Partial Temporal Vertex Cover with Bounded Activity Intervals

Riccardo Dondi[1]
Università degli Studi di Bergamo, Italy

Fabrizio Montecchiani
Università degli Studi di Perugia, Italy

Giacomo Ortali
Università degli Studi di Perugia, Italy

Tommaso Piselli
Università degli Studi di Perugia, Italy

Alessandra Tappini
Università degli Studi di Perugia, Italy

Abstract

Different variants of Vertex Cover have recently garnered attention in the context of temporal graphs. One of these variants is motivated by the need to summarize timeline activities in social networks. Here, the activities of individual vertices, representing users, are characterized by time intervals. In this paper, we explore a scenario where the temporal span of each vertex's activity interval is bounded by an integer ℓ, and the objective is to maximize the number of (temporal) edges that are covered. We establish the APX-hardness of this problem and the NP-hardness of the corresponding decision problem, even under the restricted condition where the temporal domain comprises only two timestamps and each edge appears at most once. Subsequently, we delve into the parameterized complexity of the problem, offering two fixed-parameter algorithms parameterized by: *(i)* the number k of temporal edges covered by the solution, and *(ii)* the number h of temporal edges *not* covered by the solution. Finally, we present a polynomial-time approximation algorithm achieving a factor of $\frac{3}{4}$.

2012 ACM Subject Classification Mathematics of computing → Graph algorithms; Theory of computation → Fixed parameter tractability; Theory of computation → Approximation algorithms analysis

Keywords and phrases Temporal Graphs, Temporal Vertex Cover, Parameterized Complexity, Approximation Algorithms

Digital Object Identifier 10.4230/LIPIcs.SAND.2024.11

Funding This work was partially supported by: (*i*) MUR PRIN Proj. 2022TS4Y3N – "EXPAND: scalable algorithms for EXPloratory Analyses of heterogeneous and dynamic Networked Data"; (*ii*) MUR PRIN Proj. 2022ME9Z78 – "NextGRAAL: Next-generation algorithms for constrained GRAph visuALization"; (*iii*) University of Perugia, Fondi di Ricerca di Ateneo, edizione 2021, project "AIDMIX – Artificial Intelligence for Decision making: Methods for Interpretability and eXplainability".

1 Introduction

The temporal graph model is designed to capture the dynamic evolution of interactions over time [14, 12, 17, 13]. A temporal graph can be viewed as a labeled graph, where every edge is endowed with time labels signifying the timestamps where the edge is defined, and thus where the interaction represented by the edge is observed; see Figure 1 for an illustration.

[1] Corresponding author

3rd Symposium on Algorithmic Foundations of Dynamic Networks (SAND 2024).
Editors: Arnaud Casteigts and Fabian Kuhn; Article No. 11; pp. 11:1–11:14
Leibniz International Proceedings in Informatics
LIPICS Schloss Dagstuhl – Leibniz-Zentrum für Informatik, Dagstuhl Publishing, Germany

Numerous foundational problems originally formulated for static graphs have recently been extended to temporal graphs. On static graphs, the Vertex Cover problem asks for a subset of vertices with minimum cardinality, such that it covers all the edges of the input graph, that is, such that for each edge at least one of its endpoints belongs to the subset. Following this line of research, different adaptations of Vertex Cover on temporal graphs have been explored in the literature [1, 11, 19]. Here we focus on the approach introduced in [19], motivated by the need to summarize interaction timelines of users in social networks.

Informally, a *temporal vertex cover* of a temporal graph G is a subset[2] $\mathcal{C}$ of its vertices and an assignment of time intervals to every vertex of $\mathcal{C}$, such that for every edge e of G and for every time label t of e, at least one end-vertex of e is part of $\mathcal{C}$ and the endowed time interval includes t (see Section 2 for a formal definition). In other words, a temporal vertex cover assigns an activity interval to a subset of users, such that for every observed interaction at least one involved user is part of the solution and active. Based on this idea, the objective function of the MinTimelineCover problem is to find a temporal vertex cover of minimum size (i.e., minimizing the sum of the interval lengths).

Recently, a sequence of works investigated the computational complexity of the MinTimelineCover problem, proving that it is NP-hard [19], even in the restricted scenarios when each label is associated with a single edge [4], and when the temporal graph is defined over two timestamps only [7]. In terms of parameterized complexity, MinTimelineCover parameterized by the solution size has first been shown to admit a fixed-parameter algorithm for temporal graphs defined over two timestamps [7] and, subsequently, this restriction has been removed [5].

The complexity of approximating MinTimelineCover has been also studied. A result given in [7] implies that, assuming the Unique Games conjecture, MinTimelineCover cannot be approximated within a constant factor, even for graphs defined on two timestamps only. On the positive side, the problem can be approximated within factor $O(T \log n)$, on a temporal graph with n vertices and T timestamps [6].

In this paper, we introduce and explore a new problem, ℓ-TimelineCover(k) and its optimization version ℓ-MaxTimelineCover, in which we relax the constraint that all the edges have to be covered, bounding instead the length of the vertex activity intervals by an integer $\ell \geq 1$. This last constraint is motivated by the observation that a solution of MinTimelineCover may define long activity intervals for some vertices, while in several applications we observe short time activities of users [19]. Hence, the ℓ-TimelineCover(k) problem asks for the definition of one interval of length at most ℓ for each vertex, so that at least k edges of the temporal graph are covered (or the maximum number of edges are covered for ℓ-MaxTimelineCover); see Figure 1 for an example. From a graph theory point of view, ℓ-TimelineCover(k) can be seen as a temporal variant of Partial Vertex Cover [8, 16]: Given a graph and two positive integers h and p, Partial Vertex Cover asks whether there exists a set of at most h vertices that cover at least p edges of the graph.

Our main contribution can be summarized as follows.

- We prove, in Section 3, that ℓ-TimelineCover(k) is NP-hard and ℓ-MaxTimelineCover is APX-hard, even in the restricted case where the time domain consists of two timestamps (and $\ell = 1$) and each edge appears at most once. Note that if ℓ is equal to the number of timestamps, then the problem admits a trivial solution where each vertex has an interval equal to the number of timestamps and all the edges are covered. Denote by T the

[2] We note that an equivalent definition can be made by replacing $\mathcal{C}$ with the entire vertex set of G, and allowing for vertices with an empty assigned time interval.

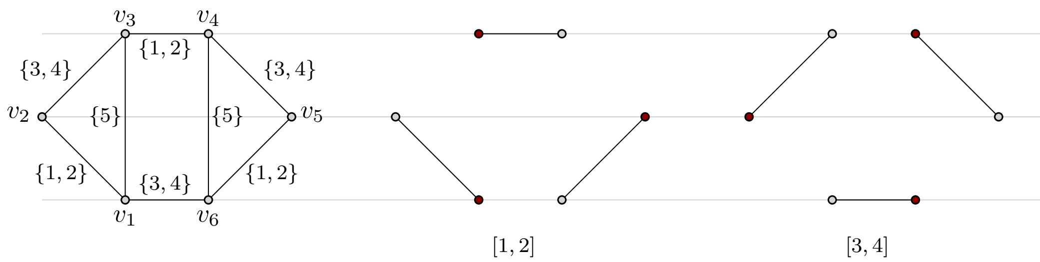

Figure 1 (Left) An example of a temporal graph $\langle G, \lambda \rangle$, where G is a graph and λ is a time-labeling function that maps every edge of G onto a set of timestamps; for example, the edge $\{v_1, v_2\}$ is associated to timestamps $\{1, 2\}$, while the edge $\{v_2, v_3\}$ to timestamps $\{3, 4\}$. (Center and Right) A solution of ℓ-TimelineCover(12) (hence at least 12 temporal edges have to be covered), for $\ell = 2$, defines: interval $[1, 2]$ for v_1, v_3, v_5, thus covering edges at timestamps 1 and 2; interval $[3, 4]$ for v_2, v_4, v_6, thus covering edges at timestamps 3 and 4. Note that the edges defined at timestamp 5 ($\{v_1, v_3\}$, $\{v_4, v_6\}$) are not covered.

number of timestamps over which the temporal graph is defined. These results imply that ℓ-TimelineCover(k) parameterized by $\ell + T$ admits no XP (and hence no FPT) algorithm, unless P=NP.

- Next, in Section 4, we focus on the parameterized complexity of the ℓ-TimelineCover(k) problem and consider two parameters: the number h of temporal edges left uncovered by the solution, and the number k of temporal edges that are covered by the solution. For both parameterizations, we prove that the problem is fixed-parameter tractable.
- Finally, in Section 5, we focus again on the approximability of the ℓ-MaxTimelineCover problem, and we present a polynomial-time approximation algorithm of factor $\frac{3}{4}$.

In Section 2 we give some definitions and we introduce the ℓ-TimelineCover(k) and ℓ-MaxTimelineCover problems. We conclude the paper in Section 6 with open problems that naturally stem from our research. Some of the proofs are deferred to the journal version.

2 Preliminaries

A *temporal graph* is a pair $\langle G, \lambda \rangle$ such that $G = (V, E)$ is a simple (undirected) graph and $\lambda : E \to 2^{\mathbb{N}}$ is a time-labeling function that maps every edge of G onto a set of integers, called *timestamps* in the following (see the example in Figure 1). Up to a relabeling, we can assume that the minimum timestamp over all edges of G is equal to 1, while T denotes the maximum timestamp (and hence it upperbounds the number of timestamps).

We say that an edge $e \in E$ of a temporal graph $\langle G, \lambda \rangle$ is *active* in $t \in \lambda(e)$ and the pair (e, t) is called a *temporal edge*, while E_t is the set of temporal edges active in t.

A *temporal vertex cover* of $\langle G, \lambda \rangle$ is a pair $(\mathcal{C}, \sigma)$, such that: (i) $\mathcal{C} \subseteq V$; (ii) σ maps each vertex v of $\mathcal{C}$ to an interval $[l_v, r_v]$ such that $1 \leq l_v \leq r_v \leq T$; and (iii) for every edge e and for every value $t \in \lambda(e)$, there is a vertex $v \in \mathcal{C}$ such that $t \in [l_v, r_v]$ and $e = \{u, v\}$. An *ℓ-partial temporal vertex cover* of $\langle G, \lambda \rangle$ is a function σ, called *assignment*, such that: (i) σ maps each vertex v of V to an interval $[l_v, r_v]$ such that $1 \leq l_v \leq r_v \leq T$; and (ii) $r_v - l_v + 1 \leq \ell$. A temporal edge (e, t), where $e = \{u, v\}$, is *covered* by σ if either $t \in [l_u, r_u]$ or $t \in [l_v, r_v]$. Note that, in a ℓ-partial temporal vertex cover, we can assume w.l.o.g. each vertex is assigned to an interval of length exactly ℓ.

We are now ready to formalize the definition of ℓ-TimelineCover(k) and of the corresponding optimization version ℓ-MaxTimelineCover.

▶ **Problem 1.** ℓ-TimelineCover(k)
Input: *a temporal graph $\langle G, \lambda \rangle$ and two positive integers ℓ and k.*
Output: *an ℓ-partial temporal vertex cover of $\langle G, \lambda \rangle$ that covers at least k temporal edges.*

▶ **Problem 2.** ℓ-MaxTimelineCover
Input: *a temporal graph $\langle G, \lambda \rangle$ and a positive integer ℓ.*
Output: *an ℓ-partial temporal vertex cover of $\langle G, \lambda \rangle$ that covers the maximum number of temporal edges over all ℓ-partial temporal vertex covers of $\langle G, \lambda \rangle$.*

3 Hardness for Single Labeling

In this section, we prove that ℓ-TimelineCover(k) is NP-hard, even if the input temporal graph $\langle G, \lambda \rangle$ has the following properties: (1) each edge has a single label and (2) $T = 2$. As a corollary of this result, we prove that ℓ-MaxTimelineCover is APX-hard for the same restriction. The result is proven via a reduction from Max 2-3-SAT(h), a variant of Max 2-SAT(h) where each literal appears in at most three clauses. Given a set X of variables and a set of clauses C on X, where each clause consists of exactly two literals and each literal appears in at most three clauses, Max 2-3-SAT(h) asks for a truth assignment to the variables in X that satisfies at least h clauses in C. Note that we assume that each clause in C consists of exactly two literals. Indeed the APX-hardness proof of Max 2-3-SAT(h) in [3, 2] constructs only clauses consisting of exactly two literals.

Construction. Consider an instance $\langle X, C, h \rangle$ of Max 2-3-SAT(h), where $X = \{x_1, \dots, x_q\}$ is a set of variables and $C = \{C_1, \dots, C_z\}$ is a set of clauses, each one defined over two literals. A clause of C is written as $x_{i,A} \vee x_{j,B}$, with $A, B \in \{\mathtt{T}, \mathtt{F}\}$, where $x_{i,\mathtt{T}}$ ($x_{i,\mathtt{F}}$, respectively) represents a positive literal (a negative literal, respectively).

In the following, given $\langle X, C, h \rangle$, we define a corresponding instance $\langle G, \lambda, k, \ell \rangle$ of ℓ-TimelineCover(k), with $k = 24q + h$, $\ell = 1$ and $T = 2$; see Figure 2 for an illustration. Note that, since $T = 2$, the labels belong to interval $[1, 2]$. The set V is defined as follows:

$$V = \{v_{i,\mathtt{T}}, v_{i,\mathtt{F}}, a_{i,1}, a_{i,2}, a_{i,3}, a_{i,4}, b_{i,1}, b_{i,2}, b_{i,3}, b_{i,4} : x_i \in X\}.$$

Next, we define the set E_t of temporal edges:

$$E_t = \{(\{v_{i,\mathtt{T}}, a_{i,p}\}, 1), (\{v_{i,\mathtt{F}}, b_{i,p}\}, 1) : 1 \le p \le 4, 1 \le i \le q\} \cup$$

$$\{(\{a_{i,s}, b_{i,t}\}, 2) : 1 \le i \le q \wedge 1 \le s, t \le 4\} \cup$$

$$\{(\{v_{i,A}, v_{j,B}\}, 2) : 1 \le i, j \le q \wedge A, B \in \{\mathtt{T}, \mathtt{F}\} \wedge (x_{i,A} \vee x_{j,B}) \in C\}.$$

Clearly $\langle G, \lambda \rangle$ is defined over two timestamps. We prove below that each pair of vertices of $\langle G, \lambda \rangle$ is connected by at most one temporal edge.

▶ **Fact 1.** *Let $\langle X, C, h \rangle$ be an instance of* Max 2-3-SAT(h)*, and let $\langle G, \lambda, k, \ell \rangle$ be the corresponding instance of ℓ-*TimelineCover(k)*. For every edge $e \in E$, it holds $|\lambda(e)| = 1$.*

Proof. The edges connecting $v_{i,\mathtt{T}}$ and $a_{i,p}$, $1 \le i \le q$ and $1 \le p \le 4$, are active only at timestamp 1 and the same property holds for the edges connecting $v_{i,\mathtt{F}}$ and $b_{i,p}$, $1 \le i \le q$ and $1 \le p \le 4$. The edges between $a_{i,p}$ and $b_{i,s}$, with $1 \le p \le 4$ and $1 \le s \le 4$, for each $1 \le i \le q$, are active only at timestamp 2. Finally, the edges between $v_{i,A}$ and $v_{j,B}$, $1 \le i \le q$ and $1 \le j \le q$, are active only at timestamp 2. ◀

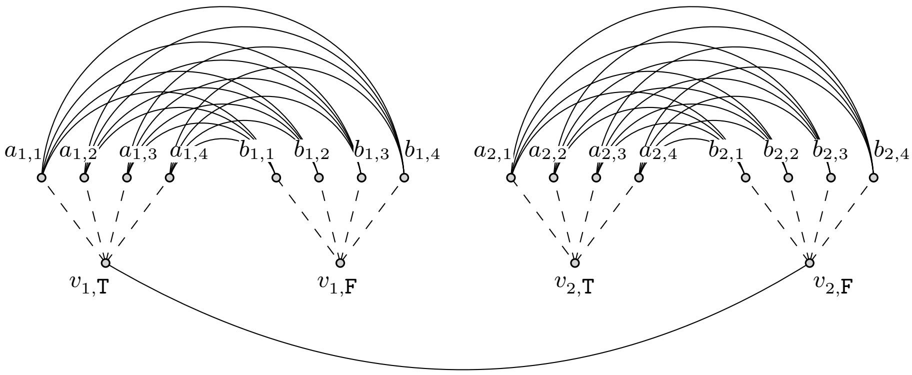

Figure 2 An example of a temporal graph G built by the reduction for clause $(x_{1,\mathtt{T}} \vee x_{2,\mathtt{F}})$. The temporal edges defined at timestamp 1 are dashed, while those defined at timestamp 2 are solid .

Correctness. A solution of ℓ-TimelineCover(k) on $\langle G, \lambda, k, \ell \rangle$ is called *canonical* if every temporal edge $(\{v_{i,\mathtt{T}}, a_{i,p}\}, 1)$, for $1 \le i \le q$ and $1 \le p \le 4$, and every temporal edge $(\{v_{i,\mathtt{F}}, b_{i,p}\}, 1)$, for $1 \le i \le q$ and $1 \le p \le 4$, are covered by such a solution. We start by proving the following property.

▶ **Lemma 1.** *Given an instance $\langle X, C, h \rangle$ of* Max 2-3-SAT(h)*, consider a corresponding instance $\langle G, \lambda, k, \ell \rangle$ of ℓ-*TimelineCover(k)*. Then, starting from a feasible solution of ℓ-*TimelineCover(k) *on $\langle G, \lambda, k, \ell \rangle$, we can compute a feasible canonical solution of ℓ-*TimelineCover(k) *on $\langle G, \lambda, k, \ell \rangle$ that covers at least the same number of temporal edges.*

Proof. Consider an assignment σ to V and assume that there exist two vertices $a_{i,p}, b_{i,s} \in V$, for some i with $1 \le i \le q$, $1 \le p, s \le 4$ that are both assigned to $t = 1$. Notice that the temporal edge $(\{a_{i,p}, b_{i,s}\}, 2)$ is not covered and that each of $a_{i,p}$, $b_{i,s}$ covers at most one temporal edge (active in $t = 1$). Then we can modify the solution σ of ℓ-TimelineCover(k) by assigning one of the two vertices, w.l.o.g. $a_{i,p}$, to $t = 2$ so that the temporal edge $(\{a_{i,p}, b_{i,s}\}, 2)$, is now covered, while $(\{v_{i,\mathtt{T}}, a_{i,p}\}, 1)$ is now possibly not covered. Notice that by iteratively applying this modification we can compute a solution of ℓ-TimelineCover(k) that covers the same number of temporal edges as σ, such that every $a_{i,p}$ or every $b_{i,s}$ is assigned to $t = 2$. Indeed assume this is not the case, then there exist two vertices $a_{i,p}$, $b_{i,s}$ both assigned to $t = 1$, thus by applying the modification described before we can compute a solution with the desired property.

Note that we assume that in σ either every $a_{i,p}$, $1 \le p \le 4$, or every $b_{i,s}$, $1 \le s \le 4$, is assigned to $t = 2$ and that either every $a_{i,p}$, $1 \le p \le 4$, or every $b_{i,s}$, $1 \le s \le 4$, is assigned to $t = 1$. Indeed, assume w.l.o.g. that every $a_{i,p}$ is assigned to $t = 2$, then all the temporal edges defined in timestamp 2 and incident in some $b_{i,s}$ are covered by vertices $a_{i,p}$. Hence we can assume that every $b_{i,s}$ is assigned to 1.

Now, we claim that at most one of $v_{i,\mathtt{T}}$, $v_{i,\mathtt{F}}$, $1 \le i \le q$, is assigned to $t = 2$. Indeed, assume that both $v_{i,\mathtt{T}}$, $v_{i,\mathtt{F}}$ are assigned to $t = 2$ (thus not to $t = 1$). Since either every vertex $a_{i,p}$ or every vertex $b_{i,s}$ is assigned to timestamp 2, it follows that either all temporal edges $(\{v_{i,\mathtt{T}}, a_{i,p}\}, 1)$, with $1 \le i \le q$, and $1 \le p \le 4$, or all the temporal edges $(\{v_{i,\mathtt{F}}, b_{i,p}\}, 1)$, $1 \le i \le q$, and $1 \le p \le 4$ are not covered. Assume w.l.o.g. that every $a_{i,p}$ is assigned to time $t = 2$. Since both $v_{i,\mathtt{T}}$ and $v_{i,\mathtt{F}}$ are assigned to time 2 and each literal in X belongs to at most three clauses, each of $v_{i,\mathtt{T}}$, $v_{i,\mathtt{F}}$ is assigned to $t = 2$ and it covers at most three temporal edges. Then we can compute a solution of ℓ-TimelineCover(k) on $\langle G, \lambda, k \rangle$ by assigning $v_{i,\mathtt{T}}$ to $t = 1$, while each $a_{i,p}$ is assigned to $t = 2$ (or $v_{i,\mathtt{F}}$ assigned to $t = 1$ and each $b_{i,p}$ is assigned to

$t = 2$). The number of covered temporal edges with respect to solution σ is increased at least by one and we have that either $v_{i,\mathtt{T}}$ is assigned to time 1 (if every $a_{i,p}$ is assigned to time 2) or $v_{i,\mathtt{F}}$ is assigned to time 1 (if every $b_{i,p}$ is assigned to time 2). Then every temporal edge $(\{v_{i,\mathtt{T}}, a_{i,p}\}, 1)$, $1 \leq i \leq q$, and $1 \leq p \leq 4$, and every temporal edge $(\{v_{i,\mathtt{F}}, b_{i,p}\}, 1)$, $1 \leq i \leq q$, and $1 \leq p \leq 4$, is covered by the solution. Thus we have computed a canonical solution of ℓ-TimelineCover(k) on $\langle G, \lambda, k, \ell\rangle$ that covers at least the same number of temporal edges as σ, hence concluding the proof. ◀

Now, we can prove the main result of this section.

▶ **Theorem 2.** ℓ-TimelineCover(k) *is NP-hard even on temporal graphs defined on two timestamps and where each edge is assigned a single time label.*

Proof. Note that, by Fact 1, each edge is assigned a single time label and by construction, the temporal graph is defined on two timestamps.

We start by proving the following fact.

▶ **Fact 2.** *Given a solution of* Max 2-3-SAT(h) *on instance* $\langle X, C, h\rangle$ *we can compute in polynomial time a canonical solution of* ℓ-TimelineCover(k) *on the corresponding instance* $\langle G, \lambda, k, \ell\rangle$ *with* $k = 24q + h$ *(hence that covers at least* k *temporal edges).*

Proof. Consider a solution of Max 2-3-SAT(h) on instance $\langle X, C, h\rangle$, we define a solution σ of ℓ-TimelineCover(k) on $\langle G, \lambda, k, \ell\rangle$ as follows. For each variable x_i, $1 \leq i \leq q$, that is set to true, then $v_{i,\mathtt{F}}$ is assigned to $t = 1$, each $a_{i,p}$, with $1 \leq p \leq 4$, is assigned to $t = 1$, $v_{i,\mathtt{T}}$ is assigned to $t = 2$ and each $b_{i,p}$, with $1 \leq p \leq 4$, is assigned to $t = 2$. For each variable x_i, $1 \leq i \leq q$, that is set to false, then $v_{i,\mathtt{T}}$ is assigned to $t = 1$, each $b_{i,p}$, with $1 \leq p \leq 4$, is assigned to $t = 1$, $v_{i,\mathtt{F}}$ is assigned to $t = 2$ and each $a_{i,p}$, with $1 \leq p \leq 4$, is assigned to $t = 2$. By construction the solution σ is canonical, hence the $8q$ temporal edges defined at time $t = 1$ are covered. Each temporal edge $(\{a_{i,p}, b_{i,s}\}, 2)$, with $1 \leq p, s \leq 4$, is covered (we have $16q$ such temporal edges). Finally, by construction, for each satisfied clause, the corresponding temporal edge defined in $t = 2$ is covered (we have h such temporal edges). ◀

For the second direction, we prove the following fact.

▶ **Fact 3.** *Given a solution of* ℓ-TimelineCover(k) *on the instance* $\langle G, \lambda, k, \ell\rangle$ *with* $k = 24q + h$ *(hence that covers at least* k *temporal edges), we can compute in polynomial time a solution* Max 2-3-SAT(h) *on instance* $\langle X, C, h\rangle$ *(hence that satisfies* h *clauses).*

Proof. By Lemma 1 we can consider a canonical solution σ of ℓ-TimelineCover(k) on instance $\langle G, \lambda, k, \ell\rangle$. By construction σ covers the $8q$ temporal edges defined at time $t = 1$. Notice that we can assume that exactly one of $v_{i,\mathtt{T}}$, $v_{i,\mathtt{F}}$ is assigned to $t = 1$. If both $v_{i,\mathtt{T}}$, $v_{i,\mathtt{F}}$ are assigned to $t = 1$, we can define all the vertices $a_{i,p}$ (all the vertices $b_{j,p}$, respectively) assigned to $t = 1$ and assign $v_{i,\mathtt{T}}$ ($v_{i,\mathtt{F}}$, respectively) to $t = 2$. Hence exactly one of $v_{i,\mathtt{T}}$, $v_{i,\mathtt{F}}$ is assigned to $t = 1$, and exactly one of $v_{i,\mathtt{T}}$, $v_{i,\mathtt{F}}$ is assigned to $t = 2$. Moreover, we can assume that for each i with $1 \leq i \leq q$, the temporal edges incident to $a_{i,p}$ and $b_{i,s}$, with $1 \leq p, s \leq 4$, are covered.

Now, construct a truth assignment as follows. For each $1 \leq i \leq q$, if $v_{i,\mathtt{T}}$ is assigned to $t = 2$, then set the corresponding variable x_i to true, if $v_{i,\mathtt{F}}$ is assigned to $t = 2$, then set the corresponding variable x_i to false. By construction if a temporal edge $(\{x_{i,A}, x_{j,B}\}, 2)$ is covered, then the corresponding clause is satisfied, thus we have defined a truth assignment that satisfies at least h clauses, hence a solution of Max 2-3-SAT(h), concluding the proof. ◀

By Fact 2 and by Fact 3 it follows that we have designed a polynomial-time reduction from Max 2-3-SAT(h) to ℓ-TimelineCover(k). By the NP-hardness of Max 2-3-SAT(h) (the decision version) [3], it follows that also ℓ-TimelineCover(k) on temporal graphs defined on two timestamps and where each edge is assigned a single time label is NP-hard. ◀

We note that the reduction described above can be used to prove the APX-hardness of ℓ-MaxTimelineCover, thus implying that ℓ-MaxTimelineCover does not admit a PTAS. Later, in Section 5, we will prove that ℓ-MaxTimelineCover admits an approximation algorithm of factor $\frac{3}{4}$.

▶ **Theorem 3.** *ℓ-MaxTimelineCover is APX-hard.*

Proof. The result follows from the fact that essentially the same reduction described in this section is an L-reduction from the optimization version of Max 2-3-SAT(h) to ℓ-MaxTimelineCover (for details on L-reduction we refer to [20]).

Denote by I an instance of the optimization version of Max 2-3-SAT(h) and by I' the corresponding instance of ℓ-MaxTimelineCover. Let $OPT_S(I)$ be the value of an optimum solution of the optimization version of Max 2-3-SAT(h) on instance I. Let $OPT_M(I')$ be the value of an optimum solution of ℓ-MaxTimelineCover on instance I'.

By Fact 2, we have that

$$OPT_M(I') \leq 24q + OPT_S(I)$$

and, observing that, since there is a truth assignment that satisfies at least $\frac{1}{2}q$ clauses, we have that $OPT_S(I) \geq \frac{1}{2}q$. It follows that

$$OPT_M(I') \leq 24q + OPT_S(I) \leq 48\, OPT_S(I) + OPT_S(I) = 49\, OPT_S(I)$$

Consider the value A' (number of covered temporal edges) of a feasible solution of ℓ-MaxTimelineCover on instance I' and the value A (number of satisfied clauses) of a feasible solution of the optimization version of Max 2-3-SAT(h) on instance I.

By Fact 3, we have that, given a feasible solution of ℓ-MaxTimelineCover of value A' on I' we can compute in polynomial time a feasible solution of the optimization version of Max 2-3-SAT(h) on instance I of value A such that

$$|OPT_S(I) - A| \leq |OPT_M(I') - A'|$$

Thus we have designed an L-reduction from the optimization version of Max 2-3-SAT(h) to ℓ-MaxTimelineCover. Since the optimization version of Max 2-3-SAT(h) is known to be APX-hard [2, 3], the APX-hardness holds also for ℓ-MaxTimelineCover. ◀

4 Fixed Parameter Tractability

In this section, we show that ℓ-TimelineCover(k) is FPT when parameterized by: the number $h = |E| - k$ of temporal edges that may not be covered by the solution (Section 4.1); parameter k and parameter $n + \ell$ (Section 4.2), where n denotes the number of vertices of the input graph.

4.1 Parameter h

The result is obtained by a parameterized reduction to Almost 2-SAT(p), which is known to be FPT when parameterized by p [18, 15], with a similar approach applied for MinTimelineCover in [7]. We recall that, given a 2-CNF formula and a positive integer p, Almost 2-SAT(p) asks whether it is possible to remove at most p clauses so that the resulting formula is satisfiable.

▶ **Theorem 4.** ℓ-TimelineCover(k) *is FPT when parameterized by* h.

Proof. Given an instance $I = \langle G, \lambda \rangle$ of ℓ-TimelineCover(k) and denoted by $h = |E| - k$ the number of temporal edges that are not covered, we define a corresponding instance I' of Almost 2-SAT(p), with $p = h$, as follows. To ease the notation, we shall assume that all timestamps $t \in [1, T]$ are such that $E_t \neq \emptyset$; if this is not the case, the algorithm can be easily modified to avoid any computation in those timestamps t for which $E_t = \emptyset$.

- For each vertex $v \in V$ and for each timestamp $i \in [1, T]$, we create a variable v_i.
- For each vertex $v \in V$, for each timestamp $i \in [1, T]$, and for each $j > i + \ell$, we create $h + 1$ copies of clause $(\overline{v_i} \vee \overline{v_j})$, which is called a *vertex clause*.
- For each edge $\{u, v\} \in E$ such that $\lambda(\{u, v\}) = i$ (with $i \in [1, T]$), we create a clause $(u_i \vee v_i)$, which is called a *temporal edge clause*.

Intuitively, each copy of the vertex clauses models the fact that a vertex is assigned to an interval of length exactly ℓ. More formally, denoted by i the first timestamp of the interval that is assigned to a vertex v, the clause $(\overline{v_i} \vee \overline{v_j})$ ensures that, for each $j > i + \ell$, time j does not belong to the interval assigned to v. Also, a temporal edge clause $(u_i \vee v_i)$ models the fact that a temporal edge $(e = \{u, v\}, i)$ is covered, because u or v is assigned some time interval that includes i.

Note that $p = h$, and that I' can be computed in polynomial time. We now prove that I' is a yes-instance of Almost 2-SAT(p) if and only if I is a yes-instance of ℓ-TimelineCover(k).

Let I be a yes-instance of ℓ-TimelineCover(k). Since I is a yes-instance, we know that $k' \geq k$ temporal edges are covered and hence $h' = |E| - k' \leq h$ temporal edges are not covered. The corresponding Almost 2-SAT(p) instance I' contains h' clauses that are not satisfied, and these clauses are temporal edge clauses. Indeed, since we have $h + 1$ copies for each vertex clause, having one vertex clause that is not satisfied would imply that $h + 1$ clauses of the formula are not satisfied, but we observed $h' \leq h$. By removing the $h' \leq p$ temporal edge clauses that are not satisfied, the formula is satisfied. This implies that I' is a yes-instance of Almost 2-SAT(p).

Now, let I' be a yes-instance of Almost 2-SAT(p). Since I' is a yes-instance, the formula is satisfied by removing at most $p = h$ clauses. Since the formula contains $h + 1 = p + 1$ copies of each vertex clause, the clauses that are not satisfied are temporal edge clauses, while all vertex clauses are satisfied. Each unsatisfied temporal edge clause in I' implies that there is a temporal edge in I that is not covered. It follows that I is a yes-instance of ℓ-TimelineCover(k). ◀

4.2 Parameter k

We shall assume that every vertex is incident to at least one temporal edge, otherwise, we can just remove such a vertex and solve the problem for the obtained subgraph of G. We distinguish two cases.

• **Case 1:** $k \leq \frac{n}{2}$. We first compute a spanning forest $\mathcal{F}$ of the graph G. Assuming w.l.o.g. that G contains no isolated vertex, we note that $\mathcal{F}$ has at least $\frac{n}{2}$ edges. We then root each tree of $\mathcal{F}$, and randomly select k edges from $\mathcal{F}$. Next, we associate each temporal edge $(\{u, v\}, i)$ of the k selected edges to vertex v, where u is the parent of v in the forest. This implies that no two temporal edges are associated to the same vertex. Let (e, i) be one of such temporal edges and let v be the vertex associated to it. We map v to an interval $[i, i + \ell - 1]$. We apply the process mentioned above for each of the k edges. This approach allows us to cover k temporal edges.

• **Case 2:** $k > \frac{n}{2}$. In this case, we use dynamic programming as follows. First, to ease the notation, we shall assume that all timestamps $t \in [1, T]$ are such that $E_t \neq \emptyset$; if this is not the case, the algorithm can be easily modified to avoid any computation in those timestamps t for which $E_t = \emptyset$.

Suppose that we already performed our computations on each timestamp smaller than i, with $i \in [2, T]$, and that we are now analyzing timestamp i. Suppose we have the records $R_1^{i-1}, \dots, R_g^{i-1}$ associated with timestamp $i-1$; the cardinality g of this set of records depends on the parameter k and will be analyzed later. Each record R_j^{i-1}, with $j \in [1, g]$, is composed of the following information:

- A table A_j^{i-1} where we store all the vertices v that we previously assigned to an interval starting in timestamp x, with $i - \ell \leq x \leq i-1$. In this table, we also associate to v the value x of the first timestamp where v is assigned. These vertices are said to be *on-vertices* or simply *on*.
- A set of vertices $\overline{A}_j^{i-1}$ that were previously assigned to an interval starting in timestamp x such that $x + \ell < i - 1$ by the algorithm. Notice that these vertices are not going to cover any temporal edge (e, q) where $q \geq i$. These vertices are said to be *off-vertices* or simply *off*.
- A number s_j^{i-1} of temporal edges that are covered by the vertices in A_j^{i-1} and $\overline{A}_j^{i-1}$. This is the *score* of the record.

Observe that the number of possible different sets of off-vertices is a function of n, which is upperbounded by $2k$ because $k \geq \frac{n}{2}$ by assumption. Also, the score of possible different costs is upperbounded by k by definition, since we only care about covering k edges. Concerning the vertices that are on-vertices in some interval $[x, x+\ell-1]$, observe that we can assume that there are no more than k temporal edges active in this interval, since in this case we can simply assign this interval to every vertex of the graph and the solution is trivial. Also, we assume that we assign a vertex to an interval $[x, x+\ell-1]$ if and only if there exists a temporal edge (e, x) incident to v. If this is not the case, we could simply associate v to the interval that starts with the first timestamp where a temporal edge incident to v is covered, thus covering not fewer temporal edges with v. Since there are less than k temporal edges (e, q) such that $q \in [x, x+\ell-1]$ and by the above assumption, we can assume w.l.o.g. that there are less than k vertices that are on-vertices in each record.

Now, we prove the next lemma:

▶ **Lemma 5.** *For each $i-1 \in [2, T]$, where $R_1^{i-1}, \dots, R_g^{i-1}$ are the records at timestamp $i-1$, we have $g \in O(2^{k \log k})$ and $\max_{i=1}^{g} |R_j^{i-1}| \in O(k \log T)$.*

Proof. Overall, each table in one record contains $O(k)$ items. Concerning the size of a single item, each item of A_j^{i-1}, for each $j \in [1, g]$, represents a value in $O(T)$. Thus, $\max_{i=1}^{g} |R_j^{i-1}| \in O(k \log T)$. Concerning the value of g, we note that the number of possible values x associated with a vertex in A_j^{i-1} is at most k, because there are k possible temporal edges incident to v in an interval of length ℓ starting in x. Consequently, we have $O(2^{k \log k})$ distinct records, i.e., $g \in O(2^{k \log k})$. ◀

`Algorithm description.` In the base case, consider timestamp 1. Let $S_1, \dots, S_{2^n}$ be all the possible subsets of vertices of V and assume that $S_1, \dots, S_g$, with $g \leq 2^n$, are the possible subsets of the vertices that have a temporal edge active in timestamp 1. For each $j \in [1, g]$, we create a record R_j^1 by setting: $A_j^1 = S_j$ and each of such vertices is associated to the value (timestamp) 1; $\overline{A}_j^1 = \emptyset$; the score s_j^1 can be computed in $O(k)$ time.

We now consider the inductive case. Let $R_1^{i-1}, \dots, R_g^{i-1}$ be the set of records computed at timestamp $i-1$. For each R_j^{i-1}, where $j \in [1, g]$, we proceed as follows. We consider the vertex set $V \setminus (A_j^{i-1} \cup \overline{A}_j^{i-1})$ and every possible subset of this set. For each subset S, we construct a new record R_p^i that we will associate with time i. We assume that these and only these are the vertices that are associated with the interval $[i, i+\ell-1]$ for the partial solution described by R_p^i. Let S' be the set of vertices in A_j^{i-1} that are assigned to interval $[i-\ell, i-1]$; note that the information about which vertices of A_j^{i-1} are assigned to interval $[i-\ell, i-1]$ is contained in A_j^{i-1} (these are the vertices of A_j^{i-1} that are associated with timestamp $i-\ell-1$).

- The vertices of R_p^i that are on-vertices (hence those in A_p^i), are going to be the vertices in $(A_j^{i-1} \setminus S') \cup S$; each vertex in S is associated with value i, each vertex in $A_p^i \setminus S$ has the same timestamp it is associated with in A_j^{i-1}. Note that we assume that each vertex u in S has a temporal edge defined in i and covered by u.
- The off-vertices of R_p^i (hence those in $\overline{A}_p^i$) are going to be the vertices in $A_j^{i-1} \cup S'$.
- The score of R_p^i is $s_j^{i-1} + s$, where s is the number of temporal edges covered by vertices of S (that we just associated with an interval $[i, i+\ell-1]$) that are not covered by vertices in A_j^{i-1}. Computing the score s of R_p^i requires $O(k^2)$ time. Indeed, there exist at most k temporal edges in the interval $[i, i+\ell-1]$. We identify the temporal edges defined in $[i, i+\ell-1]$ and not covered by on-vertices of A_j^{i-1} in $O(k^2)$ time (for each temporal edge defined in $[i, i+\ell-1]$ we can check if it is covered by some on-vertex of A_j^{i-1} in $O(k)$ time, since A_j^{i-1} contains less than k on-vertices). Then, for each uncovered temporal edge, we check that it is covered by some vertex in S in $O(k^2)$ time (for each temporal edge we can check if it is covered by some vertex of S in $O(k)$ time, since S contains at most k vertices).

Following from the discussion above, the time complexity needed to perform the above operations does not depend on ℓ, but only on k. Since the number of vertices of the graph is $n \leq 2k$, updating a record requires $O(f(k))$ time, where $f(\cdot)$ is a computable function.

Hence, by Lemma 5 we have the following theorem.

▶ **Theorem 6.** ℓ-TimelineCover(k) *is FPT when parameterized by* k.

It is worth noting that, since in each timestamp a vertex can cover at most $n-1$ edges, it follows that $k \leq \ell \cdot n^2$. Thus, we observe the following.

▶ **Theorem 7.** ℓ-TimelineCover(k) *is FPT when parameterized by* $n + \ell$.

5 A $\frac{3}{4}$-Approximation Algorithm

In this section, we present an approximation algorithm achieving factor $\frac{3}{4}$ based on randomized rounding and inspired by the approximation algorithm given in [9] for Max Sat.

▶ **Theorem 8.** *There is a polynomial-time approximation algorithm for* ℓ-MaxTimelineCover *with factor* $\frac{3}{4}$.

To prove Theorem 8, we first define an ILP formulation for the ℓ-MaxTimelineCover problem (Section 5.1). Next, we describe an algorithm based on randomized rounding an LP relaxation of this formulation (Section 5.2).

5.1 ILP formulation

We present an Integer Linear Programming (ILP) formulation of ℓ-MaxTimelineCover. We make use of the following variables:

- For each temporal edge $(\{v_i, v_j\}, t)$, the variable $e_{i,j,t}$ is 1 if $(\{v_i, v_j\}, t)$ is covered, and it is 0 otherwise.
- For each vertex v_i and for each $t \in [1, T - \ell + 1]$, the variable $A_i(t)$ is 1 if v_i is assigned to interval $[t, t + \ell - 1]$, and it is 0 otherwise.

$$\max \sum_{i,j,t} e_{i,j,t} \tag{1}$$

$$\text{s. t.}$$

$$e_{i,j,t} \leq \sum_{t_1 \in [t-\ell+1,t]} A_i(t_1) + \sum_{t_2 \in [t-\ell+1,t]} A_j(t_2) \qquad \forall(\{v_i, v_j\}, t) \tag{2}$$

$$\sum_t A_i(t) \leq 1 \qquad \forall v_i \in V \tag{3}$$

$$e_{i,j,t} \in \{0, 1\}, \qquad \forall(\{v_i, v_j\}, t) \tag{4}$$

$$A_i(t) \in \{0, 1\} \qquad \forall v_i \in V, \; \forall t \in [1, 2, \ldots, T - \ell] \tag{5}$$

Inequality (2) guarantees that a variable $e_{i,j,t}$ can be set to 1 only if at least one end-vertex is mapped to an interval containing t, while inequality (3) guarantees that each vertex is mapped to at most one interval.

5.2 The Approximation Algorithm

The $\frac{3}{4}$-factor approximation algorithm for the ℓ-MaxTimelineCover problem is presented in Algorithm 1. The algorithm solves in polynomial time an LP relaxation of the ILP formulation described in Section 5.1, where variables $e_{i,j,t} \in [0, 1]$ and $A_i(t) \in [0, 1]$. We denote by $A^*_i(t)$ and $e^*_{i,j,t}$ the values of variables $A_i(t)$ and $e_{i,j,t}$, respectively, of the optimal solution of the LP relaxation.

Starting from a solution of the relaxation, the approximation algorithm defines a solution for ℓ-MaxTimelineCover by assigning each vertex $v_i \in V$ to interval $[t, t+\ell-1]$ with probability $A^*_i(t)$.

Algorithm 1 $\frac{3}{4}$-approximation algorithm for the ℓ-MaxTimelineCover problem.

Solve the LP relaxation of the ILP formulation from Section 5.1, with constraints $e_{i,j,t} \in [0, 1]$ and $A_i(t) \in [0, 1]$
Let $e^*_{i,j,t}$ and $A^*_i(t)$ be the values of a solution to the relaxation of the ILP from Section 5.1
For every vertex v_i, define it active in interval $[t, t + \ell - 1]$ with probability $A^*_i(t)$

Let $\mathbb{E}(\sigma)$ be the expected value of a solution σ returned by Algorithm 1. Denote by $P[e_{i,j,t}]$ the probability that the temporal edge $(\{v_i, v_j\}, t)$ is covered. It holds that

$$\mathbb{E}(\sigma) = \sum_{i,j,t} P[e_{i,j,t}].$$

Consider now $P[e_{i,j,t}]$, it holds that

$$P[e_{i,j,t}] = 1 - P[\overline{e_{i,j,t}}],$$

where $\overline{e_{i,j,t}}$ is the event that the temporal edge $(\{v_i, v_j\}, t)$ is not covered by solution σ. We have that

$$1 - P[\overline{e_{i,j,t}}] = 1 - P[(\{v_i, v_j\}, t) \text{ not cov. by } v_i \wedge (\{v_i, v_j\}, t) \text{ not cov. by } v_j]. \tag{6}$$

Now, we have that

$$P[(\{v_i, v_j\}, t) \text{ not cov. by } v_i \wedge (\{v_i, v_j\}, t) \text{ not cov. by } v_j] = \left(1 - \sum_{t_1 \in [t-\ell+1, t]} A_i^*(t_1)\right)\left(1 - \sum_{t_2 \in [t-\ell+1, t]} A_j^*(t_2)\right) \quad (7)$$

By combining the previous equations we have that

$$1 - P[\overline{e_{i,j,t}}] = 1 - \left(1 - \sum_{t_1 \in [t-\ell+1, t]} A_i^*(t_1)\right)\left(1 - \sum_{t_2 \in [t-\ell+1, t]} A_j^*(t_2)\right). \quad (8)$$

From the arithmetic mean inequality, we have that

$$1 - \left(1 - \sum_{t_1 \in [t-\ell+1, t]} A_i^*(t_1)\right)\left(1 - \sum_{t_2 \in [t-\ell+1, t]} A_j^*(t_2)\right) \geq$$

$$1 - \left(\frac{1 - \sum_{t_1 \in [t-\ell+1, t]} A_i^*(t_1) + 1 - \sum_{t_2 \in [t-\ell+1, t]} A_j^*(t_2)}{2}\right)^2 =$$

$$1 - \left(1 - \frac{(\sum_{t_1 \in [t-\ell+1, t]} A_i^*(t_1) + \sum_{t_2 \in [t-\ell+1, t]} A_j^*(t_2))}{2}\right)^2.$$

Recall that $e^*_{i,j,t}$ is the value of variable $e_{i,j,t}$ returned by the relaxation of the ILP formulation of Section 5.1. By Inequality (2) of this formulation, we have that:

$$\sum_{t_1 \in [t-\ell+1, t]} A_i^*(t_1) + \sum_{t_2 \in [t-\ell+1, t]} A_j^*(t_2) \geq e^*_{i,j,t}. \quad (9)$$

Thus

$$1 - P[\overline{e_{i,j,t}}] \geq$$

$$1 - \left(1 - \frac{(\sum_{t_1 \in [t-\ell+1, t]} A_i^*(t_1) + \sum_{t_2 \in [t-\ell+1, t]} A_j^*(t_2))}{2}\right)^2 \geq 1 - \left(1 - \frac{e^*_{i,j,t}}{2}\right)^2.$$

Hence, $P[e_{i,j,t}]$ can be bounded as follows:

$$P[e_{i,j,t}] = 1 - P[\overline{e_{i,j,t}}] \geq 1 - \left(1 - \frac{e^*_{i,j,t}}{2}\right)^2.$$

The function

$$1 - \left(1 - \frac{e^*_{i,j,t}}{2}\right)^2$$

is a concave function and it has value 0 for $e^*_{i,j,t} = 0$ and value $\frac{3}{4}$ for $e^*_{i,j,t} = 1$. It follows that

$$P[e_{i,j,t}] \geq 1 - \left(1 - \frac{e^*_{i,j,t}}{2}\right)^2 \geq \frac{3}{4} e^*_{i,j,t},$$

thus concluding the proof.

6 Conclusion

In this paper, we introduced and studied the ℓ-TimelineCover(k) problem (and its optimization version ℓ-MaxTimelineCover), a variant of the classical Vertex Cover problem inspired by a recent stream of work on temporal graphs. We have established the NP-hardness of ℓ-TimelineCover(k) and the APX-hardness of ℓ-MaxTimelineCover, under the restricted condition where the temporal domain consists of only two timestamps and each edge appears at most once. We have presented two fixed-parameter algorithms for the following parameters: *(i)* the number k of temporal edges covered by the solution, and *(ii)* the number h of temporal edges *not* covered by the solution. Furthermore, we have contributed a $\frac{3}{4}$-approximation algorithm for ℓ-MaxTimelineCover based on randomized rounding.

There are some interesting research directions to explore. First, the parameterized complexity of the problem can be further investigated, similarly to what has been done for MinTimelineCover in [7]. Second, it would be interesting to improve the approximation factor for ℓ-MaxTimelineCover, possibly considering the semidefinite programming technique applied for Max Sat [10]. A third possible direction involves expanding the definition of vertex activity by permitting a finite number of intervals during which a vertex can be active, as done for MinTimelineCover in [7, 19].

References

1 Eleni C. Akrida, George B. Mertzios, Paul G. Spirakis, and Viktor Zamaraev. Temporal vertex cover with a sliding time window. *J. Comput. Syst. Sci.*, 107:108–123, 2020. doi:10.1016/j.jcss.2019.08.002.

2 Piotr Berman and Marek Karpinski. On some tighter inapproximability results. *Electron. Colloquium Comput. Complex.*, TR98-029, 1998. arXiv:TR98-029.

3 Piotr Berman and Marek Karpinski. On some tighter inapproximability results (extended abstract). In Jirí Wiedermann, Peter van Emde Boas, and Mogens Nielsen, editors, *Automata, Languages and Programming, 26th International Colloquium, ICALP'99, Prague, Czech Republic, July 11-15, 1999, Proceedings*, volume 1644 of *Lecture Notes in Computer Science*, pages 200–209. Springer, 1999. doi:10.1007/3-540-48523-6_17.

4 Riccardo Dondi. Untangling temporal graphs of bounded degree. *Theor. Comput. Sci.*, 969:114040, 2023. doi:10.1016/J.TCS.2023.114040.

5 Riccardo Dondi and Manuel Lafond. An FPT algorithm for temporal graph untangling. In Neeldhara Misra and Magnus Wahlström, editors, *18th International Symposium on Parameterized and Exact Computation, IPEC 2023, September 6-8, 2023, Amsterdam, The Netherlands*, volume 285 of *LIPIcs*, pages 12:1–12:16. Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 2023. doi:10.4230/LIPICS.IPEC.2023.12.

6 Riccardo Dondi and Alexandru Popa. Timeline cover in temporal graphs: Exact and approximation algorithms. In Sun-Yuan Hsieh, Ling-Ju Hung, and Chia-Wei Lee, editors, *Combinatorial Algorithms - 34th International Workshop, IWOCA 2023, Tainan, Taiwan, June 7-10, 2023, Proceedings*, volume 13889 of *Lecture Notes in Computer Science*, pages 173–184. Springer, 2023. doi:10.1007/978-3-031-34347-6_15.

7 Vincent Froese, Pascal Kunz, and Philipp Zschoche. Disentangling the computational complexity of network untangling. In Luc De Raedt, editor, *Proceedings of the Thirty-First International Joint Conference on Artificial Intelligence, IJCAI 2022, Vienna, Austria, 23-29 July 2022*, pages 2037–2043. ijcai.org, 2022. doi:10.24963/ijcai.2022/283.

8 Rajiv Gandhi, Samir Khuller, and Aravind Srinivasan. Approximation algorithms for partial covering problems. *Journal of Algorithms*, 53(1):55–84, 2004.

9 Michel X. Goemans and David P. Williamson. New 3/4-approximation algorithms for the maximum satisfiability problem. *SIAM J. Discret. Math.*, 7(4):656–666, 1994. doi:10.1137/S0895480192243516.

10 Michel X. Goemans and David P. Williamson. Improved approximation algorithms for maximum cut and satisfiability problems using semidefinite programming. *J. ACM*, 42(6):1115–1145, 1995. doi:10.1145/227683.227684.

11 Thekla Hamm, Nina Klobas, George B. Mertzios, and Paul G. Spirakis. The complexity of temporal vertex cover in small-degree graphs. In *Thirty-Sixth AAAI Conference on Artificial Intelligence, AAAI 2022, Thirty-Fourth Conference on Innovative Applications of Artificial Intelligence, IAAI 2022, The Twelveth Symposium on Educational Advances in Artificial Intelligence, EAAI 2022 Virtual Event, February 22 - March 1, 2022*, pages 10193–10201. AAAI Press, 2022.

12 Petter Holme. Modern temporal network theory: a colloquium. *The European Physical Journal B*, 88(9):234, 2015.

13 Petter Holme and Jari Saramäki. A map of approaches to temporal networks. In *Temporal Network Theory*, pages 1–24. Springer, 2019.

14 David Kempe, Jon M. Kleinberg, and Amit Kumar. Connectivity and inference problems for temporal networks. *J. Comput. Syst. Sci.*, 64(4):820–842, 2002. doi:10.1006/jcss.2002.1829.

15 Stefan Kratsch and Magnus Wahlström. Representative sets and irrelevant vertices: New tools for kernelization. *J. ACM*, 67(3):16:1–16:50, 2020. doi:10.1145/3390887.

16 Julián Mestre. A primal-dual approximation algorithm for partial vertex cover: Making educated guesses. *Algorithmica*, 55:227–239, 2009.

17 Othon Michail. An introduction to temporal graphs: An algorithmic perspective. *Internet Math.*, 12(4):239–280, 2016. doi:10.1080/15427951.2016.1177801.

18 Igor Razgon and Barry O'Sullivan. Almost 2-sat is fixed-parameter tractable. *J. Comput. Syst. Sci.*, 75(8):435–450, 2009. doi:10.1016/J.JCSS.2009.04.002.

19 Polina Rozenshtein, Nikolaj Tatti, and Aristides Gionis. The network-untangling problem: from interactions to activity timelines. *Data Min. Knowl. Discov.*, 35(1):213–247, 2021. doi:10.1007/s10618-020-00717-5.

20 David P. Williamson and David B. Shmoys. *The Design of Approximation Algorithms*. Cambridge University Press, 2011. URL: http://www.cambridge.org/de/knowledge/isbn/item5759340/?site_locale=de_DE.