a maximum matching in G. Whenever it is clear from the context, we omit G.

Two graphs $G_1 = (V_1, E_1)$ and $G_2 = (V_2, E_2)$ are *isomorphic* if there is a bijection $\sigma : V_1 \to V_2$ such that for all $u, v \in V_1$ we have that $\{u, v\} \in E_1$ if and only if $\{\sigma(u), \sigma(v)\} \in E_2$. Given a graph $G = (V, E)$ and an edge $\{u, v\} \in E$, *subdividing* the edge $\{u, v\}$ results in a graph isomorphic to $G' = (V', E')$ with $V' = V \cup \{w\}$ for some $w \notin V$ and $E' = (E \setminus \{\{u, v\}\}) \cup \{\{v, w\}, \{u, w\}\}$. We call a graph H a *subdivision* of a graph G if there is a sequence of graphs $G_1, G_2, \dots, G_x$ with $G_1 = G$ such that for each $G_i = (V_i, E_i)$ with $i < x$ there is an edge $e \in E_i$ and subdividing e results in a graph isomorphic to G_{i+1}, and G_x is isomorphic to H. We call H a *topological minor* of G if there is a subgraph G' of G that is a subdivision of H. We call H an *induced topological minor* of G if there is an *induced* subgraph G' of G that is a subdivision of H.

A *line graph* of a (static) graph $G = (V, E)$ is a graph $L(G)$ with $V(L(G)) = \{v_e \mid e \in E\}$ and for all $v_e, v_{e'} \in V(L(G))$ we have that $\{v_e, v_{e'}\} \in E(L(G))$ if and only if $e \cap e' \neq \emptyset$ [23]. Recall that a *maximum independent set* of a (static) graph $G = (V, E)$ is a vertex set $V' \subseteq V$ of maximum cardinality such that for all $u, v \in V'$ we have that $\{u, v\} \notin E$. In the context of matchings, line graphs are of special interest since the cardinality of a maximum matching in a graph equals the cardinality of a maximum independent set in its line graph. Indeed, a matching in a graph can directly be translated into an independent set in its line graph and vice versa [23].

Parameterized complexity. We use standard notation and terminology from parameterized complexity [22, 24]. A *parameterized problem* is a language $L \subseteq \Sigma^* \times \mathbb{N}$, where Σ is a finite alphabet. We call the second component the *parameter* of the problem. A parameterized problem is *fixed-parameter tractable* (in the complexity class FPT) if there is an algorithm that solves each instance (I, r) in $f(r) \cdot |I|^{O(1)}$ time, for some computable function f. If a parameterized problem L is NP-hard for a constant parameter value, it cannot be contained in FPT[3] unless P = NP.

A parameterized problem L admits a *polynomial kernel* if there is a polynomial-time algorithm that transforms each instance (I, r) into an instance (I', r') such that $(I, r) \in L$ if and only if $(I', r') \in L$ and $|(I', r')| \leq r^{O(1)}$.

A.2 Preliminary results and observations

Note that when the input parameter Δ in Maximum Temporal Matching is equal to 1, the problem can be solved efficiently, because it reduces to T independent instances of (static) Maximum Matching.

At the other extreme are instances $(\mathcal{G} = (G, \lambda), \Delta, k)$ in which Δ coincides with the lifetime T, i.e. $\Delta = T$. In this case the problem can also be solved in polynomial time. Indeed, a maximum Δ-temporal matching M can be found as follows:

1. Find a maximum matching R in the underlying graph G;
2. Initialize $M = \emptyset$. For every edge e in R add in the final solution M exactly one (arbitrary) time-edge (e, t), where $t \in \lambda(e)$.
3. Output M.

The time complexity of the above procedure is dominated by the time required to construct the underlying graph G and the time needed to find a maximum matching in G. The former can be done in time $O(Tm) = O(\Delta m)$. The latter can be solved in $O(\sqrt{n}m)$ [53]. Thus, we have the following.

▶ **Observation 22.** *Let $\mathcal{G} = (G, \lambda)$ be a temporal graph, and let $\Delta = T$. Then* Maximum Temporal Matching *on the instance $(\mathcal{G}, \Delta)$ can be solved in time $O(m(\sqrt{n} + T))$.*

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

▶ **Observation 23.** *For every fixed Δ, the problem* Temporal Matching *on instances $(\mathcal{G}, \Delta + 1, k)$ is at least as hard as* Temporal Matching *on instances $(\mathcal{G}, \Delta, k)$.*

---

[3] It cannot even be contained in the larger parameterized complexity class XP unless P = NP.

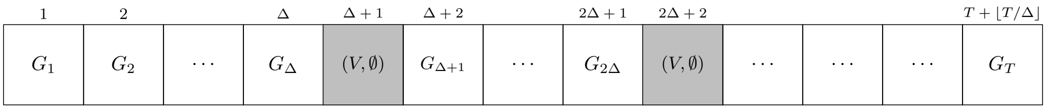


Figure 3 Inserting "empty" snapshots to reduce Temporal Matching on instances $(\mathcal{G}, \Delta, k)$ to Temporal Matching on instances $(\mathcal{G}, \Delta + 1, k)$.

Proof. The result immediately follows from the observation that a temporal graph $\mathcal{G}$ has a Δ-temporal matching of size at least k if and only if the temporal graph $\mathcal{G}'$ has a $(\Delta + 1)$-temporal matching of size at least k, where $\mathcal{G}'$ is obtained from $\mathcal{G}$ by inserting one edgeless snapshot after every Δ consecutive snapshots (see Figure 3). ◀

Lastly, it is easy to see that one can check in polynomial time whether a given set of time-edges is a Δ-temporal matching. This implies that Temporal Matching is contained in NP and in subsequent NP-completeness statements we will only discuss hardness.

A.3 Relation to γ-Matching by Baste et al. [9]

We refer to the variant of temporal matching introduced by Baste et al. [9] as γ-Matching. They defined γ-matchings in a very similar way. Their definition requires a time-edge to be present γ consecutive time slots to be eligible for a temporal matching. There is an easy reduction from their model to ours: For every sequence of γ consecutive time-edges starting at time slot t, we introduce *only one* time-edge at time slot t, and set Δ to γ. This already implies that Temporal Matching is NP-complete [9, Theorem 1] and that our algorithmic results also hold for γ-Matching. We do not know an equally easy reduction in the reverse direction.

In addition, it is easy to check that the algorithmic results of Baste et al. [9] also carry over to our model. Hence, there is a 2-approximation algorithm for Maximum Temporal Matching [9, Corollary 1] and Temporal Matching admits a polynomial kernel when parameterized by $k + \Delta$ [9, Theorem 2]. Some of our hardness results can also easily be transferred to γ-Matching. Whenever this is the case, we will indicate this.

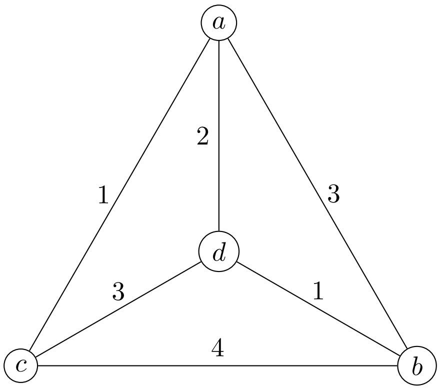
B Additional Material for Section 3

B.1 Proof that Construction 1 is an L-reduction

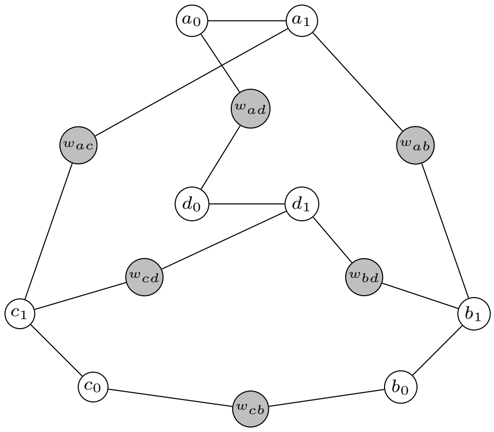
We first show that if we find a 2-temporal matching in the constructed graph (H, λ), then we can assume w.l.o.g. that if $\{u, v\} \in E$, then the temporal matching contains at most one of the time-edges $(\{u_0, u_1\}, 2)$ and $(\{v_0, v_1\}, 2)$. This will allow us to construct an independent set for the original graph G from the temporal matching.

▶ **Lemma 24.** *Let $G = (V, E)$ be a cubic graph and let (H, λ) be the temporal graph obtained by applying Construction 1 to G. Let M be a 2-temporal matching of (H, λ). Then there exists a 2-temporal matching M' of (H, λ) such that $|M'| = |M|$, and for every edge $e = \{u, v\} \in E$ the matching M' contains at most one of the time-edges $(\{u_0, u_1\}, 2)$ and $(\{v_0, v_1\}, 2)$. Moreover, M' can be constructed from M in polynomial time.*

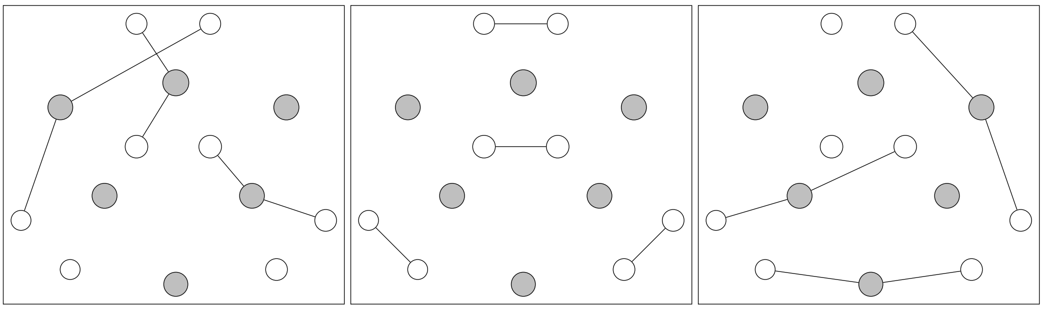
Proof. We prove the first part of the lemma by induction on the number of edges $\{u', v'\} \in E$ such that M contains both $(\{u'_0, u'_1\}, 2)$ and $(\{v'_0, v'_1\}, 2)$. Let us denote this number by k. For $k \leq 1$ the statement is trivial. Let $k > 1$, and let $e = \{u, v\} \in E$ be an edge such



(a) A cubic graph G. The edge labels correspond to the 4-edge coloring.



(b) The underlying graph H. Gray vertices correspond to edges of G, white vertices correspond to vertices of G.



(c) The temporal graph (H, λ).

Figure 4 Example of the reduction from Maximum Independent Set on cubic graphs to Maximum Temporal Matching.

that both $(\{u_0, u_1\}, 2)$ and $(\{v_0, v_1\}, 2)$ are in M. Without loss of generality we assume that $c(e) = 1$. Since the lifetime of (H, λ) is three and $(\{u_0, u_1\}, 2) \in M$, no time-edge in M other than $(\{u_0, u_1\}, 2)$ is incident with u_0 or u_1. Similarly, no time-edge in M besides $(\{v_0, v_1\}, 2)$ is incident with v_0 or v_1. In particular, $(\{w_e, u_1\}, 1), (\{w_e, v_1\}, 1) \notin M$. Hence, M'' obtained from M by replacing $(\{u_0, u_1\}, 2)$ with $(\{w_e, u_1\}, 1)$ is a 2-temporal matching of (H, λ) with $|M''| = |M|$, and the number of edges $\{u', v'\} \in E$ such that M'' contains both $(\{u'_0, u'_1\}, 2)$ and $(\{v'_0, v'_1\}, 2)$ is $k - 1$. Hence, by the induction hypothesis, there exists a desired 2-temporal matching M'.

Clearly, the above arguments can be turned into a polynomial-time algorithm that transforms M into M' by iteratively finding edges $\{u', v'\} \in E$ such that both $(\{u'_0, u'_1\}, 2)$ and $(\{v'_0, v'_1\}, 2)$ are in the current temporal matching and replacing one of the time-edges by an appropriate incident time-edge. ◀

Next, we formally show how to obtain an independent set of G from a 2-temporal matching of the constructed graph (H, λ).

▶ **Lemma 25.** *Let $G = (V, E)$ be a cubic graph and let (H, λ) be the temporal graph obtained by applying Construction 1 to G. Let M be a 2-temporal matching of (H, λ). Then G contains an independent set S of size at least $|M| - \frac{3n}{2}$. Moreover, S can be computed from M in* *polynomial time.*

Proof. First, by Lemma 24, we can assume that for every $\{u, v\} \in E$ the temporal matching M contains at most one of the time-edges $(\{u_0, u_1\}, 2)$ and $(\{v_0, v_1\}, 2)$. Now we compute in polynomial time $S := \{v \mid (\{v_0, v_1\}, 2) \in M\}$. The above assumption implies that S is an independent set.

Furthermore, notice that for every edge $e \in E$ the underlying graph H contains exactly two edges incident with w_e and both of them appear in the same time slot. Hence M can contain at most one time-edge incident with w_e, and therefore $|M| \leq |S| + |E| = |S| + \frac{3n}{2}$, which completes the proof. ◀

Now we investigate how the size of a temporal matching in the constructed graph relates to the size the corresponding independent set in the original graph.

▶ **Lemma 26.** *Let $G = (V, E)$ be a cubic graph and let (H, λ) be the temporal graph obtained by applying Construction 1 to G. Let μ_2 be the size of a maximum 2-temporal matching in (H, λ), and let α be the size of a maximum independent set in G. Then $\mu_2 = \alpha + \frac{3n}{2}$.*

Proof. We start by proving $\mu_2 \leq \alpha + \frac{3n}{2}$. Let M be a maximum 2-temporal matching of (H, λ). By Lemma 25 there exists an independent set S in G of size at least $|S| \geq |M| - \frac{3n}{2}$. Hence we have $\mu_2 = |M| \leq |S| + \frac{3n}{2} \leq \alpha + \frac{3n}{2}$.

To prove the converse inequality, we consider a maximum independent set S in G, and show how to construct a 2-temporal matching M of (H, λ) of size at least $|S| + \frac{3n}{2}$. First, for every $v \in S$ we include $(\{v_0, v_1\}, 2)$ in M. Second, for every edge $e = \{u, v\} \in E$ we add one more time-edge in M as follows. Since S is independent, at least one of u and v is not in S, say u. Then we add to M

1. $(\{w_e u_1\}, 1)$ if $c(e) = 1$,
2. $(\{w_e u_0\}, 1)$ if $c(e) = 2$,
3. $(\{w_e u_1\}, 3)$ if $c(e) = 3$, and
4. $(\{w_e u_0\}, 3)$ if $c(e) = 4$.

By construction we have $|M| = |S| + \frac{3n}{2}$. Now we show that M is a 2-temporal matching. For any two distinct vertices u and v in S the edges $\{u_0, u_1\}$ and $\{v_0, v_1\}$ are not adjacent in H, therefore the time-edges $(\{u_0, u_1\}, 2)$ and $(\{v_0, v_1\}, 2)$ are not in conflict. Furthermore, for any pair of adjacent edges $\{w_e, u_\alpha\}, \{u_0, u_1\}$ in H the corresponding time-edges are not in conflict in M, as, by construction, at most one of them is in M. For the same reason, for every edge $e = \{u, v\} \in E$ the time-edges corresponding to $\{w_e, u_\alpha\}$ and $\{w_e, v_\alpha\}$, where $c(e) \equiv \alpha \pmod 2$, are not in conflict in M. It remains to show that the time-edges $(\{w_e, u_\alpha\}, i)$ and $(\{w_{e'}, u_\alpha\}, j)$ corresponding to the adjacent edges $\{w_e, u_\alpha\}$ and $\{w_{e'}, u_\alpha\}$ in H are not in conflict in M. Suppose to the contrary that the time-edges are in conflict. Then both of them are in M and $|i - j| \leq 1$. Since by definition $i, j \in \{1, 3\}$, we conclude that $i = j$, i.e. the time-edges appear in the same time slot. Notice that e and e' share vertex u, and hence $c(e) \neq c(e')$. Hence, since $c(e) \equiv \alpha \pmod 2$ and $c(e') \equiv \alpha \pmod 2$, we conclude that either $\{c(e), c(e')\} = \{1, 3\}$, or $\{c(e), c(e')\} = \{2, 4\}$, but, by construction, this contradicts the assumption that $i = j$. This completes the proof that M is a 2-temporal matching, and therefore we have $\mu_2 \geq |M| = |S| + \frac{3n}{2} = \alpha + \frac{3n}{2}$. ◀

Lastly, we formally show that Construction 1 together with the prodecure described in Lemma 25 to obtain an independent set from a temporal matching is actually an L-reduction.

▶ **Lemma 27.** *Construction 1 together with the procedure described by Lemma 25 constitute an L-reduction.*

Proof. Recall the definition of an L-reduction. Let A and B be two maximization problems and let s_A and s_B be their respective cost functions. By definition, a pair of functions f and g is an L-reduction if all of the following conditions are met:

(1) functions f and g are computable in polynomial time;
(2) if I is an instance of problem A, then $f(I)$ is an instance of problem B;
(3) if M is a feasible solution to $f(I)$, then $g(M)$ is a feasible solution to I;
(4) there exists a positive constant β such that $OPT_B(f(I)) \leq \beta \cdot OPT_A(I)$; and
(5) there exists a positive constant γ such that for every feasible solution M to $f(I)$

$$OPT_A(I) - c_A(g(M)) \leq \gamma \cdot (OPT_B(f(I)) - c_B(M)).$$

In our case Maximum Independent Set in cubic graphs corresponds to problem A and Maximum Temporal Matching corresponds to problem B. The reduction mapping a cubic graph G to a temporal graph (H, λ) described in Construction 1 corresponds to function f. Clearly, the reduction is computable in polynomial time. The polynomial-time procedure guaranteed by Lemma 25 corresponds to function g. It remains to show that conditions (4) and (5) in the definition of an L-reduction are met.

By Lemma 26 we know that $\mu_2(H, \lambda) = \alpha(G) + \frac{3n}{2} = \alpha(G) + \frac{6n}{4} \leq 7\alpha(G)$, where the latter inequality follows from the fact that the independence number of an n-vertex cubic graph is at least $\frac{n}{4}$. Hence, condition (4) holds with parameter $\beta = 7$.

Let now M be a 2-temporal matching of (H, λ), and let S be an independent set in G guaranteed by Lemma 25, then

$$\alpha(G) - |S| = \mu_2(H, \lambda) - \frac{3n}{2} - |S| \leq \mu_2(H, \lambda) - \frac{3n}{2} - |M| + \frac{3n}{2} = \mu_2(H, \lambda) - |M|,$$

where the first equality follows from Lemma 26 and the inequality follows from Lemma 25. Thus, condition (5) holds with parameter $\gamma = 1$. ◀

B.2 Approximation Lower Bound for Maximum Temporal Matching

We show that our reduction together with a polynomial-time δ-approximation algorithm $\mathcal{A}$ for Maximum Temporal Matching, where $\delta \geq \frac{664}{665}$, imply a polynomial-time $\frac{94}{95}$-approximation algorithm for Maximum Independent Set in cubic graphs. The result will then follow from the fact that it is NP-hard to approximate Maximum Independent Set in cubic graphs to within factor of $\frac{94}{95}$ [18].

Let G be a cubic graph and (H, λ) be the corresponding temporal graph from the reduction. Let also M be a 2-temporal matching found by algorithm $\mathcal{A}$, and let S be the independent set in G corresponding to M. Since $\mathcal{A}$ is a δ-approximation algorithm, we have $\frac{|M|}{\mu_2(H,\lambda)} \geq \delta$. Furthermore, by Lemma 27, our reduction is an L-reduction with parameters $\beta = 7$ and $\gamma = 1$, that is, $\mu_2(H, \lambda) \leq 7\alpha(G)$ and $\alpha(G) - |S| \leq \mu_2(H, \lambda) - |M|$. Hence, we have

$$\alpha(G) - |S| \leq \mu_2(H, \lambda) - |M| \leq \mu_2(H, \lambda) \cdot (1 - \delta) \leq 7\alpha(G) \cdot (1 - \delta),$$

which together with $\delta \geq \frac{664}{665}$ imply $\frac{|S|}{\alpha(G)} \geq 7\delta - 6 \geq \frac{94}{95}$, as required.

B.3 ETH Lower Bound for Maximum Temporal Matching

The Exponential Time Hypothesis (ETH) implies (together with the Sparsification Lemma) that there is no $2^{o(\#\text{variables} + \#\text{clauses})}$-time algorithm for 3SAT [41, 42]. When investigating the original reduction from 3SAT to Vertex Cover on cubic graphs [33], it is easy to verify that the size of the constructed instance is linear in the size of the 3SAT formula. Hence, it follows that there is no $2^{o(|V|)} \cdot \text{poly}(|V|)$-time algorithm for Vertex Cover on cubic graphs unless the ETH fails. It follows that there is no $2^{o(|V|)} \cdot \text{poly}(|V|)$-time algorithm for Independent Set on cubic graphs unless the ETH fails. If we treat the reduction presented in Construction 1 as a polynomial-time many-one reduction, then we set the solution size for the Temporal Matching instance to the solution size of the Independent Set instance plus 3/2 times the number of vertices in the Independent Set instance (see Lemma 25 and Lemma 26). It follows that the existence of a $2^{o(k)} \cdot |\mathcal{G}|^{f(T)}$-time algorithm (for any function f) for Temporal Matching implies a $2^{o(|V|)} \cdot \text{poly}(|V|)$-time algorithm for Independent Set on cubic graphs (note that T is constant in the reduction), which contradicts the ETH.

B.4 Proof of Observation 6

Proof Sketch. We observe that Construction 1 can be modified in such a way that it produces a temporal graph that has a complete underlying graph. Namely, we can add two additional snapshots to the construction, one edgeless snapshot at time slot four, and one snapshot that is a complete graph at time slot five. This has the consequence that the size of the matching increases by exactly $\lfloor n/2 \rfloor$ and the underlying graph of the constructed temporal graph is a complete graph. Hence, we obtain Observation 6. ◀

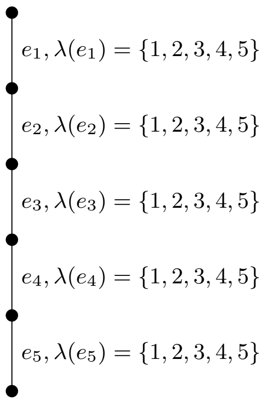
B.5 Adapting Construction 1 to the Model of Baste et al. [9]

We remark that our reduction for Theorem 5 can easily be adapted to the model of Baste et al. [9]: recall that every edge of the underlying graph of the temporal graph constructed in the reduction (see Construction 1) appears in exactly one time step. Hence, for each of these time-edges, we can add a second appearance exactly one time step after the first appearance without creating any new matchable edges. Of course in order to do that for time-edges appearing in the third time step, we need another fourth time step. It follows that γ-Matching [9] is NP-hard and its canonical optimization version is APX-hard even if $\gamma = 2$ and $T = 4$, which improves the hardness result by Baste et al. [9].

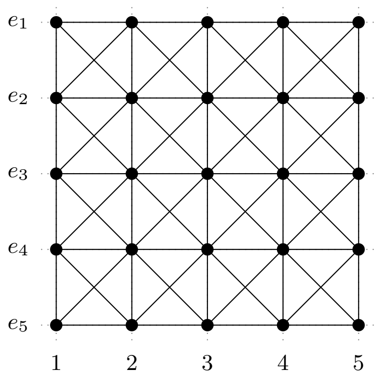
B.6 Proof of Theorem 11

We prove Theorem 11 in several steps. We first use that a cubic planar graph admits a planar embedding in such a way that the vertices are mapped to points of a grid and the edges are drawn along the grid lines. Moreover, such an embedding can be computed in polynomial time and the size of the grid is polynomially bounded in the size of the planar graph. Furthermore, if we scale the embedding by a factor of two, i.e. subdivide every edge once, then the embedding is also guaranteed to be an *induced* subgraph of the grid. In other words, we argue that every cubic planar graph is an induced topological minor of a polynomially large grid graph.

▶ **Proposition 28** (Special case of Theorem 2 from Valiant [59]). *Let $G = (V, E)$ be a cubic planar graph. Then G is an induced topological minor of $Z_{n,m}$ for some n, m with $n \cdot m \in O(|V|^2)$ and the corresponding subdivision of G can be computed in polynomial time.*



(a) Temporal graph $\mathcal{G} = (P_6, \lambda)$ with $\lambda(e) = [5]$ for all $e \in E(P_6)$.



(b) 2-Temporal line graph $L_2(\mathcal{G})$. The horizontal dimension corresponds to time slots 1 to 5, the vertical dimension corresponds to the edges of P_6.

Figure 5 A temporal line graph with a path as underlying graph where edges are always active and its 2-temporal line graph.

We discuss next how to replace the edges of a cubic planar graph by paths of appropriate length such that it is an induced subgraph of a diagonal grid graph. In other words, we show that every cubic planar graph is an induced topological minor of a polynomially large diagonal grid graph.

▶ **Lemma 29.** *Let $G = (V, E)$ be a cubic planar graph. Then G is an induced topological minor of $\widehat{Z}_{n,m}$ for some n, m with $n \cdot m \in O(|V|^2)$ and the corresponding subdivision of G can be computed in polynomial time.*

Proof. Let $G = (V, E)$ be a cubic planar graph. By Proposition 28 we know that there are integers n, m with $n \cdot m \in O(|V|^2)$ such that $G = (V, E)$ is an induced topological minor of $Z_{n,m}$. Let $G' = (V', E')$ with $V' \subseteq \mathbb{N} \times \mathbb{N}$ be the corresponding subdivision of G that is an induced subgraph of $Z_{n,m}$, i.e. $Z_{n,m}[V'] = G'$. Furthermore, for each vertex $v \in V$ of G, let $v' \in V'$ denote the corresponding vertex in the subdivision G'.

Let $G'' = (V'', E'')$ be the graph resulting from subdividing each edge in G' eleven additional times and shift the graph three units away from the boundary of $Z_{n,m}$ in both dimensions. Intuitively, this is necessary to ensure that all paths in the grid are sufficiently far away from each other, which is also important in a later modification.

More formally, for each vertex $(i, j) \in V'$ create a vertex $(12i + 3, 12j + 3) \in V''$. For each edge $\{(i, j), (i, j + 1)\} \in E'$ create eleven additional vertices, one for each grid point on the line between $(12i + 3, 12j + 3)$ and $(12i + 3, 12j + 15)$. We connect these vertices by edges such that we get an induced path on the new vertices together with $(12i + 3, 12j + 3)$ and $(12i + 3, 12j + 15)$ that follows the grid line they lie on. For each edge $\{(i, j), (i + 1, j)\} \in E'$ we make an analogous modification to G''. Furthermore, for each vertex $v \in V$ of G, let $v'' \in V''$ denote the corresponding vertex in the subdivision G''. It is clear that G'' is an induced subgraph of $Z_{12n+6,12m+6}$. We now show how to further modify G'' such that it is an induced subgraph of the diagonal grid graph $\widehat{Z}_{12n+6,12m+6}$.

For each vertex $v \in V$ let $v'' = (i, j) \in V''$, we check the following.

1. If $\deg_{G''}((i, j)) = 2$ and $\{(i, j), (i, j + 1)\}, \{(i, j), (i + 1, j)\}, \{(i, j), (i + 2, j)\} \in E''$, then we delete $(i + 1, j)$ from V'' and all its incident edges from E''. We add vertex $(i + 1, j - 1)$

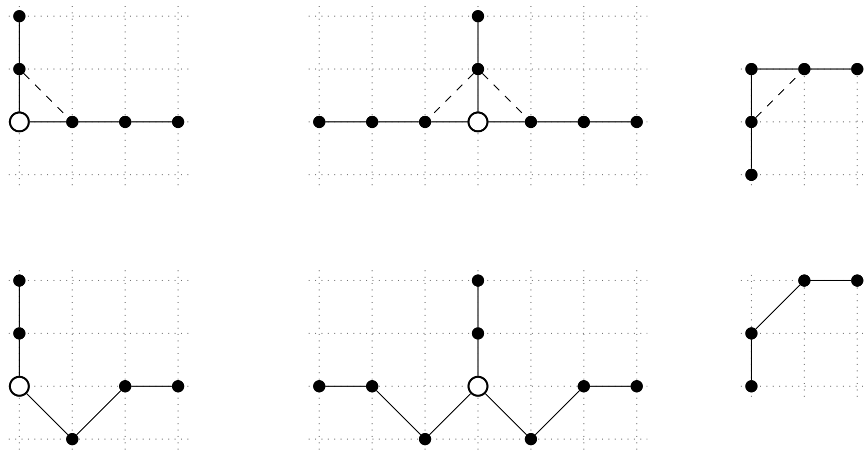


Figure 6 Illustration of the modifications described in the proof of Lemma 29. The situation before the moficication is depiced above, dashed edges show unwanted edges present in an induced subgraph of a diagonal grid graph. The situation after the modification is depicted below.

to V'' and add edges $\{(i,j),(i+1,j-1)\}$ and $\{(i+1,j-1),(i+2,j)\}$ to E''. This modification is illustrated in Figure 6a. Rotated versions of this configuration are modified analogously.

2. If $\deg_{G''}((i,j)) = 3$ and $\{(i,j),(i,j+1)\},\{(i,j),(i+1,j)\},\{(i,j),(i+2,j)\},\{(i,j),(i-1,j)\},\{(i,j),(i-2,j)\} \in E''$, then we delete $(i+1,j)$ from V'' and all its incident edges from E''. We add vertex $(i+1,j-1)$ to V'' and add edges $\{(i,j),(i+1,j-1)\}$ and $\{(i+1,j-1),(i+2,j)\}$ to E''. Furthermore, we we delete $(i-1,j)$ from V'' and all its incident edges from E''. We add vertex $(i-1,j-1)$ to V'' and add edges $\{(i,j),(i-1,j-1)\}$ and $\{(i-1,j-1),(i-2,j)\}$ to E''. This modification is illustrated in Figure 6b. Rotated versions of this configuration are modified analogously.

Lastly, whenever a path in G'' that corresponds to an edge in G bends at a square angle, we remove the corner vertex and its incident edges and reconnect the path by a diagonal edge.

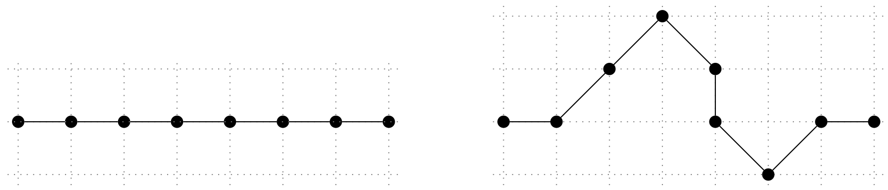
More formally, let $(i,j-1),(i,j),(i+1,j) \in V''$ be adjacent vertices in a path in G'' that corresponds to an edge in G, then we remove (i,j) from V'' and all its incident edges and add the edge $\{(i,j-1),(i+1,j)\}$ to E''. This modification is illustrated in Figure 6c. Rotated versions of this configuration are modified analogously.

Now it is easy to see that G'' is an induced subgraph of $\widehat{Z}_{12n+6,12m+6}$. Furthermore, G'' can be computed in polynomial time. ◀

Next we argue that we can always embed a cubic planar graph into a diagonal grid graph in a way that preserves NP-hardness. This is based on the observation that subdividing an edge of a graph twice increases the size of a maximum independent set exactly by one.

▶ **Observation 30** (Poljak [55]). *Let $G = (V,E)$ be a graph. Then for every $\{u,v\} \in E$, the graph $G' = (V \cup \{u',v'\}, (E \setminus \{\{u,v\}\}) \cup \{\{u,u'\},\{u',v'\},\{v',v\}\})$ contains an independent set of size $k+1$ if and only if G contains an independent set of size k.*

From this observation follows that if we can guarantee that for every cubic planar graph there is a subdivision that subdivides every edge an even number of times and that is an induced subgraph of a diagonal grid graph of polynomial size, then we are done.



(a) Before. **(b)** After.

Figure 7 Illustration of the modification described in the proof of Lemma 31. It shows how to increase the length of an induced path of a diagonal grid graph by one.

▶ **Lemma 31.** *Let $G = (V, E)$ be a cubic planar graph. Then there is a subdivision of G that is an induced subgraph of $\widehat{Z}_{n,m}$ for some n, m with $n \cdot m \in O(|V|^2)$ and where each edge of G is subdivided an even number of times. Furthermore, the subdivision of G can be computed in polynomial time.*

Proof. Let $G = (V, E)$ be a cubic planar graph. By Lemma 29 we know that there are some n, m with $n \cdot m \in O(|V|^2)$ such that $G = (V, E)$ is an induced topological minor of $\widehat{Z}_{n,m}$. Let $G' = (V', E')$ with $V' \subseteq \mathbb{N} \times \mathbb{N}$ be a subdivision of G constructed as described in the proof of Lemma 29.

Recall that every edge e in G is replaced by a path P_e in G'. From Observation 30 follows that if we can guarantee that all these paths have an odd number of edges (and hence result from an even number of subdivisions), then G' contains an independent set of size $k + \sum_{e \in E} \lfloor \frac{|E(P_e)|-1}{2} \rfloor$ if and only if G contains an independent of size k. In the following we show how to change the parity of the number of edges of a path P_e in G' that corresponds to an edge e in G.

The number of subdivisions performed in the construction in the proof of Lemma 29 ensures that each path P_e in G' that corresponds to an edge e in G contains seven consecutive edges that are either all horizontal or all vertical. Assume that P_e contains an even number of edges and contains horizontal edges $\{(i,j),(i+1,j)\}, \{(i+1,j),(i+2,j)\}, \{(i+2,j),(i+3,j)\}, \{(i+3,j),(i+4,j)\}, \{(i+4,j),(i+5,j)\}, \{(i+5,j),(i+6,j)\}, \{(i+6,j),(i+7,j)\}$. We remove vertices $(i+2,j), (i+3,j), (i+5,j)$ and all their incident edges. We add vertices $(i+2,j+1), (i+3,j+2), (i+4,j+1), (i+5,j-1)$ and edges $\{(i+1,j),(i+2,j+1)\}, \{(i+2,j+1),(i+3,j+2)\}, \{(i+3,j+2),(i+4,j+1)\}, \{(i+4,j+1),(i+4,j)\}, \{(i+4,j),(i+5,j-1)\}, \{(i+5,j-1),(i+6,j)\}$. It is easy to check that this reconnects the path and increases the number of edges by one. This modification is illustrated in Figure 7. The vertical version of this configuration is modified analogously.

Using this modification we can easily modify G' in polynomial time in a way that all paths that correspond to edges of G have an odd number of edges. ◀

This concludes the proof of Theorem 11, which now follows directly from Lemma 31 and Observation 30.

C Additional Material for Section 4

C.1 Correctness of Algorithm 4.1

The notions of Δ-window, partial Δ-window, and Δ-template are illustrated in Figure 8. A time slot t is *covered* by a Δ-template $\mathcal{S}$ if t belongs to an interval of $\mathcal{S}$. We show the

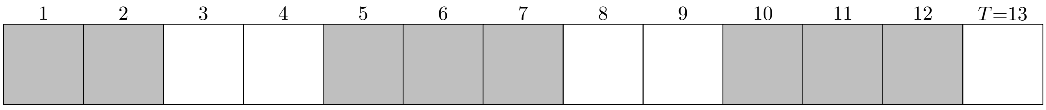


Figure 8 The gray slots form the intervals of a Δ-template, where $\Delta = 3$. Interval $[1, 2]$ is a partial Δ-window. Intervals $[5, 7]$ and $[10, 12]$ are Δ-windows.

following properties of Δ-templates which we need to prove the approximation ratio of our algorithm.

▶ **Lemma 32.** *Let Δ and T be natural numbers such that $\Delta \leq T$. Then*

(1) there are exactly $2\Delta - 1$ different Δ-templates with respect to lifetime T;
(2) every time slot in $[T]$ is covered by exactly Δ different Δ-templates.

Proof. To prove (1), we first observe that a Δ-template $\mathcal{S}$ is uniquely determined by its leftmost interval. Indeed, by fixing the leftmost interval of $\mathcal{S}$, by definition, the subsequent intervals of $\mathcal{S}$ are located in $[T]$ uniformly at distance exactly $\Delta - 1$ from each other. Now, the maximality of $\mathcal{S}$ implies that the first interval in $\mathcal{S}$ is either a partial Δ-window that starts at time slot 1 or a (possibly partial) Δ-window that starts in one of the first Δ time slots of $[T]$. Since there are $\Delta - 1$ intervals of the first type and Δ intervals of the second type, we conclude that there are exactly $2\Delta - 1$ different Δ-templates with respect to lifetime T.

To prove (2), we note that all Δ-templates can be successively obtained from the Δ-template $\mathcal{S}$ whose first interval is the single-slot partial Δ-window $[1]$ by shifting by one time slot to the right all the intervals of the current Δ-template (in each shift we augment the leftmost interval if it was a partial Δ-window and truncate the rightmost interval if it covered the last time slot T). It is easy to see that every time slot will be covered in exactly Δ of $2\Delta - 1$ shifting iterations. ◀

Next, we formally define the matchings that our algorithm computes. Let $\mathcal{S}$ be a Δ-template. A Δ-temporal matching $M^{\mathcal{S}}$ in $\mathcal{G} = (G, \lambda)$ is called a Δ-temporal matching *with respect to Δ-template* $\mathcal{S}$ if $M^{\mathcal{S}}$ has the maximum possible number of edges in every interval $W \in \mathcal{S}$, i.e. $\left|M^{\mathcal{S}}|_W\right| = \mu_\Delta(\mathcal{G}|_W)$ for every $W \in \mathcal{S}$. By definition, for any two distinct intervals W_1, W_2 in $\mathcal{S}$ and for any two time slots $t_1 \in W_1$ and $t_2 \in W_2$ we have $|t_1 - t_2| > \Delta$, which implies that no two time-edges of $\mathcal{G}$ that appear in time slots of different intervals of $\mathcal{S}$ are in conflict. This observation together with the fact that every interval in $\mathcal{S}$ is of length at most Δ imply that a Δ-temporal matching with respect to $\mathcal{S}$ can be computed in polynomial time by computing a maximum Δ-temporal matching in $\mathcal{G}|_W$ for every $W \in \mathcal{S}$ and then taking the union of these matchings[4]. Since every Δ-template has $O\left(\frac{T}{\Delta}\right)$ intervals and, a maximum Δ-temporal matchings in $\mathcal{G}|_W$, $W \in \mathcal{S}$ can be computed in $O(m(\sqrt{n} + \Delta))$ time, which follows from Observation 22, we conclude that a Δ-temporal matching with respect to $\mathcal{S}$ can be computed in $O\left(Tm\left(\frac{\sqrt{n}}{\Delta} + 1\right)\right)$ time.

▶ **Lemma 33.** *Algorithm 4.1 is an $O\left(Tm(\sqrt{n} + \Delta)\right)$-time $\frac{\Delta}{2\Delta-1}$-approximation algorithm for* Maximum Temporal Matching.

Proof. Let $\mathcal{G} = (G, \lambda)$ be an arbitrary temporal graph of lifetime T and Δ be a natural number such that $\Delta \leq T$. Let also M^* be a maximum Δ-temporal matching of $\mathcal{G}$.

[4] The obtained Δ-temporal matching can further be extended greedily to a maximal Δ-temporal matching.

We show that, given the instance $(\mathcal{G}, \Delta)$, Algorithm 4.1 produces in time $O\left(Tm(\sqrt{n} + \Delta)\right)$ a Δ-temporal matching M of size at least $\frac{\Delta}{2\Delta-1}|M^*|$, where n and m are the number of vertices and the number of edges in the underlying graph G, respectively.

Clearly, the algorithm outputs a feasible solution as M is a Δ-temporal matching with respect to some Δ-template. We show next that M is the desired approximate solution. As in the pseudocode of Algorithm 4.1, for a Δ-template $\mathcal{S}$ we denote by $M^{\mathcal{S}}$ the Δ-temporal matching with respect to $\mathcal{S}$ computed in Line 3 of Algorithm 4.1. Let $\mathfrak{S}$ be the family of all Δ-templates with respect to lifetime T, and let $\mathcal{S}' \in \mathfrak{S}$ be a Δ-template such that $M = M^{\mathcal{S}'}$. It follows from the algorithm that $|M^{\mathcal{S}'}| \geq |M^{S}|$ for every $S \in \mathfrak{S}$. By definition, for every $\mathcal{S} \in \mathfrak{S}$ and for every interval $W \in \mathcal{S}$ we have $\sum_{t \in W} |M_t^{\mathcal{S}}| \geq \sum_{t \in W} |M_t^*|$, where $M_t = M \cap E_t$. Hence

$$|M^{\mathcal{S}}| \geq \sum_{W \in \mathcal{S}} \sum_{t \in W} |M_t^{\mathcal{S}}| \geq \sum_{W \in \mathcal{S}} \sum_{t \in W} |M_t^*|.$$

Using the above inequalities and Lemma 32 we derive

$$\begin{aligned}(2\Delta - 1)|M^{\mathcal{S}'}| &\geq \sum_{\mathcal{S} \in \mathfrak{S}} |M^{\mathcal{S}}| \\ &\geq \sum_{\mathcal{S} \in \mathfrak{S}} \sum_{W \in \mathcal{S}} \sum_{t \in W} |M_t^{\mathcal{S}}| \geq \sum_{\mathcal{S} \in \mathfrak{S}} \sum_{W \in \mathcal{S}} \sum_{t \in W} |M_t^*| = \Delta \sum_{t=1}^{T} |M_t^*| = \Delta |M^*|,\end{aligned}$$

which implies the $|M| = |M^{\mathcal{S}'}| \geq \frac{\Delta}{2\Delta-1}|M^*|$.

Now we analyze the time complexity of the algorithm. By Lemma 32 there are exactly $2\Delta - 1$ different Δ-templates, and therefore the for-loop in Line 2 of Algorithm 4.1 performs exactly $2\Delta - 1$ iterations. At every iteration the algorithm computes a Δ-temporal matching with respect to a Δ-template, which, as we discussed, can be done in $O\left(Tm\left(\frac{\sqrt{n}}{\Delta} + 1\right)\right)$ time. Altogether, the total time complexity is $O\left(Tm(\sqrt{n} + \Delta)\right)$, as claimed. ◀

C.2 Tightness of the analysis of Algorithm 4.1 for $\Delta = 2$

We remark that our analysis ignores the fact that the algorithm may add time-edges from the gaps between the Δ-windows defined by the template to the matching if they are not in conflict with any other edge in the matching. Hence, there is potential room for improvement. On the other hand, our analysis of the approximation factor of Algorithm 4.1 is tight for $\Delta = 2$. Namely, there exists a temporal graph $\mathcal{G}$ (see Figure 2) such that on the instance $(\mathcal{G}, 2)$ our algorithm (in the worst case) finds a 2-temporal matching of size 2, while the size of a maximum 2-temporal matching in $\mathcal{G}$ is 3. In this example any improvement of the algorithm that utilizes the gaps between the Δ-windows would not lead to a better performance. More specifically, temporal graph $\mathcal{G}$ has lifetime 3, the underlying graph of $\mathcal{G}$ is a 5-vertex paths $P = (v_1, v_2, v_3, v_4, v_5)$, and the first snapshot consists of the two internal edges of P, the second snapshot consists of the two pendant edges of P, and the third snapshot consists of the internal edge $\{v_2, v_3\}$. There are three 2-templates with respect to lifetime 3, which are $\{[1, 2]\}$, $\{[1, 1], [3, 3]\}$, and $\{[2, 3]\}$. Possible 2-temporal matchings with respect to these 2-templates that the algorithm could compute are $\{(v_3, v_4, 1), (v_1, v_2, 2)\}$, $\{(v_3, v_4, 1), (v_2, v_3, 3)\}$, and $\{(v_1, v_2, 2), (v_4, v_5, 2)\}$, respectively. In this scenario the algorithm would output a 2-temporal matching of size 2, while $\{(v_2, v_3, 1), (v_4, v_5, 2), (v_2, v_3, 3)\}$ is a 2-temporal matching of size 3. Furthermore, it is easy to verify that each of these 2-temporal matchings is a maximal 2-temporal matching in the whole temporal graph $\mathcal{G}$, and therefore none of them could be extended with time-edges from the gaps.

Algorithm C.1: Fixed-Parameter Algorithm for the Solution Size k (Theorem 13).

Input: A temporal graph $\mathcal{G} = (G, \lambda)$ of lifetime T and $\Delta, k \in \mathbb{N}$.
Output: *yes* if there is a Δ-temporal matching of size k, otherwise *no*.

```
1  if k = 0 or maximum matching size of G is at least k then return yes.
2  if 𝒢 has no edge appearances then return no.
3  Let t_0 be the time slot of the first non-empty snapshot of 𝒢.
4  λ(e) ← {t − t_0 + 1 | t ∈ λ(e)}, for all e ∈ E(G).
5  K ← kernel for the first Δ-window of 𝒢 computed by Algorithm 4.2.
6  foreach non-empty Δ-temporal matching S in K do
7      A ← {(e,t) | (e,t) ∈ Λ(𝒢) is not Δ-independent with some (e',t') ∈ S}.
8      𝒢' ← 𝒢|_[Δ+1,T] \ A.
9      return call Algorithm C.1 for 𝒢', Δ, and k ← max{k − |S|, 0}.
```

C.3 Proof of Lemma 14

Proof. The lemma follows from the observation that since $t_2 \leq \Delta$, no time-edge (e, t), $t < t_2$, is in conflict with any time-edge in $M \setminus \{(e, t_2)\}$. ◀

C.4 Proof of Lemma 18

Proof. The underlying graph G' of the first Δ-window in Line 1 of Algorithm 4.2 can be computed in $O(\Delta m)$ time. Using the standard augmenting path-based procedure and the linear-time algorithm for finding an augmenting path [29], a maximum matching A of G' in Line 2 can be computed in $O(\nu(n + m))$ time. Since $|V_A| \leq 2\nu$, the for-loop in Line 4 performs at most 2ν iterations. At each of these iterations the corresponding set R_v can be computed in $O(n)$ time, because it contains at most $n - 1$ time-edges, and the list of time labels of every edge is ordered by time. Finally, observe that $R' \subseteq R_v$ can be computed in $O(\nu \cdot n)$ time and that at each iteration we add at most $4\nu + 1$ time-edges to K. Thus, overall Algorithm 4.2 runs in $O(\nu^2(n + m\Delta))$ time. ◀

C.5 Proof of Theorem 13

The pseudocode for the algorithm behind Theorem 13 is stated in Algorithm C.1. We show its correctness in Lemma 34 and the claimed running time in Lemma 35.

▶ **Lemma 34.** *Algorithm C.1 is correct.*

Proof. First, observe that an instance with $k = 0$ is a trivial *yes*-instance and an instance with $k > 0$ and no edge appearances is a trivial *no*-instance. Second, if there is a matching M of size at least k in the underlying graph G, then $\{(e, t) \mid e \in M, t = \min \lambda(e)\}$ is a Δ-temporal matching in $\mathcal{G}$ of size $|M|$. Hence, Lines 1–2 are correct. In Lines 3–4, we remove the leading edgeless snapshots from the temporal graph if any. Note that this does not change the size of any Δ-temporal matching. However, after this preprocessing every Δ-temporal matching M of maximum size in $\mathcal{G}$ contains at least one time-edge from the first Δ-window, because otherwise M could be extended by a time-edge from the first snapshot. In Line 5, a kernel K for the first Δ-window of $\mathcal{G}$ is computed by Algorithm 4.2. Hence, there is a maximum Δ-temporal matchings M in $\mathcal{G}$ such that $M|_{[1,\Delta]} \subseteq K$. Thus, at the iterations of the for-loop in Line 6 that corresponds to $S = M|_{[1,\Delta]}$ the algorithm constructs in Line 8 the temporal graph $\mathcal{G}'$ obtained from $\mathcal{G}$ by removing the first Δ-window and all time-edges which are not Δ-independent with all the time-edges in S. Hence, for any Δ-temporal matching X in $\mathcal{G}'$ the set $M|_{[1,\Delta]} \cup X$ is a Δ-temporal matching in $\mathcal{G}$ of size $\left|M|_{[1,\Delta]}\right| + |X|$. Moreover, no time-edge in $M|_{[\Delta+1,T]}$ is removed in Line 8. Thus, there is a Δ-temporal matching of size at least k in $\mathcal{G}$ if and only if there is a Δ-temporal matching of size at least $k - |S|$ in $\mathcal{G}'$. This implies correctness of Line 9.

Algorithm C.1 terminates, because we decrease the parameter k in each recursion until zero is reached. ◀

It remains to show that Algorithm C.1 is indeed a linear-time fixed-parameter algorithm when parameterized by the solution size k.

▶ **Lemma 35.** *Algorithm C.1 runs in $2^{O(k^3)} \cdot |\mathcal{G}|$ time.*

Proof. In Line 1 of Algorithm C.1, we use the standard augmenting path-based algorithm for maximum matching to check if G has a matching of size k. Since an augmenting path can be found in linear time [29], this step can be executed in $O(k(n+m))$ time. If G has a matching of size k, then the algorithm terminates in Line 1 and the lemma holds. Hence, we assume that the maximum matching size ν of G is strictly smaller than k. To compute Line 4, we first determine in linear time the time slot t_0 of the first non-empty snapshot and then iterate a second time over the temporal graph to set the new labels. By Lemma 16, Line 5 can be computed in $O(\nu^2 \cdot |\mathcal{G}|)$ time. Thus, Lines 1–5 are computable in $O(k^2 \cdot |\mathcal{G}|)$ time.

By Lemma 16, the kernel K for the first Δ-window contains at most $O(k^2)$ time-edges. Hence, the for-loop in Line 6 runs at most $2^{O(k^2)}$ iterations. To compute the temporal graph $\mathcal{G}'$ of Line 8 in $O(|\mathcal{G}|)$ time, we first iterate once over the temporal graph to remove the first Δ-window. Next, we iterate over the time-edges in S and store for each vertex how long it is Δ-blocked by any time-edge S. Finally, we iterate a second time over the temporal graph and remove a time-edge (e,t) if one of its endpoints is Δ-blocked at time slot t.

In total, Lines 1-8 of a single call of Algorithm C.1 run in $2^{O(k^2)} \cdot |\mathcal{G}|$ time. In Line 9 the algorithm calls itself recursively. However, since the parameter k is decreased at every recursive call, the depth of the recursion tree is at most k, which implies that the size of the tree is $2^{O(k^3)}$. Hence Algorithm C.1 runs in $2^{O(k^3)} \cdot |\mathcal{G}|$ time. ◀

C.6 Tools from matroid theory

We use standard terminology from matroid theory [54]. A pair (U, I), where U is the *ground set* and $I \subseteq 2^U$ is a family of *independent sets*, is a *matroid* if the following holds:

- $\emptyset \in I$;
- if $A' \subseteq A$ and $A \in I$, then $A' \in I$;
- if $A, B \in I$ and $|A| < |B|$, then there is an $x \in B \setminus A$ such that $A \cup \{x\} \in I$.

An inclusion-wise maximal independent set $A \in I$ of a matroid $\mathcal{Q} = (U, I)$ is a *basis*. The cardinality of the bases of $\mathcal{Q}$ is called the *rank* of $\mathcal{Q}$. The *uniform matroid of rank* r on U is the matroid (U, I) with $I = \{S \subseteq U \mid |S| \leq r\}$. A matroid (U, I) is *linear* or *representable over a field* $\mathbb{F}$ if there is a matrix A with entries in $\mathbb{F}$ and the columns labeled by the elements of U such that $S \in I$ if and only if the columns of A with labels in S are linearly independent over $\mathbb{F}$. The matrix A is called a *representation* of (U, I).

▶ **Definition 36** (Max q-Representative Family). *Given a matroid (U, I), a family $\mathcal{S} \subseteq I$ of independent sets, and a function $w \colon \mathcal{S} \to \mathbb{R}$, we say that a subfamily $\widehat{\mathcal{S}} \subseteq \mathcal{S}$ is a* max q-representative for $\mathcal{S}$ with respect to w *if for each set $Y \subseteq U$ of size at most q it holds that if there is a set $X \in \mathcal{S}$ with $X \uplus Y \in I$, then there is a set $\widehat{X} \in \widehat{\mathcal{S}}$ such that $\widehat{X} \uplus Y \in I$ and $w(\widehat{X}) \geq w(X)$.*

For linear matroids, there are fixed-parameter algorithms parametrized by rank that compute representatives for large families of independent sets with respect to additive set functions [60]. A function $w : 2^U \to \mathbb{R}$ on the subsets of a set U is *additive set function* if $w(A \uplus B) = w(A) + w(B)$ for all disjoint sets $A, B \subseteq U$.

▶ **Theorem 37** (van Bevern et al. [60, Proposition 4.8]). *Let α, β, and γ be non-negative integers such that $r = (\alpha + \beta)\gamma \geq 1$. Let $\mathcal{Q} = (U, I)$ be a linear matroid of rank r and $w \colon 2^U \to \mathbb{N}$ be an additive set function. Furthermore, let $\mathcal{H} \subseteq 2^U$ be a γ-family of size t and let*

$$\mathcal{S} = \{S = H_1 \uplus \cdots \uplus H_\alpha \mid S \in I \text{ and } H_j \in \mathcal{H} \text{ for } j \in \{1, \ldots, \alpha\}\}.$$

Then, given a representation of $\mathcal{Q}$ over a finite field $\mathbb{F}$, one can compute a max $\beta\gamma$-representative $\widehat{\mathcal{S}}$ of size $\binom{r}{\alpha\gamma}$ for the family $\mathcal{S}$ with respect to w using $2^{O(r)} \cdot t$ operations over $\mathbb{F}$ and calls to the function w.

Theorem 37 is based on results of Fomin et al. [27] and Marx [49]. We use Theorem 37 only for uniform matroids. For this reason we expect that one can improve the base of the exponential function in $\nu\Delta$ of the running time in Theorem 19 by replacing Theorem 1.1 of Fomin et al. [27] for linear matroids with its special case Theorem 1.2 for uniform matroids and tighten the running time analysis in Theorem 37.

Furthermore, van Bevern et al. [60] proved Theorem 37 for multiple matroids and for more general weight functions than additive set functions. However, for our purpose the stated version suffices. The crucial point of Theorem 37 is that for a linear matroid of rank $(\alpha+\beta)\gamma$ and a γ-family $\mathcal{H}$, we can compute a small (of size $\binom{r}{\alpha\gamma}$) max $\beta\gamma$-representative $\widehat{\mathcal{S}}$ for a potentially very large (unbounded in the rank of the matroid) family $\mathcal{S}$ of all independent sets of size $\alpha\gamma$ which are disjoint unions of sets from $\mathcal{H}$. An important property of $\widehat{\mathcal{S}}$ is that for any independent set Y of size $\beta\gamma$ such that there is a set $X \in \mathcal{S}$ which is disjoint from Y and the union of X and Y is an independent set, $\widehat{\mathcal{S}}$ contains a set $\widehat{X}$ which is also disjoint from Y, the union of $\widehat{X}$ and Y is also an independent set, and the weight of $\widehat{X}$ is at least as large as the weight of X.

C.7 Proof of Lemma 20

Before we show Lemma 20, we prove several intermediate lemmata. These lemmata are used in the proof of Lemma 20 which is deferred to the end of this paragraph. The primary tool in the proof of Lemma 20 is Theorem 37 applied to a properly chosen matroid $\mathcal{Q}$, a family $\mathcal{H}$, and a weight function w. The idea is that a disjoint union of sets from $\mathcal{H}$ corresponds to a Δ-temporal matching in $\mathcal{G}|_{[\Delta(\ell-1)+1, \Delta\ell]}$ and the weight function tells us how large the Δ-temporal matching is.

▶ **Construction 2** (Matroid, Family, and Weight Function). We define

1. the $(5\nu + 5\nu(\Delta - 1))$-uniform *matroid* $\mathcal{Q}$ on the ground set $U := V \cup E' \cup V' \cup D$, where
   - $E' = \{e_t \mid e \in E_t \text{ and } t \in [\Delta(\ell-1)+1, \Delta\ell]\}$,

Algorithm C.2: Construction of an ℓ-Complete Family (Lemma 20).

Input: A temporal graph $\mathcal{G} = (G, \lambda)$ of lifetime T and $\ell, \Delta \in \mathbb{N}$ such that $\ell\Delta \leq T$.
Output: An ℓ-complete family of Δ-temporal matchings for $\mathcal{G}|_{[\Delta(\ell-1)+1,\Delta\ell]}$ of size $2^{O(\nu\Delta)}$.

```
ν ← the maximum matching size of G.
For input 𝒢, Δ, and ν, compute a representation of the matroid 𝒬 = (U, I) over a
  finite field 𝔽_p with p ∈ O(|𝒢|), and the family ℋ according to Construction 2.
ℱ̂ ← max (5ν(Δ − 1))-representative family of
      ℱ = {F = H_1 ⊎ ··· ⊎ H_ν | F ∈ I and H_j ∈ ℋ for j ∈ [ν]} with respect to w.
ℳ ← ∅.
foreach F ∈ ℱ̂ do
    M = {(e, t) | e_t ∈ F}.
    ℳ ← ℳ ∪ {M}.
return ℳ.
```

- $V' = \{v_t \mid v \in V$ and $t \in [\Delta(\ell-1)+1, \Delta\ell]$ and v is not isolated in $G_t\}$, and
- $D = \{d_i \mid i \in [5\nu]\}$;

2. a 5-*family* $\mathcal{H} := \mathcal{H}_E \cup \mathcal{H}_D$, where
   - $\mathcal{H}_E = \{E^{(t)}_{\{v,w\}} = \{v, w, v_t, w_t, e_t\} \mid e = \{v, w\} \in E_t$ and $t \in [\Delta(\ell-1)+1, \Delta\ell]\}$, and
   - $\mathcal{H}_D = \{D_i = \{d_{5(i-1)+j} \mid j \in [5]\} \mid i \in [\nu]\}$;
3. a *weight function* $w : 2^U \to \mathbb{N}; X \mapsto |X \cap E'|$.

Observe that each set in $\mathcal{H}_E$ corresponds to a time-edge of the temporal graph. Furthermore, D is the set of *dummy* elements and $\mathcal{H}_D$ is a family of sets of dummy elements, which we introduce for technical reasons in order to able to apply Theorem 37 and they can be ignored for the moment.

An important property of Construction 2 that we will employ in the proof of Lemma 20 is formalized in the following simple observation.

▶ **Observation 38.** *Let M be a set of time-edges in $\mathcal{G}|_{[\Delta(\ell-1)+1,\Delta\ell]}$. Then M is a Δ-temporal matching in $\mathcal{G}|_{[\Delta(\ell-1)+1,\Delta\ell]}$ if and only if the sets $E^{(t)}_e$, $(e,t) \in M$ are pairwise disjoint.*

Before we proceed to the proof of Lemma 20, we show that both a representation of the matroid $\mathcal{Q}$ and the family $\mathcal{H}$ can be computed efficiently.

▶ **Lemma 39** ($\star$). *A representation of the matroid $\mathcal{Q}$ over a finite field $\mathbb{F}_p$ with $p \in O(|\mathcal{G}|)$ and the family $\mathcal{H}$ can be computed in $O(\nu\Delta|\mathcal{G}|)$ time. Furthermore, one operation over the finite field $\mathbb{F}_p$ can be computed in constant time.*

Now we are ready to prove Lemma 20. The algorithm behind Lemma 20 is stated in Algorithm C.2. Observe that in the following proof we will use the dummy elements, introduced in Construction 2, to fill up the sets such that their size matches the rank of the matroid $\mathcal{Q}$.

Proof of Lemma 20. To prove the lemma we use that Algorithm C.2. We start with the running time analysis of Algorithm C.2.

1. To compute the maximum matching size ν of the underlying graph G, we use the standard augmenting path-based algorithm for maximum matching. Since an augmenting path can be found in linear time [29], the computation of ν in Line 1 takes $O(\nu|\mathcal{G}|)$ time.
2. In Line 2 the algorithm computes a representation of the matroid $\mathcal{Q}$ over a finite filed $\mathbb{F}_p$ with $p \in O(|\mathcal{G}|)$ and the family $\mathcal{H}$ from Construction 2. By Lemma 39, this can be done in $O(\nu\Delta|\mathcal{G}|)$ time.
3. Since the rank of $\mathcal{Q}$ is $5(\nu+\nu(\Delta-1))$ and $|\mathcal{H}| \in O(|\mathcal{G}|)$, by Theorem 37, the computation of a max $(5\nu(\Delta-1))$-representative family $\widehat{\mathcal{F}}$ in Line 3 performs $2^{O(\nu\Delta)} \cdot |\mathcal{G}|$ operations in $\mathbb{F}_p$ and calls to the function w. The algorithm behind Theorem 37 evaluates function w on sets of cardinality at most the rank of $\mathcal{Q}$, and hence a single call to the function w from Construction 2 can be implemented to work in $O(\nu\Delta)$ time. Furthermore, by Lemma 39 a single operation in $\mathbb{F}_p$ takes constant time. Hence, the overall time complexity of Line 3 is $2^{O(\nu\Delta)} \cdot |\mathcal{G}|$.
4. Since the family $\widehat{\mathcal{F}}$ is of size at most $\binom{5\nu+5\nu(\Delta-1)}{5\nu} \in 2^{O(\nu\Delta)}$, the for-loop in Line 5 runs $2^{O(\nu\Delta)}$ iterations. Each of these iterations runs in $O(\nu)$ time, and hence, in total, the for-loop is executed in $2^{O(\nu\Delta)}$ time.

Overall the algorithm outputs a family $\mathcal{M}$ of size $2^{O(\nu\Delta)}$ in time $2^{O(\nu\Delta)} \cdot |\mathcal{G}|$.

We are left to show that $\mathcal{M}$ is an ℓ-complete family of Δ-temporal matchings of $\mathcal{G}|_{[\Delta(\ell-1)+1,\Delta\ell]}$. First, we argue that every set in $\mathcal{M}$ is a Δ-temporal matching of $\mathcal{G}|_{[\Delta(\ell-1)+1,\Delta\ell]}$. Indeed, by construction, a set M in $\mathcal{M}$ corresponds to a set F in $\widehat{\mathcal{F}}$ that contains $\biguplus_{(e,t)\in M} E_e^{(t)}$ as a subset. Hence, by Observation 38, the set M is a Δ-temporal matching of $\mathcal{G}|_{[\Delta(\ell-1)+1,\Delta\ell]}$.

We now show that $\mathcal{M}$ is ℓ-complete. Let M be a Δ-temporal matching of $\mathcal{G}$, $M^\ell = M|_{[\Delta(\ell-1)+1,\Delta\ell]}$, and $M' = M \setminus M^\ell$. Let also W be the set of vertex appearances in $\mathcal{G}|_{[\Delta(\ell-1)+1,\Delta\ell]}$ which are Δ-blocked by M'. Note that since M is a Δ-temporal matching, no time-edge in M^ℓ is incident with a vertex appearance in W. The latter together with Observation 38 imply that the sets $Y = \{v_t \in U \mid (v,t) \in W\}$ and $E_e^{(t)}, (e,t) \in M^\ell$ are pairwise disjoint. Since the maximum matching size of the underlying graph G is ν, we have that $|Y| = |W| \leq 4\nu(\Delta-1)$. For the same reason $|M^\ell| \leq \nu$ and therefore $\mathcal{F}$ contains a set $X = \biguplus_{(e,t)\in M^\ell} E_e^{(t)} \uplus D'$ of size 5ν, where D' is a set of dummy elements. Consequently, the cardinality of $X \uplus Y$ is at most $5\nu + 4\nu(\Delta-1)$ and hence $X \uplus Y$ is an independent set of $\mathcal{Q}$. Furthermore, observe that $w(X) = |M|$. Now, since $\widehat{\mathcal{F}}$ is a max $(5\nu(\Delta-1))$-representative of $\mathcal{F}$ with respect to w, the family $\widehat{\mathcal{F}}$ contains a set $\widehat{X}$ such that $\widehat{X}$ is disjoint from Y, the union $\widehat{X} \uplus Y$ is an independent set of $\mathcal{Q}$, and $w(\widehat{X}) \geq w(X)$. Let $\widehat{X}'$ be the set obtained from $\widehat{X}$ by removing the dummy elements. Hence $w(\widehat{X}') = w(\widehat{X})$ and by construction $\widehat{X}'$ is the union of pairwise disjoint sets $E_e^{(t)}$, $(e,t) \in M''$ for some set M'' of time-edges of $\mathcal{G}|_{[\Delta(\ell-1)+1,\Delta\ell]}$. Thus, $w(\widehat{X}') = |M''|$. By Observation 38 we conclude that M'' is a Δ-temporal matching of $\mathcal{G}|_{[\Delta(\ell-1)+1,\Delta\ell]}$. Moreover, no time-edge in M'' is incident with vertex appearances in W, as $\widehat{X}'$ is disjoint from Y. Hence, $M' \cup M''$ is a Δ-temporal matching in $\mathcal{G}$ and $|M' \cup M''| = |M'| + |M''| = |M'| + w(\widehat{X}) \geq |M'| + w(X) = |M'| + |M^\ell| = |M|$. ◀

C.8 Proof of Theorem 19

▶ **Lemma 40.** *Algorithm C.3 is correct, that is, for a given temporal graph $\mathcal{G} = (G, \lambda)$ of lifetime T and an integer $\Delta < T$, the algorithm returns the maximum size of a Δ-temporal matching in $\mathcal{G}$.*

Proof. To prove the lemma we first show by induction on $i \in [\frac{T}{\Delta}]$ that for every $M' \in \mathcal{M}_i$ the entry $\mathcal{T}[i, M']$ contains the maximum size of a Δ-temporal matching M in $\mathcal{G}|_{[1,\Delta i]}$ such

Algorithm C.3: Fixed-Parameter Algorithm for the Combined Parameter Δ and Maximum Matching Size ν of the Underlying Graph (Theorem 19).

Input: A temporal graph $\mathcal{G} = (G, \lambda)$ of lifetime T and an integer $\Delta < T$.
Output: The maximum size of a Δ-temporal matching in $\mathcal{G}$.

1 $\mathcal{T}[i, M'] \leftarrow 0$, for every $i \in [\frac{T}{\Delta}]$ and a subset M' of time-edges of $\mathcal{G}|_{[\Delta(i-1)+1,\Delta i]}$.
2 $\mathcal{M}_0 \leftarrow \{\emptyset\}$.
3 **for** $i \leftarrow 1$ **to** $\frac{T}{\Delta}$ **do**
4     $\mathcal{M}_i \leftarrow i$-complete family of Δ-temporal matchings of $\mathcal{G}|_{[\Delta(i-1)+1,\Delta i]}$.
5   **foreach** $M_L \in \mathcal{M}_{i-1}$ *and* $M_R \in \mathcal{M}_i$ **do**
6   **if** $M_L \cup M_R$ *is a* Δ*-temporal matching in* $\mathcal{G}$ **then**
7             $\mathcal{T}[i, M_R] \leftarrow \max\big\{\mathcal{T}[i, M_R], \mathcal{T}[i-1, M_L] + |M_R|\big\}$.
8 **return** $\max_{M' \in \mathcal{M}_{\frac{T}{\Delta}}} \mathcal{T}[\frac{T}{\Delta}, M']$.

that $M|_{[\Delta(i-1)+1,\Delta i]} = M'$. The statement is easily verifiable for $i = 1$.

Let now $i \geq 2$ and assume the statement holds for indices smaller than i. Let M^i be an arbitrary element in $\mathcal{M}_i$ and assume towards a contradiction that there is a Δ-temporal matching M in $\mathcal{G}|_{[1,\Delta i]}$ such that $|M| > \mathcal{T}[i, M^i]$ and $M|_{[\Delta(i-1)+1,\Delta i]} = M^i$.

Since $\mathcal{M}_{i-1}$ is an $(i-1)$-complete family of Δ-temporal matchings of $\mathcal{G}|_{[\Delta(i-2)+1,\Delta(i-1)]}$, there exists an $M^{i-1} \in \mathcal{M}_{i-1}$ such that $M' := \big(M \setminus M|_{[\Delta(i-2)+1,\Delta(i-1)]}\big) \cup M^{i-1}$ is a Δ-temporal matching and $|M'| \geq |M|$.

Since $M'|_{[\Delta(i-2)+1,\Delta(i-1)]} = M^{i-1}$, by the induction hypothesis we have $\mathcal{T}[i-1, M^{i-1}] \geq \big|M'|_{[1,\Delta(i-1)]}\big|$. Furthermore, since both M^{i-1} and M^i are subsets of M', their union $M^{i-1} \cup M^i$ is a Δ-temporal matching in $\mathcal{G}$. Consequently, Line 7 of the algorithm implies that $\mathcal{T}[i, M^i] \geq \mathcal{T}[i-1, M^{i-1}] + |M^i|$, and therefore $|M| > \mathcal{T}[i, M^i] \geq \big|M'|_{[1,\Delta(i-1)]}\big| + |M^i| = |M'|$, which is a contradiction.

To complete the proof, we observe that since $\mathcal{M}_{\frac{T}{\Delta}}$ is a $\frac{T}{\Delta}$-complete family of Δ-temporal matchings of $\mathcal{G}|_{[T-\Delta+1,T]}$, the above statement implies that the value $\max_{M' \in \mathcal{M}_{\frac{T}{\Delta}}} \mathcal{T}[\frac{T}{\Delta}, M']$ returned by the algorithm is the size of a maximum Δ-temporal matching of $\mathcal{G}$. ◀

Next, we analyze the running time of the algorithm.

▶ **Lemma 41.** *Algorithm C.3 runs in $2^{O(\nu\Delta)} \cdot |\mathcal{G}| \cdot \frac{T}{\Delta}$ time, where ν is the maximum matching size of underlying graph of $\mathcal{G}$.*

Proof. We represent our table $\mathcal{T}$ by a sparse set [12] that stores only non-zero entires of $\mathcal{T}$. Hence, Line 1 can be computed in constant time. By Lemma 20, Line 4 can be computed in $2^{O(\nu\Delta)} \cdot |\mathcal{G}|$ time and $|\mathcal{M}_i| \in 2^{O(\nu\Delta)}$. The latter implies that the for-loop of Line 5 executes $2^{O(\nu\Delta)}$ iterations. Furthermore, each of the iterations runs in $O(\nu)$ time. Hence, all in all, Algorithm C.3 runs in $2^{O(\nu\Delta)} \cdot |\mathcal{G}| \cdot \frac{T}{\Delta}$ time. ◀

Finally, we have everything at hand to show Theorem 19.

Proof of Theorem 19. Let $(\mathcal{G}, \Delta, k)$ be an instance of Temporal Matching, where $\mathcal{G}$ is a temporal graph of lifetime T and $\Delta, k \in \mathbb{N}$.

If $\Delta \geq T$, then we check whether the underlying graph G of $\mathcal{G}$ admits a matching of size at least k, which can be done in $O(k|\mathcal{G}|)$ time using the standard augmenting path-based method.

If $\Delta < T$, then we add at most $\Delta - 1$ trailing edgeless snapshots to $\mathcal{G}$ to guarantee that the lifetime of the resulting temporal graph is divisible by Δ. Note that this does not change the maximum size of a Δ-temporal matching. We then apply Algorithm C.3 to find the maximum size of a Δ-temporal matching in $\mathcal{G}$ and compare the resulting value with k. By Lemma 41 this can be done in $2^{O(\nu\Delta)} \cdot |\mathcal{G}| \cdot \frac{T}{\Delta}$ time, which implies the theorem. ◀

C.9 Proof of Proposition 21

We need the following notation for the proof. An equivalence relation R on the instances of some problem L is a *polynomial equivalence relation* if

(i) one can decide for each two instances in time polynomial in their sizes whether they belong to the same equivalence class, and
(ii) for each finite set S of instances, R partitions the set into at most $(\max_{x\in S} |x|)^{O(1)}$ equivalence classes.

An *AND-cross-composition* of a problem $L \subseteq \Sigma^*$ into a parameterized problem P (with respect to a polynomial equivalence relation R on the instances of L) is an algorithm that takes ℓ R-equivalent instances $x_1, \ldots, x_\ell$ of L and constructs in time polynomial in $\sum_{i=1}^{\ell} |x_i|$ an instance (x, k) of P such that

(i) k is polynomially upper-bounded in $\max_{1\leq i\leq \ell} |x_i| + \log(\ell)$ and
(ii) (x, k) is a yes-instance of P if and only if $x_{\ell'}$ is a yes-instance of L for every $\ell' \in \{1, \ldots, \ell\}$.

If an NP-hard problem L AND-cross-composes into a parameterized problem P, then P does not admit a polynomial-size kernel, unless NP $\subseteq$ coNP/poly [11], which would cause a collapse of the polynomial-time hierarchy to the third level.

Proof of Proposition 21. We provide an AND-cross-composition from Independent Set on graphs with maximum degree three [33]. Intuitively, we can just string together instances produced by Construction 1 in the time axis such that the large instance contains a large Δ-temporal matching if and only if all original instances are *yes*-instances.

In this problem we are asked to decide whether a given graph $H = (U, F)$ with maximum degree three contains a set of at least h pairwise non-adjacent vertices. Furthermore, it is important to observe that, given graph $H = (U, F)$ with maximum degree three, it is NP-complete to decide whether H contains an independent set of size h even if it is known that H does not contain an independent set of size $h + 1$ [33]. In the following, we assume that all instances have this property. We define an equivalence relation R as follows: Two instances $(H = (U, F), h)$ and $(H' = (U', F'), h')$ are equivalent under R if and only if the number of vertices is the same, that is, $|U| = |U'|$ and we have that $h = h'$. Clearly, R is a polynomial equivalence relation.

Now let $(H_1 = (U_1, F_1), h_1), \ldots, (H_\ell = (U_\ell, F_\ell), h_\ell)$ be R-equivalent instances of Independent Set with the above described extra conditions. We arbitrarily identify the vertices of all instances, that is, let $U = U_1 = \ldots = U_\ell$. For each (H_i, h_i) with $i \in [\ell]$ we construct an instance of Temporal Matching as described in Construction 1 (for an illustration see Figure 4) with the only difference that we add a fourth snapshot that does not contain any edges. Now we put all constructed temporal graphs next to each other in temporal order, that is, if $\mathcal{G}^{(i)} = (G^{(i)} = (V^{(i)}, E^{(i)}), \lambda^{(i)})$ with $\lambda^{(i)} : E^{(i)} \to [4]$ is the graph constructed for (H_i, h_i), then the overall temporal graph is $\mathcal{G} = (G(\bigcup_{i\in[\ell]} V^{(i)}, \bigcup_{i\in[\ell]} E^{(i)}), \lambda)$ with $\lambda(e) = \bigcup_{i\in[\ell]} \lambda^{(i)}(e)$, where we assume that $\lambda^{(i)}(e) = \emptyset$ if $e \notin E^{(i)}$. Note that $|\bigcup_{i\in[\ell]} V^{(i)}| \leq 2|U| + \binom{|U|}{2}$ since the temporal graphs produced by Construction 1 contain two vertices for every vertex of the Independent Set instance and one vertex for every edge of the Independent Set instance. Further, we set $\Delta = 2$ and $k = \ell \cdot h_1 + \sum_{i\in[\ell]} |F_i|$.

This instance can be constructed in polynomial time and $|V|$ is polynomially upper-bounded by the maximum size of an input instance. It is easy to check that the extra edgeless snapshot contained in each constructed temporal graph $\mathcal{G}^{(i)}$ prevents the Δ-temporal matchings from two adjacent constructed graphs $\mathcal{G}^{(i)}$ and $\mathcal{G}^{(i+1)}$ for $i \in [\ell-1]$ to interfere, that is, matching two vertices with a time edge from $\mathcal{G}^{(i)}$ cannot block vertices from $\mathcal{G}^{(i+1)}$ from being matched. Furthermore, since we assume that no instance (H_i, h_i) of Independent Set contains an independent set of size $h_1 + 1$, it cannot happen that the Δ-temporal matching of a constructed temporal graph $\mathcal{G}^{(i)}$ is larger than $h_1 + |F_i|$. It follows from the proof of Theorem 5, that the constructed Temporal Matching instance is a *yes*-instance if and only if for every $i \in [n]$ the Independent Set instance (H_i, h_i) is a *yes*-instance.

Since Independent Set is NP-hard under the above described restrictions [33] and we AND-cross-composed it into Temporal Matching with $\Delta = 2$ parameterized by $|V|$, this proves the proposition. ◀