Highlights

Computing Maximum Matchings in Temporal Graphs*

George B. Mertzios, Hendrik Molter, Rolf Niedermeier, Viktor Zamaraev, Philipp Zschoche

- We establish new results and refine previous work on the temporal matching problem.
- We show NP-hardness for very restricted cases and employ a new natural concept of so-called "temporal line graphs" in this process.
- We present an approximation algorithm and a fixed-parameter algorithm for the solution size.

Computing Maximum Matchings in Temporal Graphs

George B. Mertzios[a,1], Hendrik Molter[b,c,2], Rolf Niedermeier[b], Viktor Zamaraev[d,3], Philipp Zschoche[b]

[a] *Department of Computer Science, Durham University, UK*
[b] *Algorithmics and Computational Complexity, TU Berlin, Germany*
[c] *Department of Industrial Engineering and Management, Ben-Gurion University of the Negev, Beer-Sheva, Israel*
[d] *Department of Computer Science, University of Liverpool, UK*

Abstract

Temporal graphs are graphs whose topology 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. We introduce and study the complexity of a natural temporal extension of the classical graph problem Maximum Matching, taking 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. We prove strong computational hardness results for Maximum Temporal Matching, even for elementary

⋆An extended abstract of this work appeared in the proceedings of the 37th International Symposium on Theoretical Aspects of Computer Science (STACS '20) [54]. This extended version provides full proof details and does not contain one result of the extended abstract [54] that has in the meantime been strictly improved by one of the authors [64].

Email addresses: george.mertzios@durham.ac.uk (George B. Mertzios), molterh@post.bgu.ac.il (Hendrik Molter), rolf.niedermeier@tu-berlin.de (Rolf Niedermeier), viktor.zamaraev@liverpool.ac.uk (Viktor Zamaraev), zschoche@tu-berlin.de (Philipp Zschoche)

[1]Supported by the EPSRC grant EP/P020372/1.

[2]Supported by the DFG, project MATE (NI 369/17) and by the ISF, grant No. 1070/2. Main part of this work was done while H. Molter was affiliated with TU Berlin.

[3]Supported by the EPSRC grant EP/P020372/1. The main part of this paper was prepared while affiliated with the Department of Computer Science, Durham University, UK.

cases, as well as fixed-parameter algorithms with respect to natural parameters and polynomial-time approximation algorithms.

Keywords: Link Streams, Temporal Line Graphs, NP-hardness, APX-hardness, Approximation Algorithms, Fixed-Parameter Tractability, Kernelization, Independent Set

---

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, 16, 27, 52, 53]. 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 from the foundational work of Kempe et al. [45]. 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 1.1 (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 to see it as an ordered set (according to the discrete time slots) of static graphs (called *snapshots*) on a fixed vertex set. Due to their vast applicability in many areas, temporal graphs have been studied from different perspectives under various names such as *time-varying* [61], *evolving* [21, 28], *dynamic* [14, 34], and *graphs over time* [49]; see also the survey papers [12–14] 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" to the temporal setting: 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.[4] Then, for any given Δ, the problem Maximum Temporal Matching asks for the largest possible set of pairwise 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. This requirement can model settings where a short "recovery" period is needed for every vertex that participates in the matching, e.g., a short rest after an energy-demanding activity. Thus it makes particular sense to study the complexity of the problem for small (constant) values of Δ. Note that the concept of Δ-windows has also been employed in other temporal graph problem settings [3, 15, 39, 53].

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 hardness results, we introduce the notion of *temporal line graphs* which form a class of (static) graphs of independent interest and may prove useful in other contexts, too. This notion paves the way 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 3.13) 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. [20] 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

[4]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).

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. [44] 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. [11, 30, 40, 41]. This difficulty of breaking the barrier of the ratio $\frac{1}{2}$ 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, 47], 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 [59] proved that MRG actually provides an approximation ratio of $\frac{1}{2}+\frac{1}{256}$. Similarly to the above problems, it is straightforward[5] to approximate MAXIMUM TEMPORAL MATCHING in polynomial time within a factor of $\frac{1}{2}$. However, we manage to provide a simple (non-randomized) approximation algorithm which, for any constant Δ, achieves an approximation ratio $\frac{1}{2}+\varepsilon$ for some $\varepsilon = \varepsilon(\Delta)$. 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 polynomial-time approximation algorithms, the classical (static) maximum matching problem (which is polynomial-time solvable [26]) has recently also attracted many research efforts in the area of parameterized algorithms for polynomial-time solvable problems. Parameters which have been studied include the solution size [35], the modular-width [48], the clique-width [22], the treewidth [29], the feedback vertex number [51], and the feedback edge number [46, 51]. Given that MAXIMUM TEMPORAL MATCHING is NP-hard, we show fixed-parameter tractability with respect to the solution size as a parameter. An extended abstract [54] of this article described an FPT-algorithm with respect to the combined parameter of time-window size Δ and the 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). Recently, this algorithm

[5] 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 current solution.

has been improved to a running time of $\Delta^{O(\rho)} \cdot |\mathcal{G}|$, where ρ is the maximum vertex cover number of the underlying graph of the temporal subgraph of $\mathcal{G}$ induced by any Δ-window [64]. Note that, the number ρ can be arbitrarily smaller than the maximum matching size of the underlying graph but it is at most twice this parameter.

It is worth mentioning that another temporal variation of Maximum Matching was recently proposed by Baste et al. [9]. The main difference to our model 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].

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. The problem has also been investigated on certain geometric temporal graphs [57].

To the best of our knowledge, the main alternative model for temporal matchings in temporal graphs is the concept of multistage (perfect) matchings which was introduced by Gupta et al. [37]. 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 approximation algorithms [8, 18, 37].

Recently, so-called 0-1 timed matchings in temporal graphs have been studied [50]. Here, each edge of the temporal graph is associated with a set of non-overlapping time intervals and the goal is to find a maximum set of edges such that their intervals are pairwise non-overlapping. Here, also mostly intractability and approximation results are obtained.

2. Preliminaries

In this section we present all necessary notation and terminology as well as some easy initial observations about our problem setting.

2.1. Notation and Terminology

Let $\mathbb{N}$ denote the natural numbers without zero. We refer to a set of consecutive natural numbers $[i, j] = \{i, i + 1, \ldots, j\}$ for some $i, j \in \mathbb{N}$ with $i \leq j$ as an *interval*, and to the number $j - i + 1$ as the *length* of the interval. If $i = 1$, then we denote $[i, j]$ simply by $[j]$.

Static graphs. We use standard notation and terminology from graph theory [24]. 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. 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 $E' = (E \setminus \{\{u, v\}\}) \cup \{\{v, w\}, \{u, w\}\}$. A graph H is a *subdivision* of a graph G if there is a sequence of graphs $G_1, G_2, \ldots, G_x$ 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. A graph H is a *topological minor* of G if there is a subgraph G' of G that is a subdivision of H. A graph H is 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 the 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 $e \cap e' \neq \emptyset$ [24]. 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 [24].

Parameterized complexity. We use standard notation and terminology from parameterized complexity [23, 25]. 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, i.e., L is *para-NP-hard*, it cannot be contained in FPT[6] 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 $|(I', r')| \leq r^{O(1)}$.

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* 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 $\mathcal{E}(\mathcal{G}) := \{(e, t) \mid e \in E$ 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 $\lambda'(e) := \lambda(e) \setminus \{t \mid (e, t) \in M\}$ for all $e \in E'$. For a time-edge set $\mathcal{E}$ and integers a and b, we denote by $\mathcal{E}[a, b] := \{(e, t) \in \mathcal{E} \mid a \leq t \leq b\}$ the subset of $\mathcal{E}$ between the time steps a and b. Analogously, for a temporal graph $\mathcal{G} := (V, (E_i)_{i=1}^{T})$ we denote by $\mathcal{G}[a, b]$ the temporal graph on the vertex set V with the time-edge set $\mathcal{E}(\mathcal{G})[a, b]$.

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

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 $|\mathcal{G}| := |V| + \sum_{t=1}^{T} \max\{1, |E_t|\} \in O(n + mT)$. Furthermore, in accordance with the literature [63, 65] 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 2.1 (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 $|t - t'| \geq \Delta$.*

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 a Δ-temporal matching of a given size k by Temporal Matching.

Temporal Matching	
Input:	A temporal graph $\mathcal{G} = (G, \lambda)$ and integers $k \in \mathbb{N}$ and $\Delta \in \mathbb{N}$.
Question:	Does $\mathcal{G}$ contain a Δ-temporal matching of size k?

We remark that our definition of a Δ-temporal matching is similar to γ-*matchings* by Baste et al. [9]. We discuss some basic observations about our problem settings in Section 2.2 and discuss the relation between our model and the model of Baste et al. [9] in Section 2.3.

Temporal line graphs. In the following, we transfer the concept of line graphs to temporal graphs and temporal matchings. 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 2.2 (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 $E(L_\Delta(\mathcal{G})) : s = \{\{e_t, e'_{t'}\} \mid e \cap e' \neq \emptyset \wedge |t - t'| < \Delta\}$. 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 2.3. *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 $L_\Delta(\mathcal{G})$.

2.2. 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, there 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:

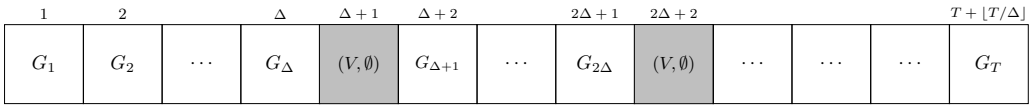


Figure 1: Inserting "empty" snapshots to reduce Temporal Matching on instances $(\mathcal{G}, \Delta, k)$ to Temporal Matching on instances $(\mathcal{G}, \Delta + 1, k)$.

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 $O(Tm) = O(\Delta m)$ time. The latter can be solved in $O(\sqrt{n}m)$ time [55]. Thus, we have the following.

Observation 2.4. *Let $\mathcal{G}$ be a temporal graph of lifetime T. We can compute a T-temporal matching in $\mathcal{G}$ in $O(m(\sqrt{n} + T))$ time.*

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

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 1). □

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 the hardness part of the proof.

*2.3. 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 for 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 are not aware of 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.

3. Hardness Results

In this section we present our computational hardness results. In Section 3.1 we show that the optimization problem Maximum Temporal Matching is APX-hard and in Section 3.2 we show that Temporal Matching is NP-hard even if the underlying graph is a path.

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 an *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 3.1. 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* $P = NP$.

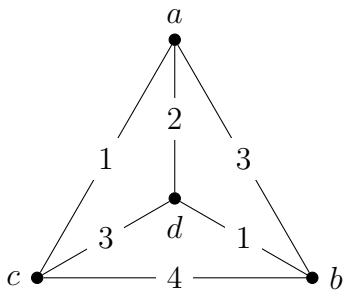
It is easy to check that the construction uses only three time steps and every edge appears in exactly one time step.

Construction 3.2. *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 Vizings's theorem and can be found in* $O(|E|)$ *time [60]. 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* $\{w_e, u_\varphi\}, \{w_e, v_\varphi\}$, *where* $c(e) \equiv \varphi \pmod 2$. *In the temporal graph* (H, λ) *every edge of the underlying graph appears in exactly one of the three time slots:*

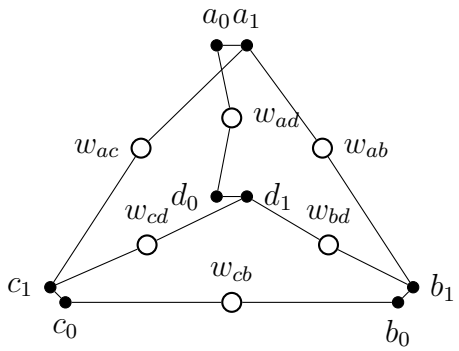
1. $\lambda(\{w_e, u_\varphi\}) = \lambda(\{w_e, v_\varphi\}) = 1$, *where* $c(e) \equiv \varphi \pmod 2$, *for every edge* $e = \{u, v\} \in E$ *such that* $c(e) \in \{1, 2\}$;
2. $\lambda(\{v_0, v_1\}) = 2$ *for every* $v \in V$;
3. $\lambda(\{w_e, u_\varphi\}) = \lambda(\{w_e, v_\varphi\}) = 3$, *where*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

Construction 3.2 is illustrated in Figure 2. 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 two 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.

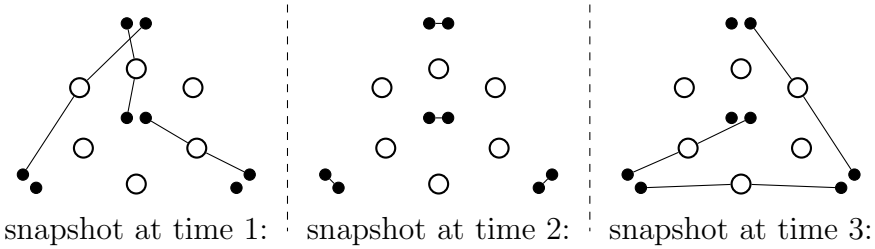
Lemma 3.3. *Let* $G = (V, E)$ *be a cubic graph and let* (H, λ) *be the temporal graph obtained by applying Construction 3.2 to* G. *Let* M *be a 2-temporal matching of* (H, λ). *Then there exists a 2-temporal matching* M' *of* (H, λ) *such that* $|M'| = |M|$, *and for every edge* $e = \{u, v\} \in E$ *the matching* M' *contains at most one of the two time-edges* $(\{u_0, u_1\}, 2)$ *and* $(\{v_0, v_1\}, 2)$. *Moreover,* M' *can be constructed from* M *in polynomial time.*



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



(b) The underlying graph H. White vertices correspond to edges of G, black vertices correspond to vertices of G.



(c) The temporal graph (H, λ).

Figure 2: Example of the reduction from Maximum Independent Set on cubic graphs to Maximum Temporal Matching.

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 $k = 0$ the statement is trivial. Let

$k \geq 1$, and let $e = \{u, v\} \in E$ be an edge such 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 3.4. *Let $G = (V, E)$ be a cubic graph and let (H, λ) be the temporal graph obtained by applying Construction 3.2 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 3.3, 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 of the corresponding independent set in the original graph. For a static graph G we denote by $\alpha(G)$ the size of a maximum independent set in G, and for a temporal graph (H, λ) we denote by $\mu_2(H, \lambda)$ the size of a maximum 2-temporal matching in (H, λ).

Lemma 3.5. *Let $G = (V, E)$ be a cubic graph and let (H, λ) be the temporal graph obtained by applying Construction 3.2 to G. Then $\mu_2(H, \lambda) = \alpha(G) + \frac{3n}{2}$.*

Proof. Let $\alpha := \alpha(G)$ and $\mu_2 := \mu_2(H, \lambda)$. We start by proving $\mu_2 \leq \alpha + \frac{3n}{2}$. Let M be a maximum 2-temporal matching of (H, λ). By Lemma 3.4 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 to 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 $\{w_e, u_\varphi\}, \{u_0, u_1\}$ 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 $\{w_e, u_\varphi\}$ and $\{w_e, v_\varphi\}$, where $c(e) \equiv \varphi \pmod 2$, are not in conflict in M. It remains to show that the time-edges $(\{w_e, u_\varphi\}, i)$ and $(\{w_{e'}, u_\varphi\}, j)$ corresponding to the adjacent edges $\{w_e, u_\varphi\}$ and $\{w_{e'}, u_\varphi\}$ 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 $c(e) \equiv \varphi \pmod 2$ and $c(e') \equiv \varphi \pmod 2$, 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 3.2 together with the procedure described in Lemma 3.4 to obtain an independent set from a temporal matching is actually an L-reduction. Before we proceed, let us recall the definition of an L-reduction [6]. Let A and B be two maximization problems and let c_A and c_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 $f(I)$, it holds that $OPT_A(I) - c_A(g(M)) \leq \gamma \cdot (OPT_B(f(I)) - c_B(M))$.

Lemma 3.6. *Construction 3.2 together with the procedure described by Lemma 3.4 constitute an L-reduction from* Maximum Independent Set *on cubic graphs to* Maximum Temporal Matching *with* $\beta = 7$ *and* $\gamma = 1$.

Proof. 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 3.2 corresponds to function f. Clearly, the reduction is computable in polynomial time. The polynomial-time procedure guaranteed by Lemma 3.4 corresponds to function g. It remains to show that conditions (4) and (5) in the definition of an L-reduction are met.

By Lemma 3.5 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 3.4, 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 3.5 and the inequality follows from Lemma 3.4. Thus, condition (5) holds with parameter $\gamma = 1$. □

We are now ready to prove the main result of this subsection.

Proof of Theorem 3.1. The NP-completeness claim follows directly from Lemma 3.5 and the NP-completeness of the Maximum Independent Set in cubic graphs [33, 56].

In Section 4.1 we provide a constant-factor approximation algorithm for Maximum Temporal Matching, which implies that the problem belongs to APX. In the rest of the proof we argue that there is no polynomial-time $\frac{664}{665}$-approximation algorithm for Maximum Temporal Matching, unless P = NP, which also implies APX-hardness, and therefore APX-completeness.

We will show that our reduction together with a polynomial-time δ-approximation algorithm $\mathcal{A}$ for Maximum Temporal Matching, where $\delta \geq \frac{664}{665}$, would 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 $\frac{94}{95}$ [19].

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 3.6, 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. □

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 [42, 43]. For Temporal Matching we can observe the following.

Observation 3.7. 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.*

Proof. When investigating the original reduction from 3SAT to Vertex Cover on cubic graphs [33, 56], 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 $2^{o(|V|)}$-time algorithm for Vertex Cover on cubic graphs unless the ETH fails. It follows that there is no $2^{o(|V|)}$-time algorithm for Independent Set on cubic graphs unless the ETH fails. If we treat the reduction presented in Construction 3.2 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 3.4 and Lemma 3.5), and hence we have $k \in O(|V|)$. 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 $2^{o(|V|)}$-time algorithm for Independent Set on cubic graphs (note that T is constant in the reduction), which contradicts the ETH. □

Observation 3.8. Temporal Matching *is NP-complete, even if* $\Delta = 2$*,* $T = 5$*, and the underlying graph of the input temporal graph is complete.*

Proof Sketch. We observe that Construction 3.2 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 3.8. □

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 (or even existence of an XP-algorithm) unless P = NP, even if combined with the lifetime T. In other words, Temporal Matching is para-NP-hard for these parameter combinations. Note that many structural parameters fall into this category, such as domination number, distance to cluster graph, clique cover number, etc.

Adapting Construction 3.2 to the Model of Baste et al. [9]. We remark that our reduction for Theorem 3.1 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 3.2) 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 strengthens the hardness result by Baste et al. [9].

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.

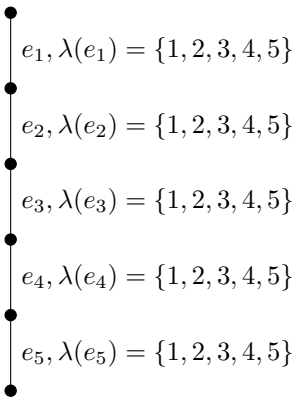
Theorem 3.9. Temporal Matching *is NP-complete even if* $\Delta = 2$ *and the underlying graph of the input temporal graph is a path.*

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 2.3, solving Independent Set on a temporal line graph is equivalent to solving Temporal Matching on the corresponding temporal graph. We proceed as follows.

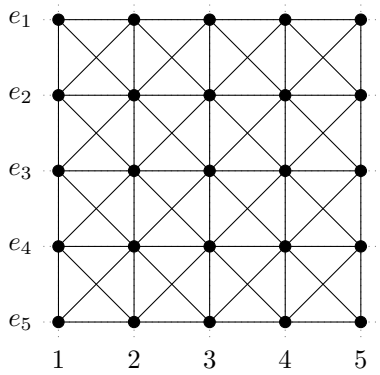
1. We show that 2-temporal line graphs of temporal graphs that have a path as underlying graph have a grid-like structure. More specifically, we show that they are induced subgraphs of so-called *diagonal grid graphs* or *king's graphs* [17, 36].

2. We show that Independent Set is NP-complete on induced subgraphs of diagonal grid graphs which together with Observation 2.3 yields Theorem 3.9.

    - 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.



(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 3: A temporal line graph with a path as underlying graph where edges are always active and its 2-temporal line graph.

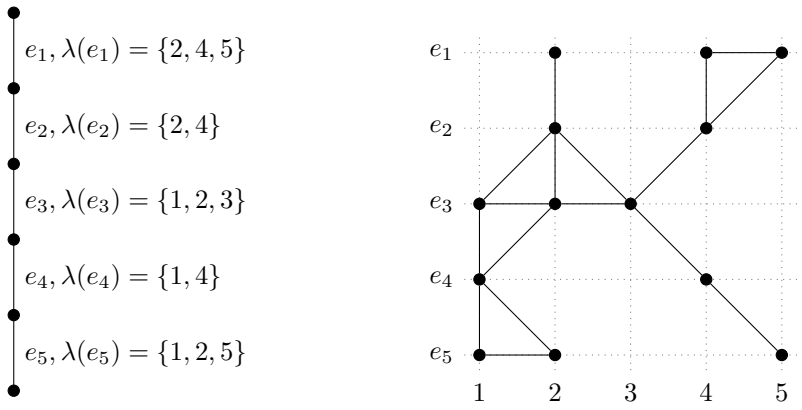
- We show how to modify the subdivision of a cubic planar graph in a way that results in an induced subgraph of a diagonal grid graph, such that NP-hardness of finding independent sets of certain size is preserved.

Definition 3.10 (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.

Observation 3.11. *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 4 for an example. Hence, we have that every induced subgraph of a diagonal grid graph is a 2-temporal line graph.



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

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

Figure 4: A temporal line graph with a path as underlying graph where edges are *not* always active and its 2-temporal line graph.

Corollary 3.12. *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 2.3, 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.

Theorem 3.13. 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. [20], who showed that Independent Set is NP-complete on unit disk graphs. It is easy to see from Definition 3.10 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 3.13. The first building block for the reduction is the fact that we can embed cubic planar graphs into a grid [62]. 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 large *diagonal* grid graph. The last step is to further modify the embedding such that it can be obtained from the original graph by subdividing each edge an even number of times, this ensures that NP-hardness of Independent Set is preserved [58].

It is easy to check that Theorem 3.13, Observation 2.3, and Corollary 3.12 together imply Theorem 3.9. Theorem 3.9 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 yields para-NP-hardness, even if the parameters are 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.

Proof of Theorem 3.13. We prove Theorem 3.13 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 an polynomially large grid graph.

Proposition 3.14 (Special case of Theorem 2 from Valiant [62])**.** *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.*

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 3.15. *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 3.14 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 $\{(i, j), (i, j{+}1)\}, \{(i, j), (i{+}1, j)\}, \{(i{+}1, j), (i{+}2, j)\} \in E''$, 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''. This modification is illustrated in Figure 5a. Rotated versions of this configuration are modified analogously.

2. If $\deg_{G''}((i, j)) = 3$ and $\{(i, j), (i, j{+}1)\}, \{(i, j), (i{+}1, j)\}, \{(i{+}1, j), (i{+}2, j)\}, \{(i, j), (i - 1, j)\}, \{(i - 1, j), (i - 2, j)\} \in E''$, then we delete

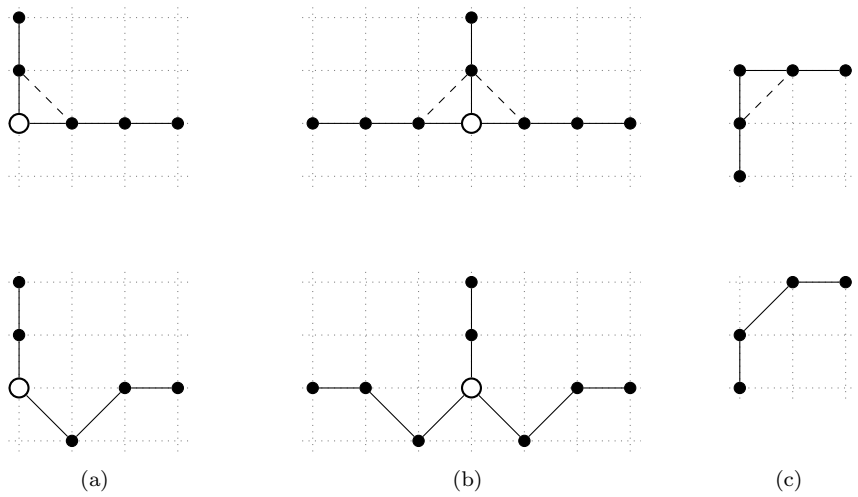


Figure 5: Illustration of the modifications described in the proof of Lemma 3.15. The situation before the moficication is depiced above, dashed edges show unwanted edges present in an induced subgraph of a diagonal grid graph. The situation after the modification is depicted below.

$(i+1, j)$ 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 5b. 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 5c. 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 two times increases the size of a maximum independent set exactly by one.

Observation 3.16 (Poljak [58])**.** *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.*

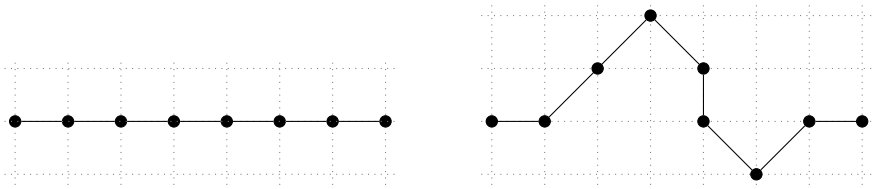
From this observation it follows that if we can guarantee that for every cubic planar graph there is a subdivision that subdivides every edge an even number of times and that is an induced subgraph of a diagonal grid graph of polynomial size, then we are done.

Lemma 3.17. *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 3.15 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 3.15.

Recall that every edge e in G is replaced by a path P_e in G'. From Observation 3.16 it 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 that is described in the proof of Lemma 3.15 ensures that each path P_e in G' that corresponds to an edge e in G contains at least 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



(a) Before. (b) After.

Figure 6: Illustration of the modification described in the proof of Lemma 3.17. It shows how to increase the length of an induced path of a diagonal grid graph by one.

$\{(i+1,j),(i+2,j+1)\}, \{(i+2,j+1),(i+3,j+2)\}, \{(i+3,j+2),(i+4,j+1)\}, \{(i+4,j+1),(i+4,j)\}, \{(i+4,j),(i+5,j-1)\}, \{(i+5,j-1),(i+6,j)\}$. It is easy to check that this reconnects the path and increases the number of edges by one. This modification is illustrated in Figure 6. 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. □

Now, Theorem 3.13 follows directly from Lemma 3.17 and Observation 3.16.

4. Algorithms

In this section, we present an approximation algorithm (Section 4.1) and an exact fixed-parameter algorithm (Section 4.2) for Temporal Matching. First, in Section 4.1 we present an $\frac{\Delta}{2\Delta-1}$-approximation algorithm for Maximum Temporal Matching. Second, in Section 4.2 we present an FPT-algorithm for Temporal Matching parameterized by the solution size k.

4.1. Approximation of Maximum Temporal Matching

In this subsection, 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 4.1. Maximum Temporal Matching *admits an* $O\left(Tm(\sqrt{n}+\Delta)\right)$*-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 $a_2 - b_1 - 1$. 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 a label in S are present.

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 notions of Δ-window, partial Δ-window, and Δ-template are illustrated in Figure 7. A time slot t is *covered* by a Δ-template $\mathcal{S}$ if t belongs to an interval of $\mathcal{S}$. We show the following properties of Δ-templates which we need to prove the approximation ratio of our algorithm.

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

```
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.
```

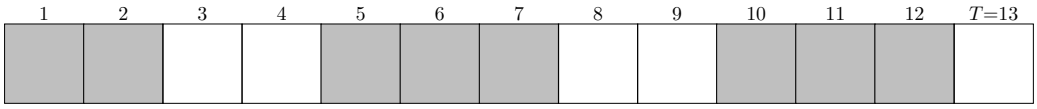


Figure 7: The gray slots form the intervals of a Δ-template, where $\Delta = 3$. Interval $[1, 2]$ is a partial Δ-window. Intervals $[5, 7]$ and $[10, 12]$ are Δ-windows.

Lemma 4.2. *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. □

By definition, for any two distinct intervals W_1, W_2 in a Δ-template $\mathcal{S}$ 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[7]. 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 2.4, 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 4.3. *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 $\mathcal{G}$.

We show that, given the instance $(\mathcal{G}, \Delta)$ of Maximum Temporal Matching, 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

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

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

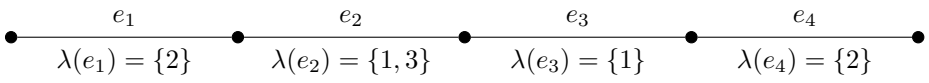


Figure 8: A temporal graph witnessing that the analysis of Algorithm 4.1 is tight for $\Delta = 2$. An optimal solution $\{(e_2, 1), (e_4, 2), (e_2, 3)\}$ consists of three time-edges. Algorithm 4.1 considers the following three templates: $\{[1, 2]\}, \{[2, 3]\}$, and $\{[1, 1], [3, 3]\}$. In the first two templates the algorithm might select $\{(e_1, 2), (e_4, 2)\}$ and in the third template $\{(e_3, 1), (e_2, 3)\}$. In each of the cases it is not possible to increase the temporal matching by adding time-edges from the gaps.

Using the above inequalities and Lemma 4.2 we derive

$$(2\Delta - 1)|M^{\mathcal{S}'}| \geq \sum_{\mathcal{S} \in \mathfrak{S}} |M^{\mathcal{S}}|$$

$$\geq \sum_{\mathcal{S} \in \mathfrak{S}} \sum_{W \in \mathcal{S}} \sum_{t \in W} |M_t^{\mathcal{S}}| \geq \sum_{\mathcal{S} \in \mathfrak{S}} \sum_{W \in \mathcal{S}} \sum_{t \in W} |M_t^*| = \Delta \sum_{t=1}^{T} |M_t^*| = \Delta |M^*|,$$

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 4.2 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. □

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. However, 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 8) 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.

4.2. Fixed-parameter tractability for the parameter solution size

In this section, we show that Temporal Matching parameterized by the solution size k is fixed-parameter tractable. Beforehand, it was

only known that Temporal Matching parameterized by $k + \Delta$ is fixed-parameter tractable [9]. Hence, this improves the result of Baste et al. [9] from a classification point of view. In particular, we show the following.

Theorem 4.4. Temporal Matching *can be solved in* $O\left((2k-1)^k \cdot |\mathcal{G}|\right)$ *time.*

The formal proof of Theorem 4.4 is deferred to the end of this section. The algorithm behind Theorem 4.4 is a simple search-tree algorithm that is looking for a small set of time-edges to branch on, meaning that at least one of the time-edges is in a solution. To this end, we introduce a simple data reduction rule.

Reduction Rule 4.5. *Let* $(\mathcal{G} := (V, (E_i)_{i=1}^T), k, \Delta)$ *be an instance of* Temporal Matching*, where* $E_T = \emptyset$*. Then, output the instance* $(\mathcal{G}' := (V, (E_i)_{i=1}^{T-1}), k, \Delta)$ *of* Temporal Matching*.*

Lemma 4.6. *Reduction Rule 4.5 can be exhaustively applied in* $O(|\mathcal{G}|)$ *time. Moreover, the instance we apply Reduction Rule 4.5 on is a* yes*-instance if and only if the output instance is an* yes*-instance as well.*

Proof. We iterate once over $\mathcal{G}$ to find the maximum $p \in [T]$ such that $E_p \neq \emptyset$. Then, we copy everything from the input to the output except the edge sets E_i with $i \in [p+1, T]$. Observe that this takes only linear time and that Reduction Rule 4.5 is not applicable on the output instance, as the last edge set E_p of the output instance contains at least one edge.

To complete the proof, observe that a time-edge set M is a Δ-temporal matching in $\mathcal{G}$ if and only if it is a Δ-temporal matching in $\mathcal{G}'$ since $\mathcal{E}(\mathcal{G}) = \mathcal{E}(\mathcal{G}')$. □

We now observe that when Reduction Rule 4.5 is not applicable, then any Δ-temporal matching of maximum size contains at least one time-edge from the last Δ layers.

Observation 4.7. *Let* M *be a* Δ*-temporal matching of maximum size in* $\mathcal{G} := (V, (E_i)_{i=1}^T)$*, where* $E_T \neq \emptyset$ *and* $\Delta \in \mathbb{N}$*. Then,* $M \cap \mathcal{E}(\mathcal{G})[T - \Delta + 1, T] \neq \emptyset$*.*

Proof. Suppose that $M \cap \mathcal{E}(\mathcal{G})[T - \Delta + 1, T] = \emptyset$. For any $e \in E_T$ we have that $M \cup \{(e, T)\}$ is a Δ-temporal matching of size $|M| + 1$ in $\mathcal{G}$—a contradiction. □

We can take this observation a little bit further to develop a branching strategy of a search-tree algorithm for Temporal Matching by selecting sufficiently many time-edges at the end of the temporal graph (that is, $\mathcal{G}[T - \Delta + 1, T]$) which are Δ-blocked by one fixed time-edge from time step T. In particular, we show the following.

Lemma 4.8. *Given a temporal graph $\mathcal{G}$ and a $k \in \mathbb{N}$, one can compute in $O(|\mathcal{G}|)$ time a time-edge set $X \subseteq \mathcal{E}(\mathcal{G})$ of size at most $2k - 1$ such that if there is a Δ-temporal matching of size at least k in $\mathcal{G}$, then there is one containing a time-edge from X.*

Proof. Let $\mathcal{G}$ be the given temporal graph, and let $k \in \mathbb{N}$. We first construct the time-edge set X and then show that one of these edges is contained in a Δ-temporal matching of size at least k (if any such Δ-temporal matching exists).

We assume that $k \leq |\mathcal{E}(\mathcal{G})|$; otherwise we can output an empty set as there is no Δ-temporal matching of size k. By Lemma 4.6, we may assume that Reduction Rule 4.5 is not applicable. Hence, there is a time-edge $(\{v, \hat{u}\}, T) \in \mathcal{E}(\mathcal{G})$.

Then, we compute a set X of time-edges which Δ-block (v, T):

$$X := \{(\{v, w\}, t) \in \mathcal{E}(\mathcal{G})[T - \Delta + 1, T] \mid \forall(\{v, w\}, t') \in \mathcal{E}(\mathcal{G}) \colon t' \leq t\}.$$

If $|X| > 2k - 1$, then we remove arbitrary time-edges from X until $|X| = 2k - 1$. Intuitively, X is a set of $2k - 1$ latest appearing time-edges that Δ-block (v, T). Note that we can compute X in $O(|\mathcal{G}|)$ time.

Let M be a Δ-temporal matching of size k in $\mathcal{G}$ with $M \cap X = \emptyset$ and pick an arbitrary time-edge $(\{v, u\}, T) \in X$. Assume that $M \cup \{(\{v, u\}, T)\}$ is not a Δ-temporal matching, otherwise we would be done. Let $(\{v, v'\}, t') \in M$ be the time-edge which Δ-blocks (v, T). We assume that vertex appearances (v, T) and (u, T) are Δ-blocked by different time-edges in M ($v' \neq u$); otherwise, we could replace $(\{v, v'\}, t')$ in M with $(\{v, u\}, T)$ to construct a Δ-temporal matching of size $|M| = k$ which contains a time-edge from X. Note that this implies that we removed some arbitrary time-edges from X, otherwise the last appearance of edge $\{v, v'\}$ must be in X and thus we could again replace $(\{v, v'\}, t')$ in M with $(\{v, u\}, T)$. Thus, $|X| = 2k - 1$. Let $Y := \{(w, t'') \in V \mid (\{v, w\}, t'') \in X\}$ be the set of endpoints of X excluding vertex v. Note that each time-edge in X is the last appearance of that edge in $\mathcal{G}$. Hence, $|Y| = 2k - 1$. Since X only contains last appearances

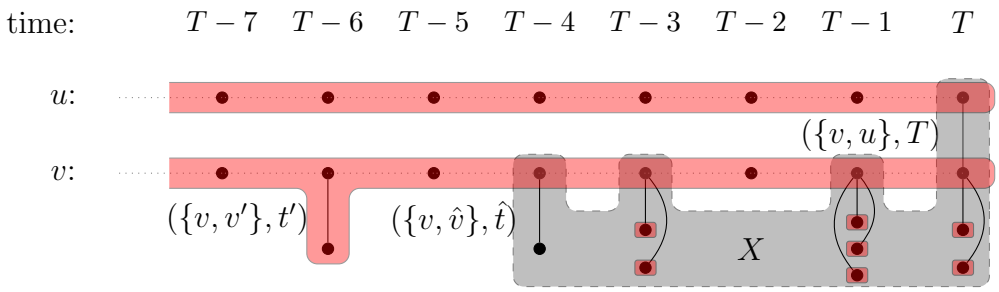


Figure 9: Depicts the situation in the proof of Lemma 4.8 where the pigeonhole principle is applied. Black dots represent vertex appearances. Vertex appearances that correspond to the same vertex lie on the same dotted line. Vertex appearances which are Δ-blocked by time-edges in M are in the red areas. The time-edge set X is marked by the gray area.

and $|Y| = 2k-1$, the $k-1$ time-edges in $M \setminus \{(\{v,v'\},t')\}$ can Δ-block at most $2k-2$ vertex appearances in Y. By the pigeonhole principle, one of the endpoints in Y is not Δ-blocked. Refer to Figure 9 for an illustration.

Hence, there is a time-edge $(\{v,\hat{v}\},\hat{t}) \in X$ such that $(M \setminus \{(\{v,v'\},t')\}) \cup \{(\{v,\hat{v}\},\hat{t})\}$ is a Δ-temporal matching of size k. □

To show Theorem 4.4 (the fixed-parameter tractability of Temporal Matching parameterized by solution size k), we employ a search-tree algorithm that branches on the time-edge set X from Lemma 4.8.

Proof of Theorem 4.4. Let $I := (\mathcal{G} := (V, (E_i)_{i=1}^T), k, \Delta)$ be an instance of Temporal Matching. Assume that $k \neq 0$, otherwise we can output *yes*. We use an algorithm comprising the following steps.

(i) Set $M := \emptyset$.

(ii) Compute time-edge set X from Lemma 4.8. If $X = \emptyset$, then output *no*.

(iii) Guess a time-edge $(e,t) \in X$ and set $M := M \cup \{(e,t)\}$.

(iv) If $|M| = k$, then output *yes*. Otherwise, remove (e,t) and all time-edges from $\mathcal{G}$ which are Δ-blocked by (e,t) and jump (recursively) to Step (ii).

A search-tree algorithm that considers each possible guesses in Step (iii) runs in $O((2k-1)^k|\mathcal{G}|)$ time as $|X| \leq 2k-1$, $|M|$ grows in each recursive call until

it reaches k, and all operations expect the recursive call can be computed in $O(|\mathcal{G}|)$ time, see Lemma 4.8. We now show that the described algorithm outputs *yes* if and only if the input instance I is a *yes*-instance.

($\Rightarrow$): Note that after Step (iii), the time-edge set M is always a Δ-temporal matching in the original input temporal graph, since either $M = \emptyset$ or we removed in Step (iv) all time-edges from $\mathcal{G}$ which are not Δ-independent with time-edges in M. Hence, if we output *yes*, then I is indeed a *yes*-instance.

($\Leftarrow$): This direction immediately follows from Lemma 4.8 as we consider each possible time-edge in X and remove only time-edge from $\mathcal{G}$ which cannot be in a Δ-temporal matching together with the time-edge chosen from X. □

5. Conclusion

The following issues remain future research challenges. First, on the side of polynomial-time approximability, improving the constant approximation factors is desirable and seems feasible. Beyond, by lifting polynomial time to FPT time, even approximation schemes in principle seem possible, thus circumventing our APX-hardness result (Theorem 3.1). 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 k, the question for a polynomial-size kernel for parameter k naturally arises. 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.

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. *Journal of Computer and System Sciences*, 107:108–123, 2020.

[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)*, volume 55 of *LIPIcs*, pages 149:1–149:14. Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 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 für Informatik, 2018.

[9] Julien Baste, Binh-Minh Bui-Xuan, and Antoine Roux. Temporal matching. *Theoretical Computer Science*, 806:184–196, 2020.

[10] Matthias Bentert, Anne-Sophie Himmel, Hendrik Molter, Marco Morik, Rolf Niedermeier, and René Saitenmacher. Listing all maximal k-plexes in temporal graphs. *ACM Journal of Experimental Algorithmics*, 24(1): 1.13:1–1.13:27, 2019.

[11] Niv Buchbinder, Danny Segev, and Yevgeny Tkach. Online algorithms for maximum cardinality matching with edge arrivals. *Algorithmica*, 81 (5):1781–1799, 2019.

[12] 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`.

[13] 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`.

[14] 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.

[15] Arnaud Casteigts, Anne-Sophie Himmel, Hendrik Molter, and Philipp Zschoche. Finding temporal paths under waiting time constraints. *Algorithmica*, 83(9):2754–2802, 2021. doi: 10.1007/s00453-021-00831-w.

[16] Arnaud Casteigts, Michael Raskin, Malte Renken, and Viktor Zamaraev. Sharp thresholds in random simple temporal graphs. In *2021 IEEE 62nd Annual Symposium on Foundations of Computer Science (FOCS)*, pages 319–326. IEEE, 2022.

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

[18] Markus Chimani, Niklas Troost, and Tilo Wiedera. Approximating multistage matching problems. *Algorithmica*, pages 1–19, 2022.

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

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

[21] 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.

[22] David Coudert, Guillaume Ducoffe, and Alexandru Popa. Fully polynomial fpt algorithms for some classes of bounded clique-width graphs. *ACM Transactions on Algorithms*, 15(3):1–57, 2019.

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

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

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

[26] Jack Edmonds. Paths, trees, and flowers. *Canadian Journal of mathematics*, 17:449–467, 1965.

[27] 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)*, volume 9134 of *LNCS*, pages 444–455. Springer, 2015.

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

[29] 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.

[30] Buddhima Gamlath, Michael Kapralov, Andreas Maggiori, Ola Svensson, and David Wajc. Online matching with general arrivals. In *2019 IEEE 60th Annual Symposium on Foundations of Computer Science (FOCS)*, pages 26–37. IEEE, 2019.

[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] M.R. Garey, D.S. Johnson, and L. Stockmeyer. Some simplified NP-complete problems. *Theoretical Computer Science*, 1(3):237–267, 1976.

[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)*, volume 8572 of *LNCS*, pages 495–507. Springer, 2014.

[35] Archontia C. Giannopoulou, George B. Mertzios, and Rolf Niedermeier. Polynomial fixed-parameter algorithms: A case study for longest path on interval graphs. *Theor. Comput. Sci.*, 689:67–95, 2017.

[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)*, volume 8572 of *LNCS*, 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] Anne-Sophie Himmel, Hendrik Molter, Rolf Niedermeier, and Manuel Sorge. Adapting the Bron-Kerbosch algorithm for enumerating maximal cliques in temporal graphs. *Social Network Analysis and Mining*, 7(1): 35:1–35:16, 2017.

[40] 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.

[41] 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 '19)*, pages 2875–2886, 2019.

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

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

[44] 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.

[45] 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.

[46] Tomohiro Koana, Viatcheslav Korenwein, André Nichterlein, Rolf Niedermeier, and Philipp Zschoche. Data reduction for maximum matching on real-world graphs: Theory and experiments. *ACM J. Exp. Algorithmics*, 26:1.3:1–1.3:30, 2021. ISSN 1084-6654. doi: 10.1145/3439801.

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

[48] 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 of *LIPIcs*, pages 55:1–55:15. Schloss Dagstuhl – Leibniz-Zentrum für Informatik, 2018.

[49] Jure Leskovec, Jon M. Kleinberg, and Christos Faloutsos. Graph evolution: Densification and shrinking diameters. *ACM Transactions on Knowledge Discovery from Data*, 1(1):2, 2007.

[50] Subhrangsu Mandal and Arobinda Gupta. Maximum 0-1 timed matching on temporal graphs. *Discrete Applied Mathematics*, 2022.

[51] 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.

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

[53] 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.

[54] George B. Mertzios, Hendrik Molter, Rolf Niedermeier, Viktor Zamaraev, and Philipp Zschoche. Computing maximum matchings in temporal graphs. In *Proceedings of the 37th International Symposium on Theoretical Aspects of Computer Science (STACS '20)*, volume 154 of *LIPIcs*, pages 27:1–27:14. Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 2020.

[55] 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.

[56] Bojan Mohar. Face covers and the genus problem for apex graphs. *Journal of Combinatorial Theory, Series B*, 82(1):102–117, 2001.

[57] Timothe Picavet, Ngoc-Trung Nguyen, and Binh-Minh Bui-Xuan. Temporal matching on geometric graph data. In *International Conference on Algorithms and Complexity*, pages 394–408. Springer, 2021.

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

[59] 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.

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

[61] 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.

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

[63] 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.

[64] Philipp Zschoche. A faster parameterized algorithm for temporal matching. *Information Processing Letters*, 174:106181, 2022. doi: 10.1016/j.ipl.2021.106181.

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