Computing Maximum Matchings in Temporal Graphs

George B. Mertzios

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

Hendrik Molter

TU Berlin, Faculty IV, Algorithmics and Computational Complexity, Berlin, Germany
h.molter@tu-berlin.de

Rolf Niedermeier

TU Berlin, Faculty IV, Algorithmics and Computational Complexity, Berlin, Germany
rolf.niedermeier@tu-berlin.de

Viktor Zamaraev

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

Philipp Zschoche

TU Berlin, Faculty IV, Algorithmics and Computational Complexity, Berlin, Germany
zschoche@tu-berlin.de

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. The requirement that a vertex cannot be matched twice in any Δ-window models some necessary "recovery" period that needs to pass for an entity (vertex) after being paired up for some activity with another entity. We prove strong computational hardness results for Maximum Temporal Matching, even for elementary cases. To cope with this computational hardness, we mainly focus on fixed-parameter algorithms with respect to natural parameters, as well as on polynomial-time approximation algorithms.

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

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

Digital Object Identifier 10.4230/LIPIcs.STACS.2020.23

Related Version A full version of the paper is available at `https://arxiv.org/abs/1905.05304`.

Funding *George B. Mertzios*: Supported by the EPSRC grant EP/P020372/1.
Hendrik Molter: Supported by the DFG, project MATE (NI369/17).
Viktor Zamaraev: 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.

37th International Symposium on Theoretical Aspects of Computer Science (STACS 2020).
Editors: Christophe Paul and Markus Bläser; Article No. 23; pp. 23:1–23:14
Leibniz International Proceedings in Informatics
LIPICS Schloss Dagstuhl – Leibniz-Zentrum für Informatik, Dagstuhl Publishing, Germany

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* [15, 18]. In a nutshell, a temporal graph is a graph whose topology is subject to discrete changes over time. We adopt a simple and natural model for temporal graphs which originates in the foundational work of Kempe et al. [16]. According to this model, every edge of a static graph is given along with a set of time labels, while the vertex set remains unchanged.

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

An alternative way to view a temporal graph is to see it as an ordered set (according to the discrete time slots) of graph instances (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*, *evolving*, *dynamic*, and *graphs over time*.

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.[1] 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. The concept of Δ-windows has been employed in many different temporal graph problem settings [1, 7, 14, 19]. It is particularly important to understand the complexity of the problem in the case where Δ is a constant, since this models short "recovery" periods.

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

In order to prove our hardness results (see Section 3), we introduce the notion of a *temporal line graph*[2] which is a class of (static) graphs of independent interest and may prove useful in other contexts, too. This notion enables us to reduce Maximum Temporal

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

[2] We remark that a different notion of temporal line graphs was introduced in a survey by Latapy et al. [18], which is somewhat similar to our definition for the case of $\Delta = 1$.

Matching to the problem of computing a large independent set in a static graph (i.e., in the temporal line graph that is defined from the input temporal graph). Moreover, as an intermediate result, we show (see Theorem 11) that the classic problem Independent Set (on static graphs) remains NP-hard on induced subgraphs of *diagonal grid* graphs, thus strengthening an old result of Clark et al. [9] for unit disk graphs.

During the last few decades it has been repeatedly observed that for many variations of Maximum Matching it is straightforward to obtain online (resp. greedy offline approximation) algorithms which achieve a competitive (resp. an approximation) ratio of $\frac{1}{2}$, while great research efforts have been made to increase the ratio to $\frac{1}{2}+\varepsilon$, for *any* constant $\varepsilon > 0$. It is well known that an arbitrary greedy algorithm for matching gives approximation ratio at least $\frac{1}{2}$ [13, 17], while it remains a long-standing open problem to determine how well a randomized greedy algorithm can perform. Aronson et al. [3] 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 [20] proved that MRG actually provides an approximation ratio of $\frac{1}{2}+\frac{1}{256}$. Similarly to the above problems, it is straightforward[3] to approximate Maximum Temporal Matching in polynomial time within a factor of $\frac{1}{2}$. However, we manage to provide a simple approximation algorithm which, for any constant Δ, achieves an approximation ratio $\frac{1}{2}+\varepsilon$ for a constant ε. For $\Delta = 2$ this ratio is $\frac{2}{3}$, while for an arbitrary constant Δ it becomes $\frac{\Delta}{2\Delta-1} = \frac{1}{2}+\frac{1}{2(2\Delta-1)}$ (see Section 4.1).

Given that Maximum Temporal Matching is NP-hard, we show fixed-parameter tractability with respect to the desired solution size parameter. From a parameterized classification standpoint, this improves a result of Baste et al. [6] who needed additionally Δ as a second parameter for fixed-parameter tractability.

Finally, we show fixed-parameter tractability with respect to the combined parameter Δ and size of a maximum matching of the underlying graph (which may be significantly smaller than the cardinality of a maximum temporal matching of the temporal graph). Our algorithmic techniques are essentially based on kernelization and matroid theory (see Section 4).

It is worth mentioning that another temporal variation of Maximum Matching, which is related to ours, was recently proposed by Baste et al. [6]. The main difference is that their model requires edges to exist in at least Δ *consecutive* snapshots in order for them to be eligible for a matching. Thus, their matchings need to consist of time-consecutive edge blocks, which requires some data cleaning on real-word instances in order to perform meaningful experiments [6].

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 positive results are also implied by ours. Baste et al. [6] showed that solving (using their definition) Maximum Temporal Matching is NP-hard for $\Delta \geq 2$. In terms of parameterized complexity, they provided a polynomial-sized kernel for the combined parameter (k, Δ), where k is the size of the desired solution.

We see the concept of multistage (perfect) matchings, introduced by Gupta et al. [12], as the main alternative model for temporal matchings in temporal graphs. This model, which is inspired by reconfiguration or reoptimization problems, is not directly related to ours:

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

roughly speaking, their goal is to find perfect matchings for every snapshot of a temporal graph such that the matchings only slowly change over time. In this setting one mostly encounters computational intractability, which leads to several results on approximation hardness and algorithms [5, 12].

Several details and proofs (marked with $\star$) are omitted due to space constraints.

2 Preliminaries

We use standard mathematical and graph-theoretic notation. In the full version of this paper there is an overview of the most important classical notation and terminology we use.

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

In the remainder of the paper we denote by n and m the number of vertices and edges of the underlying graph G, respectively, unless otherwise stated. We assume that there is no compact representation of the labeling λ, that is, $\mathcal{G}$ is given with an explicit list of labels for every edge, and hence the *size* of a temporal graph $\mathcal{G}$ is $|\mathcal{G}| := |V| + \sum_{t=1}^{T} |E_t| \in O(n + mT)$. Furthermore, in accordance with the literature [23, 24] 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 $|t - t'| \ge \Delta$. 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 $|t - t'| \le \Delta - 1$. A Δ-*temporal matching* M of a temporal graph $\mathcal{G}$ is a set of time-edges of $\mathcal{G}$ which are pairwise Δ-independent. Formally, it is defined as follows.

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

We remark that this definition is similar to the definition of γ-*matchings* by Baste et al. [6].

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 given size k by Temporal Matching.

For some basic observations about our problem settings and more details about the relation between our model and the model of Baste et al. [6] we refer to the full version of this paper.

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

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

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

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

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

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

3 Hardness Results

In this section we show that Maximum Temporal Matching is APX-hard and that Temporal Matching is NP-complete when 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 a so-called *L-reduction* [4] from the APX-complete Maximum Independent Set problem on cubic graphs [2] to Maximum Temporal Matching. Together with the constant-factor approximation algorithm that we present in Section 4.1 this implies APX-completeness for Maximum Temporal Matching. The reduction also implies NP-completeness of Temporal Matching. Formally, we show the following result.

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

We provide the following construction for a reduction from Maximum Independent Set on cubic graphs. It is easy to check that it uses only three time steps and every edge appears in exactly one time step.

► **Construction 1.** Let $G = (V, E)$ be an n-vertex cubic graph. We construct in polynomial time a corresponding temporal graph (H, λ) of lifetime three as follows. First, we find a proper 4-edge coloring $c : E \to \{1, 2, 3, 4\}$ of G. Such a coloring exists by Vizing's theorem and can be found in $O(|E|)$ time [21]. Now the underlying graph $H = (U, F)$ contains two vertices v_0 and v_1 for every vertex v of G, and one vertex w_e for every edge e of G. The set F of the edges of H contains $\{v_0, v_1\}$ for every $v \in V$, and for every edge $e = \{u, v\} \in E$ it contains $\{w_e, u_\alpha\}, \{w_e, v_\alpha\}$, where $c(e) \equiv \alpha \pmod 2$. In temporal graph (H, λ) every edge of the underlying graph appears in exactly one of the three time slots:

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

It is easy to check that the reduction also implies NP-completeness of Temporal Matching. The full proof of Theorem 5 can be found in the full version of this paper.

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

The importance of this observation is due to the following parameterized complexity implication. Parameterizing Temporal Matching by structural graph parameters of the underlying graph that are constant on complete graphs cannot yield fixed-parameter tractability unless P = NP, even if combined with the lifetime T. Note that many structural parameters fall into this category, such as domination number, distance to cluster graph, clique cover number, etc. We discuss how our reduction can be adapted to the model of Baste et al. [6] in the full version of this paper.

3.2 NP-completeness of Temporal Matching with Underlying Paths

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

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

We show this result by a reduction from Independent Set on connected cubic planar graphs, which is known to be NP-complete [11]. More specifically, we show that Independent Set is NP-complete on the temporal line graphs of temporal graphs that have a path as underlying graph. Recall that by Observation 4, solving Independent Set on a temporal line graph is equivalent to solving Temporal Matching on the corresponding temporal graph. We proceed as follows.

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

▶ **Definition 8** (Diagonal Grid Graph). *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 9.** *Let* $\mathcal{G} = (P_n, \lambda)$ *with* $\lambda(e) = [T]$ *for all* $e \in E(P_n)$*, then* $L_2(\mathcal{G}) = \widehat{Z}_{n-1,T}$.

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

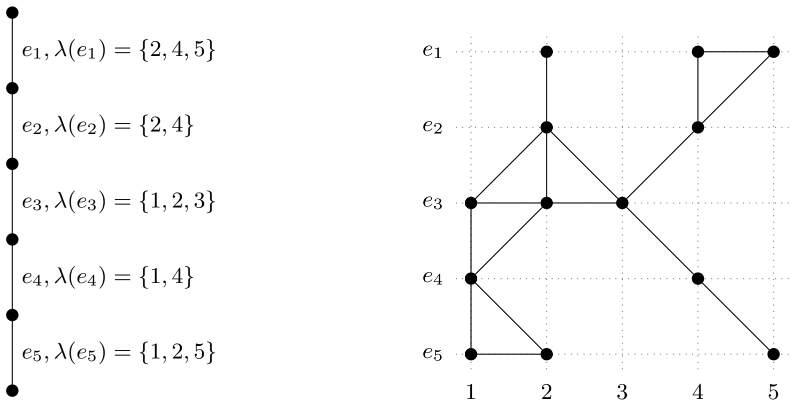
▶ **Corollary 10.** *Let* Z' *be a connected* induced *subgraph of* $\widehat{Z}_{n-1,T}$*. Then there is a* λ *and an* $n' \leq n$ *such that* $Z' = L_2((P_{n'}, \lambda))$.

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

▶ **Theorem 11** (⋆). 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. [9], who showed that Independent Set is NP-complete on unit disk graphs. It is easy to see from Definition 8 that diagonal grid graphs and their induced subgraphs are a (proper) subclass of unit disk graphs.

In the following, we give the main ideas of how we prove Theorem 11. The first building block for the reduction is the fact that we can embed cubic planar graphs into a grid [22]. 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.



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

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

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

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.

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

4 Algorithms

Here, we show one approximation and two exact algorithms for Temporal Matching.

4.1 Approximation of Maximum Temporal Matching

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

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

$M \leftarrow \emptyset$.
foreach Δ-*template* $\mathcal{S}$ **do**
    Compute a Δ-temporal matching $M^{\mathcal{S}}$ with respect to $\mathcal{S}$.
    if $|M^{\mathcal{S}}| > |M|$ **then** $M \leftarrow M^{\mathcal{S}}$.
return M.

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

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

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

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

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

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, on the one hand, there is potential room for improvement. On the other hand, our analysis of the approximation factor of Algorithm 4.1 is tight for $\Delta = 2$. Namely, there exists a temporal graph $\mathcal{G}$ (see Figure 2) such that on the instance $(\mathcal{G}, 2)$ our algorithm (in the worst case) finds a 2-temporal matching of size 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.

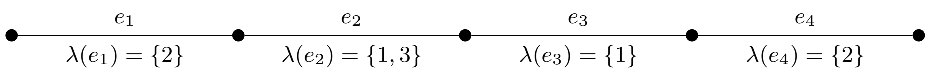


Figure 2 A temporal graph witnessing that the analysis of Algorithm 4.1 is tight for $\Delta = 2$.

4.2 Fixed-parameter tractability for the parameter solution size

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

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

We discuss the proof Theorem 13 in the remainder of this section. Recall that due to Baste et al. [6] it is already known that Temporal Matching is fixed-parameter tractable when parameterized by the solution size k *and* Δ. In comparison to the algorithm of Baste et al. [6] the running time of our algorithm is independent of Δ, hence improving their result from a parameterized classification standpoint.

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

Note that this result is incomparable to Theorem 13. In some sense, we trade off replacing the solution size parameter k with the structurally smaller parameter ν but we do not know how to do this without combining it with Δ. In comparison to the exact algorithm by Baste et al. [6] (who showed fixed-parameter tractability with k and Δ) we replace k by the structurally smaller ν, hence improving their result from a parameterized classification standpoint. Furthermore, we note that Theorem 19 is asymptotically optimal for any fixed Δ since there is no $2^{o(\nu)} \cdot |\mathcal{G}|^{f(\Delta,T)}$ algorithm for Temporal Matching, unless ETH fails (see Theorem 5).

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

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

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

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

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

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

In the proof of Lemma 20 we employ representative families and other tools from matroid theory [8, 10].

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

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

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

5 Conclusion

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

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

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

References

1. E. C. Akrida, G. B. Mertzios, P. G. Spirakis, and V. Zamaraev. Temporal vertex cover with a sliding time window. *J. Comput. Syst. Sci.*, 107:108–123, 2020.
2. P. Alimonti and V. Kann. Some APX-completeness results for cubic graphs. *Theor. Comput. Sci.*, 237(1-2):123–134, 2000.
3. J. Aronson, M. Dyer, A. Frieze, and S. Suen. Randomized greedy matching ii. *Random Struct. Algorithms*, 6(1):55–73, 1995.
4. G. Ausiello, P. Crescenzi, G. Gambosi, V. Kann, A. Marchetti-Spaccamela, and M. Protasi. *Complexity and Approximation: Combinatorial Optimization Problems and Their Approximability Properties.* Springer, 2012.
5. E. Bampis, B. Escoffier, M. Lampis, and V. Th. Paschos. Multistage matchings. In *Proc. of 16th SWAT*, volume 101 of *LIPIcs*, pages 7:1–7:13. Schloss Dagstuhl - LZI, 2018.
6. J. Baste, B. Bui-Xuan, and A. Roux. Temporal matching. *Theor. Comput. Sci.*, 806:184–196, 2020.
7. M. Bentert, A-S. Himmel, H. Molter, M. Morik, R. Niedermeier, and R. Saitenmacher. Listing all maximal k-plexes in temporal graphs. *ACM J. Exp. Algorithmics*, 24(1):1–13, 2019.
8. R. van Bevern, O. Y. Tsidulko, and P. Zschoche. Fixed-parameter algorithms for maximum-profit facility location under matroid constraints. In *Proc. of 11th CIAC*, volume 11485 of *LNCS*, pages 62–74. Springer, 2019.
9. B. N. Clark, C. J. Colbourn, and D. S. Johnson. Unit disk graphs. *Discrete Math.*, 86(1-3):165–177, 1990.
10. F. V. Fomin, D. Lokshtanov, F. Panolan, and S. Saurabh. Efficient computation of representative families with applications in parameterized and exact algorithms. *J. ACM*, 63(4):29:1–29:60, 2016.
11. M. R. Garey and D. S. Johnson. The rectilinear Steiner tree problem is NP-complete. *SIAM J. Appl. Math.*, 32(4):826–834, 1977.
12. A. Gupta, K. Talwar, and U. Wieder. Changing bases: Multistage optimization for matroids and matchings. In *Proc. of 41st ICALP*, volume 8572 of *LNCS*, pages 563–575. Springer, 2014.
13. D. Hausmann and B. Korte. k-greedy algorithms for independence systems. *Oper. Res.*, 22(1):219–228, 1978.
14. A-S. Himmel, H. Molter, R. Niedermeier, and M. Sorge. Adapting the bron-kerbosch algorithm for enumerating maximal cliques in temporal graphs. *Soc. Netw. Anal. Min.*, 7(1):35:1–35:16, 2017.
15. P. Holme and J. Saramäki. Temporal networks. *Physics Reports*, 519(3):97–125, 2012.
16. D. Kempe, J. Kleinberg, and A. Kumar. Connectivity and inference problems for temporal networks. *J. Comput. Syst. Sci.*, 64(4):820–842, 2002.
17. B. Korte and D. Hausmann. An analysis of the greedy heuristic for independence systems. *Discrete Math.*, 2:65–74, 1978.
18. M. Latapy, T. Viard, and C. Magnien. Stream graphs and link streams for the modeling of interactions over time. *Soc. Netw. Anal. Min.*, 8(1):61, 2018.
19. G. B. Mertzios, H. Molter, and V. Zamaraev. Sliding window temporal graph coloring. In *Proc. of 33rd AAAI*, pages 7667–7674. AAAI Press, 2019.
20. M. Poloczek and M. Szegedy. Randomized greedy algorithms for the maximum matching problem with new analysis. In *Proc. of 53rd FOCS*, pages 708–717. IEEE, 2012.
21. A. Schrijver. Bipartite edge coloring in $O(\Delta m)$ time. *SIAM J. Comput.*, 28(3):841–846, 1998.
22. L. G. Valiant. Universality considerations in VLSI circuits. *IEEE Trans. Comput.*, 100(2):135–140, 1981.
23. H. Wu, J. Cheng, Y. Ke, S. Huang, Y. Huang, and H. Wu. Efficient algorithms for temporal path computation. *IEEE Trans. Knowl. Data. Eng.*, 28(11):2927–2942, 2016.
24. P. Zschoche, T. Fluschnik, H. Molter, and R. Niedermeier. The complexity of finding small separators in temporal graphs. *J. Comput. Syst. Sci.*, 107:72–92, 2020.